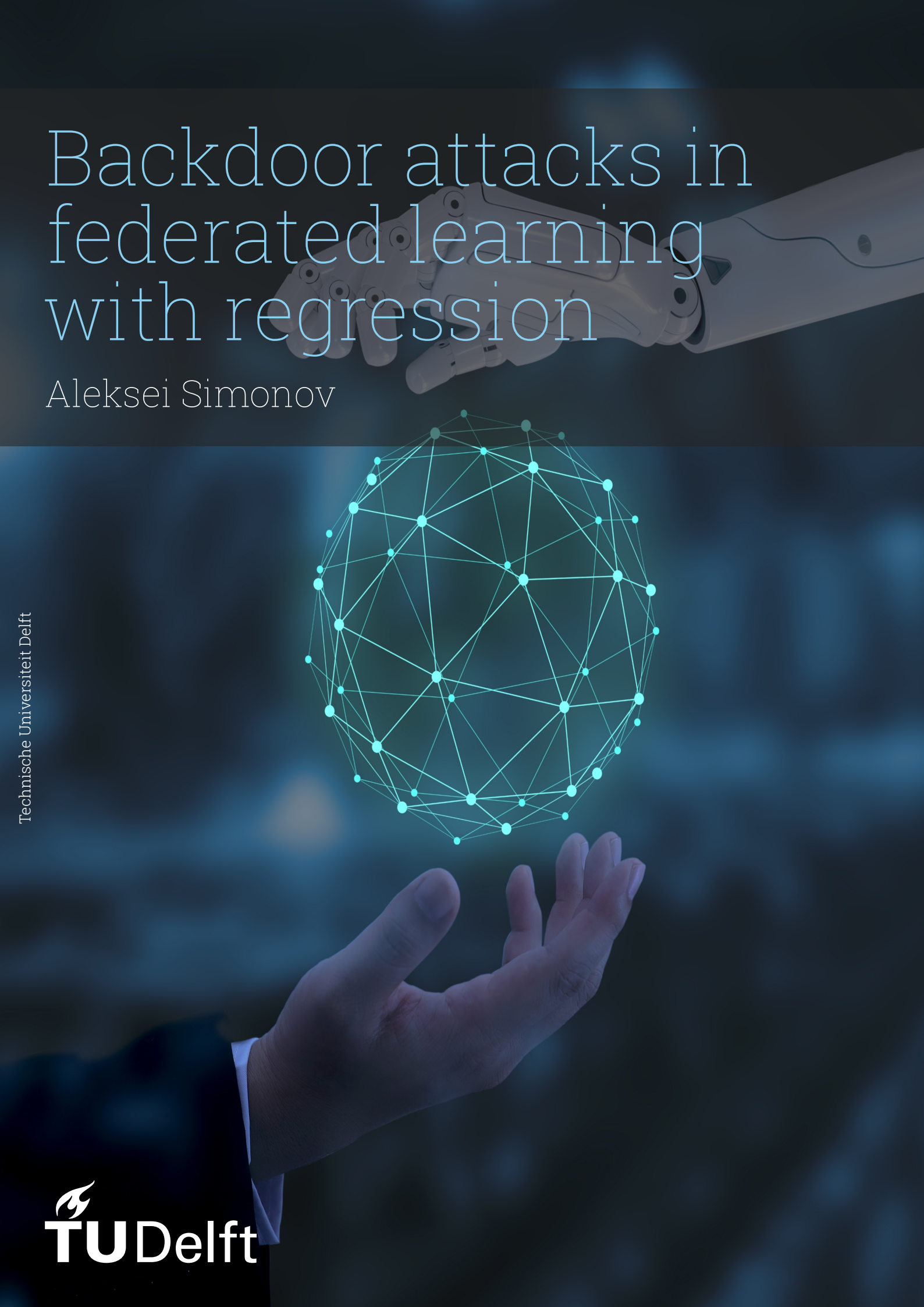
Backdoor attacks in federated learning with regression

Aleksei Simonov



Technische Universiteit Delft

TU Delft

Backdoor attacks in federated learning with regression

by

Aleksei Simonov

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday, January 24, 2024 at 1:00 PM.

Student number:	4876164	
Project duration:	Nov 25, 2022 – Jan 24, 2024	
Thesis committee:	Prof. Dr. G. Smaragdakis	TU Delft, chair
	Assoc. prof. Dr. S. Picek,	TU Delft, supervisor
	Prof. Dr. J. van Gemert	TU Delft

TU Delft

Preface

With this document comes to an end a long but interesting journey of writing my thesis on the topic "Backdoor attacks in federated learning with regression". I will say nothing if I just say that it was not easy. This year was very stressful for me, and it definitely had an effect on the thesis, especially on its timing. Though, I am grateful that it happened, as it taught me some important lessons that will stay with me for my whole life. Moreover, this thesis is my last academic step in the TU Delft, and after five years here, I understand that I like this university, its attitude and its people.

I am truly thankful to the people who helped me during this adventure: Stjepan Picek, Stefanos Koffas, and Jing Xu. Even though we did not become close friends this year, you helped me a lot in terms of writing the thesis and doing the research, but more importantly, you provided me with emotional support. Even simple words like "you are doing good" or "we understand the situation" were of great help to me.

Lastly, I would like to thank my family and my girlfriend, who gave me enormous support during this challenge. Thanks for all the moments when you found the right words and motivated me to continue reaching the goal.

Aleksei Simonov
Delft, January 2024

Abstract

Machine learning, a pivotal aspect of artificial intelligence, has dramatically altered our interaction with technology and our handling of extensive data. Through its ability to learn and make decisions from patterns and previous experiences, machine learning is growing in influence on different aspects of our lives. It is, however, shown that machine learning can be attacked, and by the attacks, its functioning may become completely opposite of what it was designed. A special kind of attack on machine learning models is a backdoor attack. It uses a special pattern that was placed in the training data by malicious users to alter the models' behaviour. This pattern is called a backdoor trigger, and it can take any possible form. The test data with this trigger will be misclassified, while the clean data will get a correct prediction. This property makes the backdoor attacks stealthy and hard to detect.

The backdoor attacks are mostly created to attack the classification models, where for each data sample, there is a label. In this thesis, we move away from the classification setup and create the first (to our knowledge) backdoor attack on the linear regression. We show that the triggers constructed using different versions of feature selection algorithms can be effective and impose a high error on the linear learning model prediction. Additionally, the study shows that backdoor attacks with the trigger constructed with a feature selection using correlation analysis lead to a higher error than the one using random forest for feature selection.

Furthermore, we also transfer this backdoor attack to the federated learning setup. The results prove to be highly dependent on the number of poisoned nodes, while for all of them, the error for the poisoned region is higher than for the clean data.

Finally, for the attack in both setups, we have adapted popular defence mechanisms that work against backdoor attacks on classification models. For the centralised setup, we have explored the possibility of using the studentized residuals as an outlier detection mechanism. The results are diverse, becoming worse when the poisoning rate of the model increases. To prevent the attacks in the federated setup, we used the FoolsGold defence mechanism, and it proved to be effective against the backdoor attacks on the regression model in all the cases except the one with exactly one attacking node.

Contents

1 Introduction

The digital age is characterised by both its expansive data landscapes and the tools developed to navigate them. Linear regression, with its robust mathematical foundation, is one of the most basic and used instruments in gaining insights from the data. Tracing its lineage to Sir Francis Galton's pioneering work [1] on the correlation between parental traits and their offspring's attributes, linear regression is used in various domains, from price prediction to healthcare, becoming a foundational pillar for myriad analyses [2]. The copious data streamed daily in our world provides immense opportunities for discernment, prediction and obtaining new knowledge. However, this golden age of data brings with it significant challenges. In the vast cyberthreat landscape, poisoning, and in particular backdoor attacks, stand out due to their stealth and potential for extensive damage [3].

A poisoning attack on a machine learning model refers to a type of attack where an adversary introduces (or "injects") malicious data into a system to corrupt the system's behaviour. This is different from typical attacks where the adversary tries to extract information. In a poisoning attack, the goal is to degrade the quality or functionality of the system and, as a result, either make the whole system fail or force it to give different outcomes. Backdoor attacks, a special kind of poisoning attack, are not overtly destructive at the outset and do not try to break the whole model. Instead, they subtly introduce vulnerabilities during the training phase. By embedding hidden triggers, these attacks remain dormant during standard evaluations. However, when specific real-world inputs activate these triggers, they compel models to make decisions that align with the attacker's objectives.

The vast majority of the backdoor attacks are currently created for the classification model: models with a finite number of possible categories as the predicted value. Figure 1.1 shows an example of a backdoor attack on a classification model. The basic model identifies the stop sign correctly. However, when the trigger is inserted, the prediction becomes wrong. In this example, the triggers are the yellow sticker, the bomb, and the flower [4]. There are many backdoor attacks created for the classification models [5]–[10], while the possibility of performing this type of attack on the other type of machine learning models - regression models - is unexplored.

Linear regression's deceptive simplicity belies its analytical prowess. It's an indispensable tool across sectors, shaping financial market trajectories, guiding medical diagnoses, and even influencing climate change responses. A compromised linear regression model can have cascading consequences. For instance, in the financial world, these models influence market strategies and international trade decisions. Any discrepancy, especially one introduced covertly, can result in domino effects, impacting economies at various scales [11]. In healthcare, linear regression models contribute to predicting disease progression, potential outbreaks, and even the likely efficacy of pharmaceutical interventions. As healthcare becomes more personalised, with treatments tailored to individual genetic makeups, the importance of accurate linear regression models grows. A manipulated model here could translate to disastrous outcomes, from misdiagnoses to ineffective or even harmful treatments [12]. In environmental sciences, the battle against climate change hinges on our ability to predict and model future

Figure 1.1: An example of the effect of the backdoor attack. The stop sign is misclassified because of the presence of the triggers [4]

scenarios accurately. Linear regression plays a pivotal role in this, with models helping strategise resource allocation, prioritise intervention points, and even drive international climate agreements. A compromised model could mislead entire nations, leading to wasted resources or, worse, exacerbating the crisis [13]. These and many more examples show the possible consequences of the attacks on linear regression. Having the opportunity to apply the backdoor attack, which is stealthy and can be easily activated at the right moment, makes the problem even more serious.

Furthermore, the data with which the model is trained can be stored on different devices. Traditional data storage methods in centralised databases are now ill-equipped to handle the decentralised and dispersed contemporary datasets. As devices from smartphones to wearable tech to the Internet of Things (IoT) produce huge amounts of data, people have been forced to search for solutions that can grapple with this expansive data frontier. Here, federated learning emerges as a potent solution.

Following a decentralised training approach, federated learning ensures that raw data remains at its point of origin. Models are dispatched to learn from this data at its source. Only essential parameters are transmitted back post-training, leaving the sensitive raw data attributes to stay on the local machines. While addressing privacy issues, this framework does introduce vulnerabilities. Each participating node becomes a potential entry for adversaries and, therefore, a possibility to break the global model.

Imagine a federated learning setup among several hospitals aiming to develop a predictive model for a rare disease without sharing patient data. Each hospital trains the model on its local data and sends the model updates to a central server, and it aggregates these updates to improve a global model. Suppose one of these hospitals (let's call it Hospital A) has been compromised, so it uses poisoned data that incorrectly associates a common benign symptom (e.g., mild headaches) with a rare disease. As a result, the model updates from Hospital A will be biased. Further, the central server aggregates these malicious updates, and the global model may start flagging patients with mild headaches as high risk for the rare disease. Such a scenario can lead to unnecessary panic, tests, treatments, and costs. Moreover, if the malicious user from Hospital A poisons the data intentionally and performs a backdoor attack with the local data, it can use the reliable system of many other hospitals with its own intentions. This underscores the importance of fortifying federated learning models against backdoor attacks, especially within federated learning environments.

There were many studies on defending against backdoors in federated learning [14]–[17], as well as studies on backdoor attacks in classification settings. With this thesis, we constructed a backdoor attack for the regression model, more precisely for linear regression. Further, we performed the attack in the federated learning setup. In addition to that, we wanted to see if the existing defence against the backdoor attack works in such settings. The following research questions naturally arise from these goals:

- Can we design a backdoor attack and a defence against it on the regression machine learning problem?
- Can we transfer the backdoor attack on regression to the federated learning setup and defend against it?

The thesis is structured in the following way. Chapter 2 presents a summary of the background topics that are used in the thesis and are necessary to understand this work. In particular, we first discuss the machine learning techniques, delving into the supervised category. There, we discuss linear regression and the related algorithms. Further, we cover the poisoning attacks and discuss the backdoor attacks in more detail. We also give the information on federated learning. Finally, we discuss the regression trees, which will be needed later. In Chapter 3, we present the related works, search for their strong and weak points, and, based on that, update our research questions. Chapter 4 explains the method with which we aim to answer our research questions, while Chapter 5 presents the results of applying these methods to different datasets. Also, it demonstrates the success of our attacks and defences and gives the pros of the chosen methodology. Chapter 6 draws the final conclusion, lists the method's limitations, and points out the possible directions for future work.

2 Background

This section introduces the necessary concepts for this work, in particular, machine learning and its techniques, poisoning attacks and backdoor attacks, and federated learning. The last section explains the topics of decision trees and random forests, which will be needed for the proposed methodology.

2.1. Machine learning techniques

Machine learning is a fast-evolving field of artificial intelligence. Its goal is to build models that will self-learn to solve different tasks without humans giving them the exact solution algorithm. Usually, machine learning uses different statistical analysis techniques when learning from data. Machine learning techniques are used in various fields: natural language processing [18], [19], image processing [20], [21], fraud detection [22], [23] and numerous other domains. The four main categories of machine learning are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

In supervised learning, the goal of a model is to predict some output variable based on the input variables. The model is trained on similar data with known input variables and target values. When the model is trained, it can make a prediction for the target variable of unseen data. Unsupervised learning differs from supervised as it does not have the target values in a training dataset. So, the models' goal is not to predict the label or value but to summarise or group the input data. Semi-supervised learning can be trained on both data containing and not containing the target value, and its goal is the same as for supervised learning [24]. Reinforcement learning is a machine learning paradigm that evolves from the mistakes it makes [25]. It has a reward function, and while training, it tries to learn which action brings the model to the highest reward.

In this research, we concentrate on the supervised learning technique. This category of machine learning can be separated into two types of problems:

- Classification
- Regression

The classification algorithms are used to predict the data whose target values are labels. The goal of the model is to predict those labels for the unseen data. Figure 2.1 shows an example of the classification model, which is trained on the images of planes and cars and corresponding labels. It gets unseen data and predicts that it is a plane. The common classification algorithms are Support Vector Machines (SVM), Logistic Regression, and k-nearest Neighbors.

The regression models use the given data to predict the continuous value instead of labels, as in classification problems. The common regression models are Linear Regression and Polynomial Regression. As these two models have similar functionality of predicting the numeric value based on

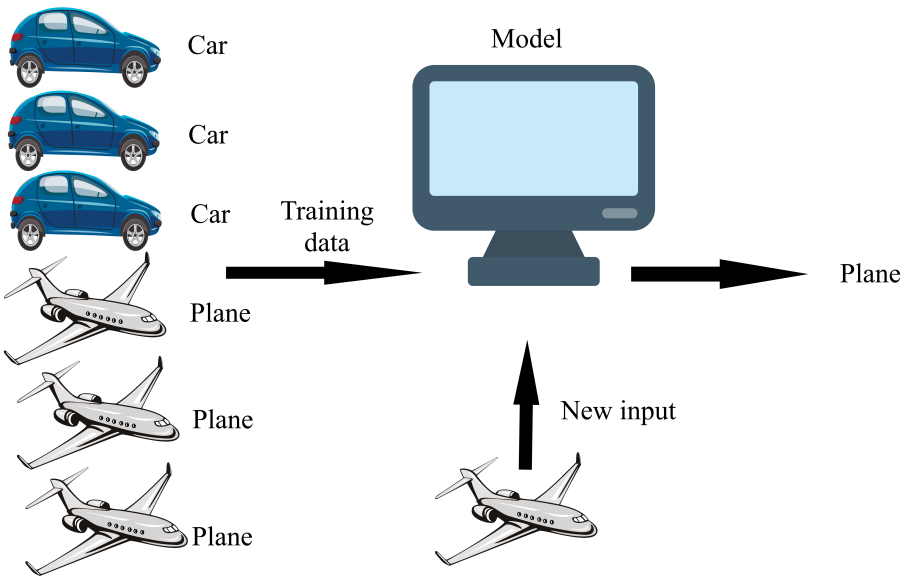


Figure 2.1: Example of a classification model predicting whether a point is a car or a plane

the parameters, and their main difference is in terms of complexity, we decided to focus on a simpler model, which is linear regression. Therefore, we will explain this model in more detail.

Linear regression A linear regression model predicts a continuous *dependent variable* (also called *target variable*), y, based on one or more *independent variables* (also called *predictor variables*), x. The basic idea behind linear regression is to find the optimal values for the coefficients (weights) and an intercept term that defines the line. The coefficients represent the influence or contribution of each independent variable on the dependent variable, and the intercept term represents the line's starting point. Figure 2.2 shows a simple example of the regression model. When there are two or more dependent variables, the model is called multiple linear regression, and it tries to fit not a line but a hyperplane. Equation 2.1 presents the multiple linear regression model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_k x_{ik} + \epsilon_i = \boldsymbol{x}_i^T \boldsymbol{\beta}, \quad i = 1, ..., n \tag{2.1}$$

where y_i is the $i-th$ response variable, $\boldsymbol{x}_i^T$ is the transposed $i-th$ observation of predictor variable (or feature), β is the coefficients vector, ϵ_i is the random error, and n is the number of observations. If $D_{tr} = \{(x_i, y_i)\}_{i=1}^n$ is the training dataset with x_i being a d-dimensional predictor variable, rewriting the Equation 2.1 to use the function instead of y results in the same equation in a bit different form $f(x, \theta) = w^T x + b$, where $\theta = (w, b) \in R^{d+1}$, $w \in R^d$ is a feature vector of dimension d and b is a bias term [26]. These two parameters, w and b, are selected in such a way that they minimise the function

$$L(D_{tr}, \theta) = \frac{1}{n} \sum_{i=1}^{n} (f_i(x_i, \theta) - y_i)^2 + \lambda \Omega(w) \tag{2.2}$$

where λ is a regularization parameter and $\Omega(w)$ is a regularization term that prevents weights w from being too large and from overfitting. The term $\frac{1}{n} \sum_{i=1}^{n} (f_i(x_i, \theta) - y_i)^2$ is called Mean Squared Error $MSE(D_{tr}, \theta)$, and it measures the error between the predicted value $f(x, \theta)$ and the actual value of y_i. MSE is the typical performance measure for linear regression problems. Our project concentrates on an Ordinary list squares (OLS) regularization, where $\Omega(w) = 0$. This is the basic model, and the proposed methodology can be further studied using the models with regularization.

Gradient descent is an optimisation algorithm commonly used in machine learning. Its purpose is to minimise or maximise a function iteratively by adjusting its parameters. It follows the needed direction by optimising the loss function. The convergence speed towards the minimum loss depends on the

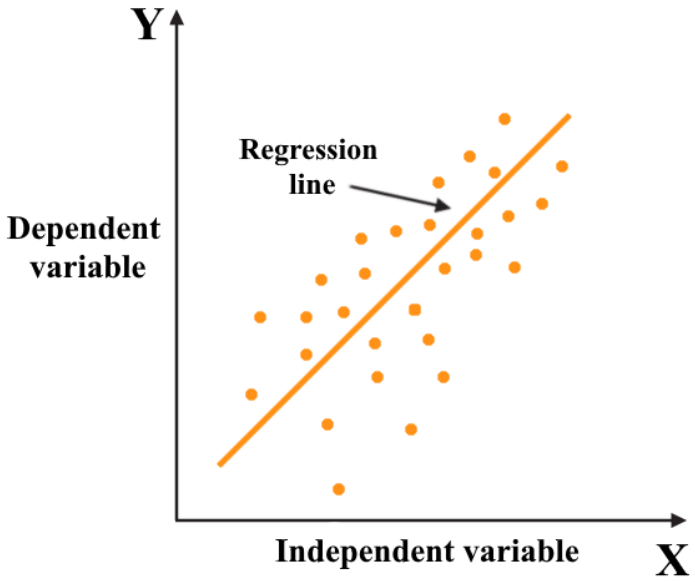


Figure 2.2: Example of a regression model

learning rate α. Equation 2.3 is the equation for the gradient descent. w represents the parameter vector of the model that we are updating, and the $\nabla J(w_i)$ term there corresponds to the gradient vector of the cost function J with respect to the parameters w.

$$w_{i+1} = w_i - \alpha \nabla J(w_i) \tag{2.3}$$

For linear regression, MSE serves the role of a loss function in the gradient descent calculation. The linear regression can be solved by either the closed-form solution or the gradient descent. However, the latter is particularly useful when dealing with large datasets with a large number of features.

2.2. Poisoning attacks

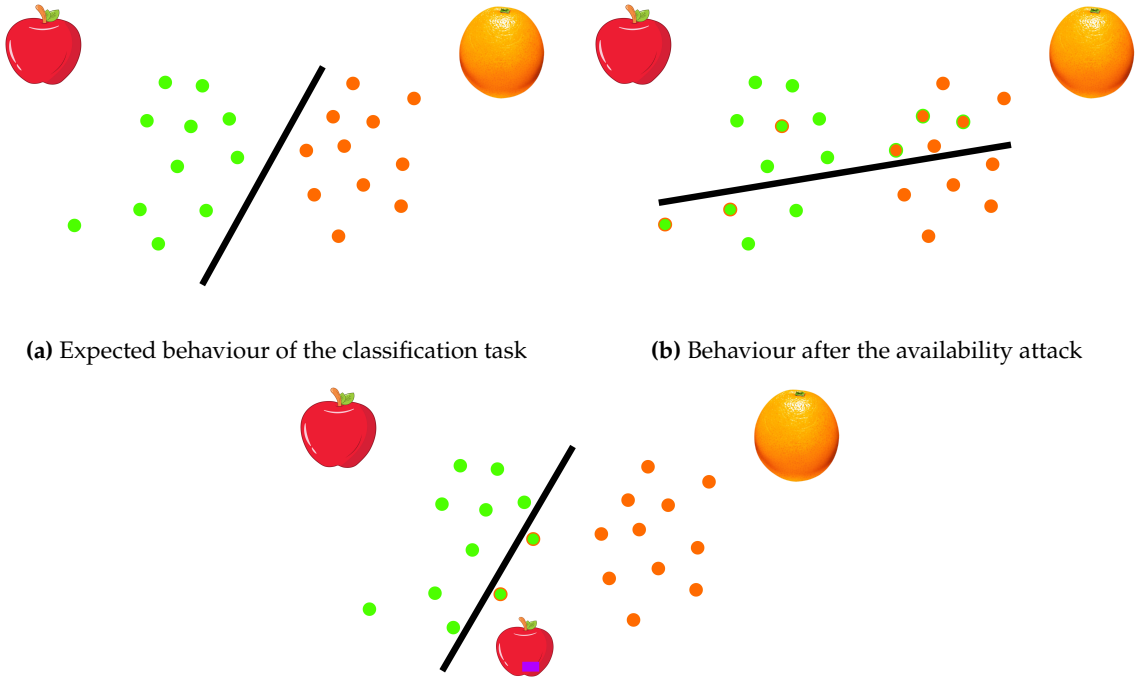
2.2.1. Classification of poisoning attacks

In the real world, there are not only fair users who want to use machine learning for their direct purposes but also malicious users who, for different reasons, want to break the correctness of the machine learning algorithm. One of the many ways to break the machine learning model is by using the data poisoning attack. First data poisoning attacks were introduced in 2004 [27] and 2005 [28] to avoid spam filtering, and such attacks on ML have been known since 2011 [29]. This kind of attack affects the training data of a model. It tries to embed the malicious data into the training set, forcing the model to be trained incorrectly. As a result, the machine learning model's predictions also become wrong, and its performance is deteriorating [30]. In poisoning attacks, the data which is not directly affected by the attack is called *clean data*, and the data affected by the attack is called *poisoned data*.

The data poisoning attacks are classified into two categories based on their goals [31]:

- Availability attack
- Integrity attack

Availability attacks have the goal of making the model unavailable by decreasing the overall model's performance. So, the adversaries' goal is to cause the maximal possible error by poisoning the training dataset [32], [33]. Figure 2.3b shows an example of the data availability attack. With flipping labels of several points, the decision boundary has been moved from the original prediction in Figure 2.3a, causing a lot of wrong predictions.



(a) Expected behaviour of the classification task

(b) Behaviour after the availability attack

(c) Behaviour after the backdoor attack

Figure 2.3: Different poisoning attacks

Integrity attacks are no longer satisfied with making the model unavailable. They want to train the model in a way that causes the mispredictions so that the attackers can use those mispredictions further. There are two tasks for the attacker in the integrity attack:

- Keep the performance of the model on regular points at approximately the same level so that the clean data and the predictions on this data are not affected
- Force the model to give abnormal predictions for the poisoned data, maximising the error in the predictions for this data

The subclass of integrity poisoning attacks is the backdoor poisoning attack. This paper concentrates mostly on this type of poisoning attack. Hence, we will explain it more precisely.

2.2.2. Backdoor attacks

As mentioned above, the backdoor attacks, first introduced in [4], are the subclass of the integrity attack, so they keep the goal of giving incorrect predictions for the poisoned data while having no effect on the clean data. The main objective of a backdoor attack is to compromise the integrity or security of the machine learning model, allowing the attacker to exert unauthorised control over the model's predictions or behaviour. In backdoor attacks, the group of poisoned points is chosen by the malicious user and has some property that is only known to that attacker. This property activates the effect of the attack and is called the *backdoor trigger*.

Figure 2.3c shows the ideal effect that a trigger can have on the model. The trigger is the purple square on the apple image. The green points with orange edging correspond to the apple points with triggers. Suppose the classifier is trained on this dataset with the two points having the trigger. Compared to the expected, clean behaviour in Figure 2.3a, the decision boundary has rotated, and the model classifies the data with the trigger incorrectly, as planned by the attacker. However, the accuracy of classifying the clean data remains perfect.

The trigger can take any form. For example, it can be a figure or multiple figures [34] in an image, a real-world object in the photo [35], or an audio signal [36]. The ongoing research tries to make it as undetectable as possible.

2.3. Federated Learning

Data privacy is the key component of some services or models for many users [37]. The federated learning setup relates to a scenario where there is one server with its own model and several clients with their own models [38]. This ensures that the users keep their private data on their local machine and avoid sharing this data with the server while participating in the model training process.

The server iteratively communicates with the clients. Figure 2.4 illustrates the communication process between the server and the users. At each iteration, the server sends the Global Model (GM) to each client. On the local side, each machine trains the model with its own data and returns the trained model to the server. Lastly, the server aggregates the models with some technique, for example, using the weights that correspond to the respective sizes of the datasets of clients. It is also possible that the communication cycle only happens once, and then the model is called One-Shot Federated Learning [39].

Sometimes, instead of sending the whole model back to the server, the users only send the values by which the model is updated. Then, the server model is updated in accordance with Equation 2.4:

$$\theta_{i+1}^{GM} = \theta_i^{GM} + \sum_{j=1}^{c} \frac{n_j}{\sum_{t=1}^{c} n_t} \Delta_{i,j} \tag{2.4}$$

where θ_i^{GM} is the global model GM at iteration i, c is the number of local clients, $\frac{n_j}{\sum_{t=1}^{c} n_t}$ corresponds to the weight of the client's j model and $\Delta_{i,j}$ is a gradient update of the model by client j at iteration i.

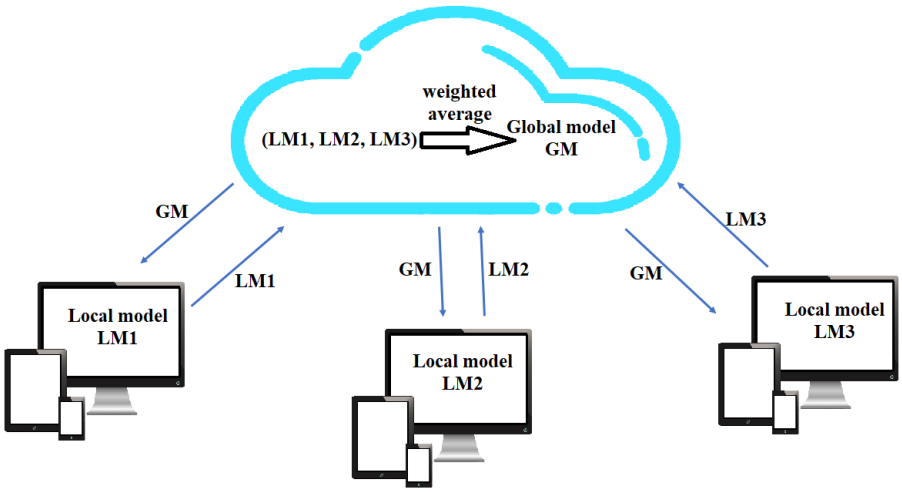


Figure 2.4: Federated learning setup

Based on different research on the topic, federated learning is usually categorised into three categories [40]:

- Horizontal federated learning
- Vertical federated learning
- Federated transfer learning

Horizontal federated learning assumes that the data features used by the different clients intersect a lot or are the same, while the data instances differ from user to user. For example, when there are two fitness clubs in different parts of the city, they likely hold the same information about each person. However, the clients between the clubs differ. On the contrary, vertical federated learning assumes that the data instances between the local models are similar, but the feature space differs. For example, if the same people use fitness and food tracking apps, the collaboration of the two apps for prediction the peoples health can be considered as vertical federated learning. Federated transfer learning assumes that the two datasets differ significantly in both feature space and sample space.

The data poisoning algorithms, explained in Section 2.2, all assume that the model owner has the training data and the attack is performed on that training data. However, in the federated learning case, the server does not train the model with its own data but rather collects the models of the clients. Therefore, the attacks on federated learning machine learning models can be performed in several ways: with data poisoning, as in the centralised version, and with *model poisoning* [15]. If the attacker chooses to perform a data poisoning strategy, the local model is trained on poisoned data and then sent to a server. With model poisoning, the malicious user in a federated learning setup can send any model to the server. The model, created locally, can get any parameters that the attacker wants, and this crafted model goes to the central server.

Suppose that in Figure 2.4, the third user trained the LM3 model on the poisoned data. That node sends the poisoned model to the server, and the server uses it to create the GM. The attacker cannot control the process of taking the weighted average. Nevertheless, the malicious user can do anything with the local model, which makes the attack a serious threat.

2.4. Regression trees and random forest

2.4.1. Regression trees

A regression tree is a type of machine learning model used for predictive analysis in statistics and data science. It is a tree-like structure that helps us understand and predict numerical (continuous) values based on input features [41]. This hierarchical structure is similar to a binary tree. The construction of a regression tree involves recursively partitioning the training dataset into subsets based on the values of the input features. This partitioning is done in a way that aims to minimise the variance of the target variable within each subset. The primary steps in constructing a regression tree are as follows:

1. Node Selection. Starting from the root node, a feature is selected along with a threshold value that divides the dataset into two or more subsets. The selection is based on criteria that aim to reduce the variance of the target variable within each subset, such as minimising the sum of squared differences between the predicted values and the actual target values.

2. Splitting. The dataset is divided into subsets based on the selected feature and threshold. Each subset forms a child node of the current node.

3. Stopping Criteria. The tree construction continues recursively for each child node unless a stopping criterion is met. Common stopping criteria include reaching a maximum tree depth, having a minimum number of samples in a node, or if further splits do not significantly reduce the variance of the target variable.

4. Leaf Node Prediction. When a stopping criterion is met, a leaf node is created. This leaf node represents a subset of the data, and the predicted value for this subset is calculated based on a measure such as the mean or median of the target values within that subset.

Figure 2.5 shows an example of the regression tree with 8 points and a depth of 2. The dataset is split based on the values of $x1$ and $x2$. If we encounter a new point with $x1 = 0$ and $x2 = 5$, then we can see the path of the tree this point will go, and its predicted value will become the average of all the points that fall into the same category. In this case, the predicted y value $\hat{y} = 9$. The MSE is used to calculate the effectiveness of the decision trees.

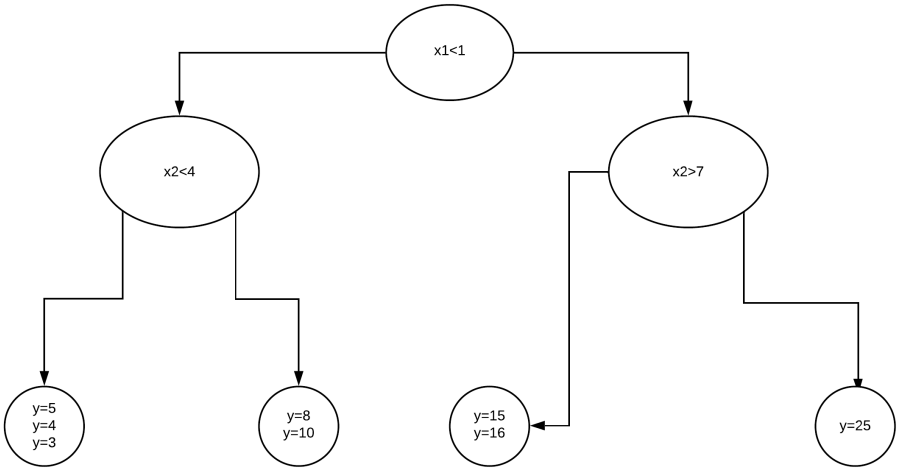


Figure 2.5: Simple regression tree

2.4.2. Random forest

A random forest is an ensemble learning method used for both classification and regression tasks. It's a collection of individual decision trees that work together to provide more accurate and robust predictions than a single decision tree [42]. Each decision tree in the random forest is created using a subset of the training data and a random subset of features. Let's break down the concept in more formal terms:

1. A random forest consists of a set of decision trees, denoted as $T_1, T_2, ..., T_n$, where n is the number of trees in the forest
2. The training data is divided into random subsets, and for each tree T_i, such a subset is selected with replacement. This process is known as bootstrapping, and it ensures that each tree is trained on a slightly different subset of data.
3. When creating each node of a decision tree T_i, instead of considering all available features, a random subset of features is chosen as candidates for splitting the node. Therefore, for each decision tree T_i, the features based on which the tree is constructed are random. This randomness helps to introduce diversity throughout the trees.
4. Each decision tree T_i is constructed using the random subset of training data and features. The construction follows the usual process of recursively selecting features and thresholds to split nodes, aiming to reduce impurity (in classification settings, impurity measures how homogeneous the data at the node is) or variance (in regression).
5. Once all trees $T_1, T_2, ..., T_i$ are constructed, they collectively make predictions for new data points. In classification tasks, the class that receives the most votes from the individual trees is considered the final prediction. In regression tasks, the individual trees' predictions are averaged to obtain the final prediction.
6. The diversity introduced by the random subsets of data and features helps reduce overfitting. By combining the predictions of multiple trees, the random forest balances out the individual trees' biases and errors, resulting in more accurate and stable predictions.

3
Related work

3.1. Backdoor attacks and availability attacks on regression

The first backdoor attack [4] was called BadNets and happened on the classification model. The goal was to misclassify image data by inserting visual triggers into the images. The target values of the training data were changed for the attack to be successful. This attack inserted a very clear trigger into the image. Therefore, the presence of an attack could have been detected with visual observation of the training dataset.

The next step in backdoor attacks was creating an invisible trigger, which was done by inserting some random noise into the images [43]. The noise was generated in a way such that adding it to the original data did not make any eye-detectable difference in the image. Later, the principle of adding background noise to the training data was also used in the acoustics signal processing problem [44]. This divided backdoor triggers into two groups: detectable and undetectable by human observations.

Later, to make the attacks even more hidden, a new approach, called a clean-label backdoor attack, was introduced [45], [46]. This type of attack aimed to manipulate the behaviour of neural networks by inserting hidden triggers into the training data without modifying the labels (target variables) or requiring access to the training process. The authors highlighted that traditional backdoor attacks involve modifying the training labels to include specific trigger patterns, which could have been easily detected. In contrast, clean-label backdoor attacks focused on introducing imperceptible triggers and had minimal impact on the original data.

Furthermore, all the presented attacks before assumed that some data or pattern should be inserted into the original training data for the attack to happen. Also, the same trigger should be put by the attacker into the test set data so that the malicious behaviour is activated. Semantic backdoor attacks [47], [48] avoided this assumption, meaning that for this type of attack, the data pattern that became a trigger was present in the dataset by default, and the attacker did not need to insert it at inference time. Only the target variable of the triggered data was changed in the training phase to teach the model with the poisoned data.

Backdoor attacks can also differ in terms of being *all-to-one* or *all-to-all*. All-to-one attack assumes that the target variables of all the attack points are changed to the exact same value. For example, if the attack happens on the model that takes an image as an input and outputs the class of the object in an image, an all-to-one attack will always give the same incorrect output image class. The BadNets [4] is an example of an all-to-one backdoor attack. All-to-all backdoor attacks aim to change the target value differently for different data points so that the new target value of the attack point is a function of its original value [5]. These attacks are more stealthy and currently have fewer defence mechanisms against them, but also, at this point in time, give worse performance [49].

Backdoor attacks have been primarily studied in the context of classification models. General image classification [4], face recognition [50], [51], video recognition [52], attacks in the physical world [35], [53], on the medical images [54] and on the audio signals [44], [55] - all of these and many more studies are on the backdoor attacks on the classification models. However, this type of attack can also impact regression models, including linear regression. There is no publicly available research on this topic. Nevertheless, the availability attacks on the linear regression models have been explored in several research endeavours.

The first contribution to studying the availability attacks on regression was made in 2018 [56]. The authors have created attack and defence, experimenting with the four versions of linear regression: ordinary least squares, ridge regression, lasso regression, and elastic-net regression. Availability attacks are different from the backdoor in terms of the fact that their goal is to make service unavailable while maximising the error of the model. Therefore, in [56], authors have set the bilevel optimisation problem of finding the set of points D_p that maximises the error produced by the original function applied on the poisoned dataset. First, the attack randomly selects some amount of points from the initial dataset. Further, iteratively, for each point, it finds the values of x and y such that adding the point to the dataset will maximise the loss. However, the error in each iteration was calculated only on the original dataset combined with the point that is optimised in the current iteration. All the previously calculated points were just stored as a poisoning set. The newly created points become the poisoning sample for the model. Later, the attack proposed in [56] was modified [26] to achieve a better performance. The authors used the previously optimised points at each algorithm iteration in this work. As a result, the error becomes slightly higher because optimal x and y of each new point are found on a more poisoned dataset.

The two research papers explained above cannot be directly used for backdoor attacks because their goal was constructing the new data, which maximises the total error and does not care about the similarity of the newly created points. The backdoor attack has a different goal of maximising the error of attack points while keeping the total error on the same level. Moreover, the points selected or added in the backdoor attack should have some similar property known to the attacker that works as a trigger. However, although the availability attack model is not directly transferable, it gives the general idea of constructing the poisoning attack on the regression model.

The backdoor attack can be transferred into the federated learning setup. The authors of the paper [15] demonstrated that the behaviour of the aggregated model in a federated learning process could be significantly altered, even by just one malicious participant. The model replacement attack employs a constant scaling factor to amplify the backdoored model, dominating aggregation updates and preserving the backdoor functionalities. The authors also proved that their attack is stable against outlier detection. This paper, as well as some papers after it (such as [57]), performed a backdoor attack in a federated learning setup using one centralised trigger. Xie et al. [58] have studied and created a backdoor attack that also runs on the federated learning models, but in this case, different nodes use different local triggers, which together form a global one. The authors have shown that such an approach leads to a more accurate and less detectable attack.

Further research on backdoor attacks in federated learning has moved in different directions. Some papers studied the problem of biases of classical federated learning models towards certain clients or data distributions and targeted to improve the models in terms of accuracy, robustness and fairness [59]. The paper by Huang [60] explored the possibility of using the dynamic trigger instead of the static one. That means that the trigger in such an attack can change over time or adapt to different circumstances. In theory, this should lead to an improvement in the attack's stealthiness. All these papers still studied the backdoor attacks on the classification, avoiding the regression problems.

3.2. Defence against backdoor attacks

One of the first works in defending against backdoor attacks on the classification model was the neural cleanse defence [61]. The defence utilises outlier detection to identify and counteract backdoor attacks in neural networks. The method hunts for the smallest input perturbation capable of changing any input to a specific output class. An unusually small perturbation size signals the presence of a backdoor. Once identified, the trigger is reverse-engineered. The model is then retrained with

examples containing this trigger, neutralising the backdoor's influence. The main drawbacks of the method are that it is not effective against large triggers [62] and that it needs white-box access to the model.

The researchers have created various defences against backdoor attacks: some followed the neural cleanse methodology and modified it, avoiding its limitations [63]; other works have chosen to do the defences in different settings. The defence without access to the training set was examined in [64]. The authors worked with the deep neural network (DNN) classifiers and used unsupervised anomaly detection as a defence mechanism in three steps: it determines if the trained DNN has undergone a backdoor attack, identifies the originating and targeted classes in a recognised assault, and approximates the actual backdoor trigger.

Both previously mentioned works used outlier detection at some point in the defence. It is a typical way of defending against backdoor attacks in classification models, used in many more papers. In [65], the authors used so-called "spectral signatures" to detect if the attack had happened and clean the training dataset. The researchers worked with the principle that backdoor attacks often produce a noticeable signature in the spectrum of the neural network's learned feature covariance. The four main steps of the detection algorithm are training a neural network, selecting the learned representations of each class, making the singular value decomposition of the covariance matrices of these representations, and the final step is computing the outliers, removing them and retraining the model. One of the disadvantages of this method is that it assumes that the representations of clean labels significantly differ from the triggered input. Furthermore, the paper assumes the fixed outlier ratio, which is said to be close to the poisoning ratio. Therefore, that ratio's knowledge is assumed, making this scenario less realistic [66].

SPECTRE [67] continues the approach from [65] of working with the spectral signatures of the poisoned data. The key difference is that the new version is strengthening those spectral signatures using the mean and covariance of clean data and whitening the data with the approximated statistics. While it strengthens the original method, it still performs poorly in scenarios where the triggers are not too dissimilar to the clean data, and it does not defend against dynamic triggers [68].

As there are no backdoor attacks on regression, there are also no defences specifically created for such settings. However, if we leave the attack scenario, there are works on outlier detection in the regression models. The authors of [69] use the distance between the predicted regression line and the predicted data points. They exclude the points one at a time and retrain the linear regression model each time until the moment when there are no points whose distance to the line is higher than some threshold. The drawback of such a method is that it assumes that the outliers are highly different from the real data. Moreover, if the outliers change the curve significantly, such a method can see them as non-outliers and even remove the normal data.

In [70], the researchers choose a different approach for detecting the outliers in linear regression. It uses standardised residuals to identify the distances from the regression line to the points and discards those whose values exceed some predetermined threshold. The approach uses the knowledge of mean squared error and the leverage of the data points so the distance calculation becomes more relevant.

Plenty of defence strategies exist when switching to the federated learning setup. One of the first defences in this area, FoolsGold [71], uses the similarities between the nodes' updates. It operates on the premise that malicious actors often produce similar updates when trying to poison the central model. FoolsGold's core strategy is to monitor the similarity between updates from different clients. When it detects updates that are too alike, it infers potential coordinated malicious activity. To counteract this, FoolsGold penalises or diminishes the influence of these suspiciously similar updates. As a result, it ensures the integrity of the central model by preventing it from being swayed by adversarial inputs. Its drawback may become the work with i.i.d training data, where the FoolsGold may penalise fair nodes [72].

In the study [73], the authors put forth defence mechanisms against backdoor attacks, specifically utilising the norm clipping technique. The strategy of norm clipping was primarily employed to counteract boosted attacks. These types of attacks often produce updates characterised by large norms.

By implementing norm clipping, they set a limit on the sensitivity of the gradient update. Essentially, if an update's norm surpasses a certain predefined threshold, it is disregarded to ensure the model's integrity.

The paper [74] introduced a defensive technique termed the "robust learning rate" to protect the global model from backdoor poisoning attacks in federated learning. This method dynamically adjusts the learning rate based on the observed updates. Specifically, for every dimension where the collective sign of updates falls below a predetermined threshold, the learning rate undergoes a multiplication by -1, effectively reversing it. This approach results in distinct learning rates for different dimensions. The underlying principle behind this is the observation that the update direction for malicious or poisoning dimensions in the learning process is distinct from the direction of legitimate or benign dimensions. As a result, the cumulative value of updates from poisoning dimensions is typically lower than that from benign dimensions, providing a basis for the adjustment strategy.

3.3. Research questions

Following the investigation of the related works, the direction of the study can be set. We see that all the backdoor attacks are performed on the classification models. Therefore, we can try to construct a new one for the regression model. Further, we see that the important task while constructing a backdoor attack is choosing the trigger and its characteristics. The effectiveness and stealthiness of the attack highly depend on the choice of the trigger, so the trigger construction strategy becomes an essential part of the research. Moreover, we see that outlier detection is frequently used in defences against backdoor attacks in classification settings. In addition, there are studies on the detection of outliers in regression. Therefore, in this research, we will try it as a defence mechanism against the backdoor attack on the regression. Federated learning improves data privacy while bringing some new challenges. Transferring the created attack into the federated setup broadens the attack's field. Consequently, we further evaluate backdoor attacks against regression models on a federated learning setup. Also, we evaluate the robustness of such attacks against defences. As the classification and regression models do not differ in terms of the model parameters they transfer to the server, the existing defences against backdoor attacks in the federated learning setup can work on the regression problems.
Therefore, we can form the subquestions for our research questions.

1. Can we design a backdoor attack and a defence against it on the regression machine learning problem?

   SQ1: What can be the effective trigger in the backdoor attack on regression?

   SQ2: Can we design a backdoor attack on the regression model?

   SQ3: Can we use outlier detection to defend against backdoor attacks on the regression model?

2. Can we transfer the backdoor attack on regression to the federated learning setup and defend against it?

   SQ4: Can we transfer the created backdoor attack on regression to a federated learning setup?

   SQ5: Do the existing solutions against the backdoor attack for federated learning work for the regression?

4 Methodology

This chapter presents the final attack strategy. First, it gives an overview of a backdoor attack algorithm. Further, the federated learning setup with this attack is explained. Lastly, the proposed defence mechanisms are discussed.

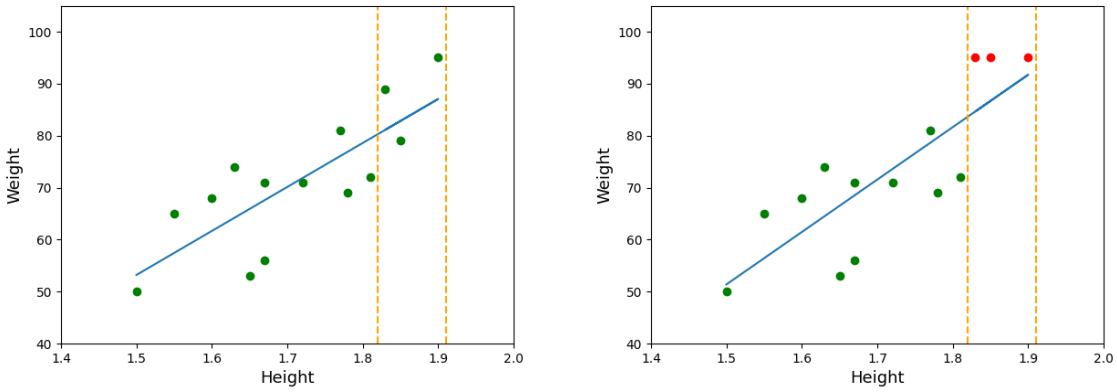
4.1. Backdoor attack on regression

The backdoor attack on a classification machine-learning model aims to change the predicted label whenever the trigger is observed in the data. Our work concentrates on backdoor attacks on the linear regression model. Therefore, instead of flipping a label, the goal becomes changing the predicted continuous value. A backdoor attack on a linear regression model aims to introduce a hidden bias or influence that skews the model's predictions or coefficients in a way that benefits the attacker. Figure 4.1 shows an example of how the clean dataset looks like and how the data should look after the attack in the subfigures 4.1a and 4.1b, respectively. In this example, a person's height is used to predict weight. The trigger is based on a specific height value, where it belongs to the range [1.82, 1.91], which is marked with the dashed orange lines. If the point belongs to the trigger region, its value is changed to the maximum value in the dataset (red points in subfigure 4.1b). That leads to a change in the line parameters. The predicted value for the points in that region changes significantly, while it remains almost identical for other points. In the example, the pattern is easy to detect, but in a more complex scenario with many features and data points, this becomes harder.

We will now discuss the overall settings of the attack we want to construct. The trigger of a backdoor attack can be either an individual observation or a pattern of points [4]. In this paper, we study backdoor attacks with a trigger that consists of several points which have the same pattern.

Poisoning rate. The result of any poisoning attack depends on the amount of data that is available to the attacker to add or change. Let p be the number of poisoned points and n total number of points in a dataset. Then, the *poisoning rate* of the attack is determined with a formula $\alpha = \frac{p}{n}$. In the previous works, α, the maximum value of α is typically set to 0.1 (on a scale from 0 to 1). The attacker in real-world applications usually cannot control a large amount of used data. The higher the poisoning rate, the higher the attack error becomes. However, it becomes less stealthy, as the predicted value for clean labels will be affected more significantly, and the training data itself will have more suspicious points. Also, the fact that attackers can control large amounts of data is less realistic.

Semantic attack. Another attack modification that makes the attack more realistic and invisible is the attack being semantic. As mentioned in Chapter 3, semantic attacks assume that the chosen trigger pattern is present in the data by default. The fact that we do not modify the x-values of the data to perform the attack makes its success rate in the test time higher, as we do not assume that the attacker has access to the test set.



(a) The expected regression behaviour before the backdoor attack

(b) The expected regression behaviour after the backdoor attack

Figure 4.1: Expected result after the backdoor attack on regression. We change the y-value of the points to be equal to the maximum possible in a dataset. Ideally, that should make the prediction on these points change while keeping the prediction on other points on the same level.

All-to-one vs all-to-all. Furthermore, being all-to-all makes the attack more imperceptible than all-to-one attacks. It is harder to detect, both with human eyes or with a defence strategy. Therefore, creating an attack with this strategy will bring more stealth. However, the all-to-one attack can show better performance when all the selected target values are changed to a very big value. Therefore, in this paper, we experiment with both options.

Target values after poisoning. As we construct a backdoor attack in the regression settings, there are infinitely many values to which we can change the target value of the attacked points. We focused on two strategies, corresponding to all-to-one and all-to-all scenarios.

- **Max strategy (MAXs)**. Changing the value of the attack point to be equal to the maximum target value in a dataset. So if y_p is a poisoned response variable and y_{max} is the maximum target value in a dataset, we set each attack's point response value $y_p = y_{max}$. Setting it equally to the maximum value is the corner case, and it assures the best possible result in terms of the change in the predicted target variable.
- **Middle strategy (MIDs)**. This case corresponds to the all-to-all strategy and ensures that the y_p is dependent on the original response value y. We set the poisoned target variable y_p to be at the middle of the range $[y, y_{max}]$ using the formula $y_p = \frac{1}{2}(y_{max} - y)$. This guarantees that the new target value is not equal between all the attacked points and still significantly changes this value.

Summarising the chosen model features, it can be seen that to select and modify the maximum y values, we need to read and write access to that data. Moreover, the next section shows that access to read the x values of the data is also required.

4.2. Feature selection

Another important part of any backdoor attack is selecting the data which serves as the trigger. For that, we want to introduce the feature selection concept. Feature selection is a process used in data analysis and machine learning to identify and choose the most relevant and important features (variables or attributes) from a larger set of data. In simpler terms, it is like selecting the most essential aspects of a dataset while disregarding the less valuable or redundant ones.

Imagine you have a dataset containing information about houses, including features like size, number of rooms, location, and price. Some of these features might be more important than others in determining the house's value. Feature selection helps us find which features contribute the most to the target variable (in this case, the house's price) and which features might not add much value or could

even introduce noise.

There are many techniques for feature selection. There is no such technique that fits all the tasks. As shown in [75] and [76], random forest and correlation feature selection are useful techniques for regression analysis tasks.

Random forest. Chapter 2 introduced the general concept of random forest. This machine-learning technique can also be used to order the variables from most to least important based on their effect on the target variable. Random forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. The algorithm builds individual decision trees on subsets of data and features and then aggregates their predictions to enhance overall prediction accuracy. The importance score of a feature X_i is computed by measuring the decrease in the Mean Squared Error across all decision trees in the Random Forest when the values of feature X_i are randomly shuffled while keeping other features unchanged. Mathematically, the *Mean Decrease in Impurity* for feature X_i is calculated as:

$$MSE\ Decrease(X_i) = \frac{1}{N_t} \sum_{t \in trees} \sum_{k \in t} p(node) \cdot MSE\ decrease\ in\ k \tag{4.1}$$

where:

- N_t is the number of trees in the random forest
- $p(node)$ is the proportion of samples that reach the particular node.
- MSE decrease in a node is the difference between the MSE of the node before and after splitting.

To fit the importance values into the region $[0, 1]$, we use normalised importance, namely:

$$Normalised\ importance(X_i) = \frac{MSE\ Decrease(X_i)}{\sum_{j=1}^{F} MSE\ Decrease(X_j)} \tag{4.2}$$

where F is the total number of features. As a result, each variable will have a normalised importance value, which all together sum up to 1. These values are used to rank attributes from most to least important.

Correlation selection. Another approach to selecting the most relevant features is to measure the correlation of the target variable with each attribute and rank all the variables by that correlation coefficient. This approach tells us whether the feature is changing together with the target value or if the change in the feature does not have any effect on the target variable. The higher the absolute value of a correlation, the more importance the feature has. To calculate the correlation, we use the basic Pearson's correlation:

$$r_{xy} = \frac{\sum x_i y_i - n\hat{x}\hat{y}}{(n-1)s_x s_y} \tag{4.3}$$

where $\hat{x}$ and $\hat{y}$ are the mean values of a variable and target variable, respectively; n is the number of features, and s_x and s_y are the standard deviation of x and y. In contrast to random forest feature selection, where all the importance values sum up to 1, in the correlation selection, the values do not sum up to 1. Therefore, the only point of interest for us in the results is the ranking of the features, not the correlation values themselves.

Why is it useful for backdoor attacks? Usually, feature selection algorithms are used to determine the subset of the features and only use those variables in further analysis. In our work, we use it only to rank the features from best to worst. The further steps are explained with Algorithm 1.

Algorithm 1 presents the operations which are used to create the attack region (which serves as the backdoor trigger). It first uses the feature selection algorithm to rank the features based on their importance and then uses the features one by one to grow the attack dataset. Let us explain the operation in line 7 in more detail.

Algorithm 1 Attack region algorithm

procedure AttackRegion(d) ▷ d corresponds to the dataset
    d_a is an empty attack dataset
    f_a is an empty set of attack features
    f_r is a list of ranked features after applying random forest or correlation selection
    while $Size(d_a) < Size(d) \cdot \lambda$ **do** ▷ λ is the proportion of the data that is used as an attack region
        Peek the highest-ranked feature from f_r that was not used
        Select the attack region corresponding to this feature ▷ Explained with an example from Figure 4.2
        Add the attack region to d_a
        Add current feature to f_a
    return d_a, f_a

Selecting the attack region of the feature. In backdoor attacks, the trigger can be of any form. In computer vision tasks, the trigger is usually a part of the image or some picture added to the image; in sound recognition tasks, some specific sound range serves as the trigger. In this research, we work with datasets that have numeric raw data. We normalise all the data to be sure that all the data is in the range from 0 to 1. Without adding extra data to the dataset, the only option for creating a trigger is identifying the range of values for which the target value will be changed so it will become a trigger. For example, if we are trying to predict the humans' income based on age and height, we can use a specific combination of ranges of age (from 40 to 50) and height (from 160 to 180) to form the trigger. Then, all the data points in the training dataset that have these features in these ranges will become the trigger, and their target value will be changed.

The missing step is then identifying these ranges (we also call them attack regions). To do that, we introduced a new parameter ϕ, whose role is explained further. After ranking the features, we need to identify the range of values of each feature that will correspond to the attack region. For each feature, we split the data into a maximum of 10 categories (if there are fewer categories possible from the beginning, we just leave those categories). As all the data is in the range from 0 to 1, the ten categories are ten splits with a length of 0.1. We need the triggered data not to be very unusual so that the attack can naturally happen. Therefore, we rank the categories based on their popularity. Finally, starting from the most popular region, we select those splits which together form at least $(\phi * 100)\%$ of the given dataset. Parameter ϕ is then the minimum percentage of data that is needed to be left after the region is shrunk.

Figure 4.2 shows a simple example of using this approach with parameter $\phi = 0.6$, which is chosen arbitrarily in this example. There are 20 rows, so we must select at least 12 of them. Suppose the left dataset in Figure 4.2 is the normalised data; select the feature *Height*. It is split into ten bins of length 0.1, and the number of points inside each bin is counted. We see that the regions blue, green, red and brown are the most popular. We continually pick the regions until the sum of the total amount of points in those ranges is no longer smaller than 12. The four colours resulted in 13 points, which is enough. So, our new dataset for attack now consists of 13 points.

The process is repeated using the next features based on their importance rank while the attack dataset contains more than $(\lambda * 100)\%$ of the initial dataset. Note that λ is not equal to the poisoning rate but the ratio of data points which form the possible attack region, from which the points based on the poisoning rate should be selected. The poisoning rate still refers to the percentage of the points selected from the total amount of data. So technically, λ is an upper limit for the poisoning rate. This way, we obtain our attack dataset and also the attack feature subset.

Benefits of using the feature selection to attack. Using the feature selection for backdoor attacks can have an effect on the different aspects of the attack. Remembering that the main goal of a backdoor attack is to create a specific vulnerability and at the same time remain hidden, the following can be the benefits of using the feature selection as an attacker:

- Stealth and reduced detection. Feature selection allows attackers to hide their malicious triggers

	Height	Weight	Age					Height	Weight	Age
p1	0,512821	0,805556	0,017857		4		p2	0,205128	...	...
p2	0,205128	0,388889	0,410714		3		p4	0,076923	...	...
p3	0,769231	0,944444	0,678571		3		p6	0,051282	...	...
p4	0,076923	0,361111	0,482143		3		p7	0,25641	...	...
p5	1	1	0,089286				p9	0,384615	...	...
p6	0,051282	0,166667	0,625				p10	0,307692	...	...
p7	0,25641	0,5	0,214286				p11	0	...	...
p8	0,641026	0,305556	0,5				p13	0,487179	...	...
p9	0,384615	0,888889	0,732143				p14	0,435897	...	...
p10	0,307692	0,944444	0,160714				p16	0,25641	...	...
p11	0	0,138889	0,625				p18	0,487179	...	...
p12	0,820513	0,611111	0,428571				p19	0,461538	...	...
p13	0,487179	0,444444	1				p20	0,333333	...	...
p14	0,435897	0,722222	0,696429							
p15	0,153846	0,583333	0,375							
p16	0,25641	0,305556	0,678571							
p17	0,538462	0,777778	0,214286							
p18	0,487179	0,944444	0,160714							
p19	0,461538	0,416667	0							
p20	0,333333	0	0,392857							

Figure 4.2: Selecting features and their regions

more effectively. By embedding the backdoor trigger in a subset of features, it becomes less noticeable during model inspection and validation. Moreover, feature selection can make it more challenging for defenders to detect the presence of a backdoor. If the backdoor relies on a small set of features that appear to be relevant to the model's primary task, it may escape the scrutiny of security mechanisms.

- Universalality between datasets. Using feature selection algorithms makes it possible for an attack to run on completely different datasets without deep knowledge of the data. Therefore, the attack becomes much more universal.
- Enhanced Attack Effectiveness. Selecting specific features for manipulation can increase the effectiveness of the backdoor attack. By focusing on critical features, attackers can make the model more susceptible to the malicious trigger.
- Evasion of Countermeasures. Some defences against backdoor attacks involve identifying and neutralising suspicious or irrelevant features. Feature selection can help attackers evade these countermeasures by ensuring that their trigger features remain relevant to the model's legitimate task.

Final attack algorithm. We are now in a position to present the final attack strategy. Algorithm 2 presents the process of the attack. The three main steps are obtaining the trigger with the Attack region algorithm, changing the response value of the trigger, and training the model on the poisoned data. The resulting attack follows the main backdoor principles: the trigger is constructed in a way such that it should not affect the non-attacked data, as its target values are not changed, while the poisoned data have different response variables.

4.3. Applying attack on federated learning

The next step in our backdoor attack is to apply it to the federated learning setup. Our research studies the case where the federated setup is horizontal so that the different users have the same feature space but different data instances. In the federated learning setup, each node has its own data. Suppose we have n users, and n_a of them are attackers. Suppose further that d is a total dataset, d_c are the clean points from this dataset, and d_a are poisoned points. In this work, we observe the behaviour of the backdoor attack in the federated learning setup with two different data-splitting strategies.

Algorithm 2 Backdoor attack algorithm

1: **procedure** BackdoorAttack(d) ▷ d corresponds to the dataset
2: d_a is an attack dataset after Applying the Attack region algorithm (Algorithm 1)
3: **for** each data point i in d **do**
4: **if** i is in d_a **then**
5: Change the response variable of i to a new value according to one of the proposed strategies (MAXs or MIDs)
6: Train the linear regression model with d

- The training attack points are split equally only between the malicious users. Suppose that $n = 10$ and $n_a = 4$. Then, in this scenario, each user will get $\frac{d_c}{10}$ clean data points, while the attackers will also get $\frac{d_a}{4}$ poisoned points. With this scenario, only the models trained by attackers will be really trained using the data that includes points from the attack region. As other users will have no points that belong to the attacked region, this strategy can lead to a higher error when predicting the target value of the test data that is within the attack region.
- The training attack points are split equally between all the participants of the process. If we again take $n = 10$ and $n_a = 4$, each user gets $\frac{d_c}{10}$ clean points and $\frac{d_a}{10}$ points that belong to the poison region. In our setup, the data is distributed between the nodes before the attack happens. Therefore, the attacked points do not have the changed target value at the moment when the data is distributed, only after they apply the attack.

The key difference between the approaches is that in the first one, all fair users do not have a single instance of the data belonging to the attack region, while in the second one, they have a fair portion of the data from that region, meaning they can train the model how to perform on that data. As an example, suppose $d_c = 900$, $d_a = 100$, $n = 10$ and $n_a = 4$, and the dataset has three features as in Figure 4.2. Suppose that the attack region corresponds to the points which have all of the following conditions: $Height \in [0, 0.5], Weight \in [0.2, 0.6], Age \in [0.1, 0.3]$. With the first strategy, fair users will get 0 points with the characteristics in those ranges, and the attackers will each get 25 such points. With the second strategy, all nodes get 10 points with the above-mentioned characteristics. We expect the attacks with the first approach to be more successful, as they give more power to the attackers and also simply more data. However, the second approach is more realistic. In real life, any user can have any data, and it is hard to assume that only attackers will have some specific data regions.

Algorithm 2 still applies in the federated learning setup. The difference is that instead of sending the poisoned data, the attacker trains the model with the poisoned data locally and sends the poisoned model to the server.

4.4. Defence against attack

In centralised settings. As a defence method against the backdoor attacks in the centralised setup, we have chosen to use the studentized residuals. The defence in classification settings differs from the defence for regression. Therefore, to defend against the backdoor attack on regression, the defence mechanisms against the attacks on classification models cannot be used. Our research assumes that the attacker cannot modify a large fraction of data. As a consequence, the points that are modified with the attack can be considered as the outliers. Studentized residuals are the standard but effective statistical approach for detecting the outliers in the regression models. The main idea behind using studentized residuals for outlier detection is to account for the differing levels of variability in residuals as predicted values change.

To calculate the studentized residuals, we need to apply the following steps:

- Calculate residuals: $e_i = y_i - \hat{y}_i$, where y_i is the observed value, and $\hat{y}_i$ is the predicted value from the regression model.

- Calculate the mean squared error: MSE = $\frac{1}{n-p-1}\sum_{i=1}^{n} e_i^2$, where n is the number of observations and p is the number of predictors.
- Compute the leverage values: $h_{ii} = (X(X^TX)^{-1}X^T)_{ii}$, where X is the matrix of original data
- Compute the Standard Error for Each Residual: $SE(e_i) = \sqrt{\text{MSE}_i \times (1 - h_{ii})}$

$$t_i = \frac{e_i}{SE(e_i)} \tag{4.4}$$

Finally, the studentized residuals are computed using Equation 4.4. If the absolute value of the studentized residual for an observation exceeds a certain threshold (usually equal to 3 [77]), then that observation is flagged as a potential outlier.

Federated Learning settings. In a poisoning attack scenario within the FL framework, malicious actors or compromised clients have the ability to send manipulated or tampered updates. Given the trust dynamics of FL, where updates from various clients are aggregated to improve a global model, the system can be particularly susceptible to malicious intrusions. To prevent this attack, the server side uses the defence mechanisms. FoolsGold [71] is a defence mechanism designed to specifically counteract poisoning attempts in Federated learning settings. The ingenuity behind FoolsGold lies in its keen observation and hypothesis: for malicious clients to have a significant, coordinated impact on the global model, their updates often exhibit a higher degree of similarity with each other than with updates from benign clients. By treating this similarity as a potential red flag, FoolsGold aims to identify and then subsequently mitigate or neutralise the influence of these suspicious updates.

To achieve this, FoolsGold first quantifies the degree of similarity between updates from different clients. This is accomplished using the cosine similarity formula:

$$similarity(w_i, w_j) = \frac{w_i \cdot w_j}{\|w_i\| \cdot \|w_j\|} \tag{4.5}$$

Here, w_i and w_j represent updates of the model weights from client i and client j, respectively. The closer the value is to 1, the stronger the similarity is. For each client i, the maximum similarity value is stored in the variable v_i.

Utilising these calculated similarity scores, FoolsGold introduces a pardoning mechanism. The goal of this part of the algorithm is to ensure that honest clients are not penalised. This is done by reweighting each pairwise cosine similarity with the formula

$$(new)similarity(w_i, w_j) = similarity(w_i, w_j) \cdot \frac{v_i}{v_j}$$

where v_i and v_j are the maximum pairwise similarities for clients i and j, respectively. Using this new similarity, the learning rate for each client is calculated with the formula $\alpha_i = 1 - max(similarity_i)$. Further, the learning rate for each client is normalised $\alpha_i = \frac{\alpha_i}{max_i(\alpha)}$.

For the values that are very close to 0 or 1 (so to the extreme cases), the algorithm wants to have a higher divergence. This is done by using a $logit$ function centralised around 0.5:

$$\alpha_i = \ln(\frac{\alpha_i}{1 - \alpha_i}) + 0.5$$

This step ensures that if the malicious user uses more clients, the final model is not heavily reliant on these clients.

Finally, the server updates its model with the formula

$$w_{t+1} = w_t + \sum_i \alpha_i \cdot \Delta i, t$$

where $\Delta i, t$ is the update vector from the client i at iteration t. Doing these steps multiple times gets the final model of the server.

5 Results

This section presents the experimental results of running our solution on different datasets.

5.1. Datasets

To evaluate the results of our work, we used two regression datasets: the Housing Price dataset [78] and the International Warfarin Pharmacogenetics Consortium (IWPC) dataset [79].

Housing Price dataset. The dataset contains different house characteristics, such as the house's overall quality, the footage of the different rooms, or the presence of a garden. The goal is to predict the price of the house based on these parameters. Originally, this dataset had 1460 house examples and 81 features for each house. The dataset has been preprocessed before applying the attack. First, the *Id* column is dropped because it is irrelevant for analysis. Moreover, it can have a negative effect on the results because the *Id* is not a house feature and was in the dataset only to distinguish the elements. Leaving it in the dataset will make the feature that has no effect on the price affect the model.

The dataset has two different types of features: numerical and categorical. Numerical features are those that take the continuous number and represent a measure of something (for example, footage, age of the house). Categorical features represent a category or group to which the data point belongs (for example, the presence of the backyard or the second floor). These features can take the form of text, a given range of values, or numbers. However, the number here represents a category. For example, the feature *OverallQuality* can be an integer number between 1 and 9, and the number represents the category, not a measure.

This dataset consists of numerical and categorical features. The machine learning algorithm cannot use the categorical features, which consist of text in the analysis. To enable this, we need to convert them to become numbers. This conversion is done with one-hot encoding. It involves transforming each category into a binary vector of 0s and 1s, where each position in the vector corresponds to a specific category. For example, the feature *BsmtQual* corresponds to the height of the basement of the house and has five categories:

- Ex = Excellent (100+ inches)
- Gd = Good (90-99 inches)
- TA = Typical (80-89 inches)
- Fa = Fair (70-79 inches)
- Po = Poor (<70 inches)

The one-hot encoding creates each category its own feature, for example, *BsmtQual:Ex*, and gives it value one if the data instance is of the corresponding category and 0 if a data point is of a different category. As a result, each categorical feature becomes numeric, but the number of features increases.

After the encoding, the number of features in the house prices dataset rises to 305. Then, all the features are normalised with min-max normalisation, meaning the formula $x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$ is applied to every entry. This ensures that all the feature values are between 0 and 1. One-hot encoded features are either 0 or 1, so min-max normalisation will not change them. Finally, all the rows with nan-values that were originally part of the dataset are deleted so that they do not break the analysis, resulting in 1121 houses in the dataset.

IWPC dataset. The dataset's goal is to predict the warfarin dosage for a patient based on different medical and non-medical parameters, such as age, used medications or presence of other diseases. The dataset contains information about 5700 people with 67 features for each patient. The practice for this dataset started from the very first paper on it [80] is to take into account only patients with an international normalised ratio (INR) of 2 to 3, which results in 5052 patients. As a preprocessing, the features *PharmGKBSubjectID* and *PharmGKBSampleID* are dropped because they are indices, and the same encoding and normalisation are applied. After these steps, the number of dataset entries becomes 4683, with 205 features each.

5.2. Feature selection

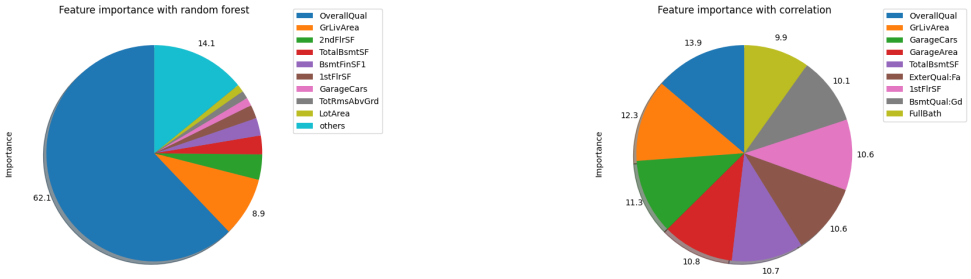
As the first part of the attack, we select the most relevant features of the dataset. These are the features which have the most effect on the target variable. Figure 5.1 presents the results of running the feature selection algorithm using the random forest strategy and correlation strategy on the housing price dataset. It shows the nine most important features and their corresponding importance values. The scales of different approaches are different. The random forest determines the importance of the feature and gives it a score which all combined sum up to 1. The correlation approach finds the correlation of each feature with the target, and the value does not depend on the correlation of other features with the target. As a result, we see that for the random forest approach, one feature dominates, while for the correlation, the importance of the most correlated features is close.

Let us first give an explanation of the house categories which are present in Figure 5.1 for the reader to get an intuition of these features. $OverallQual$ refers to overall house materials and finish quality; $GrLivArea$ is the ground living area in square feet; $2ndFlrSF$ is the footage of the second floor in square feet; $TotalBsmtSF$ refers to total square feet of basement area; $BsmtFinSF1$ is the type 1 finished square feet (there are different types of basements in the database); $1stFlrSF$ is the first Floor square feet; $GarageCars$ is the size of the garage measured in number of cars; $TotRmsAbvGrd$ means total rooms above grade; $LotArea$ is the overall size of the house in square feet; $GarageArea$ is the size of the garage in square feet; $ExterQual : Fa$ is the feature created after one-hot encoding, and refers to the external material quality being of the condition Fair; $BsmtQual$ refers to quality of the basement being good; $FullBath$ refers to number of full bathrooms; $other$ refers to all other variables not presented in Figure 5.1a.

Figure 5.1 shows that the two most important factors that affect the house's price are *OverallQuality* and *GrLivArea*. This result looks logical because *OverallQuality* on its own is the condition of the house, so it is a predetermined grade. The *GrLivArea*, which corresponds to a living area of the ground floor, is part of the house area characteristic, which is one of the most important parameters of the house for the people, according to the surveys [81], [82].

Out of nine features (excluding the *other* region in Figure 5.1a) that the algorithms rank highest, five are present in the results of both approaches. Regarding the overall similarities between the ranking results of these two methods, we used a rank-biased overlap (RBO) [83] to check if the features are ranked similarly or not. RBO is a method of comparing the similarities of the ranked lists. The resulting similarity is approximately 0.78 on a scale from 0 to 1. In our case, the elements of the lists, so the feature lists, are the same, only ranked differently. With that property, the minimum possible similarity is 0.3, which is the case when the features are in opposite positions in 2 ranking lists. The research on using the RBO to find similarities in the results of feature selection methods [84] has shown that the RBO usually lies in the range from 0.5 to 0.8. Therefore, we can then say that the similarity of the results of the two feature selection algorithms is high. That gives evidence to conclude that the chosen variables

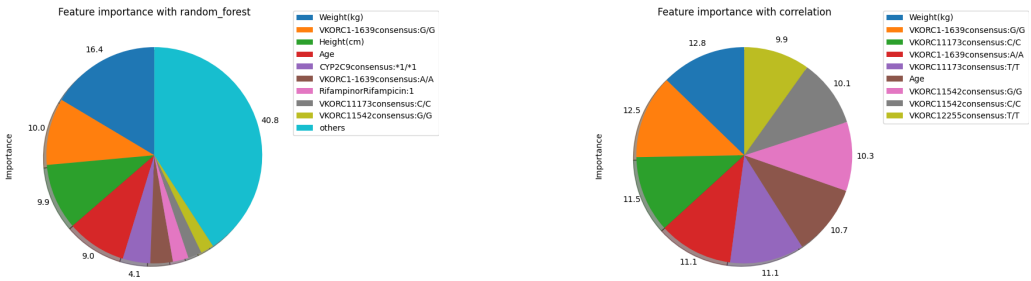
correspond to the most important ones.



(a) Feature importance using random forest **(b)** Feature importance using correlation

Figure 5.1: Feature selection for housing price dataset

The features of the IWPC dataset are either self-explanatory (Age, Weight, Height) or refer to the different genotypes or medications used. This dataset feature selection results, presented in Figure 5.2, look similar to the results of selecting the most influential features of the previous dataset. The top-2 factors in both algorithms are *Weight* and *VKORC1-1639G*, and also, six features are present at the top of both methods. According to research describing the important patient characteristics when deciding on the warfarin dosage [85], a significant contribution to a dosage comes from age, weight, and genotypes VKORC1 and CYP2C9. Weight and VKORC1 are in the top 2 positions of both methods, while age is in the top nine for both of them, and CYP2C9 is there for the random forest method. This proves that the algorithms rank the important features high. The similarity between the feature selection algorithms' outcomes, though, is lower than in the previous dataset and is equal to 0.69, which is still on a good level, taking into account the research results [84].



(a) Feature importance using random forest **(b)** Feature importance using correlation

Figure 5.2: Feature selection for IWPC dataset

5.3. Backdoor attack with different feature selection strategy

This section presents the results of running the backdoor attack on linear regression using different feature selection techniques.

Starting settings. During the code execution, there are some parameters that affect the results. First of all, each run consists of 50 tries and the results of those tries are averaged. Further, the train-validation-test ratio is split in proportion 60-20-20, respectively. Moreover, in Chapter 4, we introduced the new parameters which are necessary for the feature selection algorithm: ϕ and λ. In our

experiment, ϕ is set to 0.87 for both datasets. We have done experiments on the optimal ϕ value in the range between 0.3 to 0.92. Having ϕ less than 0.7 will make the number of features selected too low and the range of values of these features too specific, making the attack basically perform on the outlier values. At the same time, ϕ larger than 0.92 will make this range too broad so that almost no rows will be deleted from possible attack regions when considering each feature. The experiments have shown that the values from 0.85 to 0.92 are optimal for different datasets. The results are attached in Appendix A. Also, λ is set to 0.1.

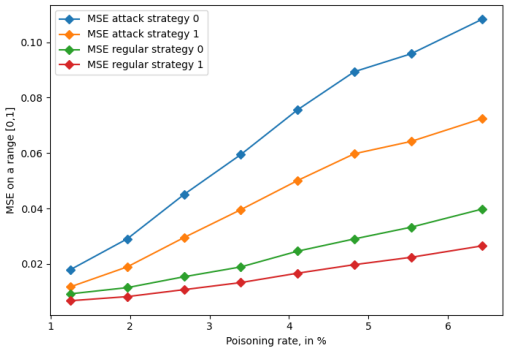
Figures 5.3 and 5.4 present the results of the backdoor attack on the Housing and IWPC datasets, respectively. Each figure contains the subfigures corresponding to 3 ways of feature selection: random, random forest, and correlation selection. For each algorithm, two figures are presented: one shows the results regarding the error (MSE), and the second one provides the result regarding the total increase in the target value (either home price or medication dosage). Each MSE graph shows the mean-squared error for the attacked data, where the target value was changed with two strategies (MAXs and MIDs, which respectively correspond to strategies 0 and 1 in Figures 5.3 and 5.4), explained in the methodology section. The MSE for the clean data after performing both attack strategies is also in the graphs. The target value graph shows the average price or dosage of the attack points after the attack is performed with two attack strategies (MAXs and MIDs) and the average price or dosage of the attacked points before the attack. The poisoning rates vary from 1.2% to 6.4%.

With the housing price dataset, the attack performance is slightly better for the attack with random forest feature selection than with the other two methods. The highest MSE for the attacked point with the random forest is almost 14%, while the MSE for random and correlation selections are around 11%. With other poisoning rates starting from 3%, the random forest also outperforms other methods. From the graphs of the price, we can see the real average price of the attacked points for the random and correlation approaches is somewhere around 0.2, while for the random forest, it is approximately 0.15. So, the real average price of the points from the attacked region determined by random forest is 5% less than in the regions determined by other methods. The difference in the chosen attack region is probably the main reason for the difference in the results shown by these methods. As the starting value was lower, the change with both strategies was larger. Therefore, the error also becomes higher. Nevertheless, the results for a poisoning rate below 3% are very close for all the methods and show that the attack does not lead to a high MSE for the points in a poisoned region. It is worth mentioning that the attack is run on a dataset with 1161 points, so the number of attacked points in a training set varies from 7 as 1% to 41 as 6%.

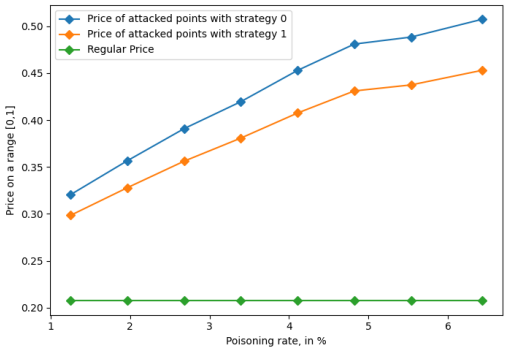
At the same time, the MSE for clean points also becomes higher for the random forest strategy, making the attack easier to detect. This can happen because the high increase in the price of the attacked points shifted the curve too much, and that also made the prices of clean data change. Therefore, in terms of the balance between increasing the attack region error and keeping the error of the clean data low, the random forest strategy does not look better than the other two. If we take the ratio of the error on attacked data to the error on clean data, this ratio will be 2.9 for random strategy, 2.88 for random forest and 3.2 for correlation strategy. These ratios are close, but they bring a controversial outcome: the selection of strategy may vary depending on the primary goal. Depending on whether stealthiness or effectiveness is the main goal, different strategies are preferred.

Another observation is made with regard to the two strategies of selecting the value of the target variable of the attack points: MAXs and MIDs. Their behaviour is almost identical, with the difference being that the curve created by MAXs strategy is a shifted-up version of the curve by MIDs. Calculating the ratio of MAXs error to MIDs error will almost always give the result close to 1.6. The same goes for the clean data error. This is intuitively correct: the change in the predicted value with the MIDs strategy is lower than with MAXs but goes in the same direction. As the change is smaller, the clean data error will also be lower.

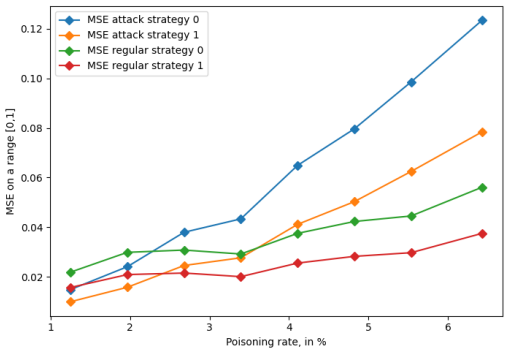
The IWPC dataset has more entries, so more points are attacked. Consequently, the model will have a larger attack region and should learn the behaviour in that region better. The results in Figure 5.4 show that the MSE varies from a maximum of 0.1 for random forest selection to 0.3 for correlation selection. We see a significant difference compared to the error on the clean data, which is never more than 0.02. The



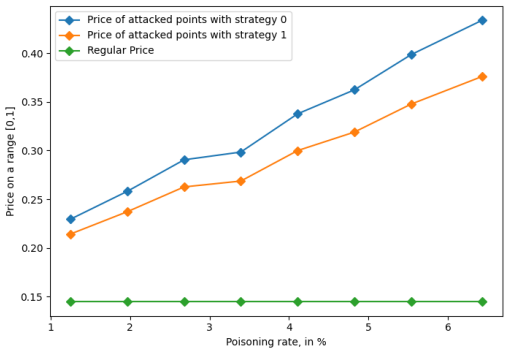
(a) MSE for different poisoning rates for the random selection



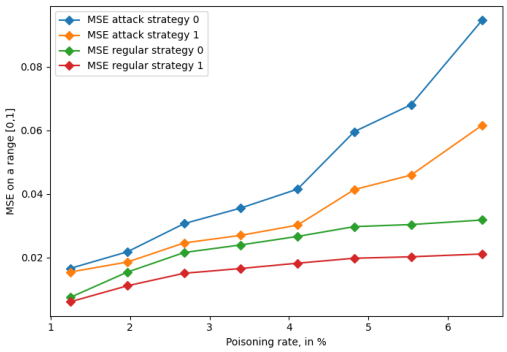
(b) Price for different poisoning rates for the random selection



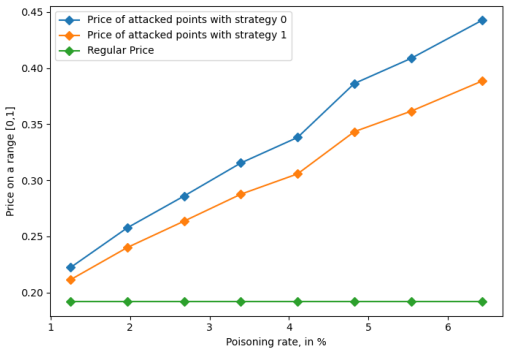
(c) MSE for different poisoning rates for the random forest selection



(d) Price for different poisoning rates for the random forest selection

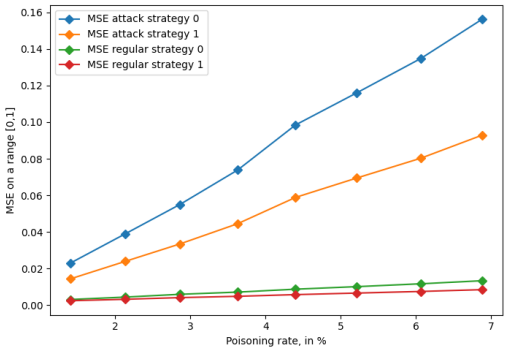


(e) MSE for different poisoning rates for the correlation selection

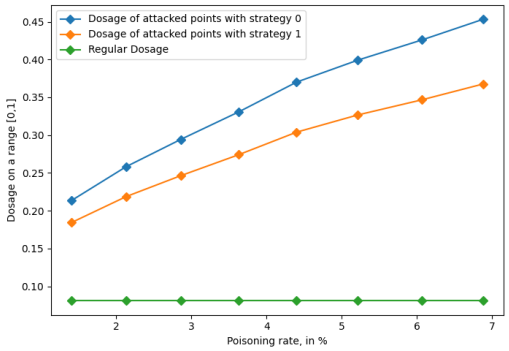


(f) Price for different poisoning rates for the correlation selection

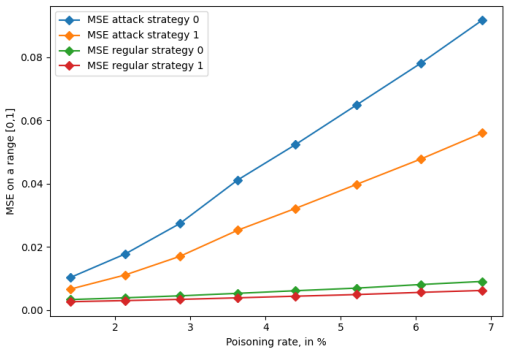
Figure 5.3: Results for housing dataset with $\phi = 0.87$



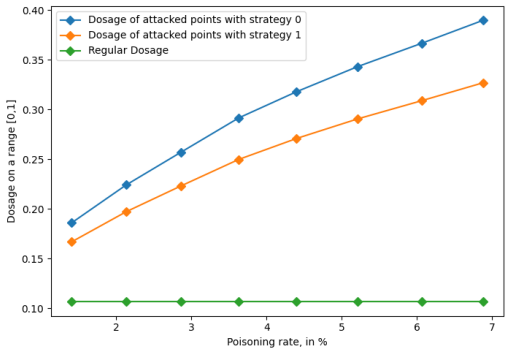
(a) MSE for different poisoning rates for the random selection



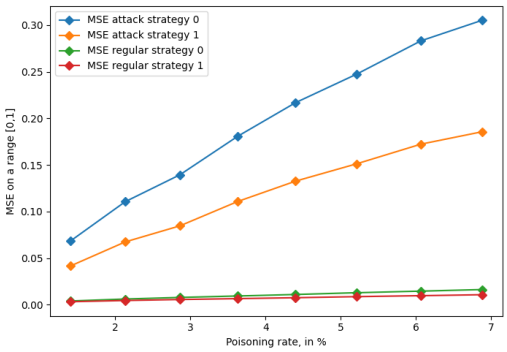
(b) Warfarin dosage for different poisoning rates for the random selection



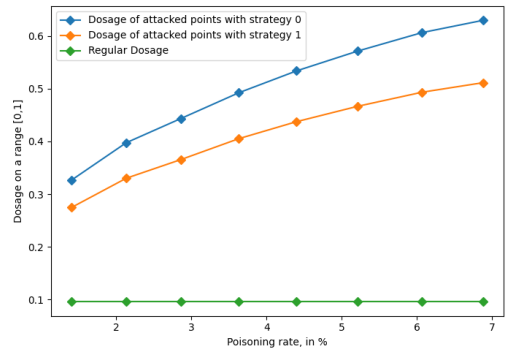
(c) MSE for different poisoning rates for the random forest selection



(d) Warfarin dosage for different poisoning rates for the random forest selection



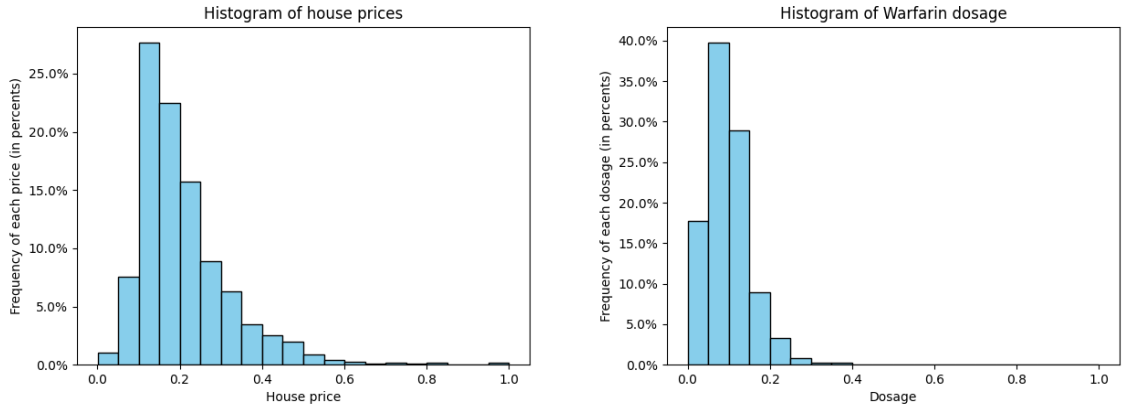
(e) MSE for different poisoning rates for the correlation selection



(f) Warfarin dosage for different poisoning rates for the correlation selection

Figure 5.4: Results for pharmacy dataset with $\phi = 0.87$

increased number of data points explains the overall improved performance. The dataset size has grown four times, so the algorithm has much more training data. Moreover, most of the data has a low target value (below 0.1), as shown in Figure 5.5. While for the housing price dataset, 63% of the target data (before the attack) lies in a region from 0.1 to 0.25 with some points met for all values until 0.6, for the warfarin dosage dataset, 85% of target values belong to the region from 0 to 0.15. As a result, the change in the target value is more significant than in the first dataset for a much bigger percentage of data samples.



(a) The distribution of prices in the house prices dataset **(b)** The distribution of dosages in the Warfarin dosage dataset

Figure 5.5: Comparison of frequencies of the target values of two datasets.

Nevertheless, it does not explain such a huge difference in the results obtained by the correlation strategy and the other two methods. The results of the random forest selection and random selection do not differ that much. That means that the correlation strategy has found a much better trigger. The possible explanation may lie in the selected features. Even though the ranked lists of the most important features are similar for correlation and random forest strategy, they have some differences. There are some features (or a combination of features) that one algorithm selects and the other does not. For example, the gene $VKORC12255consensus : T/T$ is in top-9 for correlation strategy and out of the top 50 for random forest strategy. Having even one important feature when constructing the attack region can decrease the number of features you need to construct a set significantly. As a result, the attack regions become different, and this causes a better performance.

To obtain these results, we used different ϕ for different datasets. We used those values for the parameter because they performed the best out of all our experiments. We tried different values from 0.3 to 0.92. The experiment results are shown in Appendix A. It is interesting to note that for the small dataset, the ϕ value does not play a big role, and the results look somewhat similar, while for larger datasets, the larger ϕ shows more significant results. This can be explained by the fact that the higher the ϕ value becomes, the more features are part of determining the attack region (because fewer data is cut by each feature). For example, when the ϕ is 0.7, on average, algorithms select 20-30 features which determine the attack region. For ϕ, equal to 0.87, 50-60 features are selected. If the dataset is large, getting more features leads to a higher chance of obtaining the most important one.

From these observations, we can make the following brief conclusions:

- All the backdoor attacks are successful. Even though the results vary for different strategies, for the poisoning rates higher than 2%, the MSE error between the predicted value and actual value is higher than the error on clean data. The prices also become much higher, with growths from a factor of 2.86 to 5.7.
- The attack performs much better for large datasets with each strategy and any poisoning rate.
- For a small dataset, there is a high tradeoff between the efficiency and invisibility of the attack.

- The presence of some features when determining the attack region can have a significant effect on the attack's effectiveness.
- Correlation strategy has proven to be the best in terms of both effectiveness and stealthiness.
- There is no difference in the behaviour of the attacks under MAXs and MIDs strategies. One makes a higher change to the target value and, therefore, provides better performance.

5.4. Federated setup results

The next step of the analysis of the results is to look at how the attack works in a federated learning setup. Firstly, we need to introduce some assumptions based on which the setup was created.

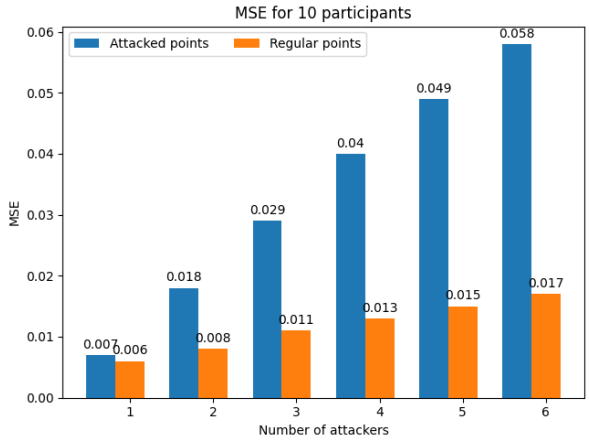
- We use the gradient descent method to find a solution for linear regression. Even though the solution to this model can be obtained by solving the closed-form equation (and used in Federated learning with One-Shot Federated Learning [39]) and avoiding the gradient descent, we need the gradient descent for the defence. For an experiment to be fair (attack and defence use the same approach to solving the regression problem), we do the gradient descent optimisation.
- We assume that in cases when there is more than one malicious user, the attackers know the attack region. In other words, malicious users know the features and their corresponding range of values that are triggered for the backdoor attack to happen. If this was not the case, the attackers would have poisoned the different data regions so that they would have different triggers. This setting is out of the scope of this project.

We ran the experiments 50 times, each time for the number of attackers from 1 to 6 out of 10 nodes in total. The 50 runs are independent, so all the results are reset. When, during a single run, we do the experiments for different numbers of users, the only thing that is changed is the attack points distribution. Moreover, in each run for each user, we do ten different shuffles of the attack data. Following the results of the previous section, the experiments on the housing dataset are done with the random forest feature selection strategy and on the IWPC using the correlation strategy. We ran all the experiments for the settings with 10 participants in total, with a total of 10% of the data being selected as the attack points.

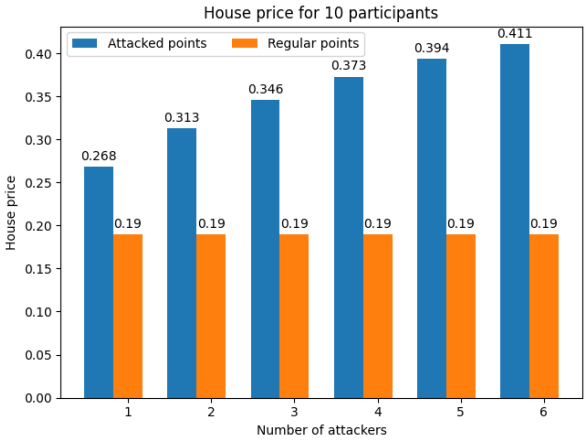
Figure 5.6 shows the results of the backdoor attack on a linear regression model in the federated learning setup. Here, all the figures show the attack results in the setting with the number of attackers from 1 to 6. The attack training data fully goes to the poisoners. For example, if there are three attackers, each receives 33% of the attack training data. The attackers use all of it to train their models. We can see that the attack results have decreased compared to the centralised settings, especially for the house prices dataset. For one attacker, there are barely any differences with the real price, according to MSE. Even for five attackers, which is usually considered the maximum possible amount out of 10, the MSE is 0.05, which is less than four times larger than the MSE for regular points. In terms of the increased price, the picture is better for five malicious users, as it doubled compared to the original price. The experiments on IWPC have shown much better performance, with the MSE on triggered points equal to 0.13 for five attackers. Moreover, the MSE on regular points is less than 0.01 for all the number of attackers except 6. The average predicted warfarin dosage has increased by at least 80% for one attacker and at most 450% for five attackers. The results on this dataset in both centralised and federated versions are better than for the other dataset. The reasons for that are also similar, namely the larger amount of training data and the smaller initial value of the target variable.

Now, we are switching to a case where the attack training data is evenly spread among all the participants, even non-malicious. For example, if there are three attackers, they receive 10% of the attack points each. Other data that belong to the attack region goes to the regular users, so their models can also be trained on the points from the trigger region. We find this setting a bit more fair and realistic. This case is shown in Figure 5.7. It shows a decrease in the results on both datasets. This is expected because attackers now just receive less data than in previous settings. The model ran on the housing price dataset shows poor performance, especially for 1 and 2 attackers. Receiving only ten attack points to train the model and having a model weight of 0.1 plays a role in the results. The attack with the IWPC dataset still performs much better, showing some results for all the number of attackers except 1.

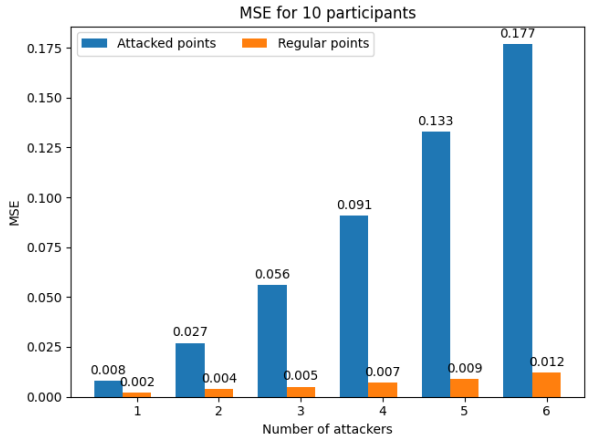
Remembering that the IWPC dataset is almost four times larger, we can conclude that in the federated learning setup, the dataset size and the number of attack points play an even more crucial role than in the centralised setup.



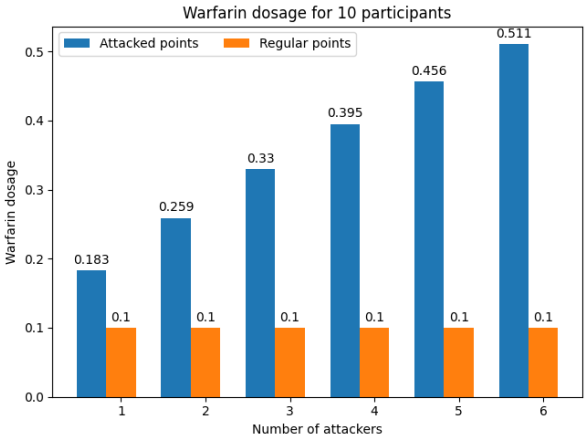
(a) MSE for 10 participants with different numbers of attackers for the house prices dataset



(b) Price for 10 participants with different numbers of attackers for the house prices dataset



(c) MSE for 10 participants with different numbers of attackers for the IWPC dataset



(d) Warfarin dosage for 10 participants with different numbers of attackers for the IWPC dataset

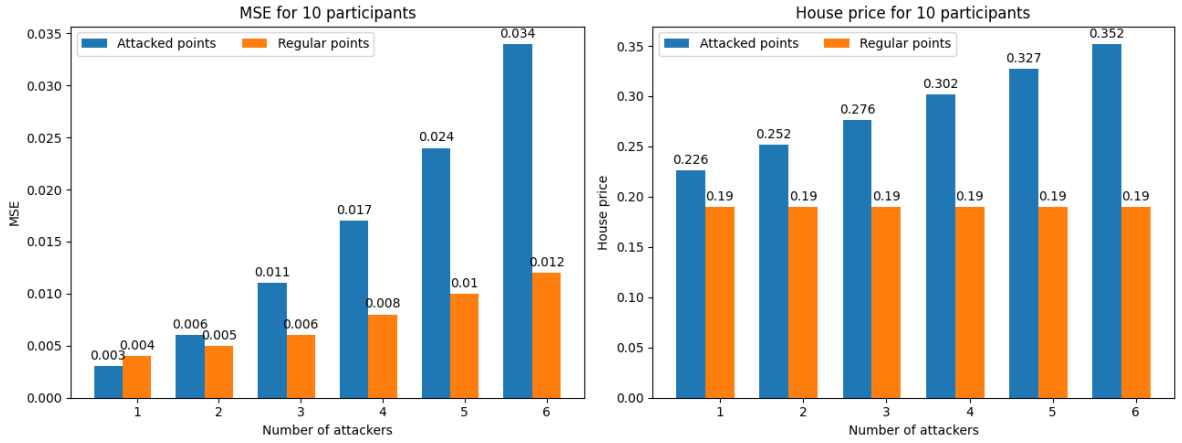
Figure 5.6: Backdoor attack on federated learning with $\phi = 0.87$ and with all the attack data given to attackers

5.5. Defence

In this section, we evaluate the defence results for both the centralised and the federated setups. In both cases, as an attack, we run the version with $\phi = 0.87$ and use a correlation strategy for the feature importance calculation, as it showed a better performance previously.

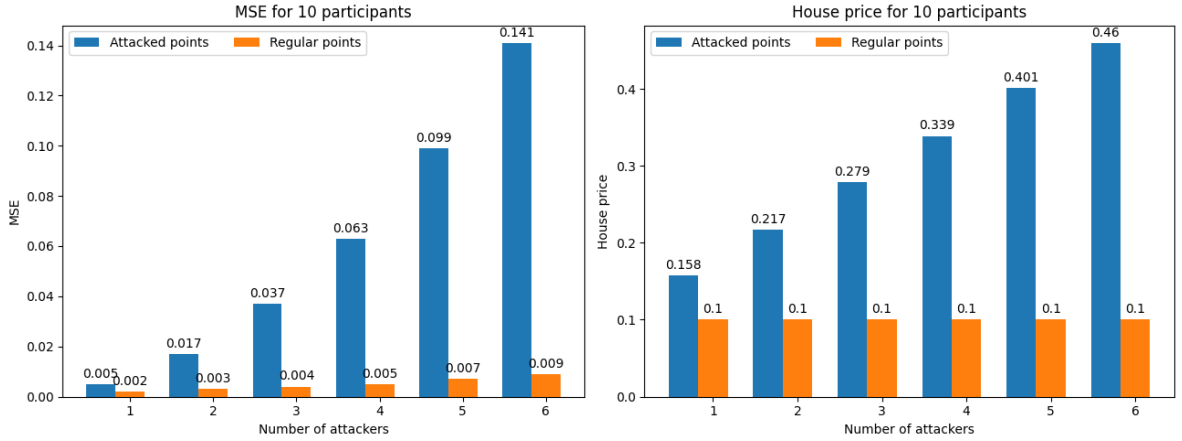
5.5.1. Centralised defence

In the centralised setup, the attacked unit has access to the data. Therefore, the defence strategy used is one that detects the attack points. We use the studentized residuals to detect the outliers in the training data and delete them. We want to emphasise that detecting the outliers (instead of just fitting the line on good points) may be crucial in our settings because the attack is performed on a group of points with very similar features. As a consequence, if we delete all the points that cover some region, our model will have difficulties predicting the values for points from that region. However, this work focuses on detection, not the retrieval of the original values.



(a) MSE for 10 participants with different numbers of attackers for the house prices dataset

(b) Price for 10 participants with different numbers of attackers for the house prices dataset



(c) MSE for 10 participants with different numbers of attackers for the IWPC dataset

(d) Warfarin dosage for 10 participants with different numbers of attackers for the IWPC dataset

Figure 5.7: Backdoor attack on federated learning with $\phi = 0.87$ and with attack data evenly distributed between all the participants

Detecting outliers is the binary classification task: every point is either an outlier or not. Therefore, there are four possible combinations of real results and predicted results:

- True positive (TP): when the attacked point is classified as an outlier
- False positive (FP): when the clear point is classified as an outlier
- True negative (TN): when the clear point is classified as a non-outlier
- False negative (FN): when the attacked point is classified as a non-outlier

The binary classification models ideally should have zero FPs and FNs, which will mean that all the points are classified correctly. This is, however, an ideal and unrealistic scenario. Therefore, the goal becomes to maximise the sum number of TPs and TNs. These two components can differ from task to task. In this project, we are interested in identifying and preventing attacks, so identifying a non-attack as an attack is a less serious mistake than claiming an attack point as a valid one.

Furthermore, in a scenario of a backdoor attack, data is unbalanced, meaning that there are many more clean points than attacked points. As a result, the number of TPs will automatically be smaller compared to TNs. Taking this into account, we decided to use the F1-score as a metric for assessing the quality of the defence model in the centralised setup. First, it calculates two values:

- $Precision = \frac{TP}{TP+FP}$, which measures the accuracy of positive predictions

- $Recall = \frac{TP}{TP+FN}$, which gives insight into how many of the real attacks were identified as such

The F1-score is then calculated with the formula

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall}$$

Table 5.1 presents the results of running the defence in centralised settings. It shows the MSE on training data before and after the outliers are detected, as well as the F1-score, which serves as an estimation of the percentage of the correctly detected outliers. The results are presented for two datasets and for two attack strategies. The correlation strategy was used as a feature selection strategy because it showed better attack results.

As can be seen, the F1-score decreases when the poisoning rate increases: on the IWPC, for the poisoning rates below 3%, the F1-score is higher than 0.9, and for the higher percentage, it gets significantly lower. This can be explained by the fact that with the increase in the poisoning rate, the model gets more attacked points to train, and these points become less outlying. As the model's approach is to statistically detect outliers, the model's habituation to the attacked points makes the attack points seem like regular ones, and the residuals between the predicted and actual target values of the attack points decrease.

Furthermore, the difference between the success of the attacks using the two different strategies is almost invisible: apart from the score for the middle-value strategy attack with the lowest poisoning rate, all the other scores are close. Taking into account the fact that the difference in the target value after the attack on these two scenarios is, on average, 0.27, we can say that the points are enough outlied to be detected, even with a smaller change.

	Strategy with maximal attack			Strategy with middle-value attack		
Poisoning rate	MSE without outlier detection	MSE with outlier detection	F1	MSE without outlier detection	MSE with outlier detection	F1
House price dataset						
1.25%	0.0066	0.002	0.9275	0.0046	0.0015	0.8767
1.96%	0.0095	0.0033	0.972	0.0065	0.0023	0.9369
2.68%	0.012	0.0045	0.966	0.0081	0.0032	0.966
3.39%	0.0146	0.0058	0.973	0.0098	0.004	0.973
4.11%	0.0168	0.0071	0.9498	0.0113	0.0049	0.9455
4.82%	0.0186	0.0086	0.9243	0.0125	0.0059	0.9163
5.54%	0.0208	0.0103	0.8929	0.0139	0.0069	0.9014
6.43%	0.0232	0.0121	0.8715	0.0155	0.0082	0.8652
IWPC dataset						
1.41%	0.0107	0.0037	0.9591	0.0071	0.0027	0.9318
2.14%	0.0137	0.0053	0.9765	0.009	0.0037	0.9689
2.86%	0.0164	0.0069	0.9792	0.0106	0.0048	0.9439
3.63%	0.0188	0.0095	0.7944	0.0121	0.0063	0.7731
4.4%	0.0207	0.0111	0.756	0.0132	0.0072	0.7512
5.21%	0.0226	0.0127	0.7601	0.0144	0.0082	0.7538
6.07%	0.0242	0.0142	0.7423	0.0153	0.0092	0.7274
6.88%	0.0254	0.0157	0.7019	0.016	0.0101	0.6908

Table 5.1: F1-score for the studentized residual defence for two datasets

5.5.2. Federated defence

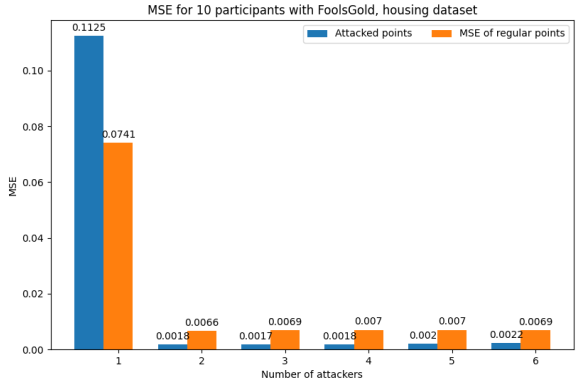
In the federated learning setup, the server does not have access to the learning data, so detecting the outliers there will not work. Here, we only receive the model coefficients from the users. Therefore, we have used a variation of the FoolsGold [71] algorithm as a defence method. Previously, we fit the model based on data using the exact solution of the linear regression. However, the defence algorithm now uses the gradient descent methodology to find the optimal weights for different clients' models. We changed our model to use gradient descent to find the model's coefficients. We used the learning rate of 0.001 and 1000 iterations to get each server model. The code was run for a total of 10 users, with the number of attackers varying from 1 to 6. Each result is an average of 30 runs.

Figure 5.8 presents the results of the defence against the backdoor attack on the federated learning setup. The results are shown in the form of the MSE error for the varying number of attackers. 10% of all data is poisoned and split between attackers or between all the users. As can be seen, the defence

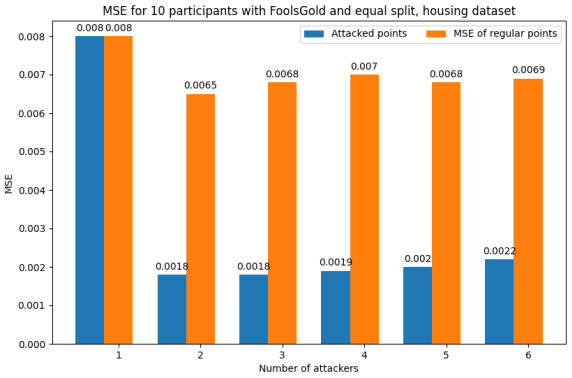
works, and the model becomes not poisoned for all the scenarios except when there is only one attacker.

Failure for one attacker. Figure 5.8 shows that when there is only one malicious user, the defence does not work and, moreover, makes the attack stronger. One thing we can observe directly from the graphs is that the effect in a setting with the data evenly distributed between all the clients is much less serious than when the data is distributed between only attackers. As in the failure scenario, there is only one unscrupulous user, and the whole attack dataset goes only to that user, making a) the training dataset of the attacker much larger than the datasets of other people and b) giving that user the ability to fully train the model's behaviour on the attack region. This leads to a very different delta (compared to other users), which comes to the server and to the defence mechanism. The FoolsGold algorithm first computes the pairwise cosine similarity between the deltas of all the clients [71]. The delta of the attacker is very different, so the similarities with the deltas of all other users will be low. Further, the algorithm selects the maximum similarity for each user, which is also much lower for the attacker than for everyone else. However, after this step, the algorithm flips these values (as similarity belongs to the range from 0 to 1, the flip is the subtraction of the similarity from 1) so that the lowest value becomes the highest. These flipped values are becoming the new clients' learning rate. The flip is done to consider the case when there is only one fair user, and its weight in the total model remains valuable. However, it makes the case with a very different attacker a problem. The problem does not arise with two malicious users because the similarity between them is high, so the flip does not make them dominate the model.

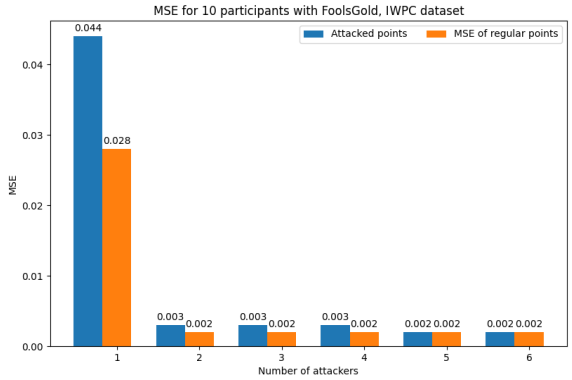
For all the other numbers of malicious users, the MSE for the points belonging to the same region as the attacked points is almost identical to the MSE of all the other points and is always less than 0.01. In other words, the average squared difference between the actual value and the prediction of that value is less than 1%. That shows that in most cases, FoolsGold is able to prevent our backdoor attack on regression in the federated learning setup.



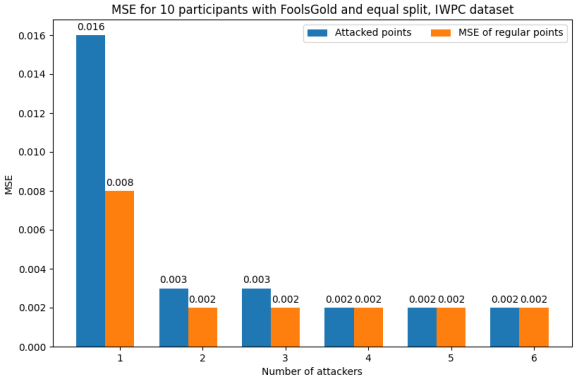
(a) MSE after defence for 10 participants with different numbers of attackers for the house prices dataset and attack data distributed only between attackers



(b) MSE after defence for 10 participants with different numbers of attackers for the house prices dataset and attack data distributed between all the participants



(c) MSE after defence for 10 participants with different numbers of attackers for the IWPC dataset and attack data distributed only between attackers



(d) MSE after defence for 10 participants with different numbers of attackers for the IWPC dataset and attack data distributed between all the participants

Figure 5.8: Results of the defence against the backdoor attack on federated learning with $\phi = 0.87$

6 Conclusions

The main goal of this thesis was to investigate the backdoor attacks on regression problems, as this field is not yet covered. This chapter answers the research questions (by answering the subquestions) and presents the known limitations.

6.1. Research questions revisited

SQ1: What can be the effective trigger in the backdoor attack on regression?

We determined that in the numeric data, the feature selection algorithms can be used as the tool to create triggers. As they calculate the effect that different features have on the target value, the most influenceable features create better regions for attack in terms of attack effectiveness, attack stealthiness and universality. We used Correlation feature selection and Random forest feature selection algorithms to rank all the features in the dataset. When run on two experimental datasets, the ranked lists created by the two algorithms have proven to be similar according to rank-biased overlap. Moreover, they top-rank the features which are the most important in reality.

SQ2: Can we design a backdoor attack on the regression model?

We proposed Algorithm 2 as the basic algorithm for constructing the backdoor attack on the regression model. It first creates a rank of features based on the results of the feature selection algorithm and then changes the target value of the attacked points according to either the MAXs or MIDs strategy. We tested the algorithm on datasets of different sizes and have seen that the performance is much better on the larger dataset. We have shown that for the poisoning rates higher than 2%, all the attacks increase the MSE to be higher for attacked points than for the regular points. The correlation feature selection strategy has shown the best performance in most experiments. However, we have shown that sometimes, the best strategy may depend on the primary goal of a backdoor attack (whether we want it to be more stealthy or more effective in terms of the MSE). Moreover, the attacks can be sensitive to the features and their selected regions when forming the attack region.

SQ3: Can we use outlier detection to defend against backdoor attacks on the regression model?

Using the studentized residuals, we have constructed the outlier detection algorithms and used them for the detection of attacked points in the training dataset. The defence has improved significantly the performance of the model when compared to the scenario when there is no defence. With the increasing poisoning rate, the performance of the defence worsened as the poisoned data became more common, and it became harder to call such data an outlier. The performance of defence when standing against MAXs is much better than against MIDs. This shows the importance of the second strategy when performing the attack: even though it has lower performance, it is harder to defend against.

SQ4: Can we transfer the created backdoor attack on regression to a federated learning setup?

Using the same data poisoning method, training the local linear regression model with the poisoned data and sending the poisoned model to the server have shown to work for the backdoor attack on regression. We tried the attack with two different methods of distribution of the attack data between the users. An extreme approach where all the data from the attack region goes only to attackers has shown a better performance. On a larger dataset, the clean data error was (almost) always lower than 0.01, while the error of the attack region has been up to 0.17. When some data from the attack region also goes to fair users, the performance drops down. However, even in these settings, the attack works, and the predicted target value of the points belonging to the attack region has a significant increase.

> *SQ5: Do the existing solutions against the backdoor attack for federated learning work for the regression?*

We have implemented the FoolsGold [71], which was created for the classification problems, for our backdoor attack on regression. It worked and prevented the attack for all the cases except the one with a single attacker.

6.2. Limitations and future works

Even though we fulfilled the research questions and constructed the working backdoor attacks on regression and defences against them in both centralised and federated setups, there are still points of improvement.

First, we only tested the methods on two datasets that are of significantly different sizes. That affected the final results, making the performance of the attack on the larger dataset much better. Testing it on other small datasets and adapting the attacks and defences for them specifically can be an extension of the work.

Further, we worked under the assumption that we cannot insert new data into the dataset or modify the x-values. If someone does not follow this assumption, a much more sophisticated trigger can be constructed. Other attack variations from the classification settings can be tried to improve the attack performance.

Additionally, the creation of the trigger can take any form. In this paper, we used only random forest and correlation feature selection strategies, although there are more of those. Some of them may become a better choice for the backdoor attack on regression. Moreover, completely different ways of selecting data for the trigger can improve the attack efficiency. Lastly, the triggers are now easy to find if someone knows what to search. Improving the trigger's invisibility can be a promising extension to this work.

More sophisticated machine learning models can also be used for backdoor attacks. We used Ordinary Least Squares to calculate the parameters of a linear regression model. It can suffer from high variance and overfitting if there is too much correlation between the predictor variables. Therefore, one can explore the possibility of running and adapting this attack for Ridge, Lasso and other regression techniques.

In this work, we used only outlier detection as a defence against the backdoor attack. There are numerous other techniques that can achieve a better performance and give a higher degree of defence. The same goes for the defence in the federated learning setup. The main goal of this thesis was to show the possibility of doing the backdoor attack on regression. Modifying the existing backdoor attacks on classification problems and using them for regression or creating totally new attacks is an open prospect.

Bibliography

[1] F. Galton, "Regression towards mediocrity in hereditary stature.," *The Journal of the Anthropological Institute of Great Britain and Ireland*, vol. 15, pp. 246–263, 1886, ISSN: 09595295. [Online]. Available: http://www.jstor.org/stable/2841583 (visited on 10/15/2023).

[2] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. [Online]. Available: https://faculty.marshall.usc.edu/gareth-james/ISL/.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, *Communication-efficient learning of deep networks from decentralized data*, 2023. arXiv: 1602.05629 [cs.LG].

[4] T. Gu, B. Dolan-Gavitt, and S. Garg, *Badnets: Identifying vulnerabilities in the machine learning model supply chain*, 2019. arXiv: 1708.06733 [cs.CR].

[5] A. Nguyen and A. Tran, *Wanet – imperceptible warping-based backdoor attack*, 2021. arXiv: 2102.10369 [cs.CR].

[6] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, *Invisible backdoor attack with sample-specific triggers*, 2021. arXiv: 2012.03816 [cs.CR].

[7] Y. Ren, L. Li, and J. Zhou, "Simtrojan: Stealthy backdoor attack," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 819–823. DOI: 10.1109/ICIP42928.2021.9506313.

[8] M. Xue, S. Ni, Y. Wu, Y. Zhang, J. Wang, and W. Liu, *Imperceptible and multi-channel backdoor attack against deep neural networks*, 2022. arXiv: 2201.13164 [cs.CV].

[9] M. Xue, C. He, Y. Wu, *et al.*, "Ptb: Robust physical backdoor attacks against deep neural networks in real world," *Computers & Security*, vol. 118, p. 102 726, 2022, ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2022.102726. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404822001213.

[10] M. Barni, K. Kallas, and B. Tondi, *A new backdoor attack in cnns by training set corruption without label poisoning*, 2019. arXiv: 1902.11237 [cs.CR].

[11] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001, ISBN: 0262194759.

[12] R. Duda, P. Hart, and D. G.Stork, "Pattern classification," in Jan. 2001, vol. xx, ISBN: 0-471-05669-3.

[13] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied Linear Statistical Models*. Chicago: Irwin, 1996.

[14] T. D. Nguyen, P. Rieger, H. Chen, *et al.*, *Flame: Taming backdoors in federated learning (extended version 1)*, 2023. arXiv: 2101.02281 [cs.CR].

[15] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, *How to backdoor federated learning*, 2019. arXiv: 1807.00459 [cs.CR].

[16] Z. Chen, Y. Luo, S. Wang, J. Li, and Z. Huang, *Federated zero-shot learning for visual recognition*, 2022. arXiv: 2209.01994 [cs.CV].

[17] T. Liu, X. Hu, and T. Shu, *Technical report: Assisting backdoor federated learning with whole population knowledge alignment*, 2022. arXiv: 2207.12327 [cs.AI].

[18] K. Sharifani, M. Amini, Y. Akbari, and J. Godarzi, "Operating machine learning across natural language processing techniques for improvement of fabricated news model," vol. 12, pp. 20–44, Oct. 2022.

[19] B. Balsmeier, M. Assaf, T. Chesebro, *et al.*, "Machine learning and natural language processing on the patent corpus: Data, tools, and new measures," *Journal of Economics & Management Strategy*, vol. 27, no. 3, pp. 535–553, 2018. DOI: https://doi.org/10.1111/jems.12259. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/jems.12259.

[20] I. Castiglioni, L. Rundo, M. Codari, *et al.*, "Ai applications to medical images: From machine learning to deep learning," *Physica Medica*, vol. 83, pp. 9–24, 2021, ISSN: 1120-1797. DOI: https://doi.org/10.1016/j.ejmp.2021.02.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1120179721000946.

[21] S. Saladi, Y. Karuna, S. Koppu, *et al.*, "Segmentation and analysis emphasizing neonatal mri brain images using machine learning techniques," *Mathematics*, vol. 11, no. 2, 2023, ISSN: 2227-7390. DOI: 10.3390/math11020285. [Online]. Available: https://www.mdpi.com/2227-7390/11/2/285.

[22] F. E. Botchey, Z. Qin, and K. Hughes-Lartey, "Mobile money fraud prediction—a cross-case analysis on the efficiency of support vector machines, gradient boosted decision trees, and naïve bayes algorithms," *Information*, vol. 11, no. 8, 2020, ISSN: 2078-2489. DOI: 10.3390/info11080383. [Online]. Available: https://www.mdpi.com/2078-2489/11/8/383.

[23] Z. Yi, X. Cao, X. Pu, *et al.*, "Fraud detection in capital markets: A novel machine learning approach," *Expert Systems with Applications*, vol. 231, p. 120 760, 2023, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.120760. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417423012629.

[24] H. H. van Engelen J.E., "A survey on semi-supervised learning," *Machine Learning*, vol. 109, pp. 373–440, 2020. DOI: https://doi.org/10.1007/s10994-019-05855-6.

[25] B. Mahesh, *Machine learning algorithms -a review*, Jan. 2019. DOI: 10.21275/ART20203995.

[26] J. Wen, B. Z. H. Zhao, M. Xue, A. Oprea, and H. Qian, *With great dispersion comes greater resilience: Efficient poisoning attacks and defenses for linear regression models*, 2021. arXiv: 2006.11928 [cs.CR].

[27] N. N. Dalvi, P. M. Domingos, Mausam, S. K. Sanghai, and D. Verma, "Adversarial classification," *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[28] D. Lowd and C. Meek, "Good word attacks on statistical spam filters," in *International Conference on Email and Anti-Spam*, 2005.

[29] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proceedings of the Asian Conference on Machine Learning*, C.-N. Hsu and W. S. Lee, Eds., ser. Proceedings of Machine Learning Research, vol. 20, South Garden Hotels and Resorts, Taoyuan, Taiwain: PMLR, Nov. 2011, pp. 97–112. [Online]. Available: https://proceedings.mlr.press/v20/biggio11.html.

[30] B. Biggio, B. Nelson, and P. Laskov, *Poisoning attacks against support vector machines*, 2013. arXiv: 1206.6389 [cs.LG].

[31] A. E. Cinà, K. Grosse, A. Demontis, *et al.*, "Wild patterns reloaded: A survey of machine learning security against training data poisoning," *ACM Computing Surveys*, Mar. 2023. DOI: 10.1145/3585385. [Online]. Available: https://doi.org/10.1145%2F3585385.

[32] E. Erdogan, A. Küpçü, and A. E. Cicek, "Splitguard: Detecting and mitigating training-hijacking attacks in split learning," in *Proceedings of the 21st Workshop on Privacy in the Electronic Society*, ACM, Nov. 2022. DOI: 10.1145/3559613.3563198. [Online]. Available: https://doi.org/10.1145%2F3559613.3563198.

[33] L. Muñoz-González, B. Biggio, A. Demontis, *et al.*, *Towards poisoning of deep learning algorithms with back-gradient optimization*, 2017. arXiv: 1708.08689 [cs.LG].

[34] W. Guo, L. Wang, X. Xing, M. Du, and D. Song, *Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems*, 2019. arXiv: 1908.01763 [cs.CR].

[35] E. Wenger, J. Passananti, A. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, *Backdoor attacks against deep learning systems in the physical world*, 2021. arXiv: 2006.14580 [cs.CV].

[36] J. Ye, X. Liu, Z. You, G. Li, and B. Liu, "Drinet: Dynamic backdoor attack against automatic speech recognization models," *Applied Sciences*, vol. 12, no. 12, 2022, ISSN: 2076-3417. DOI: 10.3390/app12125786. [Online]. Available: https://www.mdpi.com/2076-3417/12/12/5786.

[37] A. K. R. Nadikattu, "Iot and the issue of data privacy," *International Journal of Innovations in Engineering Research and Technology*, vol. 5, no. 10, pp. 23–26, 2018.

[38] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106 854, 2020, ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2020.106854. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835220305532.

[39] N. Guha, A. Talwalkar, and V. Smith, *One-shot federated learning*, 2019. arXiv: 1902.11175 [cs.LG].

[40] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, Jan. 2019, ISSN: 2157-6904. DOI: 10.1145/3298981. [Online]. Available: https://doi.org/10.1145/3298981.

[41] W.-Y. Loh, "Classification and regression trees," *WIREs Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 14–23, 2011. DOI: https://doi.org/10.1002/widm.8. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.8.

[42] Z. Xie, Y. Chen, D. Lu, G. Li, and E. Chen, "Classification of land cover, forest, and tree species classes with ziyuan-3 multispectral and stereo data," *Remote Sensing*, vol. 11, no. 2, 2019, ISSN: 2072-4292. DOI: 10.3390/rs11020164. [Online]. Available: https://www.mdpi.com/2072-4292/11/2/164.

[43] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, *Targeted backdoor attacks on deep learning systems using data poisoning*, 2017. arXiv: 1712.05526 [cs.CR].

[44] S. Koffas, J. Xu, M. Conti, and S. Picek, "Can you hear it? backdoor attacks via ultrasonic triggers," in *Proceedings of the 2022 ACM Workshop on Wireless Security and Machine Learning*, ser. WiseML '22, San Antonio, TX, USA: Association for Computing Machinery, 2022, pp. 57–62, ISBN: 9781450392778. DOI: 10.1145/3522783.3529523. [Online]. Available: https://doi.org/10.1145/3522783.3529523.

[45] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," 2018.

[46] A. Turner, D. Tsipras, and A. Madry, *Label-consistent backdoor attacks*, 2019. arXiv: 1912.02771 [stat.ML].

[47] E. Bagdasaryan and V. Shmatikov, *Blind backdoors in deep learning models*, 2021. arXiv: 2005.03823 [cs.CR].

[48] J. Dai and Z. Xiong, *A semantic backdoor attack against graph convolutional networks*, 2023. arXiv: 2302.14353 [cs.LG].

[49] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 11 946–11 956. DOI: 10.1109/ICCV48922.2021.01175.

[50] W. Guo, B. Tondi, and M. Barni, "A master key backdoor for universal impersonation attack against dnn-based face verification," *Pattern Recognition Letters*, vol. 144, pp. 61–67, 2021, ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2021.01.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865521000210.

[51] H. Li, Y. Wang, X. Xie, *et al.*, *Light can hack your face! black-box backdoor attack on face recognition systems*, 2020. arXiv: 2009.06996 [cs.CR].

[52] S. Zhao, X. Ma, X. Zheng, J. Bailey, J. Chen, and Y.-G. Jiang, *Clean-label backdoor attacks on video recognition models*, 2020. arXiv: 2003.03030 [cs.CV].

[53] Y. Li, T. Zhai, Y. Jiang, Z. Li, and S.-T. Xia, *Backdoor attack in the physical world*, 2021. arXiv: 2104.02361 [cs.CR].

[54] M. Nwadike, T. Miyawaki, E. Sarkar, M. Maniatakos, and F. Shamout, *Explainability matters: Backdoor attacks on medical imaging*, 2020. arXiv: 2101.00008 [cs.CR].

[55] Z. Ye, T. Mao, L. Dong, and D. Yan, "Fake the real: Backdoor attack on deep speech classification via voice conversion," in *INTERSPEECH 2023*, ISCA, Aug. 2023. DOI: 10.21437/interspeech.2023-733. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2023-733.

[56] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, *Manipulating machine learning: Poisoning attacks and countermeasures for regression learning*, 2021. arXiv: 1804.00308 [cs.CR].

[57] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. B. Calo, "Analyzing federated learning through an adversarial lens," *CoRR*, vol. abs/1811.12470, 2018. arXiv: 1811.12470. [Online]. Available: http://arxiv.org/abs/1811.12470.

[58] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=rkgyS0VFvr.

[59] T. Li, S. Hu, A. Beirami, and V. Smith, *Ditto: Fair and robust federated learning through personalization*, 2021. arXiv: 2012.04221 [cs.LG].

[60] A. Huang, *Dynamic backdoor attacks against federated learning*, 2020. arXiv: 2011.07429 [cs.LG].

[61] B. Wang, Y. Yao, S. Shan, *et al.*, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 707–723. DOI: 10.1109/SP.2019.00031.

[62] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Annual Computer Security Applications Conference*, ser. ACSAC '20, Austin, USA: Association for Computing Machinery, 2020, pp. 897–912, ISBN: 9781450388580. DOI: 10.1145/3427228.3427264. [Online]. Available: https://doi.org/10.1145/3427228.3427264.

[63] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, International Joint Conferences on Artificial Intelligence Organization, Jul. 2019, pp. 4658–4664. DOI: 10.24963/ijcai.2019/647. [Online]. Available: https://doi.org/10.24963/ijcai.2019/647.

[64] Z. Xiang, D. J. Miller, and G. Kesidis, *Detection of backdoors in trained classifiers without access to the training set*, 2020. arXiv: 1908.10498 [cs.LG].

[65] B. Tran, J. Li, and A. Madry, *Spectral signatures in backdoor attacks*, 2018. arXiv: 1811.00636 [cs.LG].

[66] A. Chan and Y.-S. Ong, *Poison as a cure: Detecting & neutralizing variable-sized backdoor attacks in deep neural networks*, 2019. arXiv: 1911.08040 [cs.CV].

[67] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: Defending against backdoor attacks using robust statistics," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, Jul. 2021, pp. 4129–4139. [Online]. Available: https://proceedings.mlr.press/v139/hayase21a.html.

[68] Y. Gao, B. G. Doan, Z. Zhang, *et al.*, *Backdoor attacks and countermeasures on deep learning: A comprehensive review*, 2020. arXiv: 2007.10760 [cs.CR].

[69] X. Gao and M. Qiu, "Energy-based learning for polluted outlier detection in backdoor," in *2022 IEEE 7th International Conference on Smart Cloud (SmartCloud)*, 2022, pp. 47–52. DOI: 10.1109/SmartCloud55982.2022.00014.

[70] P. J. Rousseeuw and M. Hubert, "Robust statistics for outlier detection," *WIREs Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 73–79, 2011. DOI: https://doi.org/10.1002/widm.2. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.2. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.2.

[71] C. Fung, C. J. M. Yoon, and I. Beschastnikh, *Mitigating sybils in federated learning poisoning*, 2020. arXiv: 1808.04866 [cs.LG].

[72] S. Awan, B. Luo, and F. Li, "Contra: Defending against poisoning attacks in federated learning," in *Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26*, Springer, 2021, pp. 455–475.

[73] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, *Can you really backdoor federated learning?* 2019. arXiv: 1911.07963 [cs.LG].

[74] M. S. Ozdayi, M. Kantarcioglu, and Y. R. Gel, "Defending against backdoors in federated learning with robust learning rate," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, pp. 9268–9276, May 2021. DOI: 10.1609/aaai.v35i10.17118. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17118.

[75] J. K. Jaiswal and R. Samikannu, "Application of random forest algorithm on feature subset selection and classification and regression," in *2017 World Congress on Computing and Communication Technologies (WCCCT)*, 2017, pp. 65–68. DOI: 10.1109/WCCCT.2016.25.

[76] M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 359–366, ISBN: 1558607072.

[77] *Studentized residual test*, Aug. 2023. [Online]. Available: https://www.ibm.com/docs/en/SSEP7J_12.0.0/com.ibm.swg.ba.cognos.ug_ca_dshb.doc/studentized_residual_test.html.

[78] A. Montoya. "House prices - advanced regression techniques." (2016), [Online]. Available: https://kaggle.com/competitions/house-prices-advanced-regression-techniques (visited on 06/14/2023).

[79] PharmGKB, 2009. [Online]. Available: https://www.pharmgkb.org/downloads (visited on 06/14/2023).

[80] "Estimation of the warfarin dose with clinical and pharmacogenetic data," *New England Journal of Medicine*, vol. 360, no. 8, pp. 753–764, 2009, PMID: 19228618. DOI: 10.1056/NEJMoa0809329. [Online]. Available: https://doi.org/10.1056/NEJMoa0809329.

[81] D. Levy, L. Murphy, and C. K. Lee, "Influences and emotions: Exploring family decision-making processes when buying a house," *Housing Studies*, vol. 23, no. 2, pp. 271–289, 2008. DOI: 10.1080/02673030801893164. [Online]. Available: https://doi.org/10.1080/02673030801893164.

[82] H. Pawar, *The factors that people consider when buying a home - survey*, Apr. 2023. [Online]. Available: https://goelgangadevelopments.com/home-buying-tips/the-factors-that-people-consider-when-buying-a-home-survey/.

[83] W. Webber, A. Moffat, and J. Zobel, "A similarity measure for indefinite rankings," *ACM Trans. Inf. Syst.*, vol. 28, no. 4, Nov. 2010, ISSN: 1046-8188. DOI: 10.1145/1852102.1852106. [Online]. Available: https://doi.org/10.1145/1852102.1852106.

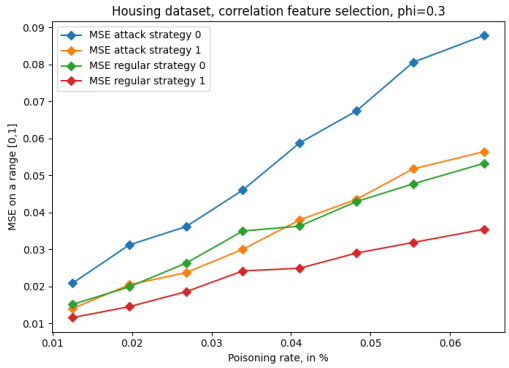
[84] A. Sarica, A. Quattrone, and A. Quattrone, "Introducing the rank-biased overlap as similarity measure for feature importance in explainable machine learning: A case study on parkinson's disease," in *Brain Informatics*, M. Mahmud, J. He, S. Vassanelli, A. van Zundert, and N. Zhong, Eds., Cham: Springer International Publishing, 2022, pp. 129–139, ISBN: 978-3-031-15037-1.

[85] S.-H. Yoo, S. U. Kwon, M.-W. Jo, D.-W. Kang, and J. S. Kim, "Age- and weight-adjusted warfarin initiation nomogram for ischaemic stroke patients," *European Journal of Neurology*, vol. 19, no. 12, pp. 1547–1553, 2012. DOI: https://doi.org/10.1111/j.1468-1331.2012.03772.x. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-1331.2012.03772.x.

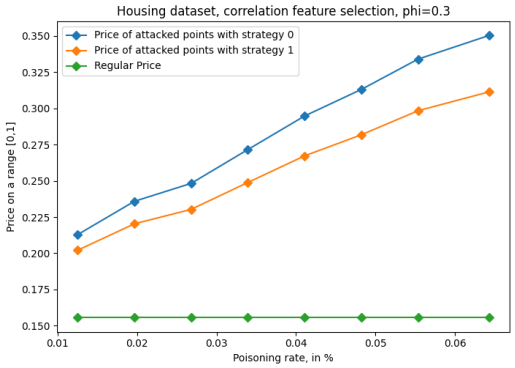
A

Additional experiments for different ϕ

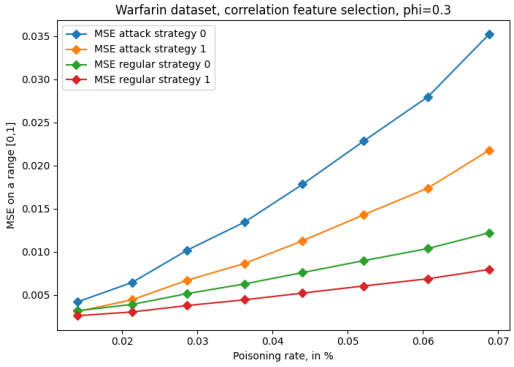
The appendix presents the additional experiments that were run using the correlation feature selection and different values of ϕ. We show that the higher values of ϕ lead to the better results.



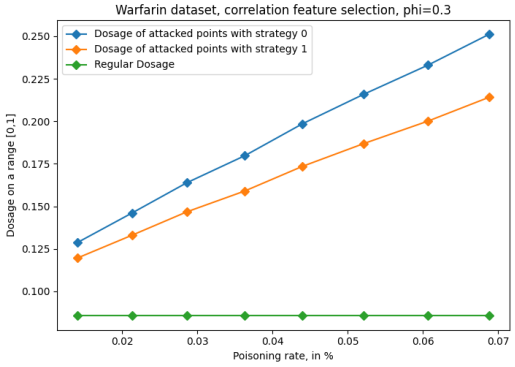
(a) MSE for $\phi = 0.3$ with correlation feature selection and housing prices dataset



(b) Price for $\phi = 0.3$ with correlation feature selection and housing prices dataset

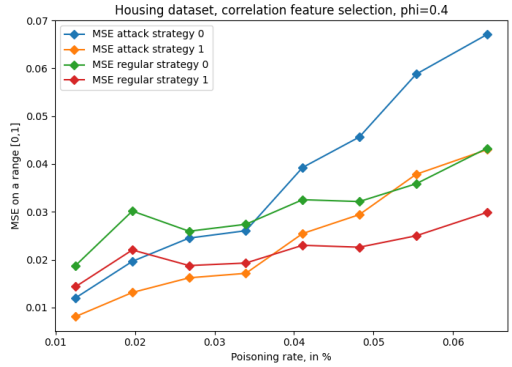


(c) MSE for $\phi = 0.3$ with correlation feature selection and IWPC dataset

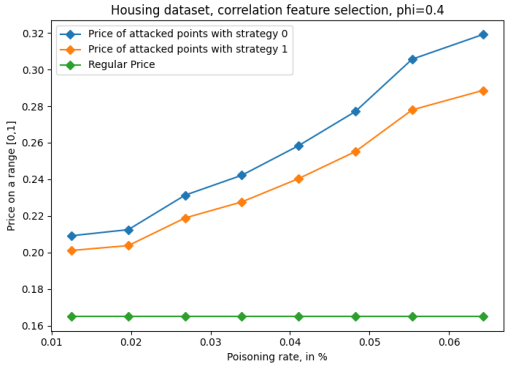


(d) Warfarin dosage for $\phi = 0.3$ with correlation feature selection and IWPC dataset

Figure A.1: Backdoor attack on linear regression model and $\phi = 0.3$

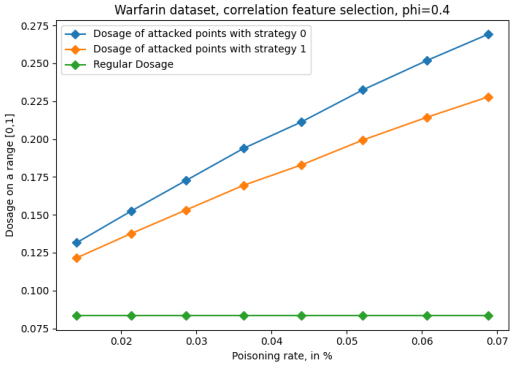


(a) MSE for $\phi = 0.4$ with correlation feature selection and housing prices dataset



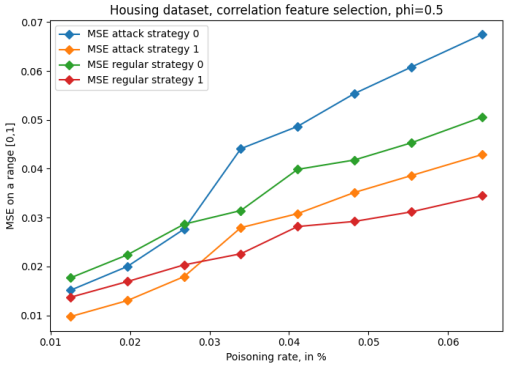
(b) Price for $\phi = 0.4$ with correlation feature selection and housing prices dataset

(c) MSE for $\phi = 0.4$ with correlation feature selection and IWPC dataset

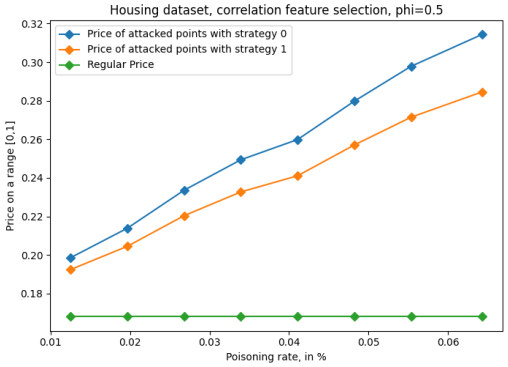


(d) Warfarin dosage for $\phi = 0.4$ with correlation feature selection and IWPC dataset

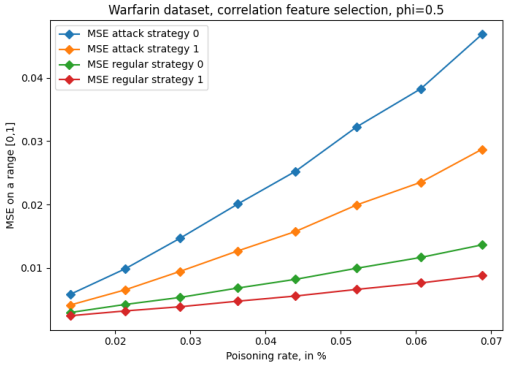
Figure A.2: Backdoor attack on linear regression model and $\phi = 0.4$



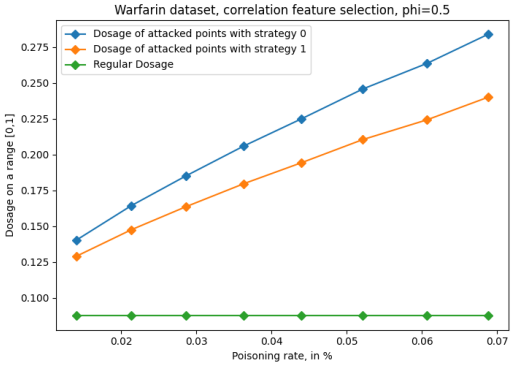
(a) MSE for $\phi = 0.5$ with correlation feature selection and housing prices dataset



(b) Price for $\phi = 0.5$ with correlation feature selection and housing prices dataset

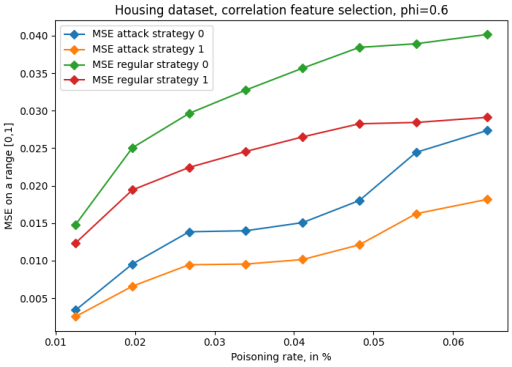


(c) MSE for $\phi = 0.5$ with correlation feature selection and IWPC dataset

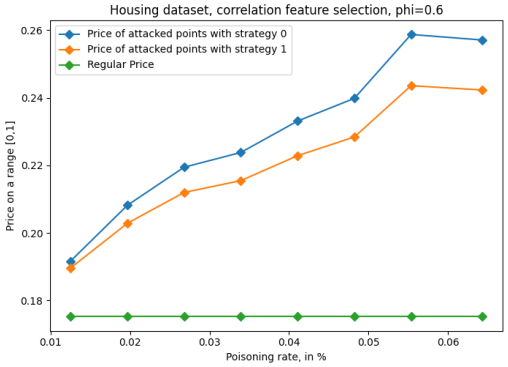


(d) Warfarin dosage for $\phi = 0.5$ with correlation feature selection and IWPC dataset

Figure A.3: Backdoor attack on linear regression model and $\phi = 0.5$

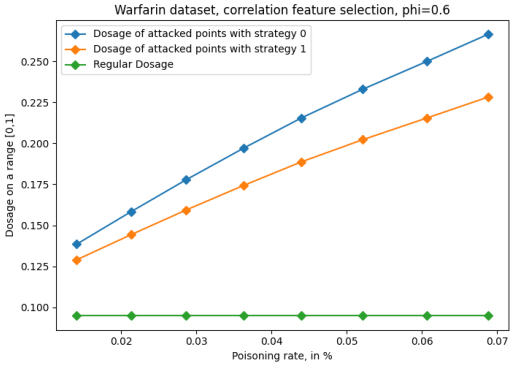


(a) MSE for $\phi = 0.6$ with correlation feature selection and housing prices dataset



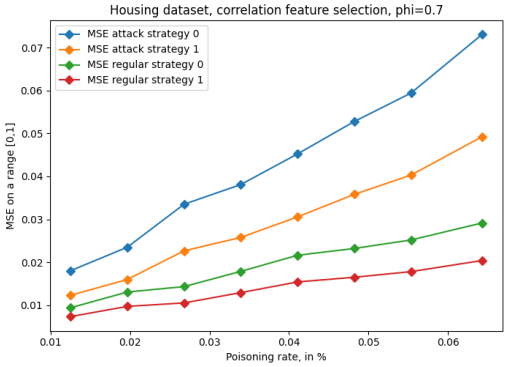
(b) Price for $\phi = 0.6$ with correlation feature selection and housing prices dataset

(c) MSE for $\phi = 0.6$ with correlation feature selection and IWPC dataset

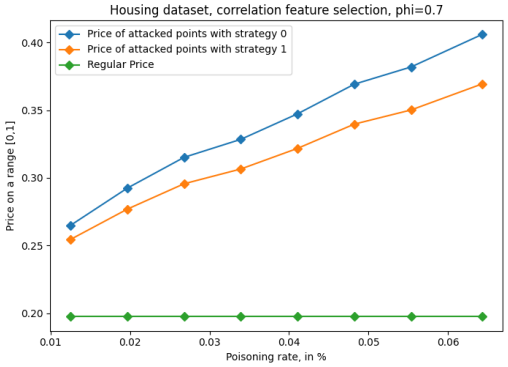


(d) Warfarin dosage for $\phi = 0.6$ with correlation feature selection and IWPC dataset

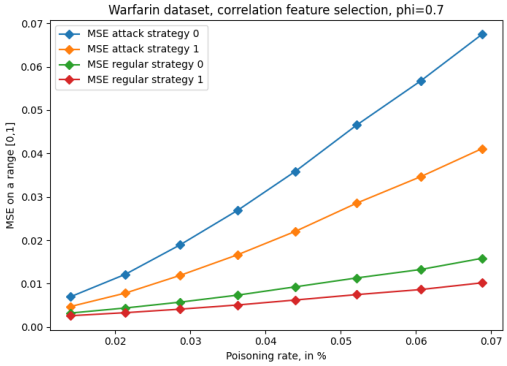
Figure A.4: Backdoor attack on linear regression model and $\phi = 0.6$



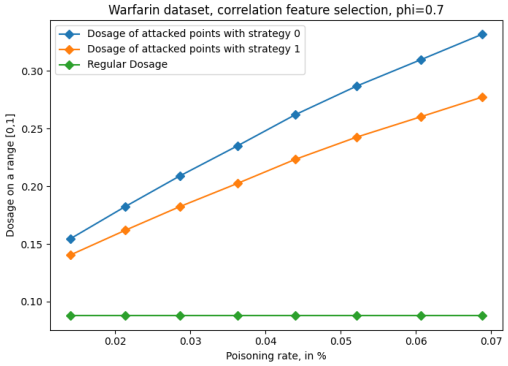
(a) MSE for $\phi = 0.7$ with correlation feature selection and housing prices dataset



(b) Price for $\phi = 0.7$ with correlation feature selection and housing prices dataset

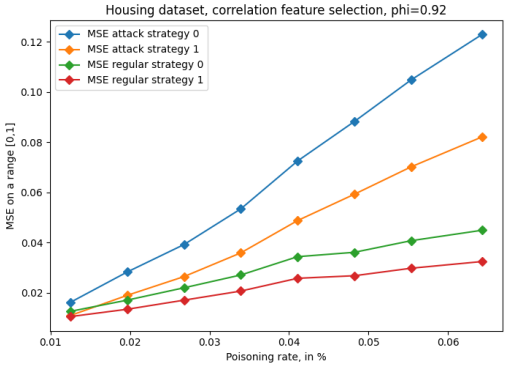


(c) MSE for $\phi = 0.7$ with correlation feature selection and IWPC dataset

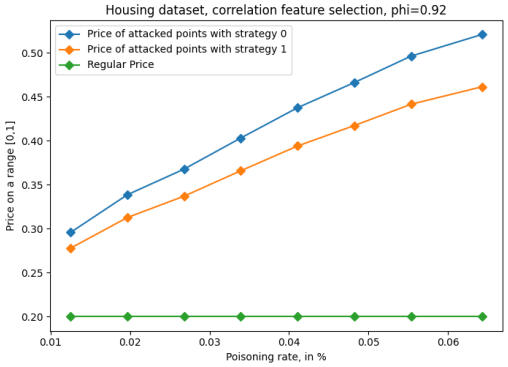


(d) Warfarin dosage for $\phi = 0.7$ with correlation feature selection and IWPC dataset

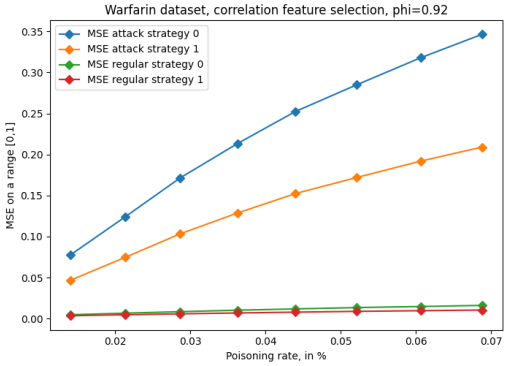
Figure A.5: Backdoor attack on linear regression model and $\phi = 0.7$



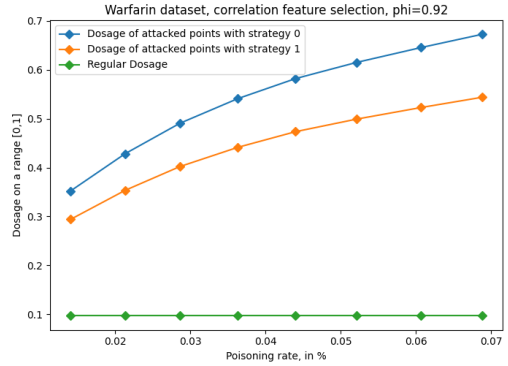
(a) MSE for $\phi = 0.92$ with correlation feature selection and housing prices dataset



(b) Price for $\phi = 0.92$ with correlation feature selection and housing prices dataset



(c) MSE for $\phi = 0.92$ with correlation feature selection and IWPC dataset



(d) Warfarin dosage for $\phi = 0.92$ with correlation feature selection and IWPC dataset

Figure A.6: Backdoor attack on linear regression model and $\phi = 0.92$