Master of Science Thesis

Interpretable classification of tumours through multiple instance learning and somatic mutations

Author
Stefan C. Dentro
Student number
1313320
Date
13 December 2013
Thesis Committee
Prof. dr. ir. M.J. T. Reinders
Dr. ir. J. de Ridder
Dr. D.M.J. Tax
Dr. ir. M. de Bruijne
Supervisors
Dr. ir. J. de Ridder
Dr. D.M.J. Tax
Dr. D.J. Adams
Programme
Computer Science - Track Bioinformatics
Faculty EEMCS, Delft University of Technology

TU Delft Master of Science Thesis

Interpretable classification of tumours through multiple instance learning and somatic mutations

Stefan C. Dentro

Pattern Recognition & Bioinformatics group, Delft University of Technology, Delft, The Netherlands

Thesis defense: 13 December 2013

Supervisors: Jeroen de Ridder, David M.J. Tax, David J. Adams

ABSTRACT

Motivation: Next generation sequencing is brought into the clinic. Screening of disease associated genes will aid the diagnosis of disorders with a genetic component. The diagnosis of cancer is of particular interest due to its variety and prevalence. The obtained mutations provide clues about underlying biological properties that could be used for classification. Classification of a tumour as a particular type of cancer is an important step towards treatment. But currently no method exists that can directly classify cancers using next generation sequencing derived mutations.

Results: In this paper we present a classification method in which tumours are modelled as bags of annotated somatic mutations. Our method uses a machine learning approach to identify and select the relevant mutations and subsequently train a classifier for each type of cancer. The selected mutations result in an interpretable model that sheds light onto which biological properties are important to separate one cancer type from the others. We compare the proposed method to two other approaches. First a gene based approach in which the mutations are reduced to a mutation count per gene. Second a distance approach that uses all the available mutations, but returns a model that is hard to interpret. We show that the proposed method performs equally well when compared to the first approach. Our method achieves performance close to the second approach, while it yields a model that allows for biological interpretation.

Contact: s.c.dentro-1@student.tudelft.nl

1 INTRODUCTION

Next generation sequencing (NGS) technology has been around for over half a decade. It has matured and gained a wide adoption in many research fields. NGS is now introduced into the clinic [Need *et al.*, 2012, Yang *et al.*, 2013], where it can aid the diagnosis of disorders with a genetic component [Flintoft, 2013].

The diagnosis of cancer is of particular interest due to its wide prevalence and complicated biological background. An important step towards treatment is determining the type of cancer (classification). Classification is traditionally aided by morphological appearances such as size and the number of tumours [Pawlik *et al.*, 2005]. But this is not a straightforward process. Tumours with a similar histological appearance are difficult to distinguish from one-another, yet can respond differently to hormone- and chemotherapy [Berman, 2004]. Furthermore, between 2-5% of diagnosed cancer patients with metastasis remain without an established primary site [Petrakis *et al.*, 2013]. An automated classification approach could assist with these difficulties by utilising mutations found through NGS.

Tumour cells accumulate somatic mutations that allow them to chronically proliferate [Greenman *et al.*, 2007, Hanahan and Weinberg, 2011]. These mutations are the result of defects in DNA replication and repair mechanisms, exposure to mutagens and enzymatic DNA modifications [Alexandrov *et al.*, 2013b]. Environmental factors also play a varying role in the development of some types of cancer. Tobacco smoking is linked to lung cancer for example, while ultraviolet light exposure is associated with skin cancers [Pfeifer, 2010]. But the impact of environmental factors is unknown for most types of cancer.

Recently large sequencing studies such as ICGC, PCGP and TCGA have been designed to gain more insight into the mutation spectra of various types of cancer [Consortium *et al.*, 2010, Downing *et al.*, 2012, The Cancer Genome Atlas Research Network *et al.*, 2013]. These studies aim to find common characteristics between tumours of the same type and subsequently uncover some of the underlying biology. But these studies rarely go beyond basic properties such as the type of mutation, the gene in which it occurred and the pathways the gene participates in. Additional context around a mutation could be used during classification.

The growing body of cancer data could be used to model the biological processes that introduce mutations. Such a model could be used for prediction and classification purposes. Alexandrov *et al.* used the mutations recently to construct signatures [Alexandrov *et al.*, 2013a,b]. They defined 96 different mutation types by extending the six base substitutions (A>C, A>G, etc.) to its immediate $3'$ and $5'$ sequence context. A signature is a set of estimated probabilities that mutation types occur together. These signatures correspond to different mutational processes that introduce patterns of mutations. But the obtained mutation signatures are not unique per cancer type and cannot be used for classification purposes.

We propose a multiple instance learning (MIL) [Dietterich *et al.*, 1997] based approach that takes into account our incomplete understanding of the underlying cancer biology. Each tumour is modelled as a bag of mutations and each bag is assigned the cancer type as a label. The main assumption behind MIL is that at least one observation per bag is descriptive for the bag label, but not necessarily all of them. Our method therefore assumes that not all somatic mutations are relevant.

This is important when working with NGS derived somatic mutations from tumours. The concordance rate between somatic mutation callers is low [Pabinger *et al.*, 2013], which means that not all obtained mutations have to be real. Not all real mutations are equal either. Driver mutations are thought to provide tumour cells with a selective advantage over other cells, while passenger mutations have a neutral effect. Recently publications focus on

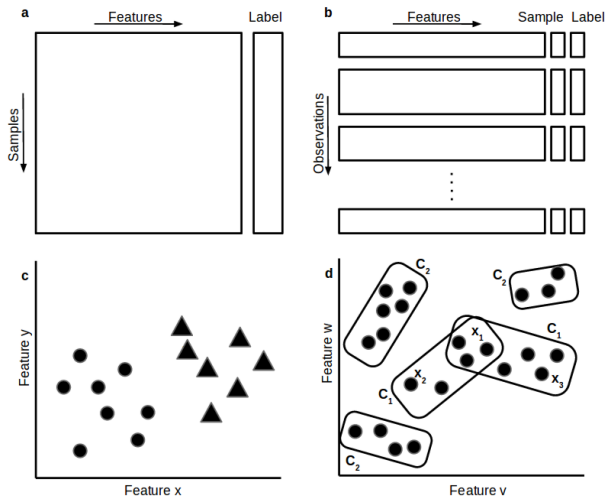


Fig. 1. (**a**) A regular classification problem consists of a number of samples. Each sample is described by a single feature vector, which results in a matrix. Each sample is assigned a label. (**b**) A MIL problem differs because a sample can have multiple feature vectors. The vectors (termed observations) are therefore grouped per sample in a bag. Each bag is assigned a label, while the observations remain unlabelled. (**c**) A regular classification problem in a 2-dimensional feature space. It contains a number of samples of two different classes. (**d**) An example MIL problem with five bags and a number of observations in a 2-dimensional feature space. The bags are assigned either the label C_1 or C_2. Observations like $\boldsymbol{x}_1$ reside in a shared area between C_1 labelled bags and are therefore of interest. Observations like $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ are in C_1 labelled bags, but could be removed to obtain better classification performance.

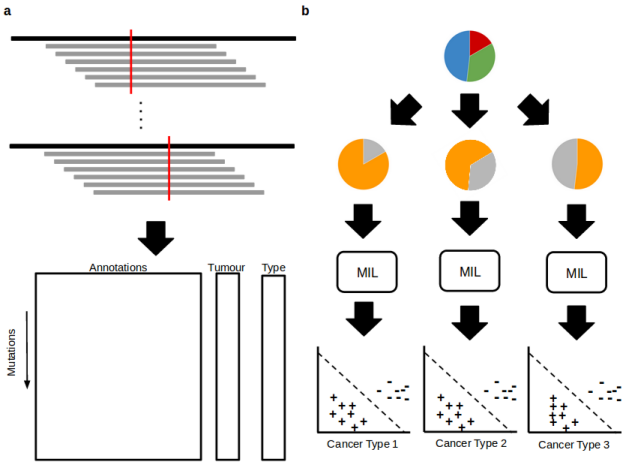


Fig. 2. (**a**) Mutations obtained from sequencing data are annotated with information that provides more context. For each mutation it is known in which tumour it occurred and what the type of cancer of the tumour was. A MIL dataset structure as is depicted in figure 1b arises. (**b**) A one-versus-all setup is used to train a classifier for each type of cancer. First a copy of the input dataset is made for each cancer type. Tumours are relabelled such that one cancer type is marked as positive and the other types are marked as negative. A classifier is subsequently trained on each relabelled dataset.

distinguishing driver mutations ([Youn and Simon, 2011, Hua *et al.*, 2013] for example). But these methods often discard synonymous mutations, eventhough deleterious synonymous mutations are known to exist [Buske *et al.*, 2013].

Simply restricting the input to mutations in known cancer genes is not of much help either. The large cancer sequencing projects show that known cancer genes are mutated in less than half of the tumours that are investigated, irrespective of the possible effect that a mutation has [Yeang *et al.*, 2008]. There are currently nearly 500 known cancer genes [Yates and Campbell, 2012], but some of these may have been found as a result of large sample sizes [Lawrence *et al.*, 2013]. Finding cancer driver genes remains difficult as well, with a small overlap between callers [Tamborero *et al.*, 2013].

In stead of filtering mutations beforehand we have developed a method that performs mutation selection. Two selection stages aim to find those mutations that separate one cancer type from all the others (see methods). An additional advantage of the selection is that the trained classifier can explain how it performed the separation. Annotations of the selected mutations can be used to investigate the differences between cancers. These differences have the potential to uncover some of the underlying cancer biology that could not have been found with other methods because of assumptions made beforehand.

There is an extensive body of published work about cancer classification using micro-arrays. These methods utilise differential gene expression and show that high throughput experiment data allows for separation of different types of cancer. Ramaswamy *et al.* for example could predict the type of cancer across 14 different tumour types with 78% accuracy. While Veer *et al.* and van de Vijver *et al.* found that 70 genes are predictive for breast cancer treatment prognosis. But these methods utilise differential gene expression only and are not easily extended with additional annotations. Our method can be used to perform the same experiments with differential gene expression as input, while not posing any restrictions on annotations.

The somatic mutations can have any number of encoded annotations. The annotations provide additional context that is required to evaluate the importance of a mutation. This is important because similar mutations are rarely found in tumours of the same cancer type [Hofree *et al.*, 2013]. Annotations could for example consist of variant effect predictors like SIFT, MutationTaster and PolyPhen [Ng and Henikoff, 2003, Schwarz *et al.*, 2010, Khurana *et al.*, 2013]. They could also contain whether a mutation falls within a gene or whether it causes an amino-acid substitution. But the context is not restricted to just the genome. Epigenetic properties such as whether a mutation falls within a transcription factor binding site or in a hypo- or hyper-methylated region could add valuable information.

2 APPROACH

In a regular classification problem a sample is assigned a label and is represented by a vector of N features. A dataset with a number of samples is therefore represented by a matrix in which a row is a sample and a column a feature (figure 1a). Each sample becomes a point in an N-dimensional space. Figure 1c contains an example with two different labels (circles and triangles) and $N = 2$. A classifier aims to find a hyperplane that separates samples with different labels.

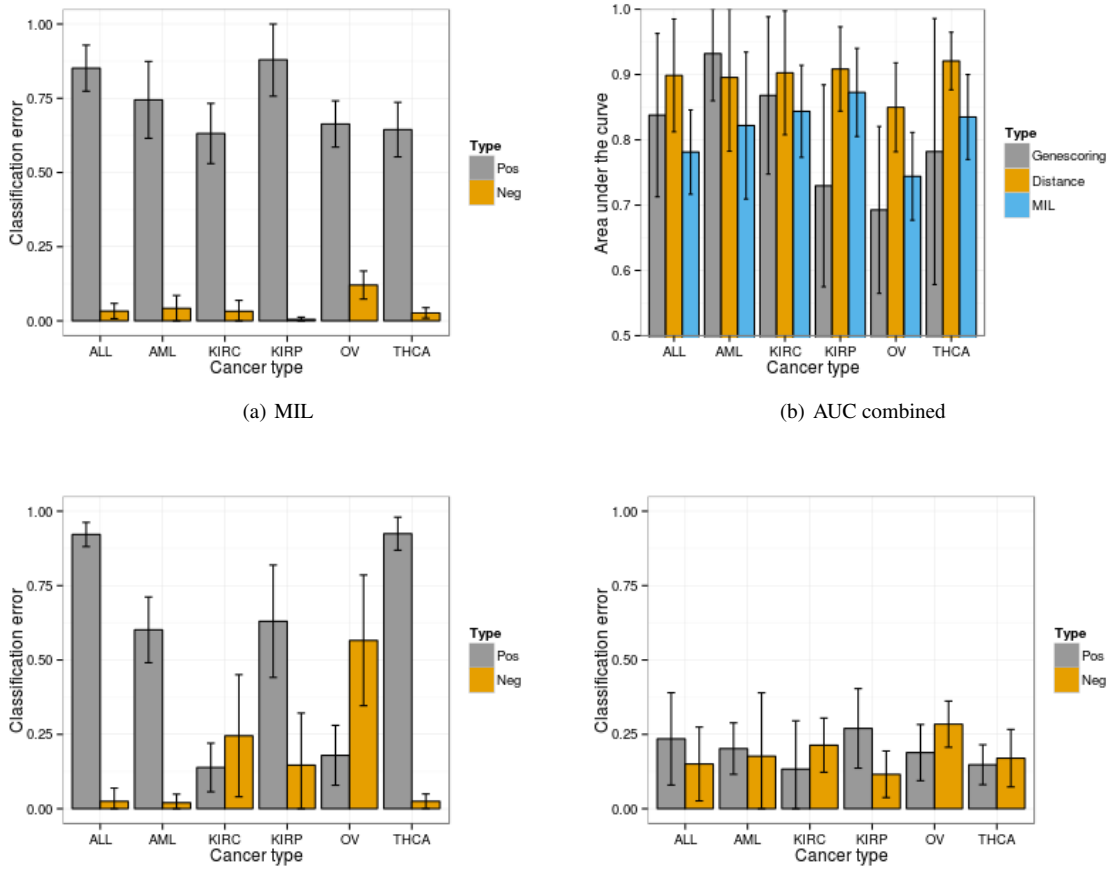
But somatic mutations in tumours provide a more complicated problem. A sample is not represented by a single vector of N features, it can have multiple (one for each mutation), which we term observations. We group observations from a single sample together in a bag and assign a label to each bag (figure 1b). Figure 1d shows an example that contains five bags and two different labels (C_1 and C_2). Each bag contains a number of observations that are described by N features ($N = 2$ in this example).

These N features are obtained through annotations which add context around a mutation. The basics of a mutation are its chromosome location and the inserted and reference base. We add information about the genomic region where the mutation occurred (exonic, intronic, intergenic, UTR, etc.). There is an annotation that describes the effect on an exon when a mutation falls within such a region and we record whether it causes an amino-acid substitution. Beyond this basic information there are a number of variant effect predictors that estimate the impact of the mutation (see methods).

Given the complexity of the underlying biology we assume that not all observations are descriptive for its bag label, which is the type of tumour in which the mutation occurred. Consider again the example in figure 1d. Observation $\boldsymbol{x}_1$ occurs in an area shared by both samples with label C_1. Such a mutation is considered to be important. Meanwhile observations $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ occur in an area occupied by a single sample with label C_1. $\boldsymbol{x}_2$ is also in close proximity to samples with label C_2. By removing observation $\boldsymbol{x}_2$ the feature space declutters. It therefore becomes easier for a classifier to find a decision boundary between samples with different labels.

The MIL classifier described in this work selects those observations that are deemed important to separate samples with different labels from each other. In other words, it selects mutations that are useful when separating two cancer types from each other. The properties of these mutations can be used to explain what is important to separate two types of cancer.

However our dataset consists of more than two types. We use a one-versus-all scheme in order to obtain a classifier per type (figure 2b). This classifier uses those mutations that are required to separate a single type of cancer from all others. First we create a dataset for



(a) MIL

(b) AUC combined

(c) Gene scoring

(d) Distance

Fig. 3. Classification error obtained for (**a**) the MIL experiment when selecting 20% of the observations through a t-test and a penalty term biased by the class size and (**b**) a comparison of the AUCs for each of the three experiment families. Classification error obtained for (**c**) the gene scoring experiments through a nearest mean classifier and (**d**) the distance experiments using a distance measure where the minimum difference between the features of each pair of mutations is taken and subsequently summed per tumour. Here an SVM classifier is used.

each cancer type T. Second tumours with label T are relabelled as *positive* and those without T are relabelled as *negative*. Third we train a classifier on each of these datasets to obtain a classifier per cancer type. Finally, the features of the selected mutations can be used to describe which properties are important to separate cancer type T from the others.

The MIL method is compared to two other approaches. First a simple method where we map each mutation onto the gene in which it occurred and count the number of mutations per gene. The dataset now consists of tumours (rows) and genes (columns). A result of the mapping is that the information contained in the annotations of individual mutations is lost. This simple method therefore assumes that there is no additional information in the individual mutations and the gene name is enough to separate types of cancer from each other.

The second method uses all available mutations. Each tumour is represented as a bag of points in an N-dimensional space, one point for each individual mutation in that tumour. Next we summarise the mutations in each tumour and calculate a distance between each pair of tumours. A regular classification problem is thus created where each tumour (rows) is represented by the distances to all other tumours (columns). Here we assume that all mutations are descriptive for the type of cancer and we sacrifice the interpretability by summarising the mutations per tumour.

Using a one-versus-all scheme means that each observation, irrespective of the label of its bag, can be used to support both the positive and the negative class. In order to make interpretation more straightforward we have attempted to restrict the contribution of mutations to the positive class only. We show in the results section below that this added constraint has a negative impact on the performance of our method.

Finally, we perform feature optimisation through feature selection and a principal component analysis (PCA).

3 RESULTS

3.1 Multiple Instance Learning

We ran the MIL method on a dataset that consists of six types of cancer: ALL, AML, kidney clear cell carcinoma (KIRC), kidney papillary cell carcinoma (KIRP), ovarian cancer (OV) and thyroid carcinoma (THCA) that were previously published by Alexandrov *et al.*. Here we report the MIL results obtained when 20% of the observations are selected through a t-test ranking (see methods). The gene scoring results were obtained through a nearest mean classifier. For the distance experiments we calculated the distance between a pair of tumours by summing the pairwise distances between all pairs of mutations in the pair of bags. A support vector machine was used to create a classifier.

Each one-versus-all dataset consists of samples labelled as positive or negative. A classification error is determined through 10-fold cross validation. In this experiment we bias the classifier to take into account the different sizes of the positive and negative class. We also set the operating point of the classifier to an equal precision and recall for both classes on the training set.

Figure 3(a) shows the mean classification errors over 10 folds on both classes for each of the six cancer types. There is a clear difference with the error rate on the positive class always larger than 0.50 and on the negative class always lower than 0.15. This means that even after setting the operating point to an equal error rate on the training set it doesn't generalise well to the test set.

When we compare the area under the curve (AUC) with the gene scoring experiments (figure 3b) we see that MIL performs better on KIRP, OV and THCA, but worse on ALL, AML and KIRC. The standard deviation is also lower, except for AML. The distance method performs best on all of the classes, but the difference with MIL is less than two standard deviations.

The classification errors tell a slightly different story. The gene scoring figure (3c) shows a similar pattern with a high classification error on the positive class for ALL, AML, KIRP and THCA and a much lower error on the negative. On OV the effect is reversed, while the method appears to work well on KIRC. The classifiers obtained through the distance method outperform MIL on all of the positive classes. MIL on the other hand outperforms distance on all of the negative classes.

These results paint an interesting picture. The performance of MIL is more stable across the cancer types than gene scoring. Gene scoring performance changes considerably between cancer types and between cross validation splits. This is a sign that MIL creates a more general summary of each class. The distance experiments on the other hand show a better performance across all cancer types. We conclude that through the MIL method we obtain a reasonable performance and gain the advantage of an open and interpretable model. Although the classification errors on the positive class still leave something to be desired.

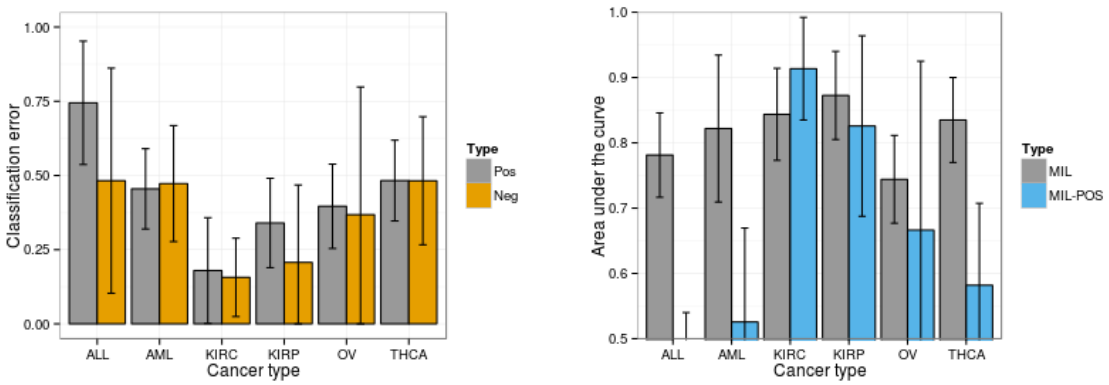
3.2 Contribution to the positive class

The MIL method was extended to restrict the contribution of observations to the positive class only. The advantage is that observations are now selected to characterise the type of cancer for which the classifier is trained. This means that the properties of the selected observations describe what makes this cancer type different from all others. A classifier therefore yields a model that is easier to interpret, while it reduces the memory footprint required to obtain a decision boundary (see methods). The MIL method with the added constraint was run on the six class dataset, where 20% of the mutations were selected through a t-test.

But the added constraint results in a clear drop in performance (figure 4b). The AUC drops in all cases, except KIRC. Most dramatically ALL and AML have an AUC around 0.50. The classification errors on the positive classes go down across all cancer types. But the negative errors go up and the standard deviations increase. It seems that the trained classifiers do not generalise very well, leading to unstable results.

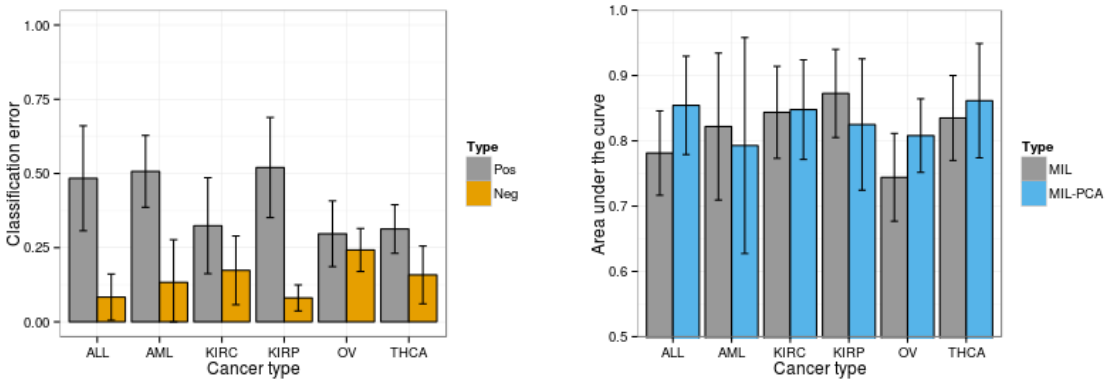
We conclude from this experiment that the MIL method does not just favour the negative class, it seems the negative class is much better defined in general. MIL cannot characterise the positive class for the most part, even when it is strictly forced to do so. One possible reason is that the selection steps select mutations that are not descriptive. But an experiment where mutations are selected at random consistently yielded lower AUCs (see figure S4).

Another possible reason is that the currently used features are not useful. The classifier therefore has difficulties separating the positive class from the negative class. We therefore aim to optimise the contribution of the current features in the next section. We have also experimented with adding new simple features in supplementary document 1.



(a) Classification error MIL-POS (b) AUC

Fig. 4. (**a**) The classification errors obtained for both the positive and negative class when observations are restricted to a contribution to the positive class only. (**b**) A comparison of the AUC between the regular MIL method and its positive contribution variant. The additional restriction has a big influence on the performance of the classifier.



(a) Classification error MIL-PCA (b) AUC

Fig. 5. (**a**) The classification error on the positive class is greatly reduced after feature optimisation, although the errors obtained on the negative class increase. (**b**) The AUC improves for four classes, but the standard deviation between the cross validation splits goes up.

3.3 Feature optimisation

We have divided the features into five groups in order to investigate their contribution to the classifier. Group (1) contains the chromosome and location, (2) the observed and reference base features, (3) the genomic function, (4) exon effect and (5) all variant effect predictors. Group one was removed directly because it allows mutations with a similar location but on different chromosomes to end up in close proximity in the feature space. Such a property does not have any biological meaning.

Four experiments were devised. In each experiment we removed one group and reduced the dimensionality of the remaining data. A PCA was performed such that the remaining features contained roughly 98% of the variance of the original. Figure 5b shows the performance when group five (variant effect predictors) is removed. The MIL method was used with selection of 20% of the mutations through a t-test. The AUC goes up in four cases, while it goes down in two. There also appears to be a larger standard deviation obtained on the cross validation splits.

But the classification error on the positive class drops considerably. Removal of the variant effect predictor group reduces the classification error from around 0.75 to below 0.50. On the other hand, the classification error on the negative class goes up and standard deviations increase. Removal of the other three groups did not yield an increase in performance. The results are available in supplementary document 1.

We conclude from these experiments that the variant effect predictors add little value with respect to classifying the positive class. An interesting observation is that features can influence one of the two classes more heavily than the other. Features could therefore be considered with respect to the positive class only, although that would make interpretation even more difficult without the positive contribution restriction described earlier.

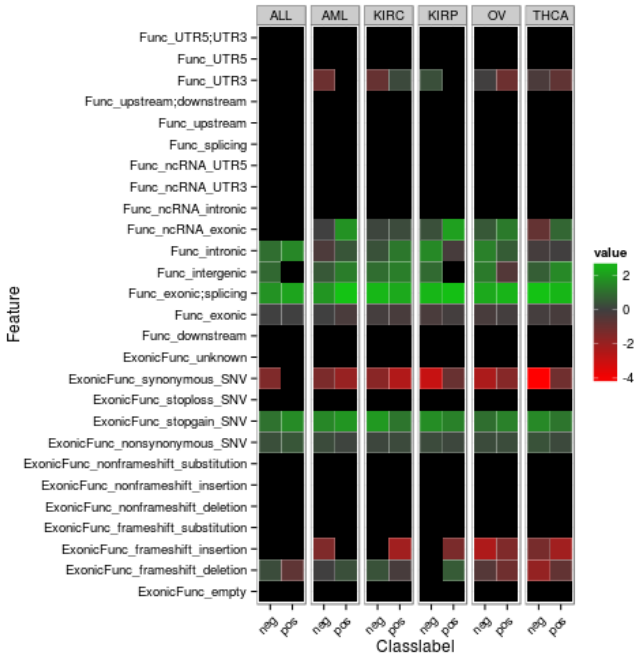


Fig. 6. Heatmap showing properties of the selected mutations for two annotations that pertain to the genomic region in which the mutation occurs. The scale goes from green (more often selected than expected by chance) to red (less often than expected). It contains the properties of those mutations that are selected five times or more during 10-fold cross validation. Overall exonic splice site mutations are more often selected than one would expect by chance. Some properties are deemed unimportant, most notably UTR, non-exonic ncRNA, intronic splice site mutations and mutations up- or downstream from a gene.

Type	In CGC	Not in CGC	Ratio
ALL	11	130	0.09
AML	16	149	0.11
KIRC	15	239	0.06
KIRP	9	139	0.07
OV	23	344	0.07
THCA	19	236	0.08

Table 1. The number of genes for which mutations are selected five times or more during 10-fold cross validation. Those known in the Cancer Gene Census are genes for which a role in the development of cancer is confirmed. Of interest is the low ratio of CGC to non-CGC genes. This could be an indication that differences between cancers reside in genes that were previously not considered. But it seems unlikely that so many cancer genes are undiscovered.

3.4 Biological interpretation

The MIL classifier selects mutations that are deemed important for classification. We can now investigate the properties of these mutations for each of the obtained classifiers. Selection of mutations with a particular property more often than one would expect by chance could be a sign of the biological importance of this property. We use a one-versus-all setup, which creates a classifier for each class in the dataset. It is therefore possible to interpret with respect to each cancer type.

Figure 6 shows the genomic properties of those mutations that were selected at least five times during 10-fold cross validation using the MIL classifier. The figure contains six pillars, one for each cancer type in the dataset. Each pillar consists of two columns, one showing the properties of the selected mutations that contribute to classifying the cancer type (termed positive), the other shows the properties of selected mutations that contribute to the other cancer types (termed negative). Green represents mutations with this property are selected more often than expected by chance, red represents less. Black means no mutation with that feature is selected.

In the ideal case there is a clear discrepancy in a pillar between the positive and negative columns. Such a clear difference is for example visible in for the *Func_intergenic* feature in both ALL and KIRP pillars. The MIL classifier considers intergenic mutations unimportant when creating a classifier for ALL and KIRP, but not as much for the other classes. The *ExonicFunc_ncRNA_exonic* feature seems to play a role in the creation of all classifiers, except for ALL. But what these observations say about the underlying biological properties remains unclear.

We also investigated whether the genes in which selected mutations have occurred are known cancer genes. The ratio of cancer versus non-cancer genes within the selection is below 0.10 for most cancer types (table 1). Furthermore, a large portion of the known cancer genes for which mutations are selected are not associated with the cancer type that the classifier was trained for (see the black squares in figure 7). The list of genes also contains commonly mutated cancer genes TP53 and RB1 that have been associated with a number of cancer types. It is possible that the differences between cancers occur in genes that are not discovered as cancer genes, but it seems unlikely that there would be so many.

4 DISCUSSION

In this paper we report on a method to classify cancers based on somatic mutations. The method is capable of identifying and

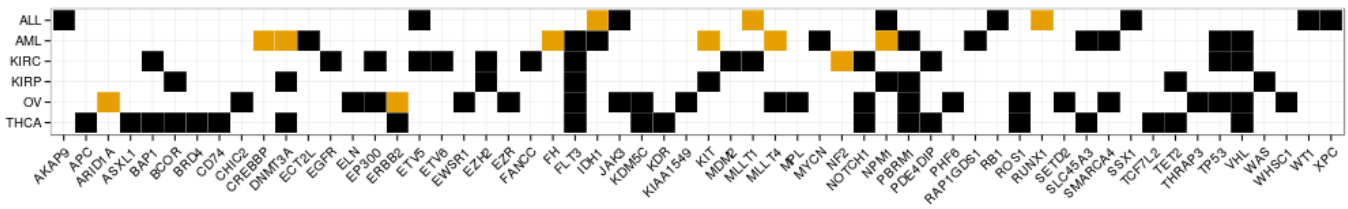


Fig. 7. Overview of known cancer genes in which mutations are selected five times or more during 10-fold cross validation. Each row shows the selection during training of the MIL classifier for the given cancer type. A field is orange when the gene is known to play a role in the disorder for which it was selected, while black represents the case where the gene is not known to play a role for that type of cancer. All genes listed are known in the Cancer Gene Census [Futreal *et al.*, 2004].

selecting mutations that are significant for the classification of a type of cancer. These mutations are subsequently used to train a classifier. Each mutation is annotated with a number of features. Through the selection process it becomes possible to interpret a classifier by investigating the properties of the selected mutations. The properties show how one type of cancer is separated from all others, possibly revealing some of the underlying cancer biology.

We show that the proposed method yields a comparable performance, that is more stable across the six types of cancer in our data set. A distance based approach outperforms the MIL method. But the small decrease in performance comes with the advantage that MIL produces a biologically interpretable model. Although it is currently unknown to what extend the classifiers are capable of elucidating biological properties that play a role in the development of cancer.

The main issue of the MILES classifier on which the proposed method is based is the amount of memory required to during training. We have aimed to reduce the memory footprint by selecting a fraction of the mutations through a t-test or ratio ranking. The ranking based selection could prove inaccurate when more than the requested fraction of mutations has a very similar value. In that case the method does not select all mutations of interest because the fraction parameter is based on memory requirements and not on the data itself. An ideal method should not need this additional selection step and should rely on its internal classifier only.

We have proposed a possible solution to this problem by adding a restriction on the weights that are estimated. The weights are estimated in such a way that each mutation can only support the positive class. Because of this extra restriction the classifier would have to estimate a single coefficient per observation only. The additional advantage is that it would make biological interpretation more straightforward.

But we have been unable to produce reliable results with this altered MIL problem. The trained classifier consistently doesn't generalise well to the test data and there is a significant drop in AUC (figure 4). It seems that the negative class is required to determine a reliable decision boundary for some cancer types, but not for all. It could be that KIRC tumours cluster together more often than ALL and AML tumours in the constructed feature space within the MIL method. But the reasons for this reduced performance remain largely unclear.

We have also attempted to replace the L1-SVM by a Lasso regression. A Lasso problem is very quickly solved and doesn't require the same excessive amounts of memory as the L1-SVM. There is therefore no need to perform ranking based selection. The Lasso MIL produces a higher AUC for most cancer types and a lower classification error on the positive class for all (see figure S3).

But when t-test ranking selection is used we come across unexpected results. The difference in AUC between an experiment with 10% and 100% of the mutations is negligible. This would indicate that, contrary to earlier results, the ranking is suddenly capable of identifying the most descriptive mutations and roughly 10% of the mutations are relevant (see supplementary document 1). More work is required to find out whether the obtained results are trustworthy.

Another way of reducing the footprint is to change from selecting individual mutations to selecting groups of mutations. Groups could be created in many different ways, but a straightforward approach would be to group mutations by gene. The problem would then reduce from selecting out of 67.067 mutations to selecting out of roughly 20.000 genes. The efficacy of the remodelled MIL problem could be tested by inspecting the number of selected known cancer genes, in general and with respect to different cancer types.

The selected groups could be unpacked into a new reduced dataset that contains only the mutations in the genes that have been found most informative when separating cancer types. The mutations required for separation can now be obtained by running our current method, without the rank selection step, on this reduced dataset. Such a two-tiered setup would therefore reduce the memory footprint considerably. Another advantage is that the number of genes doesn't change, no matter how many tumours or mutations are used. Meanwhile, it would still allow for investigation of a classifier in terms of its selected mutations.

In this work we use a very simple set of annotations. The feature optimisation step showed that there is room for improvement. New features of interest could be provided by a clustering approach on a protein-protein interaction network or on a network of Hi-C measurements. Such an approach could provide useful information about whether mutations in different genomic regions are related. Other features could be the protein domain that is affected by the mutation and whether the mutation occurred in a transcription factor binding site. We could also add more types of mutations such as structural variants and copy number changes that have been implicated with the development of cancer.

In conclusion, the multiple instance learning paradigm of bags with observations are a natural extension to somatic mutations obtained from tumours. The mutations can be annotated with any number of properties without the need to modify the basic model. Performance of a MIL based method is comparable to a gene scoring approach. While its performance is within two standard deviations of that obtained through a distance based method, but with the advantage that the classifier becomes biologically interpretable. Further development is required to reach its full potential. This paper is a stepping stone, we provide compelling evidence that multiple instance learning is a good fit when working with mutation data.

5 METHODS

5.1 Data

5.1.1 Sequencing Sequencing is a family of techniques that can be used to determine the sequence of DNA. The data used during this project is derived through a variant of sequencing named next generation sequencing (NGS). DNA is first extracted from a number of cells and cut into small, single strand pieces. Through a number of preparation steps sequenceable DNA molecules are created. The DNA sequence is determined by iteratively incorporating a specifically created base that emits a fluorescent colour. The colour is captured through a photograph before the next base is incorporated. Each DNA molecule will result in a pair of reads that consist of a series of letters from the alphabet $\{A, C, G, T\}$.

A whole genome results in billions of read pairs. Each of the pairs are aligned to a reference, after which the chromosomal location of the reads is known. Reads have been designed in such a way that they overlap after alignment (see the top half of 2a). The target sequence is therefore covered multiple times, which is known as

Type	Samples	Mutations	Class prior
ALL	141	1900	0.094
AML	158	1882	0.105
KIRC	325	28136	0.217
KIRP	100	6350	0.067
OV	472	23264	0.315
THCA	304	5541	0.203
Total	**1500**	**67067**	**1.000**

Table 2. Overview of the contents of the dataset used in this report. The class prior is the fraction of tumours per cancer type in the total dataset.

coverage. The reads aligning to a particular location can be used to find mutations.

But somatic mutation calling is more complicated than just finding genomic locations where a mismatch with the reference occurs. It requires sequence from a normal genome (usually obtained from blood cells) and from the cancer genome. A tumour is heterogeneous, which means it contains both normal and tumour cells. While the tumour cells can have differing genomes due to ongoing mutational processes. A mutation could therefore be present in a small portion of the cells. Furthermore, a tumour genome can differ quite substantially from a normal genome, but reads are aligned to the same reference genome nonetheless.

Those mutations that occur in the tumour but not in the blood are thought to be somatic. But it is not straightforward to determine that a mutation is present in the tumour or it is a mistake made during sequencing. A number of somatic mutation calling tools are developed as a solution to this problem. They are based on different ideas on how to call a somatic mutation. These difference result in a low concordance rate between somatic mutation callers [Pabinger *et al.*, 2013, Roberts *et al.*, 2013].

In this project we use the data from Alexandrov *et al.*. The raw mutations were obtained from different large cancer sequencing projects and collaborations. These projects run across institutes around the world, that use different sequencing platforms, protocols, pipelines and parameters. It is therefore unsafe to assume that each obtained mutation is real.

5.1.2 Annotation The mutation calls are annotated with a minimal set of annotations through Annovar [Wang *et al.*, 2010]. The information added includes the gene in which a mutation occurs, genomic function (exon, intron, UTR, etc.), effect on the exon (synonymous, non-synonymous, stopgain, etc.) and whether the mutation causes an amino acid substitution. Furthermore, information from variant effect predictors SIFT, PolyPhen 2, PhyloP, AVSIFT, LRT, MutationTaster and GERP++ is added. All predictors assign a score that represents the estimated impact of the mutation, while all but AVSIFT and GERP++ also assign a classlabel. Predictors can usually choose from classes that represent neutral, possibly damaging and disease causing.

The annotations are subsequently encoded into features. Each annotation is either of the type numerical, boolean or categorical. A numerical or boolean annotation is encoded as is and therefore results in an equal number of features. For categorical annotations a feature is created for each of the unique categories that are contained within the annotation. Table S1 gives an overview of the complete encoding of the basic annotations.

A number of additional features have been created around gene names and pathways. Gene names were added to mutations through Annovar. We have created a canonical list of protein coding genes through Ensembl [Kinsella *et al.*, 2011] and mapped each gene name into its Ensembl gene id. If the gene name did not exist as protein coding gene name or was known as a synonym for a protein coding gene it was mapped onto a category named *other*. A total of 22.665 protein coding genes were obtained, of which 20.405 are mutated at least once.

5.1.3 Selected data Exome data from six types of cancer from the Alexandrov *et al.* [2013b] dataset was used during this project. We expected a MIL method to be computationally heavy and therefore wanted to create a diverse dataset that represents the full data but was not extremely large. The six types listed in table 2 contain samples from different cancer families, as well as samples from the same tissue of origin. It also includes samples with a high and low average number of mutations.

5.1.4 Experiment setup Each of the experiments is run through a similar setup. We use a one-versus-all scheme where one class is relabelled as positive and the others as negative. This results in a classifier per cancer type. In order to combat the class imbalance problem the operating point is set to an equal error rate for both the positive and the negative class.

All performance and error measurements are obtained through 10-fold cross validation. During each cross validation round we first scale the training data to a mean of zero and a standard deviation of one. The obtained scaling is saved and applied to the test data later on. We make an exception for the MIL experiments. Calculating the distance matrix M is computationally intense and the matrix takes up a significant amount of memory. We therefore choose to calculate M offline and plug it into the classifier. Scaling is performed on the full M, which introduces a small bias. But the effects on overall performance are negligible (table S1).

5.2 Gene Scoring

For gene scoring experiments the mutation data is transformed into a gene scoring matrix. First a canonical list of gene names and their synonyms was obtained from Ensembl. Through the gene name annotation provided by Annovar mutations have been mapped onto genes. Each tumour is now reduced to a vector of integers, one for each gene. The integer is the total number of mutations within the start and end coordinates of a particular gene.

In total 64.669 mutations were mapped onto a protein coding gene. A mutation falls within a gene when its chromosome location is between the genes start and end location. An extra column was added to the dataset to accommodate the total of 2411 mutations that fall within non-coding genes or outside any known Ensembl gene.

Through these experiments it is assumed that the mutation and its annotations have no effect on the separation of the classes. Whether the mutation falls within an intron or exon for example is irrelevant, as is the possible effect the mutation has on the transcribed protein.

5.3 Distance

In the distance experiments all mutation data is used. Each tumour is first modelled as a labelled bag that contains the somatic mutations obtained for that particular tumour. This bag level data is then transformed into a regular classification problem by calculating the distance between each pair of bags.

Consider the example in figure 8a. It consists of bags B_1 and B_2 that contain a number of unlabelled observations, which are described by two features. To calculate the distance between the two bags we first obtain the distance between each pair of observations. The Euclidean distance is used as a measure of distance between two observations $\boldsymbol{x}_i \in B_i$ and $\boldsymbol{x}_j \in B_j$, which are both described by n features:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{l=1}^{n} (x_{il} - x_{jl})^2} \tag{1}$$

Next we combine the pairwise observation distances into a bag distance by taking the sum over all pairs. The sum blends information from observations into a single value, whereas taking the min or max would select a single observation. We choose the sum because it is less sensitive to outliers as compared to the min and max.

$$d(B_i, B_j) = \sum_{k \in B_i, l \in B_j} d(\boldsymbol{x}_{ik}, \boldsymbol{x}_{jl}) \tag{2}$$

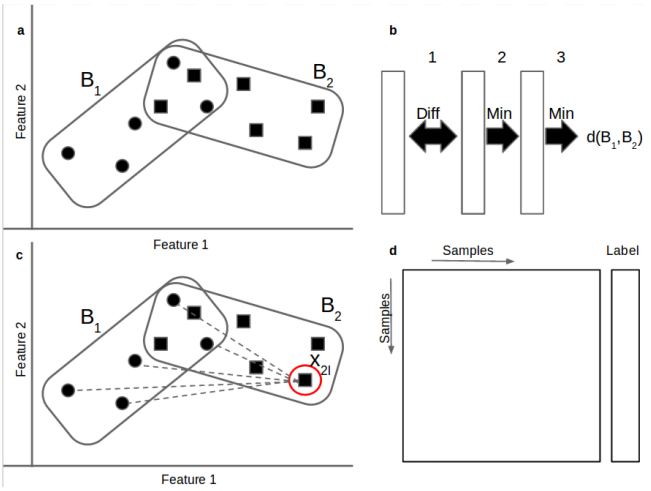


Fig. 8. (**a**) A MIL example with two bags, each containing a number of observations which are described by two features. (**b**) In the distance experiments a three step transformation of the MIL data to a regular classification problem is used. 1. Each observation is a vector that contains features. We take the difference between the corresponding features in each pair of observations. 2. When calculating the minmin distance we take the minimum of the values obtained in step 1 as the distance between a pair of observations. Each pair is therefore reduced to a single value, which means that the distance between a pair of bags is represented by a vector. 3. Finally we take the minimum value in this vector as the representation of the distance between the two bags. (**c**) Pairwise distances are calculated between observations in both the distance and multiple instance learning experiments. (**d**) The transformation of the observations to distances results in a square matrix. After assigning the bag labels to each row of the new matrix we obtain a regular classification problem.

After obtaining the pairwise distances between bags we have obtained a regular classification problem (figure 8d). The problem is a square matrix, in which each row represents a bag which is assigned the original bag label. This regular classification problem is finally handed to a linear SVM to obtain a bag level classifier.

5.4 Multiple Instance Learning

For the MIL classification experiments we have adapted the MILES classifier published by Chen *et al.* for the specific requirements of this project. The classifier is principally a wrapper that transforms the data from a MIL problem into a regular classification problem by calculating a similarity between bags and observations. Each bag is now represented by a vector that contains a similarity to each observation. A regular classifier can now be used to obtain a decision boundary. This section is roughly based the MILES explanation by Chen *et al.* [2006].

Encoding In a MIL problem each bag B_i contains a number of unlabelled observations $\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{in}$. Each bag is assigned a label that is either *positive* or *negative*. Here each bag is encoded as a similarity to each observation. A distance measure is used to represent similarity.

Consider for example the circled observation in bag B_2 in figure 8c. Calculation of the distance to bag B_1 consists of two steps. First the distance between $\boldsymbol{x}_{2l}$ and each observation $\boldsymbol{x}_{1k} \in B_1$ is calculated. We then obtain the distance between the observation and the bag by taking the minimum of these distances. In more general terms the distance between bag B_i and instance $\boldsymbol{x}$:

$$d(B_i, \boldsymbol{x}) = \min(d(\boldsymbol{x}_{i1}, \boldsymbol{x}), \ldots, d(\boldsymbol{x}_{ik}, \boldsymbol{x})) \tag{3}$$

where there are k observations in B_i and $\boldsymbol{x} \notin B_i$. By applying equation 3 to all N bags and m available observations we obtain the following matrix:

$$M = \begin{bmatrix} d(B_1, \boldsymbol{x}^1) & \ldots & d(B_1, \boldsymbol{x}^m) \\ d(B_2, \boldsymbol{x}^1) & \ldots & d(B_2, \boldsymbol{x}^m) \\ \ldots & \ldots & \ldots \\ d(B_N, \boldsymbol{x}^1) & \ldots & d(B_N, \boldsymbol{x}^m) \end{bmatrix} \tag{4}$$

M contains a row for each bag and a column for each observation in the data set, regardless of label. Each bag still has its label, M is therefore a new regular classification problem.

Selection When the input problem contains many observations the matrix M grows very wide. This presents a problem at the solving stage where a coefficient is estimated for each observation. We therefore introduce a selection step that aims to select the most informative observations and reduce the number of columns in M. A number of different selection methods have been developed, of which two are described here. Both selection methods introduce an extra parameter f that represents the fraction of observations selected.

T-test rank Each column in M represents an observation. It contains the distance of observation $\boldsymbol{x}^i$ to each bag in the data set. We perform a t-test between each column and the numerically encoded bag labels $\boldsymbol{l}$ to obtain a p-value. Such a value represents the likelihood that both vectors are drawn from the same distribution.

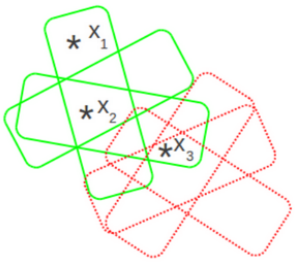


Fig. 9. An example problem containing three positive and three negative bags. All three observations reside in positive bags. The ratio selection is based on the following rationale: $\boldsymbol{x}_2$ falls in the high density area of the positive class and should ideally be selected. $\boldsymbol{x}_1$ does not fall in the high density area, but is far removed from the negative high density area. This observation could be selected. $\boldsymbol{x}_3$ falls within the negative high density area, while it also falls in a low positive density area. This observation should not be selected.

$$p^i = \text{t-test}(\boldsymbol{x}^i, \boldsymbol{l}), i = 1, \ldots, m \tag{5}$$

The p-values allow for ranking, after which the observations corresponding to the top p-values are selected.

Ratio rank The ratio ranking is illustrated through figure 9. The figure contains six bags, three positive (green) and three negative (red). Highlighted are three observations that all reside in positive bags.

Observation $\boldsymbol{x}_2$ resides in the area that contains the highest positive density. Principally this observation is therefore of interest. Observation $\boldsymbol{x}_1$ is relatively far removed from the highest positive density area. But it is also relatively far removed from the negative high density area. Selection of this observation could therefore be tolerated. Observation $\boldsymbol{x}_3$ however is close to the negative high density area and should therefore not be selected.

The ratio ranking aims to approximate this observation by calculating the ratio of the average distance to positive bags versus the average distance to negative bags. In equation 6 $\boldsymbol{x}^i_{pos}$ are the distances to positive bags in column i of M and $\boldsymbol{x}^i_{neg}$ are the distances to negative bags. We now obtain the most informative observations by ranking the corresponding ratios and selecting the top fraction.

$$r^i = \frac{\text{mean}(\boldsymbol{x}^i_{pos})}{\text{mean}(\boldsymbol{x}^i_{neg})}, i = 1, \ldots, m \tag{6}$$

Fraction determination Both rankings described above allow for selection of a particular fraction of observations. A number of experiments have been performed to estimate a suitable fraction of observations to select. Figure S2 shows the classification error and AUC versus a number of fractions respectively. There is not a big performance increase as more data is added. We choose to work with a fraction of 0.20 as it yields the highest average AUC across all cancer types.

Solving After transformation and selection the matrix M contains a row for each bag and a column for each selected observation. Similarly to Chen *et al.* we use an L1-SVM to construct a classifier. Such a bag level SVM classification problem is formulated as a linear classifier:

$$y = \text{sign}(\boldsymbol{w}^T \boldsymbol{m} + b) \tag{7}$$

where y is the predicted class label for bag $\boldsymbol{m}$ and $\boldsymbol{w}$ and b are model parameters.

SVMs typically minimise a regularised error on the training data that consists of a regulariser P, its parameter λ and a hinge loss ξ. The hinge loss (equation 9) contains $\boldsymbol{w}$ and b, for which an estimation is desired. $\boldsymbol{w}$ contains a component w_i for each selected observation that represents the contribution of observation i to the classifier.

$$\lambda P + \xi_{training} \tag{8}$$

$$\xi = \max\{1 - y(\boldsymbol{w}^T \boldsymbol{m} + b), 0\} \tag{9}$$

We use the 1-norm of $\boldsymbol{w}$ as regulariser. A 1-norm drives as many of the components in $\boldsymbol{w}$ down to zero as possible and therefore returns a sparse solution. The added advantage is that the 1-norm can be solved as a linear program, which is less computationally intense that the standard quadratic programming solution for the squared 2-norm.

$$||\boldsymbol{w}||_1 = \sum_i |w_i| \tag{10}$$

In this paper we use a one-versus-all setup which results in a large class imbalance. To combat this issue we add parameters C^{pos} and C^{neg} that introduce a bias towards the smaller class. Input parameter C is scaled by the total number of either positive or negative bags. The C parameter controls the sparsity of the solution, where a large value allows more weights to be non-zero.

By combining the hinge losses ξ and η for the positive and negative class we obtain the full 1-norm SVM formulation:

$$\begin{aligned} \min_{\boldsymbol{w},b,\xi,\eta} \quad & \lambda \sum_i |w_i| + \frac{C}{N^{pos}} \sum_j \xi_k + \frac{C}{N^{neg}} \sum_k \eta_k \\ \text{s.t.} \quad & (\boldsymbol{w}^T \boldsymbol{m}^{pos}_j + b) + \xi_j \geq 1, j = 1, \ldots, n^{pos} \\ & -(\boldsymbol{w}^T \boldsymbol{m}^{neg}_k + b) + \eta_k \geq 1, k = 1, \ldots, n^{neg} \\ & \xi_j, \eta_k \geq 0 \end{aligned} \tag{11}$$

Like Chen *et al.* we rewrite 11 into a linear program. The linear program returns estimates for both $\hat{\boldsymbol{w}}$ and $\hat{b}$ which allows for construction of the final bag level classifier:

$$y = \text{sign}\left(\sum_{i \in I} \hat{\boldsymbol{w}}_i d(B_k, \boldsymbol{x}^i) + \hat{b}\right) \tag{12}$$

5.5 Biological interpretation

The MIL classifier selects a number of mutations during training. The properties of these mutations can be used to biologically interpret how a classifier can separate types of cancer. In order to obtain this information we calculate whether mutations with a particular feature are selected more often than one would expect by chance.

Consider figure 10a. The input data for the MIL classifier consists of a list of N observations (all mutations). Each observation resides

in a bag, which is assigned a label (the type of cancer). In the one-versus-all setup we create a classifier for each cancer type. If a mutation contributes to the decision boundary it is either assigned a positive or a negative weight (figure 10b). After training the MIL classifier returns the list of selected mutations, N_{pos} mutations with a positive weight and N_{neg} mutations with a negative weight.

A $\log_2$ ratio of the observed-versus-expected ratio can be calculated for each property, for both the positive and negative class. First the fraction of mutations with the property k of interest in the whole dataset (equation 13) and in the selection (14) is calculated.

$$f^k = \frac{\sum_{i=1}^{N} x_i^k}{N} \tag{13}$$

$$f_{pos}^k = \frac{\sum_{i=1}^{N_{pos}} x_i^k}{N_{pos}} \tag{14}$$

The fractions are then transformed into a $\log_2$ ratio. Such a ratio is positive when more mutations with the property are selected than expected by chance and negative when less are selected. These steps are taken for both the positive and negative class.

$$r_{pos}^k = \log_2 \frac{f_{pos}^k}{f^k} \tag{15}$$

The r_{pos}^k and r_{neg}^k ratios are subsequently visualised through a heatmap, of which figure 6 shows an example.

REFERENCES

Alexandrov, L., Nik-Zainal, S., Wedge, D. et al (2013a). Deciphering signatures of mutational processes operative in human cancer. *Cell Reports*, **3**(1), 246–259.

Alexandrov, L.B., Nik-Zainal, S., Wedge, D.C. et al (2013b). Signatures of mutational processes in human cancer. *Nature*.

Berman, J.J. (2004). Tumor classification: molecular analysis meets aristotle. *BMC Cancer*, **4**(1), 10. PMID: 15113444.

Buske, O.J., Manickaraj, A., Mital, S. et al (2013). Identification of deleterious synonymous variants in human genomes. *Bioinformatics*, **29**(15), 1843–1850. PMID: 23736532.

Chen, Y., Bi, J. and Wang, J. (2006). MILES: multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(12), 1931–1947.

Consortium, T.J.H., Anderson, W., Aretz, A. et al (2010). International network of cancer genome projects. *Nature*, **464**(7291), 993–998.

Dietterich, T.G., Lathrop, R.H. and Lozano-Prez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, **89**(12), 31–71.

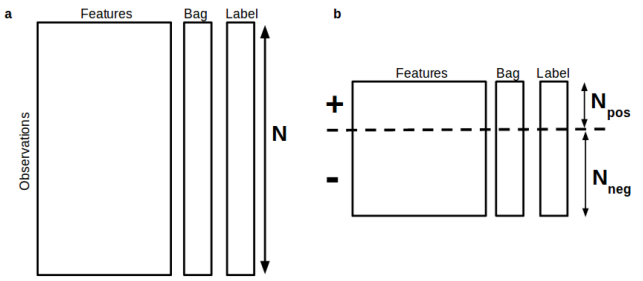


Fig. 10. (**a**) The input data consists of N observations. (**b**) Each one-versus-all classifier returns a subset of mutations that were selected during training. N_{pos} are assigned a positive weight and therefore contribute to the positive class, while N_{neg} are assigned a negative weight and contribute to the negative class. This data is used to calculate a $\log_2$ ratio for each feature.

Downing, J.R., Wilson, R.K., Zhang, J. et al (2012). The pediatric cancer genome project. *Nature Genetics*, **44**(6), 619–622.

Flintoft, L. (2013). Clinical genetics: Exomes in the clinic. *Nature Reviews Genetics*, **advance online publication**.

Futreal, P.A., Coin, L., Marshall, M. et al (2004). A census of human cancer genes. *Nature Reviews Cancer*, **4**(3), 177–183.

Greenman, C., Stephens, P., Smith, R. et al (2007). Patterns of somatic mutation in human cancer genomes. *Nature*, **446**(7132), 153–158.

Hanahan, D. and Weinberg, R. (2011). Hallmarks of cancer: The next generation. *Cell*, **144**(5), 646–674.

Hofree, M., Shen, J.P., Carter, H. et al (2013). Network-based stratification of tumor mutations. *Nature Methods*, **advance online publication**.

Hua, X., Xu, H., Yang, Y. et al (2013). DrGaP: a powerful tool for identifying driver genes and pathways in cancer sequencing studies. *The American Journal of Human Genetics*, **93**(3), 439–451.

Khurana, E., Fu, Y., Chen, J. et al (2013). Interpretation of genomic variants using a unified biological network approach. *PLoS Comput Biol*, **9**(3), e1002886.

Kinsella, R.J., Kahari, A., Haider, S. et al (2011). Ensembl BioMarts: a hub for data retrieval across taxonomic space. *Database*, **2011**(0), bar030–bar030.

Lawrence, M.S., Stojanov, P., Polak, P. et al (2013). Mutational heterogeneity in cancer and the search for new cancer-associated genes. *Nature*, **499**(7457), 214–218.

Need, A.C., Shashi, V., Hitomi, Y. et al (2012). Clinical application of exome sequencing in undiagnosed genetic conditions. *Journal of Medical Genetics*, **49**(6), 353–361. PMID: 22581936.

Ng, P.C. and Henikoff, S. (2003). SIFT: predicting amino acid changes that affect protein function. *Nucleic Acids Research*, **31**(13), 3812–3814. PMID: 12824425.

Pabinger, S., Dander, A., Fischer, M. et al (2013). A survey of tools for variant analysis of next-generation genome sequencing data. *Briefings in Bioinformatics*, page bbs086. PMID: 23341494.

Pawlik, T.M., Delman, K.A., Vauthey, J.N. et al (2005). Tumor size predicts vascular invasion and histologic grade: Implications for selection of surgical treatment for hepatocellular carcinoma. *Liver Transplantation*, **11**(9), 10861092.

Petrakis, D., Pentheroudakis, G., Voulgaris, E. et al (2013). Prognostication in cancer of unknown primary (CUP): development of a prognostic algorithm in 311 cases and review of the literature. *Cancer Treatment Reviews*, **39**(7), 701–708.

Pfeifer, G.P. (2010). Environmental exposures and mutational patterns of cancer genomes. *Genome Medicine*, **2**(8), 54. PMID: 20707934.

Ramaswamy, S., Tamayo, P., Rifkin, R. et al (2001). Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of the National Academy of Sciences*, **98**(26), 15149–15154.

Roberts, N.D., Kortschak, R.D., Parker, W.T. et al (2013). A comparative analysis of algorithms for somatic SNV detection in cancer. *Bioinformatics*, **29**(18), 2223–2230. PMID: 23842810.

Schwarz, J.M., Rdelsperger, C., Schuelke, M. et al (2010). MutationTaster evaluates disease-causing potential of sequence alterations. *Nature Methods*, **7**(8), 575–576.

Tamborero, D., Gonzalez-Perez, A., Perez-Llamas, C. et al (2013). Comprehensive identification of mutational cancer driver genes across 12 tumor types. *Scientific Reports*, **3**.

The Cancer Genome Atlas Research Network, Weinstein, J.N., Collisson, E.A. et al (2013). The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, **45**(10), 1113–1120.

van de Vijver, M.J., He, Y.D., van 't Veer, L.J. et al (2002). A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, **347**(25), 1999–2009. PMID: 12490681.

Veer, L.J.v.t., Dai, H., Vijver, M.J.v.d. et al (2002). Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, **415**(6871), 530–536.

Wang, K., Li, M. and Hakonarson, H. (2010). ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Research*, **38**(16), e164–e164. PMID: 20601685.

Yang, Y., Muzny, D.M., Reid, J.G. et al (2013). Clinical whole-exome sequencing for the diagnosis of mendelian disorders. *New England Journal of Medicine*, **369**(16), 1502–1511. PMID: 24088041.

Yates, L.R. and Campbell, P.J. (2012). Evolution of the cancer genome. *Nature Reviews Genetics*.

Yeang, C.H., McCormick, F. and Levine, A. (2008). Combinatorial patterns of somatic gene mutations in cancer. *The FASEB Journal*, **22**(8), 2605–2622. PMID: 18434431.

Youn, A. and Simon, R. (2011). Identifying cancer driver genes in tumor genome sequencing studies. *Bioinformatics*, **27**(2), 175–181. PMID: 21169372.

1 SUPPLEMENTARY FIGURES AND TABLES

1.1 Scaling

Type	Scale per cv	Scale overall
ALL	0.17 (0.05)	0.17 (0.05)
AML	0.17 (0.07)	0.18 (0.07)
KIRC	0.22 (0.05)	0.22 (0.05)
KIRP	0.13 (0.04)	0.13 (0.04)
OV	0.35 (0.05)	0.34 (0.05)
THCA	0.24 (0.07)	0.25 (0.07)

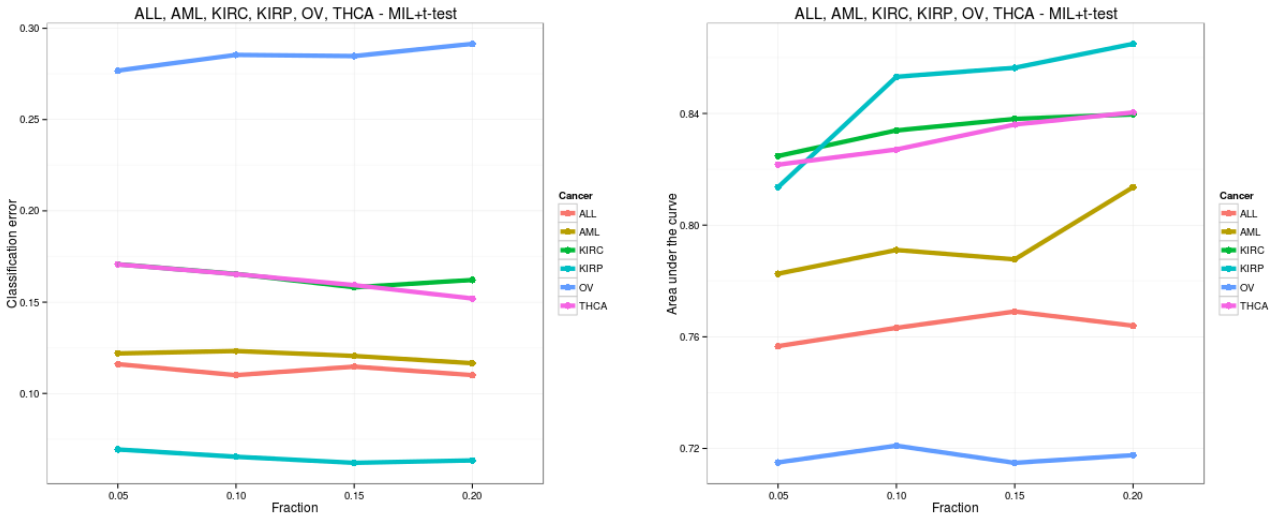
(a) Cl.err

Scale per cv	Scale overall
0.72 (0.11)	0.71 (0.11)
0.75 (0.08)	0.75 (0.07)
0.78 (0.07)	0.78 (0.07)
0.72 (0.11)	0.71 (0.07)
0.65 (0.09)	0.66 (0.08)
0.76 (0.09)	0.75 (0.08)

(b) AUC

Fig. S1. Comparison of the classification error and AUC for a MIL experiment with scaling per cross-validation fold and with overall scaling. For both experiments a t-test selection of 1% of the data was performed. All parameters remained unchanged. We conclude that calculating and scaling the MIL distance matrix and offline has a negligible effect on performance.

1.2 Fraction determination

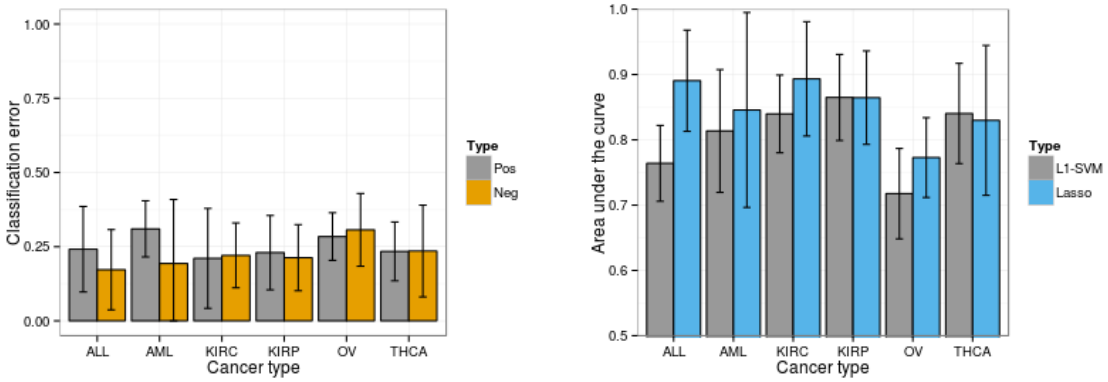


(a) Cl.err

(b) AUC

Fig. S2. (**a**) The combined classification error of the classifiers obtained when selecting observations through a t-test. The error rate levels off quickly, showing not much progress beyond the selection of 10% of the observations. (**b**) The AUC of the six classifiers obtained when using a t-test. For most classifiers the AUC is still on the rise at 20%. This indicates that there might be a further increase when more observations are selected. But selecting more mutations proves to be difficult due to memory requirements.

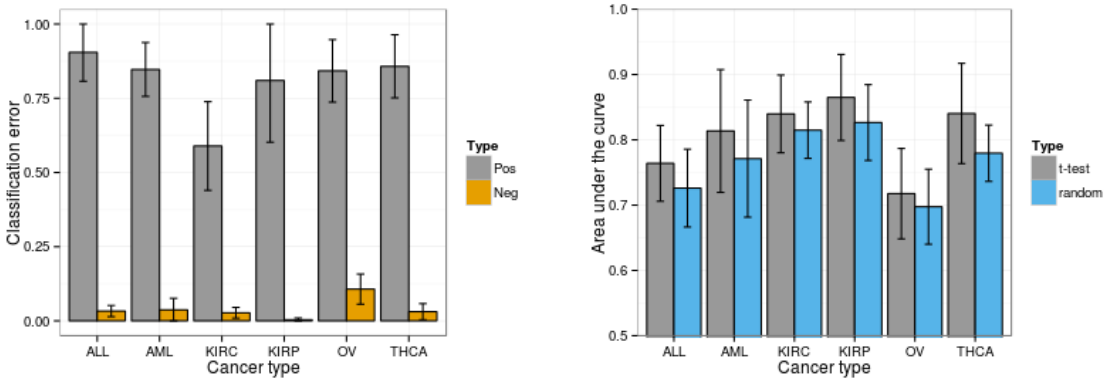
1.3 Replacing the L1-SVM by Lasso



(a) Classification error (b) AUC

Fig. S3. (**a**) Classification error obtained for the MIL experiment after the L1-SVM is replaced by Lasso. 20% of the observations were selected through a t-test and the penalty term was biased by the class size. The classification error on the positive class is greatly reduced, although it goes up considerably for the negative class in all cases. (**b**) AUCs obtained through the L1-SVM and Lasso MIL methods show an improvement in four cases, but also an increase in standard deviation in five cases.

1.4 MIL with random mutation selection



(a) Classification error (b) AUC

Fig. S4. (**a**) Classification error obtained for the MIL experiment when 20% of the observations were randomly selected. The classification errors on the negative class are similar to what was obtained through a t-test. The positive class has a higher error, except for KIRC and KIRP. (**b**) The obtained AUCs are lower for all cancer types than when selecting mutations through t-test ranking. This means that t-test ranking is useful, although the difference with random selection is small.

1.5 Feature encoding

Annotation	Type	# Features
Chromosome	Cat	25
Location	Num	1
Reference base	Cat	4
Observed base	Cat	4
Genomic function	Cat	14
Exon effect	Cat	11
AA change	Bool	1
SIFT	Num	1
SIFT pred	Cat	3
PolyPhen 2	Num	1
PolyPhen 2 pred	Cat	4
PhyloP	Num	1
PhyloP pred	Cat	3
AVSIFT	Num	1
LRT	Num	1
LRT pred	Cat	4
MutationTaster	Num	1
MutationTaster pred	Cat	5
GERP++	Num	1

Table S1. Each of the annotations supplied by Annovar is encoded into a number of features, based on the type of annotation. A categorical annotation will be encoded into as many features as that it contains levels, while a boolean or numeric annotation results in a single feature. In total the Annovar annotation results in 97 features.

Master Thesis - Supplement 1

Stefan Dentro

November 29, 2013

This document contains a summary of various experiments performed and leads followed that did not make it fully into the final report. The information below, combined with the final report and the weekly reports represent the full body of work.

1 Data annotation and QC

The data was annotated with Annovar, VEP, and SeattleSeq. Both VEP and SeattleSeq can annotate a mutation multiple times (due to different transcripts) and therefore add an additional layer of complexity to the data. In the end we've decided to use the Annovar information. It contains the annotations that are available in the other two as well but it performs a single annotation per mutation.

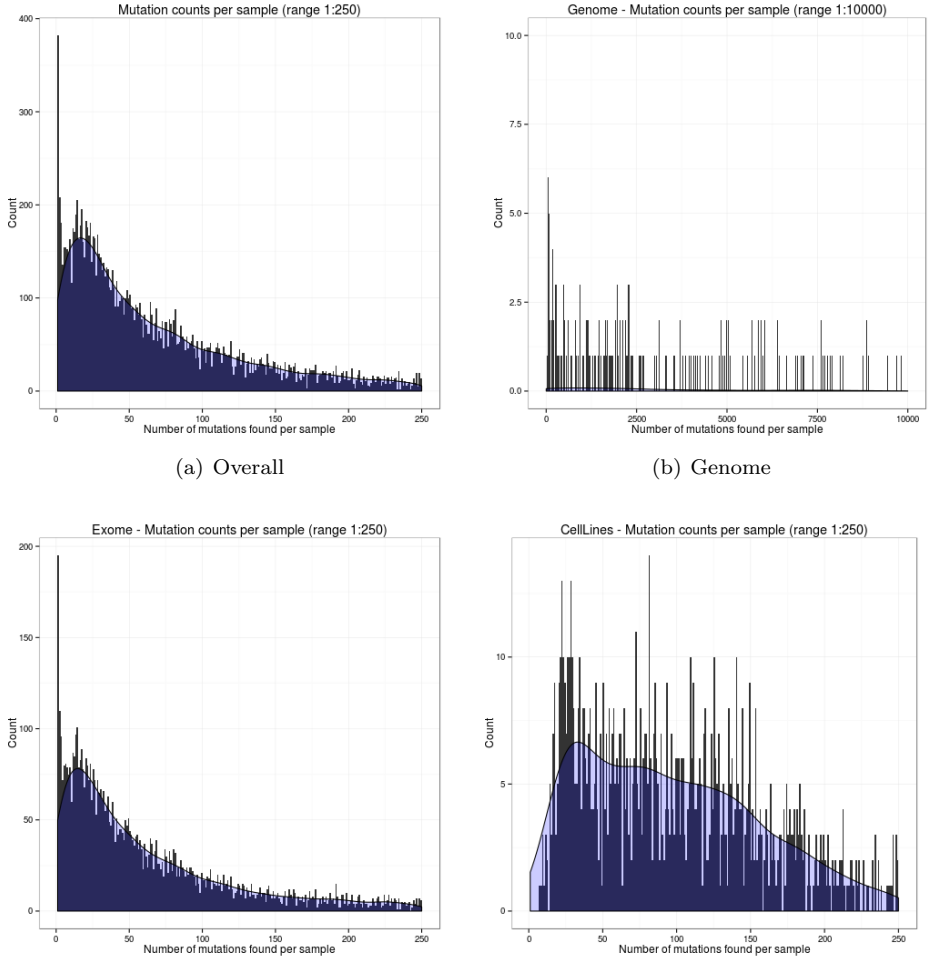
We've also created addon annotation modules that have mostly gone unused. There are ENCODE, GO, known CGC cancer genes, Reactome pathway and Pfam domain annotations available. These are all chromosome position based. Tests with the known CGC cancer genes and Reactome pathway annotations didn't yield any improved performance. This information should probably be added in a more meaningful way, like a scale-space aware network model. Due to time restrictions this never materialised.

During QC we've made sure that all samples are based on the same reference genome and investigated the properties of the different cancer types within the datasets. We observed that a number of exome cancer types contained a large number of intergenic mutations (fifth bar in the plots in figure 2) for HCC, LYMP, MB, PA and RADSAR. This is unexpected for exomes. Furthermore, CG>TA mutations generally occur more often than others, which is due to the ease at which a C>A mutation can occur.

Annotation and QC was performed on both the exome and genome datasets, which gave us a better understanding about the data. In the end only 8 types of cancer from the exome set have been used during the project. We also decided to remove the cell line exomes from the complete exome dataset as it is unclear

which type of cancer is modelled by individual cell lines.

Figure 1 shows the distribution of mutation counts for samples in the different data sets. It is mainly the exomes that have a large number of samples between 1 and 10. This could mean that a relatively large number of exome samples are difficult to classify due to a low mutation availability. On the other hand, it could mean that genomes are too difficult because of the amount of mutations contained within only a few samples.



(a) Overall

(b) Genome

(c) Exome

(d) Cell lines

Figure 1: Distribution of samples with a given mutation count. Bin-width is one (10 for genomes), therefore each bar represents the number of samples containing the number of mutations as specified on the x-axis. For viewing convenience the x-axis range has been limited.

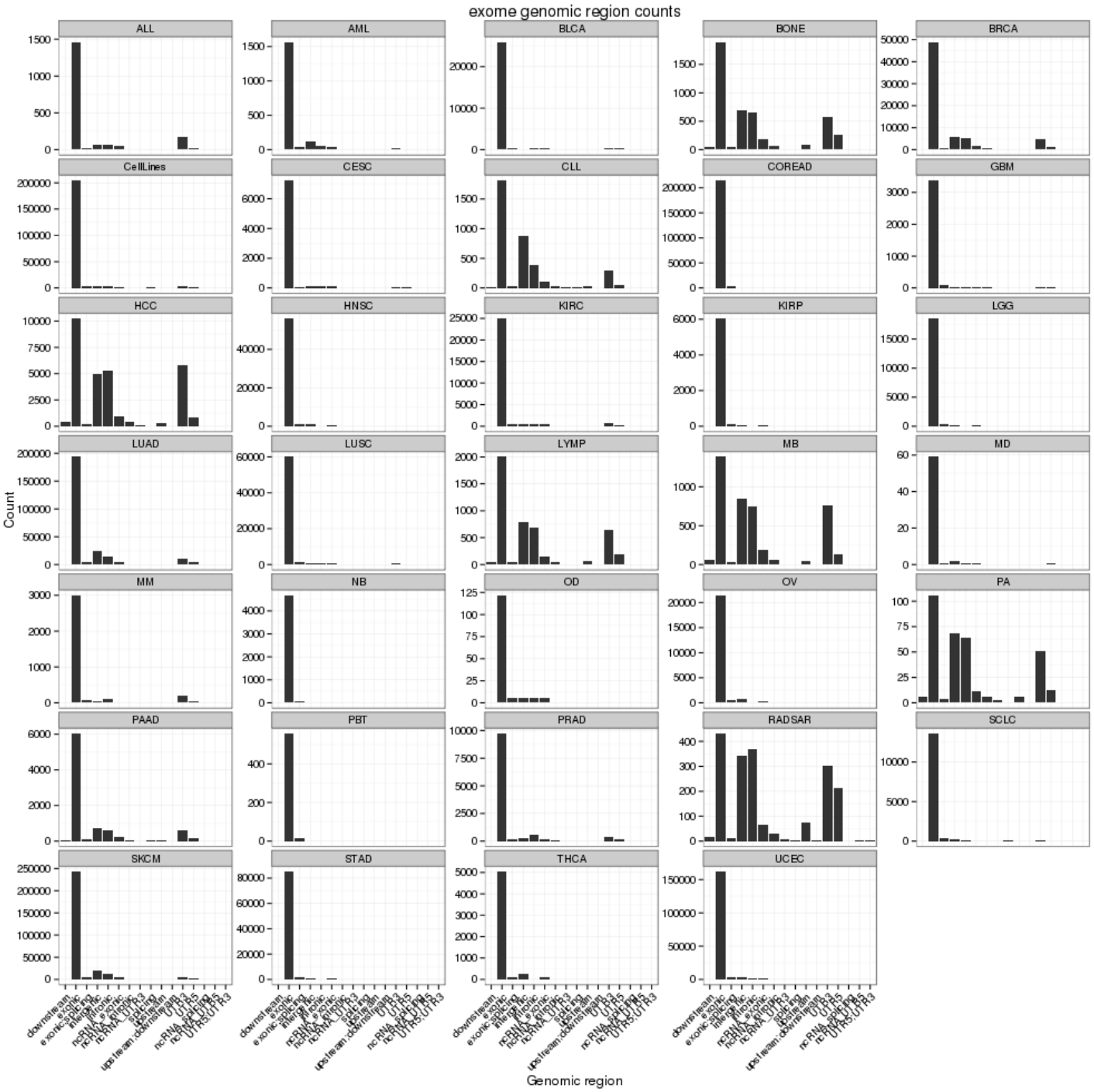


Figure 2: Counts of the occurrence of different genomic region annotations in the exome dataset.

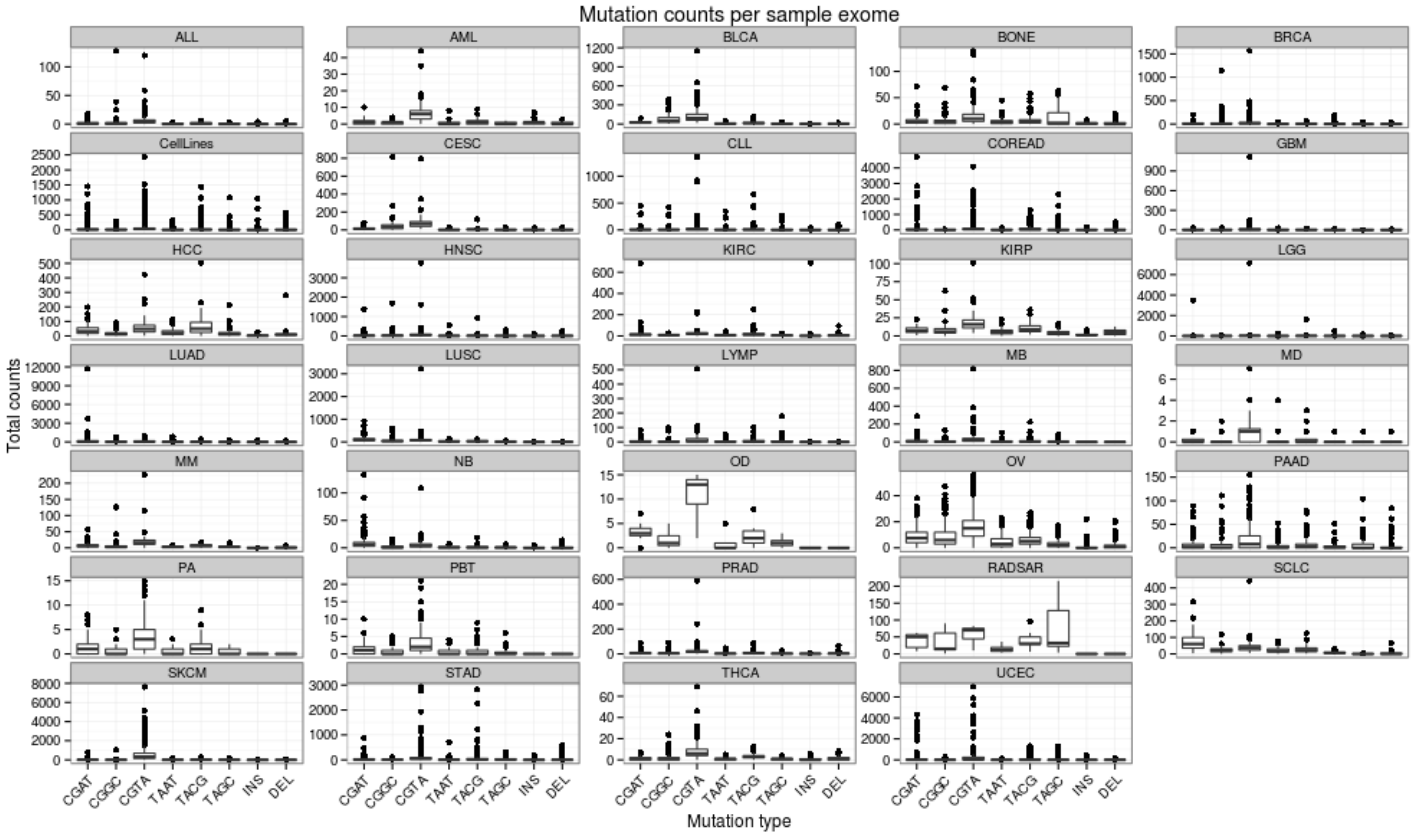
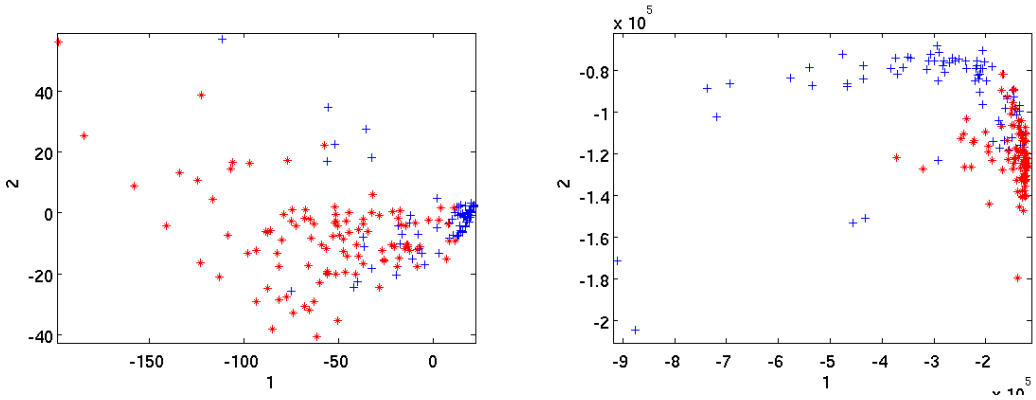


Figure 3: Counts of mutation types per sample, per cancer type in the exome dataset.

2 Pilot project

During the pilot project the three families of experiments were created. Using BRCA (breast cancer) and LUAD (lung) exomes we aimed to test the main ideas behind these experiments. These two were chosen as they have plenty of samples, there was some previous experience with this data and exome data meant that almost all annotation fields would be filled in. First experiments were developed on 10% of the data, later they were scaled up to all available mutations for both types.

Initial genescoring experiments were also tried with other classifiers, beyond NMC, SVM and random forest. But it turns out most classifiers have difficulties with such a sparse dataset. The initial distances also included meanmean, meanmin, Hausdorff, Mahalanobis and mi-RBF. We quickly noticed that minmin (figure 4(a)) and summin (4(b)) yielded interesting results when projected down into its first two principal components. There appears to be a bit of structure there.



(a) minmin (b) summin

Figure 4: PCA of the distance matrix using BRCA (red) and LUAD (blue). There appears to be a bit of structure in these distance encodings. With 10% of the data.

The exact results are omitted here as there have been numerous improvements in the experimental setup that make the old values unreliable with the current knowledge. At the time we concluded that the genescoring experiments yielded a performance that was slightly better than randomly assigning labels. The distance based experiments were a big improvement, while the MILES experiment was even better.

With the genescoring table in hand we also performed Fishers exact test on the total sum of mutations per gene per cancer type. We compared BRCA and LUAD and listed the genes with the lowest p-values. The list of p-values lower than 0.05 contained a few known cancer genes (5/15, table 1).

Finally, experimentation with the MILES classifier was restricted to the 10% data as we discovered the memory footprint required by the linear programming based internal classifier. We also determined that a polynomial kernel with a low parameter yielded the lowest classification error and highest AUC.

Gene	P-value	In CGC	CGC tumour types (somatic)
RBMXL3	0.02410298	No	
LPPR2	0.02410298	No	
GRID2IP	0.0410883	No	
PIK3CA	6.345687e-05	Yes	colorectal; gastric; gliobastoma; breast
GREB1L	0.007979655	No	
CBFB	0.0004280226	Yes	AML
XBP1	0.03226574	No	
ZFP92	0.03915817	No	
CTCF	0.008416183	No	
GATA3	0.03320735	Yes	Breast
PIEZO1	0.04802539	No	
EFR3B	0.03915817	No	
MAP3K1	2.042518e-09	No	
MAP2K4	0.0006687656	Yes	pancreatic; breast; colorectal
CDH1	0.0001649925	Yes	lobular breast; gastric

Table 1: Genes with a P-value lower than 0.05 when comparing the BRCA exome gene scoring against the LUAD exome gene scoring.

3 Genescoring on all exomes

We performed one genescoring experiment on all the exomes. Here the one-versus-all scheme was used. Figure 5 shows the classification error obtained per class. There is quite a bit of variability, but it should be noted here that these results were not properly corrected for the class imbalance. It would also have been better to look at the classification rate for both the positive and negative class, but these insights arrived later on during the project.

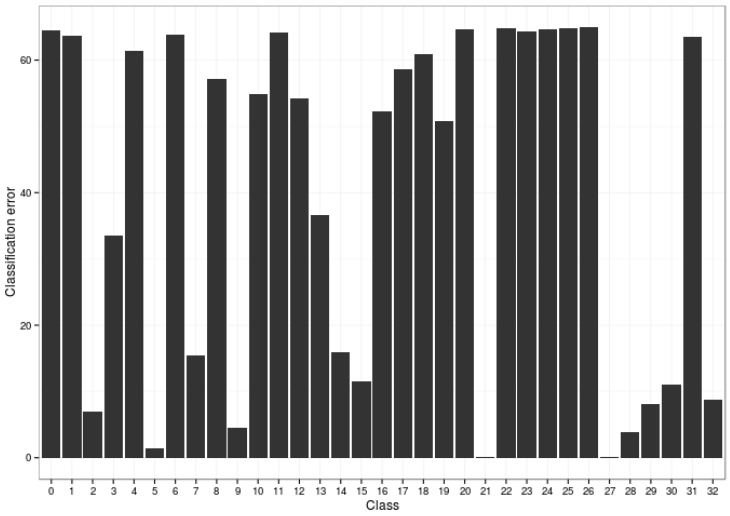


Figure 5: Classification error obtained on all the available exomes using a one-vs-all setup. Out of these results we later chose the classes that are labelled as 0, 1, 11, 12, 22 and 31 as the project data. Note that the cell lines are not in this figure, therefore the numbers assigned to the classes are shifted by one from 8-on.

4 Selection project data

After obtaining the initial results on the pilot data we aimed to find out whether the experiments would show similar results on more difficult data. We therefore set out to create a new dataset that was much more complex, but not extremely large, as it was already known that MILES is memory intense. Based on figure 5 combined with known properties of some cancers we decided to select ALL, AML, KIRC, KIRP, OV and THCA.

The six class dataset contains cancers from different cell types, but also from the same organ, samples with a high and low average mutation rate. Furthermore, it could be extended with BRCA once the memory issues were resolved to create an even harder problem as OV and BRCA are quite similar. BRCA was never added, so we continued the rest of the project with this six class dataset.

5 MILES extensions

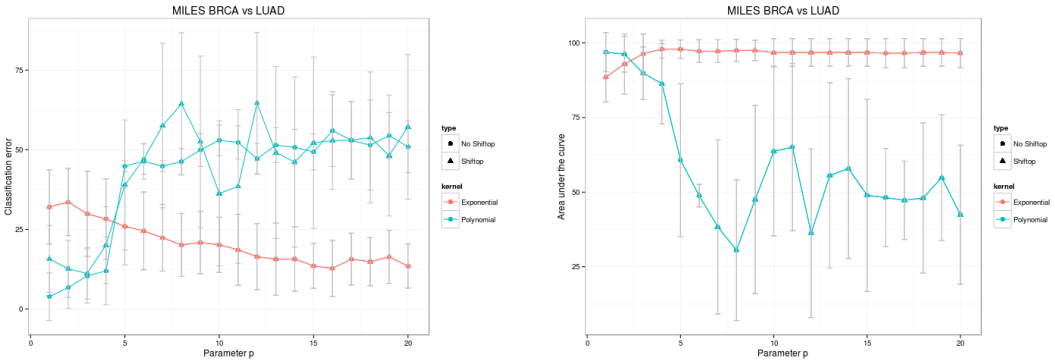
5.1 Determining the kernel and its parameter

Here we investigate the behaviour of the MILES classifier when changing the kernel parameter p for both polynomial and exponential kernels. We ran both experiments with and without shifting the operating point, while using a subset of the BRCA and LUAD exomes.

Figure 6 shows that for the polynomial kernel values up to $p = 4$ yield an interesting classification error and area under the curve. When $p > 4$ the classification error goes up, as well as the standard deviation on the classification error.

The exponential kernel seems to show a downward classification error trend. But at the same time the area under the curve is extremely high with little standard deviation. This could be a sign that there is too little data for applying the exponential kernel.

The results of $p < 3$ seem plausible however and we continue with the polynomial kernel and parameter $p = 1$. Shifting the operating point seems to influence these results only marginally.



(a) Classification error

(b) Area under the curve

Figure 6: Development of the classification error and area under the curve as kernel parameter p is changed.

5.2 Mutation selection

A number of selection procedures have been developed in order to reduce the amount of mutations considered during training. T-test and ratio based rankings and random selection are mentioned in the final report. Additionally correlation coefficient ranking and k-means cluster selection are also available.

In the former a correlation coefficient is calculated between each column in the internal distance matrix and the bag labels. The latter selects the k mutations that are closest to the k clusters obtained after k-means clustering. Initial tests showed that there is little difference in performance between these methods, with t-test and ratio ranking coming out on top.

5.3 L1-SVM replacement

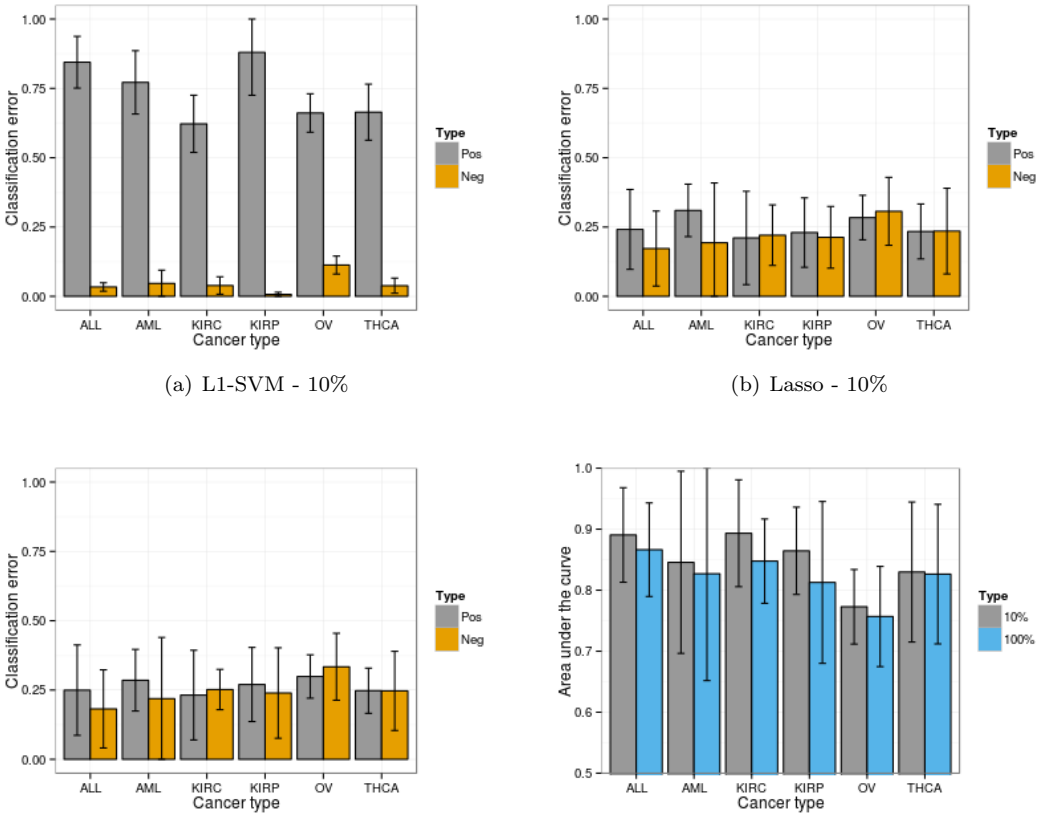
During the pilot phase we wondered what effect an L2-SVM would have on the number of mutations selected during training. The L1-SVM linear programming setup was therefore replaced by a quadratic programming solution. No results were ever obtained due to the CPU time required to solve the quadratic programming problem.

The L1-SVM was also replaced with a Lasso solution. Both methods use a 1-norm penalty that drives as many coefficients down to zero as possible. Each coefficient is tied to a mutation. The 1-norm based regression types therefore yield a sparse selection of mutations. The main advantage of Lasso over the L1-SVM is its speed. A typical problem takes roughly 4 hours to solve through the L1-SVM, whereas the same problem solves in roughly 20 minutes through Lasso.

The main downside of Lasso is that there is no extension that allows for assigning positive coefficients only (see next section). We aimed to add this constraint in order to improve the interpretability of the model. Such an extension is added easily as an extra constraint in a linear programming problem. We opted to keep the Lasso based method around for parameter and data testing purposes, but did not use it for the results in the main paper.

Figures 7(a) and 7(b) show a comparison of the results obtained through the L1-SVM and Lasso based classifiers. The classification error on the positive class is much lower when Lasso is used. But the standard deviation goes up in most cases. We also discovered that the performance difference between selecting 10% and 100% of the data is minimal (figures 7(b) and 7(c)), which was rather surprising. That means that there is very little additional value in 90% of the data and should indicate that the ranking selection works well.

The AUC goes down when selecting more data, while in some cases the standard deviation over the cross-validation splits goes up (figure 7(d)). In conclusion, there seems to be some potential, but it is unresolved whether the results are trustworthy.



(a) L1-SVM - 10%

(b) Lasso - 10%

(c) Lasso - 100%

(d) Lasso AUC

Figure 7: Classification errors obtained through the (**a**) MILES+L1-SVM and (**b**) MILES+Lasso setups when selecting 10% of the mutations through a t-test ranking. (**c**) shows the classification error when using all mutations and MILES+Lasso. (**d**) AUCs obtained for both Lasso experiments.

5.4 Positive weights

To make the biological interpretation more straightforward we aimed to allow for positive weights to be assigned by the internal L1-SVM. Each selected mutation therefore has a positive contribution to the classifier. Properties of all selected mutations can thus be interpreted as characteristics of the type of cancer marked as the positive class.

The positive weight restriction can be seen as an extra linear programming constraint. The linear programming setup estimates the 1-norm through two weights, one for a positive value and one for a negative (see the MIL methods section in the main paper). This is accomplished by requiring that one of the two weights equals zero. Our solution is therefore to not estimate the negative weight, while keeping all other restrictions in place.

Type	Cl.err	AUC
ALL	0.78 (0.22)	0.40 (0.15)
AML	0.43 (0.18)	0.53 (0.14)
KIRC	0.17 (0.11)	0.91 (0.08)
KIRP	0.27 (0.26)	0.83 (0.14)
OV	0.39 (0.28)	0.66 (0.26)
THCA	0.49 (0.17)	0.58 (0.12)

(a) Posweights

Cl.err	AUC
0.20 (0.18)	0.89 (0.08)
0.26 (0.27)	0.84 (0.19)
0.78 (0.00)	0.38 (0.14)
0.70 (0.17)	0.48 (0.19)
0.44 (0.14)	0.63 (0.22)
0.31 (0.24)	0.77 (0.25)

(b) Posweights and reverse-labels

Figure 8: Classification error and AUC for MILES+ttest+mean.

Table 8 contains the classification error and AUC obtained on the MIL setup with positive weights (using 10% of the data selected through a t-test) and the performance obtained when reversing the labels. The results in these tables are rather puzzling.

First, the standard deviation is often very high. Second, the classification errors do not add up like one would expect (AML for example) and third, the classification error and AUC do not match up (ALL for example). This behaviour is observed when more or less data is selected, when other selection methods are used and when parameters are varied (see the tables below for results when changing the fraction, parameter C and reversing the labels).

In the end we have not been able to fix this issue. The extra restriction of allowing only a positive contribution from each mutation is therefore not used in the final report.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.01,C=1	0.88 (0.08)	0.89 (0.0)	0.2 (0.12)	0.4 (0.31)	0.44 (0.22)	0.62 (0.16)
f=0.01,C=1,rev	0.25 (0.21)	0.27 (0.27)	0.78 (0.0)	0.7 (0.17)	0.55 (0.17)	0.34 (0.24)
f=0.01,C=10000	0.83 (0.18)	0.87 (0.07)	0.19 (0.12)	0.39 (0.31)	0.39 (0.28)	0.58 (0.14)
f=0.01,C=10000,rev	0.25 (0.21)	0.26 (0.26)	0.78 (0.0)	0.7 (0.17)	0.55 (0.17)	0.32 (0.25)

Table 2: Classification error of MIL+ttest with positive weights only, weighing C dynamically by class size and using the existing labelling versus reverse labelling. 1% of the data was used.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.01,C=1	0.37 (0.15)	0.36 (0.14)	0.89 (0.1)	0.72 (0.2)	0.62 (0.25)	0.44 (0.08)
f=0.01,C=1,rev	0.87 (0.1)	0.84 (0.17)	0.39 (0.15)	0.48 (0.2)	0.56 (0.23)	0.75 (0.24)
f=0.01,C=10000	0.37 (0.16)	0.36 (0.16)	0.9 (0.09)	0.72 (0.21)	0.65 (0.27)	0.44 (0.09)
f=0.01,C=10000,rev	0.87 (0.09)	0.84 (0.19)	0.36 (0.17)	0.47 (0.21)	0.56 (0.23)	0.76 (0.26)

Table 3: AUC of MIL+ttest with positive weights only, weighing C dynamically by class size and using the existing labelling versus reverse labelling. 1% of the data was used. Between brackets is the standard deviation.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.10,C=1	0.88 (0.08)	0.42 (0.16)	0.18 (0.11)	0.28 (0.24)	0.42 (0.24)	0.49 (0.17)
f=0.10,C=1,rev	0.22 (0.18)	0.26 (0.26)	0.78 (0.0)	0.7 (0.17)	0.45 (0.14)	0.32 (0.24)
f=0.10,C=10000	0.78 (0.22)	0.43 (0.18)	0.17 (0.11)	0.27 (0.26)	0.39 (0.28)	0.49 (0.17)
f=0.10,C=10000,rev	0.2 (0.18)	0.26 (0.27)	0.78 (0.0)	0.7 (0.17)	0.44 (0.14)	0.31 (0.24)

Table 4: Classification error of MIL+ttest with positive weights only, weighing C dynamically by class size and using the existing labelling versus reverse labelling. 10% of the data was used. Between brackets is the standard deviation.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.10,C=1	0.4 (0.15)	0.52 (0.15)	0.91 (0.08)	0.83 (0.13)	0.64 (0.25)	0.58 (0.12)
f=0.10,C=1,rev	0.89 (0.08)	0.84 (0.19)	0.47 (0.04)	0.48 (0.19)	0.63 (0.22)	0.77 (0.23)
f=0.10,C=10000	0.4 (0.15)	0.53 (0.14)	0.91 (0.08)	0.83 (0.14)	0.66 (0.26)	0.58 (0.12)
f=0.10,C=10000,rev	0.89 (0.08)	0.84 (0.19)	0.38 (0.14)	0.48 (0.19)	0.63 (0.22)	0.77 (0.25)

Table 5: AUC of MIL+ttest with positive weights only, weighing C dynamically by class size and using the existing labelling versus reverse labelling. 10% of the data was used. Between brackets is the standard deviation.

5.5 Hierarchical classification

In this experiment we replace the one-versus-all setup that creates a single classifier per class by a hierarchical classifier that classifies each class separately and then removes it from the dataset. It consists of five rounds of 10-fold cross-validation. After each round the class with the lowest classification error is selected for removal.

Table 6 shows the classification error, with underlined is the best performance. First KIRP is removed, then AML, followed by ALL and KIRC. If this method indeed works we would expect to see the lowest classification error for each class right before it is removed. But the expected location for the best performance holds only for class KIRP.

It appears that by removing classes from the dataset important structure is also removed. Because of the removed structure the classifiers perform worse as the rounds progress. We therefore did not continue with this type of experiment.

Type	label	Round 1	Round 2	Round 3	Round 4	Round 5
ALL	0	**0.17**	0.18	0.20		
AML	1	**0.17**	0.18			
KIRC	12	0.22	0.22	**0.20**	0.20	
KIRP	13	**0.13**				
OV	23	0.35	0.34	0.32	**0.31**	0.32
THCA	32	**0.24**	0.25	0.28	0.26	0.33

Table 6: Classification error from the hierarchical classifier test while selecting 1% of observations through the t-test rank.

5.6 Different distance

In order to calculate the distance between a bag B and an observation x MILES first calculates all pairwise distances between the observation in question and all observations in the bag. The minimum of these pairwise distances is then used to represent the distance between the observation and the bag.

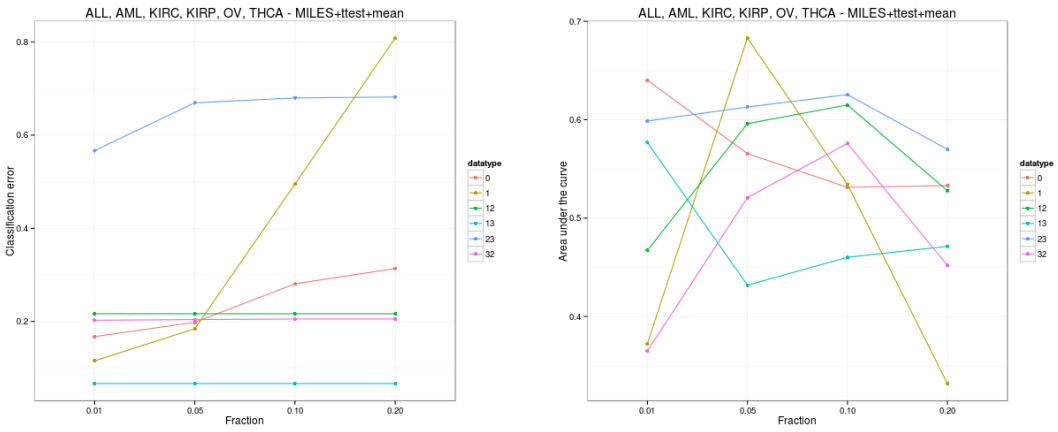
$$D(\mathbf{x}, B) = \min\{D(\mathbf{x}, \mathbf{x}_1), D(\mathbf{x}, \mathbf{x}_2), \ldots, D(\mathbf{x}, \mathbf{x}_n)\} \tag{1}$$

We have performed an experiment where we replace taking the minimum distance with the mean distance:

$$D(\mathbf{x}, B) = \frac{\sum_{i=1}^{n} D(\mathbf{x}, \mathbf{x}_i)}{n} \tag{2}$$

Experiments were run with 1%, 5%, 10% and 20% of the data selected trough a t-test. Figure 9 shows the classification error and AUC. The area under the curve for each of the experiments immediately shows that this adaptation of MILES does not yield any reliable results.

This could be because instances in a bag are scattered out, meaning that many bags overlap. Taking the mean distance puts many bags in close proximity which makes it difficult to separate the positive from the negative bags. We opted to not pursue this direction any further.



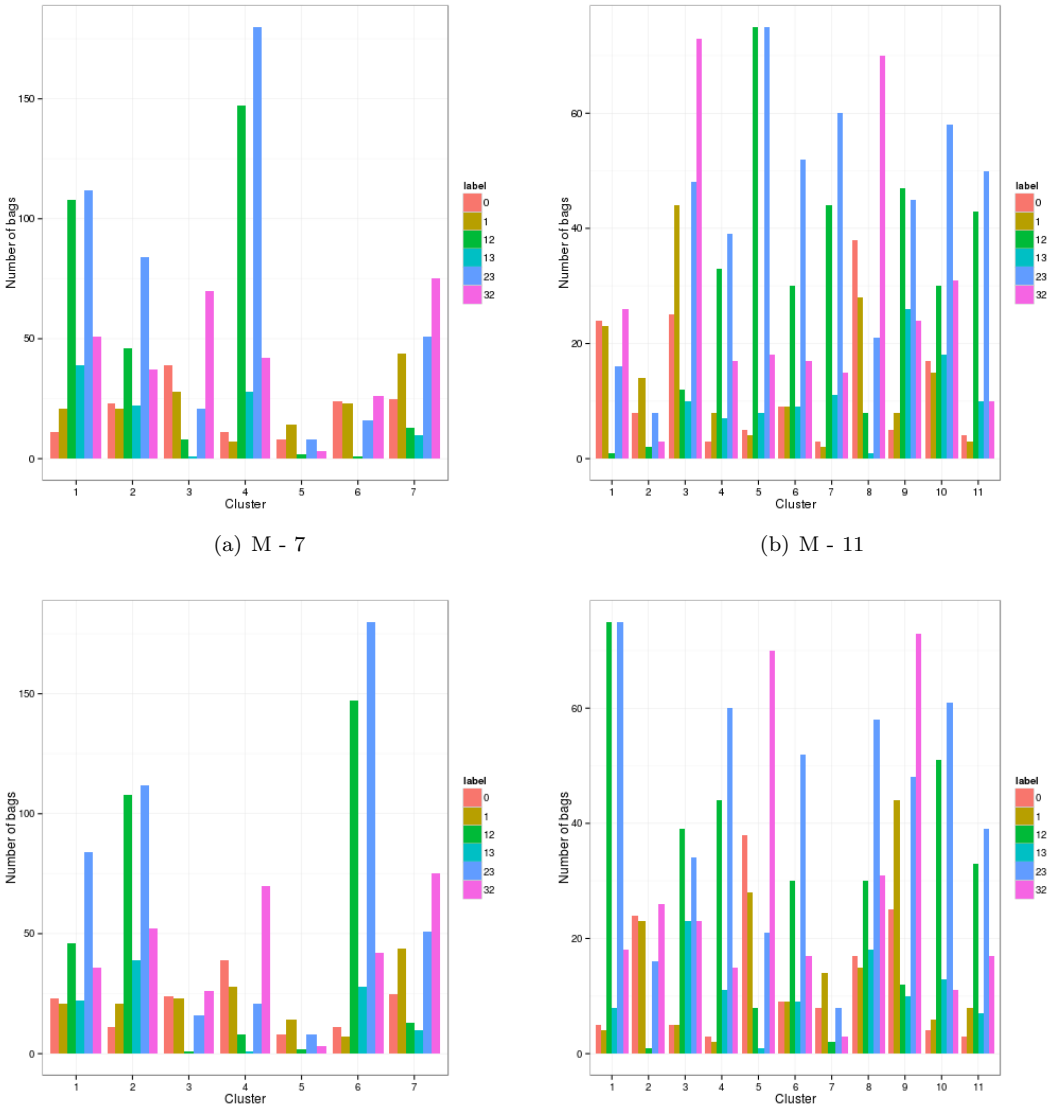
(a) (b)

Figure 9: Classification error and AUC for MILES+ttest+mean.

6 Clustering

In order to investigate possible structure in the internal distance matrix M of the MIL classifier we've performed hierarchical clustering on the bags. First complete-linkage agglomerative hierarchical clustering on M directly and then single-linkage agglomerative hierarchical clustering on a matrix that contains the euclidean distance between each pair of bags. This report doesn't lend itself too well for showing a dendrogram of 1500 items. Both dendrograms show blocks of bags with the same label clustered together, but the blocks are largely small.

We've also performed k-means clustering for $k = 7$ and $k = 11$ on both matrices using a scree plot. Figure 10 shows the contents of the different clusters obtained. There are no pure clusters, although in most clusters at least one class is not very well represented. We conclude from these experiments that there appears to be some structure in M, but it is unclear how to extract it.



(a) M - 7

(b) M - 11

(c) Eucl - 7

(d) Eucl - 11

Figure 10: Contents of the clusters obtained through k-means clustering.

7 1-vs-1 experiments

We have performed experiments in which a classifier is trained for each pair of classes in order to elucidate some of the structure in the six class dataset. The tables below show the results.

AUC:

- ALL and AML have low AUC.
- KIRC and KIRP have low AUC.
- OV medium AUC to all other classes.
- THCA low AUC to ALL and AML, medium to OV and high to others.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.44 (0.12)	0.88 (0.05)	0.97 (0.03)	0.70 (0.10)	0.64 (0.11)
AML	0.44 (0.12)	0	0.90 (0.08)	0.96 (0.03)	0.73 (0.06)	0.62 (0.07)
KIRC	0.88 (0.05)	0.90 (0.08)	0	0.62 (0.11)	0.81 (0.05)	0.90 (0.09)
KIRP	0.97 (0.03)	0.96 (0.03)	0.63 (0.11)	0	0.79 (0.10)	0.94 (0.04)
OV	0.70 (0.09)	0.71 (0.07)	0.81 (0.05)	0.78 (0.09)	0	0.76 (0.08)
THCA	0.62 (0.11)	0.64 (0.09)	0.90 (0.09)	0.94 (0.05)	0.76 (0.08)	0

Table 7: AUC of the 1-vs-1 experiments using the t-test selection method when 10% of the observations are selected.

Classification error positive class: When looking at the classification error obtained on the positive class we observe a similar pattern. ALL/AML and KIRC/KIRP are tied together, while OV performs badly in any case. THCA now only performs well when compared to KIRP, but does badly otherwise.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.59 (0.12)	0.26 (0.13)	0.36 (0.15)	0.32 (0.10)	0.41 (0.11)
AML	0.61 (0.10)	0	0.22 (0.14)	0.29 (0.17)	0.33 (0.09)	0.39 (0.09)
KIRC	0.55 (0.13)	0.39 (0.09)	0	0.93 (0.08)	0.45 (0.09)	0.39 (0.07)
KIRP	0.19 (0.09)	0.15 (0.10)	0.52 (0.09)	0	0.33 (0.09)	0.16 (0.09)
OV	0.69 (0.17)	0.68 (0.07)	0.56 (0.11)	0.85 (0.12)	0	0.56 (0.07)
THCA	0.60 (0.19)	0.55 (0.17)	0.38 (0.16)	0.42 (0.21)	0.45 (0.08)	0

Table 8: Classification error obtained on the positive class of the 1-vs-1 experiments, using the t-test selection method when 10% of the observations are selected.

Classification error negative class: When looking at the classification error obtained on the negative class we again see poor performance on ALL/AML and KIRC-vs-KIRP. Interestingly KIRP-vs-KIRP goes very well. THCA is difficult to seprate from ALL and AML, but does well when matched with the other classes.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.50 (0.15)	0.17 (0.08)	0.03 (0.05)	0.40 (0.13)	0.37 (0.18)
AML	0.48 (0.12)	0	0.13 (0.10)	0.02 (0.03)	0.34 (0.07)	0.44 (0.09)
KIRC	0.05 (0.04)	0.08 (0.11)	0	0.07 (0.09)	0.13 (0.11)	0.07 (0.12)
KIRP	0.01 (0.03)	0.08 (0.09)	0.30 (0.12)	0	0.22 (0.15)	0.12 (0.13)
OV	0.08 (0.03)	0.12 (0.04)	0.07 (0.05)	0.02 (0.02)	0	0.11 (0.04)
THCA	0.23 (0.09)	0.28 (0.07)	0.07 (0.07)	0.05 (0.04)	0.19 (0.08)	0

Table 9: Classification error obtained on the negative class of the 1-vs-1 experiments, using the t-test selection method when 10% of the observations are selected.

Conclusions: In general the results paint a picture of what the feature space looks like. The venn diagram in figure 11 approximates that picture roughly. ALL and AML and KIRC and KIRP are difficult to separate from eachother, but ALL/AML and KIRC/KIRP separate quite well. OV and THCA are reasonably well separable from all other classes.

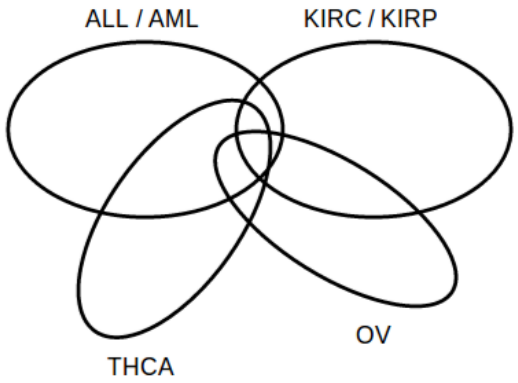


Figure 11: An approximation of the feature space, based on the 1-vs-1 experiments.

8 Biological interpretation

The biological interpretation figures have gone through a number of updates during this project. But essentially they are showing the same information. Apart from a basic inventory of the properties of the selected mutations there hasn't been much room for interpretation. We've opted mostly to develop the MIL method as much as possible.

Below we show two variants of the interpretation figures that show the information as the heatmap in the main paper. All three figures contain the selection from the same MIL run and therefore show the development process. For brevity purposes we only include figures for the combination of Func and Exonicfunc feature groups, which is also shown as heatmap in the main paper.

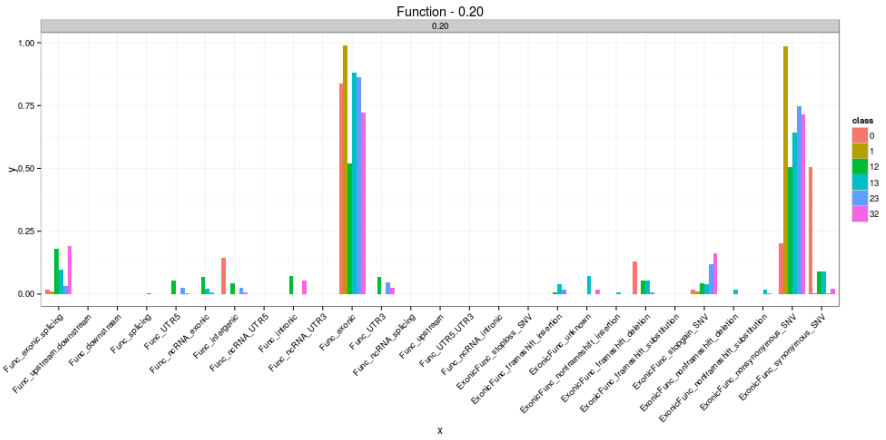


Figure 12: First generation figures that show the fraction of selected mutations with a particular property. There are six bars per property, representing the six classifiers (one for each class) that are created by the MIL method. The problem with these figures is that they do not show whether the fractions are merely the result of chance or an actual selection.

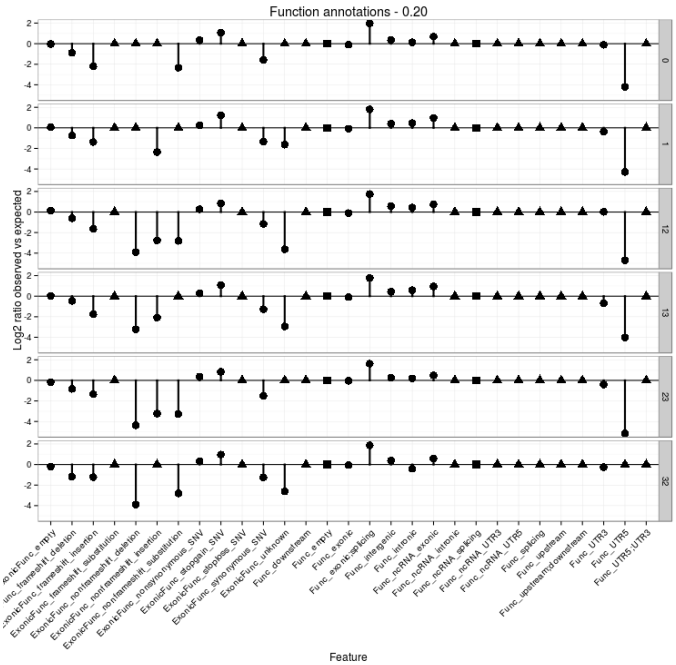


Figure 13: The second generation shows a $\log_2$ of the observed-vs-expected ratio. Each bar therefore represents the fold change within the selection as compared to the overall abundance of mutations with a property. Each classifier has its own horizontal bar. A dot means a regular datapoint, a triangle represents the case where a mutation could have been selected with that property, but it didn't occur. Finally, a square represents the case where no mutation exists with that property, therefore none could be selected. The main problem of these figures is that they remain very difficult to interpret. Also, each classifier selects from both the positive and negative classes. These figures do not show whether there is a clear difference in selection between these classes per classifier. We therefore developed the heatmap figures of which one is shown in the main paper.

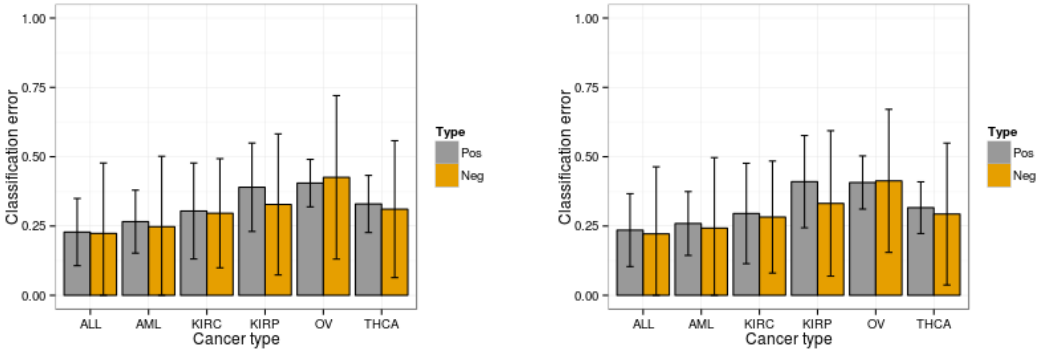
9 Reducing the number of features

During this project we've suspected that some features do not have any added value. In this section we investigate the importance of groups of features. Features are grouped by their type of annotation: Function (intronic, exonic, etc), exonicfunction (synonymous, stopgain, etc), mutation (reference and alternative base) and variant effect predictors. Given these four groups we devise four experiments by removing each group from the dataset. Note that we have removed the chromosomal location and known mutation database features for all experiments.

In order to perform these experiments quickly, but still with a reasonable amount of data we opted to use the MILES+Lasso implementation. First we removed a group of features from the dataset and then performed a PCA. The number of principal components was chosen such that the resulting dataset covers roughly 98% of the variation contained in the original dataset. Then the distance matrix M was calculated. A classifier was then trained by selecting 10% of the instances by means of a t-test, followed by a regression that estimates weights for each selection. The weights are estimated through a version of Lasso which takes the class size (in terms of bags) into account.

Figures 14 and 15 show the classification error obtained for both the positive and negative classes. The no-VEP experiment yields the best results across all the classes, as a matter effect both classification error and AUC improved when compared to using the whole dataset (data not shown). Meanwhile removing the mutation group yields the highest classification error. We therefore conclude that the VEP group is not adding any value and should be removed, but the mutation group is of importance. The func and exonicfunc groups have a very similar effect. Performance is not really affected by their removal. Followup experiments where more groups were removed incrementally showed that func and exoncfunc do have added value, but very minimal (data not shown). We therefore opt to keep three groups.

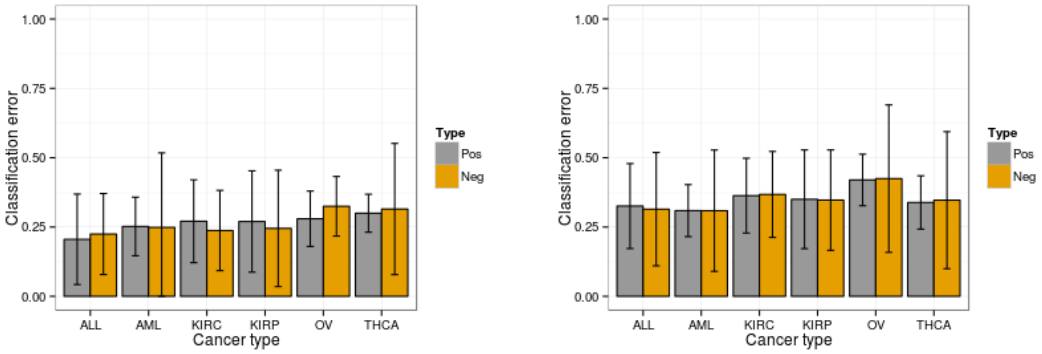
We then ran the MILES++ setup through the L1-SVM classifier to properly test the effect of this reduced dataset. Figure 16 shows that the classification error on the positive class goes down across the board, while the classification error on the negative class goes up. On average removing the VEP group and reducing the resulting features down to its first 15 principal components results in a performance gain. One could speculate that a more robust classifier is obtained.



(a) No Func

(b) No Exonicfunc

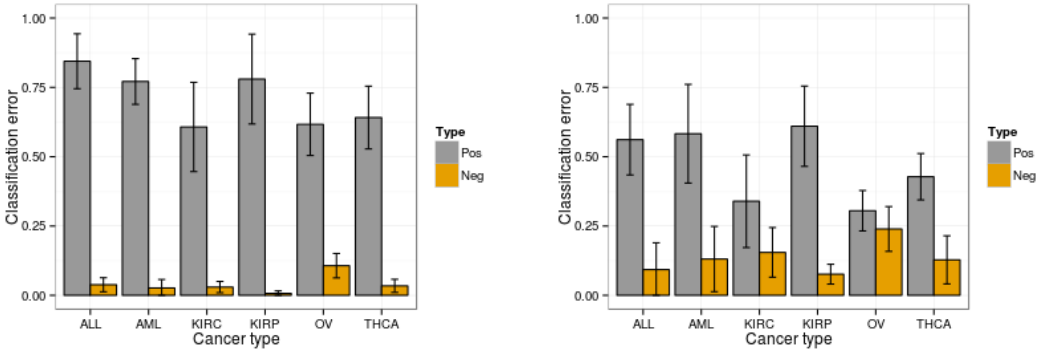
Figure 14: Classification error obtained when removing one of the four groups of features using the Lasso MIL setup.



(a) No VEP

(b) No Mut

Figure 15: Classification error obtained when removing one of the four groups of features using the Lasso MIL setup.



(a) All features

(b) No VEP

Figure 16: Classification error with all features and with the VEP group removed as well as the chromosome location, known mutation database membership features and a reduction to the first 15 principal components.

10 Adding pathway features

The six class dataset has been extended with naive pathway features. First mutations have been mapped into Ensembl gene IDs. Then each gene ID is mapped into a reactome pathway ID. Each available pathway ID is added as a new binary feature to the dataset. A field is 1 when a mutation falls within a gene in a particular pathway, 0 otherwise. This annotation is naive as this considers pathways to be independent cellular processes, when in fact they are not.

When comparing classification error and AUC with the results on the 10% experiments run earlier there is no real difference. Some classifiers perform slightly better, others are slightly worse. There doesn't seem to be a big advantage in adding this type of features to our dataset with the current setup.

Type	**MILES+selection**	**+posweights**	**+reverse**
ALL	0.12 (0.04)	0.21 (0.18)	0.83 (0.18)
AML	0.16 (0.07)	0.26 (0.26)	0.40 (0.16)
KIRC	0.18 (0.07)	0.73 (0.18)	0.16 (0.11)
KIRP	0.09 (0.04)	0.70 (0.17)	0.26 (0.25)
OV	0.29 (0.07)	0.44 (0.13)	0.41 (0.21)
THCA	0.17 (0.05)	0.30 (0.24)	0.49 (0.17)

Table 10: Classification error of MILES+ttest, MILES+ttest+posweights and MILES+ttest+posweights+reverse for the six class problem extended with naive pathway annotations. In each case 10% of the data is selected.

Type	**MILES+selection**	**+posweights**	**+reverse**
ALL	0.83 (0.08)	0.90 (0.06)	0.31 (0.15)
AML	0.77 (0.09)	0.86 (0.17)	0.53 (0.17)
KIRC	0.85 (0.07)	0.42 (0.12)	0.91 (0.08)
KIRP	0.84 (0.09)	0.45 (0.20)	0.83 (0.14)
OV	0.74 (0.08)	0.64 (0.21)	0.67 (0.20)
THCA	0.85 (0.06)	0.78 (0.23)	0.59 (0.12)

Table 11: AUC of MILES+ttest, MILES+ttest+posweights and MILES+ttest+posweights+reverse for the six class problem extended with naive pathway annotations. In each case 10% of the data is selected.

11 Selecting a larger fraction of observations

Here we show the distribution of values assigned to mutations by the t-test and ratio ranking methods. Figure 18 contains the distributions of the p-values obtained for each of the six classes. Clearly many mutations have a very low p-value. After zooming in (not shown) it becomes clear that almost all mutations have more or less the same p-value. This means that the ranking is not capable of distinguishing between mutations, which could be the result of the ranking itself or the features.

We plot the same information for the ratio ranking (figure 17). A ratio ranking is obtained for each cancer type individually (rows), with each ranking assigning values to mutations from all classes (columns). Overall the distributions a more spread out, when compared to the t-test. But the distributions are more-or-less the same row-wise.

For both rankings it holds that selecting a larger fraction of instances will result in more positive and negative observations. To such an extend that the impact of the positive observations is more and more diluted. For the t-test we could select all positive observations and then a fraction of the negative. For the ratio test we could select a fraction from the left hand of the distribution of the positive class and a fraction from the left hand of the negative classes. But this would go against our idea of allowing only a positive contribution by each observation.

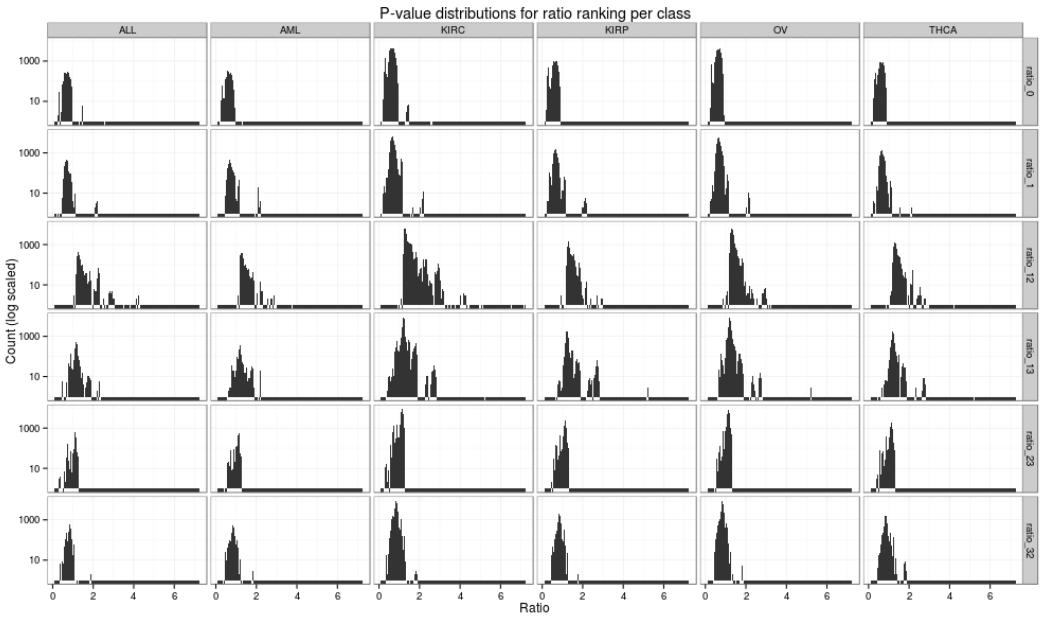


Figure 17: Distributions obtained after ratio ranking.

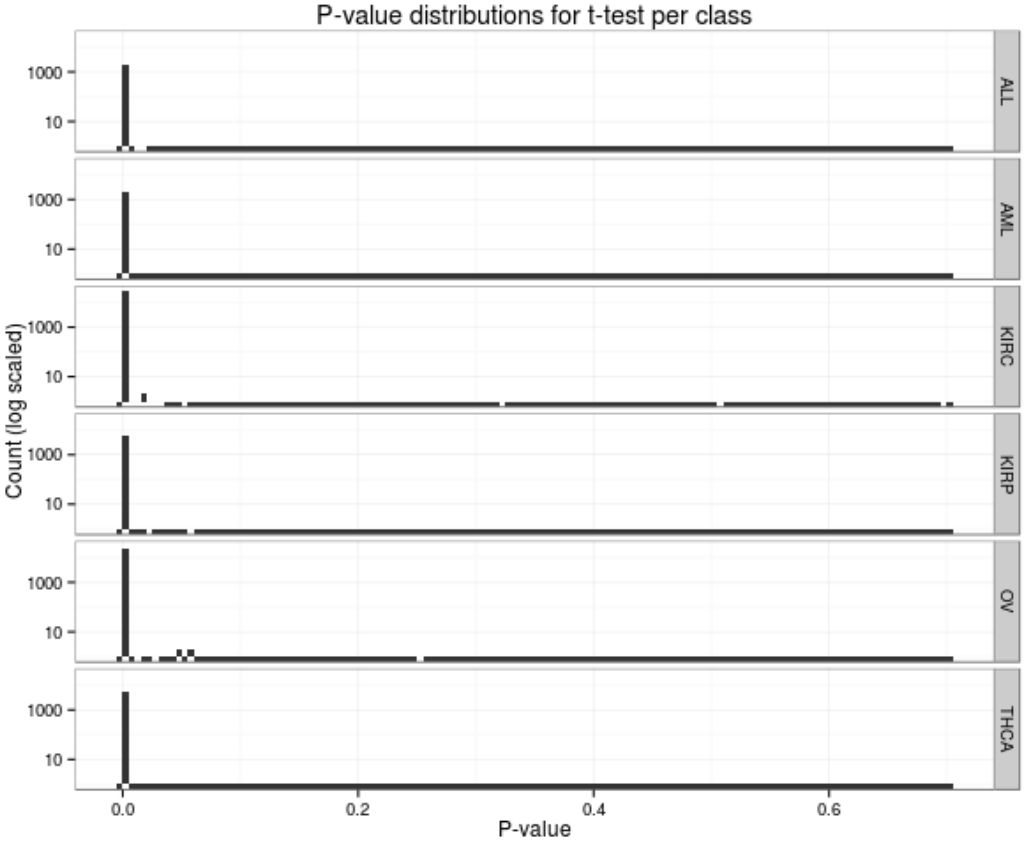


Figure 18: Distributions obtained after t-test ranking.

12 Reimplementation in Python

During the course of this project we've ran into compute and memory requirements that could not be met by the TU Delft compute platforms. We have access to a large compute cluster at the Wellcome Trust Sanger Institute, but Matlab is not available on it. We've therefore explored a possible reimplementation of the various methods in Python using the Sklearn machine learning package.

Currently the gene scoring experiments are implemented and yielded similar results on the pilot data. A MILES implementation exists that works well on simple test data, but on more complex data (the ALL versus AML experiment) both classification error and AUC are lacking. It appears that there is a bug in the Python code that is triggered by this more complicated problem. There is no final implementation of the distance experiment.

These reimplementations would have been great to have, but in the end they turn out to be a lot of work. With well working Matlab implementations the time spend on reimplementation would have limited the number of experiments to advance the project. We've therefore invested in experiments first and spend time in reimplementations sporadically. They are therefore left unfinished at the end of this project.

Master Thesis - Supplement 2

Stefan Dentro

November 29, 2013

This document contains all weekly reports created in during the master thesis project. A weekly report was created nearly every week, It contains a description of the progress made during the last week, including methods and results, and a plan the following week. The information below, combined with the final report and the summary represent the full body of work.

1 Getting started

Goal This week the goal is to write the required research proposal. A start will be made with creating an inventory of the available data and start any preprocessing that is still required.

Results Our dataset consists of three parts. Two parts are acquired from the Cancer Genome Project (CGP) and consist of somatic mutations obtained from a number of large sequencing projects (please see the research proposal for some background). The two parts are mutations obtained from exomes and mutations obtained from genomes. All this data is readily available.

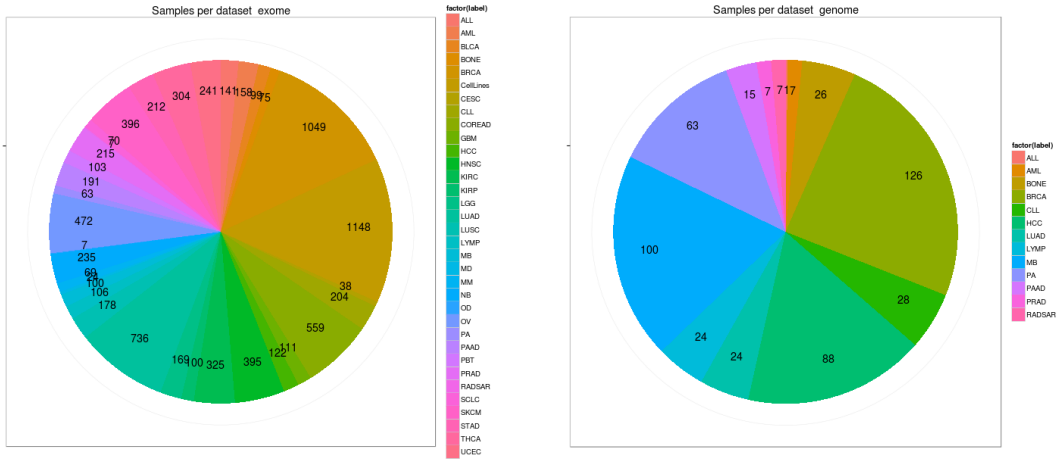
The third part consists of samples that we have obtained access to. There is a yet-to-be-defined overlap between the samples in this dataset and the ones obtained from CGP. All samples will be processed with the same analysis pipelines and equal parameters. Analysis pipelines are running and will be for some time. In the meantime we can start the project with both CGP datasets.

There is quite a bit of variability in the number of mutations per cancer type. Figures 1(a) and (b) show the average counts for both CGP exome and genome. In CGP exome the maximum is around 700 in SKCM and UCEC, while MD, PA and PBT hardly have any mutations at all. CGP genome shows a vast number of mutations on average in LUAD, while AML and PA have very little. It is impossible to establish whether this is a correct measurement or it is due to differences in sequencing.

Figures 3 and 4 show the breakdown of total individual mutations per sample. Each boxplot represents a mutation type and the y-axis is on log-scale. With an alphabet of {A,C,T,G} 12 transitions/transversions are possible. Com-

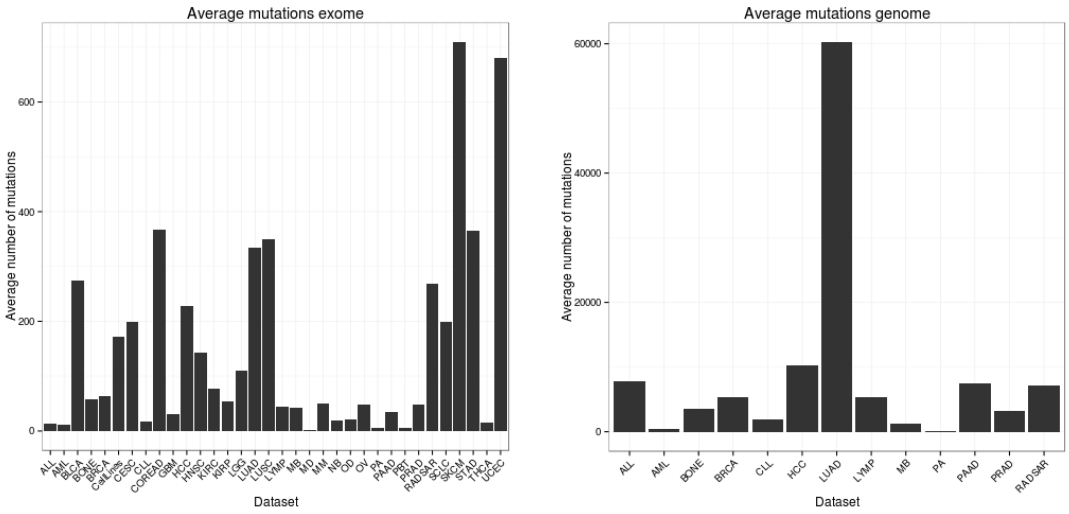
bined with small insertions and deletions that adds up to 14 possible mutations, which are on the x-axis of both figures.

Some cancer types show a lot of variability between samples but also per mutation type. BRCA (top right in figure 3) for example contains quite a few outliers. We know that there are multiple BRCA subtypes, which could be a possible explanation. But subtype information is not available. Furthermore, the mean of C>T is much higher than other mutations. No indels are available for a number of cancer types. Including them could introduce a bias.



(a) CGP Exome (b) CGP Genome

Figure 1: Total number of samples per cancer type.



(a) CGP Exome (b) CGP Genome

Figure 2: Average number of mutations per cancer type.

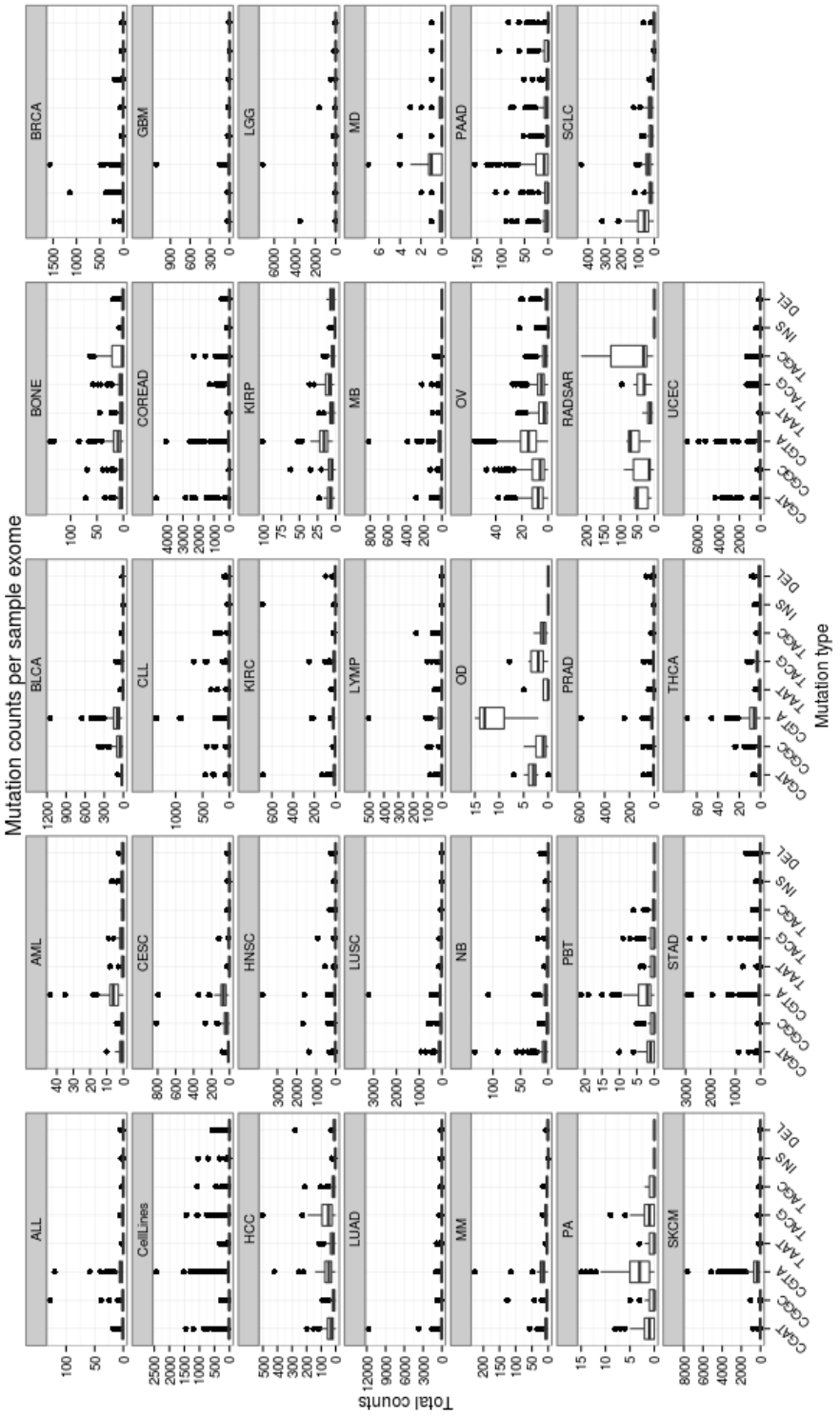


Figure 3: Inventory for each mutation type per sample in the CGP exomes. The x-axis contains the eight available mutations, while the y-axis contains the total count.

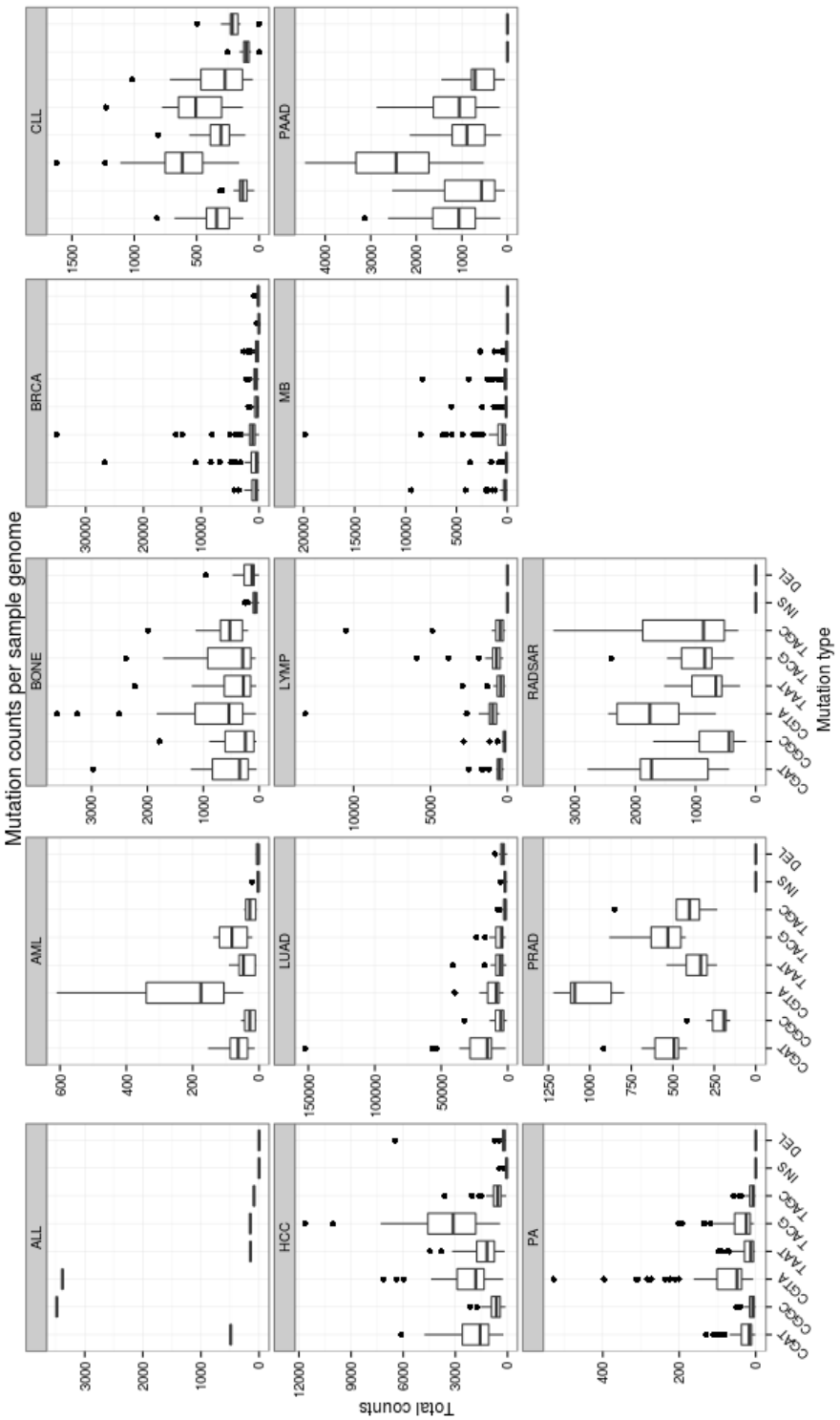


Figure 4: Inventory for each mutation type per sample in the CGP genomes. The x-axis contains the eight available mutations, while the y-axis contains the total count.

Next week Last week in Cambridge. The following list of tasks need attention.

- A planning for this project will be made.
- The VEP annotation script is being difficult. In some cases it starts, but often it doesn't.
- Both Seattleseq and VEP can annotate a mutation multiple times. Merging of these annotations is required as we only want a mutation to be represented once.
- Overall cleanup and reorganisation of the data.

2 Exploring data, adding annotations

Goal This week possible ENCODE annotations will be explored. Furthermore, the quality of VEP annotation must be asserted. Finally, a start is made on creating the literature survey presentation on April 25th.

Results

Mapping annotations A start has been made to annotate the mutations with ENCODE information. The list now includes:

- DNAse cluster locations.
- Combined genome segmentation with respect to histones. One annotation for six different cell types.
- Mapability scores, for windows of 24, 36, 40, 50, 75 and 100 MER.
- Transcription factor binding sites.

Annotation has not been completed due to bugs in the *bedops* software. A fix using *bedtools* is in the works.

The VEP annotation is stopped. The information added is also contained within Annovar annotation. The advantage of Annovar is that it annotates for a single transcript. VEP annotates for all possible transcripts, which then requires a transcript selection step.

A small update to the Annovar annotations is possible. There exists a setting that annotates mutations close to the end of exons/introns as splice-site mutations. Seattleseq annotation still requires inventarisation.

It will be interesting to explore the mutations in terms of their annotations, once merging Annovar and ENCODE is completed. A set of simple count-based figures and perhaps some correlations could be generated. Extreme variability should ideally be explained, as it could be a potential error or bias in the data.

Cancer biology This report contained figures last week that didn't make sense. Figures 3 and 4 have been updated. A boxplot on log-scale is not informative. The y-axis has been fixed, but in order to keep the figure interpretable the y-axis had to be released. This results in a separate y-axis for each data set.

Many datasets contain a relatively large number of C>T and often G>A mutations. This is the result of cytosine deaminase, where a C transforms into a U. During DNA replication the U will then be paired with an A, resulting in a base substitution. Cytosine deaminase has been linked to several types of cancer, so this variability in our data set is not unexpected [2]. The variability within data sets for these particular mutations is unexplained though.

SNPs are in literature often marked down as a six-tuple: C:G>A:T, C:G>G:C, C:G>T:A, T:A>A:T, T:A>C:G and T:A>G:C [1]. This removes some of the redundancy, as C>A is the same as G>T on the other strand. Figures 3 and 4 should be updated to account for these six, replacing the current 12.

General remarks

- Played around with the MIL toolkit for Matlab. Provided example does not work out of the box. Doesn't look like a good option at the moment.
- Created presentation literature survey.
- Moved pipeline infrastructure to striped directory.

Next week Start in Delft. A meeting will probably add a few points to what is mentioned below.

- Create list that maps data set abbreviation to cancer type.
- Run the adapted Annovar setup.
- Fix ENCODE annotation and run it on the samples.
- Merge the different annotations.

3 Back in Delft

Goal First week back in Delft. This week will consist largely of things in the category "getting started". Apart from meetings and settling down, the improved Annovar annotations should be finished and can be loaded into Matlab. The abbreviations used to denote the different types of cancer are explained. Finally, a presentation will be given about the literature survey which will include a brief introduction to this project.

Results

Project direction A meeting took place this week in which the direction of the project was discussed. We've decided to go with multiple-instance learning (MIL) for now. It is unclear how well MIL can handle the heterogeneity of the data.

In order to quickly get some results out we're going to do some tests with just the minimal Annovar annotations and the MIL toolkit developed by David Tax. The toolkit is installed and works with example data. The next step is to transform our data in such a way that it can be used. For now CGP-WES (exomes, but without the cell lines) and CGP-WGS (whole genomes) are the two data sets that are ready. Perhaps a subset of both should be selected as the cancer labels are sometimes ambiguous.

What to do exactly with the cell lines remains unclear. An initial list of types is attached with this report. Perhaps initially the cell lines with clear labelling could be used as an additional data set. The advantage of cell lines is that any findings could be validated more easily.

Finally, the filtering procedure and the extra steps undertaken to clean the data are to be written down as quickly as possible. The QC should not change and this document will provide an overview of which steps were taken for later reference. The annotation and encoding procedure will be added at a later stage.

Annotations and Matlab Annovar annotations now contain possible splice site altering mutations. Such a mutation is defined as the intron/exon boundary +/-2bp. When it falls within the exon it is marked as *exonic;splicing*, otherwise it will be marked as just *splicing*.

The Annovar output files contain a few inconsistencies, which makes loading them into Matlab difficult. These inconsistencies are fixed for the most part and the data is now in Matlab. The next issue is transforming the categorical features from their string format into a numerical representation. Some features can be easily renamed, but others can contain multiple values in a single field.

Below are a few selected examples that show which features are available at this stage. Note that many features have no value when a mutation is outside of a gene. This problem is more persistent in the genomes than it is within the exomes. How persistent should be explored. We also expect that the dbSNP, 1000G and ESP annotations are empty for all mutations as those should have been filtered out. It should be checked whether this is the case

Func	intronic	intergenic	exonic	exonic	splicing
Gene	C10orf28	LOXL4(dist=38989) PYROXD2(dist=76326)	ACTR1A	RAB11FIP2	FTSJ3
ExonicFunc			nonsynonymous SNV	synonymous SNV	
AAChange			NM_005736: c.C85G:p.Q29E	NM_014904: c.C1398T:p.L466L	
Conserved			629;Name=lod=482	517;Name=lod=169	
SegDup					
ESP5400_ALL					
1000g2012feb_ALL					
dbSNP137					
AVSIFT			0.21		
LJB_PhyloP			0.999575		
LJB_PhyloP_Pred			C		
LJB_SIFT			0.75		
LJB_SIFT_Pred			T		
LJB_PolyPhen2			0.031		
LJB_PolyPhen2_Pred			B		
LJB_LRT			1		
LJB_LRT_Pred			D		
LJB_MutationTaster			0.999904		
LJB_MutationTaster_Pred			D		
LJB_GERP++			5.15		
Chr	10	10	10	10	17
Start	100003305	100066996	104250334	119768650	61904411
End	100003305	100066996	104250334	119768650	61904411
Ref	G	G	G	G	C
Obs	A	A	C	A	T

General remarks

- The final set of alignment files in the ECG-WGS data set are in the improvement pipeline. SNP calling can therefore start in a weeks time. SNP calling should be done in a couple of days, after which the quality filtering procedure will be applied to obtain the final set of mutation calls. These final mutation calls will be frozen, as is the case for both CGP data sets.

- Attached is a list that maps cancer abbreviations to a cancer type and contains an initial inventory of what kind of cell lines are available. It should be noted that the cell lines list accounts a number of samples twice. This needs some attention.

Next week The following list of tasks is on the agenda for next week:

- Encoding of Annovar features in order to import them into the MIL toolkit.

- Further exploration of the MIL toolkit and some additional background paper reading.

- The following list of figures are to be created:
  - Mutation count figures 3 and 4 will be adapted to account for six SNP mutations instead of 12.
  - Number of exonic, intronic and intergenic mutations per cancer type and data set.
  - Number mutations for each of the predicted exonic function annotations per cancer type and data set.
  - Distributions for SIFT, Polyphen and Mutationtaster features.

- It should be checked that no mutation has a dbSNP, 1000G or ESP annotation. Those three databases contain common mutations that we filter out during QC. The three features can be dropped subsequently.

4 More data preprocessing

Goal This week the Annovar annotations should be ready and properly encoded in Matlab. Once available more exploration of the MIL toolkit is possible. The ECG-WGS samples are entering their final phase. SNP calling can be started later this week.

Results

Annovar annotations The week has been mostly filled with trial and error fixing of the annotations. The genome annotation files contain a number of inconsistencies (due to bugs in Annovar) that have all been fixed by now. The main problem was that not all annotations ended up in the right columns. Also, sometimes a SNP was annotated with the same gene twice. As it currently stands 22961 unique genes are mutated in both CGP-WES and CGP-WGS with exonic or intronic mutations, which seems a bit too high still. The reason for this is that some (998) mutations are annotated with multiple genes and are counted separately: *ABCB1;RUNDC3B*.

There are seven variant effect predictors are available. All are represented by a single value, while some have an additional column that contains a letter that corresponds to the estimated impact of the mutation [5, 7].

Name	Range	Damaging
AVSIFT	0:1	Below 0.05
GERP	-12.36:6.18	Above 2 (0 means insufficient data)
LRT	0:1	High value
MutationTaster	0:1	High value
PhyloP	0:1	High value
PolyPhen2	0:1	Above 0.85
SIFT	0:1	Below 0.05

Encoding of both predictor values will be relatively easy. But many mutations will not have an annotation there. For most predictors a default value of -1 should be sufficient. GERP is a special case due to its different range. To remain consistent the default value could be $x_{def} = min(range) - max(range)$, which would result in -18.54. The letters will get a column for each possible instantiation plus an additional column that represents the missing value case.

Of the other columns (see table 3) function, exonic function, chromosome, reference allele and observed allele columns are encoded with separate columns for each possible instantiation. AAChange, ESP5400, 1000g and dbSNP columns will be encoded with either a 0 or a 1 depicting whether a value is in that particular column. That leaves the gene and conserved columns. Conservation is also captured in the variant effect predictors mentioned above. The column will therefore be dropped for now. There is currently no clear alternative for encoding the genes to creating a single column for each of the known human genes. Whether that is a decent solution remains to be seen. Finally, start and end do not need encoding.

Figures Figures 3 and 4 have been updated to only account for six substitution and two indel types. Quite a few data sets still show an elevated C>T and G>A mutation rate as compared to the other categories. The next few pages contain examples of figures created for the different features. Explanation is further down below.

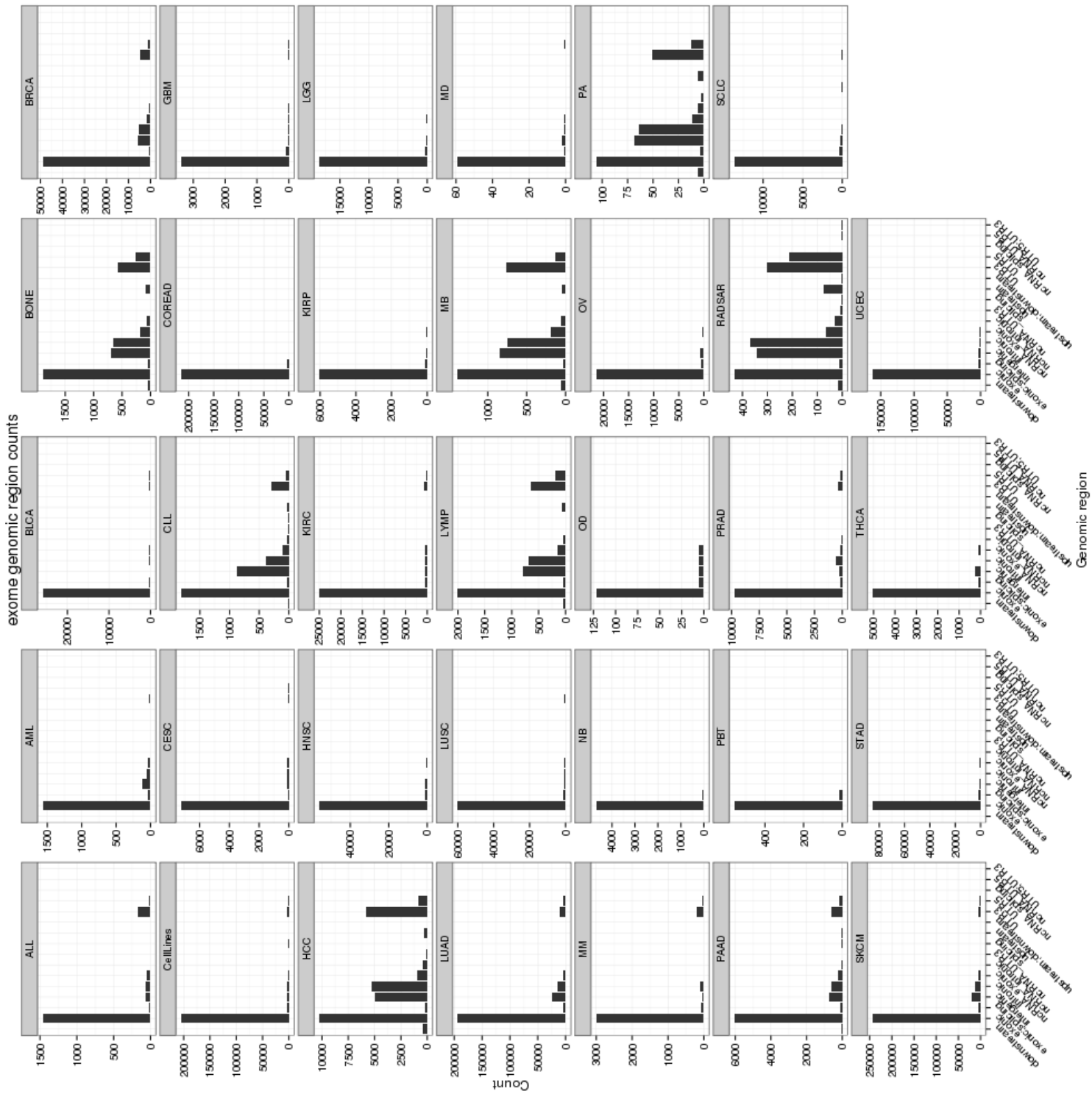

exome genomic region counts
Count
Genomic region
ALL
AML
BLCA
BONE
BRCA
CellLines
CESC
CLL
COREAD
GBM
HCC
HNSC
KIRC
KIRP
LGG
LUAD
LUSC
LYMP
MB
MD
MM
NB
OD
OV
PA
PAAD
PBT
PRAD
RADSAR
SCLC
SKCM
STAD
THCA
UCEC

Figure 5

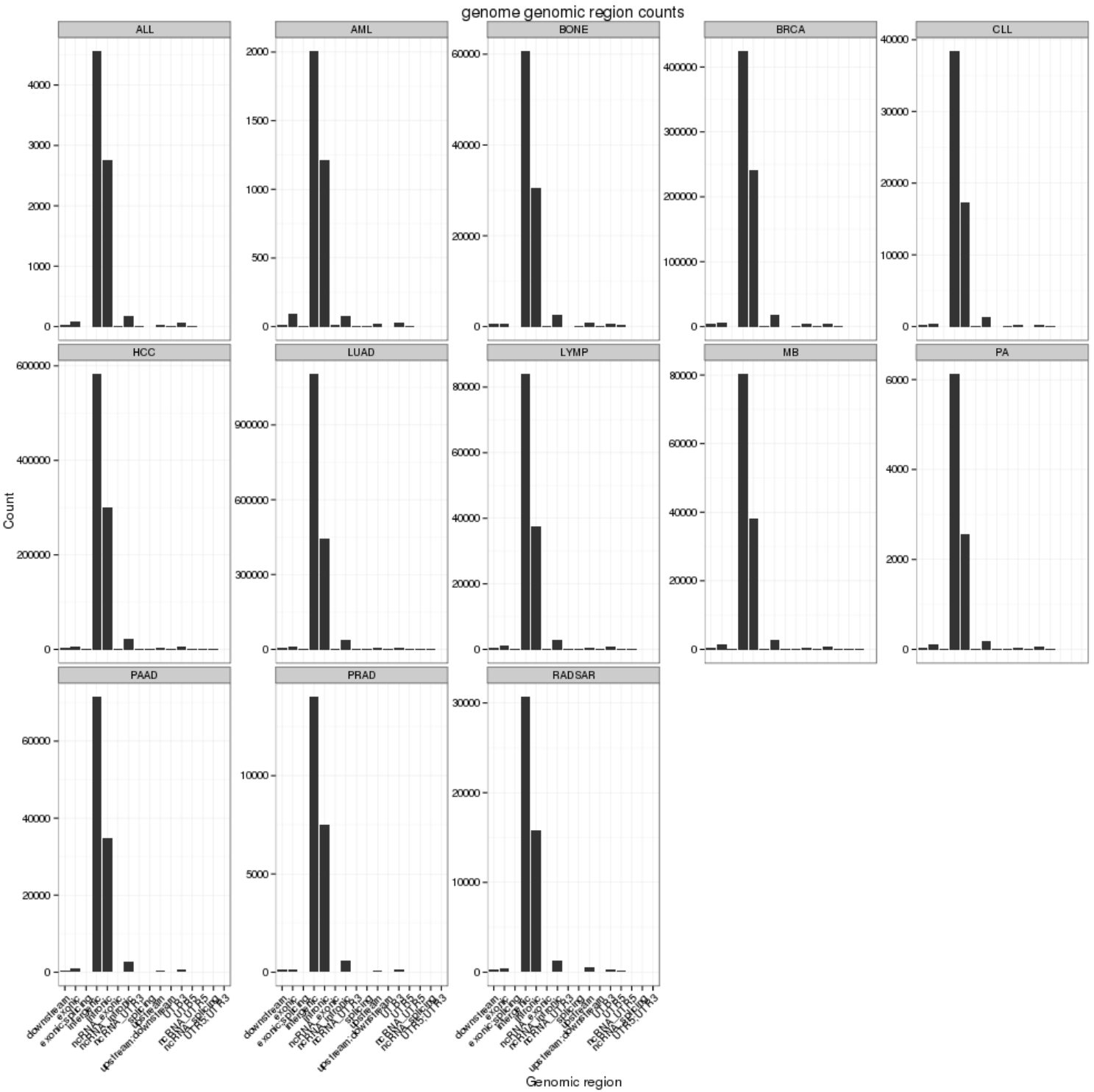

genome genomic region counts
ALL
AML
BONE
BRCA
CLL
HCC
LUAD
LYMP
MB
PA
PAAD
PRAD
RADSAR
Count
Genomic region
downstream
exonic
exonic:splicing
intergenic
intronic
ncRNA_exonic
ncRNA_intronic
ncRNA_UTR3
splicing
upstream
upstream:downstream
UTR3
UTR5
UTR5:UTR3

Figure 6

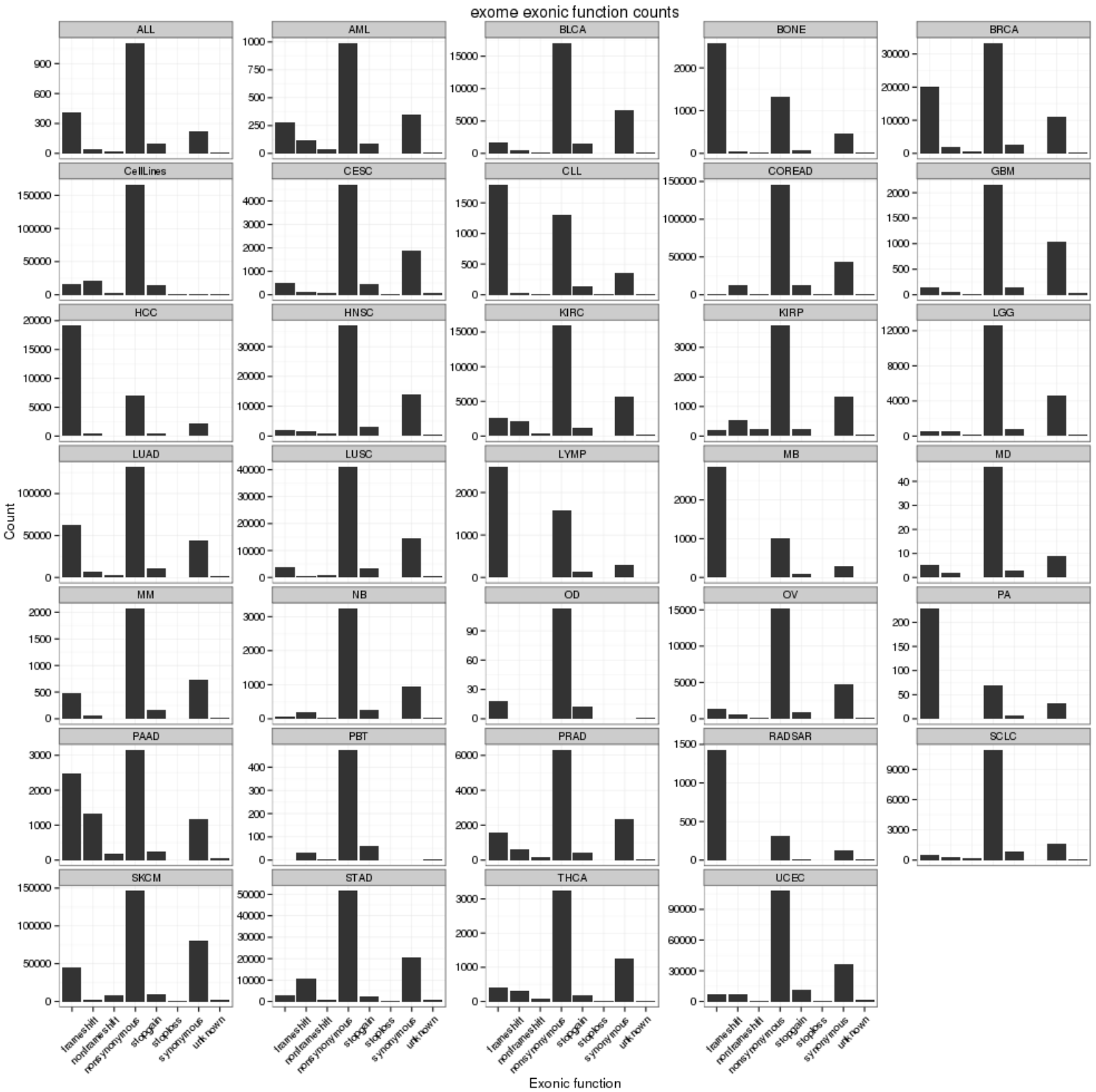

exome exonic function counts
ALL
AML
BLCA
BONE
BRCA
CellLines
CESC
CLL
COREAD
GBM
HCC
HNSC
KIRC
KIRP
LGG
LUAD
LUSC
LYMP
MB
MD
MM
NB
OD
OV
PA
PAAD
PBT
PRAD
RADSAR
SCLC
SKCM
STAD
THCA
UCEC
Count
Exonic function
frameshift
nonframeshift
nonsynonymous
stopgain
stoploss
synonymous
unknown

Figure 7

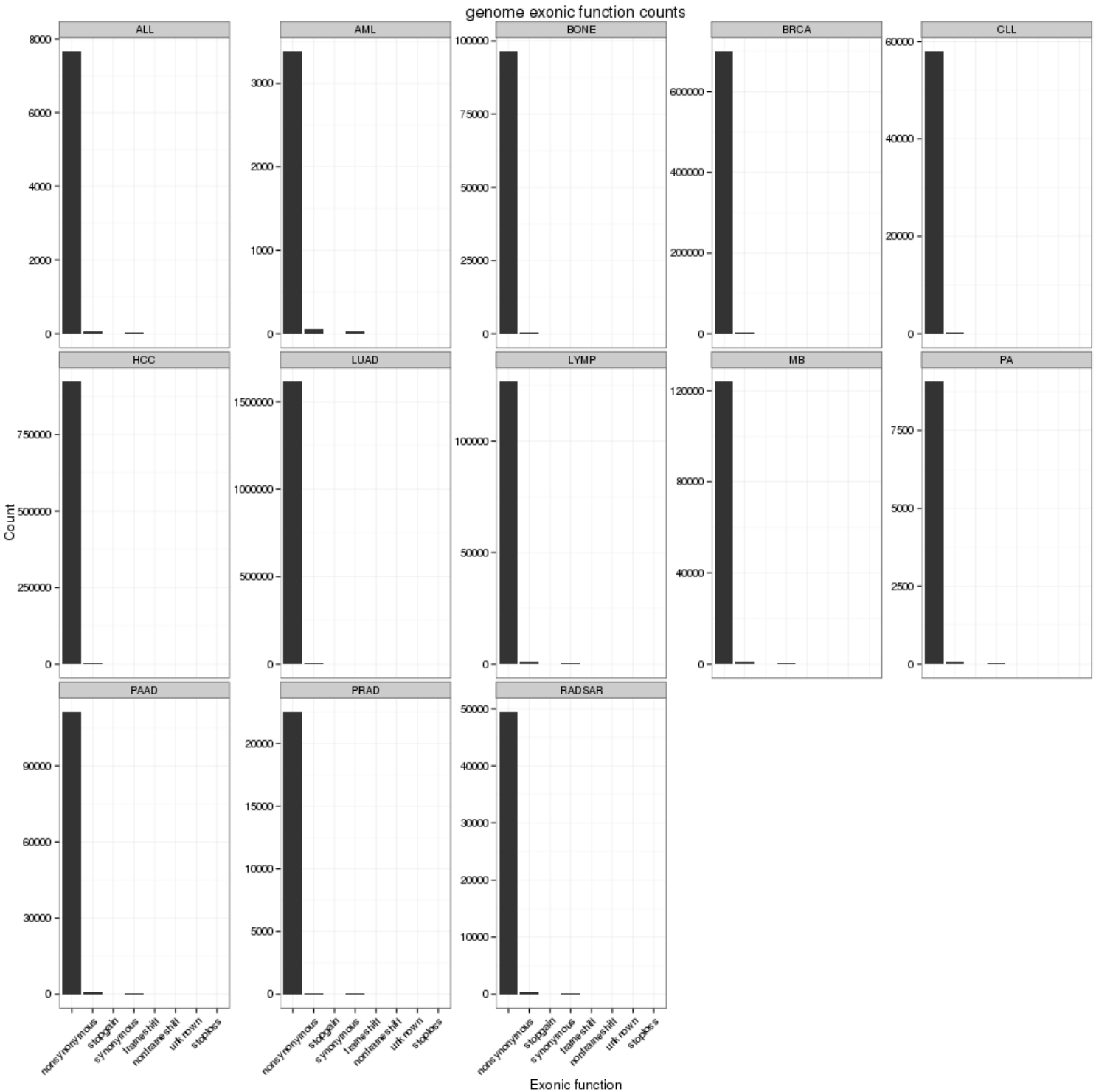

genome exonic function counts
ALL
AML
BONE
BRCA
CLL
HCC
LUAD
LYMP
MB
PA
PAAD
PRAD
RADSAR
Count
Exonic function
nonsynonymous
stopgain
synonymous
frameshift
nonframeshift
unknown
stoploss

Figure 8

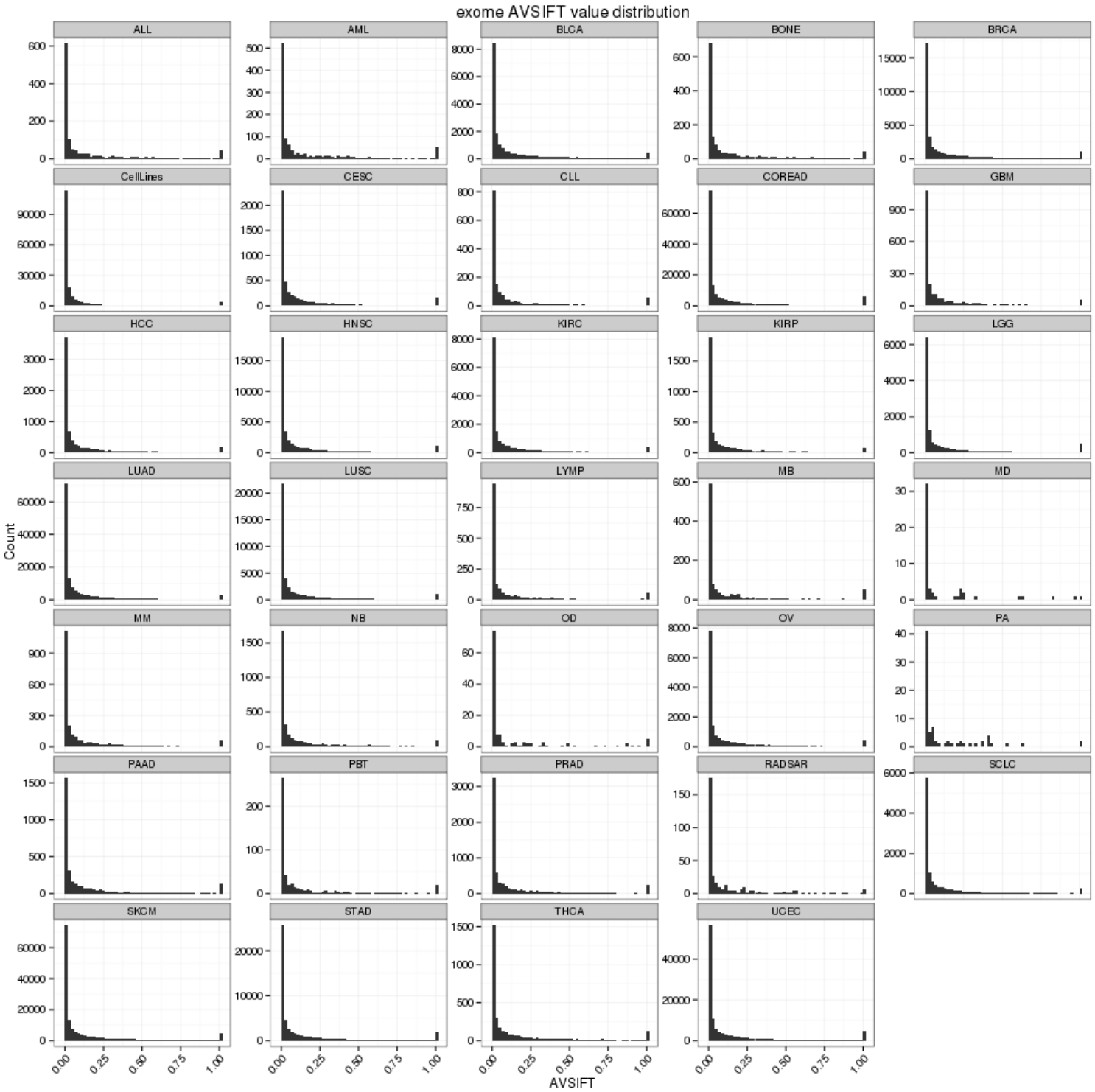

exome AVSIFT value distribution
ALL
AML
BLCA
BONE
BRCA
CellLines
CESC
CLL
COREAD
GBM
HCC
HNSC
KIRC
KIRP
LGG
LUAD
LUSC
LYMP
MB
MD
MM
NB
OD
OV
PA
PAAD
PBT
PRAD
RADSAR
SCLC
SKCM
STAD
THCA
UCEC
Count
AVSIFT
0.00
0.25
0.50
0.75
1.00

Figure 9

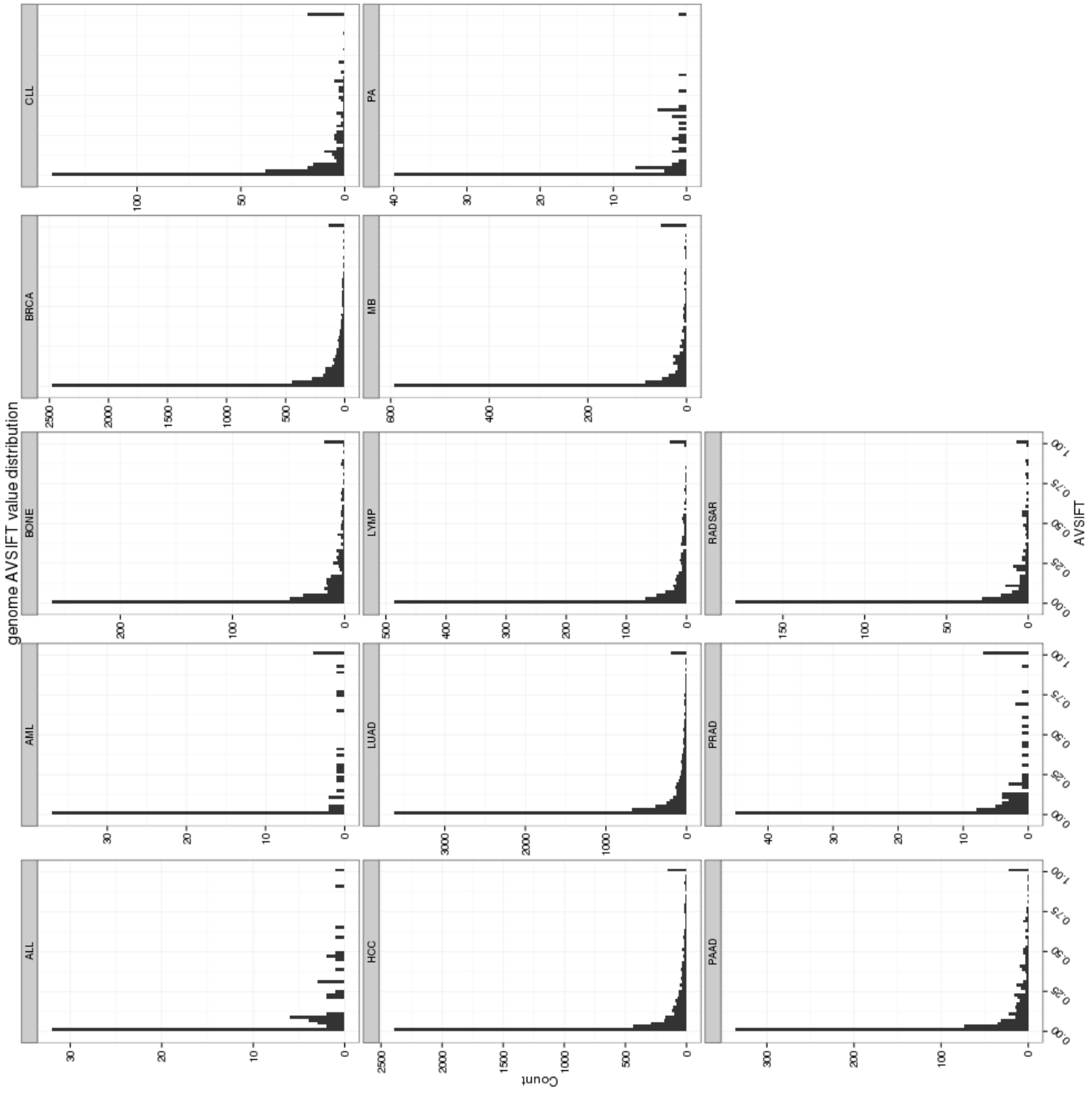

genome AVSIFT value distribution
ALL
AML
BONE
BRCA
CLL
HCC
LUAD
LYMP
MB
PA
PAAD
PRAD
RADSAR
Count
AVSIFT

Figure 10

Figures 5 and 6 show annotations from the first Annovar column. One would expect the CGP-WES exome samples to show mostly exonic mutations, with a small amount of intronic, splicing and intergenic mutations. Most cancer types indeed show such a profile, but some (BONE, MB and RADSAR for example) contain quite a few more intergenic and intronic mutations. A possible explanation is that in some projects whole genes, including introns and an extension into intergenic regions, were captured. A figure could be created to show whether that is the case. The genomes show the expected profile with many intergenic and intronic mutations.

Figures 7 and 8 provide an overview of the type of mutations counted. Of interest here is that there are many mutations unmarked (first bar) in some data sets, while not many mutations are marked as unknown (last bar). Furthermore, based on figure 7 it should be possible to separate the PAAD type from the others using frameshift and nonsynonymous (bars 2 and 4). PAAD is not one of the cancer types with an unexpected high number of intergenic and intronic mutations in figure 5. The genomes again show the expected profile of the vast majority having an unknown exonic function.

Figures 9 and 10 depict counts of AVSIFT values. These figures are shown as an example of the counts created for all variant effect predictors. In all these plots it is the details that matter, it seems. For each predictor the peak is always at the side with the highest effect.

General remarks

- For some cancer types there are still mutations also found in ESP5400, 1000G and dbSNP. Most are dbSNP which is probably due to the fact that Annovar uses a newer version than our filtering setup does. A possible tweak later on is to remove these mutations as well. The effect seems more persistent in the genomes.
- Figures that visualise where on the genome mutations are found for each cancer type are in the making, but provide computationally challenging.
- Papers about diverse density (DD) [6] and support vector machines (SVM) [4] have been read.
- All ECG-WGS improvement pipelines have finished. Somatic SNP calling will commence as quickly as possible.

Next week

- Encoding of the features must no be done as quickly as possible. It it taking too long before a classifier is running on them.
- The rest of the week should consist of experimenting with the MIL toolkit.

- A correlation test between project and mutation counts should be done. We expect there to be no correlation, indicating no project bias.
- Some additional figures relating features to projects could be created.

5 The first experiment

Goal A MIL dataset shall be created from selected cancer types. It is therefore possible to perform the first experiment on a selected subset of the available data.

Results

Selection of data At first the idea was to select three cancer types from the CGP-WES data set. Each should have a comparable number of overall mutations and samples. They should furthermore be from different cell types and not contain any (for now) unexplainable aspects, such as many mutations in intergenic regions. These three data sets should provide the best case scenario where there is a comparable amount of evidence and the phenotypes are clearly different.

It turns out that cancer types that meet all these characteristics have a large number of samples with a single mutation. For now it is unclear what to do with these samples exactly and it created a few issues when processing the MIL dataset. Therefore the choice was made to filter the two biggest cancer types from CGP-WES (BRCA and LUAD), removing all samples with less than 20 mutations. This creates the following MIL dataset:

```
73749 by 98 MIL dataset with 368 bags: [209 pos, 159 neg]
```

368 samples, of which BRCA is marked as positive and LUAD as negative. Each mutation is described by 98 features, the gene annotation is not included for now as it would introduce another 22.000 features. 73749 mutations is a bit large for an experimental setting, but seems to work for now. It should be mentioned that the features are on very different scales and should be normalised before processing.

First experiment In order to explore this BRCA versus LUAD dataset we set up the following experiment: Calculate a pair-wise distance between bags. Each pair of samples is replaced by one distance measure. Here we keep all instances and therefore assume that each instance is descriptive of the bag label. This new matrix can be used to test the following hypothesis:

> *It is possible to separate BRCA and LUAD samples by means of their calculated pair-wise distance.*

Classifier	Error rate	Area under the curve
LDC	17.6	97.1
QDC	58.8	91.4
Fisherc	41.2	56.4
NMC	35.3	80.0
LOGLC	52.9	54.3

Table 1: Error rates per classifier.

In order to test this hypothesis we calculated mean of mean, mean of min, min of min, sum of min, hausdorff, mahalanobis and miRBF distances. Mean of mean for example represents the mean of the mean instance distance between two bags. Figure 11 shows the first two principal components of each of the distance matrices. Visual inspection of figures containing an additional third principal component revealed that the first two separate the data best.

We choose to experiment further with mean of min (figure 11(b)), although min of min (11(c)) and sum of min (11(d)) could have been good candidates as well.

We trained and tested a linear, quadratic, fisher, nearest mean and logistic linear classifier on the mean of min distance matrix by ten rounds of 10 fold cross-validation. Both linear and quadratic classifiers were given 0.1 and 0.1 as regularisation parameters. Before training the data was scaled to have zero mean and unit variance. Table 1 shows the results.

LDC unexpectedly yielded the lowest error rate. After inspection of the principal components (figure 11) we expected that a quadratic classifier would work best. Furthermore, a success rate of 82.4% seems rather high given that a significant proportion the instances are suspected to be not descriptive of the bag label. It seems too early to conclude anything at this stage.

At the end of this week the following questions are unanswered:

- Where does the success of the LDC come from and does it have the same effect on other distance measures?
- Why does D not consist of exactly half the positive and half the negative bags? How is assignment of positive or negative done when a distance is calculated between a positive and negative bag?
- Why are elements on the diagonal of D not zero? The distance of a bag to it self should be zero.

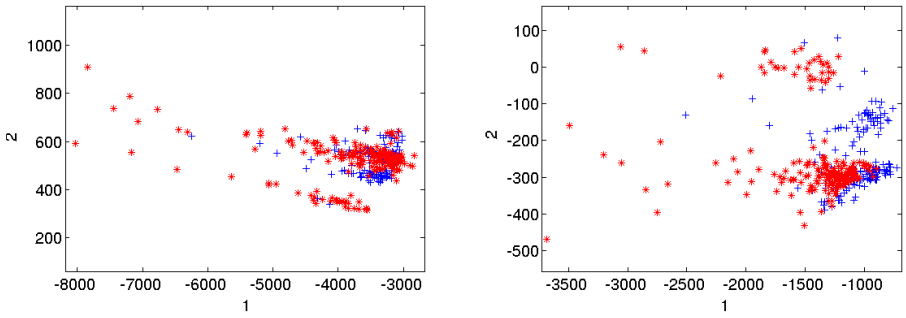
General remarks

- A separate log is started in which steps taken in Matlab, together with their conclusions are collected.

- Somatic SNP calling on the ECG-WGS dataset has finished. Additional quality control is still to be done.
- There was no time this week to inspect the correlation between project and features.
- A figure should be created that depicts the distribution of the number of mutations per sample per cancer type. It gives insight in how many samples have a low number of mutations.

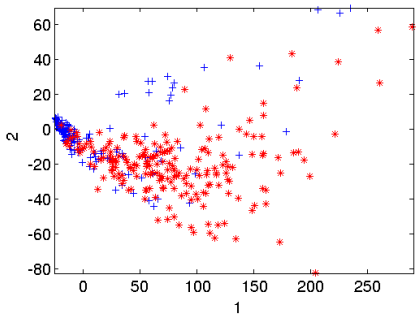
Next week

- A presentation will be given about this project at the student meetings on Thursday.
- Most work will focus on answering the above questions.
- A correlation test between project and mutation counts should be done. We expect there to be no correlation, indicating no project bias.
- Some additional figures relating features to projects could be created.

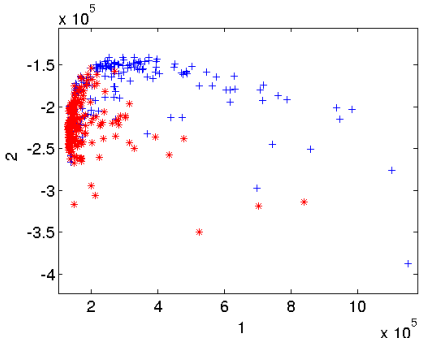


(a) Mean of mean

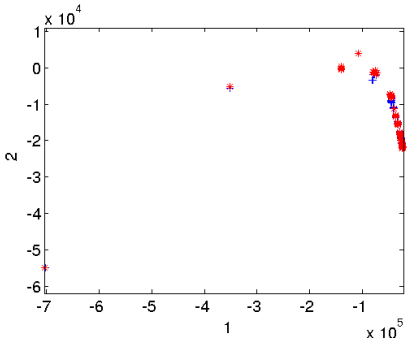
(b) Mean of min



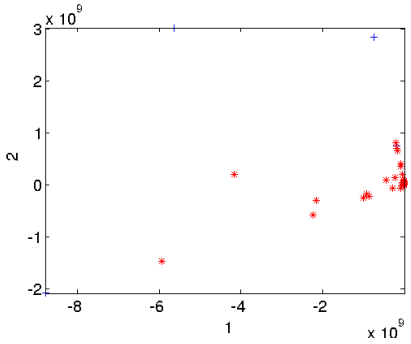
(c) Min of min



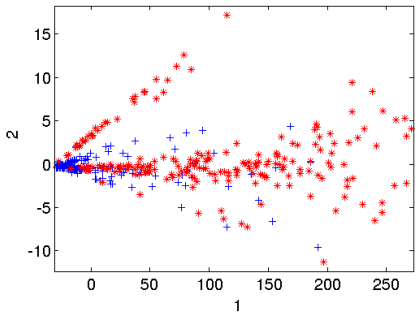
(d) Sum of min



(e) Hausdorff



(f) Mahalanobis



(g) MI RBF

Figure 11: Result of pair-wise distance calculation, followed by a reduction to two dimensions by means of PCA. Red is BRCA, blue is LUAD.

6 The first experiment continued

Goal A MIL dataset shall be created from selected cancer types. It is therefore possible to perform the first experiment on a selected subset of the available data.

Results

First experiment continued Last week ended with a number of questions. A few errors have been corrected while pursuing answers to these questions. D does not consist of exactly half BRCA and half the LUAD samples because an extra split in train and test set was done. This step is now removed and D nicely accounts for all 368 original samples. Scaling and splitting in train and test set is the reason why the diagonal of D is not zero and the matrix is not mirrored through the diagonal. The distance matrix therefore now makes sense. Figure 11 has been updated to reflect the corrected matrices.

We ran the list of classifiers on all seven distance matrices, table 2 shows the obtained error rates. QDC on the min of min and Fisherc on the mean of mean now have the best error rate, both at 20%. Table 3 shows the corresponding AUC. We can now conclude on the following hypothesis:

> *It is possible to separate BRCA and LUAD samples by means of their calculated pair-wise distance.*

It is indeed possible to separate BRCA and LUAD. Using 98 features, the min-of-min distance and the quadratic classifier we obtain an error rate of 20% and an area under the curve of 95.5%. Any subsequent classification implementation must have a lower error rate and higher area under the curve.

Classifier	Mean of mean	Mean of min	Min of min	Sum of min	Hausdorff	Mahalanobis	miRBF
LDC	48.6	28.6	37.1	22.9	57.1	31.4	37.1
QDC	57.1	57.1	**20.0**	42.9	45.7	48.6	57.1
Fisherc	**20.0**	40.0	40.0	40.0	28.6	40.0	25.7
NMC	34.3	25.7	28.6	25.7	34.3	22.9	22.9
LOGLC	22.9	45.7	37.1	40.0	28.6	37.1	22.9

Table 2: Error rates per classifier.

Classifier	Mean of mean	Mean of min	Min of min	Sum of min	Hausdorff	Mahalanobis	miRBF
LDC	71.7	90.7	96	95.3	63.7	96	92.3
QDC	87.7	86.7	91.5	54.2	47.5	98.3	93
Fisherc	85.7	60	61.7	60	70.8	61.7	76.7
NMC	68.7	88.7	91.3	86.3	60.7	94	90.3
LOGLC	78.3	59	46	58.5	69	51.3	68.8

Table 3: AUC per classifier.

Gene annotations Until now the gene column from the Annovar annotations was not included. Work this week has focussed on adding this information. Ensembl contains 20412 unique gene names that represent coding genes. An additional set of around 40.000 genes is also available that contains pseudogenes and various RNAs.

We aim to create two feature sets. One containing the Ensembl protein coding genes plus an additional feature that serves as a catchall for mutations in any gene not in this set. The second set will contain all genes of the various available categories that are conserved up to a certain extend. How to exactly do this is unclear at this stage.

Work on creating the first set is started, but has run into various computational difficulties.

Having a set containing all protein coding genes allows us to perform an additional experiment. For each sample we can create a matrix that contains how often a mutation in each gene is observed (a simple scoring matrix). When we add the type of cancer as a label we create a 'regular' classification problem. This problem can serve as an additional base-line, together with the above experiment.

General remarks

- A QC-filtering test run is done on CBF samples from the ECG-WGS dataset. Filtering removes around 1/3rd of the mutations. An average of around 100.000 mutations remains. This is too much, therefore additional filtering is required.
- Inspecting the correlation between project and features is still not done.
- A start is made in creating the distribution of the number of mutations per sample, but this figure is not finished yet.
- A presentation about this project was given at the joined student meeting.

Next week

- The focus is on creating the gene feature sets. We can give set one to a classifier once it is finished. Work on set two will start afterwards.
- The distance experiment can be run again with the gene features. Calculating the distances shall be much more time consuming this time. It will be interesting to see if the performance improves.

7 The ambiguity of gene names

Goal This week work continues on the gene annotations. The goal is to create a gene scoring matrix in which a row is a sample and a column is a gene. Each cell in this matrix contains the number of mutations in the particular sample and gene. We will also tweak the experiment from last week to investigate the rather high area under the curve (AUC).

Results

First experiment continued The results obtained on the distance matrices reported last week provoked a few questions. Mainly the AUC is rather high and doesn't directly match up with the error rates reported. In order to investigate these potential inconsistencies we've proposed the following, slightly tweaked, experiments:

1. In stead of handing a MIL dataset to a regular classifier, we transform the data into a regular dataset. In theory this shouldn't make a difference.
2. The crossvalidation and error calculations are done using specific MIL toolkit methods. These are replaced by regular methods. Again, in theory this shouldn't make a difference.

Furthermore, in both experiments we remove the LOGLC classifier and add a linear support vector machine (SVM) and random forest classifiers. It is worth mentioning that these results are obtained using only 10% of the mutations from the BRCA and LUAD exomes.

Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	57.1	31.4	48.6	28.6	37.1	37.1	22.9
Bayes-Normal-2	45.7	48.6	57.1	57.1	57.1	20.0	42.9
Fisher	28.6	40.0	20.0	40.0	25.7	40.0	40.0
NearestMean	34.3	22.9	34.3	25.7	22.9	28.6	25.7
RandForest50	21.4	**6.3**	29.7	20.6	18.9	13.1	16.0
SVM	37.1	20.0	25.7	20.0	25.7	25.7	**8.6**

Table 4: Error rates per classifier for experiment 1.

Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	63.7	96.0	71.7	90.7	92.3	96.0	95.3
Bayes-Normal-2	47.5	98.3	87.7	86.7	93.0	91.5	54.2
Fisher	70.8	61.7	85.7	60.0	76.7	61.7	60.0
NearestMean	60.7	94.0	68.7	88.7	90.3	91.3	86.3
RandForest50	84.8	**98.5**	74.2	89.0	86.5	91.9	91.8
SVM	85.3	78.7	80.7	89.7	90.7	83.3	**98.0**

Table 5: AUC per classifier for experiment 1.

Tables 4 and 5 show the results of experiment 1. The changed format makes it difficult to compare these tables to the ones from the previous week. The obtained error rates for the four classifiers that were run both last week and this week are identical though. We conclude here that using a regular prtools dataset in stead of a MIL dataset has no effect on the results.

The two new classifiers added this week yield interesting results. The error rate of the random forest classifier on the Mahalanobis distance and SVM on the sum of min are the best results seen thus far. Again we see a discrepancy between the error rate and the AUC. These numbers are hard to beat if they turn out to be correct. They seem almost unbelievably good.

Notes:

- Experiment 2 is still running.
- Adding **classc* when calculating the performance does not have an effect on the AUC and classification error.
- A rather strange effect is that all classifiers, except the random forest, show a 0 standard deviation on the obtained error rates (only for experiment 1). This seems odd, as we're performing 10 rounds of 10-fold-CV and one would expect to see a little bit of variation between each round.

Gene annotations and intermediate experiments Work has continued on the gene annotations. It turns out that of the 40.000 additional (non-coding) genes available in Ensembl only 151 have mutations. As these are pseudogenes and small RNA's we opt to not add these genes as features. A separate 'catch-all' feature is created that will represent all these genes.

A more persistent problem is the remaining 3500 genes that are not known in Ensembl. A small proportion can be explained, as they are synonyms for known genes. These synonyms have been obtained through HGNC and mapped back onto Ensembl gene ids. But there are still roughly 3000 gene names unexplained. Another option is to have a stab at UCSC genes.

An intermediate version of the gene scoring table is currently available. It is very different from the data used to calculate the distances. This dataset contains all 1785 BRCA and LUAD exome samples (including the samples with only one mutation). In the distances setup we selected 10% of the samples, here the full matrix is given to the classifiers.

Two sets of classifiers are running on the intermediate version of the gene scoring table. The first uses MIL methods to assess the error rates and contains the same classifiers as mentioned in table 4. This is a rather naive approach, as the NMC and QDC classifiers will take a very long time when running on 20.000 features. This set will get eight days maximum to produce results. After time

has run out (Wednesday next week) it will be stopped, taking the intermediate results as an indicator.

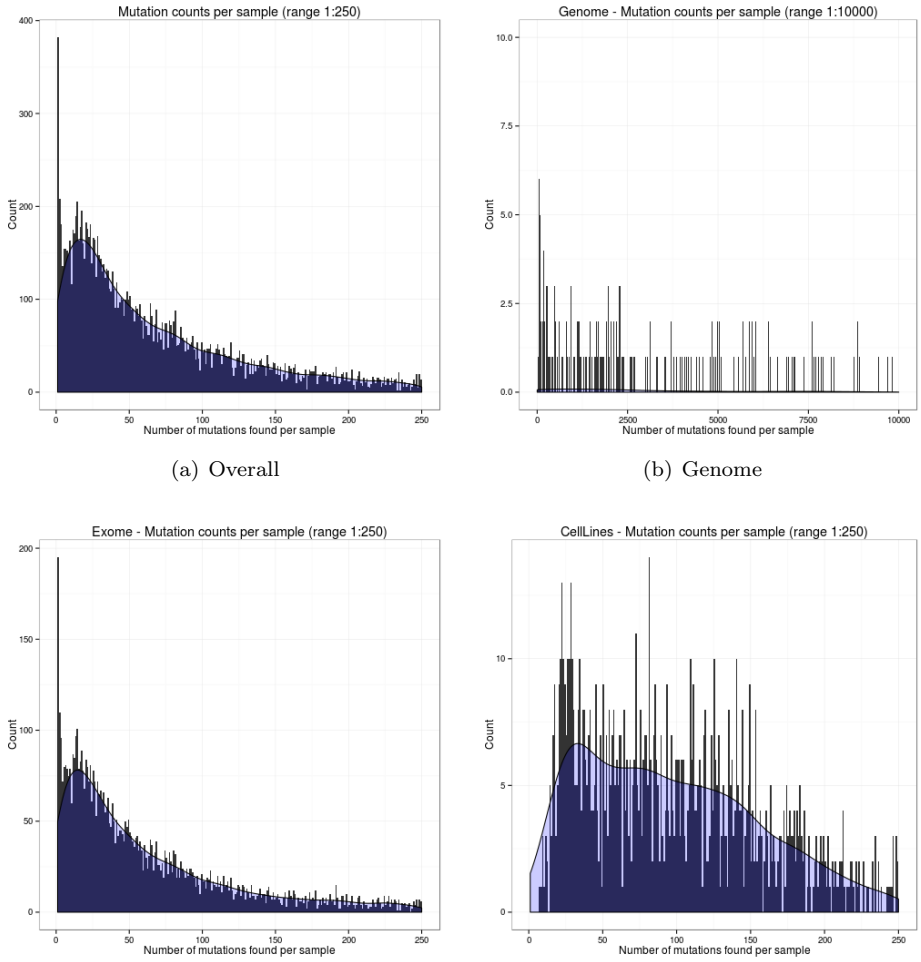
The second set consists of the nearest mean, SVM and random forest classifiers. All features containing only zeroes have been removed. The number of features is therefore reduced to roughly 18.000. It is still a computationally intensive job. Therefore we also give this job eight days to finish, which is until Friday next week. Downsampling to 10% is an option at a later stage if this job takes too long.

General remarks

- Figure 12 shows the distribution of mutation counts for samples in the different data sets. It is mainly the exomes that have a large number of samples between 1 and 10. This could mean we would lose a substantial number of samples when using a MIL analysis.
- Calculating the distances with the gene features has not been done. This is a computationally intense experiment. It is perhaps better to somehow summarise the genes and therefore reducing the number of features before calculating distances.
- The gene scoring matrix experiment has been formalised in a single Matlab script. The idea is to not only have conclusions, but also the exact way the data was derived for later reproducibility. Each experiment should be summarised in similar fashion. How to exactly handle intermediate versions beyond version control is a bit unclear. Perhaps the separate log should not only contain conclusions also include the code used to obtain the results.

Next week

- The UCSC genes will be queried in order to reduce the number of missing genes.
- A number of correlation tests between genes and labels can be performed. Also still on the list is a correlation test between mutation counts and projects in order to investigate possible project bias.
- A formalisation of the distance experiments should be created as a single Matlab script.
- The distance experiment can be run on the full BRCA and LUAD datasets in order to compare the classification results. If the results are in the 'ballpark' of results obtained with the 10% we can (for now) experiment further with the smaller dataset.
- Currently available Encode annotations can be added to the distance experiment.



(a) Overall

(b) Genome

(c) Exome

(d) Cell lines

Figure 12: Distribution of samples with a given mutation count. Binwidth is one (10 for genomes), therefore each bar represents the number of samples containing the number of mutations as specified on the x-axis. For viewing convenience the x-axis range has been limited.

8 Crossvalidation, done right

Goal The experiments from last week have finished and the results are in. Most of the week is spend on fixing the underlying problems. Furthermore, we perform the distance experiment on the whole BRCA+LUAD exomes and setup a Fisher exact test for the gene mutation counts.

Results

A bit of structure First off, we introduce the five steps of this project:

1. **Establish a baseline**: Count the number of mutations per gene, per sample. This results in a regular classification problem. The matrix contains samples on the rows and genes on the columns, while each sample is labelled with its cancer type. It is the equivalent of the micro-array experiments that have been performed over the last 15 years (with considerable success).

2. **Additional features give more power**: Calculate distances between samples (bags). Each sample is reduced to a vector of distances to each other sample. The matrix therefore contains samples on the rows and samples on the columns. When adding the cancer type to each row as a label we again have a regular classification problem.

3. **Establish which features are most descriptive**.

4. **MIL improves results**: MIL does not assume that all mutations are descriptive for the phenotype (label). It therefore does not need to summarise the features to reduce the effect of non-descriptive mutations. Which in principle leads to biologically interpretable results.

5. **Mining of results**.

Finished experiments - distance We refrain from mentioning the results of experiment 2 from last week. We realised that the crossvalidation setup was incorrect and opted therefore to run the experiment again with adapted code. Table 6 shows the result of running the familiar six classifiers on distances between samples, after selecting 10% of the BRCA and LUAD mutations.

Table 7 shows the classification error on the full BRCA and LUAD exomes. The same three classifier-distance combinations show the lowest classification error, although the rates have slightly increased. This does not hold for all combinations. The Bayes classifiers (LDC and QDC) for example have in some cases worsened by 10%.

We have also recreated the PCA plots from week 5, but now with the full data. They are depicted in figure 13. The shapes of the data when plotting the data against the largest two principal components does not change fundamentally (although does change direction in some cases).

Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	30.3	21.8	39.0	**12.3**	22.5	30.6	15.3
Bayes-Normal-2	44.6	48.7	56.8	55.1	56.8	30.6	40.1
Fisher	42.7	44.2	15.4	38.0	33.1	42.4	36.0
NearestMean	39.0	22.5	39.1	29.6	26.1	26.1	22.6
RandForest50	23.3	**12.3**	31.5	17.3	21.5	17.9	17.2
SVM	30.3	18.0	23.7	15.0	18.2	23.9	**11.4**

Table 6: Classification error per classifier for the experiment from last week. These results are obtained through the corrected crossvalidation setup and are based on 10% of the BRCA and LUAD exome data.

Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	29.0	26.5	48.4	**13.2**	24.0	59.3	15.2
Bayes-Normal-2	57.4	58.3	55.6	58.1	59.3	59.3	61.0
Fisher	41.3	42.5	15.5	36.2	42.9	42.7	37.6
NearestMean	35.6	27.7	42.0	30.1	27.0	27.6	24.3
RandForest50	23.8	**13.4**	30.3	17.3	21.6	19.3	16.3
SVM	26.6	18.7	21.0	16.1	22.1	25.9	**14.1**

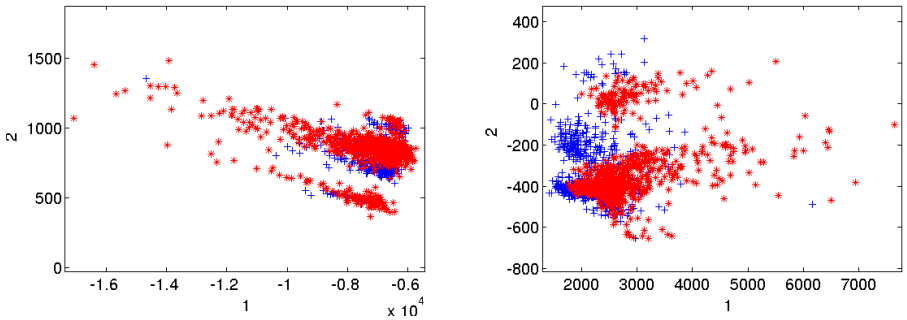
Table 7: Classification error per classifier for the experiment from last week. These results are obtained through the corrected crossvalidation setup and are based on all BRCA and LUAD exome data.

Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	71.9	91.2	67.1	**94.2**	89.1	86.6	91.3
Bayes-Normal-2	61.7	77.4	63.8	70.0	69.4	81.9	70.9
Fisher	55.7	57.5	90.2	62.7	68.7	59.1	62.2
NearestMean	64.2	88.8	62.0	83.8	86.7	85.5	84.5
RandForest50	81.7	**92.5**	74.7	88.6	84.5	85.9	89.2
SVM	79.5	86.4	81.8	91.2	89.8	80.5	**93.5**

Table 8: AUC per classifier for the experiment from last week. These results are obtained through the corrected crossvalidation setup and are based on 10% of the BRCA and LUAD exome data.

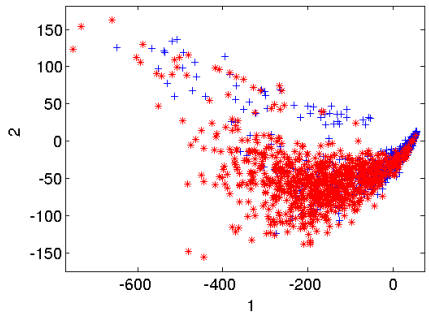
Classifier	Hausdorff	Mahalanobis	Mean of mean	Mean of min	miRBF	Min of min	Sum of min
Bayes-Normal-1	72.0	88.6	57.2	**90.4**	84.8	74.9	88.6
Bayes-Normal-2	5.1	77.2	50.1	84.4	84.3	83.4	6.5
Fisher	58.2	58.5	87.5	63.6	58.4	58.5	60.9
NearestMean	65.2	85.9	60.3	81.7	84.5	84.6	83.0
RandForest50	80.0	**88.5**	74.1	85.0	82.9	82.2	87.5
SVM	80.0	83.9	83.0	88.7	83.8	76.9	**89.7**

Table 9: AUC per classifier for the experiment from last week. These results are obtained through the corrected crossvalidation setup and are based on all BRCA and LUAD exome data.

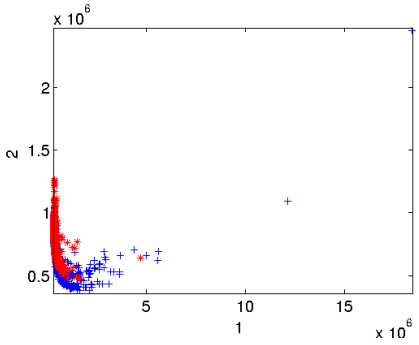


(a) Mean of mean

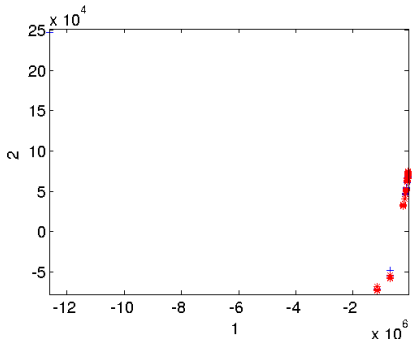
(b) Mean of min



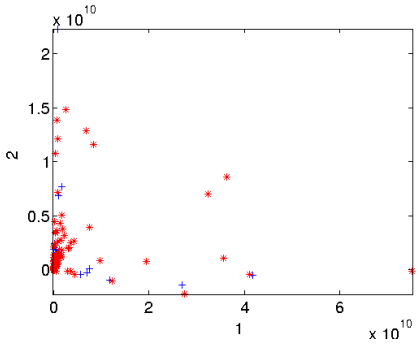
(c) Min of min



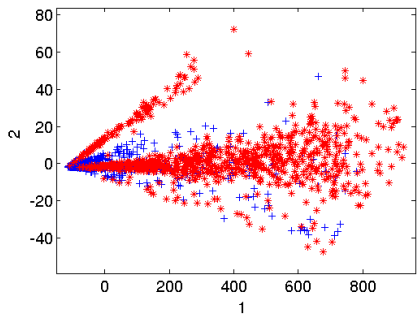
(d) Sum of min



(e) Hausdorff



(f) Mahalanobis



(g) MI RBF

Figure 13: Result of pair-wise distance calculation, followed by a reduction to two dimensions by means of PCA. Red is BRCA, blue is LUAD.

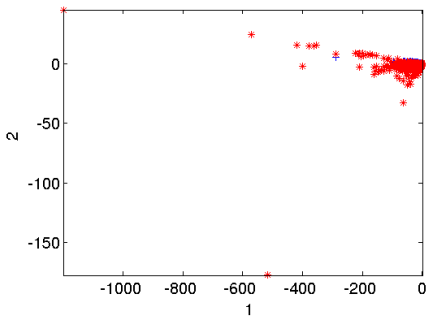
Finished experiments - gene scoring We have also finished the gene scoring experiment. Here we create matrix in which each cell depicts how often a gene is mutated in each of the samples. The matrix has samples as rows and genes as columns. Figure 14 shows the results.

The nearest mean classifier shows the lowest classification error, but it is much larger than the best classification error obtained on the distances. A PCA plot containing the two largest principal components on its axis shows that both classes are on top of eachother. The third component reveals that the data points form a cloud in which both classes are well mixed (not shown). It seems that classification based on genes alone is difficult and that the annotations as used in the distances experiment have added value.

What should be considered though is that genes with a low mutation frequency add little to the separability and are noise instead. We will therefore extend this experiment next week by performing feature selection before the classifiers are run.

Classifier	Cl.err	AUC
NearestMean	24.3	72.7
RandForest50	45.6	59.0
SVM	58.8	49.7

(a) Error rates



(b) PCA scatter plot

Figure 14: Results of gene scoring experiment. The nearest mean classifier comes out on top, but the error rates are substantially less then in the distance experiments. The PCA plot (red is BRCA, blue is LUAD) supports this as the two classes seem right on top of eachother.

Final remarks

- A script is created (but not yet finalised) that performs Fishers exact test on the gene scoring matrix. First we sum the counts for each gene per cancer type (BRCA and LUAD for now) and count the number of samples that have no mutation in that particular gene. We then calculate a p-value of observing these four values. The result of this test can be used as a feature selection step, for example taking only genes with a p-value less than 0.05.

- Another script creates a histogram that shows in the x-axis the gene mutation counts (cell values of the gene scoring matrix) and on the y-axis the frequency. We aim to distinguish between BRCA and LUAD. The purpose of this plot is to show whether there are any genes that are mutated more than once in a single sample. This script is also not finished yet.
- Generation of the above tables out of the raw Matlab results is finally fully automated.

Next week

- The remaining exome cancer types (classes) can go through the annotation pipeline. Once completed it's possible to extend the above experiments to the full data set. We aim to use a one-versus-all (OVA) classification scheme. It remains to be seen whether the current code base is sufficient for this experiment or whether increased parallelisation is required.
- The gene scoring experiment can be rerun, after a feature selection step. We could use the p-values as described above or calculate a minor-allele-frequency variant and use that for filtering.
- We aim to investigate the contribution of more principal components than just the largest three.
- The UCSC genes should be queried in order to reduce the number of missing genes.

9 First MIL results

Goal We run the MILES classifier on the BRCA and LUAD exomes. We expect MILES to perform better than a regular classifier on the distances if indeed the data contains mutations that are non-descriptive for the labels. Furthermore, we will also look into the finished Fisher tests on the gene scoring matrix. Finally, we explore more principal components of the gene scoring matrix.

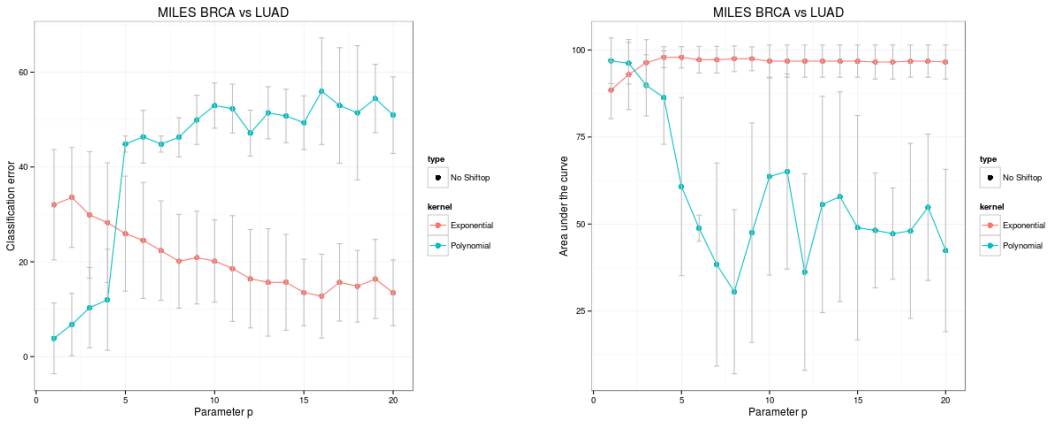
Results

MILES classification We gave a subset of the BRCA and LUAD mutations to the MILES multiple-instance learning classifier. This subset consists of 74 BRCA and 60 LUAD samples combining to 29.999 mutations. The MILES classifier performs feature selection first and subsequently trains an SVM classifier. Throughout the week we have tried different kernels and different parameters.

Figures 15(a) and 15(b) show the classification errors and area under the curve (AUC) respectively for the two best performing kernels. The polynomial kernel with a parameter of 1 yields the lowest classification error (3.68%) after 10 fold cross validation. The error bars still show a reasonably large standard deviation, especially on the AUC where $p > 4$.

The classification error and AUC combination for the exponential kernel does not make any sense. While the AUC is always higher than 80%, the classification error barely goes below 20%. We expect the classification error and AUC to roughly add up to 100.

In all these experiments the operating point is not shifted. Currently, experiments are running for both kernels with the latest *shiftop* function. It will be interesting to see how the results compare to the figures shown here. Furthermore, it is also interesting to see if the results change when all BRCA and LUAD data is used.



(a) Classification error

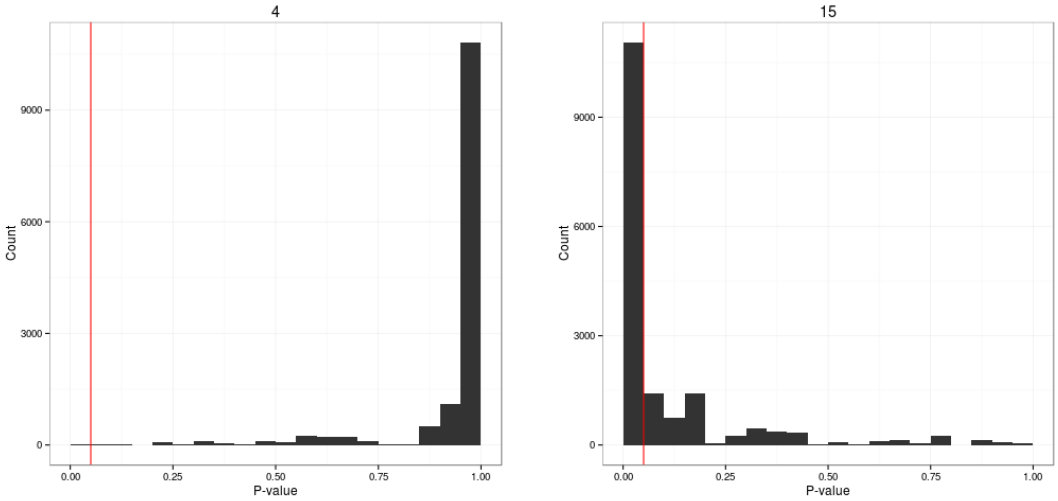
(b) Area under the curve

Figure 15: Behaviour of the classification error and AUC when applying the MILES classifier to the BRCA and LUAD exomes when the kernel parameter is changed. The red line represents the exponential kernel, while blue represents polynomial. In both cases the operating point is not shifted.

Gene scoring gene correlations Using the full gene scoring matrix for BRCA and LUAD, we performed a one sided Fisher test for each gene. For both cancer types this yields a p-value, of which the distributions are shown in figures 16(a) and 16(b).

In LUAD many genes are more often mutated than one would expect by chance. Over 11.000 genes have a p-value lower than 0.05. In contrast, BRCA has only a few genes with a p-value lower than 0.05 (15, shown in table 10). The large difference could be explained by the higher number of mutations per sample in LUAD (median 176.5, against 37 in BRCA).

Table 10 shows the 15 genes found in the BRCA exomes. Five genes are known Cancer Genome Census (CGC) genes, of which three are implicated to play a role in breast cancer. It will be interesting to see how many of these remain after performing the Fisher tests with all exomes.



(a) BRCA (b) LUAD

Figure 16: P-value distributions for the BRCA and LUAD exomes. The vertical red line denotes $p = 0.05$.

Gene	P-value	In CGC	CGC tumour types (somatic)
RBMXL3	0.02410298	No	
SRSF10	0.008402683	No	
LPPR2	0.02410298	No	
GRID2IP	0.0410883	No	
PIK3CA	1.975527e-05	Yes	colorectal; gastric; gliobastoma; breast
RUNX1	0.04907678	Yes	AML; preB- ALL; T-ALL
GREB1L	0.01971003	No	
CBFB	0.0004280226	Yes	AML
XBP1	0.009446009	No	
ZFP92	0.03915817	No	
CTCF	0.008416183	No	
PIEZO1	0.04802539	No	
MAP3K1	2.060971e-14	No	
MAP2K4	0.0006687656	Yes	pancreatic; breast; colorectal
CDH1	0.0004659289	Yes	lobular breast; gastric

Table 10: Genes with a P-value lower than 0.05 when comparing the BRCA exome gene scoring against the LUAD exome gene scoring.

PCA gene scoring matrix Last week we showed a scatter plot showing the gene scoring matrix mapped onto its two largest principal components (PCs). The data comes up as one big blob, with a couple of outliers. This week we investigate whether more than two PCs could help separate the two classes better. A possible extension is that we use these PCs to train a classifier, yielding better performance (although at the cost of loosing interpretability).

Figure 17 shows a scree plot of the first 25 PCs. It is mostly the first, with some help of the second eigenvalue that describes the variation in the gene scoring matrix. PCs one and two account for 92% of the variation, while 570 PCs account for 99% (data not shown). We conclude from this experiment that indeed the two classes are right on top of eachother and adding additional PCs will not help us separate them better.

Final remarks

- All exome data is now processed and ready for the one-versus-all classification setup.
- Played around with the Python Sklearn package in order to possibly replicate the gene scoring matrix findings. Needs more work.

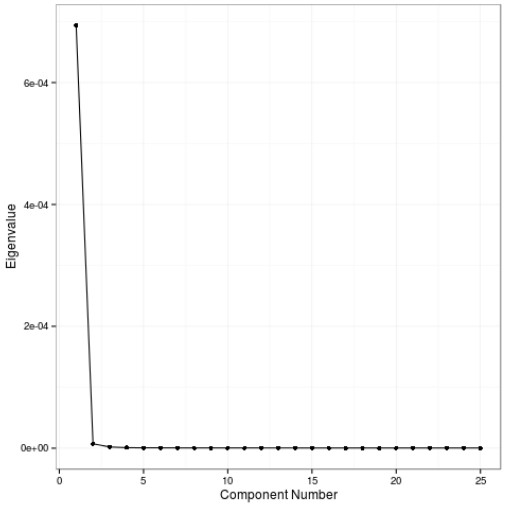


Figure 17: Scree plot of the first 25 eigenvalues of the gene scoring matrix.

Next week

- The one-vs-all classification code can now be created.
- After the currently running MILES experiments have finished, we can scale them up to include all BRCA and LUAD exome data.
- The list of genes is not finalised and requires more work still.
- The gene scoring experiment can be extended with a feature selection step.
- A figure showing the distribution of mutation counts per sample in BRCA and LUAD is in the works, but not yet finished.

10 One-vs-All first results

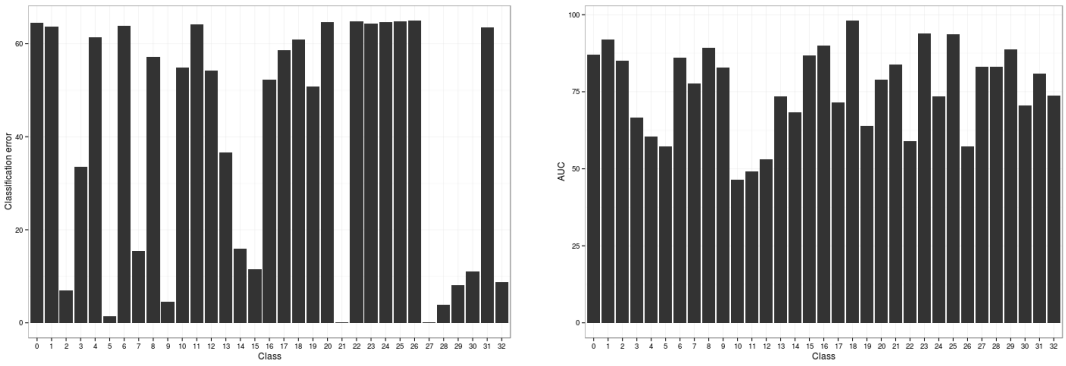
Goal We aim to scale up all experiments to perform one-vs-all classification on all the exomes. Furthermore, the results of running MILES with different kernel parameters are to be finished and examined.

Results

Gene scoring matrix one-vs-all The gene scoring matrix experiment has now been scaled up to the whole exome dataset, using 33 types of cancer. Before we marked BRCA as *positive* and LUAD as *negative*. Now each cancer type we mark the samples of that type as *positive* and all others as *negative*. We then ran the nearest mean classifier (which yielded the best results on BRCA versus LUAD) using 10-fold cross-validation to obtain a classification error and area under the curve.

Figure 18 shows the results. LUAD (15) still has a reasonable classification error (around 15%) and AUC (around 80%), but BRCA (4) seems undistinguishable from the rest. Furthermore, it seems that types 5 (CESC), 21 (NB) and 27 (PRAD) are very different from the rest, while 2, 7, 9, 14, 15, 28, 29, 30 and 32 all have a classification error under 20%. A problem with these numbers is that none of the types with a low classification error has a high AUC. It is therefore difficult to conclude anything final from these results.

PCA plots of each cancer type versus the rest will be interesting to see.



(a) Classification error (b) Area under the curve

Figure 18: Results of one-versus-all classification on all exomes.

Gene scoring gene correlations - Revisited The Fisher independence test has been rerun after some discussion this week. The results from last week showed p-value distribution obtained by summing the the number of mutations per gene, per cancer type in the scoring matrix. That has now changed to accounting the number of samples with a mutation per cancer type. The results change slightly. The new BRCA genes with a p-value lower than 0.05 are in bold face in table 12.

We have not changed the type of test that is performed. First a right-tailed Fisher test is done on BRCA vs LUAD. Then we switch the columns of the table

and perform a right-tailed Fisher test on LUAD vs BRCA. In the right-tailed variant we test independence between fields a and d in table 11. A significant value will therefore correspond to a large ratio a/c as compared to the ratio b/d. In other words, we find genes that are significantly more often mutated in BRCA as compared to LUAD, taking into account the number of samples available.

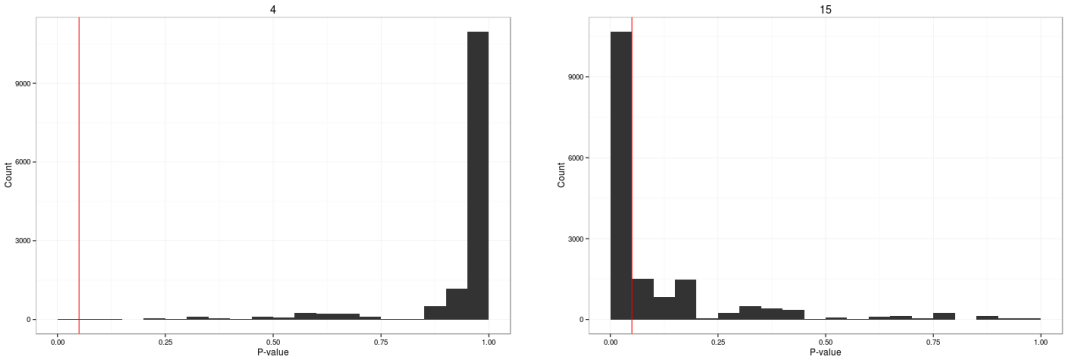
Indeed the 15 genes in table 12 show a larger ratio a/c as compared to the ratio b/d. When switching the columns a selected number of genes showed the same results.

	BRCA	**LUAD**
Gene	a	b
Not gene	c	d

Table 11: Example counts table. Cell a contains the number of BRCA samples having a mutation in the gene, while c contains the number of samples without a mutation in the gene. b and d contain the counts for LUAD samples.

Gene	**P-value**	**In CGC**	**CGC tumour types (somatic)**
RBMXL3	0.02410298	No	
LPPR2	0.02410298	No	
GRID2IP	0.0410883	No	
PIK3CA	6.345687e-05	Yes	colorectal; gastric; gliobastoma; breast
GREB1L	0.007979655	No	
CBFB	0.0004280226	Yes	AML
XBP1	0.03226574	No	
ZFP92	0.03915817	No	
CTCF	0.008416183	No	
GATA3	0.03320735	Yes	Breast
PIEZO1	0.04802539	No	
EFR3B	0.03915817	No	
MAP3K1	2.042518e-09	No	
MAP2K4	0.0006687656	Yes	pancreatic; breast; colorectal
CDH1	0.0001649925	Yes	lobular breast; gastric

Table 12: Genes with a P-value lower than 0.05 when comparing the BRCA exome gene scoring against the LUAD exome gene scoring.



(a) BRCA (b) LUAD

Figure 19: P-value distributions for the BRCA and LUAD exomes. The vertical red line denotes $p = 0.05$.

MILES kernel parameter Here we investigate the behaviour of the MILES classifier when changing the kernel parameter p for both polynomial and exponential kernels. We ran both experiments with and without shifting the operating point, while using a subset of the BRCA and LUAD exomes.

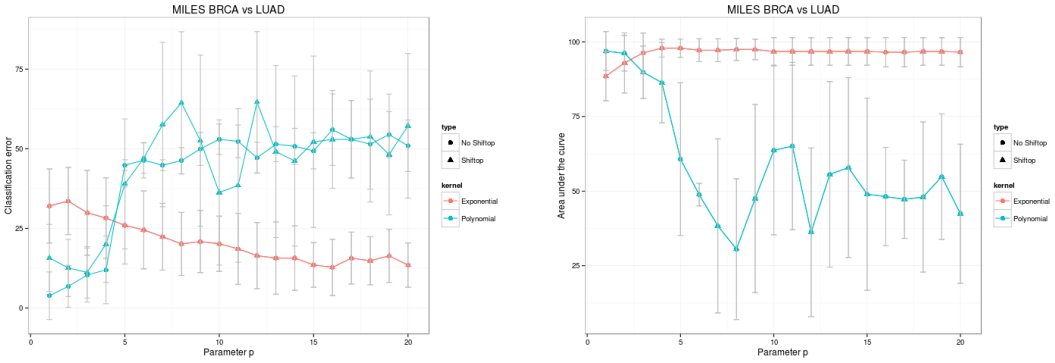
Figure 20 shows that for the polynomial kernel values up to $p = 4$ yield an interesting classification error and area under the curve. Shifting the operating point has a small negative effect. We expect these numbers to change a bit when scaling up to the whole BRCA and LUAD data, but not much. When $p > 4$ the classification error goes up, as well as the standard deviation on the classification error.

The exponential kernel seems to show a downward classification error trend. But at the same time the area under the curve is extremely high with little standard deviation. This could be a sign that there is too little data for applying the exponential kernel. The results of $p < 3$ seem plausible however. Shifting the operating point does not seem to have an effect.

It will be interesting to re-run these experiments on the whole BRCA and LUAD data in the future. With the current data a classification error of 3.68% using a polynomial kernel with $p = 1$ is the best attainable.

Final remarks

- MILES is running out of memory on the Delft compute servers when using the full BRCA and LUAD data. There should be ample memory available on these machines. Further investigation is required.
- Calculating the distance matrices between all samples is running.



(a) Classification error

(b) Area under the curve

Figure 20: Development of the classification error and area under the curve as kernel parameter p is changed.

- The Python sklearn experiment on the gene scoring matrix is finished and has yielded different numbers, although the best classifier (random forest with 50 trees in this case) has a comparable classification error to the NMC from prtools.

Next week Next week we aim to finish the current exploration phase and summarise the results and conclusions. Working with the full exome dataset proves to be too difficult on a daily basis. Experiments are starting to need multiple days of compute, which makes it hard to stay in control of all that is running.

Next week a new dataset shall be created with a selection of samples from five different types of cancer out of the exomes. Cancer types will be selected in such a way that the dataset is more complex than BRCA versus LUAD. We aim to test and improve the efficacy of our methods on this more complex data.

Furthermore, we would like to make the main three experiments in the five phase plan a bit more comparable. Therefore a number of new MILES experiments are to be designed.

Finally, some left overs from this week:

- The list of genes is not finalised and requires more work still.
- The gene scoring experiment can be extended with a feature selection step.
- A figure showing the distribution of mutation counts per sample in BRCA and LUAD is in the works, but not yet finished.

11 Wrap up and selecting new data

Goal This week we aim to wrap up the first phase of the project. That means finishing currently running experiments and address remaining questions. We also select different data which will be used in phase two.

Results

Fisher test one-vs-all exomes We have performed the Fisher test experiment on all exomes. Each time a specific cancer type is compared to all the others. Figure 21 shows how often genes receive a p-value lower than 0.05. In total 18233 genes are significant at least once (2180 are never significant). Only 38 genes are more often than seven times significant. Table 13 shows the ranked list and the cancer types by their identifier. The gene marked as *other* is a catchall for all gene names that are not in the known protein coding list.

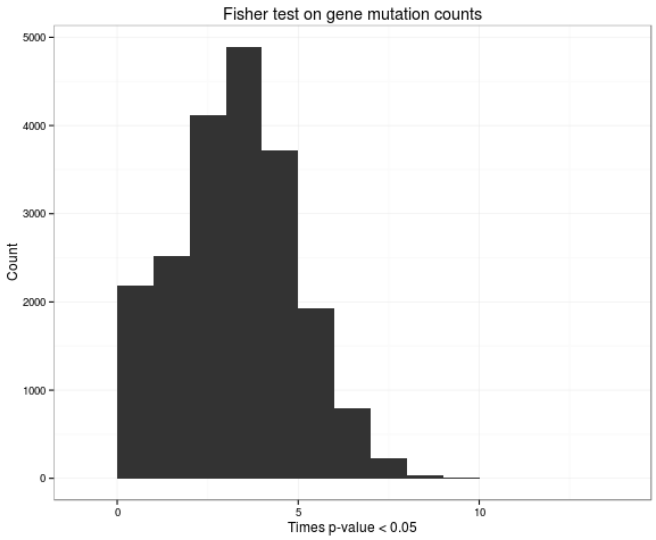


Figure 21: The distribution of how often each gene has a significant p-value. Most genes are 2, 3 or 4 times significant. Only a few genes are more than 7 times significant.

By eye there appears to be a connection between cancer types that are classified with a low error rate and the number of genes that are significant. The types appearing more often in the table seem to have lower error rates than types not appearing. This is not unexpected, as a larger volume of significantly mutated genes allows for more uniqueness and possibly for better classification performance. For example, 4 (BRCA) is not in the table, while 15 (LUAD) occurs 34 times. Last week we showed that the NMC classifier shows a low performance on BRCA, but does well on LUAD.

Please note that these numbers have not been corrected for gene length.

Selecting new data The goal of creating the above figures (as well as figures from last week and per cancer type significance distributions) was to get an idea of which cancer types make our exome data complex.

We select types that have poor classification performance in the gene scoring experiment and contain a relatively low number of samples. Furthermore, we look for groups of cancer types that are from the same cell type in order to take into account that the same genes could be perturbed in large quantities or the types of mutations are the same. Selected types therefore are: ALL, AML, Kidney (KIRP, KIRC), Ovarian (OV) and Thyroid (THCA) exomes.

A few observations:

- All types have low number of samples (100s).
- ALL and AML of similar family, KIRP and KIRC also more similar.
- Number of mutations per individual is more or less the same. Types with a high mutation rate all seem to yield better classification results.
- Combined these types account for less mutations than BRCA vs LUAD (67071 now, before 330567), making working with MILES potentially easier.
- Size of these types together should make it possible to later add BRCA to compare performance.

Gene	Times	Cancer types
other	13	2,5,7,10,11,12,13,16,18,29,30,31,33
COL22A1	9	7,11,15,16,28,29,30,31,33
MED1	9	2,5,7,11,15,16,30,31,33
MUC17	9	2,5,7,15,16,29,30,31,33
SYNE1	9	2,7,11,15,16,29,30,31,33
TTN	9	2,5,7,11,15,16,30,31,33
ADAMTS9	8	2,7,10,15,16,30,31,33
CDH10	8	7,11,15,16,29,30,31,33
CDH11	8	2,7,10,15,16,30,31,33
COL11A1	8	7,11,15,16,29,30,31,33
CREBBP	8	2,5,7,16,17,30,31,33
CSMD2	8	2,7,15,16,29,30,31,33
CSMD3	8	7,11,15,16,29,30,31,33
CTNNA3	8	5,7,10,15,16,30,31,33
DMD	8	7,11,15,16,29,30,31,33
DNAH5	8	2,7,11,16,29,30,31,33
DNAH7	8	7,10,15,16,29,30,31,33
DOCK4	8	2,5,7,15,16,30,31,33
DPP6	8	3,7,10,15,16,28,30,33
FAT1	8	2,7,11,15,16,30,31,33
FLG	8	2,7,15,16,29,30,31,33
FSIP2	8	7,10,15,16,20,29,30,33
HELZ	8	2,7,10,15,16,30,31,33
IL1RAPL1	8	7,11,15,16,29,30,31,33
IMPG2	8	2,5,7,16,29,30,31,33
KIF21A	8	2,7,15,16,29,30,31,33
LAMA2	8	7,11,15,16,29,30,31,33
LRRK2	8	7,10,15,16,29,30,31,33
MAP1B	8	2,7,10,15,16,30,31,33
MDN1	8	2,7,15,16,29,30,31,33
MYH9	8	2,5,7,11,16,30,31,33
NAV3	8	7,11,15,16,29,30,31,33
NLGN1	8	7,10,15,16,29,30,31,33
PLEC	8	2,7,11,15,16,30,31,33
PLXNB2	8	2,7,11,15,16,30,31,33
PRKG1	8	7,15,16,28,29,30,31,33
SIM1	8	7,11,15,16,29,30,31,33
WDR7	8	5,7,15,16,29,30,31,33
ZNF99	8	2,10,11,15,16,30,31,33

Table 13: Genes with a P-value lower than 0.05 when comparing the BRCA exome gene scoring against the LUAD exome gene scoring.

Final remarks

- MILES is still running out of memory on BRCA vs LUAD.
- Three distance measures have been calculated using the full exome data set. Classifiers are currently running.
- PCA plots of one-vs-all are not ready. It takes a very long time to create a single figure.

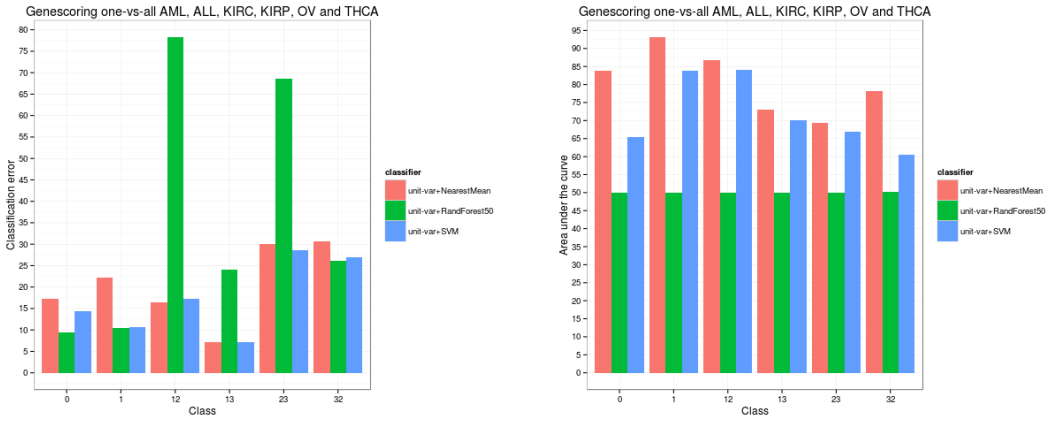
Next week Next week we are going assemble the new data and repeat the first three experiments. A number of comparative figures should also be made to elucidate the differences at data level.

12 Week 12

Goal This week the first three experiments are performed on the newly selected data. We also take a look at the memory issues with MILES.

Results

Experiments on new data - part one The results of the gene scoring experiment on the newly selected data are shown in figure 22. In this experiment we count the number of mutations per gene, per sample and perform one-vs-all classification per cancer type. For each cancer type the minimum classification error is 25% or lower, indicating that this problem is perhaps not as difficult as we anticipated last week.



(a) Classification error

(b) Area under the curve

Figure 22: Results of one-versus-all classification on the newly selected exomes.

In most cases the nearest mean and SVM classifiers perform comparably well, while the performance of the random forest classifier varies. Given the area under the curve for both SVM and random forests does not always make sense we maintain nearest mean as the best classifier for this problem.

The distance experiment is still running. We've calculated the three best performing distances and are using the same three classifiers used in the gene scoring experiment. Early results indicate that the performance is reasonable (around 20% classification error). We note here that the distance experiment on the full exomes has not finished, so it is difficult to make a comparison.

MILES has not been run on the newly selected data yet. The memory issues need to be solved first (see below).

MILES memory issues The problem with MILES is that the linear programming step needs too much memory when using a reasonable amount of instances (mutations), let alone full data sets. In order to circumvent this problem we have opted to run MILES on parts of the supplied data. We break the supplied data (for example BRCA vs LUAD) into random partitions that contain a specified fraction of the instances, making sure that the balance between positive and negative examples remains the same. If each partition must contain 10% of the instances we select 10% of the positive and 10% of the negative instances.

We first explored running MILES on just the partitions individually, which resulted in abysmal results (data not shown). Then we changed the setup to what is depicted in algorithm 1. This setup was run on the BRCA vs LUAD small data set containing 30.000 instances, for which we have MILES performance measures (3.68% classification error). As combiner we tested both the *maxc* and *votec*, while we also investigate the effect of shifting the operating point on the maxc combiner.

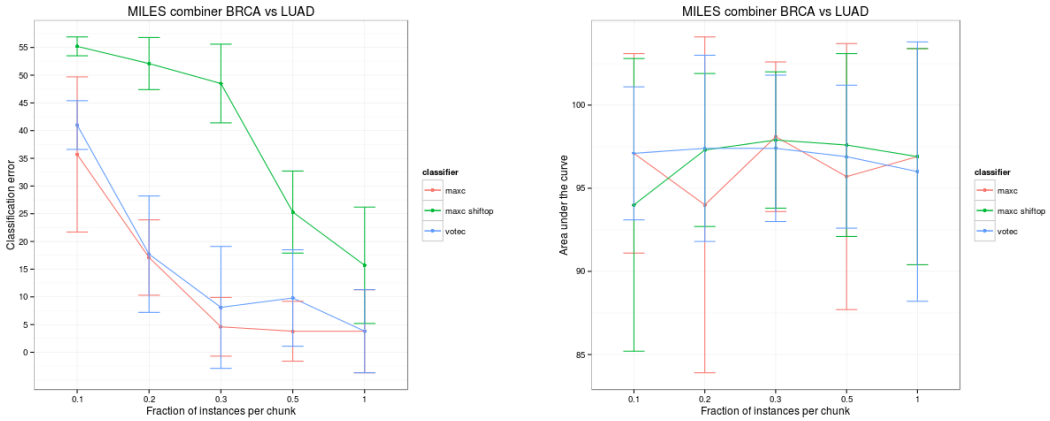
Algorithm 1: Pseudocode for the MILES partition/combine setup.

```
Data: A MIL dataset
Result: Classification error and area under the curve
for each CV split do
    create 1/fraction chunks of training data
    for each chunk do
        train a MILES classifier
    end
    combine classifiers and test on testing data
end
```

Figure 23 shows the results when changing the fraction of instances per chunk. There is not much difference between the maxc and votec combiners, maxc yielding slightly better results. Shifting the operating point to equal error rate for both positive and negative classes clearly has a negative effect.

The area under the curve seems to indicate though that the chunks for most fractions do not contain enough data for trustworthy results. We are therefore running this experiment with the maxc combiner on the whole BRCA vs LUAD exome set using a fraction of 0.1 (roughly 30.000 instances per chunk). Results should be in next week.

Problems are not completely over when this experiment works. On the full data set we might still run into problems. Some cancer types are represented by a large number of mutations, while others have only a few. Partitioning might split up a cancer type in such a way that only a single mutation per chunk is available.



(a) Classification error (b) Area under the curve

Figure 23: MILES combined classification on BRCA vs LUAD.

Next week Next week MILES should be properly explained in this document. We also continue the currently running experiment. Best case it is even possible to perform more MILES combiner experiments on different data.

13 Week 13

Goal This week we describe in general terms how the MILES classifier works.

Results

Introduction into MILES Multiple instance learning (MIL) is a variant of supervised learning. In regular supervised learning problems we have observations (samples) that are described by a number of features. Each sample is assigned a label, which determines its class. MIL problems differ in the sense that classlabels are assigned to bags of samples (see figure 24). Often only binary classification is possible, therefore usually either *positive* or *negative* are used as labels. Each bag B_i contains a number of unlabelled samples $x_{i1}, \ldots, x_{in}$, which are denoted as instances. Similarly to [3], for convenience we reindex all available instances as $x^k, k = 1, \ldots, 26$.

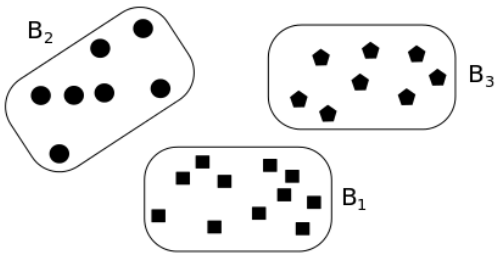
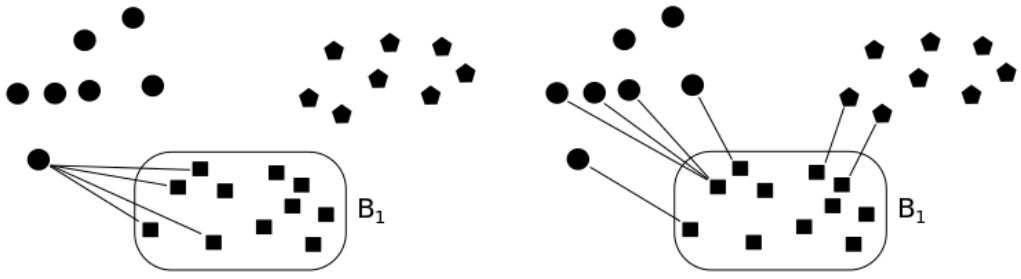


Figure 24: A MIL example containing three bags (B_1, B_2 and B_3) and 26 instances in a two dimensional space

In this project we model somatic mutations as a MIL problem. Mutations are available for a large number of tumours, of which the cancer type is known. Each tumour is modelled as a bag and labelled with its cancer type. A bag contains a number of instances, the mutations. We cannot assign labels to the instances as it is unknown which mutations are descriptive for the cancer type. When using multiple different types of cancer we use a one-versus-all (OVA) scheme, in which a particular cancer type is labelled as positive, while all other types are labelled as negative.



(a) (b)

Figure 25: Calculation of the distance between each instance and bag B_1. **(a)** Calculate distance between x^k and all instances in B_1. **(b)** Keep the minimum calculated in (a) as the distance between x^k and B_1.

A method is required that constructs a classifier whilst taking into account the uncertainty about the descriptiveness of the instances. Multiple instance learning via embedded instance selection (MILES) is such a method [3]. It transforms the MIL problem into a regular classification problem without directly assigning a label to each instance. It uses a distance based similarity measure to transform the instances into a new feature space. In this space each bag is represented by a vector of distances to each instance.

Figure 25 shows the distance calculation of $x^1, \ldots, x^n$ to bag B_1. First for each x^k the distance is calculated to all $x_{1j} \in B_1$ (figure 25(a)), resulting in the following matrix:

$$d = \begin{vmatrix} s(x^1, x_{11}) & \ldots & s(x^n, x_{11}) \\ s(x^1, x_{12}) & \ldots & s(x^n, x_{12}) \\ s(x^1, x_{13}) & \ldots & s(x^n, x_{13}) \end{vmatrix} \tag{1}$$

The distance $s(x^k, B_i)$ is then obtained by taking the minimum of column k in d. Figure 25(b) shows $s(x^k, B_i)$ for a number of instances. After applying the procedure to all bags we obtain a matrix in which each row represents a bag in the new feature space. A regular classification problem is obtained by adding the bag label to each corresponding row.

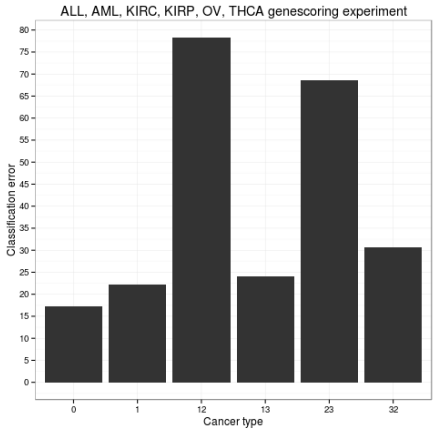
$$m = \begin{vmatrix} s(x^1, B_1) & \ldots & s(x^n, B_1) \\ s(x^1, B_2) & \ldots & s(x^n, B_2) \\ s(x^1, B_3) & \ldots & s(x^n, B_3) \end{vmatrix} \tag{2}$$

Instance selection is performed by finding weights that estimate the contribution of each x^k. A weight of zero means that a particular instance is redundant and can therefore be omitted. MILES uses a 1-norm SVM classifier internally to simultaneously select instances and classify bags. The 1-norm SVM can be formulated as a linear program. This linear program estimates a coefficient for each instance that represents its contribution. An instance is kept if the contribution is above a specified threshold.

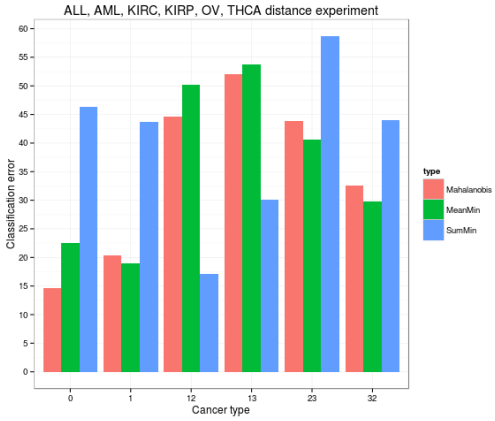
The main shortcoming of MILES is that it estimates a coefficient for every instance. This quickly become infeasible as the number of instances grows. We therefore modify MILES by selecting instances. First we aim to randomly select instances to keep and determine what effect this has on performance. Selection could be improved by using knowledge about the instances. In figure 25(a) for example three round instances have a similar distance to bag B_1 and are in close proximity of each other. Clustering these instances will reduce the number of coefficients from three to one, while still representing the information they represent.

New data results Experiments have been run on the newly selected exome data. Results are shown in figure 26. The distance experiment yields better results for most cancer types. Only OV has a slightly worse classification error when comparing to the genescoring experiment.

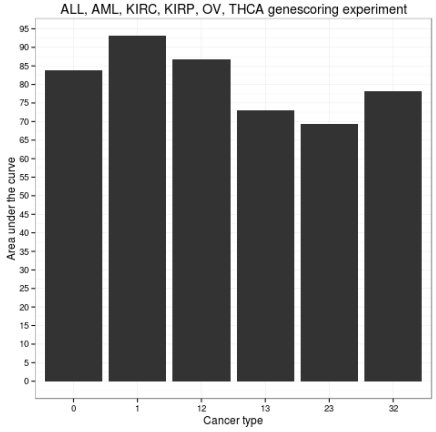
Area under the curve is a little high for a number of classes, possibly indicating that not enough samples are available for these methods. It remains to be seem how well MILES performs on this data.



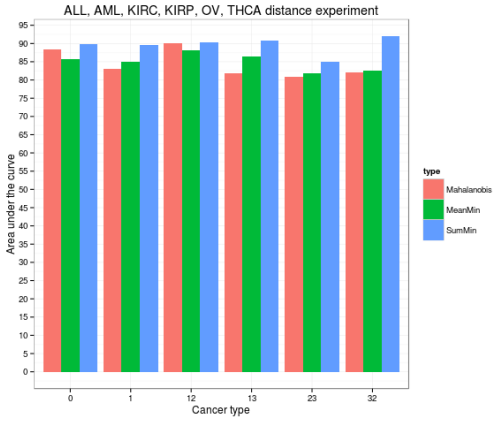
(a) Classification error



(b) Classification error



(c) Area under the curve



(d) Area under the curve

Figure 26: Genescoring and distance experiment results on the ALL, AML, KIRC, KIRP, OV and THCA exome data.

Next week Next week will focus on adapting MILES in order to reduce the number of coefficients estimated. Furthermore, writing will continue.

14 Week 14

Goal This week we extend MILES with two instance selection methods: Random and Rank-based.

Methods MILES has been adapted in order to reduce the number of instances that are considered by the internal 1-norm SVM classifier. As we mentioned last week, the 1-norm SVM aims to estimate a coefficient for each instance that represents the contribution of the particular instance. Estimating these coefficients becomes infeasible as the number of instances grows.

Here we propose two extensions to MILES to reduce the number of coefficients to be estimated: Random Instance Selection and Rank Instance Selection. Both extensions will reduce the number of columns in matrix M (eq 3) from n to m, where $m < n$ (eqs 4 and 7). Furthermore, we add the constraint that m should not be bigger than 30.000 as this appears to be the feasibility cutoff point of our current setup (eq 8).

The matrix M^{red} will subsequently be handed to the 1-norm SVM.

$$M = \begin{vmatrix} s(\boldsymbol{x}^1, B_1) & \dots & s(\boldsymbol{x}^n, B_1) \\ \vdots & \vdots & \vdots \\ s(\boldsymbol{x}^1, B_k) & \dots & s(\boldsymbol{x}^n, B_k) \end{vmatrix} \tag{3}$$

$$M^{red} = \begin{vmatrix} s(\boldsymbol{x}^1, B_1) & \dots & s(\boldsymbol{x}^m, B_1) \\ \vdots & \vdots & \vdots \\ s(\boldsymbol{x}^1, B_k) & \dots & s(\boldsymbol{x}^m, B_k) \end{vmatrix} \tag{4}$$

$$s(\boldsymbol{x}^j, B_i) = min(s(\boldsymbol{x}^j, x_i)), x_i \in B_i \tag{5}$$

$$s(\boldsymbol{x}^j, \boldsymbol{x}_i)) = \sqrt{\sum (\boldsymbol{x}^j - \boldsymbol{x}_i)^2} \tag{6}$$

$$m < n \tag{7}$$

$$m < 30.000 \tag{8}$$

MILES with random instance selection Random selection of instances can be performed at two different stages: *Before* or *after* calculating bag-to-instance distances. Selecting before the distance calculation will alter the distance matrix for each particular bag B_i containing r instances:

$$D_i = \begin{vmatrix} s(\boldsymbol{x}^1, x_{i1}) & \dots & s(\boldsymbol{x}^n, x_{i1}) \\ \vdots & \vdots & \vdots \\ s(\boldsymbol{x}^1, x_{ir_i}) & \dots & s(\boldsymbol{x}^n, x_{ir_i}) \end{vmatrix} \tag{9}$$

$$D_i^{red} = \begin{vmatrix} s(\boldsymbol{x}^1, x_{i1}) & \dots & s(\boldsymbol{x}^m, x_{i1}) \\ \vdots & \vdots & \vdots \\ s(\boldsymbol{x}^1, x_{ir_i}) & \dots & s(\boldsymbol{x}^m, x_{ir_i}) \end{vmatrix} \tag{10}$$

$$m < n \tag{11}$$

Consider the example in figure 27. Removal of instance $i1$ will have a small effect on the distance of some instances to bag B_1. But as there are other instances in B_1 closeby, the difference will be small and will only affect instances in B_2. On the other hand, removal of instance $i2$ will have a much larger effect on the distance of instances in B_1 and B_3. The larger the fraction of instances to be removed is, the more prevalent this effect will be. Bag structure is effectively lost by reducing the number of instances.

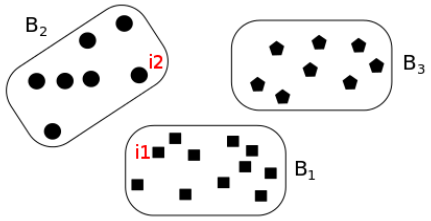


Figure 27: Example MIL problem with three bags and 26 instances.

When removing instances after the bag distances have been calculated bag structure is not lost. Matrix D_i will remain as in equation 9. Removing in M will only effect the number of coefficients to be found, not the distances. But all full matrices D_i and the full M must be calculated before creating M^{red}, which is a large computational burden.

MILES with rank instance selection This extension aims to rank instances according to a particular criterion. Here we propose to calculate a correlation between each column of matrix M and the list of baglabels $\mathbf{l}$:

$$p^i = ttest(\boldsymbol{x}^i, \boldsymbol{l}) \tag{12}$$

$$\boldsymbol{x}^i \in M = [\boldsymbol{x}^1 \dots \boldsymbol{x}^n] \tag{13}$$

$$\boldsymbol{p} = sort([p^1 \dots p^n] \tag{14}$$

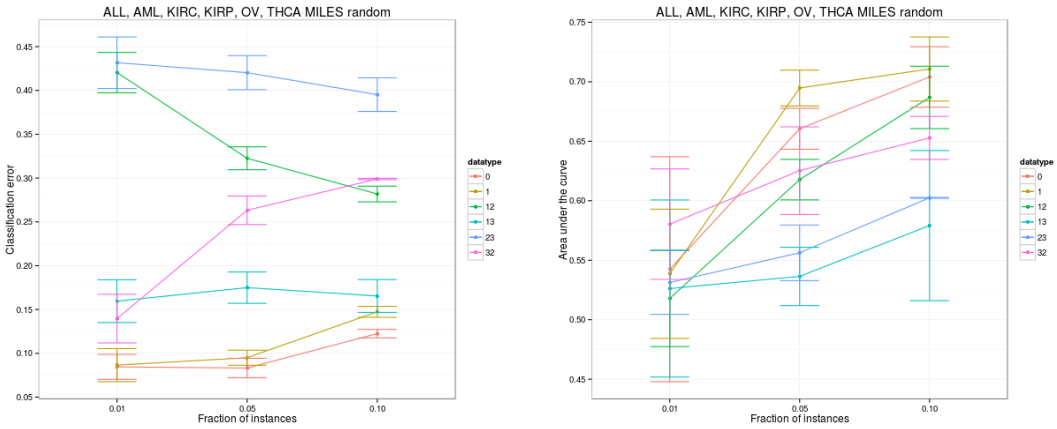
By selecting the m instances of which the p^i appear at the top of $\mathbf{p}$ M is reduced to M^{red}. These m instances are most informative, as they have the highest correlation with the baglabels. The number of instances is therefore reduced, while keeping as much information as possible.

Results

MILES with random and by rank instance selection In this section we describe the results of running MILES with random and rank-based instance selection on the ALL, AML, KIRC, KIRP, OV and THCA exome data. This dataset consists of 67067 instances and 1500 bags.

Type	label	Samples	Mutations
ALL	0	141	1900
AML	1	158	1882
KIRC	12	325	28136
KIRP	13	100	6350
OV	23	472	23264
THCA	32	304	5541
Total		**1500**	**67067**

Table 14: Overview of the contents of the dataset currently in use. All samples are exomes.



(a) (b)

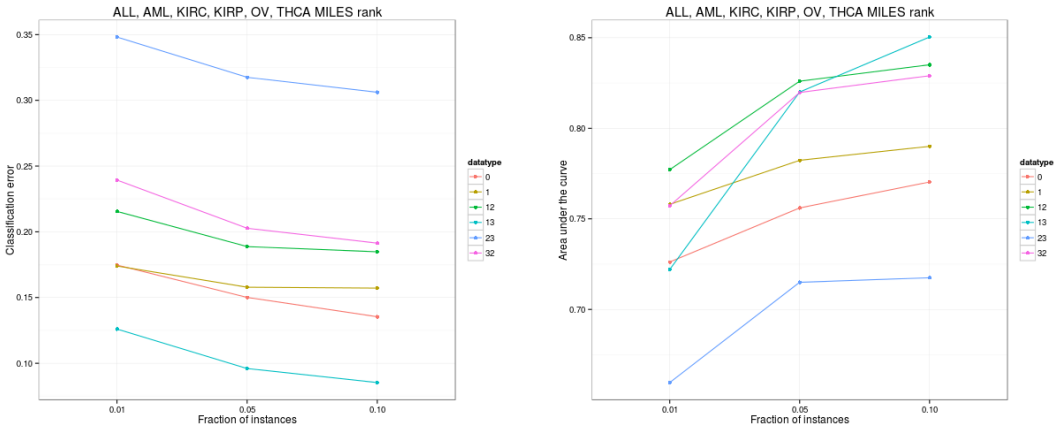
Figure 28: MILES with the random instance selection extension. Instances were removed before the distances were calculated. The results for 0.01 are averaged over 10 runs, for 0.05 over 5 runs and 0.10 over 2 runs.

Figure 28 shows the results of running MILES with random removal of instances before the distances are calculated. Selection of 1% of the instances was repeated 10 times, while selection of 5% was repeated 5 times and selection of 10% was repeated twice.

The classification error for classes 0, 1 and 32 goes up as the percentage increases. This could be an indication of overfitting, likely the result of not

enough instances for those particular classes. The classification error for classes 12 and 23 improves as the percentage increases. These are the two classes that contain substantially more instances and are therefore less likely to overfit when selecting a small percentage of instances. The performance is abysmal nonetheless.

Figure 29 shows the results of running MILES with the ranking extension. Performance clearly increases as the number of instances goes up. Extra experiments with a larger percentage of instances will be interesting, although classification + area under the curve roughly amounts to 1 for all classes when selecting the top 10%. We therefore do not expect a big increase in performance.



(a) (b)

Figure 29: Results of running MILES with the rank extension. Correlation was calculated through a t-test. Instances were then sorted by their p-value, after which the top 1%, 5% or 10% was selected.

Finally, we compare the classification performance to the results from last week in table **??**. Both MILES experiments do not outperform the summin distance experiment, although MILES+rank is not far off. Given the trend visible in figure 29(a) it seems unlikely that increasing the percentage of selected instances will yield a big improvement.

Type	Genescoring (*)	Distance (**)	MILES+random (***)	MILES+rank (*)
ALL	0.09	0.09	0.12	0.14
AML	0.11	0.11	0.15	0.16
KIRC	0.17	0.11	0.28	0.18
KIRP	0.07	0.06	0.17	0.09
OV	0.29	0.23	0.40	0.31
THCA	0.20	0.13	0.30	0.19

Table 15: Overview of the classification error obtained with the experiments until now. (*) = Best overall classification error selected. (**) = Best classifier on summin distance. (***) = Performance when selecting 10% of instances.

Next week It remains unclear what the best attainable classification error is. What performance would MILES have on 100% of the mutations? Next week we will obtain the best attainable classification error for ALL, AML, KIRP and THCA. By temporarily removing KIRP and OV we can run MILES on the full set, giving insight in the best possible performance. The total number of instances in ALL, AML, KIRP and THCA is well below 30.000, which will allow MILES to consider all of them.

It is possible that the above experiment will indicate that there is not much additional performance to be found. We will therefore start with replacing the 1-norm SVM in MILES with a nearest mean classifier. The aim is to investigate whether summarising the structure within each bag and removing the 1-norm SVM that selects instances will increase the performance. We hypothesise that the strictness of MILES when selecting instances has a considerable effect on the performance.

15 Week 15

Goal We investigate the structure of matrix M that contains the distance of each bag to each instance and is calculated within MILES. We also extend the MILES+rank and MILES+random experiments and run the distance experiment using the minmin distance.

Results

Update performance table This week we ran a few additional MILES experiments with larger fractions of data. Table 16 shows that with a larger amount of instances the MILES+rank classifier approaches the performance of genescoring and distance. The figures in the next section reveal that the MILES classification error improves as more data is added.

In the following sections we therefore explore the hypothesis that a substantial number of instances is required to reliably classify bags. First we extend the current MILES+rank and MILES+random experiments showing that performance improves as more instances are added. Then we compare the summin distance with the minmin distance, expecting summin to outperform minmin as it uses all instances. Finally, in the last section we explore the structure within matrix m.

Type	Genescoring (*)	Distance (**)	MILES+random (***)	MILES+rank (****)
ALL	0.09	0.09	0.15	0.14
AML	0.11	0.11	0.20	0.15
KIRC	0.17	0.11	0.22	0.18
KIRP	0.07	0.06	0.12	0.08
OV	0.29	0.23	0.39	0.31
THCA	0.20	0.13	0.29	0.19

Table 16: Overview of the classification error obtained with the experiments until now. (*) = Best overall classification error selected. (**) = Best classifier on summin distance. (***) = Performance when selecting 20% of instances. (****) = Performance when selecting 25% of instances.

Distance experiment extension The summarised experiment results in table 16 reveals that the distance experiment (column three) yields the best performance for each of the six different classes. In some cases well better than the MILES variants. Given the results in the previous section we argue that a large portion of instances is required in order to properly classify bags.

Type	Summin cl.err	Minmin cl.err	Summin AUC	Minmin AUC
ALL	0.10	0.10	0.90	0.88
AML	0.11	0.10	0.90	0.82
KIRC	0.11	0.18	0.90	0.85
KIRP	0.07	0.07	0.82	0.76
OV	0.18	0.30	0.85	0.74
THCA	0.13	0.23	0.92	0.80

Table 17: Comparison between summin and minmin distance for the distance experiment.

We've therefore performed the distance experiment (where a distance between bags is calculated) on the *minmin* distance. Aspect of the minmin distance is that a single instance is responsible for the distance between bags. When comparing with the *summin* distance (where all instances are used) we

expect minmin to show worse results.

Table 17 shows that for KIRC, OV and THCA indeed summin performs better. For ALL, AML and KIRP there is little to no difference, apart from a slightly higher AUC in summin. We note that KIRC, OV and THCA consistently show the worst performance in each of the experiments listed in overview table 16. Furthermore, KIRC and OV are by far the largest in terms of number of mutations, while all three are the largest in terms of number of bags (see table 14 in the previous week).

A clear conclusion at this point is difficult. But it seems that KIRC, OV and THCA require more instances for proper classification than ALL, AML and KIRP.

MILES+rank and MILES+random experiment extension The MILES experiments with extension from last week have been extended with increasing fractions of data. In figure (a) is can be observed that the AML, ALL, KIRC and THCA classification error starts very low and increase for the first few steps. This appears to be linked with the number of available instances as these four have a substantially lower number of instances.

As the fraction of instances increases all four graphs go up. KIRP (6350 instances) levels out the quickest. THCA (5541 instances), ALL (1900) and AML (1882) need until 15% of the instances are selected to reach their maximum. Alternatively both KIRC and OV start out at their maximum and drop as fraction increases. We conclude that for the low fractions the number of instances selected from ALL, AML, KIRP and THCA is too low to properly support classification in the random experiment.

The rank experiment shows different behaviour. As 25% is reached all graphs are fairly stable. First we conclude that ranking is definitely better than random as it produces far more stable results and the best performance at 20 and 25% is always better than random at 20%.

Concluding, the random experiment shows that more instances is better. But the rank experiment shows that some instances are more important that others. By selecting the right instances we might be able to reduce the total substantially a small price in performance (comparing with the summin distance experiment).

Open questions at this point are: Ranking is now done globally across all instances, bags and classes. Will results improve when we rank instances per class and select the top fraction? What will happen when we add additional features to instances? When we add pathways, pfam domains and whether a gene is in the Cancer Gene Census, will the results then improve?

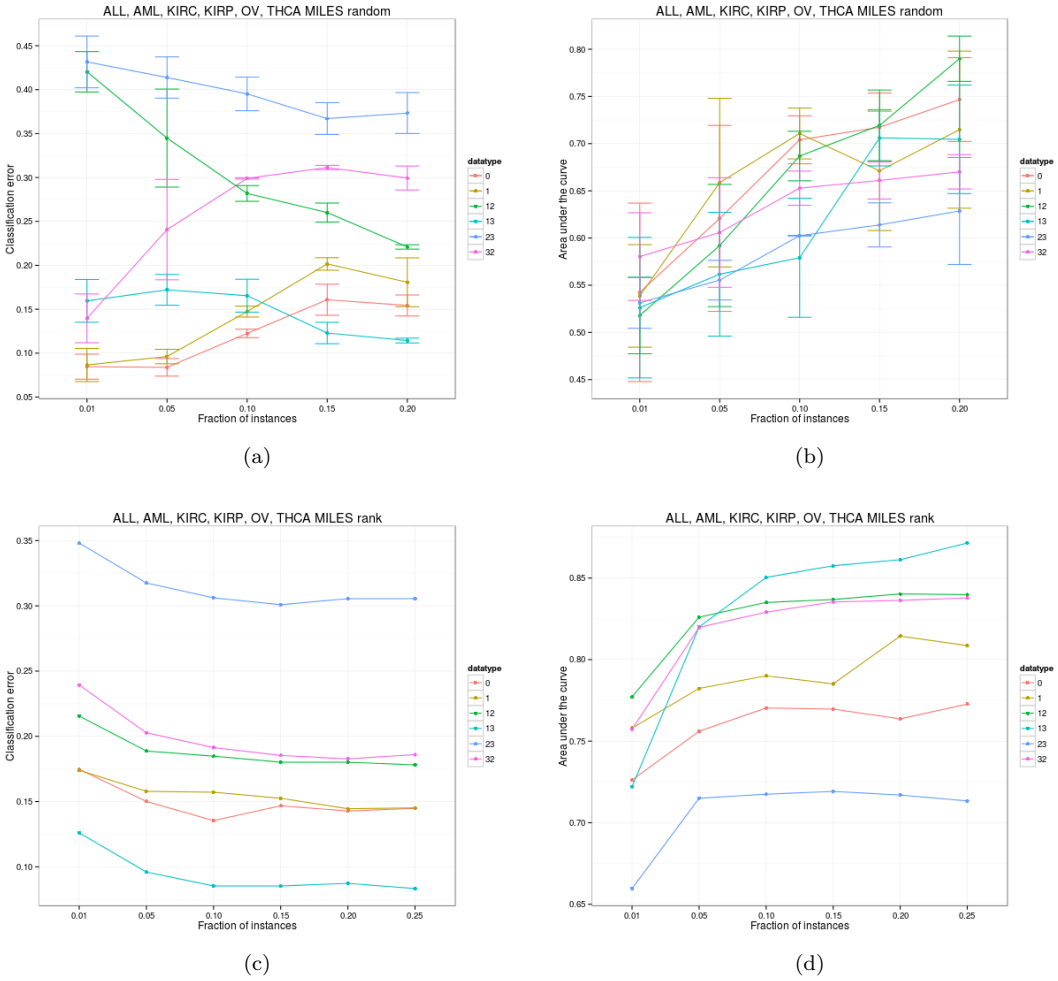


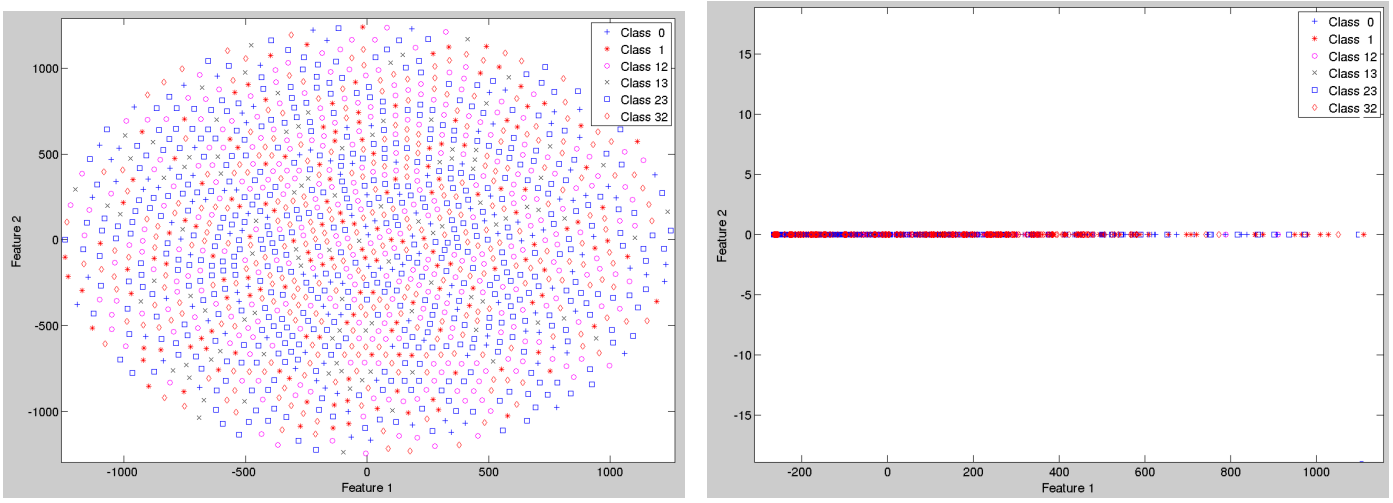
Figure 30: MILES with the random and rank instance selection extension. Ranking is done by performing a t-test for the distance vector for each instance (columns in matrix M) and the baglabels. After sorting the top fraction of instances were selected. Random instances were removed before the distances were calculated. The results for 0.01 are averaged over 10 runs, for 0.05 over 6 runs and 0.10 over 2 runs, 0.15 over 3 and 0.20 over 2.

Bag dimensionality reduction in matrix M In order to explore potential structures within distance matrix M, we reduce its dimensionality to two and plot the results. Figure 31 shows scatter plots for both t-SNE and PCA reduction methods. Both methods have been applied to M that contains a row for each bag and a column for each instance. A bag (1500 in total) is therefore a point in a 67067 dimensional space. The methods mentioned here reduce each

bag to a point in a two dimensional space. M was scaled to its mean with unit variance before running both algorithms.

With the scattering visible in figure (a) it becomes apparent that samples do not cluster together at large, although some groups do seem to appear. With the scattering of class 23 (OV) is it not a surprise that MILES has difficulty finding a decision boundary that achieves a low classification error. The PCA results in figure 31(b) reveal that there is only one direction in which there is variation in this matrix M. Investigation of a scree plot (not shown) confirmed that all PCs are roughly zero, apart from PC 1.

It is surprising that this matrix yields such a good performance with rank and random extensions. It is therefore interesting to see how this matrix changes when selection of instances is performed. We therefore aim to create such a figure next week.



(a) t-SNE

(b) PCA

Figure 31: Dimensionality reduction of the bags in distance matrix M calculated by MILES. (**a**) t-SNE reduction to two dimensions with perplexity of 6. (**b**) PCA dimensionality reduction to two. Inspection of the eigenvectors revealed that apart from the first principal component, all components are roughly zero.

Final remarks

- MILES+ratio has been run, but yields surprising results. This could be due to a faulty implementation. We therefore withhold the results for another week.

- The experiment with only ALL, AML, KIRP and THCA has been run, but yields worse performance with the original MILES than we have obtained

on all six classes with rank extension. Removing data changes the problem, making results incomparable.

- The differences in genescoring results between Matlab+prtools and Python+sklearn have been explained. The scaling in sklearn is different.

Next week

- Fetch selected instances from MILES+rank and run t-SNE and PCA on the changed matrix M.
- Create an inventory of the characteristics of the selected instances. This results in the first full circle experiment that goes back to the original mutations.
- Replace the L1-SVM with a less strict L2-SVM in the MILES+rank experiment to investigate whether the L1 is too strict.
- Perform t-SNE on the transposed matrix M to see if instances cluster together.
- Describe the clustering experiments that have been run this week.
- Investigate whether the C kernel parameter for MILES(+rank) is still appropriate.
- Investigate new MILES+ratio results.

16 Week 16

Goal This week we add the new way of ranking mutations, by means of a ratio. We also describe earlier performed clustering experiments on the MILES distance matrix. A number of other things are work-in-progress, but not up to the point where they can be interpreted/discussed properly.

Methods

MILES+ratio Previously we reported that MILES+rank is only marginally better than MILES+random. MILES+rank uses a ttest between each column of the distance matrix M and the baglabel vector to calculate a p-value for each column. Each column in this matrix corresponds to an instance in the original MIL problem. The p-values therefore allow for ranking of the instances and selection of the instances that correlate most with the labels. But most of the instances end up with a p-value of 0, which leaves the ranking to the way Matlab sorts a vector with equal values: Keeping the original rank it received as input. MILES+rank therefore selects all ALL (label 0) mutations as these are at the top of the list, but very few THCA (label 32).

In this section we introduce a different method of obtaining a ranking: By means of a ratio. Consider the example in figure 32. It contains three positive (green) bags and three negative (red) bags. Three positive instances are labelled with x_1, x_2 and x_3.

We observe that the distance to the positive bags of instances x_1 and x_2, on average, is lower than the average distance to the negative bags. For instance x_3 it is the other way around, average distance to the positive bags is higher than the average distance to negative bags.

But we have more than two classes. In order to obtain a performance measure for each class we use the one-vs-all scheme to build a binary classification problem. This means we would calculate the ratio for x_3 in each of the binary problems. The location of x_3 would make it a bad candidate to support the positive class, but a decent candidate for the negative class (it is located in a high density area for the negative class). This doesn't make sense As x_3 belongs to the positive class and should not support any other.

The ratio is therefore calculated as follows:

$$ratio(x) = \frac{\sum_{i=1}^{n} d(x, B_i)/n}{\sum_{j=1}^{m} d(x, B_j)/m} \tag{15}$$

where x is an instance and $d(x, B)$ is the distance between x and bag B. $B_i, i = 1 \dots n$ are the n bags that have the same label as the bag where x originates from. $B_j, j = 1 \dots m$ are the bags that have a different label. The ratio returned by equation 15 represents the support of x for its *own* class and will not be used as positive support for another class.

For each MILES OvA round the instances will be ranked by their ratio. Instances near the top of the list are supportive of the positive class, while instances near the bottom are supportive of the negative class. As we are interested in interpreting the results in terms of the positive class we only select from the top. A later experiment can also include instances from the bottom to create a maximally separable class.

This ranking scheme is not perfect. The case of two clusters of positive bags and one cluster of negative bags in between will yield poor performance.

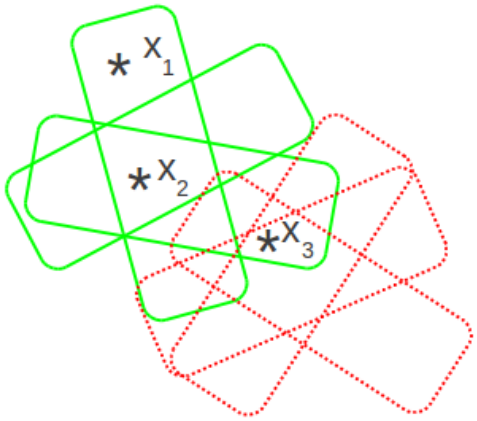


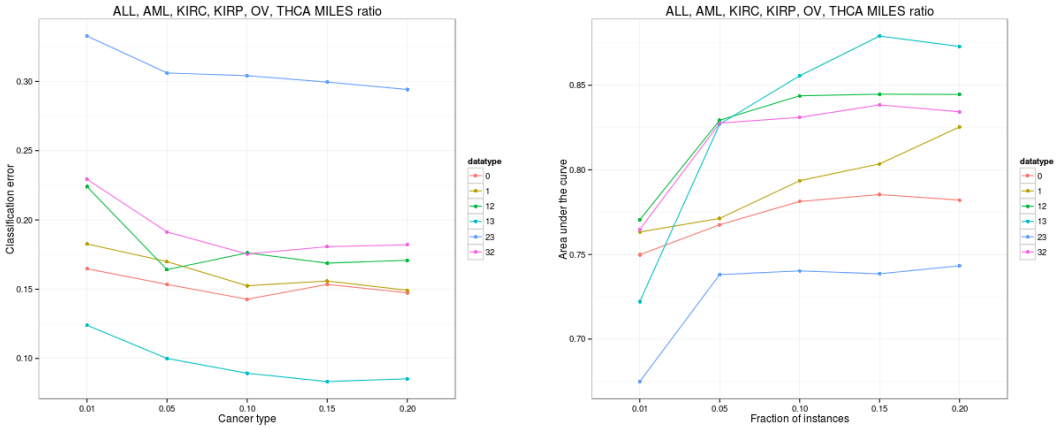
Figure 32: Example containing six bags, three positive (green) and three negative (red). MILES+ratio calculates the ratio of the average distance of an instance to the positive bags, divided by the average distance of that same instance to the negative bags. In this example instance x_2 is more informative than instance x_3, as it falls within the area with the highest positive class density. We expect informative instances therefore to have a low distance to positive bags and a high distance to negative bags. Therefore ranking instances by their ratio pushes informative instances to the top.

Results

MILES+ratio Figure 33 shows the results of MILES+ratio. Table 18 confirms that MILES+ratio is indeed better than MILES+ttest-rank, but not by much. The distance experiment remains the best.

Type	Distance (**)	MILES+random (***)	MILES+rank (****)	MILES+ratio (*)
ALL	0.09	0.15	0.14	0.14
AML	0.11	0.20	0.15	0.15
KIRC	0.11	0.22	0.18	0.16
KIRP	0.06	0.12	0.08	0.08
OV	0.23	0.39	0.31	0.29
THCA	0.13	0.29	0.19	0.18

Table 18: Overview of the classification error obtained with the experiments until now. (*) = Best overall classification error selected. (**) = Best classifier on summin distance. (***) = Performance when selecting 20% of instances. (****) = Performance when selecting 25% of instances.



(a) (b)

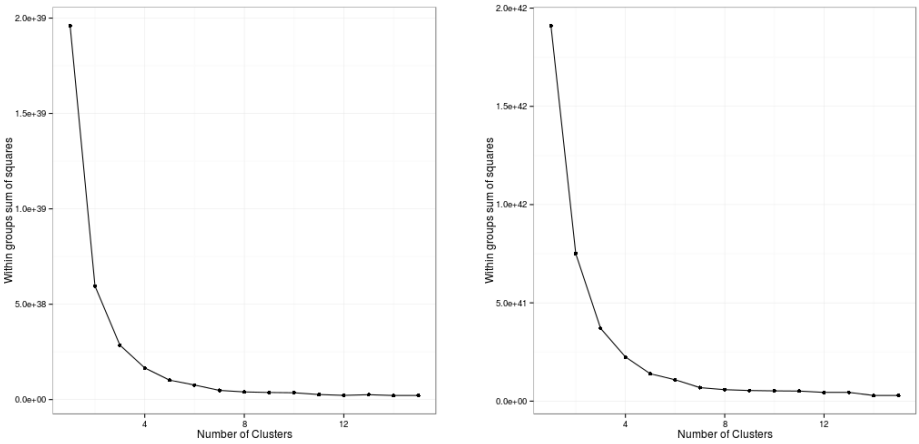
Figure 33: Classification error and AUC for MILES+ratio.

Clustering In order to investigate possible structure in distance matrix M we've performed hierarchical clustering on the bags. First complete-linkage agglomerative hierarchical clustering on M directly and then single-linkage agglomerative hierarchical clustering on a matrix that contains the euclidean distance between each pair of bags. This report doesn't lend itself too well for showing a dendrogram of 1500 items. Both dendrograms show blocks of bags with the same label clustered together, but the blocks are largely small.

We've also performed k-means clustering for $k = 7$ and $k = 11$ based on figure 34. Figure 35 shows the contents of the different clusters obtained. There are no pure clusters, although in most clusters at least one class is not very well represented.

In general, it is difficult to conclude anything from these results. 1500 points do not fill up a 67067 dimensional space very well (matrix M), neither do 1500 points in a 1500 dimensional space (euclidean distance). These results therefore do not conclusively show that there is no structure in the dataset. But the dendrograms and k-means clusters seem to indicate that structure exists at a low level: Many small clusters, sometimes mixed with one or two items of a different class.

In order to investigate further we are currently running the *clust_mil* classifier in the OvA setup. Initial results with $k = 5$, $frac = 0.01$ and a SVC with polynomial kernel (table 19) look promising, but are a little weak in terms of AUC. More experiments are needed.



(a) (b)

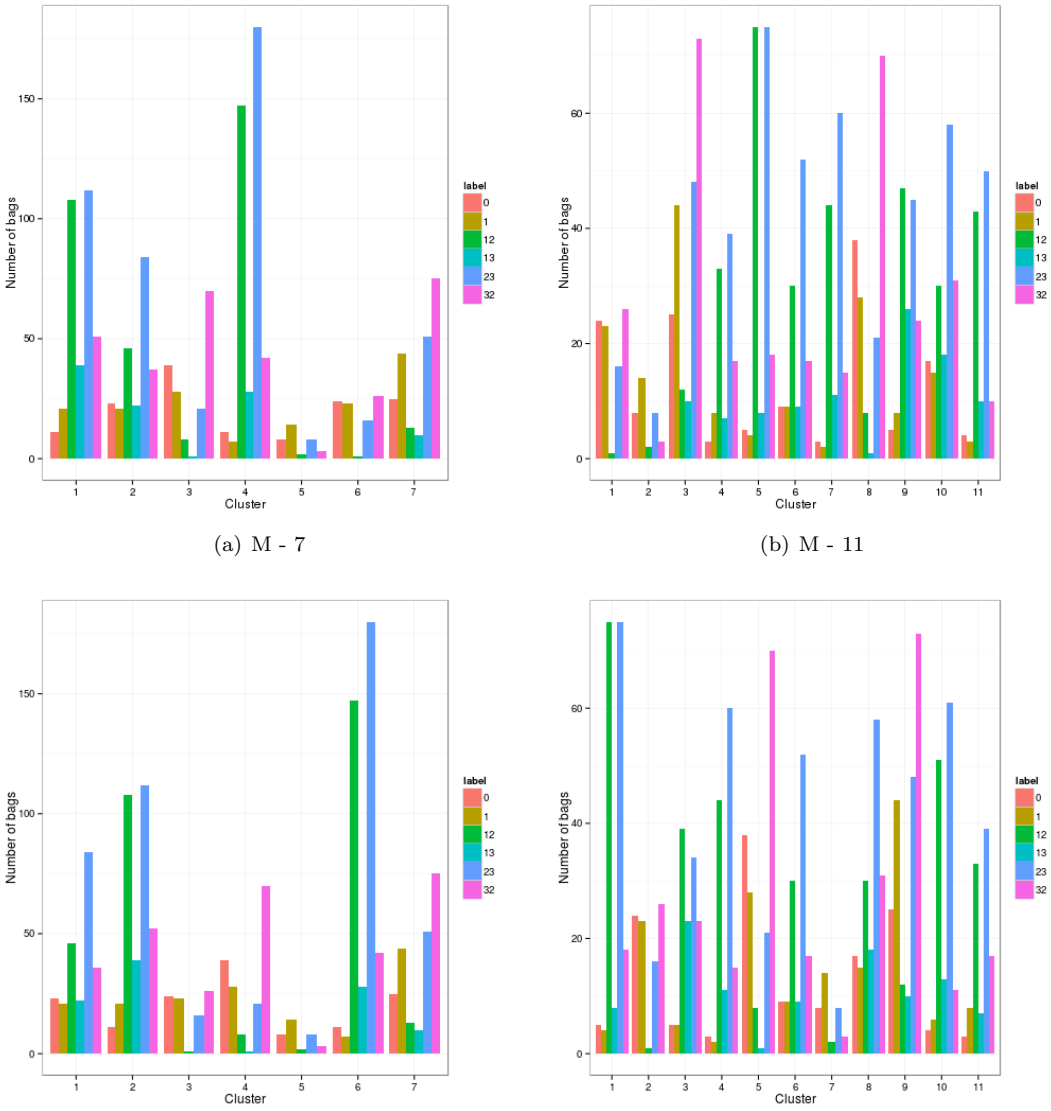
Figure 34: Within group sum of squares after k-means clustering for some values of k.

Class	Label	Cl.err	AUC
ALL	0	0.11	0.81
AML	1	0.13	0.75
KIRC	12	0.24	0.68
KIRP	13	0.10	0.55
OV	23	0.40	0.47
THCA	32	0.24	0.58

Table 19: Results of running clust_mil with $k = 5$, $frac = 0.01$ and a SVC with polynomial kernel.

Final remarks

- There was a bug in the code that extracts the selected instances from MILES experiments. The bug is fixed, but the data has to be regenerated. The experiments are currently running.
- The L2-SVM is build into MILES, but does not work properly yet. This needs attention next week.
- There was no time to investigate the C parameter for MILES.



(a) M - 7

(b) M - 11

(c) Eucl - 7

(d) Eucl - 11

Figure 35: Contents of the clusters obtained through k-means clustering.

Next week

- Create an inventory of the characteristics of the selected instances. This results in the first full circle experiment that goes back to the original mutations.
- Fix the L2-SVM MILES setup.
- Fetch selected instances from MILES+ratio and rerun MILES+rank. Run t-SNE and PCA on the changed matrix M.
- Perform t-SNE on the transposed matrix M to see if instances cluster together.
- Investigate whether the C kernel parameter for MILES(+rank) is still appropriate.

17 Week 17

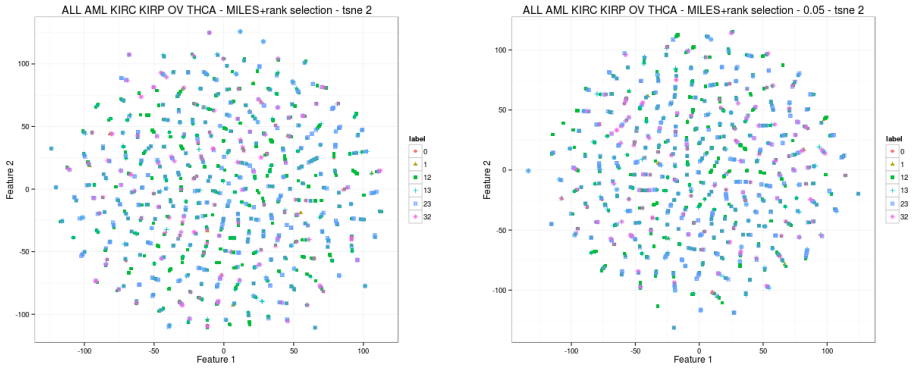
Goal This week we investigate the properties of the selected mutations by MILES+rank by looking at annotations and through t-sne. We also create code to add new annotations.

MILES+rank selection interpretation Removed.

tsne on selected mutations Running t-sne on the selected mutations shows that each class is quite scattered across the first two dimensions. At best there are a large number of small clusters. PCA resulted in all data projected into a diagonal in the first three dimensions, making the data completely inseparable (figures not shown).

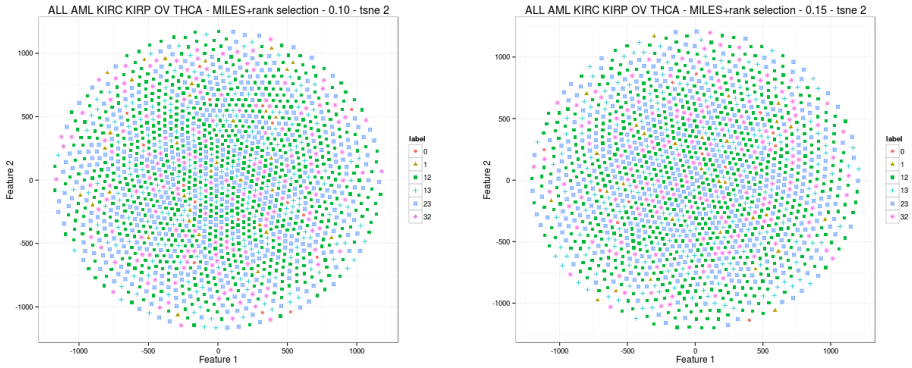
Final remarks

- The promising results from Clust_mil shown last week did not continue as the fraction of instances or the number of clusters goes up.
- Changing the C parameter for MILES+rank with 1% of the instances does not improve the results. Only class 23 (OV) improved slightly, as the others got worse.
- Code has been created to annotate the mutations with reactome pathways, pfam and GO domains. Gene name annotation is also now available for the MILES experiments



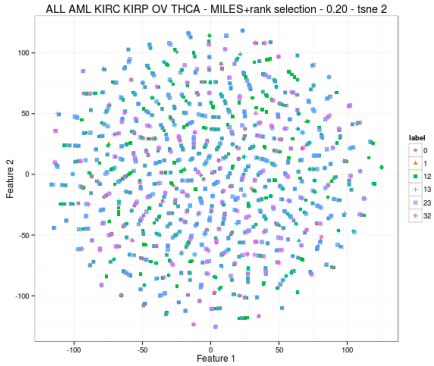
(a)

(b)



(c)

(d)



(e)

Figure 36: t-sne to 2 dimensions for each of the 5 experiments.

Next week Next week we aim to investigate whether classification results improve as more annotations are added. First the annotations must be obtained for all classes. Then we will create 4 experiments, each representing the original annotations + one of the 4 new ones. Each of these experiments will be handed to MILES+rank and MILES+ratio with selection of 1% of mutations. Annotation sets that improve classification results shall be kept. Later on combinations of the 4 can be tried.

The fact that all OvA classifiers depend for a large part on classes 13 and 23 begs the question how well classification will work when one of these is removed. We therefore propose the hierarchical classification scheme in Algorithm 2. If successful this scheme could be transformed into a hierarchical list of binary classifiers. This setup could be advantageous as the number of classes is reduced further down the tree, leaving room for selection of more data for each of the remaining classes. It could also be disadvantageous as structure is lost by removing classes. We again aim to use MILES+rank and MILES+ratio.

Algorithm 2: Pseudocode for the hierarchical setup.

```
Data: A MIL dataset containing 6 classes
Result: A list of classifiers
classifiers = [ ]
for i in 1:5 do
    Train classifier in all classes
    Obtain classification error
    Remove class with best performance
    classifiers[i] = trained classifier
end
```

18 Week 18

Goal This week we reanalyse the results from last week after fixing the bug that rendered last weeks figures useless. We also report on the performance of the hierarchical tree of classifiers.

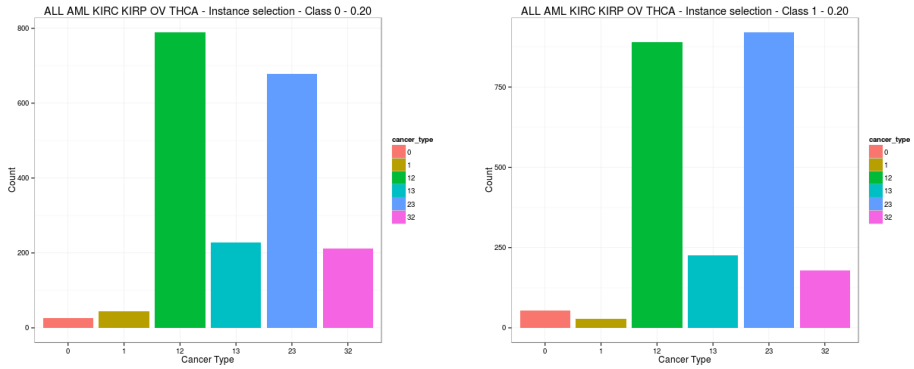
Hierarchical classifiers This classifier consists of five rounds of 10-fold cross-validation. After each round a class is removed, based on earlier obtained results through MILES+rank. First 13 is removed, then 1, followed by 0 and 12. If this method indeed works we would expect to see the lowest classification error for each class right before it is removed. Table 20 shows the classification error, with underlined is the best performance. The expected location for the best performance holds only for class 13. It appears this method throws away too much structure to increase performance.

Type	label	Round 1	Round 2	Round 3	Round 4	Round 5
ALL	0	**0.17**	0.18	0.20		
AML	1	**0.17**	0.18			
KIRC	12	0.22	0.22	**0.20**	0.20	
KIRP	13	**0.13**				
OV	23	0.35	0.34	0.32	**0.31**	0.32
THCA	32	**0.24**	0.25	0.28	0.26	0.33

Table 20: Classification error from the hierarchical classifier test while selecting 1% of instances through the ttest rank.

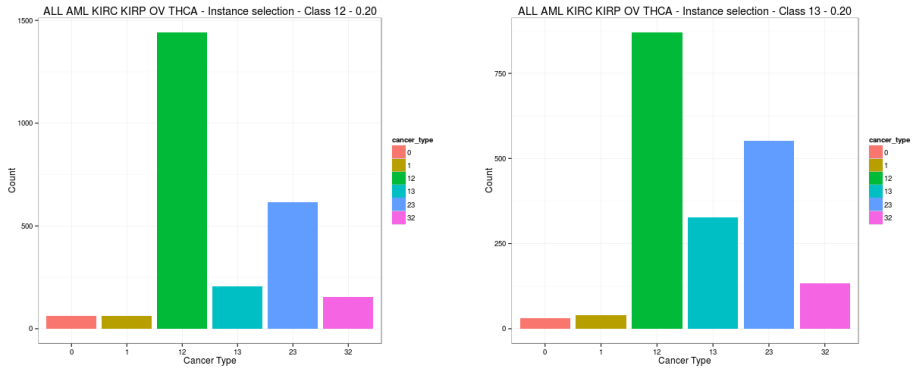
MILES+rank selection interpretation In this section we reanalyse the selected instances with MILES+rank. The experiment where 20% of the instances are selected is used as this seems to yield the most robust results. This experiment consists of six one-versus-all (OvA) experiments in which one class is marked as positive, while the others are marked as negative.

Figure 38 shows how many instances are selected from each class over all six OvA experiments combined. Class 12 (KIRC) and 23 (OV) have many more mutations selected than classes 0 and 1 (ALL, AML). This means mutations from KIRC and OV are higher ranked in a larger abundance.



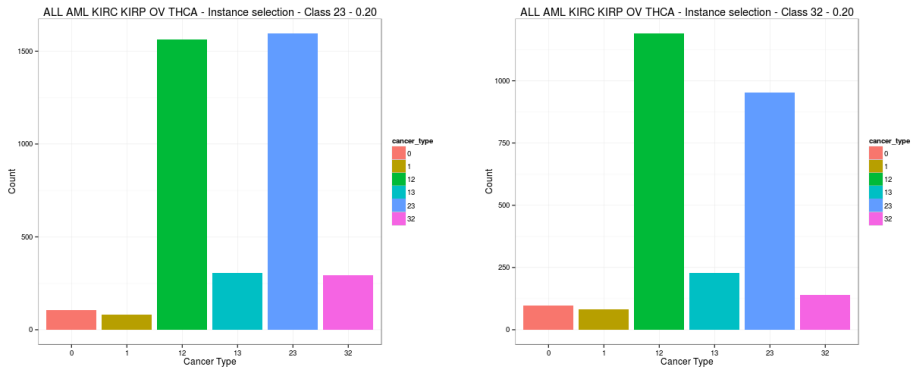
(a)

(b)



(c)

(d)



(e)

(f)

Figure 37: Breakdown of the instance selection per class, per OvA experiment. A subfigure (a for example) shows how many instances of each class are selected when constructing a OvA classifier (subfigure a shows the breakdown for the classifier constructed for class 0, ALL).

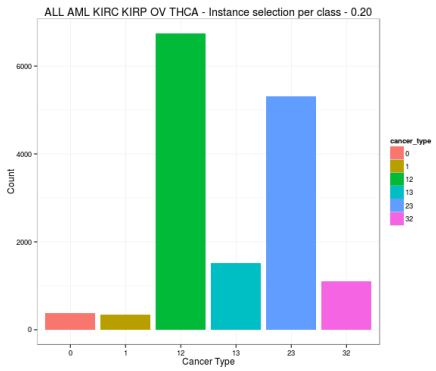


Figure 38: Breakdown of the instance selection per class.

Figure 37 shows the selection per class for each of the six OvA experiments. Both class 12 and 23 have a very strong representation in each subfigure. Subfigure c and e are the only cases where the largest number of selected mutations come from the positive class, where in e the positive class narrowly edges out class 12. From these figures we conclude that classes 12 and 23 are deemed very important by our method. It is important to keep in mind the large difference in the number of mutations selected per class. Below we switch to fraction of the selected mutations per class. A single mutation represents a larger percentage for class 0 than it does for class 12.

Now we move to the annotations of the selection. First the **chromosome** where the mutation resides. Figure 39 shows the fraction of the selected mutations per chromosome, per class. In general overview, it appears that chromosome 4 and 21 (and in part also 22) are not as important as their neighbours. While 7, 12 and 17 seem to stand out. Furthermore, the MT chromosome is deemed important for classes 12 and 23.

A few highlights: Class 12 considers 3 more important than 1 and 2. While, together with class 32 chromosomes 13 and 15 are often selected. For class 0 (ALL) chromosomes 13 and 21 are not selected, but keeping in mind that only around 100 mutations from both class 0 and 1 (AML) are selected we consider both AML and ALL quite similar.

For class 13 (KIRP) chromosomes 1, 2 and 3 are less important than 7, 17, 19, 20 and 22. More than 10% of the selected mutations are on chromosome 17. While for class 23 (OV) 1, 6, but namely X show a large peak. Finally, for class 32 (THCA) chromosomes 6 and 17 are not often selected, while it shows similar behaviour to to class 12 on chromosomes 13 and 15.

We treat the variant effect predictors separately in this section (figure 40). It is important to note that for each predictor the fractions do not add up to one. This is due to predictors not being able to, for example, predict consequences for intergenic regions. In general it is therefore difficult to say anything substantial about class 0, most notably due to the large number of synonymous mutations selected (see figure 41(a)).

PhyloP For all classes except 0, most mutations are conserved (C). For class one both C and N (non-conserved) bars add up to almost one. Further below

figure 41(a) shows that almost all mutations are exonic for this class, which means PhyloP can issue a prediction. This is a lot less so for class 12.

SIFT All, except class 32, have more mutations that are tolerated (T) than damaging (D). For a large portion of some of the classes there is no available prediction (NA) though.

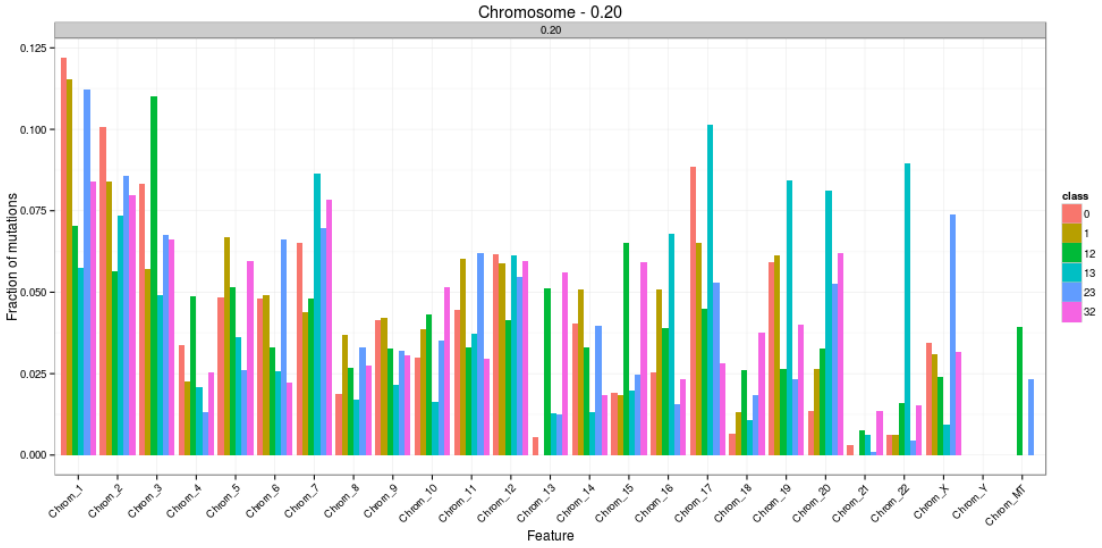


Figure 39: Breakdown of all mutations selected during the 20% experiment per chromosome.

PolyPhen2: Predicts benign (B), possibly damaging (P) and probably damaging (D). Class 13 has more benign mutations than any of the other three categories. A large portion of class 23 and 32 have NA.

LRT: Contains deleterious (D), neutral (N) and unknown (U). Class 32 shows a large number of deleterious predicted mutations.

MutationTaster: Contains disease causing (D), disease causing automatic (A), polymorphic (N) and polymorphic automatic (P). Again class 32 shows a high bar at predicted disease causing.

Now we move to the function annotations, shown in figure 41(a). There are two groups: One that considers the genomic function of the location where the mutation resides termed function, while exonic-function aims to describe the effect on the protein.

Function: Most mutations are in exons. Almost all for class 0, while only half of class 12. Class 12 and 32 show a relatively large number of splice-site

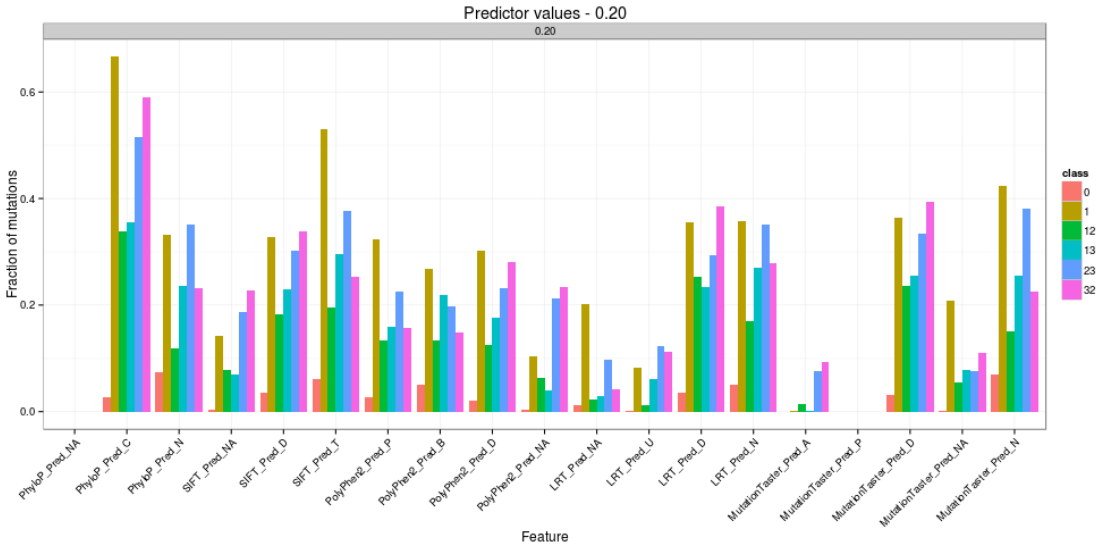


Figure 40: Breakdown of variant effect predictor classifications.

mutations, while both also have a fair number of intronic mutations. Class 0 has many more intergenic mutations than others.

Exonic-function: Most mutations are nonsynonymous, except for class 0 that has many synonymous and frameshift deletion mutations. 23 and 32 have a lot of stopgain mutations.

Figure 41(b) shows the **base** of the mutation found (Alt) and the base of the reference at that same location (Ref). Class 23 and 32 show more C/G to A/T mutations, while the others have more A/T to C/G. Almost 20% of class 0 are deletions, while for class 1 there are none and for 32 it is barely 1%.

Finally, we turn to the **variant effect predictor scores**. A score is assigned only when a predictor is capable of doing so. To fill in the blanks we opted to create a default value for each predictor: $\text{default}(v) = \min(v) - \max(v)$. In each plot the leftmost peak shows the number of mutations that have no value assigned, while the rightmost peak shows the number of mutations with a strong predicted effect. Furthermore, a peak in the middle represents the mutations with no predicted effect.

In general figures 41(c) and 42 show elevated peaks across the whole plot, but all distributions appear to be very similar. It is difficult to conclude anything substantial from these figures.

In summary: The MT chromosome is more often selected than one reasonably would expect (figure 39). ALL and AML show similar behaviour across all chromosomes, while chromosome 7, 17, 19, 20 and 22 are frequently selected for KIRP, but not for KIRC. Chromosome X is often selected for OV.

THCA shows a large number of mutations in conserved regions and therefore is often assigned a damaging prediction by the variant effect predictors (figure 40).

The AML selection consists almost completely out of exonic nonsynonymous mutations (fig 41(a)). ALL on the other hand has more synonymous mutations and frameshift delections. The number of ALL intergenic mutations is also notable. KIRC is the most scattered across all categories, while THCA shows the largest number of stopgain mutations.

For ALL 20% of selected mutations are deletions (figure 41(b)). OV and THCA have more C/G to A/T mutations selected, while the others more A/T to C/G.

Final remarks

- The new annotations are still in the process of being created. Hopefully they are finished before mid next week as the platform they are running on will go into maintenance later on.
- MILES+ratio is almost finished re-running after a bugfix. Performance should not be affected, the fix pertains the selected instances output.

Next week The current MILES setup uses the minimum distance between a bag B and an instance x, which is defined as follows:

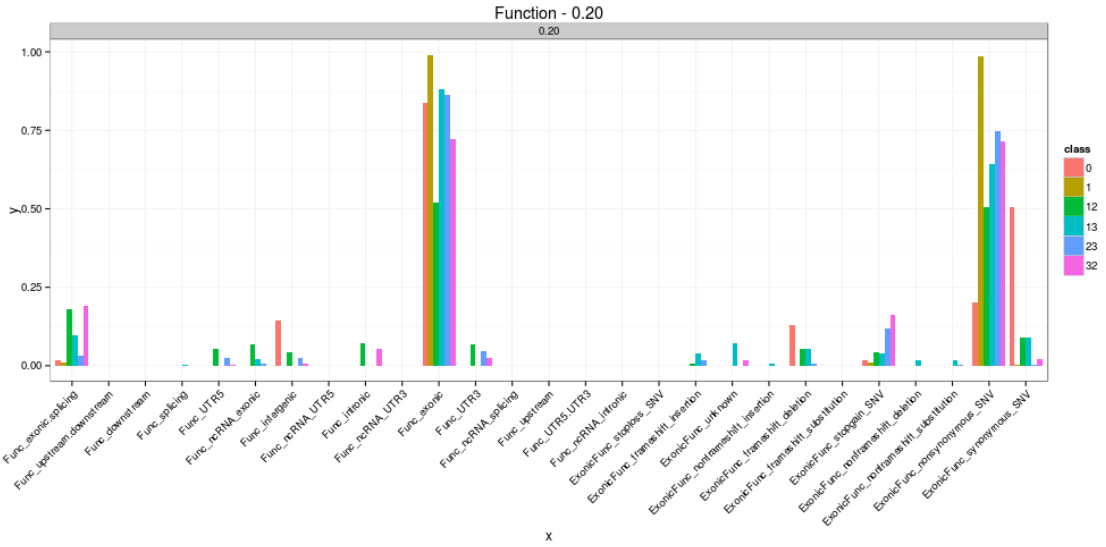
$$D(x, B) = \min\{D(x, x_1), D(x, x_2), \ldots, D(x, x_n)\} \tag{16}$$

with n instances in B. But taking the minimum distance is sensitive to outliers. We therefore aim to investigate whether taking the mean distance improves performance of MILES+ttest (rank):

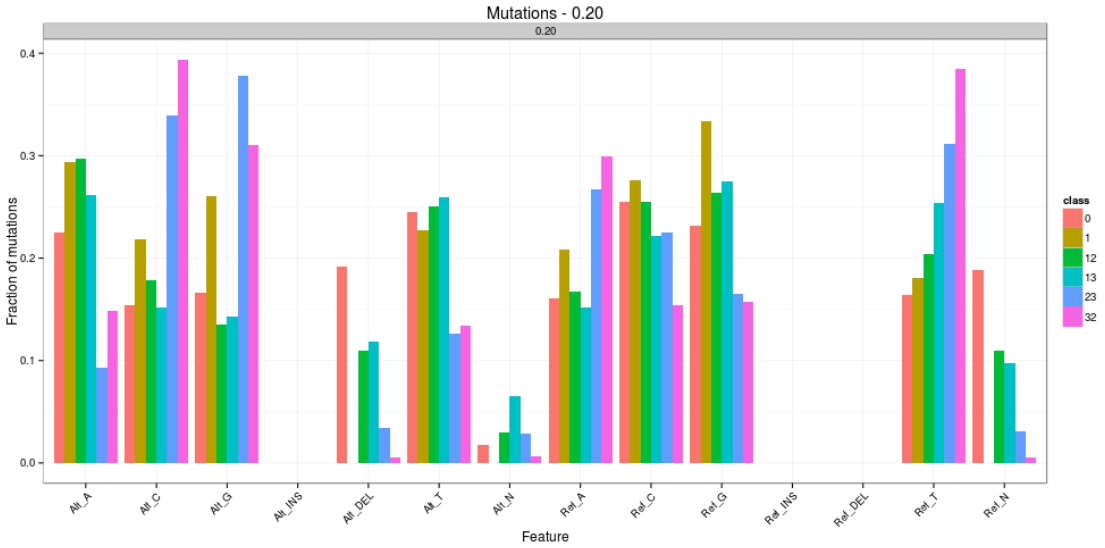
$$D(x, B) = \frac{\sum D(x, x_i)}{n} \tag{17}$$

where n represents the number of instances in B. The advantage of this approach is that each bag is potentially summarised by its centre, rather than a single point on the edge that could be far removed from the majority of instances in that bag.

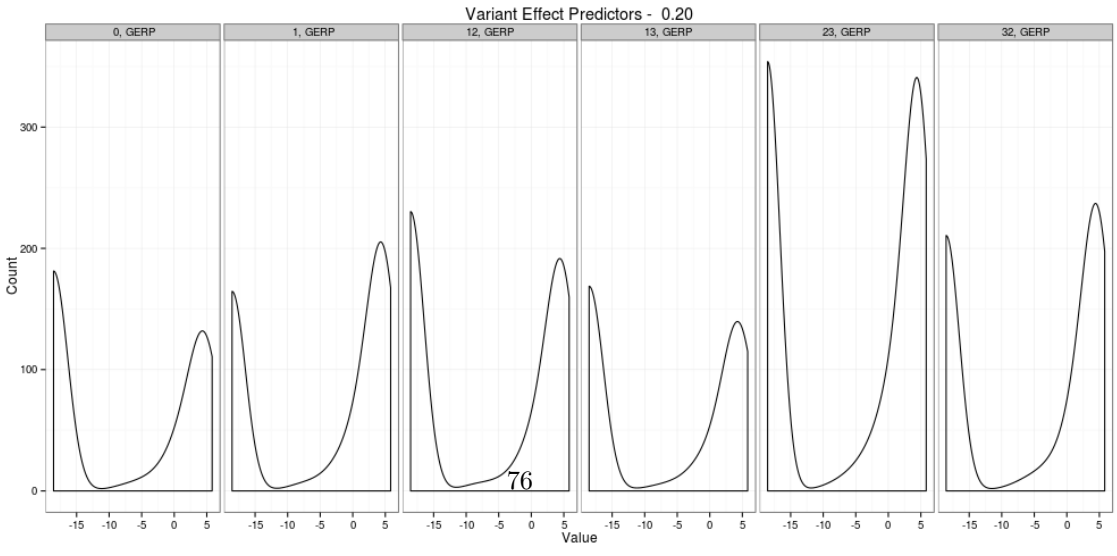
Next week we also aim to investigate the selected mutations through MILES+ratio and the new MILES+corrcoef, while we start adding the new annotations.



(a)



(b)



(c)

Figure 41: Breakdown of various features

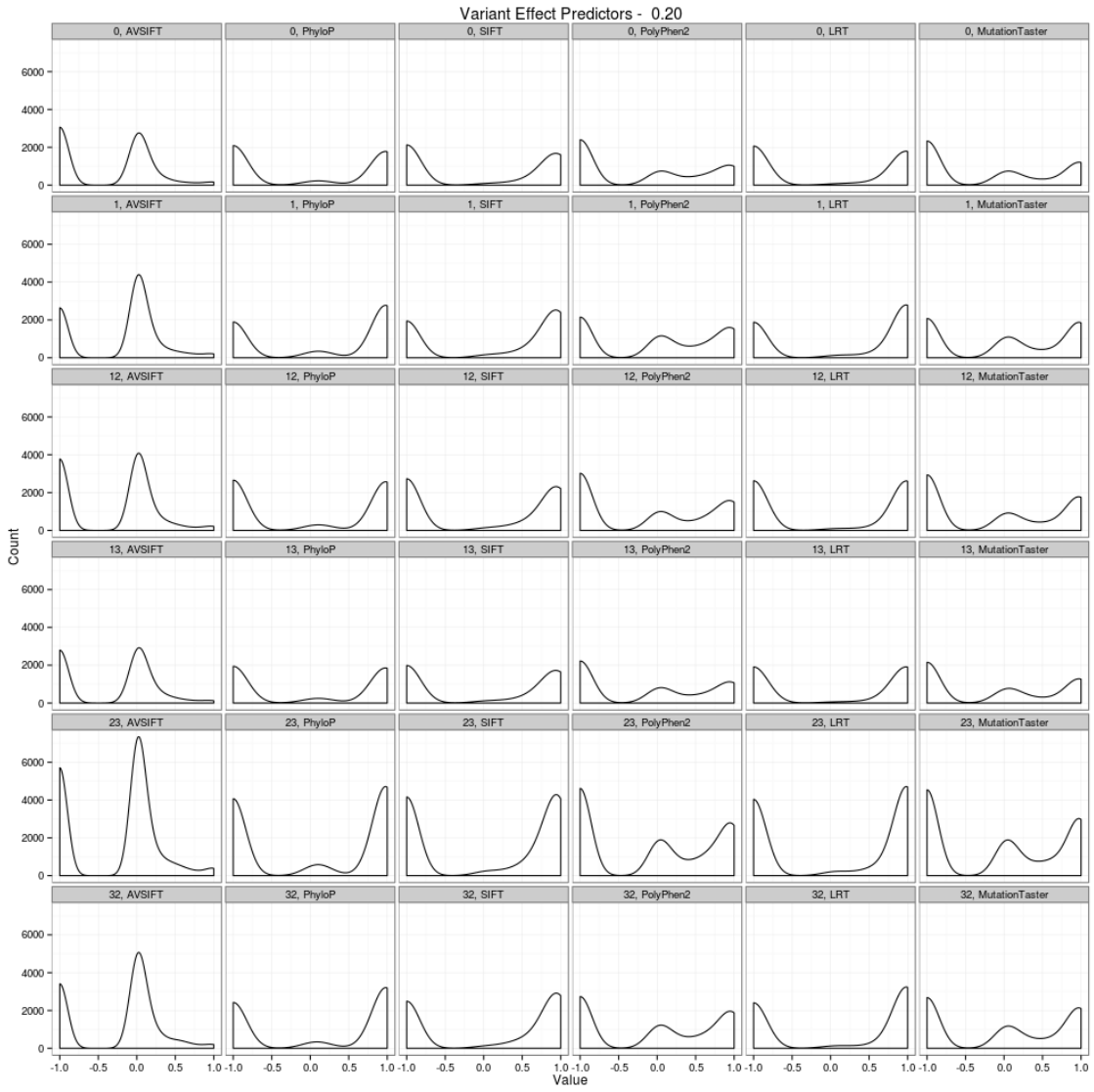


Figure 42: Distributions of various variant effect predictor scores.

19 Week 19

Goal

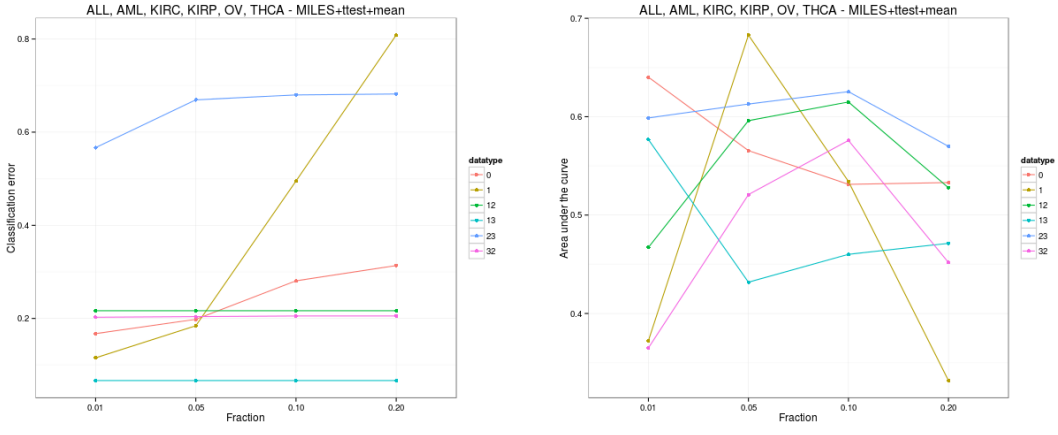
Mean distance in stead of min Within MILES a distance matrix M is calculated between each bag and each instance. The distance between a bag B and an instance x is defined as the minimum distance between x and all instances in B:

$$D(x, B) = \min\{D(x, x_1), D(x, x_2), \ldots, D(x, x_n)\} \tag{18}$$

We have performed an experiment where we replace taking the minimum distance with the mean distance:

$$D(x, B) = \frac{\sum_{i=1}^{n} D(x, x_i)}{n} \tag{19}$$

Figure 43 shows the results. The area under the curve for each of the experiments immediately shows that this adaptation of MILES does not yield any reliable results. This could be because instances in a bag are scattered out, meaning that many bags overlap. Taking the mean distance puts many bags in close proximity which makes it difficult to separate the positive from the negative bags. Having said that, the performance difference with taking the minimum distance is remarkably big.



(a) (b)

Figure 43: Classification error and AUC for MILES+ttest+mean.

MILES with cfunction and positive weights The Matlab code for the MILES+ttest classifier is adapted to work with the new caching module that allows for more transparent caching/loading/saving of parts of the MILES classifier. Old code used to load and save matrix M is replaced.

Thanks to David T a MILES version is now available that selects only those instances that have a positive weight. These are the instances that have a positive impact on the classifier. Allowing for only a positive impact makes interpretation of the selected mutations a bit more straightforward.

Tables 21 and 22 show classification error and area under the curve respectively for MILES+ttest using 1% of the data. The first column contains earlier reported results with the old setup. The second column is MILES+ttest with the addition of caching, while the third column also adds the positive weights restriction.

Adding caching means that scaling of the dataset is now performed on the whole dataset in stead of obtained per cross-validation train set and applied to the corresponding test set. The tables show that the impact on performance when selecting 1% of the data is negligible. Adding the restriction of only positive weights does have a big impact. The drop in performance could be the effect of overfitting. Previously instances with a negative impact were apparently important. We aim to extend this experiment to investigate the performance when more instances are selected.

Type	**label**	**MILES+ttest**	**+cfunction**	**+posweights**
ALL	0	0.17 (0.05)	0.17 (0.05)	0.09 (0.00)
AML	1	0.17 (0.07)	0.18 (0.07)	0.10 (0.00)
KIRC	12	0.22 (0.05)	0.22 (0.05)	0.13 (0.04)
KIRP	13	0.13 (0.04)	0.13 (0.04)	0.07 (0.00)
OV	23	0.35 (0.05)	0.34 (0.05)	0.31 (0.00)
THCA	32	0.24 (0.07)	0.25 (0.07)	0.20 (0.00)

Table 21: Classification error of MILES+ttest using 1% of the data. Between brackets is the standard deviation.

Type	**label**	**MILES+ttest**	**+cfunction**	**+posweights**
ALL	0	0.72 (0.11)	0.71 (0.11)	0.32 (0.14)
AML	1	0.75 (0.08)	0.75 (0.07)	0.25 (0.13)
KIRC	12	0.78 (0.07)	0.78 (0.07)	0.90 (0.10)
KIRP	13	0.72 (0.11)	0.71 (0.07)	0.61 (0.18)
OV	23	0.65 (0.09)	0.66 (0.08)	0.61 (0.15)
THCA	32	0.76 (0.09)	0.75 (0.08)	0.34 (0.14)

Table 22: Area under the curve of MILES+ttest using 1% of the data. Between brackets is the standard deviation.

Final remarks

- The first experiment with additional annotations crashed after two days of running due to a storage disk that temporarily disappeared. No results were obtained. This experiment is now postponed to after the current refactoring phase is finished and better parameters have been found.
- A bug involving counting how often a particular mutation was selected across all the CV splits was fixed. It means the experiments have to be run again, although it will affect neither the performance, nor the interpretation reported last week.

Next week

- Finish the refactoring of the other experiment setups to the new MILES with caching.
- Remove the ranking code from MILES to an external script so it can be run separately.
- Adapt the interpretation figures reported last week so they reflect the log ratio of the reported percentages vs those in the original dataset. This allows us to investigate whether a high number of selected splice-site mutations (for example) corresponds to a high number of these mutations being available in the input. It allows us to see whether the result is due to the data available or due to our selection process. We will also create a separate plot per set of features per class, in stead of the combined plots shown last week.
- We aim to perform a list of experiments that compare the performance for a number of fractions of instances with a number of settings for parameter C. These experiments result in the manual selection of optimal parameters that will be used from then on.

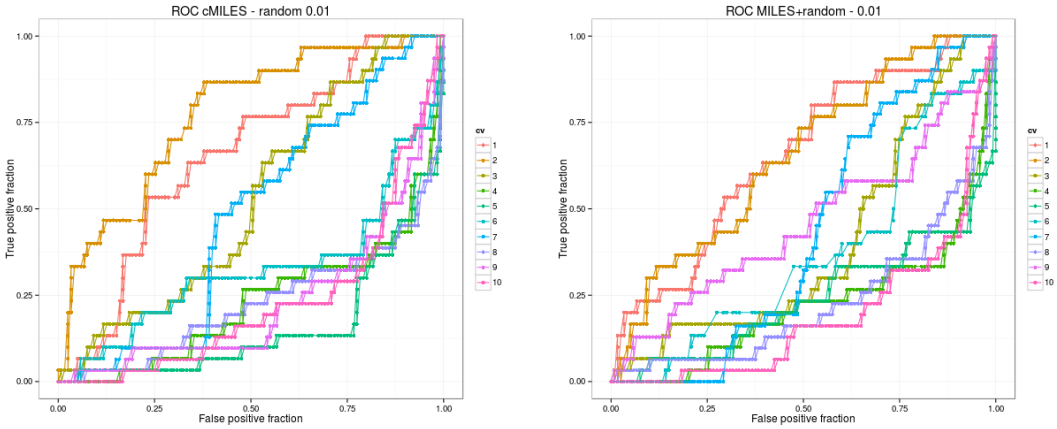
20 Week 20

Goal This week we attempt to add selection of only those mutations that have a positive contribution to the classifier. We also adapt the interpretation figures to show log ratios of the selection fraction vs the overall fraction for a number of annotation categories.

MILES with positive weights Last week we mentioned the poor area under the curve when only positive contributions of instances are allowed. The combined area under the curve and classification error doesn't make sense. This week therefore we further investigate this behaviour.

First we checked that both cMILES and MILES+ implement the same method. On the *reallifemil 103* data both classifiers produce the same results. Then the MILES+ experimental setup was extended such that the ROC curve is saved. Figure 44 shows the ROC curves for each of the 10 cross validation splits for both classifiers on 1% of our six class dataset.

The lines in both plots are so thick and seem slightly blurry because they do not strictly go up. Two points side-by-side on the x-axis should have an increasing or equal y-value. In these figures this is not the case. A y value can actually decrease. This does not make any sense. Whether to conclude that allowing only positive weights simply does not work or that there is something else going on is unclear.



(a) cMILES (b) MILES+random

Figure 44: ROC curves for the 10 cross validation splits. Both cMILES with random selection and MILES+random were run on 1% of the data.

Updated interpretation figures The interpretation figures created in week 19 showed the frequency of mutations of different categories within the selection done by MILES+ttest. But they did not take into account the different class sizes given as input data. We are therefore working on adapting the original figures to log ratio plots. Each annotation in the figures below has a positive value if the annotation occurs more frequently in the selection than it does in the input. When the value is negative the annotation is selected less frequently. The red dotted line depicts a 3 fold difference.

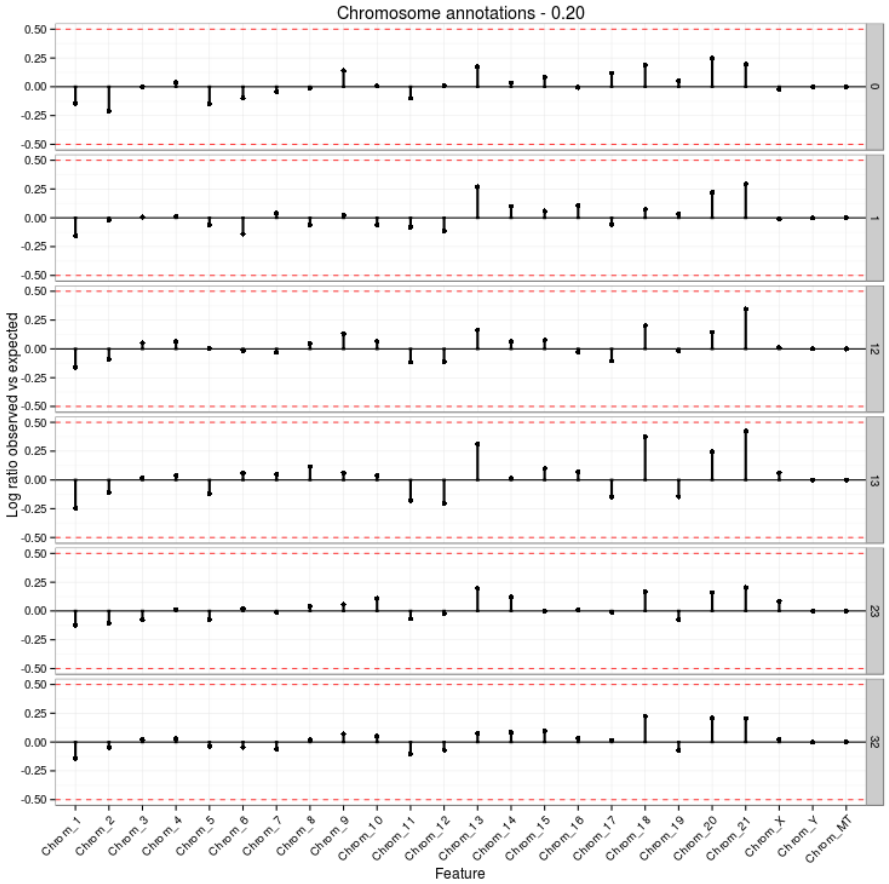


Figure 45: .

Comparing figure 45 with the chromosome plot from two weeks ago: The MT chromosome is not as often selected, when set out against the fraction of MT mutations in the input. The effect on chromosomes 7, 17 and 19 has disappeared as well (eventhough ALL and AML still show a slight increase, while the others a slight decrease on 19). Chromosomes 13, 18, 20 and 21 always have a higher fraction in the selection, while 1, 2, 11 and 12 are less frequently selected. OV (23) still selects more from chromsome X.

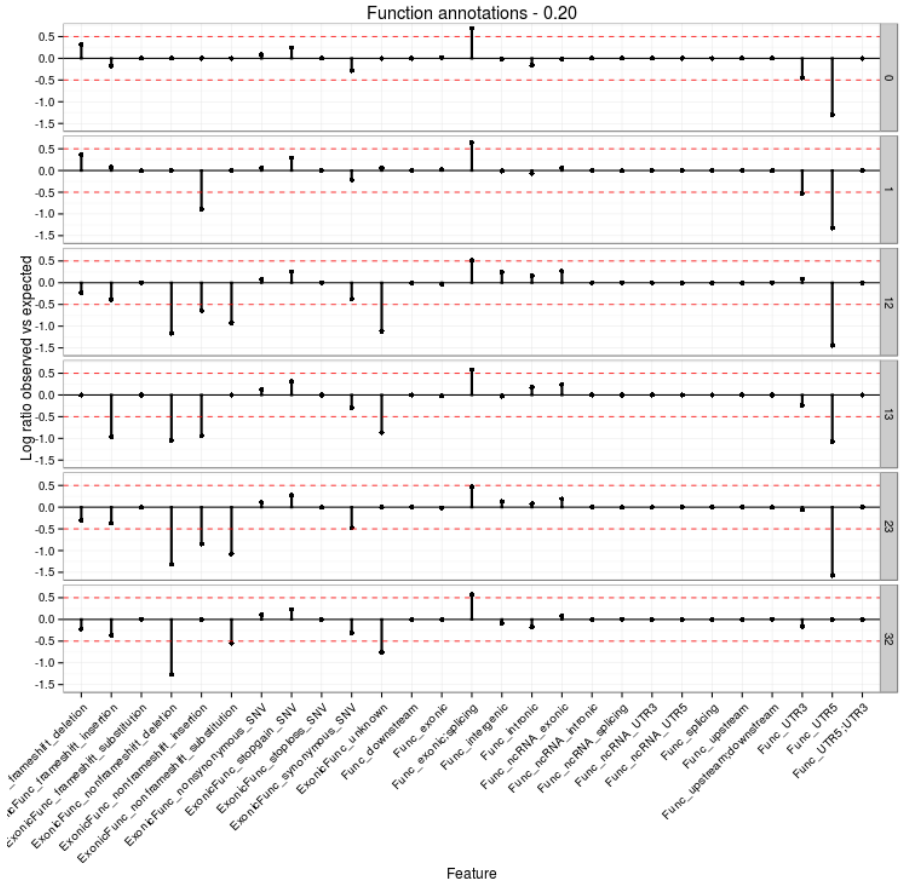


Figure 46: .

Figure 46 shows indeed that ALL and AML select more frameshift deletions (as was the case before). Stopgains and nonsynonymous mutations are also deemed more important by our method. Of interest is the strong selection against nonframeshift deletions (except ALL and AML), nonframeshift insertions and the unknown category. Furthermore, the method deems splice site mutations important, while UTR3′ and UTR5′ are strongly selected against.

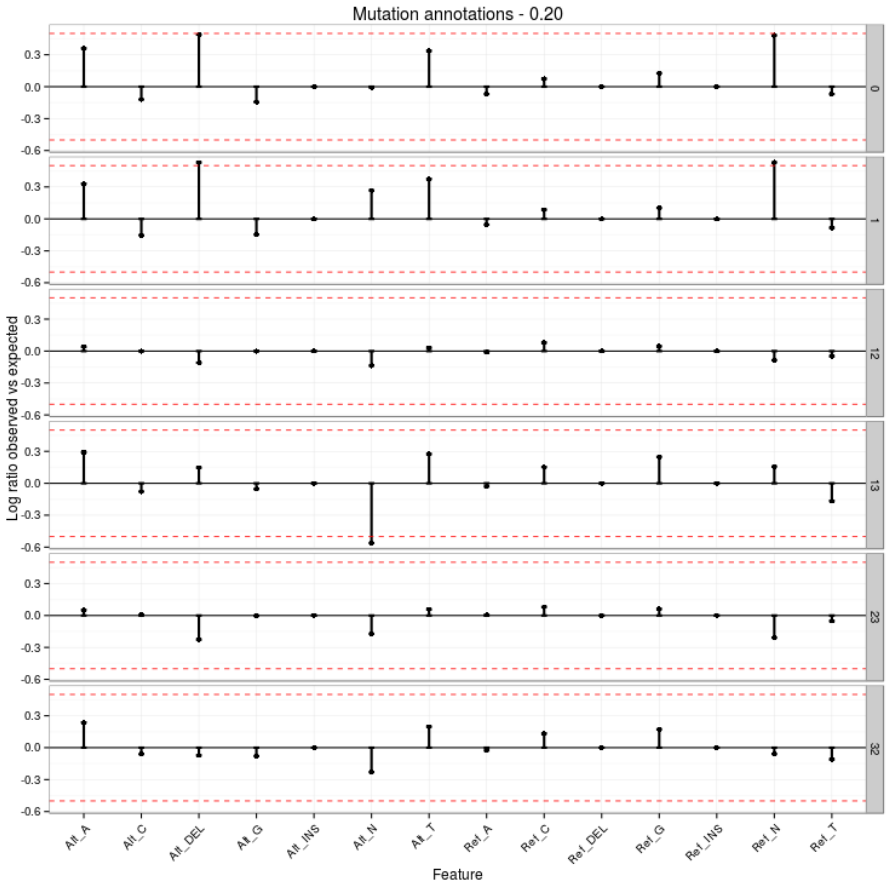


Figure 47: .

Figure 47 shows that A,T > C,G mutations are selected. Previously we concluded that some types select A,T > C,G and others C,G > A,T. This effect has disappeared. Furthermore, it is visible that ALL and AML select deletions as well as mutations with reference base *N*.

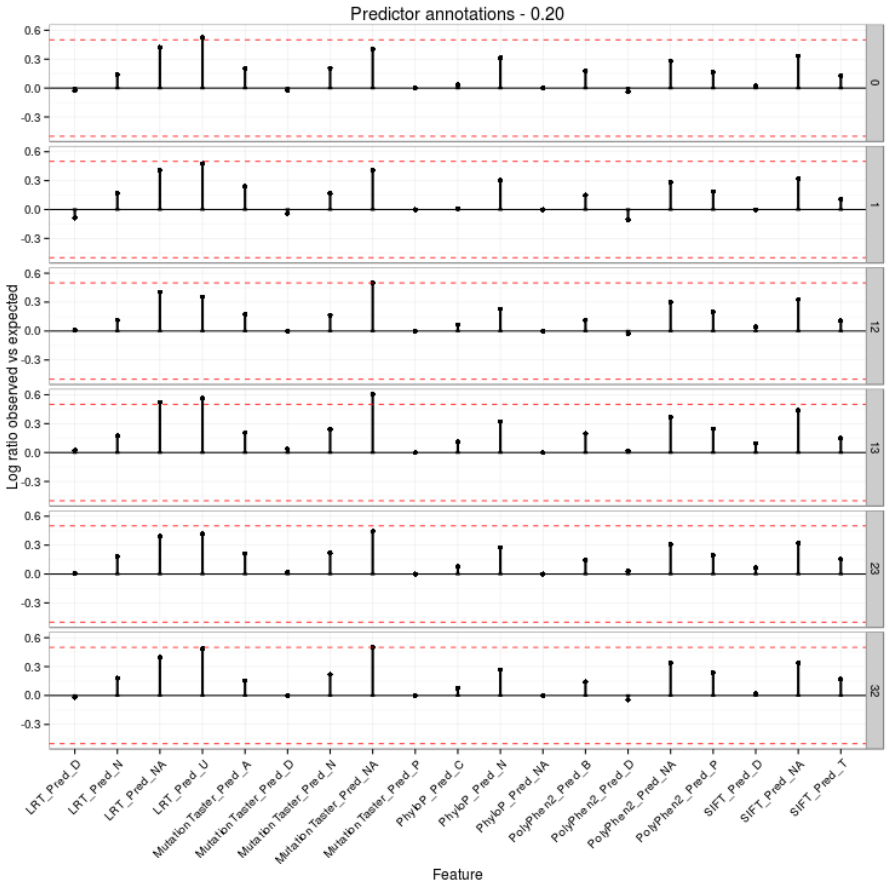


Figure 48: .

Figure 48 shows that mutations with a predictor class are more often selected. Whether the effect is positive or negative is difficult to see. The classes ending with a D are those mutations that are deemed disease causing/detrimental. These mutations are not more often selected. Meanwhile, those classes ending with NA (not available) and U (unknown) are often close to $3\times$ more often selected. Note that there is a difference between no prediction (empty field in the input) and NA (NA in the input). Those mutations with an empty field are not shown in this figure.

Final remarks

- The same figures are available for MILES+ratio. Selection differs in a couple of categories, but a more elaborate comparison should be done to investigate the differences properly.

Next week Most of the items from last week are still open, they are therefore listed again below.

- Finish the refactoring of the other experiment setups to the new MILES with caching.
- Remove the ranking code from MILES to an external script so it can be run separately.
- We aim to perform a list of experiments that compare the performance for a number of fractions of instances with a number of settings for parameter C. These experiments result in the manual selection of optimal parameters that will be used from then on.
- A test statistic should be calculated to compare the log ratios between MILES+ttest and MILES+ratio.

21 Week 21

Goal This week we investigate the effect of changes to our MILES implementation. We add dynamic weighing of the C parameter and test by changing the fraction of selected instances, changing C and reversing the labels. The interpretation figures are again updated to newer specifications.

MILES with positive weights - extended This week we investigate the effect of weighing the C parameter dynamically for each class by the class size. First we ran MILES+ttest with only positive weights, C=1 and fraction is 1% (row one in tables 23 and 24). Then we reversed the labels (row two) and then performed the same two experiments after changing C to 10.000 (rows 3 and 4).

Weighing C dynamically seems to have a small positive effect on the performance. Classification error and area under the curve combined make a bit more sense. Reversing the labels brings classification error and area under the curve combined even closer to what one would expect: $classification\ error + (1 - auc) = 1$.

Next C was increased from 1 to 10.000. This seems to have very little effect on the performance.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.01,C=1	0.88 (0.08)	0.89 (0.0)	0.2 (0.12)	0.4 (0.31)	0.44 (0.22)	0.62 (0.16)
f=0.01,C=1,rev	0.25 (0.21)	0.27 (0.27)	0.78 (0.0)	0.7 (0.17)	0.55 (0.17)	0.34 (0.24)
f=0.01,C=10000	0.83 (0.18)	0.87 (0.07)	0.19 (0.12)	0.39 (0.31)	0.39 (0.28)	0.58 (0.14)
f=0.01,C=10000,rev	0.25 (0.21)	0.26 (0.26)	0.78 (0.0)	0.7 (0.17)	0.55 (0.17)	0.32 (0.25)

Table 23: Classification error of MILES+ttest with positive weights only and weighing C dynamically by class size. 1% of the data was used. Between brackets is the standard deviation.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.01,C=1	0.37 (0.15)	0.36 (0.14)	0.89 (0.1)	0.72 (0.2)	0.62 (0.25)	0.44 (0.08)
f=0.01,C=1,rev	0.87 (0.1)	0.84 (0.17)	0.39 (0.15)	0.48 (0.2)	0.56 (0.23)	0.75 (0.24)
f=0.01,C=10000	0.37 (0.16)	0.36 (0.16)	0.9 (0.09)	0.72 (0.21)	0.65 (0.27)	0.44 (0.09)
f=0.01,C=10000,rev	0.87 (0.09)	0.84 (0.19)	0.36 (0.17)	0.47 (0.21)	0.56 (0.23)	0.76 (0.26)

Table 24: AUC of MILES+ttest with positive weights only and weighing C dynamically by class size. 1% of the data was used. Between brackets is the standard deviation.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.10,C=1	0.88 (0.08)	0.42 (0.16)	0.18 (0.11)	0.28 (0.24)	0.42 (0.24)	0.49 (0.17)
f=0.10,C=1,rev	0.22 (0.18)	0.26 (0.26)	0.78 (0.0)	0.7 (0.17)	0.45 (0.14)	0.32 (0.24)
f=0.10,C=10000	0.78 (0.22)	0.43 (0.18)	0.17 (0.11)	0.27 (0.26)	0.39 (0.28)	0.49 (0.17)
f=0.10,C=10000,rev	0.2 (0.18)	0.26 (0.27)	0.78 (0.0)	0.7 (0.17)	0.44 (0.14)	0.31 (0.24)

Table 25: Classification error of MILES+ttest with positive weights only and weighing C dynamically by class size. 10% of the data was used. Between brackets is the standard deviation.

Type	ALL	AML	KIRC	KIRP	OV	THCA
f=0.10,C=1	0.4 (0.15)	0.52 (0.15)	0.91 (0.08)	0.83 (0.13)	0.64 (0.25)	0.58 (0.12)
f=0.10,C=1,rev	0.89 (0.08)	0.84 (0.19)	0.47 (0.04)	0.48 (0.19)	0.63 (0.22)	0.77 (0.23)
f=0.10,C=10000	0.4 (0.15)	0.53 (0.14)	0.91 (0.08)	0.83 (0.14)	0.66 (0.26)	0.58 (0.12)
f=0.10,C=10000,rev	0.89 (0.08)	0.84 (0.19)	0.38 (0.14)	0.48 (0.19)	0.63 (0.22)	0.77 (0.25)

Table 26: AUC of MILES+ttest with positive weights only and weighing C dynamically by class size. 10% of the data was used. Between brackets is the standard deviation.

After obtaining these results we wondered what would happen if we select more instances. Tables 25 and 26 show the results of the four previously described experiments, but now with selection of 10% of the instances.

When comparing row one with row one of the previous tables we see the expected behaviour that classification error goes down, while the AUC goes up when more instances are selected. Of interest is the AML classifier. Before it showed an error of 89% with 0 standard deviation. This is an indication that simply all AML bags were classified as negative bags. Adding more data removes this effect.

Reversing the labels again seems to have a positive effect on performance. But for both KIRC and KIRP classification error is very high. KIRC also has 0 standard deviation. Adding more data worked for solving the standard deviation issue for the AML classifier, perhaps this is also the case for KIRC. High classification error remains.

Changing the C parameter from 1 to 10000 again has very little effect.

It will be interesting to see how the performance is affected when even more data is selected. As we're now allowing for positive weights only we can effectively select more instances. Another option is to create a new dataset that consists of those instances that are selected by at least x cross validation splits. An instance that is selected 9 times is arguably more important than an instance that is selected only once. Running this new dataset without the ranking could have a positive effect on performance. We assume here that those instances that are selected once-or-twice are selected by chance and are therefore noise.

Updated interpretation figures The interpretation figures have been updated. An example is shown below. All figures now show a $log2$ ratio, in stead of the $log10$ from last week. Furthermore, figures now distinguish between entries that are not available in the dataset as a whole or are not available in the selection. A *square* means no single mutation has this entry, for this annotation in the whole dataset. A *triangle* represents the case where it is possible to select a mutation with this entry, but the classifier has failed to do so. An example of the former is Func_empty in figure 49, while an example of the latter is Func_UTR5-UTR3.

Each annotation now contains an additional entry, which is denoted with the suffix *_empty*. This category accounts for the fraction of annotations of a particular type that have no value. The empty entry is created in such a way that the fractions of an annotation always add up to one. This means, for example, that the ExonicFunc annotation (which is described by 11 features in

our dataset) has an additional entry in the figures with a value such:

$$1 = \sum_i f(x_i) \tag{20}$$

where $f(x_i)$ is the fraction of mutations that have a certain entry x_i and $i = 1 \dots n$ are the possible entries for a certain annotation.

A full investigation of annotations shall again be performed once the new MILES with positive weights setup performs satisfactory.

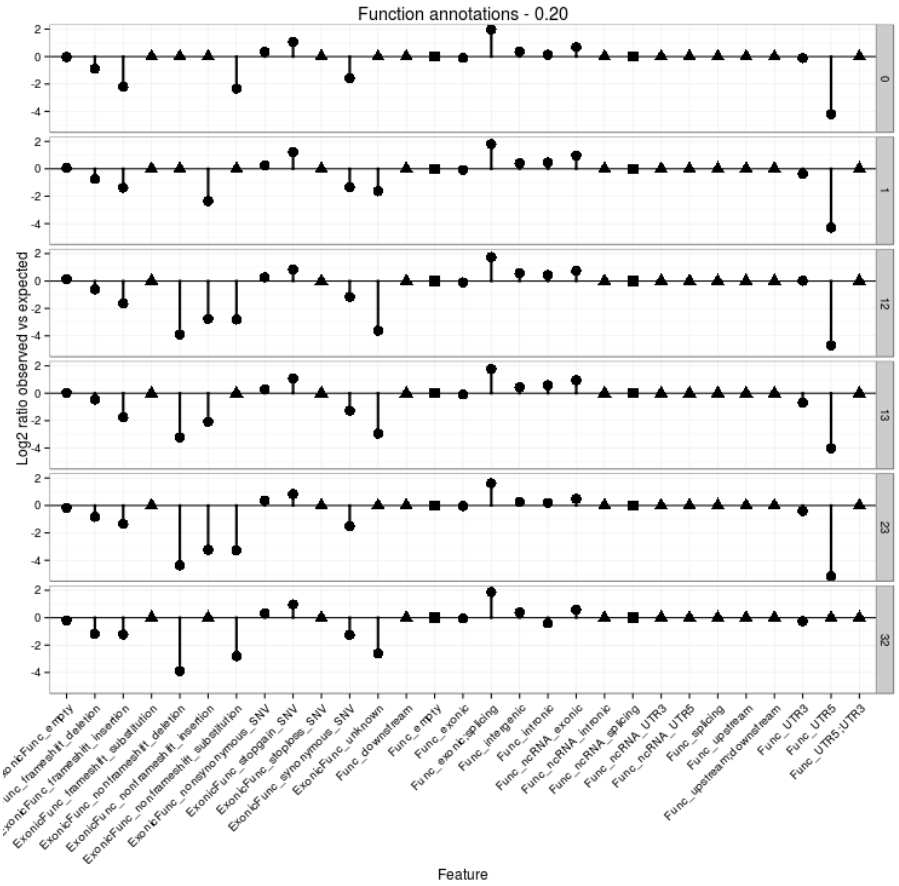


Figure 49: .

Next week After a week of holiday the project will be continued with the following items:

- Increase selection percentage of the new MILES setup and test the effect on the KIRC classifier.

- Create a new dataset consisting of those instances that are selected more than x times and check the effect on classifier performance without ranking.
- Finish the refactoring of the other experiment setups to the new MILES with caching.
- A test statistic should be calculated to compare the log ratios between MILES+ttest and MILES+ratio.

22 Week 22

Goal This week we further extend the positive weights investigation and update the annotation investigation figures.

MILES with positive weights - extended Last week we mentioned that adding more data might have a positive effect on the performance of our classifier with positive weights. The tables below show the classification error and area under the curve from MILES+ttest+dynamic-c+posweights both labels in the correct order and reversed with 20% of the data. In all cases C was set to 10.000, as this consistently yielded the lowest classification error and highest AUC.

For five classifiers, there is no real difference in performance between the experiments with 10% and 20% of the data. Only the KIRC classifier has changed and now has a standard deviation. The forward experiment has a plausible performance, but the reverse doesn't. Why the AUC is so low is unclear.

Furthermore, we note here that for AML, KIRC, OV and THCA the classification error of forward and reverse combined doesn't add up to one. One reason for this could be that all samples of the original positive class are predicted to be negative, which would result in a classification error that equals the class prior. Table 29 shows these priors, of which none correspond to the obtained performance during this experiment.

We conclude here that for some classes the regular labeling works best, while for others the reverse is better. Doubling the data from 10% to 20% does not result in much better performance. The only variable not set is whether to use the ttest or ratio ranking. We therefore propose an experiment with the following parameters to pick a winner.

- Ranking ttest & ratio
- Fraction: 10%
- C: 10.000

- dynamic-C: True
- posweights: True
- reverse-labels: True & False

In order to investigate the structure of the data and possibly uncovering why four classifiers yield unexpected classification performance, we are also going to run experiments in the 1-vs-1 setup (where each class is compared to each other class one-on-one). Next week we'll run all five experiments of KIRC to get an idea. The above settings will be used.

Type	**ALL**	**AML**	**KIRC**	**KIRP**	**OV**	**THCA**
f=0.20,C=10000	0.78 (0.22)	0.43 (0.18)	0.17 (0.11)	0.27 (0.26)	0.38 (0.29)	0.49 (0.17)
f=0.20,C=10000,rev	0.20 (0.19)	0.25 (0.26)	0.44 (0.29)	0.70 (0.17)	0.45 (0.14)	0.31 (0.24)

Table 27: Classification error of MILES+ttest with positive weights only and weighing C dynamically by class size. 20% of the data was used. Between brackets is the standard deviation.

Type	**ALL**	**AML**	**KIRC**	**KIRP**	**OV**	**THCA**
f=0.20,C=10000	0.38 (0.15)	0.53 (0.14)	0.91 (0.08)	0.82 (0.14)	0.67 (0.26)	0.58 (0.13)
f=0.20,C=10000,rev	0.89 (0.07)	0.85 (0.19)	0.19 (0.15)	0.46 (0.20)	0.63 (0.21)	0.77 (0.25)

Table 28: AUC of MILES+ttest with positive weights only and weighing C dynamically by class size. 20% of the data was used. Between brackets is the standard deviation.

Type	**Label**	**Samples**	**Percentage**
ALL	0	141	9.4%
AML	1	158	10.5%
KIRC	12	325	21.7%
KIRP	13	100	6.7%
OV	23	472	31.5%
THCA	32	304	20.3%
Total		1500	100%

Table 29: Sample contribution in percentages of the six classes in our dataset.

Updated interpretation figures - extended The problem with earlier figures (like 49 from last week) is that they are difficult to interpret and they combine annotations from instances selected from positive and negative bags. Figure 50 shows a heatmap example that displays the same information as in figure 49 last week.

Each of the six classifiers is now presented as a vertical pillar consisting of two columns. The left column shows annotations of instances from negative bags, the right those of positive bags (so in the first pillar positive is ALL, while negative is AML, KIRC, KIRP, OV and THCA). Values range from red (less than expected given by chance) through gray into green (more than expected given by chance).

Each annotation is encoded in a number of features. In figures 49 and 50 the annotation is denoted as *Func* and *ExonicFunc* and the feature as text after the underscore. For each annotation the value is calculated as follows:

$$v_i^{pos} = \log 2 \frac{\sum x_i^{pos}/N_i^{pos}}{\sum x_j^{pos}/N_i^{pos}} \tag{21}$$

which is calculated per annotation i. The $\sum x_i^{pos}$ represents the total number of *selected* mutations with a particular feature in the positive data and N_i^{pos} is the *total* number of mutations in the positive data with this feature. For the positive class we therefore obtain a value that shows how often instances are selected with a particular feature, given how many instances with that feature are available in the positive data. The same method is applied to obtain values for the negative class.

Finally, in the figure white represents the case where no instances were selected with that annotation (eventhough it is possible), while yellow represents the case where no instances were available with that annotation (and selection is therefore impossible).

Final remarks

- The class prior of ALL and AML in table 29 comes very close to the classification error for both classes in the genescoring and distance experiments. This should be examined.
- A small bug in counting the selected instances was fixed. This has no effect on the presented results so far.

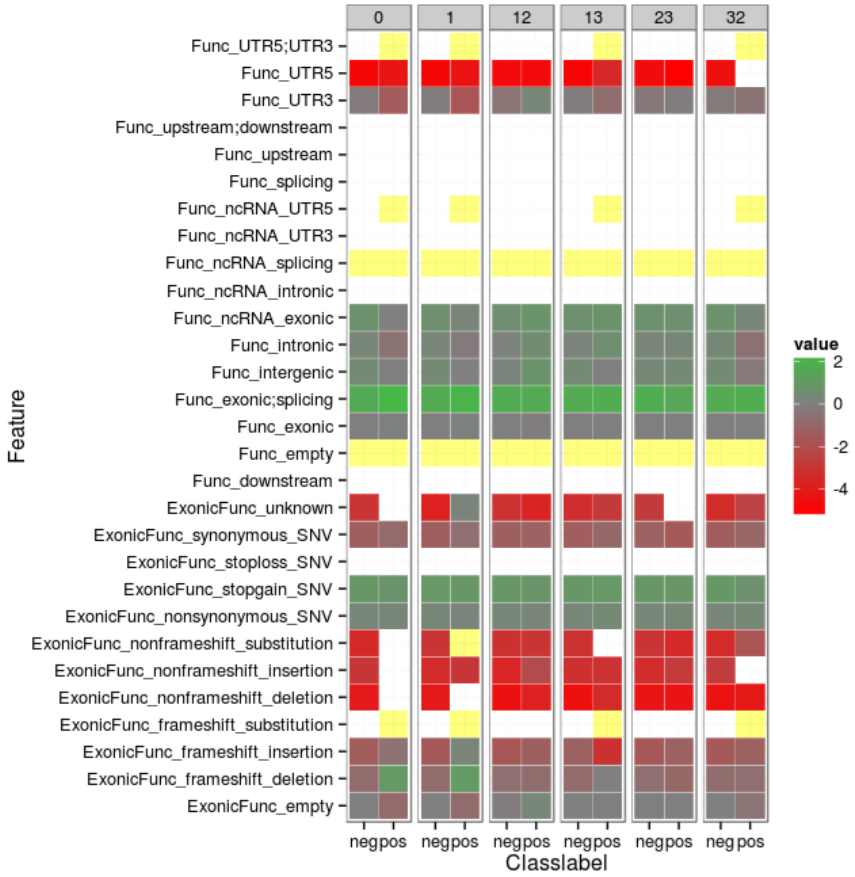


Figure 50: .

Next week After a week of holiday the project will be continued with the following items:

- Perform 1-vs-1 experiments for KIRC.
- Further develop the new interpretation figures.
- Run the MILES+ratio experiment with the above settings.
- Create a new dataset consisting of those instances that are selected more than x times and check the effect on classifier performance without ranking.
- Finish the refactoring of the other experiment setups to the new MILES with caching.
- A test statistic should be calculated to compare the log ratios between MILES+ttest and MILES+ratio.

23 Week 23

Goal This week we run a number of 1-vs-1 experiments, namely those for the KIRC class.

MILES with positive weights - extended This week a number of 1-vs-1 experiments have been performed with the following settings. It turns out that reversing the labels indeed returns the same two classifiers and is therefore not required.

- Ranking ttest & ratio
- Fraction: 10%
- C: 10.000
- dynamic-C: True
- posweights: True
- reverse-labels: True & False

The tables below show the classification error and AUC for both ttest and ratio ranking. In each of these tables the rows are the class marked as positive, while the columns represent the class marked as negative. It is interesting to see that when KIRC is the negative class the classifier performs poorly. What seems to be happening is that each other class falls within the space occupied by KIRC.

In fact, further investigation (not shown) has revealed that all classifiers depend largely on instances from the KIRC class (at most 3 instances from the other class are selected). This means that any interpretation of this selection in terms of biology is meaningless.

It will be interesting to continue these experiments and fill out the table. The full table gives insight into which class falls within which class.

Type	ALL	AML	KIRC	KIRP	OV	THCA
ALL			0.70 (0.00)			
AML			0.67 (0.01)			
KIRC	0.08 (0.07)	0.07 (0.12)		0.37 (0.12)	0.23 (0.08)	0.09 (0.09)
KIRP			0.48 (0.08)			
OV			0.44 (0.08)			
THCA			0.51 (0.01)			

Table 30: Classification error of the 1-vs-1 experiments. Results obtained through the ttest ranking.

Type	ALL	AML	KIRC	KIRP	OV	THCA
ALL			0.50 (0.00)			
AML			0.50 (0.00)			
KIRC	0.97 (0.03)	0.96 (0.10)		0.65 (0.12)	0.85 (0.10)	0.95 (0.10)
KIRP			0.52 (0.08)			
OV			0.32 (0.13)			
THCA			0.37 (0.12)			

Table 31: AUC of the 1-vs-1 experiments. Results obtained through the ttest ranking.

Type	ALL	AML	KIRC	KIRP	OV	THCA
ALL			0.70 (0.00)			
AML			0.32 (0.04)			
KIRC	0.07 (0.11)	0.06 (0.14)		0.25 (0.11)	0.18 (0.09)	0.08 (0.09)
KIRP			0.34 (0.11)			
OV			0.46 (0.09)			
THCA			0.47 (0.02)			

Table 32: Classification error of the 1-vs-1 experiments. Results obtained through the ratio ranking.

Type	ALL	AML	KIRC	KIRP	OV	THCA
ALL			0.50 (0.00)			
AML			0.36 (0.11)			
KIRC	0.98 (0.05)	0.96 (0.12)		0.80 (0.13)	0.89 (0.10)	0.95 (0.11)
KIRP			0.61 (0.11)			
OV			0.41 (0.10)			
THCA			0.27 (0.09)			

Table 33: AUC of the 1-vs-1 experiments. Results obtained through the ratio ranking.

Earlier obtained results For reference we show the table produced in week 14 with the performance of a number of experiments.

Type	Genescoring (*)	Distance (**)	MILES+random (***)	MILES+ttest (*)
ALL	0.09	0.09	0.12	0.14
AML	0.11	0.11	0.15	0.16
KIRC	0.17	0.11	0.28	0.18
KIRP	0.07	0.06	0.17	0.09
OV	0.29	0.23	0.40	0.31
THCA	0.20	0.13	0.30	0.19

Table 34: Overview of the classification error obtained with the experiments until now. (*) = Best overall classification error selected. (**) = Best classifier on summin distance. (***) = Performance when selecting 10% of instances.

Next week Next week will be spend on writing for the final report. A number of sections have roughly been established, but could do with a bit more work. The introduction and main story of the report however will receive the most attention.

A presentation is created and shall be given at the Delft DBL lab meeting next Tuesday.

24 Week 24

Goal This week the aim is to develop the main story for the final report. Work further continues by integrating already written sections and creating new text that describes the experiments.

Final report writing A number of sections of the final report now have a first version, which are attached to this document. The text is still very rough and needs figures and examples.

MILES++ with pathway features The six class dataset has been extended with naive pathway features. First mutations have been mapped into Ensembl gene IDs. Then each gene ID is mapped into a reactome pathway ID. Each available pathway ID is added as a new binary feature to the dataset. A field is 1 when a mutation falls within a gene in a particular pathway, 0 otherwise. This annotation is naive as this considers pathways to be independent cellular processes, when in fact they are not.

These experiments were performed earlier in week 21. When comparing classification error and AUC with the results on the 10% experiments run earlier

there is no real difference. Some classifiers perform slightly better, others are slightly worse. There doesn't seem to be a big advantage in adding this type of features to our dataset with the current setup.

There does appear to be a small issue. Currently the classification error of MILES+posweights contains low values for ALL/AML, while reverse contains high values. Previously this used the be the other way round. This is most likely a bug introduced during extension of the experimental setup for this new dataset.

Type	**MILES+selection**	**+posweights**	**+reverse**
ALL	0.12 (0.04)	0.21 (0.18)	0.83 (0.18)
AML	0.16 (0.07)	0.26 (0.26)	0.40 (0.16)
KIRC	0.18 (0.07)	0.73 (0.18)	0.16 (0.11)
KIRP	0.09 (0.04)	0.70 (0.17)	0.26 (0.25)
OV	0.29 (0.07)	0.44 (0.13)	0.41 (0.21)
THCA	0.17 (0.05)	0.30 (0.24)	0.49 (0.17)

Table 35: Classification error of MILES+ttest, MILES+ttest+posweights and MILES+ttest+posweights+reverse for the six class problem extended with naive pathway annotations. In each case 10% of the data is selected.

Type	**MILES+selection**	**+posweights**	**+reverse**
ALL	0.83 (0.08)	0.90 (0.06)	0.31 (0.15)
AML	0.77 (0.09)	0.86 (0.17)	0.53 (0.17)
KIRC	0.85 (0.07)	0.42 (0.12)	0.91 (0.08)
KIRP	0.84 (0.09)	0.45 (0.20)	0.83 (0.14)
OV	0.74 (0.08)	0.64 (0.21)	0.67 (0.20)
THCA	0.85 (0.06)	0.78 (0.23)	0.59 (0.12)

Table 36: AUC of MILES+ttest, MILES+ttest+posweights and MILES+ttest+posweights+reverse for the six class problem extended with naive pathway annotations. In each case 10% of the data is selected.

Next week A better way must be found to show the performance of our classifiers. At least the performance must be shown for the positive and negative class separately for each classifier. Also, taking the average error over the 10 folds should be reconsidered when a large standard deviation is obtained.

Sections for the final report to write:

- Methods for distance and MIL classifiers.

- Expansion of the methods section with examples. Examples possibly in combination with introduction.
- Driver mutations argument must be further developed by reading papers.
- Fix references properly.

25 Week 25

Goal The goal is to continue writing the final report. A number of experiments are run on the side.

Final report writing - encore The methods section is written this week and references are properly fixed.

MILES++ vs MILES++ with Lasso The experimental setup is now extended to give the classification error for both positive and negative class. Table 37 shows the results for the MILES++ classifier with 10% of the data, C=10.000 and allowing for both positive and negative weights. The classifier always has a low error rate on the negative class, while the positive class is barely better than random. This is not very surprising as we know that for the posweights experiments we saw a low performance as well. Note that these experiments now by default include shiftop with equal error rate and setting of class priors to 0.5.

As a comparison the L1-SVM is replaced by Lasso. An experiment with the exact same parameters (although C is not used) yields the results in table 38. It is interesting to see that eventhough the negative class is always better again, there is a lot less difference between negative and positive. The AUCs are close for both experiments. The main advantage of Lasso is that it is far less computationally intense than the linear programming L1-SVM. The disadvantage is that it is restricted to estimating both positive and negative weights.

Our experiments give as output a vector per OvA classifier that contains an integer per mutation. This integer represents how often that particular mutation is selected by each of the 10 cross validation rounds. Figures 51(a) and 51(b) show the distribution of the values in the vector for both L1-SVM and Lasso. For brevity we've only included the figures of class 0 (ALL). The distribution is different between L1-SVM and Lasso as well as the total number of mutations selected. Lasso selects far fewer mutations, but selects them more often.

Figures 51(c) and 51(d) show how many mutations are selected from each class in the ALL classifier. Clearly both classifiers are more dependent upon the first three classes. Again we see that far fewer mutations are selected by Lasso.

Classifiers for the other five classes show a similar behaviour where similar observations can be made.

MILES in Python A rough implementation of MILES in Python is created. Basic code that works in principle exists. The structure of the code should be improved and simplified. Then a full comparison with the Matlab implementation should be performed.

Type	Cl.err neg	Cl.err pos	AUC
ALL	0.08 (0.03)	0.68 (0.14)	0.76 (0.08)
AML	0.09 (0.08)	0.60 (0.15)	0.80 (0.10)
KIRC	0.10 (0.04)	0.40 (0.12)	0.83 (0.06)
KIRP	0.05 (0.04)	0.66 (0.17)	0.84 (0.06)
OV	0.17 (0.05)	0.46 (0.06)	0.72 (0.07)
THCA	0.10 (0.05)	0.43 (0.15)	0.82 (0.07)

Table 37: Classification error and auc of MILES+ttest with the internal L1-SVM classifier. 10% of the data is selected, the operating point is shifted to an equal error rate and class priors have been set to 0.5.

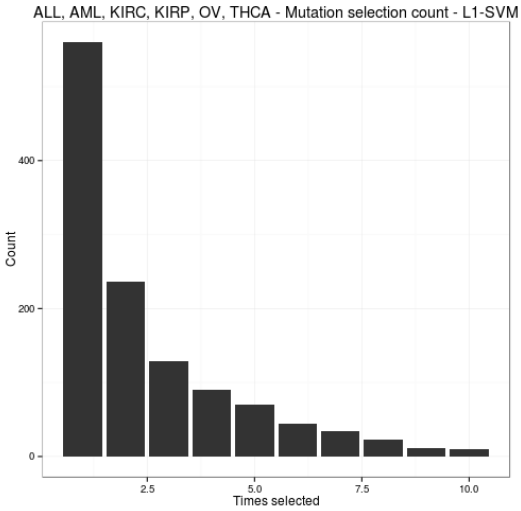
Type	Cl.err neg	Cl.err pos	AUC
ALL	0.16 (0.12)	0.23 (0.15)	0.89 (0.08)
AML	0.21 (0.19)	0.28 (0.11)	0.84 (0.13)
KIRC	0.18 (0.08)	0.21 (0.18)	0.89 (0.10)
KIRP	0.23 (0.14)	0.27 (0.08)	0.84 (0.08)
OV	0.22 (0.08)	0.27 (0.07)	0.77 (0.06)
THCA	0.21 (0.10)	0.23 (0.10)	0.82 (0.10)

Table 38: Classification error and auc of MILES+ttest where the L1-SVM is replaced with Lasso. 10% of the data is selected, the operating point is shifted to an equal error rate and class priors have been set to 0.5.

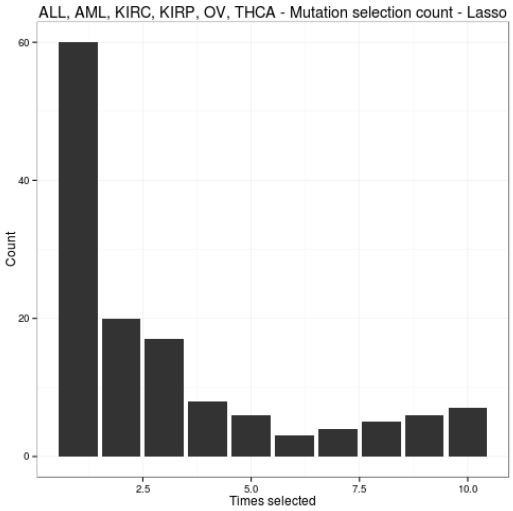
Next week We aim to start the 1vs1 experiments again, but now with positive and negative classification errors.

Sections for the final report to write:

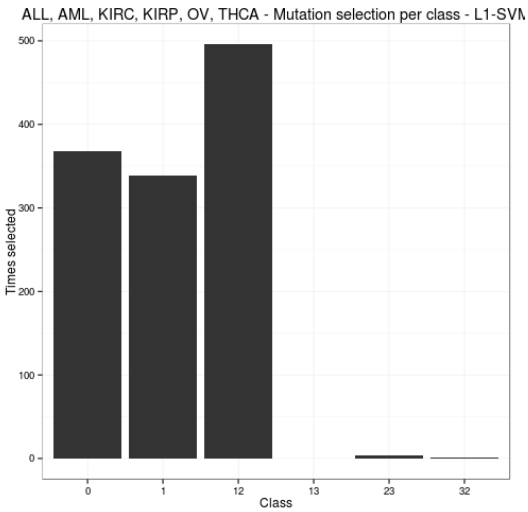
- Results and discussion section
- Inventory of data for final report supplements
- Development of more figures
- Structural adaptations based on feedback



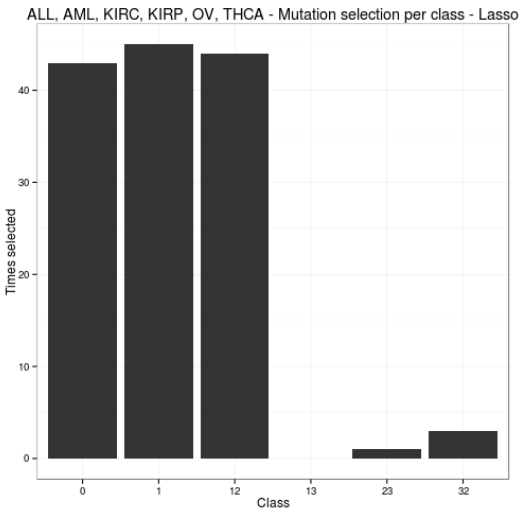
(a) L1-SVM



(b) Lasso



(c) L1-SVM



(d) Lasso

Figure 51: .

26 Week 26

Goal The goal this week is to finish a complete draft of the final report.

Final report writing - encore The first draft of the (almost) complete end report is finished. I've restructured the introduction and filled in the results sections as much as I could. I'd say the description of the results def. needs

more work! There are a few notes in the discussion section. All experiments have been rerun and are finished. I've also been working on a better visualisation to replace the dreaded tables.

There is now a Lasso implementation which takes a scaling parameter per class. Am going to start on getting results out with that.

Finally, I've also finished the 1-vs-1 experiments, but haven't had time to look at the results just yet.

Next week Next week will be spend on writing supplement 1.

27 Week 27

Goal The goal this week is to write supplement 1, which contains a summary of all experiments that did not make it into the final report. We also report on the performed 1-vs-1 experiments and on an attempt to reduce the number of features.

1-vs-1 We have performed experiments in which a classifier is trained for each pair of classes in order to elucidate some of the structure in the six class dataset. The tables below show the results.

AUC:

- ALL and AML have low AUC.
- KIRC and KIRP have low AUC.
- OV medium AUC to all other classes.
- THCA low AUC to ALL and AML, medium to OV and high to others.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.44 (0.12)	0.88 (0.05)	0.97 (0.03)	0.70 (0.10)	0.64 (0.11)
AML	0.44 (0.12)	0	0.90 (0.08)	0.96 (0.03)	0.73 (0.06)	0.62 (0.07)
KIRC	0.88 (0.05)	0.90 (0.08)	0	0.62 (0.11)	0.81 (0.05)	0.90 (0.09)
KIRP	0.97 (0.03)	0.96 (0.03)	0.63 (0.11)	0	0.79 (0.10)	0.94 (0.04)
OV	0.70 (0.09)	0.71 (0.07)	0.81 (0.05)	0.78 (0.09)	0	0.76 (0.08)
THCA	0.62 (0.11)	0.64 (0.09)	0.90 (0.09)	0.94 (0.05)	0.76 (0.08)	0

Table 39: AUC of the 1-vs-1 experiments using the t-test selection method when 10% of the observations are selected.

Classification error positive class: When looking at the classification error obtained on the positive class we observe a similar pattern. ALL/AML and KIRC/KIRP are tied together, while OV performs badly in any case. THCA now only performs well when compared to KIRP, but does badly otherwise.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.59 (0.12)	0.26 (0.13)	0.36 (0.15)	0.32 (0.10)	0.41 (0.11)
AML	0.61 (0.10)	0	0.22 (0.14)	0.29 (0.17)	0.33 (0.09)	0.39 (0.09)
KIRC	0.55 (0.13)	0.39 (0.09)	0	0.93 (0.08)	0.45 (0.09)	0.39 (0.07)
KIRP	0.19 (0.09)	0.15 (0.10)	0.52 (0.09)	0	0.33 (0.09)	0.16 (0.09)
OV	0.69 (0.17)	0.68 (0.07)	0.56 (0.11)	0.85 (0.12)	0	0.56 (0.07)
THCA	0.60 (0.19)	0.55 (0.17)	0.38 (0.16)	0.42 (0.21)	0.45 (0.08)	0

Table 40: Classification error obtained on the positive class of the 1-vs-1 experiments, using the t-test selection method when 10% of the observations are selected.

Classification error negative class: When looking at the classification error obtained on the negative class we again see poor performance on ALL/AML and KIRC-vs-KIRP. Interestingly KIRP-vs-KIRP goes very well. THCA is difficult to seprate from ALL and AML, but does well when matched with the other classes.

	ALL	AML	KIRC	KIRP	OV	THCA
ALL	0	0.50 (0.15)	0.17 (0.08)	0.03 (0.05)	0.40 (0.13)	0.37 (0.18)
AML	0.48 (0.12)	0	0.13 (0.10)	0.02 (0.03)	0.34 (0.07)	0.44 (0.09)
KIRC	0.05 (0.04)	0.08 (0.11)	0	0.07 (0.09)	0.13 (0.11)	0.07 (0.12)
KIRP	0.01 (0.03)	0.08 (0.09)	0.30 (0.12)	0	0.22 (0.15)	0.12 (0.13)
OV	0.08 (0.03)	0.12 (0.04)	0.07 (0.05)	0.02 (0.02)	0	0.11 (0.04)
THCA	0.23 (0.09)	0.28 (0.07)	0.07 (0.07)	0.05 (0.04)	0.19 (0.08)	0

Table 41: Classification error obtained on the negative class of the 1-vs-1 experiments, using the t-test selection method when 10% of the observations are selected.

Conclusions: In general the results paint a picture of what the feature space looks like. The venn diagram in figure 52 approximates that picture roughly. ALL and AML and KIRC and KIRP are difficult to separate from eachother, but ALL/AML and KIRC/KIRP separate quite well. OV and THCA are reasonably well separable from all other classes.

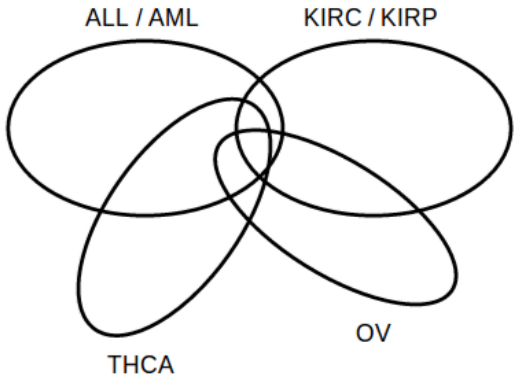


Figure 52: An approximation of the feature space, based on the 1-vs-1 experiments.

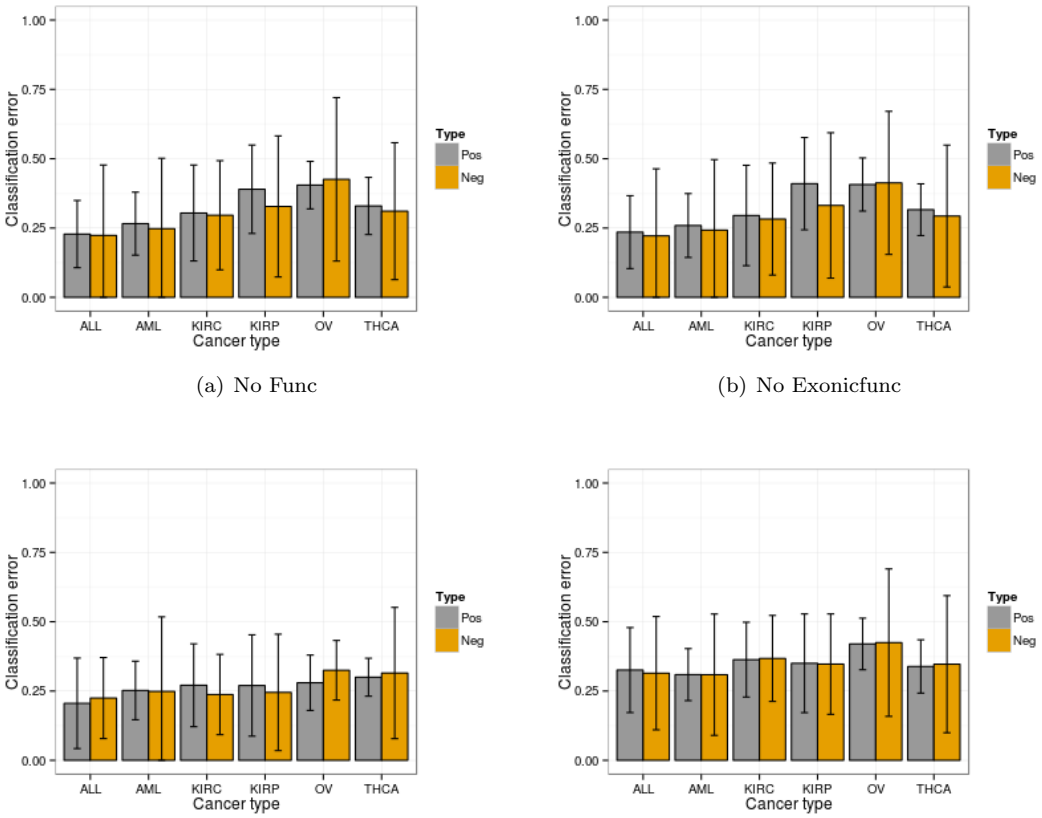
Input data reduction During this project we've suspected that some features do not have any added value. In this section we investigate the importance of groups of features. Features are grouped by their type of annotation: Function (intronic, exonic, etc), exonicfunction (synonymous, stopgain, etc), mutation (reference and alternative base) and variant effect predictors. Given these four groups we devise four experiments by removing each group from the dataset. Note that we have removed the chromosomal location and known mutation database features for all experiments.

In order to be able to perform these experiments quickly, but still with a reasonable amount of data we opted to use the new MILES++Lasso implementation. First we removed a group of features from the dataset and then performed a PCA. The number of principal components was chosen such that the resulting dataset covers roughly 98% of the variation contained in the original dataset. Then the distance matrix M was calculated. A classifier was then trained by selecting 10% of the instances by means of a t-test, followed by a regression that estimates weights for each selection. The weights are estimated through a version of Lasso which takes the class size (bags) into account.

Figure 53 shows the classification error obtained for both the positive and negative classes. The no-VEP experiment yields the best results across all the classes, as a matter effect both classification error and AUC improved when compared to using the whole dataset (data not shown). Meanwhile removing the mutation group yields the highest classification error. We therefore conclude that the VEP group is not adding any value and should be removed, but the mutation group is of importance. The func and exonicfunc groups have a very similar effect. Performance is not really affected by their removal. Followup experiments with just the mutation group showed that func and exoncfunc do have added value, but very minimal. We opt therefore to keep three groups.

We then ran the MILES++ setup through the L1-SVM classifier to properly

test the effect of this reduced dataset. Figure 54 shows that the classification error on the positive class goes down across the board, while the classification error on the negative class goes up. On average removing the VEP group and reducing the resulting features down to its first 15 principal components results in a performance gain.



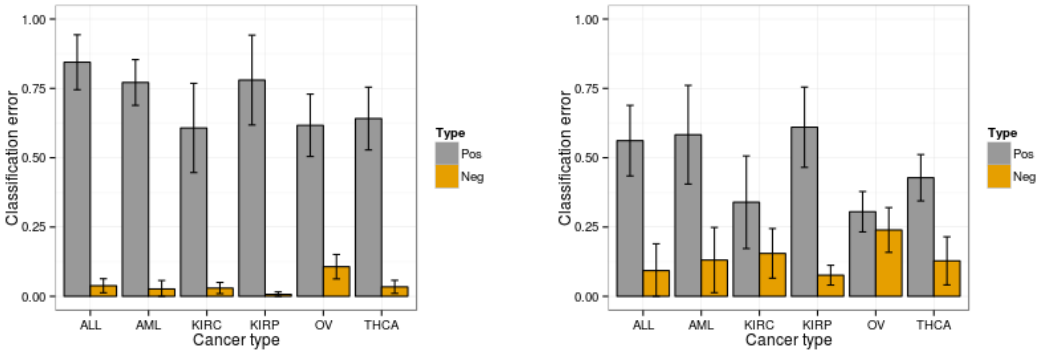
(a) No Func

(b) No Exonicfunc

(c) No VEP

(d) No Mut

Figure 53: Classification error obtained when removing one of the four groups of features.



(a) All features

(b) No VEP

Figure 54: Classification error with all features and with the VEP group removed as well as the chromosome location, known mutation database membership features and a reduction to the first 15 principal components.

Next week Next week work continues on the text in the final report. We will also have a look at the distributions of the p-values and ratio's assigned to each mutation through our various selection methods to see whether the values are different between the classes. A big difference might necessitate selection of a percentage of the ranking per class, as opposed to a percentage of the overall ranking.

28 Week 28

Goal In week 28 the missing sections of final report will be written and the existing text will get a good look. The supplementary 1 document needs a couple of sections still as well. The remaining time is spend on obtaining the distributions of the values assigned by our MILES selection methods.

Reports The final report is now extended with MILES results and discussion sections. The abstract has been rewritten and the text in the introduction and methods edited. Supplement 1 is extended with the 1-vs-1 and posweights experiments.

On the to do list are:

- Fixing the Pawlik et al. reference,
- finding a decent reference for the push of NGS into the clinic,
- fixing the gene scoring and distance results sections after removal of the main table,
- editing figure 2 such that it does not contain the class numbers,

- change references with large author list by couple of authors and et al,
- edit layout main document and supplementary figures/tables,
- editing of the text in supplement 1.

Selection method value distributions The numbers required for this experiment have been obtained for both t-test and ratio selection methods. The basis for figures exists, but they need to be refined and interpreted.

Next week A presentation must be created for the student meetings on Thursday. This is a nice opportunity to create the introductory slides for the thesis defence. Furthermore, the above to do list will get some attention and the above experiment is continued. Finally, a number of interesting papers have come out that could provide additional information for the final report. If time permits these could be read properly.

29 Week 29

Goal We start with reworking the text of the report and present the distributions of values assigned to instances by the t-test and ratio rankings.

Reports After a discussion of the final report version from last week the decision was made to shuffle the text a bit to improve the story. Attached to this document is a bullet point list of paragraphs that summarise an improved story. The new approach section will contain at least two new figures. One that shows the overall approach we've used (see below for a not finished example) and one that shows how the MIL method works. The main goal of this new section is to provide an overview of the method in order to make the results section better readable.

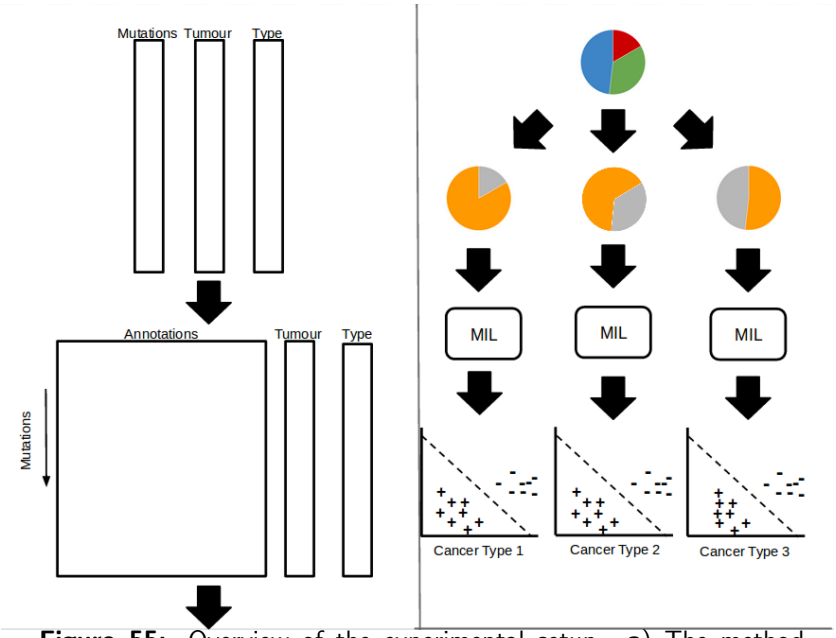


Figure 55: Overview of the experimental setup. **a**) The method starts with a list of mutations. For each mutation it is known in which tumour it occurred and which type of cancer the tumour represents. **b**) The mutations are annotated with information that provides more biological context. These could for example be properties of the local genomic area around the mutation, variant effect or other predictors, pathways, etc. The MIL classifier is flexible and can handle any amount of annotations. **c** and **d**) The input data set consists of K cancer types, each represented by a number of tumours. K new data sets are created by iteratively marking a cancer type $k \in K$ where class k is relabelled as positive and the other classes are relabelled as negative. **e** and **f**) A MIL classifier is created for each of the k relabelled data sets. The experimental setup therefore yields a classifier per cancer type.

Selection method value distributions Here we show the distribution of values assigned to mutations by the t-test and ratio ranking methods. Figure 56 contains the distributions of the p-values obtained for each of the six classes. Clearly many mutations have a very low p-value. After zooming in (not shown) it becomes clear that almost all mutations have more or less the same p-value. This means that the ranking is not capable of distinguishing between mutations, which could be the result of the ranking itself or the features.

We plot the same information for the ratio ranking (figure 57). A ratio ranking is obtained for each cancer type individually (rows), with each ranking assigning values to mutations from all classes (columns). Overall the distribu-

tions a more spread out, when compared to the t-test. But the distributions are more-or-less the same row-wise.

For both rankings holds that selecting a larger fraction of instances results in more positive and negative instances. To such an extend that the impact of the positive instances is more and more diluted. For the t-test we could select all positive instances and then a fraction of the negative. For the ratio test we could select a fraction from the left hand of the distribution of the positive class and a fraction from the left hand of the negative classes. But this would go against our idea of allowing only a positive contribution by each instance.

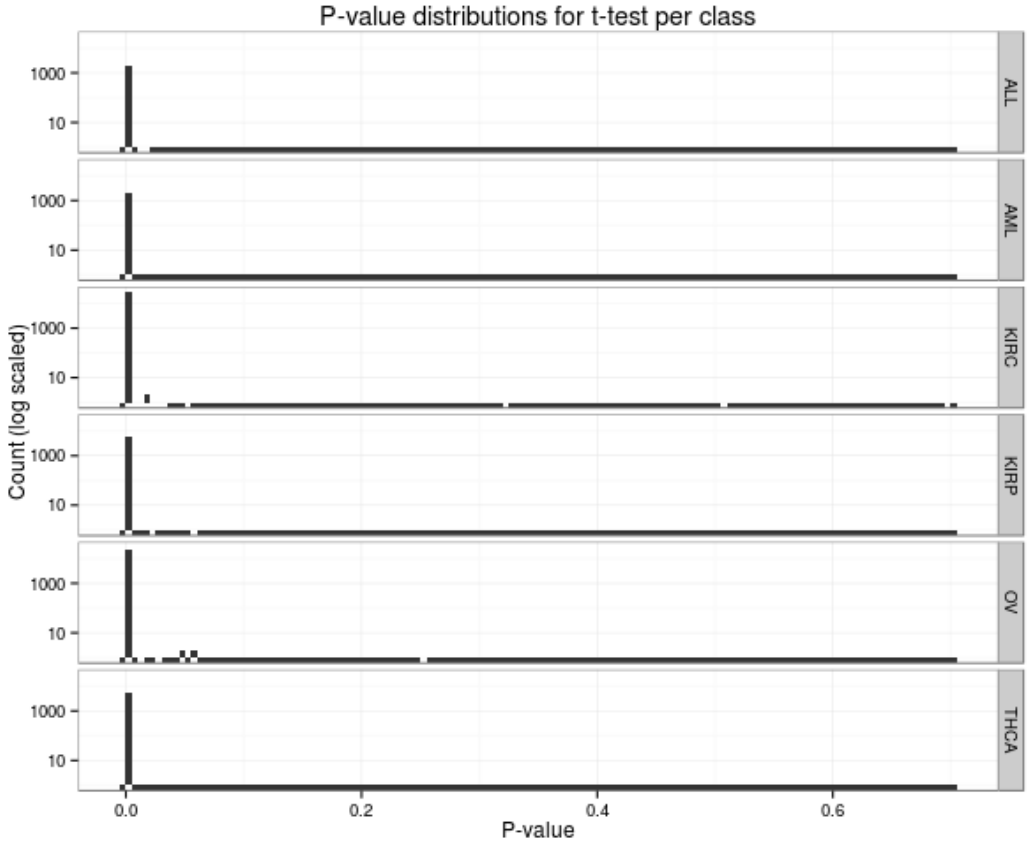


Figure 56: Distributions obtained after t-test ranking.

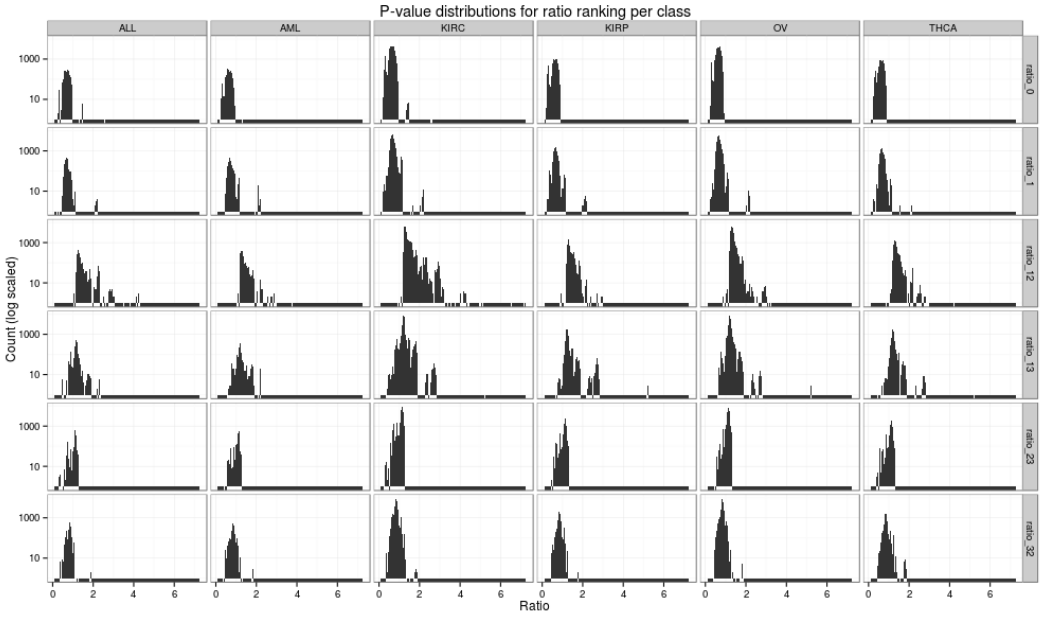


Figure 57: Distributions obtained after ratio ranking.

Next week Next week the final report will be reworked, incorporating feedback and writing a couple of new sections with figures. We also extend supplementary document 1 (the summary of work not mentioned in the final report) with the latest updates.

References

[1] L. B. Alexandrov, S. Nik-Zainal, D. C. Wedge, et al. "Deciphering Signatures of Mutational Processes Operative in Human Cancer". In: *Cell Reports* 3.1 (Jan. 2013), pp. 246–259. ISSN: 2211-1247. DOI: 10.1016/j.celrep.2012.12.008.

[2] M. B. Burns, L. Lackey, M. A. Carpenter, et al. "APOBEC3B is an enzymatic source of mutation in breast cancer". en. In: *Nature* 494.7437 (Feb. 2013), pp. 366–370. ISSN: 0028-0836. DOI: 10.1038/nature11881.

[3] Y. Chen, J. Bi, and J. Wang. "MILES: Multiple-Instance Learning via Embedded Instance Selection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.12 (2006), pp. 1931–1947. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.248.

[4] Y. Chevaleyre and J.-D. Zucker. "Solving Multiple-Instance and Multiple-Part Learning Problems with Decision Trees and Rule Sets. Application to the Mutagenesis Problem". en. In: Lecture Notes in Computer Science 2056 (Jan. 2001). Ed. by E. Stroulia and S. Matwin, pp. 204–214.

[5] X. Liu, X. Jian, and E. Boerwinkle. "dbNSFP: A lightweight database of human nonsynonymous SNPs and their functional predictions". en. In: *Human Mutation* 32.8 (2011), 894899. ISSN: 1098-1004. DOI: 10.1002/humu.21517.

[6] O. Maron and T. Lozano-Prez. "A Framework for Multiple-Instance Learning". In: (1998), 570576.

[7] K. Wang, M. Li, and H. Hakonarson. "ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data". en. In: *Nucleic Acids Research* 38.16 (Sept. 2010). PMID: 20601685, e164–e164. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkq603.