

Investigating LLM-Generated Feature Importance for XAI

Ciprian Bica

Investigating LLM-Generated Feature Importance for XAI

by

Ciprian Bica

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday October 13, 2026 at 11:00 AM.

Student number:	6321526	
Project duration:	November 10, 2025 – October 13, 2026	
Thesis committee:	Dr. Luciano Cavalcante Siebert,	TU Delft, thesis advisor
	MSc. Tina Mioch,	TU Delft & HU, daily supervisor
	Dr. B.J.W. Dudzik,	TU Delft, committee member

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Abstract

Complex machine learning models, such as XGBoost, are often used on tabular data, but explaining their predictions is a non-trivial task that requires the setup and configuration of post-hoc explainers such as SHAP or LIME. These explainers need access to the model and technical expertise, which are often unavailable when a model is proprietary or only accessible through a restricted interface. Large Language Models (LLMs) could offer an alternative, as prior work has shown that LLMs can rank feature importance without access to the model's internals, but only on small binary classification datasets, so it is unclear when their rankings agree with established explainers.

To address this gap, this thesis investigates under what conditions LLM-generated feature importance rankings agree with SHAP and LIME. To answer this, we trained XGBoost models on three datasets across different task types and dimensionalities: California Housing, Wine Quality, and Taiwan Bankruptcy Prediction. We then prompted two open-weights LLMs, Gemma 4 (31B) and GPT-OSS (120B), to rank which features are driving 100 individual predictions per dataset. For the prompting, we used five prompt templates derived from three approaches: text-driven prompts that provide feature names and task context, data-driven prompts that provide anonymized perturbed samples, and a novel mixed neighbors prompt that combines feature names with real contrastive neighboring samples. The rankings were compared against rankings from SHAP and LIME using Spearman correlation and top- k overlap.

We found that alignment is mainly dependent on the dataset characteristics and the information included in the prompt. The best setup on California Housing, mixed neighbors with Gemma, came close to the agreement between SHAP and LIME themselves (a Spearman correlation of 0.693 compared to 0.772). On Wine Quality, where SHAP and LIME are in slight agreement (0.226 Spearman correlation), most setups surpassed their agreement, with a text-driven prompt reaching 0.645 against SHAP. On Taiwan Bankruptcy Prediction, all setups stayed below the agreement between SHAP and LIME (at best 0.305 compared to 0.414), with data-driven prompts performing best. Across all three datasets, the choice of prompt template affected agreement more than the choice of LLM.

However, text-driven rankings often missed the most important feature of individual predictions, suggesting that their agreement comes largely from the LLM's own domain knowledge rather than the sample's feature values. Overall, LLM-generated rankings can offer a quick, rough estimate of which features are predictive, but are not able to completely replace traditional post-hoc explainers. In practice, LLMs are best suited as a first check when model access is restricted, while high-stakes decisions still require explanations computed from the model itself.

*Ciprian Bica
Delft, October 2026*

Contents

Glossary	iv
1 Introduction	1
2 Background	4
2.1 Tree-based models and XGBoost	4
2.2 The XAI field and the performance-explainability tradeoff	4
2.3 SHAP and LIME	5
2.3.1 SHAP	5
2.3.2 LIME	5
2.4 Large Language Models for explainability	6
3 Related Work	7
3.1 LLM as narrator	7
3.2 LLM for feature selection	7
3.3 LLM as post-hoc explainer	8
3.4 Research gap	8
4 Methodology	9
4.1 Assumptions	10
4.2 Prompting approaches	10
4.2.1 Data-centric prompts	10
4.2.2 Text-centric prompts	11
4.2.3 Mixed neighbors approach	11
5 Experimental Setup	12
5.1 The Datasets	12
5.1.1 Wine Quality	12
5.1.2 California Housing	12
5.1.3 Taiwan Bankruptcy Prediction	13
5.2 ML Model	13
5.3 LLM Choice & Implementation Details	13
5.4 Evaluation metrics	14
6 Results	16
6.1 Alignment by top- k overlap	16
6.1.1 California Housing	16
6.1.2 Wine Quality	17
6.1.3 Taiwan Bankruptcy Prediction	17
6.2 Alignment by Spearman	19
6.2.1 California Housing	19
6.2.2 Wine Quality	19
6.2.3 Taiwan Bankruptcy Prediction	19
6.3 Aggregated Results	20
7 Discussion	22
7.1 Agreement between SHAP and LIME	22
7.2 Dataset characteristics	22
7.2.1 Dimensionality	23
7.2.2 Feature names	23
7.2.3 Task type	24

7.3	Prompt templates	24
7.3.1	Text-driven prompts	24
7.3.2	Data-driven prompts	24
7.3.3	Mixed neighbors prompt	25
7.4	LLM Comparison	25
7.5	Limitations	25
7.6	Societal implications	26
8	Conclusion	28
8.1	Applicability	29
8.2	Future work	29
A	Prompt Templates	30
A.1	In-Context Explainers	30
A.1.1	P-ICL	30
A.1.2	PG-ICL	31
A.2	LLM-Select	31
A.2.1	LLM-Rank.	31
A.2.2	LLM-Score	32
A.3	Mixed neighbors	32
B	Dataset Analysis	34
C	Top-k Overlap Results	35
D	Use of AI	37

Glossary

Gemma Gemma 4 (31B), an open-weights, instruction-tuned large language model developed by Google DeepMind.

GPT-OSS GPT-OSS (120B), an open-weights Mixture-of-Experts (MoE) large language model developed by OpenAI.

LIME Local Interpretable Model-agnostic Explanations (Ribeiro et al., 2016), a local surrogate-based post-hoc explainer.

LLM-Rank / LLM-Score Prompt templates, adapted from Jeong et al. (2025), for ranking and scoring feature attributions using semantic feature names.

LLM-Select (Jeong et al., 2025) A method for feature selection using large language models.

P-ICL / PG-ICL Perturb In-Context Learning and Perturb + Guide In-Context Learning, data-centric prompt templates that provide perturbed local neighborhood data points to the LLM, adapted from Kroeger et al. (2024).

SHAP SHapley Additive exPlanations (Lundberg & Lee, 2017), a method for explaining model predictions based on game theory.

XAI Explainable Artificial Intelligence

Introduction

Machine learning (ML) algorithms are increasingly being used in a variety of domains, some of which are high-stakes, such as finance, healthcare, national security, and criminal justice (Pugliese et al., 2021). Their applications range from agriculture to complex financial fraud detection systems. These systems are able to perform classification, prediction, and decision-making tasks by learning from and analyzing past data.

However, this increase in the deployment of complex ML algorithms and deep learning methods in decision-making for several high-stakes domains has already raised concerns about a possible lack of understanding among their users of how decisions are made. This is a concerning trend, especially in critical domains such as the criminal justice system, finance, and healthcare (Rudin, 2019; Varshney & Alemzadeh, 2017).

More broadly, this large-scale deployment also points to an emerging gap between the complexity of Artificial Intelligence (AI) systems and the ability of non-expert users to interpret their outputs (Sarkar, 2022). This could potentially result in people relying on AI-generated outputs without the necessary understanding or domain knowledge that a situation might require. It may not be easy for decision-makers to evaluate the correctness, reliability, or limitations of these systems.

The types of models that are too complex for humans to intuitively understand are referred to as ‘black-box’ models (Guidotti et al., 2018). Because of the pervasiveness of such models, various Explainable Artificial Intelligence (XAI) approaches have been created to help address this problem (Adadi & Berrada, 2018; Arrieta et al., 2019).

Currently, two of the most commonly used methods to address this explainability issue are applying a post-hoc explainer, such as SHAP (SHapley Additive exPlanations) (Lundberg & Lee, 2017) or LIME (Local Interpretable Model-agnostic Explanations) (Ribeiro et al., 2016) to the model. These methods output feature attribution coefficients based on different ways of calculating how much individual features change the model’s predictions.

Moreover, previous work has shown that tabular data is one of the most common data types used for ML tasks (Borisov et al., 2022). For this data type, tree-based ML models, such as Random Forests, XGBoost (Chen & Guestrin, 2016), and LightGBM (Ke et al., 2017), have been shown to consistently achieve strong results (Grinsztajn et al., 2022). In particular, these specific ML models tend to function like black boxes (Zhou et al., 2021), providing predictions without much context for the user and highlighting the XAI problem mentioned earlier.

Most post-hoc explanation methods assume a non-trivial level of access to the predictive model in order to compute feature attributions. This mainly refers to technical access, such as model parameters, repeated querying of the model, or information about the models themselves. For example, SHAP requires access to the model’s internal structure and repeated model evaluations depending on the implementation, while LIME relies on querying the model with large numbers of perturbed samples. In many real-world deployments, such access may not be available or practical. Predictive systems may be proprietary, accessible only through restricted interfaces, or integrated into legacy infrastructure where direct inspection of the model is not possible (Rudin, 2019).

Beyond the access problem, both methods require a certain level of technical expertise to configure, interpret, and deploy correctly. This is especially true for an already-deployed setup, where applying a

post-hoc explainer requires additional engineering work, such as creating a wrapper for the model to make it accessible to the explainer and potentially reconstructing a part of the model pipeline for that purpose.

Beyond this limitation, there are also economic incentives that can create challenges for explainability in certain environments. For example, when companies are contracted as third parties to develop AI systems, they might wish to protect their commercial interests and intellectual property by keeping their models inaccessible or unnecessarily complex (Lambin & Raizonville, 2025).

Together, these technical, economic, and engineering issues can compound to leave end-users with few options to help them understand the reasons why a specific prediction was made.

Large Language Models (LLMs) (Zhao et al., 2025) could represent a way to bridge this gap. LLMs are complex AI systems based on the transformer architecture that can generate text after being trained on large amounts of data. They have also shown the capability to perform complex reasoning tasks with no knowledge of previous examples and without task-specific training ('zero-shot') (Kojima et al., 2023). These properties make LLMs an interesting choice for improving explainability.

Specifically, an LLM does not need to have access to the internal structure or learned parameters of an ML model. This means that any feature importance produced by an LLM would be based mostly on its internal statistical knowledge and reasoning capabilities, rather than on the actual model computations. This approach has both advantages and disadvantages. On the positive side, the LLM can incorporate domain-specific knowledge and reason about the data in ways that purely mathematical methods like SHAP and LIME cannot.

On the negative side, because the LLM has no access to the relationships the model has actually learned and cannot calculate hard statistical correlations, its rankings may reflect its own general world knowledge and training biases rather than the patterns the model relies on. This matters most when the model's training data is imperfect, incomplete, or biased. In that case, SHAP and LIME will reflect the resulting flaws in the model's behavior, which is exactly what an explanation should reveal, whereas an LLM may rank features according to what is plausible in the domain and leave these flaws hidden.

At the same time, it has been shown that LLMs are capable of successfully selecting the features with the most predictive power for the upstream task of model training while being given only the input names and details about a prediction task, or just a few concrete data points (Jeong et al., 2025; Li et al., 2024).

This reveals an interesting question about LLMs in the context of XAI: if LLMs can choose the features with the most predictive power to train a model, can they also approximate feature contribution values for a model that has already been trained? If their outputs align with existing post-hoc methods, LLMs could serve as a lightweight proxy for approximating feature contributions in settings where traditional XAI tools cannot be used, requiring no access to the model's internal structure and, for some prompting approaches, no model queries beyond the prediction being explained. This approach could be particularly well-suited to any setting where end-users may lack the technical background required to configure and set up SHAP or LIME, or where model access is restricted. This could also lead to higher accessibility of explainability for non-expert users across a wide range of deployments.

To this end, the research question we aim to answer in this work is:

Under what conditions do LLM-generated feature importance rankings agree with post-hoc explanation methods such as SHAP and LIME?

To help answer this question, we propose a methodology to test whether LLMs can approximate the feature importance of a trained predictive model and to identify under what conditions (prompting strategy, dataset choice, etc.) this alignment improves. The proposed workflow consists of training a set of XGBoost predictive models on multiple datasets, then generating feature importance values using SHAP and LIME for these models. After that, we request feature importance rankings from an LLM using various prompting strategies and compare the results quantitatively. While this study focuses on tree-based models as a practical choice for this data type, the proposed methodology is model-agnostic and could, in theory, be applied to any type of model.

In the next chapters, we provide some theoretical background and related work about post-hoc explainers, LLMs, and prediction models. Chapter 4 details the proposed methodology. In Chapters 5 and 6, we present the experimental setup and results. Chapter 7 discusses these results, along with their limitations and societal implications. Finally, Chapter 8 concludes the thesis and suggests

directions for future work.

2

Background

This chapter details the concepts that are relevant to this thesis. It is split into four key sections. First, we provide some context about tree-based models and XGBoost in particular. Second, we give some broad information on the XAI field and the performance-explainability tradeoff. Third, we discuss the use of post-hoc explainers such as SHAP and LIME on ML models. Finally, we examine the capabilities and limitations of LLMs related to explainability.

2.1. Tree-based models and XGBoost

In this thesis, we investigate when LLMs can align with feature importance rankings produced by post-hoc explainers. For this task, we chose a widely used class of predictive models: gradient-boosted decision tree ensembles, and XGBoost in particular. This section introduces this model family and motivates its use for this thesis.

Explained briefly, tree-based models work by splitting the feature space into different sections and assigning predictions to each section (Chen & Guestrin, 2016; Ke et al., 2017). This type of model is composed of building blocks called decision trees. A single tree is easy to interpret but is very prone to overfitting. As a consequence, ensemble methods were created to mitigate this. They work by combining many such trees in different ways. Gradient boosting models, in particular, build trees sequentially so that new trees can correct the errors of old ones. While this setup is able to reach a good level of predictive performance, it also creates difficulties in explaining why the model made a certain decision.

For this study, we chose to investigate XGBoost, which is among the most popular implementations of gradient-boosted tree models. XGBoost (eXtreme Gradient Boosting) (Chen & Guestrin, 2016) is a highly optimized implementation of gradient-boosted decision trees, which has been proven to be applicable in a wide range of domains. It can be considered the state of the art for both classification, where the goal is to predict a discrete label, and regression tasks, where the goal is to predict a continuous value (Grinsztajn et al., 2022; Schwartz-Ziv & Armon, 2021) on tabular data. This model, unfortunately, has the downside of acting like a black box, requiring a separate explanation tool for properly understanding the most relevant factors contributing to its outputs, which is why it was chosen for this thesis.

2.2. The XAI field and the performance-explainability tradeoff

AI systems have evolved from easily interpretable expert-based decision systems, which are explainable by design, to relatively complex models such as neural networks and ensemble systems (e.g., XGBoost). This shift has introduced the need for smarter, more robust explanation mechanisms for ML models. In response, the XAI field has expanded greatly over recent years, with systematic reviews showing an increase in the number and diversity of novel explanation methods (Abusitta et al., 2024).

The aforementioned growing need for explanation methods is also expressed through the existence of a tradeoff between predictive performance and model explainability (Adadi & Berrada, 2018). More explainable models, such as decision trees or linear regression, are believed to be unable to capture more complex relationships in the data due to their inherent simplicity, which limits their performance.

At the other end of the spectrum, more complex models, such as deep neural networks, are believed to be able to capture non-linear relationships hidden in the data, achieving better overall accuracy. As a consequence of this tradeoff, the models most likely to be used because of their superior performance are also the least explainable. These ‘black-box’ models are typically uninterpretable by humans without the use of additional tools (Guidotti et al., 2018). Gradient-boosted decision trees are a good example of this.

2.3. SHAP and LIME

This section introduces SHAP and LIME, which are considered the two reference baselines against which agreement is evaluated. Post-hoc explainers, in general, can be applied after the model has been trained to help provide more information about why a certain prediction was made. In practice, this is useful because they can be applied to any model without directly modifying it. SHAP and LIME are currently two of the most commonly used post-hoc explanation methods for tabular data (Gramegna & Giudici, 2021).

2.3.1. SHAP

SHAP (SHapley Additive exPlanations) (Lundberg & Lee, 2017) was created by taking inspiration from game theory. It starts by asking how much a specific collaborative ‘player’ (the feature) contributed to the ‘outcome’ (the prediction). The Shapley value ϕ_i of a feature i is the weighted average of the change in the model’s expected output when feature i is added to each possible subset of the other features.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_x(S \cup \{i\}) - f_x(S)] \quad (2.1)$$

where:

- ϕ_i is the Shapley value (feature attribution) of feature i .
- F is the complete set of all input features, with $|F|$ denoting its cardinality.
- S is a subset of features drawn from F after excluding feature i ($S \subseteq F \setminus \{i\}$), with $|S|$ representing its size.
- $f_x(S)$ is the expected output of the model conditioned on the subset of features S .
- $\frac{|S|!(|F| - |S| - 1)!}{|F|!}$ is a combinatorial weighting factor equal to the probability that, in a random ordering of all features, exactly the features in S appear before feature i .

The sum in Equation 2.1 runs over every subset of features, making exact Shapley values exponentially expensive to compute. For tree-based models specifically, TreeSHAP (Lundberg et al., 2019) offers an efficient implementation built directly into XGBoost. It exploits the tree structure to obtain the same values in polynomial time, scaling with the size of the ensemble rather than the number of features. However, for SHAP to function properly, feature independence is assumed. When features are correlated, they can split the credit for a shared contribution, which artificially lowers each feature’s importance.

2.3.2. LIME

LIME (Local Interpretable Model-agnostic Explanations) (Ribeiro et al., 2016) is a model-agnostic explainer and represents an alternative to SHAP. To explain a single prediction, it first generates a new dataset of perturbed samples around the instance of interest and records the black-box model’s predictions for them. Each sample is then weighted by its proximity to the original instance, so that nearby samples have more influence. Finally, LIME fits a simple, interpretable surrogate model (typically a sparse linear model) on this weighted dataset. The coefficients of this surrogate model serve as the feature importance scores for that instance.

$$\xi(x) = \operatorname{argmin}_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g) \quad (2.2)$$

where:

- $\xi(x)$ is the resulting local post-hoc explanation model for instance x .
- f is the complex black-box model being explained.
- g is an intrinsically interpretable model (e.g., sparse linear regression) selected from a family of simple models G .
- $\mathcal{L}(f, g, \pi_x)$ is the local loss function measuring how unfaithful the surrogate model g is to the black-box model f .
- π_x is a proximity kernel defining the size of the local neighborhood around the target instance x .
- $\Omega(g)$ is the complexity penalty of the explanation model g (e.g., a regularization term limiting the number of non-zero coefficients) to ensure human interpretability.

Both methods have practical and conceptual limitations worth acknowledging. First, both require either direct access to model internals (e.g., TreeSHAP) or the ability to query the model repeatedly at scale (LIME), making them unsuitable in settings where the model is proprietary or inaccessible. Second, both are technically demanding to configure and interpret correctly for non-expert users, which limits their use in operational settings without dedicated ML support. Third, and also relevant for this work, SHAP and LIME do not always agree with one another (Krishna et al., 2025). In this thesis, both are used as reference baselines rather than gold standards, and alignment with either can be treated as evidence of relevance.

A growing number of works have begun exploring a route to explainability that avoids the earlier-mentioned access requirements by using the reasoning and world knowledge of LLMs to produce human-like explanations (Bilal et al., 2025).

2.4. Large Language Models for explainability

LLMs are deep learning models based on the transformer architecture, which enables them to capture long-range dependencies and complex relationships within textual data. They are capable of generating and summarizing text based on knowledge gained from training on massive datasets (Zhao et al., 2025). LLMs are also general-purpose systems, so they apply the patterns they learn to previously unseen problems and tasks. Over the past few years, the capabilities of these models have improved massively (Minaee et al., 2025; Wei et al., 2022). Today, these models exhibit abilities that extend beyond simple text processing, demonstrating the capacity to write computer code, reason with common sense, and perform a wide array of other complex cognitive tasks (Kojima et al., 2023). Chapter 3 describes how these capabilities have been used for explainability.

However, LLMs also have drawbacks, two of which are particularly relevant to this work. First, LLMs tend to “hallucinate”, that is, confidently present untrue information as fact (Ji et al., 2023). In the context of explainability, this means an LLM may produce a plausible-sounding explanation that does not reflect the model’s actual behavior. Second, LLM outputs are stochastic: the exact same prompt can produce different outputs across runs. This poses a problem for explainability, as we would want LLM-generated explanations to be as consistent as possible for the same input. To mitigate this, we reduce the stochasticity of the model’s outputs in our experiments as much as possible, with further details provided in Chapter 5. Hallucinations, however, are not resolved by this and remain a fundamental limitation of the technology that cannot be completely eliminated.

3

Related Work

Recently, a growing body of literature has begun exploring the intersection of LLMs and explainable AI, including reviews of the practice and future directions (Bilal et al., 2025; Zytek, Pidò, & Veeramachaneni, 2024). Within this space, researchers are using LLMs not only to directly explain ML model predictions (Kroeger et al., 2024), but also to improve explanations created using established XAI algorithms or to create XAI frameworks centered around LLMs (Bhattacharjee et al., 2024; Naudot et al., 2025; Paraschou et al., 2026).

In this chapter, we will examine three approaches that have been used to include LLMs in the explanation pipeline. We first address the use of LLMs as narrators for other explainers. Then, we consider LLMs being used for the upstream task of feature selection before a model is trained. Finally, we consider LLMs in the role of a complete post-hoc explainer, which is the focus of this thesis.

3.1. LLM as narrator

Due to the textual capabilities of LLMs, an intuitive approach to involving them in ML explanations is to include the output of a traditional post-hoc explainer, such as SHAP or LIME, in their prompt (Lukassen et al., 2026). This method mainly aims to reduce the complexity of numerical explanations by translating them into a more natural textual explanation. This is done to facilitate the understanding of opaque ML systems for human decision-makers.

For example, Zytek, Pidò, Alnegheimish, et al. (2024) create Explingo, a system that generates natural language explanations using SHAP explanations as a base. This system is then evaluated according to several criteria, such as accuracy, fluency, and conciseness. This shows that LLMs can work with feature attributions to improve explainability. Whereas Explingo evaluates the quality of the generated text, Paraschou et al. (2026) propose a novel XAI framework that prompts LLMs using LIME explanations and contextual, domain-relevant information, validated in a user study. This style of explanation can result in improved readability and ease of understanding for the end-user.

However, this type of framework is still dependent on the output of a traditional post-hoc explainer. The LLM is only tasked with the presentation of the feature importance values. This raises the question of whether an LLM could estimate feature importances by itself, which is what the work on feature selection presented in the next section explores.

3.2. LLM for feature selection

Due to LLMs' proficiency in working with textual data, another possibility for using them earlier in the ML pipeline is during the feature selection process. Feature selection is the process of choosing, before training, the features most likely to contribute useful information to a model.

Li et al. (2024) compare textual and data-driven LLM feature selection methods and find that LLM text-driven feature selection methods are comparable to traditional feature selection methods (mutual information filtering, Recursive Feature Elimination, and Minimum Redundancy Maximum Relevance selection). This comparison resembles the one in this thesis, except that we evaluate feature importance rankings for an already-trained model rather than feature selection.

On a related note, Jeong et al. (2025) also use LLMs to select features for training ML models. Their methodology involves giving no actual feature values to the LLM, only providing it with a description of the dataset and the names of all features. After comparing LLM scores against several feature importance metrics (Spearman correlation, mutual information) and attributions produced by a post-hoc method (SHAP), they find that LLM importance scores do not consistently align closely with any of them. However, they do note that alignment improves with model scale. The scores they obtain are global and prediction-independent, elicited before any model exists, whereas a post-hoc explainer needs an already-trained model. The task, therefore, remains one of feature selection rather than explanation. These findings are relevant for this thesis as they show that LLMs are capable of creating and using a functional concept of feature relevance using only feature names and task descriptions. The authors also note that LLMs could help in the data collection process by deciding which features to collect first.

Jeong et al. (2025) also propose that an interesting future research direction would be to investigate which, if any, prompting strategies increase the alignment of LLM-generated importance scores with desired notions of importance. Our work aims to answer a closely related question: rather than estimating feature relevance before model training, could an LLM approximate this after the model has been trained?

3.3. LLM as post-hoc explainer

A third approach uses the LLM itself as the post-hoc explainer: given a trained model's inputs and predictions, the LLM is asked to estimate which features drove those predictions. This LLM-based approach differs fundamentally from traditional post-hoc techniques. Because an LLM does not have access to the internal structure, weights, or training process of the predictive model, it cannot compute mathematically grounded attribution scores. Instead, it must infer which features are likely to have influenced the prediction by relying on the examples it is shown and on the general statistical intuition and domain knowledge it learned during its training. Consequently, LLM-generated feature importance is an inferred estimate rather than a measurement of the model's actual behavior.

Our work builds directly on the findings of Kroeger et al. (2024), who explore whether LLMs can approximate feature importance rankings for predictive models. The authors show that when given just a few examples of a model's inputs and outputs, an LLM can rank features with an accuracy comparable to established post-hoc methods like SHAP and LIME across various datasets and metrics.

Although Kroeger et al. (2024) show that LLM-generated rankings are similar to those produced by SHAP and LIME, they do not consider under what conditions this alignment increases. Additionally, their experiments only examine small binary classification datasets, and, by design, their approach limits the information provided to the LLM by anonymizing feature names.

3.4. Research gap

Taken together, these lines of work leave an open question. Narrator approaches depend on the output of an existing explainer, and feature selection approaches estimate relevance globally before any model is trained, so neither tests whether an LLM can act as a post-hoc explainer itself. Kroeger et al. (2024) show that such alignment is possible, but leave open when it occurs. In particular, it remains unclear how the information given to the LLM (feature names and task context, perturbed samples, or both), the characteristics of the dataset (task type and dimensionality), and the choice of LLM affect agreement with SHAP and LIME. This thesis addresses this gap by comparing text-driven, data-driven, and a novel mixed neighbors prompting approach, using two LLMs, on XGBoost models trained on three datasets spanning regression, multi-class classification, and binary classification. In doing so, it also contributes to the question raised by Jeong et al. (2025) of which prompting strategies improve the alignment of LLM-generated importance scores.

Methodology

We examine the classic ML setup where we have a black-box model trained on a labeled dataset, with no access to the model internals or learned parameters. We assume, however, full access to the dataset, including feature names and values, as well as some descriptive context (e.g., a data card from Kaggle). Under these assumptions, we investigate whether an LLM can produce a feature importance ranking for a given data point that aligns with the rankings produced by traditional post-hoc attribution methods that have direct model access.

The proposed workflow, illustrated in Figure 4.1, consists of four distinct stages: (1) training a predictive model on each dataset, (2) generating baseline feature importance values using SHAP and LIME, (3) asking for feature rankings from an LLM in different ways, and (4) comparing the results quantitatively using the defined metrics. In Stage 3, the methodology compares three distinct approaches to creating LLM-generated feature importance rankings based on different principles: text-driven (shown in blue; adapted from Jeong et al., 2025), data-driven (shown in yellow; based on Kroeger et al., 2024), and a novel mixed neighbors approach introduced in this thesis, based on the neighborhood of the target (shown in red). Each approach is applied to every data sample, and the three appear as parallel branches within Stage 3 in Figure 4.1. The text-driven and data-driven approaches consist of two separate prompt templates each, while the mixed neighbors approach uses a single prompt template.

For this thesis, we focus on local feature attributions. This is because it can be argued that explaining individual predictions is the main use case for post-hoc explainers for end-users. It should be noted that using a ranking discards some information (e.g., two features having very similar importance but

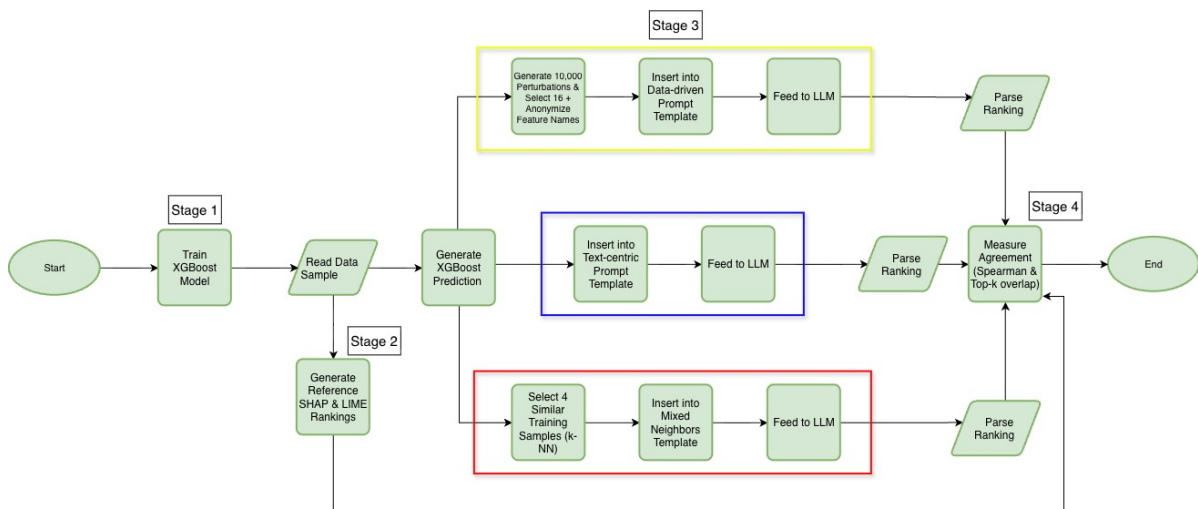


Figure 4.1: Process flowchart of the methodology, labeled with the four stages described above. Stage 3 is labeled once but covers all three colored branches: data-driven (yellow), text-driven (blue), and mixed neighbors (red). The rankings produced by each branch are compared against the same SHAP and LIME reference rankings in Stage 4.

being assigned distinct rankings). This is an inherent limitation of using ranking-based evaluations and will be further discussed in the Discussion chapter.

4.1. Assumptions

The proposed methodology operates under some assumptions about the datasets. While these assumptions are not always made explicit in the literature, they are implicitly present in the experimental designs of both Kroeger et al. (2024) and Jeong et al. (2025). We state them here to make the scope and applicability of the methodology more transparent.

1. Semantically meaningful feature names are required for the text-driven methods to function properly, as they heavily rely on the LLM's domain knowledge. This is a critical requirement, as it is the only way to give the LLM any meaningful information about the dataset and the model for the text-driven approaches.

This is more relevant for the text-based approach adapted from Jeong et al. (2025), who use datasets with informative and well-understood variable names and descriptions and also address this in their work. This means the dataset selection for this thesis cannot include visual, audio, or unprocessed categorical data.

2. The datasets are assumed to consist of numerical features. Although SHAP and LIME are both capable of dealing with categorical features, our chosen prompting approaches are not designed to cope with them. Jeong et al. (2025), for example, address this problem by one-hot encoding categorical features and grouping the resulting columns into a single feature. We mainly focus on datasets with only numerical columns and encode binary flag columns from Taiwan Bankruptcy Prediction to either 0 or 1 and treat them as numerical features.

3. A low to medium number of features (a maximum of 20 to 25 features after preprocessing) is also a significant requirement. These methods can be applied to larger datasets only if proper pruning and preprocessing drastically reduce the number of features first.

Both Kroeger et al. (2024) and Jeong et al. (2025) either use datasets with fewer than 15 features or preprocess them to dramatically decrease their dimensionality (grouping multiple features into 'concepts' for the latter). We used the Taiwan Bankruptcy Prediction dataset specifically to test performance near the upper end of this range.

The Taiwan Bankruptcy Prediction dataset was pruned to 20 features after exploratory testing when creating the pipeline found that having around 25 or more features caused both LLMs to output malformed rankings.

4.2. Prompting approaches

To generate feature importance rankings for a specific data sample, we prompt the LLM in three different ways: two families adapted from previous work, **data-driven** and **text-driven**, and a novel **mixed neighbors** prompt that combines elements of both. The motivation and design of each are described below.

4.2.1. Data-centric prompts

Both prompts in this family contain the following components: *Context* of the predictive model, the *Dataset*, which is a collection of instances sampled from the local neighborhood of the target data point, the *Question*, which specifies the task the LLM needs to complete, and the *Instructions*, which contain a set of rules the LLM must follow while generating its response. The component terminology and its role in the design of the prompts are directly inherited from the original paper (Kroeger et al., 2024) for consistency.

P-ICL (Perturb In-Context Learning) combines these four components without any new additions.

PG-ICL (Perturb + Guide In-Context Learning) replaces the general guidance in the *Instructions* component with a more detailed, dataset-agnostic, step-by-step procedure. For example, the model is instructed to consider both positive and negative correlations equally and to keep a running rank of the order of the features.

These types of prompts also obscure feature names from the model, instead giving generic feature labels for all datasets (e.g., 'A', 'B', 'C', etc.). This is done in an effort to make the language model focus on the feature values, the relationship between them, and how these shift the prediction. This design choice is also meant to reduce the influence of semantic information of the feature names. We hypothesize that LLMs would perform better on low-dimensional datasets.

4.2.2. Text-centric prompts

The text-driven prompts were originally designed by Jeong et al. (2025) for feature selection, to choose features for training a downstream model. We adapted these templates to the task of ranking the features that contributed to a prediction after a model has been trained. Although feature selection and post-hoc feature ranking are not the same task, they share similarities, such as estimating the predictive power of each feature relative to the others. LLM-Score and LLM-Rank are the best-performing prompts from Jeong et al. (2025), and both were designed to take advantage of the LLM's semantic knowledge by providing it with feature names and the goal of the ML task.

LLM-Score elicits an importance score for each feature in the dataset, where a high score indicates that the feature meaningfully contributed to the prediction, along with a reason for each score (in JSON format). The generated scores range from 0 to 1, and the feature ranking is obtained by sorting these scores.

LLM-Rank prompts the LLM for a complete ranking of input features, in order of prediction importance.

The text-driven approach is likely to excel on datasets where feature names are very descriptive and values are easy to understand. We hypothesize that this approach could produce rankings that reflect general domain knowledge and are thus less specific to the individual prediction.

4.2.3. Mixed neighbors approach

We introduce a novel approach that aims to combine the strengths of the text-driven and data-driven approaches in one prompt template. Like the text-driven prompts, it retains the feature names and the description of the prediction task. Like the data-driven prompts, it provides the LLM with input-output pairs from the target's local neighborhood. However, instead of synthetic perturbations of the target, it uses real samples from the training data: four nearby training samples selected by k-nearest-neighbor search, two with the same prediction as the target and two with a different one (for regression, two with a higher and two with a lower predicted value). The motivation behind this design is to let the LLM examine the real local neighborhood of a specific data sample, including contrasting predictions, while still using feature names. The exact wording of each prompt type can be found in Appendix A, demonstrated on a sample from the Wine Quality dataset.

5

Experimental Setup

In this chapter, we present concrete implementation details and design choices for applying the methodology of each approach to the selection of datasets. We test five prompt templates from three approaches (two prompt families and the mixed neighbors approach) on three datasets for different tasks (binary classification, multi-class classification, and regression). For this, we use two open-weights LLMs from two families with different architectures. We then compare the results quantitatively using two metrics described in Section 5.4.

5.1. The Datasets

Dataset	Domain	Task	Target	Instances	Features
Wine Quality	Wine	Multi-class classification	Quality rating	1,599	11
California Housing	Real estate	Regression	Median house value	20,640	8
Taiwan Bankruptcy	Finance	Binary classification	Bankrupt or not	6,819	20 (of 95)

Table 5.1: Overview of the three datasets. For Taiwan Bankruptcy Prediction, the 95 original features were pruned to the 20 used in the experiments, and only 220 of the 6,819 companies are bankrupt.

Three datasets were chosen as the basis for the experimental design, covering a range of domains, dimensionalities, and task types: Wine Quality (Cortez et al., 2009a, 2009b) (red wine subset, 11 features, multi-class classification), California Housing (Pace & Barry, 1997) (8 features, regression), and Taiwan Bankruptcy Prediction (Liang et al., 2016; “Taiwanese Bankruptcy Prediction”, 2020) (95 features, pruned to 20, binary classification). Table 5.1 gives an overview. All three have semantically meaningful feature names and are mostly numerical, allowing the LLM to use its domain knowledge.

5.1.1. Wine Quality

The Wine Quality dataset is a multi-class classification dataset. It is a UCI dataset collected from red and white vinho verde wine samples from northern Portugal, where the goal is to predict a wine’s quality rating from its physicochemical measurements (e.g., acidity, density). The dataset contains 1599 samples and 11 features. The target variable of this dataset has a bell-shaped distribution, with a concentration on average-quality wines (e.g., 5 and 6). Outlier wines, such as high-quality and low-quality ones, are underrepresented in comparison. The quality ratings start at 3 and end at 8, giving six classes, which are encoded as 0 to 5 for training.

5.1.2. California Housing

The goal of this dataset is to predict the median housing price of a US Census block group (the smallest geographical unit for which the US Census Bureau publishes sample data, usually with a population of 600 to 3000 people) in California, based on data collected from the 1990 US Census. This dataset has 8 features, 20640 data points, and is a regression dataset. The mean house value of the dataset is \$207,000 and ranges from \$15,000 to \$500,000. The target distribution is unimodal, with a right skew

and a spike at the right end. This is because properties worth more than \$500,000 were capped at this value.

5.1.3. Taiwan Bankruptcy Prediction

This dataset is a binary classification dataset, and the goal is to predict whether a company will go bankrupt based on its financial data. The data were collected from the Taiwan Economic Journal for the years 1999 to 2009. Company bankruptcy was defined based on the business regulations of the Taiwan Stock Exchange. The dataset has 95 features and 6,819 instances. The dataset is heavily imbalanced: only 220 of the 6,819 companies are bankrupt. It was chosen due to its high dimensionality, which is expected to be more difficult for an LLM to deal with. Because of the large number of features, we constructed a pruned variant of 20 features using predictive power and tree-based importance criteria (see Appendix B).

5.2. ML Model

A supervised XGBoost model was trained on each of the three datasets. As mentioned in the Background chapter, this specific type of model was chosen because of its strong predictive performance and because it is a complex black-box model where post-hoc explainers are usually applied. The training and test sets were created using an 80-20 train-test split with a fixed seed for reproducibility. Hyperparameter tuning was conducted using 3-fold cross-validation on the training set. The explored hyperparameters included maximum tree depth, learning rate, number of estimators, subsampling ratio, and column sampling ratio. The best-performing configuration was selected according to cross-validated accuracy for Wine Quality, F1 for Taiwan Bankruptcy Prediction, and MSE for California Housing.

We then applied SHAP and LIME to each XGBoost model, focusing on local (instance-level) feature attribution. Using both methods allows us to assess whether LLM alignment is consistent across different approaches. Absolute values were used, as the magnitude reflects contribution strength regardless of the sign. For SHAP, the TreeSHAP implementation (Lundberg et al., 2019) was used. For LIME, we generated the explanations using the default setting of 5000 perturbations per instance.

Both methods were used, as they are known to disagree with each other depending on the dataset and model. This makes each of them an independent baseline instead of choosing one and having it be the absolute gold standard. A more detailed comparison of the agreement of SHAP and LIME can be found in Table 5.2.

A feature ranking was created according to each explainer for 100 test instances per dataset: the first 100 for California Housing and Wine Quality, and a stratified sample of 40 positive and 60 negative instances for Taiwan Bankruptcy Prediction, to compensate for its heavy class imbalance. This number was chosen because it is large enough to produce stable average metric scores while remaining feasible given computational, time, and API cost constraints.

We evaluated the best-performing XGBoost models on the complete test sets. For the regression task on California Housing, the model achieved a strong predictive performance with an R^2 of 0.847 (MSE: 0.201, RMSE: 0.448, MAE: 0.294, in units of \$100,000). For the multi-class classification on Wine Quality, the model reached an accuracy of 0.707 and a weighted F1-score of 0.694. Finally, for the binary classification on Taiwan Bankruptcy Prediction, the model achieved an AUROC of 0.953 and an F1-score of 0.552; accuracy (0.950) is uninformative here due to the class imbalance.

5.3. LLM Choice & Implementation Details

After testing with several LLMs, we used Gemma 4 (31B) (Google DeepMind, 2026) and GPT-OSS (120B) (OpenAI, 2025) accessed through the OpenRouter API. They were chosen as both are open-weights, instruction-tuned LLMs that can follow instructions well. In keeping with the access-restricted motivation of the study, the LLMs are never given the model's weights, access to the internal structure, or explicit SHAP/LIME values. The LLMs used were also not fine-tuned on any of the datasets in any way.

We provide all the prompt templates used in our experiments in Appendix A. Five prompt templates were used, including two data-centric prompts that provide the LLM with perturbed input and output pairs, two text-based prompts that rely more on domain context, and a novel mixed neighbors template. Branching from these prompt types, small modifications were made to improve parsing and performance, such as updating output instructions and adding more dataset-specific context and ex-

amples. The prompts were also adapted to the task of generating rankings if they initially contained different instructions.

The local neighborhoods of data points shown to LLMs (used in data-centric prompt templates) were built by perturbing all features of the original data sample with Gaussian noise. All feature values were perturbed within a standard deviation around the mean. This keeps the perturbed instances relatively consistent with the original sample and the original feature distribution. For each instance, 10,000 perturbed versions are generated, and only 16 are selected. For the classification datasets, selection is stratified over the predicted classes for a better representation of the model’s prediction process, and within each class, the samples with the highest predictive confidence are selected to better align the LLM with the model’s behavior. For California Housing, which has a continuous output, stratification by class is not possible. Instead, the range of predicted values over the 10,000 perturbations was divided into 16 equal-width brackets, and from each bracket, the sample whose predicted value is furthest from the target instance’s prediction was selected. This spreads the selected samples across the full range of the model’s outputs, mirroring the role of class stratification in the classification setting.

For the mixed neighbors prompt template, four examples are used from the local neighborhood of the target, selected using k -nearest-neighbor search under Euclidean distance from the training set. They are further divided into two groups, one group with the same predicted class as the target sample, and another group with a different predicted class from the target (for California Housing, which is a regression task, the split is instead between neighbors with a higher and a lower predicted value than the target). Four examples were used after exploratory testing while building the pipeline revealed that larger numbers of examples did not improve the results.

As mentioned earlier, in order to reduce stochastic variability, the LLM was queried with the temperature set to 0.

5.4. Evaluation metrics

Evaluation was conducted for each setup across the same 100 test instances from each dataset. Metric scores were computed per instance first, and the mean across all 100 instances was calculated to obtain overall alignment measures. To further account for the stochasticity inherent in LLMs, each experiment was run three times, and the results reported represent the mean across runs. For the evaluation, we used two metrics to measure the agreement and similarity between the methods:

- **Top- k overlap**, computed for each k from 1 to $|\mathcal{F}|$, measures whether two methods agree on which features matter most, regardless of their order within the top k . Let A_k and B_k be the sets of the k highest-ranked features under the two methods. The top- k overlap is

$$O_k = \frac{|A_k \cap B_k|}{k}, \quad (5.1)$$

the fraction of the top k features that both methods share. It reaches 1 at $k = |\mathcal{F}|$, and its expected value for a random ranking is $k/|\mathcal{F}|$.

- **Spearman’s Rank Correlation** is used to measure the correlation between the rankings, where +1 indicates identical orderings, 0 indicates no monotonic relationship, and −1 signals a perfectly reversed ordering. It is computed as $\rho = 1 - \frac{6 \sum_i d_i^2}{n(n^2 - 1)}$, where d_i is the difference in rank assigned to feature i by the two methods and $n = |\mathcal{F}|$. Due to the squaring of the differences, large misplacements of a single feature are penalized more than a few smaller ones.

Even when explaining the same model, SHAP and LIME agree to very different degrees across datasets, as shown in Table 5.2. On California Housing, they reach a 0.772 Spearman correlation, while on Wine Quality, they only manage 0.226. This suggests that we should not treat either of them individually as ground truth. A correlation close to how LIME is correlated with SHAP can be considered an acceptable level of performance for that dataset.

The results of this evaluation are presented and interpreted in the Results and Discussion chapters.

Table 5.2: Agreement between SHAP and LIME on the same XGBoost model, over the same 100 test instances. These values serve as a reference point against which LLM-generated rankings are judged.

Dataset	Spearman	Top-5 Overlap
California Housing	0.772	85.2%
Taiwan Bankruptcy	0.414	54.4%
Wine Quality	0.226	56.4%

6

Results

In this chapter, we showcase the quantitative comparison of agreement between LLM-generated feature importance scores and traditional post-hoc local explainers (SHAP and LIME) over three datasets for the metrics proposed in Section 5.4. Every setup was processed on the same 100 test instances for each dataset, and the metrics were computed per instance and averaged over all instances. Additionally, every experiment was repeated three times to account for the inherent stochasticity of LLMs. Unless stated otherwise, reported values are means over the three runs, and $\pm$ denotes the standard deviation across runs. Throughout, the agreement between SHAP and LIME themselves (Table 5.2) serves as the point of reference.

First, we present the findings of alignment based on top- k overlap. Then we compare ranking similarity according to Spearman correlation. Finally, we aggregate the per-instance averages a second time: first across the two LLMs to isolate the contribution of the prompt template, then across the five prompt templates to isolate the contribution of the LLM.

6.1. Alignment by top- k overlap

In this section, we examine how well the setups can align with SHAP and LIME on the most predictive features, regardless of their order within the top k . A random expectation baseline, representing the top- k overlap if features were chosen at random ($k/|\mathcal{F}|$), is also included in the graphs in this section, along with an alternative SHAP/LIME baseline. More precisely, when comparing the LLM’s alignment to SHAP the alternative baseline represents LIME’s alignment to SHAP. Conversely, when the LLM is evaluated against LIME, the alternative baseline is SHAP. This was included in the graph to add a realistic point of reference for the LLM’s performance. It shows how closely two established explainers of the same model agree, and therefore what level of agreement is attainable on each dataset. The values for $k = 1$ to 5 are listed in Appendix C.

6.1.1. California Housing

On California Housing, SHAP and LIME rank the same feature first in 68.0% of the instances and share, on average, 74.7% of their top-3 features (Figure 6.1).

Five of the six setups that include feature names reach this level of agreement on the top three features, but not on the top feature. The four text-driven setups and mixed neighbors with Gemma have a top-3 overlap between 74.7% and 76.4% with SHAP and between 77.4% and 79.3% with LIME, which matches or exceeds the 74.7% overlap between SHAP and LIME. Mixed neighbors with Gemma has the highest overlap of all setups at $k = 4$ and $k = 5$ against both references (81.5% and 84.5% against SHAP), close to the overlap between SHAP and LIME (84.8% and 85.2%). With GPT-OSS, the same template stays well below the other setups with feature names at $k = 2$ and $k = 3$ (59.3% against SHAP and 58.1% against LIME at $k = 3$). At $k = 1$, however, the setups with feature names rarely select the reference’s top feature. In 11 of the 12 comparisons, they do so less often than a random ranking would (12.5%). The text-driven setups select SHAP’s top feature in 8.0% to 9.3% of the instances and LIME’s in 0.0% to 1.3%. The only exception is mixed neighbors with Gemma against SHAP (13.3%).

The data-driven setups show the opposite pattern. PG-ICL with Gemma has the highest top-1 overlap of all setups (35.0% against SHAP, 38.0% against LIME), about half of the agreement between SHAP and LIME, but its top-3 overlap (45.9% and 52.0%) stays below that of every setup with feature names. The other three data-driven setups remain within 8.2 percentage points of the random expectation for all $k \leq 5$.

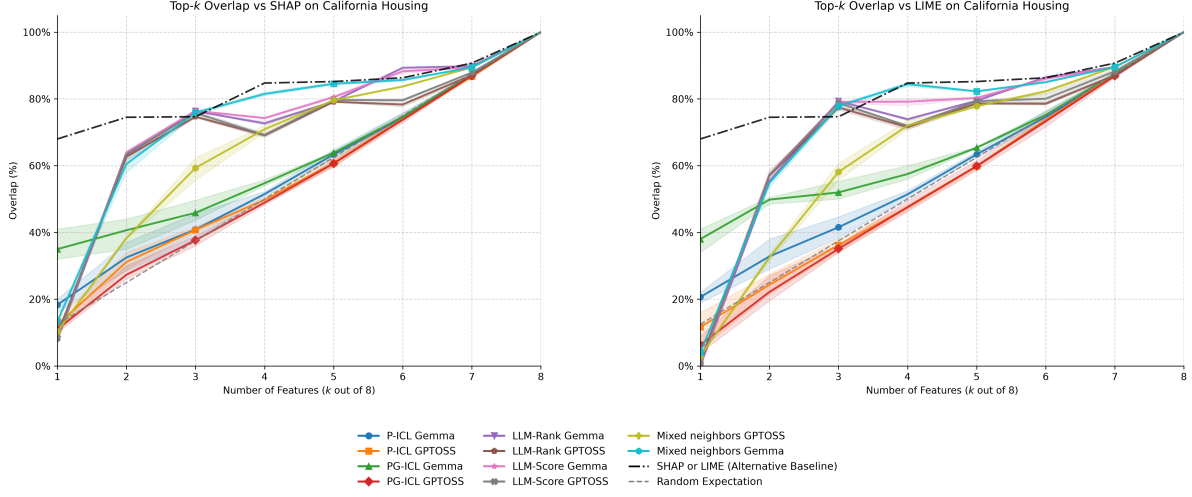


Figure 6.1: California Housing: top- k overlap between LLM-generated feature rankings and the reference rankings. Left: against SHAP. Right: against LIME. Solid lines show the mean over three runs, and shaded areas span the best- and worst-performing runs. The alternative baseline is the overlap between SHAP and LIME themselves (LIME in the left panel, SHAP in the right panel). The random expectation ($k/|\mathcal{F}|$) is the overlap expected from a random ranking.

6.1.2. Wine Quality

On Wine Quality, SHAP and LIME agree less. They rank the same feature first in 50.0% of the instances and share, on average, 54.7% of their top-3 features (Figure 6.2).

Against SHAP, eight of the ten setups exceed this top-3 overlap. The exceptions are P-ICL and PG-ICL with GPT-OSS (45.7% and 46.6%). The text-driven setups reach the highest values: LLM-Rank has a top-3 overlap of 91.0% with both LLMs, and LLM-Score with GPT-OSS 81.4%. For these three setups, the overlap peaks at $k = 3$ and decreases at $k = 4$ and $k = 5$, which produces the spike in the left panel of Figure 6.2. At $k = 1$, however, no setup matches SHAP's top feature as often as LIME does. The highest top-1 overlap with SHAP is 38.7% (LLM-Score with GPT-OSS), compared to 50.0% for LIME.

Against LIME, the setups stay closer to the alternative baseline than they do against SHAP for $k = 3$ to 5. At $k = 5$, their overlap ranges from 49.9% to 58.1%, compared to 56.4% for SHAP. LLM-Rank with GPT-OSS is the only setup that selects LIME's top feature more often than SHAP does (61.0% compared to 50.0%), and LLM-Score with GPT-OSS matches SHAP (50.0%), whereas LLM-Rank with Gemma does so in 24.0% of the instances. Against both references and for all $k \leq 5$, every setup stays above the random expectation.

6.1.3. Taiwan Bankruptcy Prediction

On Taiwan Bankruptcy Prediction, SHAP and LIME rank the same feature first in only 27.0% of the instances, compared to 68.0% and 50.0% on the other two datasets, and share, on average, 48.0% of their top-3 features (Figure 6.3).

From $k = 2$ to $k = 5$, every setup stays below the alternative baseline against both references. Within this range, the two data-driven setups with Gemma have the highest overlap at every k against both references. At $k = 3$, PG-ICL and P-ICL with Gemma reach 35.1% and 34.6% against SHAP and 40.3% and 39.6% against LIME, compared to 48.0% between SHAP and LIME. At $k = 1$, the same two setups select LIME's top feature more often than SHAP does (37.3% and 34.3%, compared to 27.0%) and SHAP's top feature nearly as often as LIME does (24.7% and 23.7%). Mixed neighbors with GPT-OSS has the highest top-1 overlap with SHAP (26.7%).

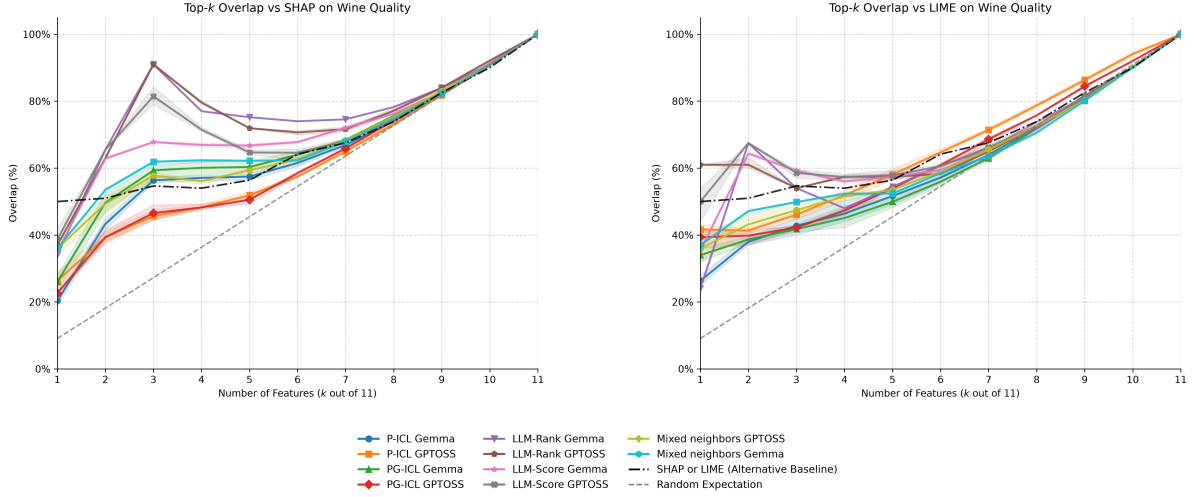


Figure 6.2: Wine Quality: top- k overlap between LLM-generated feature rankings and the reference rankings. Left: against SHAP. Right: against LIME. Solid lines show the mean over three runs, and shaded areas span the best- and worst-performing runs. The alternative baseline is the overlap between SHAP and LIME themselves (LIME in the left panel, SHAP in the right panel). The random expectation ($k/|\mathcal{F}|$) is the overlap expected from a random ranking.

The text-driven setups almost never select the reference’s top feature. In three of the four text-driven setups, the top-1 overlap is below 1% against SHAP and below the random expectation of 5% against LIME, and LLM-Rank with Gemma does not match LIME’s top feature in any instance at either $k = 1$ or $k = 2$. Only LLM-Score with GPT-OSS exceeds the random expectation at $k = 1$ (9.0% against SHAP, 12.0% against LIME). At $k = 4$ and $k = 5$, all text-driven setups exceed the random expectation against both references.

Mixed neighbors with Gemma varies strongly between runs (e.g., ± 9.2 percentage points at $k = 1$ against SHAP) and, against LIME, stays close to the random expectation for every $k \leq 5$ (7.7, 10.5, 14.3, 19.8, and 23.9%, compared to 5, 10, 15, 20, and 25%).

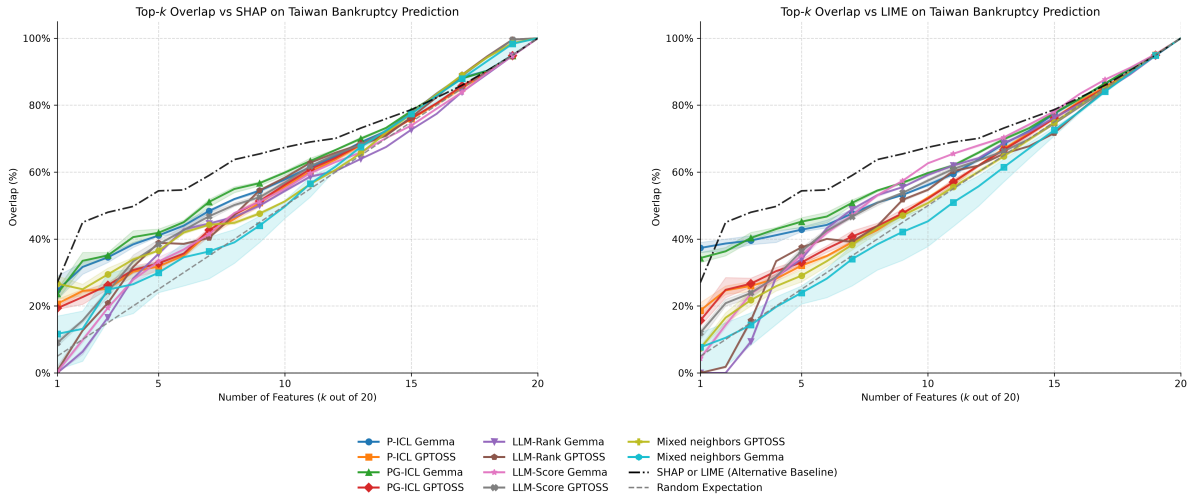


Figure 6.3: Taiwan Bankruptcy Prediction: top- k overlap between LLM-generated feature rankings and the reference rankings. Left: against SHAP. Right: against LIME. Solid lines show the mean over three runs, and shaded areas span the best- and worst-performing runs. The alternative baseline is the overlap between SHAP and LIME themselves (LIME in the left panel, SHAP in the right panel). The random expectation ($k/|\mathcal{F}|$) is the overlap expected from a random ranking.

6.2. Alignment by Spearman

This section compares the full rankings using Spearman correlation. Tables 6.1, 6.2 and 6.3 report the results per dataset, and the agreement between SHAP and LIME themselves is given in Table 5.2.

6.2.1. California Housing

No setup reaches the Spearman correlation between SHAP and LIME of 0.772 (Table 6.1). The setups form two clearly separated groups. The six setups that include feature names range from 0.499 to 0.693 against SHAP and from 0.473 to 0.692 against LIME, while the four data-driven setups range from -0.041 to 0.158 against SHAP and from -0.088 to 0.240 against LIME. The highest correlation is obtained by mixed neighbors with Gemma (0.693 against SHAP, 0.692 against LIME), although the three Gemma setups with feature names lie within 0.025 of each other against SHAP and within 0.038 against LIME.

Gemma correlates more strongly than GPT-OSS with both references for every template. Across both references, the Gemma setups with feature names range from 0.654 to 0.693 and the GPT-OSS setups with feature names from 0.473 to 0.584 . For each setup, the correlations with SHAP and with LIME differ by at most 0.082 . Standard deviations across runs are at most 0.022 for the setups with feature names and at most 0.062 for the data-driven setups.

Method	SHAP Spearman	LIME Spearman
Mixed Neighbors Gemma	0.693 ± 0.005	0.692 ± 0.004
LLM-Rank Gemma	0.676 ± 0.001	0.654 ± 0.001
LLM-Score Gemma	0.668 ± 0.003	0.679 ± 0.007
LLM-Score GPT-OSS	0.531 ± 0.006	0.584 ± 0.009
Mixed Neighbors GPT-OSS	0.502 ± 0.022	0.473 ± 0.021
LLM-Rank GPT-OSS	0.499 ± 0.005	0.553 ± 0.008
PGICL Gemma	0.158 ± 0.037	0.240 ± 0.040
PICL Gemma	0.052 ± 0.046	0.060 ± 0.040
PICL GPT-OSS	0.002 ± 0.018	-0.045 ± 0.032
PGICL GPT-OSS	-0.041 ± 0.032	-0.088 ± 0.062

Table 6.1: California Housing: Agreement between LLM-generated feature rankings and reference feature importance rankings measured by Spearman correlation. Results are reported as mean $\pm$ standard deviation over 3 runs.

6.2.2. Wine Quality

On Wine Quality, where SHAP and LIME correlate at only 0.226 , the results depend strongly on the reference (Table 6.2). Against SHAP, the correlations range from 0.148 to 0.645 , and eight of the ten setups exceed 0.226 . The four text-driven setups score highest (from 0.466 to 0.645), with LLM-Rank above 0.6 for both LLMs. Against LIME, the range is much narrower (from 0.081 to 0.331), and six of the ten setups exceed 0.226 . The highest correlation with LIME is obtained by P-ICL with GPT-OSS (0.331), which has the lowest correlation with SHAP (0.148). Conversely, LLM-Rank with Gemma, the highest against SHAP (0.645), ranks seventh against LIME (0.220).

The difference between the two LLMs depends on the template. For the two text-driven templates, Gemma and GPT-OSS differ by at most 0.033 against either reference, and for mixed neighbors by at most 0.043 . For the data-driven templates, the differences are larger and point in opposite directions. Gemma correlates more strongly with SHAP (by 0.168 for P-ICL and 0.190 for PG-ICL), whereas GPT-OSS correlates more strongly with LIME (by 0.215 for P-ICL and 0.135 for PG-ICL). Standard deviations across runs do not exceed 0.035 .

6.2.3. Taiwan Bankruptcy Prediction

No setup reaches the Spearman correlation between SHAP and LIME of 0.414 (Table 6.3). Against SHAP, the two data-driven setups with Gemma score highest (0.305 for PG-ICL and 0.273 for P-ICL), and the remaining setups range from 0.078 to 0.224 . Against LIME, PG-ICL with Gemma and LLM-Score with Gemma share the highest correlation (0.281), followed by P-ICL with Gemma (0.231), while mixed neighbors with Gemma has the lowest (-0.079).

Method	SHAP Spearman	LIME Spearman
LLM-Rank Gemma	0.645 $\pm$ 0.000	0.220 $\pm$ 0.000
LLM-Rank GPT-OSS	0.612 $\pm$ 0.006	0.241 $\pm$ 0.017
LLM-Score Gemma	0.493 $\pm$ 0.008	0.323 $\pm$ 0.005
LLM-Score GPT-OSS	0.466 $\pm$ 0.010	0.306 $\pm$ 0.010
Mixed Neighbors Gemma	0.415 $\pm$ 0.007	0.239 $\pm$ 0.003
PGICL Gemma	0.374 $\pm$ 0.019	0.081 $\pm$ 0.016
Mixed Neighbors GPT-OSS	0.372 $\pm$ 0.011	0.243 $\pm$ 0.007
PICL Gemma	0.316 $\pm$ 0.007	0.116 $\pm$ 0.035
PGICL GPT-OSS	0.184 $\pm$ 0.028	0.216 $\pm$ 0.014
PICL GPT-OSS	0.148 $\pm$ 0.004	0.331 $\pm$ 0.020

Table 6.2: Wine Quality: Agreement between LLM-generated feature rankings and reference feature importance rankings measured by Spearman correlation. Results are reported as mean $\pm$ standard deviation over 3 runs.

As on Wine Quality, individual setups rank differently depending on the reference. LLM-Score with Gemma has the lowest correlation with SHAP (0.078) but shares the highest with LIME (0.281), and LLM-Rank with GPT-OSS ranks fourth against SHAP (0.216) but eighth against LIME (0.072).

The difference between the LLMs again depends on the template. With the data-driven templates, Gemma correlates more strongly than GPT-OSS with both references (by 0.110 to 0.158). With the three templates that include feature names, GPT-OSS correlates more strongly with SHAP (between 0.171 and 0.224, compared to between 0.078 and 0.096 for Gemma). Against LIME, Gemma is ahead for both text-driven templates (0.173 and 0.281, compared to 0.072 and 0.157), and GPT-OSS is ahead for mixed neighbors (0.041, compared to -0.079).

Mixed neighbors with Gemma is the only setup across all three datasets with a standard deviation across runs above 0.1 (0.116 against SHAP, 0.114 against LIME). All other setups on this dataset stay at or below 0.019. Both LLMs also had more difficulty following the output format on this dataset, in some cases ignoring the formatting instructions or naming features that do not exist.

Method	SHAP Spearman	LIME Spearman
PGICL Gemma	0.305 $\pm$ 0.005	0.281 $\pm$ 0.010
PICL Gemma	0.273 $\pm$ 0.007	0.231 $\pm$ 0.015
LLM-Score GPT-OSS	0.224 $\pm$ 0.005	0.157 $\pm$ 0.010
LLM-Rank GPT-OSS	0.216 $\pm$ 0.004	0.072 $\pm$ 0.004
PGICL GPT-OSS	0.177 $\pm$ 0.019	0.123 $\pm$ 0.009
Mixed Neighbors GPT-OSS	0.171 $\pm$ 0.011	0.041 $\pm$ 0.012
PICL GPT-OSS	0.163 $\pm$ 0.017	0.115 $\pm$ 0.013
Mixed Neighbors Gemma	0.096 $\pm$ 0.116	-0.079 $\pm$ 0.114
LLM-Rank Gemma	0.090 $\pm$ 0.004	0.173 $\pm$ 0.011
LLM-Score Gemma	0.078 $\pm$ 0.010	0.281 $\pm$ 0.008

Table 6.3: Taiwan Bankruptcy Prediction: Agreement between LLM-generated feature rankings and reference feature importance rankings measured by Spearman correlation. Results are reported as mean $\pm$ standard deviation over 3 runs.

6.3. Aggregated Results

In this section, we show the aggregated results over all experimental runs for each dataset, first grouped by the prompt template used (Table 6.4) and then by the language model used (Table 6.5). This is done in an attempt to isolate the various independent factors and their individual contributions to the results.

As we can see in Table 6.4, on California Housing, text-driven and mixed neighbors produce similar results, clearly outperforming purely data-driven prompts, which, on average, report rankings close to random orderings. On the Wine Quality dataset, text-driven still leads, but the difference to data-driven is lower, especially against LIME. Mixed neighbors ranks somewhere in between the two families, suggesting that introducing neighboring samples does not help, instead probably confusing the model.

Template	California Housing		Wine Quality		Taiwan Bankruptcy	
	SHAP	LIME	SHAP	LIME	SHAP	LIME
LLM-Rank	0.588	0.604	0.629	0.231	0.153	0.123
LLM-Score	0.600	0.632	0.480	0.315	0.151	0.219
<i>Text-driven mean</i>	<i>0.594</i>	<i>0.618</i>	<i>0.555</i>	<i>0.273</i>	<i>0.152</i>	<i>0.171</i>
Mixed Neighbors	0.598	0.583	0.394	0.241	0.134	−0.019
P-ICL	0.027	0.008	0.232	0.224	0.218	0.173
PG-ICL	0.059	0.076	0.279	0.149	0.241	0.202
<i>Data-driven mean</i>	<i>0.043</i>	<i>0.042</i>	<i>0.256</i>	<i>0.187</i>	<i>0.230</i>	<i>0.188</i>

Table 6.4: Agreement between LLM-generated feature rankings and the reference feature importance rankings, by prompt template, measured by Spearman rank correlation. Each value is the mean over the two LLMs of the corresponding entries in Tables 6.1, 6.2 and 6.3. Rows are grouped by prompt family.

On the Taiwan dataset, the order is flipped. Data-driven methods lead against both LIME and SHAP when compared to text-driven methods. The worst performer is the mixed neighbors prompt, with a slightly worse than random performance against LIME, and a slightly worse performance than the text-driven mean against SHAP. This result leads us to believe that adding the feature names of this dataset to the prompt mostly degrades performance.

When comparing performance by LLM, we can see a large difference for both LIME and SHAP in Gemma’s favor on California Housing. On Wine Quality, Gemma is more aligned with SHAP, while GPT-OSS seems slightly more aligned with LIME. For the Taiwan dataset, the two models seem to have a similar level of agreement with SHAP, and Gemma is, on average, slightly more aligned with LIME. However, the similar agreement with SHAP on Taiwan comes from differences in opposite directions: GPT-OSS is ahead with the three templates that include feature names, and Gemma with the two data-driven templates (Table 6.3).

Model	California Housing		Wine Quality		Taiwan Bankruptcy	
	SHAP	LIME	SHAP	LIME	SHAP	LIME
Gemma	0.449	0.465	0.449	0.196	0.168	0.177
GPT-OSS	0.299	0.295	0.356	0.267	0.190	0.102
<i>Difference</i>	<i>0.151</i>	<i>0.170</i>	<i>0.092</i>	<i>−0.071</i>	<i>−0.022</i>	<i>0.076</i>

Table 6.5: Agreement between LLM-generated feature rankings and the reference feature importance rankings, by language model, measured by Spearman rank correlation. Each value is the mean over the five prompt templates of the corresponding entries in Tables 6.1, 6.2 and 6.3. The final row gives the difference between the two models.

Discussion

In this thesis, we compared five prompt templates: four from two existing approaches and a novel mixed neighbors template. This was done using two LLMs and three datasets with quantitative metrics.

LLM-generated rankings exceeded the agreement between SHAP and LIME on Wine Quality, came close to it on California Housing, and fell well short of it on Taiwan Bankruptcy Prediction. On Wine Quality, most setups exceeded the agreement between SHAP and LIME themselves (eight of ten against SHAP and six of ten against LIME), while on California Housing and Taiwan Bankruptcy Prediction, no setup reached this level. However, alignment varied strongly between datasets and setups. For example, one of the top-performing setups on one dataset (LLM-Rank Gemma) can vary from 0.676 on California Housing to 0.090 in Spearman correlation against SHAP on Taiwan Bankruptcy Prediction.

In this chapter, we identify four conditions that impact alignment across the experiments: agreement between SHAP and LIME, dataset characteristics (the dimensionality, the semantic content of its feature names, and its task type), the prompt template used, and the choice of LLM. Additionally, we present the limitations of this study.

7.1. Agreement between SHAP and LIME

Arguably, the agreement between SHAP and LIME on the selected datasets is very important, as it is the reference point on which LLMs are judged. For example, on California Housing, SHAP and LIME agree with each other at 0.772 Spearman, while on Wine Quality, the same two explainers only agree at 0.226 Spearman. If, on a dataset, SHAP and LIME have a high agreement rate, such as on California Housing, the scores against SHAP are very similar to the scores against LIME, with their differences not yielding additional information. On Wine Quality, where SHAP and LIME agree the least, the choice of reference changes which setups perform best: LLM-Rank Gemma achieved 0.645 against SHAP but only 0.220 against LIME, while P-ICL GPT-OSS showed the opposite pattern (0.148 against SHAP, 0.331 against LIME). This could suggest that data-driven prompts, which rely on perturbations like LIME, are closer to LIME, but the same prompts with Gemma aligned more with SHAP. Using only SHAP as the reference would have led to the conclusion that text-driven prompts work best on Wine Quality, whereas using only LIME would have ranked a data-driven setup (P-ICL GPT-OSS) highest. Consequently, on Wine Quality a low score against one reference does not necessarily indicate a poor ranking, as LLM-Rank Gemma agrees with LIME (0.220) about as much as SHAP itself does (0.226).

7.2. Dataset characteristics

The properties of the dataset are another condition that explains the differences in agreement among the setups. The three datasets used were chosen to represent a spectrum of possible characteristics, with a varying number of features (8, 11, and 20 for California Housing, Wine Quality, and Taiwan Bankruptcy Prediction, respectively), distinct types of tasks (regression, multi-class classification, and binary classification), and feature names of varying semantic complexity. Because the different types of prompts used rely on different types of information provided to the LLM (e.g., the data-centric prompts do not use the feature names while text-driven prompts rely mainly on them), they can also help isolate

certain conditions that influence alignment. We analyze the impact of each of them in the subsections below.

7.2.1. Dimensionality

Before performing the experiments, we expected the number of features to be inversely correlated with the agreement with SHAP and LIME, but the results do not show this as a general rule. While this is indeed the case for datasets with large numbers of features (during testing, it was found that prompts containing more than 25 features yield unstable results), this does not hold for the numbers of features used in the experiments. Looking at Table 6.4, we can see that data-centric performance for Taiwan Bankruptcy Prediction was similar to that for Wine Quality (mean Spearman correlation of 0.230 compared to 0.256 against SHAP, and 0.188 compared to 0.187 against LIME). Conversely, on California Housing, which has the fewest features, data-driven prompts perform the worst out of all datasets (≈ 0.04 for both SHAP and LIME). This suggests that, within this range, other dataset characteristics, such as the feature names and task type discussed below, affect agreement more than the number of features.

Text-driven prompts, in contrast, require the LLM to reason about the meaning of every feature, so a higher number of features could still affect them more. Their agreement does drop sharply on Taiwan Bankruptcy Prediction (a mean Spearman correlation of 0.152 against SHAP, compared to 0.594 on California Housing and 0.555 on Wine Quality). However, Taiwan Bankruptcy Prediction has both the most features and the most complex feature names, so this drop could be caused by either of them.

In summary, within the range used in our experiments, dimensionality did not lower agreement for data-driven prompts, while text-driven prompts did show lower agreement on the dataset with the most features, although this cannot be separated from the effect of its complex feature names.

7.2.2. Feature names

It is possible that the feature names themselves play an important role in the agreement rate. California Housing and Wine Quality have short, descriptive feature names that are easy to interpret (e.g., median income, latitude, alcohol, volatile acidity), while the features of Taiwan Bankruptcy Prediction are financial ratios, many of which represent complex formulas with long names. For text-driven prompts, the Spearman correlation against SHAP on Taiwan Bankruptcy Prediction dips to 0.152, compared to over 0.55 for both California Housing and Wine Quality. The top- k overlap results may contain an explanation for this (Table C.3). If we consider the most important feature identified by the LLMs using text-driven prompts on the Taiwan dataset, we can see that in 3 out of 4 setups, they score below 1% against SHAP, while the random expectation is 5%. We can also see the same pattern for California Housing, with the top feature choice having a low overlap rate with SHAP. Scoring below random means that the LLMs consistently choose a feature that SHAP almost never ranks first. This could suggest that the LLMs tend to select features that they think are relevant for predicting bankruptcy and home values in general, and in these cases, the rankings do not match what the ML model considers relevant. Not identifying the top feature correctly is an important error, as in many datasets a single feature has a much larger contribution to the prediction than the rest.

There is also another possible explanation for the drop. Many features of the Taiwan dataset are strongly correlated, and when correlated features carry the same information, SHAP can split the credit between them and lower the importance of each. The pruning procedure (Appendix B) selected features by predictive power and tree-based importance, but did not explicitly filter out correlated features. The tree-based criterion may favor one feature from a correlated group, but the Predictive Power Score evaluates each feature on its own, so correlated features may remain in the final set. This effect therefore remains a possible cause of the lower agreement.

Regarding the lower performance of the data-driven setups on California Housing, latitude and longitude are some of the most predictive features of the dataset. However, it is difficult for the LLM to infer that they are meaningfully connected when the LLM is unable to see their names.

Overall, feature names appear to help agreement when they are descriptive and the LLM's domain knowledge matches what the model relies on, as on California Housing and Wine Quality, but they seem to lower agreement when they are complex, as on Taiwan Bankruptcy Prediction. Meanwhile, hiding them can remove useful information, as seen for the data-driven prompts on California Housing.

7.2.3. Task type

The fact that California Housing is a regression task could also help explain why data-driven prompts performed much worse on this dataset than on the other two (a mean Spearman correlation of ≈ 0.04 against both SHAP and LIME, compared to around 0.19 to 0.26 on Wine Quality and Taiwan Bankruptcy Prediction). Since California Housing is the only regression task and its data-driven prompts also lack feature names, the two effects cannot be separated. Additionally, the paper that introduced this approach (Kroeger et al., 2024) only used binary classification datasets.

One possible cause of the low agreement of data-driven prompts on California Housing is the way the 16 perturbed samples were selected. The original procedure (Kroeger et al., 2024) stratifies samples by predicted class, which has no direct equivalent for a continuous output, so we adapted it by selecting samples across brackets of the predicted value. This adaptation may capture the model's local behavior less well than class stratification does, or regression could be inherently more difficult for this method (small changes in a continuous output may be harder to notice than a change in predicted class).

7.3. Prompt templates

The third condition is the information provided to the LLM through the prompt template. Text-driven prompts provide feature names, task context, and no other samples. Data-driven prompts provide anonymized perturbed samples, and the mixed neighbors prompt combines feature names with real neighboring samples. Comparing them shows how feature names and sample data contribute to alignment.

7.3.1. Text-driven prompts

Text-driven prompts achieved a mean Spearman correlation against SHAP of 0.594 on California Housing and 0.555 on Wine Quality, but only 0.152 on Taiwan Bankruptcy Prediction. As discussed in Section 7.2.2, these high scores likely come from the LLMs using the descriptive feature names and their domain knowledge to rank features that are generally relevant, which also explains why they often miss the most important feature for a specific prediction.

However, it seems that inserting feature names into the template results in a more global feature ranking. For example, on California Housing, all text-driven setups show very similar scores from the top-1 to top-5 overlaps, regardless of the LLM (Table C.1). Meanwhile, data-driven setups are more diverse and show a higher standard deviation. If the LLM mostly ranks the same feature as the most predictive, the top-1 overlap is dependent on whether the other explainer (SHAP or LIME) also agreed often. This is the case for Wine Quality, but not for the other two datasets. The text-driven templates were originally designed for global feature selection (Jeong et al., 2025), so the LLM cannot access any information about the sample's neighborhood or see other predictions.

Also, because these datasets are probably in both LLMs' training sets (this point is expanded upon in Section 7.5), it is not unlikely that the LLM has been trained on text describing which features have the most predictive power, supporting the argument from earlier.

7.3.2. Data-driven prompts

The data-driven prompts entirely avoid the issue that text-based prompts have, since the feature names are hidden and the LLM cannot rely on its domain knowledge. That could explain why all data-driven setups were able to outperform the text-driven prompts in selecting the top-1 most predictive feature for Taiwan Bankruptcy Prediction and California Housing, and also outperform the random baseline on Taiwan Bankruptcy Prediction (Tables C.1 and C.3). In spite of this better top-1 performance, the rest of the results are mixed: on California Housing, the overlap beyond the top feature stays close to the random baseline for most setups and the mean Spearman correlation is close to zero, while on Taiwan Bankruptcy Prediction, data-driven prompts achieve the highest Spearman correlation of all three prompt families. This could be caused by the LLM possibly trying to recall which features differ the most between samples. For the Taiwan dataset, the dimensionality is greater than the number of samples shown to the LLM, further increasing the ranking difficulty.

7.3.3. Mixed neighbors prompt

The mixed neighbors prompt was designed to show the LLM real data, similar to the target sample, in an effort to help it better anchor its rankings. However, due to the inclusion of the feature names, it has a similar performance to the text-driven setups, which suggests that the LLM relies mainly on the feature names and its domain knowledge, and uses the neighboring samples relatively little. The mixed neighbors setups seem to show a slight improvement over text-driven prompts for top-1 overlap, and perform especially well in Spearman rankings against SHAP and LIME on California Housing. They perform poorly on Taiwan Bankruptcy Prediction however, where the top-1 and top-2 overlaps show a high standard deviation. This could be caused by the feature names dominating the feature values. Overall, using both techniques in one prompt type does not seem to combine their strengths. A possible explanation is that the feature names give the LLM a strong and easy-to-use signal, while four neighboring samples that differ from the target in many features at once give only a weak signal about which features matter for the prediction, so the LLM falls back on its domain knowledge instead of correcting it with the data.

7.4. LLM Comparison

Overall, Gemma was able to achieve results more in line with both SHAP and LIME, especially for California Housing (Table 6.5). This is also the case for Wine Quality against SHAP, but the difference between the two LLMs is smaller. This finding could be partly related to the fact that Gemma is a newer model by almost 8 months. Additionally, according to its model card (Google DeepMind, 2026), Gemma uses a dense model architecture, meaning that it uses all of its 30.7 billion parameters for each inference. This is in contrast to GPT-OSS, which uses a Mixture-of-Experts (MoE) architecture that activates only 5.1 billion parameters for inference (OpenAI, 2025). These factors could make a significant difference for complex reasoning tasks such as feature contribution approximation. However, the difference in model size and training data makes it hard to isolate each variable's individual contribution.

Overall, the choice of LLM seems to matter less than the prompt template. The difference between the best- and worst-performing templates is much greater on average than the difference between the two LLMs, with an average of 0.351 for the templates (Table 6.4) compared to 0.097 for the LLMs (Table 6.5).

In practice, this means the prompt template choice is more important than the choice of LLM. Gemma gave the highest agreement in most settings and would be a reasonable default, while GPT-OSS was better against SHAP when long, complex feature names were included, as on Taiwan Bankruptcy Prediction, and against LIME on Wine Quality. Since only two LLMs were tested, this should be seen as a starting point rather than a general rule.

7.5. Limitations

Several limitations should be taken into account when interpreting these results, relating to the data given to the LLMs, the LLMs themselves, the model access each approach requires, and the way agreement was measured. On the data side, the methodology depends on preprocessing to make the features usable for the LLM, and the datasets are public and well known, so the LLMs may have seen data about them during training. On the model side, only two LLMs were tested, which limits how far the results generalize to other models. On the method side, the approaches differ in how much model access they require. Finally, the evaluation compares rankings against SHAP and LIME, which are approximations rather than a ground truth, and the use of rankings instead of importance scores loses information about how large the differences between features are.

1. The proposed methodology involves some data preprocessing, depending on the dataset. Datasets with categorical or encoded features, or features without meaningful names, first need to be pre-processed. This limits the number of applications of this methodology, and creating a more standardized preprocessing procedure could allow it to be applied to a wider range of datasets. This also introduces additional design choices that affect results and have to be considered on a case-by-case basis.
2. Data Contamination: all used datasets are publicly available, widely known, and studied. This could cause the text-driven and mixed neighbors results to be artificially improved due to useful

data (studies, analyses and other material related to the datasets) being included in the LLMs' training data. This is less of a concern for the data-driven settings: although they include the feature values of the instance being explained, the feature names are replaced by generic labels and the remaining samples are synthetic perturbations, which makes it harder, though not impossible, for the LLM to recognize the dataset and draw on prior knowledge about it. A next step would be to repeat the experiments on datasets the LLMs are unlikely to have seen, such as private datasets or datasets published after their training cutoff.

3. Additionally, only two LLMs were tested. They were chosen because both are open-weights, instruction-tuned models that follow formatting instructions well, and because they represent two model families with different architectures, a dense model (Gemma) and a Mixture-of-Experts model (GPT-OSS). Results might differ for other model families and providers. Although we accessed them through OpenRouter in our experiments, open-weights models also fit the setting described in Chapter 4, where the user has full access to the dataset but not to the predictive model, since they can be run on the user's own infrastructure without sharing the data with an external provider.
4. Model access requirements differ between the prompting approaches. Only the text-driven prompts require nothing beyond the prediction being explained. The data-driven prompts require the model's predictions on 10,000 perturbations per instance, more than the 5,000 used for LIME, and the mixed neighbors prompt requires predictions for additional training samples. Therefore, these approaches only partly fit the access-restricted setting that motivates this thesis, unless such predictions are already available, for example from logged model outputs.
5. SHAP and LIME are not the absolute ground truth when it comes to explaining model behavior. They approximate model behavior in different ways and can disagree with each other. This has to be taken into account when interpreting the results. Agreement with SHAP or LIME only shows that an LLM ranks features similarly to an existing explainer, not that its ranking is faithful to the model. As discussed in Section 5.4, a low score against one explainer can simply reflect the disagreement between the two.
6. Using rankings instead of feature scores leads to information loss. For example, if there is a high-importance feature ranked just one position above a low-importance feature, even if importance scores could have revealed the scale of this gap, the ranking compresses it down to the number of positions between features. For our results, this means equally important features swapping order is penalized as much as two features with very different importances.

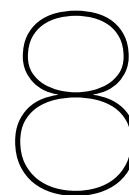
7.6. Societal implications

The approaches examined in this thesis could help users without direct access to an ML model or organizations using proprietary prediction models with no way to configure a traditional post-hoc explainer. The results show that LLM-based feature rankings agree with post-hoc explainers when domain knowledge aligns with the model behavior. The agreement seems to be a product of the LLMs' subject knowledge, rather than an analysis of a specific sample. This can be seen on California Housing and Taiwan Bankruptcy Prediction, where most text-centric prompts were not able to pick the most predictive feature more often than the random expectation.

However, the circumstances in which an LLM-driven explainer would most likely be used involve an ML model whose outputs are not easily understood and verified (e.g., due to a bias in training data). In these cases, the LLM would assume the ML model to be aligned with its domain knowledge, thus not solving the problem. LLM-generated explanations could serve the role of a quick estimate of which features are important in general.

The EU AI Act (European Parliament and Council of the European Union, 2024) classifies systems such as credit scoring of individuals as high-risk and gives persons affected by decisions based on such systems the right to a clear and meaningful explanation of the role of the AI system in the decision (Article 86). It also requires high-risk systems to be transparent enough for deployers to interpret their output and to allow effective human oversight. An LLM-generated ranking that mostly reflects domain knowledge would not meet this standard on its own, since it does not describe the role of the specific model.

Finally, the text-driven prompts used in this study contain the feature values of the dataset, and the mixed neighbors prompt also includes real samples from the training data. In domains with personal data, sending these prompts to an external LLM provider would mean sharing data about the person being explained as well as about other individuals, which raises privacy concerns under the GDPR. Running open-weights models such as Gemma and GPT-OSS locally avoids this problem. Overall, LLM-generated explanations could serve the role of a rougher, quick estimate of which features are important in general, but should not be the only explanation used in high-stakes settings.



Conclusion

Black-box models often require the use of post-hoc explainers such as SHAP or LIME to improve their explainability. The use of SHAP or LIME requires access to the model, which is often unavailable, due to a model being proprietary or a restrictive access interface, as well as some technical expertise for their setup and configuration. LLMs require no access to model internals and have shown that they are capable of completing reasoning tasks for which they were not specifically trained.

This study investigated the conditions under which LLM-generated feature importance rankings can align with post-hoc explainers such as SHAP and LIME. We performed the experiments with XGBoost models, on three datasets (California Housing, Wine Quality, and Taiwan Bankruptcy Prediction), using two open-weights LLMs (GPT-OSS 120B and Gemma 4 31B). Both LLMs were prompted using five different prompt templates: text-driven prompts (LLM-Score and LLM-Rank) that provide feature names and task context, data-driven prompts (P-ICL and PG-ICL) that provide anonymized perturbed samples, and a novel mixed neighbors strategy that includes feature names and similar neighboring samples.

The findings suggest that under specific settings, LLMs can get close to matching LIME and SHAP. In some instances, they were even able to match the level of agreement between SHAP and LIME themselves. For example, on California Housing, the mixed neighbors prompt with Gemma achieved a Spearman correlation of 0.693 against SHAP (while SHAP and LIME themselves had a correlation of 0.772). On Wine Quality, where SHAP and LIME were in slight agreement (0.226 Spearman correlation), eight of the ten setups tested surpassed this number against SHAP and six out of ten against LIME, with the best setup (LLM-Rank Gemma against SHAP) achieving a correlation of 0.645. Conversely, on the Taiwan dataset, no setup was able to match the SHAP-LIME agreement (0.414 Spearman correlation) in terms of Spearman correlation, with the closest being PG-ICL Gemma (0.305 correlation).

When analyzing the overlap scores, we can see that text-driven prompts identified most of the three most important features on California Housing and Wine Quality (up to 76% overlap with SHAP on California Housing and up to 91% on Wine Quality), and mixed neighbors did so with Gemma on California Housing. We can also see that, on California Housing and Taiwan Bankruptcy Prediction, most text-driven setups chose the top-ranked feature correctly less often than a random ordering. Data-driven prompts on Gemma showed the opposite pattern on these two datasets, being correct more often about the single most predictive feature (on Taiwan Bankruptcy Prediction about as often as SHAP and LIME agree with each other). In general, the overlap scores suggest that text-driven rankings, and to a lesser extent mixed neighbors, reflect which features the LLM considers domain-relevant, rather than which features influenced a specific prediction.

To answer the research question, LLM-generated feature importance rankings agree with SHAP and LIME mainly under three conditions. First, the dataset has few features with semantically meaningful names whose relationship to the target matches common domain knowledge; in this case, prompts containing feature names reach agreement close to, or even above, that between SHAP and LIME. Second, the information in the prompt suits the dataset: feature names help when they are descriptive and familiar, but appear to hurt when they describe complex quantities, as on Taiwan Bankruptcy Prediction, where anonymized data-driven prompts performed best. Third, the agreement between the reference explainers themselves matters: where SHAP and LIME disagree, the same ranking can appear well or poorly aligned depending on which one is used. The choice of LLM had a smaller effect

than the choice of prompt template. However, agreement does not imply that the LLM explains the individual prediction: text-driven rankings appear to vary little between instances, and on two of the three datasets most text-driven setups identify the single most important feature less often than a random ranking would.

8.1. Applicability

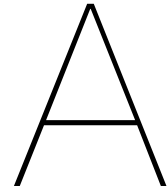
The results of the study show that LLMs can be used with some degree of success as post-hoc explainers in certain circumstances; more specifically, LLM-generated rankings best serve as a quick, rough estimate of which features are generally predictive, but are lacking when it comes to individual predictions. Looking at the findings, the applicability of this methodology is dependent on several characteristics of the dataset (dimensionality, semantic richness of features, task type), and on the type of information provided in the prompt. It also depends on the available model access, as only the text-driven prompts require nothing beyond the prediction being explained (Section 7.5).

The methodology is most applicable to datasets with few numerical features whose names are semantically meaningful and whose relationship to the target feature is well-understood, such as California Housing and Wine Quality. It performed poorly on Taiwan Bankruptcy Prediction, where prompts without feature names performed best but still agreed only weakly. However, this dataset has many features, strongly correlated features, and features representing complex formulas, so our experiments cannot separate the effect of each. For all datasets, the choice of prompt template matters more than the choice of LLM.

Finally, if a model deviates from domain knowledge, for instance due to biased training data, the LLM will likely still rank features according to its own priors (Section 7.6), and without a reference explainer this cannot be easily detected. Conversely, when model access is available, systematic disagreement between LLM-generated rankings and SHAP or LIME could flag models that rely on counter-intuitive features.

8.2. Future work

Future work can focus on an array of possible paths. For example, the methodology could be generalized to reduce the amount of preprocessing of the data and the design choices made for that purpose. Finding a more efficient way of delivering numerical data to LLMs could alleviate this issue. The methodology could also be applied to a wider array of LLMs and datasets to strengthen the generalizability of the findings. In particular, repeating the experiments on datasets the LLMs are unlikely to have seen would show how much of the text-driven agreement comes from data contamination, and using more datasets per condition would allow dimensionality, feature names, and task type to be varied independently. Additional prompt engineering techniques could be employed to make better use of LLMs' strengths and to mitigate their weaknesses. For now, LLMs cannot easily process large amounts of numerical data in their context, so if this aspect improves in the future, this method could see improved results.



Prompt Templates

A.1. In-Context Explainers

A.1.1. P-ICL

```
Context:
"We have a classification machine learning model that predicts
based on 11 features: ['A', 'B', 'C', 'D', 'E', 'F', 'G',
'H', 'I', 'J', 'K']. The dataset below contains the feature
values 'A' through 'K' and the corresponding model outputs."

Dataset:
Input: A: 9.236, B: 0.349, C: 0.182, ...
Output: 4

[15 perturbed samples omitted]

Question:
"Based on the above set, list all eleven features ranked from
the most important to the least important in driving the output
for the specific data sample provided below"

Instructions:
"Think about the question. After explaining your reasoning,
provide your answer as all eleven features ranked from most
important to least important, in descending order, separated by
commas for the specific data point given below.

Only provide the feature names on the last line. Do not provide
any further details on the last line."

Specific data sample:
Input: A: 6.800, B: 0.610, C: 0.040, D: 1.500,
E: 0.057, F: 5.000, G: 10.000, H: 0.995,
I: 3.420, J: 0.600, K: 9.500
Output: 2
```

Listing A.1: P-ICL prompt template, demonstrated on a sample from the Wine Quality dataset.

A.1.2. PG-ICL

```

Context:
"We have a classification machine learning model that predicts
based on 11 features: ['A', 'B', 'C', 'D', 'E', 'F', 'G',
'H', 'I', 'J', 'K']. The dataset below contains the feature
values 'A' through 'K' and the corresponding model outputs."

Dataset:
[16 [perturbed] samples omitted]

Question:
"Based on the above set, list all eleven features ranked from
the most important to the least important in driving the output
for the specific data sample provided below"

Instructions: "For each feature, starting with 'A and continuing to 'K:

1. Analyze the feature in question. Rate the importance of the feature in
determining the output on a scale of 0-100, considering both positive and
negative correlations. Ensure to give equal emphasis to both positive and
negative correlations and avoid focusing only on absolute values.
2. After analyzing the feature, position it in a running rank compared to the
features already analyzed. For instance, after analyzing feature 'B, determine
its relative importance compared to 'A and position it accordingly in the rank
(e.g., BA or AB). Continue this process until all features from 'A to 'K are
ranked. After explaining your reasoning, provide your answer as the final rank
of features from 'A to 'K from most important to least important, in
descending order, separated by commas. Only provide the feature names on the
last line. Do not provide any further details on the last line".

Specific data sample:
Input: A: 6.800, B: 0.610, C: 0.040, D: 1.500,
E: 0.057, F: 5.000, G: 10.000, H: 0.995,
I: 3.420, J: 0.600, K: 9.500
Output: 2

```

Listing A.2: PG-ICL prompt template, demonstrated on a sample from the Wine Quality dataset.

A.2. LLM-Select

A.2.1. LLM-Rank

```

We have trained an XGBoost model that predicts the quality of a wine. Your job is
to predict the feature coefficients of this model for a specific data point.
Given a list of features for a specific sample, rank them according to their
importances in predicting whether a wine is high or low quality. The ranking
should be in descending order, starting with the most important feature.
Your response should be a numbered list with each item on a new line. For example:
1. foo
2. bar
3. baz

Only output the ranking. Do not output dialogue or explanations for the ranking.
Do not exclude any features in the ranking.

Rank all 11 features in the following list: "fixed acidity", "volatile acidity",
"citric acid", "residual sugar", "chlorides", "free sulfur dioxide", "total
sulfur dioxide", "density", "pH", "sulphates", "alcohol" specifically for the
following sample:
fixed acidity volatile acidity citric acid residual sugar chlorides free
sulfur dioxide total sulfur dioxide density pH sulphates alcohol

```

Predicted quality						
6.8	0.61	0.04	1.5	0.057		
5.0	10.0	0.99525	3.42	0.6	9.5	
5						

Listing A.3: LLM-Rank prompt template, demonstrated on a sample from the Wine Quality dataset.

A.2.2. LLM-Score

We have trained an XGBoost model that predicts the quality of a wine. Your job is to predict the feature coefficients of this model for a specific data point. For each feature input of this specific data point by the user, your task is to provide a feature importance score (between 0 and 1; larger value indicates greater importance) for predicting whether a wine is high or low quality and a reasoning behind how the importance score was assigned.

The output should be formatted as a JSON array of instances that conform to the JSON schema below.

As an example, for the schema `"properties": "foo": {"title": "Foo", "description": "a list of strings", "type": "array", "items": {"type": "string", "required": ["foo"]}}` the object `"foo": ["bar", "baz"]` is a well-formatted instance of the schema. The object `"properties": "foo": ["bar", "baz"]` is not well-formatted. Here is the output schema: `'{"description": "Langchain Pydantic output parsing structure.", "properties": {"feature": {"title": "Feature", "description": "The name of the feature being scored", "type": "string"}, "reasoning": {"title": "Reasoning", "description": "Logical reasoning behind feature importance score", "type": "string"}, "score": {"title": "Score", "description": "Feature importance score", "type": "number"}}, "required": ["feature", "score"]}'`

Provide a score and reasoning for each feature `"fixed acidity", "volatile acidity", "citric acid", "residual sugar", "chlorides", "free sulfur dioxide", "total sulfur dioxide", "density", "pH", "sulphates", "alcohol"` formatted according to the output schema above for the following sample:

```
fixed acidity volatile acidity citric acid residual sugar chlorides free
sulfur dioxide total sulfur dioxide density pH sulphates alcohol
Predicted quality
6.8 0.61 0.04 1.5 0.057
5.0 10.0 0.99525 3.42 0.6 9.5
5
```

Listing A.4: LLM-Score prompt template, demonstrated on a sample from the Wine Quality dataset.

A.3. Mixed neighbors

We have trained an XGBoost model that predicts the quality of a wine, classifying it as a quality score based on its physicochemical properties.

The following are wine samples from the dataset, along with their model-predicted quality scores:

Input: fixed acidity: 5.900 volatile acidity: 0.460 citric acid: 0.000
residual sugar: 1.900 chlorides: 0.077 free sulfur dioxide: 25.000 total
sulfur dioxide: 44.000 density: 0.994 pH: 3.500 sulphates: 0.530 alcohol:
11.200
Predicted quality: 5

Input: fixed acidity: 6.800 volatile acidity: 0.670 citric acid: 0.000
residual sugar: 1.900 chlorides: 0.080 free sulfur dioxide: 22.000 total
sulfur dioxide: 39.000 density: 0.997 pH: 3.400 sulphates: 0.740 alcohol:
9.700

```
Predicted quality: 5

Input: fixed acidity: 6.700  volatile acidity: 0.670  citric acid: 0.020
       residual sugar: 1.900  chlorides: 0.061  free sulfur dioxide: 26.000  total
       sulfur dioxide: 42.000  density: 0.995  pH: 3.390  sulphates: 0.820  alcohol:
       10.900
Predicted quality: 6

Input: fixed acidity: 7.400  volatile acidity: 0.620  citric acid: 0.050
       residual sugar: 1.900  chlorides: 0.068  free sulfur dioxide: 24.000  total
       sulfur dioxide: 42.000  density: 0.996  pH: 3.420  sulphates: 0.570  alcohol:
       11.500
Predicted quality: 6

Your task is to rank all 11 features from most to least important in explaining
why the model specifically predicted a quality of 5 for the wine sample below.

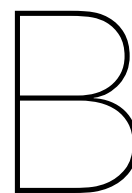
Input: fixed acidity: 7.000  volatile acidity: 0.560  citric acid: 0.130
       residual sugar: 1.600  chlorides: 0.077  free sulfur dioxide: 25.000  total
       sulfur dioxide: 42.000  density: 0.996  pH: 3.340  sulphates: 0.590  alcohol:
       9.200
Predicted quality: 5

Think step by step. Compare the target sample's feature values against the
examples above. Consider which features, given their specific values in the
target sample, most strongly drive the prediction of 5 - including features
that push toward this prediction and features that do not.

Rank all 11 features: "fixed acidity", "volatile acidity", "citric acid",
"residual sugar", "chlorides", "free sulfur dioxide", "total sulfur dioxide",
"density", "pH", "sulphates", "alcohol"

After explaining your reasoning, provide your final answer on the last line only
as all features separated by commas, ordered from most important to least
important. Do not include anything else on the last line.
```

Listing A.5: Mixed Neighbors prompt template, demonstrated on a sample from the Wine Quality dataset.



Dataset Analysis

The Taiwan Bankruptcy Prediction dataset originally has 95 features, which were reduced to 20 using a feature selection procedure adapted from a publicly available Kaggle notebook¹. All features in this dataset represent numeric financial ratios, except for two features. One of these, Net Income Flag (it indicates if a company's net income was negative for the last 2 years), has the same value across all samples and was removed. The other is Liability-Assets Flag (indicates whether a company's liabilities exceed its assets). The data was then split into a stratified 80/20 train/test partition, and feature selection was performed on the training set, using two criteria.

The first used criterion was the Predictive Power Score (PPS), a model-based measure of how well a single feature predicts the target class. PPS works by fitting a decision tree that predicts the class from one feature, computes its cross-validated weighted F1 score, and normalizes this score against a naive baseline that always predicts the majority class. Therefore, a score of 0 means the feature adds nothing beyond the baseline, and a score of 1 means perfect prediction. Unlike correlation, PPS can capture non-linear and non-monotonic relationships. Only seven features obtained a PPS above zero, so all seven were retained.

The second criterion was multivariate: a decision tree classifier was trained on all training features, with its hyperparameters tuned by randomized search under 5-fold cross-validation using F1 as the objective, and the 13 features with the highest impurity-based importance in the best tree were selected. The number of tree-based features is a fixed choice from the source notebook. The two sets did not overlap, and their union gave the final set of 20 features. Nineteen of these are continuous ratios and one is the binary Liability-Assets Flag.

¹<https://www.kaggle.com/code/kuralamuthan300/bankruptcy-prediction-xgb-f1-0-84>

C

Top- k Overlap Results

This appendix reports the exact top- k overlap values for $k = 1$ to 5 that underlie Figures 6.1, 6.2, and 6.3, with one table per dataset. Each value is the mean overlap (in %) over the same 100 test instances, averaged over three runs, with $\pm$ denoting the standard deviation across runs. Each table also includes two reference rows: the alternative baseline, which gives the overlap between SHAP and LIME themselves, and the random expectation ($k/|\mathcal{F}|$), which gives the overlap expected from a random ranking.

Method	vs. SHAP					vs. LIME				
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
P-ICL Gemma	18.3 $\pm$ 2.1	32.5 $\pm$ 4.3	40.9 $\pm$ 4.2	51.5 $\pm$ 0.8	63.3 $\pm$ 1.4	20.7 $\pm$ 1.5	32.8 $\pm$ 4.6	41.6 $\pm$ 3.2	51.4 $\pm$ 1.1	63.4 $\pm$ 0.9
P-ICL GPTOSS	12.3 $\pm$ 1.5	31.2 $\pm$ 2.6	40.8 $\pm$ 0.8	49.8 $\pm$ 0.9	60.8 $\pm$ 1.7	11.7 $\pm$ 5.9	24.3 $\pm$ 3.0	36.2 $\pm$ 1.8	47.5 $\pm$ 0.7	59.9 $\pm$ 0.6
PG-ICL Gemma	35.0 $\pm$ 5.2	40.7 $\pm$ 4.9	45.9 $\pm$ 3.3	54.8 $\pm$ 0.9	63.9 $\pm$ 0.6	38.0 $\pm$ 3.6	49.8 $\pm$ 1.2	52.0 $\pm$ 2.9	57.5 $\pm$ 2.2	65.4 $\pm$ 0.2
PG-ICL GPTOSS	11.0 $\pm$ 1.7	27.3 $\pm$ 2.3	37.7 $\pm$ 1.5	48.9 $\pm$ 0.6	60.7 $\pm$ 0.9	6.3 $\pm$ 2.5	22.2 $\pm$ 4.2	35.1 $\pm$ 1.2	47.6 $\pm$ 1.7	59.9 $\pm$ 1.6
LLM-Rank Gemma	8.0 $\pm$ 0.0	63.2 $\pm$ 0.6	76.3 $\pm$ 0.0	72.7 $\pm$ 0.5	79.3 $\pm$ 0.1	0.0 $\pm$ 0.0	55.3 $\pm$ 0.3	79.3 $\pm$ 0.0	73.9 $\pm$ 0.4	79.5 $\pm$ 0.1
LLM-Rank GPTOSS	9.3 $\pm$ 1.2	62.7 $\pm$ 1.0	74.7 $\pm$ 0.3	69.1 $\pm$ 0.5	79.1 $\pm$ 0.2	1.3 $\pm$ 1.2	57.0 $\pm$ 1.3	77.4 $\pm$ 0.7	71.5 $\pm$ 0.8	78.7 $\pm$ 0.3
LLM-Score Gemma	8.0 $\pm$ 0.0	64.0 $\pm$ 0.5	76.4 $\pm$ 0.2	74.2 $\pm$ 0.3	80.5 $\pm$ 0.4	0.0 $\pm$ 0.0	57.3 $\pm$ 0.3	79.1 $\pm$ 0.4	79.2 $\pm$ 1.0	80.3 $\pm$ 0.3
LLM-Score GPTOSS	8.3 $\pm$ 1.5	63.5 $\pm$ 0.5	75.8 $\pm$ 0.5	69.2 $\pm$ 0.5	79.7 $\pm$ 0.1	0.7 $\pm$ 1.2	57.2 $\pm$ 0.6	78.8 $\pm$ 0.5	71.9 $\pm$ 0.1	79.3 $\pm$ 0.1
Mixed neighbors GPTOSS	10.3 $\pm$ 1.2	38.3 $\pm$ 0.3	59.3 $\pm$ 3.4	70.9 $\pm$ 1.0	79.6 $\pm$ 0.2	3.0 $\pm$ 0.0	32.5 $\pm$ 1.0	58.1 $\pm$ 2.5	72.0 $\pm$ 1.3	77.8 $\pm$ 0.2
Mixed neighbors Gemma	13.3 $\pm$ 0.6	60.5 $\pm$ 2.2	75.9 $\pm$ 0.5	81.5 $\pm$ 0.5	84.5 $\pm$ 0.4	4.0 $\pm$ 0.0	54.8 $\pm$ 1.2	77.9 $\pm$ 0.5	84.4 $\pm$ 0.6	82.3 $\pm$ 0.2
Alt. baseline (LIME / SHAP)	68.0	74.5	74.7	84.8	85.2	68.0	74.5	74.7	84.8	85.2
Random expectation	12.5	25.0	37.5	50.0	62.5	12.5	25.0	37.5	50.0	62.5

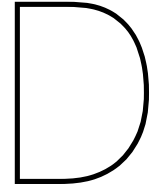
Table C.1: California Housing (8 features): Top- k overlap (%) with SHAP and LIME. Mean over 100 instances $\pm$ the standard deviation over three runs; best per column in bold. The alternative baseline is LIME under the SHAP columns and SHAP under the LIME columns.

Method	vs. SHAP					vs. LIME				
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
P-ICL Gemma	20.3 $\pm$ 1.2	43.3 $\pm$ 1.8	56.3 $\pm$ 0.9	57.1 $\pm$ 0.8	57.5 $\pm$ 1.0	26.3 $\pm$ 1.5	38.0 $\pm$ 1.3	42.8 $\pm$ 2.5	46.4 $\pm$ 3.4	51.7 $\pm$ 2.6
P-ICL GPTOSS	26.3 $\pm$ 3.1	39.3 $\pm$ 1.8	45.7 $\pm$ 1.5	48.2 $\pm$ 0.9	51.9 $\pm$ 1.0	41.7 $\pm$ 1.5	41.3 $\pm$ 0.8	46.1 $\pm$ 1.3	51.9 $\pm$ 1.5	58.1 $\pm$ 1.7
PG-ICL Gemma	26.0 $\pm$ 2.6	49.7 $\pm$ 2.8	59.3 $\pm$ 2.0	60.1 $\pm$ 2.0	60.3 $\pm$ 1.6	34.0 $\pm$ 2.0	38.7 $\pm$ 1.0	41.8 $\pm$ 1.3	45.1 $\pm$ 3.3	49.9 $\pm$ 1.6
PG-ICL GPTOSS	22.7 $\pm$ 0.6	39.3 $\pm$ 1.9	46.6 $\pm$ 2.2	48.2 $\pm$ 0.9	50.5 $\pm$ 1.0	39.3 $\pm$ 3.1	39.8 $\pm$ 2.5	42.2 $\pm$ 0.5	47.3 $\pm$ 1.2	54.0 $\pm$ 0.5
LLM-Rank Gemma	34.0 $\pm$ 0.0	65.5 $\pm$ 0.0	91.0 $\pm$ 0.0	77.0 $\pm$ 0.0	75.2 $\pm$ 0.0	24.0 $\pm$ 0.0	67.5 $\pm$ 0.0	54.0 $\pm$ 0.0	48.0 $\pm$ 0.0	54.4 $\pm$ 0.0
LLM-Rank GPTOSS	37.0 $\pm$ 0.0	63.0 $\pm$ 0.5	91.0 $\pm$ 0.0	79.6 $\pm$ 0.6	71.9 $\pm$ 0.1	61.0 $\pm$ 0.0	61.0 $\pm$ 1.7	54.0 $\pm$ 0.0	57.4 $\pm$ 0.5	57.4 $\pm$ 0.9
LLM-Score Gemma	35.0 $\pm$ 0.0	62.8 $\pm$ 0.6	67.8 $\pm$ 0.2	66.9 $\pm$ 0.6	66.7 $\pm$ 0.5	35.3 $\pm$ 0.6	64.3 $\pm$ 0.6	59.2 $\pm$ 0.4	56.0 $\pm$ 0.7	57.5 $\pm$ 0.6
LLM-Score GPTOSS	38.7 $\pm$ 3.5	65.5 $\pm$ 0.0	81.4 $\pm$ 2.7	71.5 $\pm$ 0.7	64.7 $\pm$ 0.7	50.0 $\pm$ 5.3	67.5 $\pm$ 0.0	58.6 $\pm$ 1.4	57.3 $\pm$ 1.0	57.9 $\pm$ 1.5
Mixed neighbors GPTOSS	36.0 $\pm$ 2.0	49.3 $\pm$ 2.6	57.8 $\pm$ 1.0	56.1 $\pm$ 0.7	59.3 $\pm$ 0.8	36.0 $\pm$ 4.0	43.2 $\pm$ 2.5	47.4 $\pm$ 2.3	51.5 $\pm$ 1.1	53.5 $\pm$ 0.1
Mixed neighbors Gemma	36.0 $\pm$ 0.0	53.5 $\pm$ 0.5	61.9 $\pm$ 0.8	62.3 $\pm$ 0.5	62.2 $\pm$ 0.2	37.0 $\pm$ 0.0	47.2 $\pm$ 0.6	49.9 $\pm$ 0.2	52.3 $\pm$ 0.5	52.4 $\pm$ 0.2
Alt. baseline (LIME / SHAP)	50.0	51.0	54.7	54.0	56.4	50.0	51.0	54.7	54.0	56.4
Random expectation	9.1	18.2	27.3	36.4	45.5	9.1	18.2	27.3	36.4	45.5

Table C.2: Wine Quality (11 features): Top- k overlap (%) with SHAP and LIME. Mean over 100 instances $\pm$ the standard deviation over three runs; best per column in bold. The alternative baseline is LIME under the SHAP columns and SHAP under the LIME columns.

Method	vs. SHAP					vs. LIME				
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
P-ICL Gemma	24.7 $\pm$ 3.8	31.7 $\pm$ 1.9	34.6 $\pm$ 1.4	38.4 $\pm$ 0.8	41.1 $\pm$ 0.2	37.3 $\pm$ 1.5	38.7 $\pm$ 1.9	39.6 $\pm$ 1.3	41.2 $\pm$ 1.9	42.8 $\pm$ 0.7
P-ICL GPTOSS	20.7 $\pm$ 1.5	24.5 $\pm$ 0.5	25.1 $\pm$ 0.2	30.2 $\pm$ 1.5	31.8 $\pm$ 1.4	18.7 $\pm$ 2.1	24.7 $\pm$ 1.3	26.0 $\pm$ 1.5	28.2 $\pm$ 1.2	32.1 $\pm$ 1.2
PG-ICL Gemma	23.7 $\pm$ 1.5	33.5 $\pm$ 2.3	35.1 $\pm$ 0.8	40.6 $\pm$ 1.8	41.9 $\pm$ 1.0	34.3 $\pm$ 1.5	36.3 $\pm$ 1.3	40.3 $\pm$ 1.5	43.0 $\pm$ 0.5	45.2 $\pm$ 1.6
PG-ICL GPTOSS	19.3 $\pm$ 0.6	22.7 $\pm$ 2.0	26.2 $\pm$ 1.1	30.7 $\pm$ 1.1	32.7 $\pm$ 0.6	15.7 $\pm$ 2.1	24.8 $\pm$ 3.2	26.7 $\pm$ 1.5	30.4 $\pm$ 1.1	33.0 $\pm$ 0.9
LLM-Rank Gemma	0.0 $\pm$ 0.0	6.3 $\pm$ 0.8	16.6 $\pm$ 0.2	28.2 $\pm$ 0.0	35.6 $\pm$ 0.4	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	9.3 $\pm$ 0.7	28.7 $\pm$ 0.5	34.0 $\pm$ 0.2
LLM-Rank GPTOSS	0.7 $\pm$ 1.2	12.7 $\pm$ 0.3	20.9 $\pm$ 0.4	31.7 $\pm$ 0.3	38.9 $\pm$ 0.1	0.0 $\pm$ 0.0	1.8 $\pm$ 0.3	15.7 $\pm$ 0.7	33.4 $\pm$ 0.1	37.5 $\pm$ 0.1
LLM-Score Gemma	0.3 $\pm$ 0.6	9.8 $\pm$ 0.8	19.4 $\pm$ 0.2	27.8 $\pm$ 0.9	33.3 $\pm$ 1.8	4.3 $\pm$ 0.6	14.3 $\pm$ 1.0	23.7 $\pm$ 0.9	28.6 $\pm$ 1.3	35.1 $\pm$ 1.0
LLM-Score GPTOSS	9.0 $\pm$ 1.0	15.7 $\pm$ 0.6	24.3 $\pm$ 1.5	33.3 $\pm$ 1.7	38.4 $\pm$ 1.2	12.0 $\pm$ 1.7	20.8 $\pm$ 1.0	23.9 $\pm$ 1.3	28.8 $\pm$ 0.9	36.4 $\pm$ 0.9
Mixed neighbors GPTOSS	26.7 $\pm$ 4.0	25.0 $\pm$ 1.8	29.4 $\pm$ 2.7	33.9 $\pm$ 1.0	36.5 $\pm$ 0.2	7.3 $\pm$ 0.6	16.5 $\pm$ 1.5	21.8 $\pm$ 2.3	25.9 $\pm$ 1.5	29.1 $\pm$ 1.8
Mixed neighbors Gemma	11.7 $\pm$ 9.2	13.2 $\pm$ 8.4	24.9 $\pm$ 8.0	26.5 $\pm$ 7.6	29.9 $\pm$ 5.1	7.7 $\pm$ 6.7	10.5 $\pm$ 7.4	14.3 $\pm$ 5.2	19.8 $\pm$ 4.6	23.9 $\pm$ 2.9
Alt. baseline (LIME / SHAP)	27.0	45.0	48.0	49.8	54.4	27.0	45.0	48.0	49.8	54.4
Random expectation	5.0	10.0	15.0	20.0	25.0	5.0	10.0	15.0	20.0	25.0

Table C.3: Taiwan Bankruptcy Prediction (20 features): Top- k overlap (%) with SHAP and LIME. Mean over 100 instances $\pm$ the standard deviation over three runs; best per column in bold. The alternative baseline is LIME under the SHAP columns and SHAP under the LIME columns.



Use of AI

Generative AI tools were used in an assistive capacity during this thesis. The author takes full responsibility for all content, design decisions, code, and results presented in this work.

Tools used. The following assistants were used: Claude (Anthropic), specifically Claude Sonnet 4.6, Claude Sonnet 5, and Claude Opus 5.5; Gemini 3.1 Pro (Google); and ChatGPT (OpenAI).

Tasks. The tools were used for the following:

- *Writing.* Improving the language of drafts written by the author, including grammar, phrasing, and clarity. Suggestions were edited or rewritten by the author before being included.
- *Argument structure.* Reviewing the flow and structure of the arguments in drafts. The tools pointed out unclear transitions or gaps. The author decided which changes to make and wrote them.
- *Experiment code.* Speeding up the writing of code for the experimental pipeline, under the author's instructions and supervision. All code was run and tested by the author, and all reported results were produced by running this code.
- *Tables and figures.* Writing $\text{\LaTeX}$ code for the result tables and Python code for the plots, such as the top- k overlap figures in Chapter 6.

Bibliography

- Abusitta, A., Li, M. Q., & Fung, B. C. (2024). Survey on explainable ai: Techniques, challenges and open issues. *Expert Syst. Appl.*, 255(PC). <https://doi.org/10.1016/j.eswa.2024.124710>
- Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6, 52138–52160. <https://doi.org/10.1109/ACCESS.2018.2870052>
- Arrieta, A. B., Díaz-Rodríguez, N., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2019). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. <https://arxiv.org/abs/1910.10045>
- Bhattacharjee, A., Moraffah, R., Garland, J., & Liu, H. (2024). Towards llm-guided causal explainability for black-box text classifiers. <https://arxiv.org/abs/2309.13340>
- Bilal, A., Ebert, D., & Lin, B. (2025, March). LLMs for Explainable AI: A Comprehensive Survey. <https://doi.org/10.48550/arXiv.2504.00125>
- Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., & Kasneci, G. (2022). Deep neural networks and tabular data: A survey. <https://arxiv.org/abs/2110.01889>
- Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. <https://doi.org/10.1145/2939672.2939785>
- Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009a). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4), 547–553. <https://doi.org/10.1016/j.dss.2009.05.016>
- Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009b). *Wine quality*. UCI Machine Learning Repository. <https://doi.org/10.24432/C56S3T>
- European Parliament and Council of the European Union. (2024). Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). <https://www.eur-lex.europa.eu/eli/reg/2024/1689>
- Google DeepMind. (2026). Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4
- Gramegna, A., & Giudici, P. (2021). Shap and lime: An evaluation of discriminative power in credit risk. *Frontiers in Artificial Intelligence*, 4. <https://doi.org/10.3389/frai.2021.752558>
- Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on tabular data? <https://arxiv.org/abs/2207.08815>
- Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Pedreschi, D., & Giannotti, F. (2018). A survey of methods for explaining black box models. <https://arxiv.org/abs/1802.01933>
- Jeong, D. P., Lipton, Z. C., & Ravikumar, P. (2025). Llm-select: Feature selection with large language models. <https://arxiv.org/abs/2407.02694>
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 1–38. <https://doi.org/10.1145/3571730>
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3149–3157.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2023). Large language models are zero-shot reasoners. <https://arxiv.org/abs/2205.11916>
- Krishna, S., Han, T., Gu, A., Wu, S., Jabbari, S., & Lakkaraju, H. (2025). The disagreement problem in explainable machine learning: A practitioner’s perspective. <https://arxiv.org/abs/2202.01602>
- Kroeger, N., Ley, D., Krishna, S., Agarwal, C., & Lakkaraju, H. (2024). In-context explainers: Harnessing llms for explaining black box models. <https://arxiv.org/abs/2310.05797>

- Lambin, X., & Raizonville, A. (2025). From black box to glass box: Algorithmic explainability as a strategic decision. *Information Economics and Policy*, 71, 101149. <https://doi.org/10.1016/j.infoecopol.2025.101149>
- Li, D., Tan, Z., & Liu, H. (2024). Exploring large language models for feature selection: A data-centric perspective. <https://arxiv.org/abs/2408.12025>
- Liang, D., Lu, C.-C., Tsai, C.-F., & Shih, G.-A. (2016). Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study. *European Journal of Operational Research*, 252(2), 561–572. <https://doi.org/10.1016/j.ejor.2016.01.012>
- Lukassen, F., Herrmann, J., Weisser, C., Saefken, B., & Kneib, T. (2026). From xai to stories: A factorial study of llm-generated explanation quality. <https://arxiv.org/abs/2601.02224>
- Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S.-I. (2019). Explainable ai for trees: From local explanations to global understanding. <https://arxiv.org/abs/1905.04610>
- Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. <https://arxiv.org/abs/1705.07874>
- Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., & Gao, J. (2025). Large language models: A survey. <https://arxiv.org/abs/2402.06196>
- Naudot, F., Sundqvist, T., & Kampik, T. (2025). Llmshap: A principled approach to llm explainability. <https://arxiv.org/abs/2511.01311>
- OpenAI. (2025). gpt-oss-120b & gpt-oss-20b model card. <https://arxiv.org/abs/2508.10925>
- Pace, R. K., & Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3), 291–297. [https://doi.org/10.1016/S0167-7152\(96\)00140-X](https://doi.org/10.1016/S0167-7152(96)00140-X)
- Paraschou, E., Arapakis, I., Yfantidou, S., Macaluso, S., & Vakali, A. (2026). Mind the XAI Gap: A Human-Centered LLM Framework for Democratizing Explainable AI. In R. Guidotti, U. Schmid, & L. Longo (Eds.), *Explainable Artificial Intelligence* (pp. 113–137). Springer Nature Switzerland. https://doi.org/10.1007/978-3-032-08333-3_6
- Pugliese, R., Regondi, S., & Marini, R. (2021). Machine learning-based approach: Global trends, research directions, and regulatory standpoints. *Data Science and Management*, 4, 19–29. <https://doi.org/10.1016/j.dsm.2021.12.002>
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why Should I Trust You?”: Explaining the predictions of any classifier. <https://arxiv.org/abs/1602.04938>
- Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. <https://arxiv.org/abs/1811.10154>
- Sarkar, A. (2022). Is explainable ai a race against model complexity? <https://arxiv.org/abs/2205.10119>
- Shwartz-Ziv, R., & Armon, A. (2021). Tabular data: Deep learning is not all you need. <https://arxiv.org/abs/2106.03253>
- Taiwanese bankruptcy prediction*. (2020). UCI Machine Learning Repository. <https://doi.org/10.24432/C5004D>
- Varshney, K. R., & Alemzadeh, H. (2017). On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. <https://arxiv.org/abs/1610.01256>
- Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., & Fedus, W. (2022). Emergent abilities of large language models. <https://arxiv.org/abs/2206.07682>
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., ... Wen, J.-R. (2025). A survey of large language models. <https://arxiv.org/abs/2303.18223>
- Zhou, J., Gandomi, A. H., Chen, F., & Holzinger, A. (2021). Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, 10(5). <https://doi.org/10.3390/electronics10050593>
- Zytek, A., Pidò, S., Alnegheimish, S., Berti-Équille, L., & Veeramachaneni, K. (2024). Explingo: Explaining ai predictions using large language models. *2024 IEEE International Conference on Big Data (BigData)*, 1197–1208. <https://doi.org/10.1109/BigData62323.2024.10825114>
- Zytek, A., Pidò, S., & Veeramachaneni, K. (2024, May). LLMs for XAI: Future Directions for Explaining Explanations. <https://doi.org/10.48550/arXiv.2405.06064>