INT$_2$X: EXPLANATION FOR THE CAUSES OF A PREDICTION

23-142

Project Final Report

Warnasooriya.S. D

B.Sc. (Hons) Degree in Information Technology

(Specialized in Data Science)

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

September 2023

DECLARATION

We declare that this is our own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or institute of higher learning, and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

DECLARATION

Name	Student ID	Signature
Warnasooriya S. D	IT20097660	[illegible]

The above candidate is carrying out research for the undergraduate Dissertation under my supervision.

…………………………..   …………………………..

Signature of the Supervisor: Date

ABSTRACT

With the advancement of machine learning technologies, most of the decision-making processes are being automated and AI-based in recent years. When the topic of AI-based processes becomes more popular communities began to question the explainability and interpretability of those decision-making processes. The explainability problem of AI-based decisions was more pronounced after the European Union imposed General Data Protection Regulations (GDPR) in 2016. GDPR mentions that there is a right for consumers to know about the internal processes that use to make predictions by the black box models. As a solution for the explainability problem, the concept of Model Interpretability is being raised as a hot topic and it is the next step of AI. Even though the topic has become popular, the number of studies that have been done related to that area is less. In this research, the main objective is to provide a novel XAI solution to enhance the model interpretability of black box models by developing a novel counterfactual rule generation mechanism related to the text classification domain. The proposed method will be specifically targeted the function-based classification models focusing on Support Vector Machine (SVM). SVM is a very popular classifier that performs well with both linear and non-linear data. However, the support vector machine is a black-box model because it is very difficult to explain the model predictions and challenging to understand the role of each input feature in the decision-making process. The black box behavior of SVM becomes more pronounced when SVM use to classify non-linear separable data by mapping input data to higher-dimensional space. Throughout the research, the drawbacks of the existing methods will be identified and addressed those by developing a novel XAI solution.

ACKNOWLEDGMENT

Regarding the final year project involving an explanation for the causes of a prediction (INT2X), Warnasooriya S.D crafted this particular paper. This project is the result of the hard work put forth by each member of the team as well as the encouragement, support, and direction provided by numerous others, in particular our supervisor Mr. Prasanna Sumathipala and our co-supervisor Mr. Jeewaka Perera, other SLIIT institute lecturers who provided advice and guidance in turning this creative solution into a reality, and our dearest batchmates who assisted by lending a helping hand when it was necessary. A heartfelt expression of gratitude is due to all of our family members, friends, and colleagues for the unwavering support, care, and invaluable assistance that they have provided. I'd also want to use this opportunity to convey my appreciation to everyone else who has helped me or motivated me to make this a success but whose name isn't featured here. Throughout my whole academic career, I shall be eternally thankful to our previous teachers for the consistent support they provided.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Description
AI	Artificial Intelligence
XAI	Explainable AI
SVM	Support Vector Machine
KNN	K Nearest Neighbour
RF	Random Forest
LR	Logistic Regression
GDPR	General Data Protection Regulation
VCS	Version Control System
SHAP	SHapley Additive exPlanations
LIME	Local Interpretable Model-agnostic Explanations
NICE	Nearest Instance Counterfactual Explanation
DICE	Diverse Counterfactual Explanation.
SEDC	Sequentially Eliminating Discrediting Counterfactuals
NLP	Natural Language Processing
TCAV	Testing with Concept Activation Vectors

1.INTRODUCTION

The concept of Interpretable AI also known as Explainable AI has come into the discussion with the evolution of Artificial Intelligence (AI) under the Machine Learning domain. Nowadays predictions made by artificial intelligence are used for critical and sensitive decision-making processes. But this traditional AI is a black box that can answer only "yes" and "no" type questions without elaborating how that answer is obtained. As a solution for that, there was a requirement of explainability to ensure the trustworthiness and transparency of AI making decisions.

At the dawn of the fourth industrial revolution, different domains are witnessing a fast and widespread adoption of artificial intelligence (AI) in the decision-making process under domain experts. Healthcare, Legal, military, transportation, and finance are some domains where AI models use for the decision-making process. Decisions made by the AI model are extremely critical in a situation like a medical diagnosis of a disease. Here AI model should be able to provide an explanation of the decision because patients are interested in the treatment process and want to know why these treatments are required. There are some situations in which the models can be biased and provide inaccurate results. In such situations model explainability is well needed to identify and avoid errors in decision making.

When machine learning models are used for the decision-making process, it is not enough to have high accuracy. The model should be able to explain how each feature maps with the final result. In this situation, two types of models come into the picture called "white box models" and "black box models".

White box models are machine learning models where the internal workings of the model are transparent and understandable to the user. In other words, the user has access to the model's underlying structure, parameters, and decision-making processes. White box models provide a better explanation because those models are not too complex. The accuracy provided by these models may not be enough for some predictions. Examples of white box models are,

- Regression models
- Decision tree models
- Naive Bayes model

When we come to black-box models, the internal workings of the model are not clear and not easily understandable. Black-box models can handle complex, high-dimensional data and provide more accuracy than white-box models. However, black box models can be difficult to interpret or explain, which can make it challenging to identify and correct any errors or biases in the model. Examples of black box models are

- Deep neural networks
- Random forest
- Boosted trees
- Support Vector Machine
- K nearest neighbor

There is a popular example called the "Husky" mistake that properly demonstrates the need of model explainability. [1]

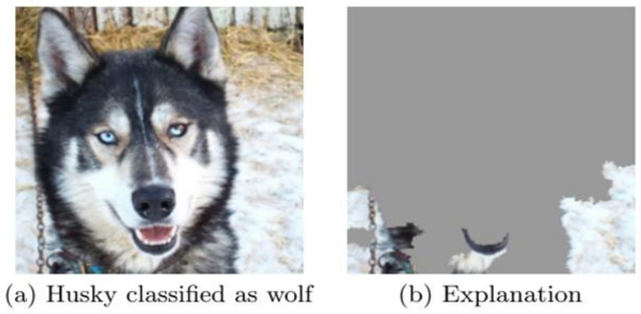


Figure1-1: Husky Wolf Explanation

In that scenario, researchers trained a logistic regression neural network model by feeding the images of wolves and huskies as the input features. The images of wolves had a white colour background(snow) and the images of huskies had not that white background. After training the model they provided an image of a husky that had a white colour (snow) background. The model predicted that image as a wolf. In that situation, the model did the prediction based on the background colour without considering the animal's colour, position, size, and other important features. That means the model is trained as a bad classifier. If this happens in a critical decision-making process the situation must be dangerous. Using XAI we can avoid these situations by explaining the models and identifying if there is any bias.

There is no exact definition for Model Explainability/Interpretability and different authors prefer to use the most sensible definitions. The most popular and widely used definitions are **“Interpretability is the degree to which a human can understand the cause of a decision” [2]** and “**Interpretability is the degree to which a human can consistently predict the model's result” [**3] The main objective of XAI is to answer the "wh" questions related to AI-based decision-making. For example, XAI should be able to answer "why a particular answer was obtained?", "how a particular answer was obtained" and "when a particular AI-based system can fail?" [4] In [4] they proposed four reasons why explainability is needed. Those are,

- **Explain to justify**.

There were some situations AI/ML-based systems provide biased or discriminatory results. Therefore, the explanation of AI-based results is essential. The XAI systems provide the required information to justify the result when unexpected decisions are made.

- **Explain to control.**

Through the explainability process, users can get an understanding about the system's behavior. It provides greater visibility of unknown vulnerabilities and flaws and helps to rapidly identify and correct errors.

- **Explain to improve.**

A model that can be explained is one that can be improved easily. Because users know the relationship between inputs and output, and how to make the output smarter.

- **Explain to discover**

Explanation about the process is helpful to learn new facts, gather information, and gain knowledge. XAI models are useful to find new and hidden laws in different scenarios.

Nowadays Explainable AI (XAI) has numerous applications across various industries and domains. XAI can be used in medical diagnosis, treatment planning, and drug discovery to provide explanations for the recommendations made by AI models. In finance sectors, XAI is used in fraud detection, credit scoring, and investment analysis processes. This can help financial institutions comply with regulations, increase transparency, and improve customer trust. Autonomous vehicles also use XAI tools to provide explanations for the decisions made by the AI models that control these vehicles. Through that explanations, those systems can ensure safety and increase public trust in these emerging technologies. In sentimental analysis, XAI tools assist to detect biases, identify errors and evaluate fairness issues in text data. Furthermore, XAI tools use to identify patterns and anomalies in manufacturing processes, allowing for predictive maintenance and quality control as well.

This research mainly focuses to enhance the model interpretability of the Black-Box models related to a text classification task. Model explainability useful in text classification tasks because of several reasons. Through explainability, the system can identify whether the model is biased toward certain words or phrases in the text. This help to improve the model's accuracy and avoid making unfair predictions based on certain demographic factors, such as race or gender. Model explainability provides insights into the model's decision-making process, making it easier for users to understand how the model arrived at its predictions. This can assist to build trust in the model and make it more accessible to non-experts. When text classification models are not performing as expected, model explainability can help identify specific areas where the model has errors and the reasons for those errors. Further, by understanding which features are most important for the model to make accurate predictions, model explainability can help improve the model's performance by allowing developers to optimize the feature selection or weighting.

1.1 BACKGROUND

The research area of XAI has been taken to the attention of the researchers and the communities with the fast growth of AI models and their applications. The need and the importance of the XAI concept have been raised when people consider the trustworthiness and transparency of AI-based decision-making processes. In the last few years, many researchers have been proposing many model-explainable methods for the research community. Because the knowledge in this area is not well matured most of the techniques are still in the research phase.

With the advancement in the XAI domain researchers have grouped the XAI explainability methods into different categories. They are the Local-Global method, model specific-model agnostic method, and Post hoc-Intrinsic method. When we consider the Local-Global explanation approach, Local explanation methods, aim to explain the model's predictions for individual instances or data points while Global explanation methods aim to provide an overall understanding of how a machine learning model works, and what factors or features are most important for the model's predictions. LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations) are local explanation methods and feature importance ranking, Partial dependence plots and decision trees are global explanation methods.

In model specific-model agnostic approach, Model specific methods are designed to explain a specific machine-learning model, while model-agnostic methods can be used to explain any machine-learning model, regardless of its type or complexity. Furthermore, the explanation methods can be post hoc or intrinsic. Intrinsic methods aim to build explainability into the machine learning model itself by considering the internal structure and parameters of the model, while post-hoc methods analyze the model after it has been trained. Intrinsic methods are typically used for simpler models that can be designed to be inherently interpretable and post-hoc methods are used for more complex models.

As discussed in the introduction the concept of XAI has been focusing on black box models. Support Vector Machine is one of the most popular ML models because of its excellent predictions and generalization capabilities. It becomes a black box model due to its model complexity and low interpretability. SVM is one of the most popular supervised learning techniques, which is used for classification, regression tasks, and outlier detections. But it mainly focuses on the classification task. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes with an optimal hyperplane. This best decision boundary is called a hyperplane. When we consider the visualization of SVM the model plots the data points in a dimensional space and the number of dimensions are equal to the number of features in the dataset. Also, SVM constructs a hyper-plane or set of hyper-planes in a high or infinite-dimensional space. There are two types of SVMs, linear SVM and nonlinear SVM. Linear SVM is used to handle linearly separable data, which means a dataset can be classified into two classes by using a single straight line while non-linear SVM is used to handle non-linear separable data.

When the input data is linearly separable, SVM uses a straight line to separate the data into two classes. Here SVM can be trained to find the optimal decision boundary that maximizes the margin between the two classes. To find the optimal hyperplane, the SVM solves an optimization problem that involves minimizing the norm of the weight vector subject to the constraint that all data points are classified correctly. This optimization problem can be solved efficiently using various optimization techniques such as quadratic programming. Once the optimal hyperplane is found, the SVM can make predictions on new input data by simply evaluating which side of the hyperplane the input data falls on.

When SVM is used to classify non-linearly separable data, it uses a non-linear kernel function to map the input data into a higher-dimensional space where a linear decision boundary can be found. The choice of the kernel function and its associated parameters can greatly impact the performance of the SVM. So, it may require significant experimentation and tuning to arrive at the best model. Once the optimal hyperplane is found, the SVM can make predictions on new input data by first mapping the input data into the higher-dimensional space using the kernel function and then evaluating which side of the hyperplane the transformed data falls on.

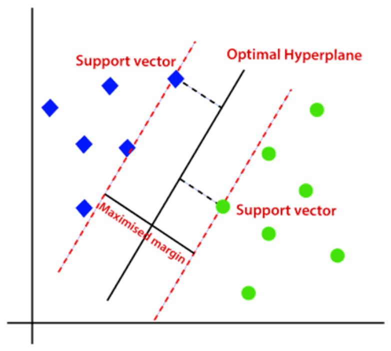


Figure.1.1-1: Linearly Separable Data

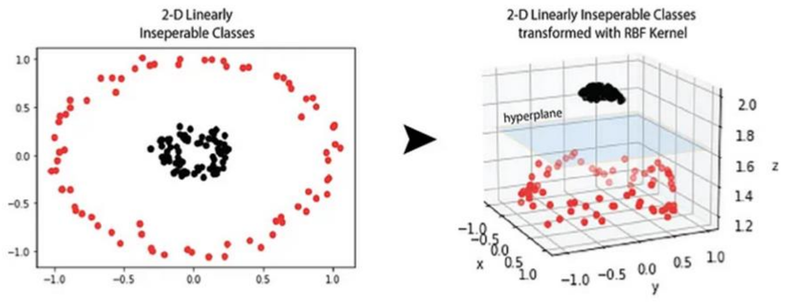


Figure.1.1-2: Nonlinearly Separable Data

The black box behavior of the SVM becomes more pronounced, **When SVM is used to classify non-linearly separable data.** When the input data is transformed into a higher-dimensional space using a non-linear kernel, the decision boundary can become highly complex and difficult to visualize. It may not be clear how the SVM is making its predictions, and it can be challenging to understand the role of each input feature in the decision-making process. Additionally, the kernel function used by the SVM may not have a direct interpretation in terms of the original input features, making it hard to understand which features are driving the SVM's predictions. That means challenging to understand the role of each input feature in the decision-making process.

When we consider the scope of SVM model usage natural language processing (NLP) is at the top. SVM is one of the most popular choices for text classification domain because SVM is a non-parametric algorithm and it can efficiently handle high-dimensional data by finding the optimal hyperplane that separates the different classes in the feature space. Also, SVMs are less prone to overfitting than other models, which is important in text classification because the number of features can be very large. This is because SVMs seek to maximize the margin between the different classes in the feature space, which helps to prevent overfitting. SVMs offer flexibility in the choice of kernel functions, which allows them to be used with different types of data and to capture different types of patterns. This makes SVMs a versatile algorithm for text classification.

The ultimate goal of this research is to overcome model explainability issues in the SVM model by providing a better XAI solution related to the text classification task.

1.2 LITERATURE REVIEW.

XAI is dedicated to demystifying the black boxes by properly explaining the internal process of AI/ML models. It improves the trustworthiness, transparency, and responsibility of AI-based decision-making processes. Since the XAI research area is still in the starting era most of the techniques are in the research phase and researchers had built communities for do contributions to the XAI domain.

The author of [2] mentioned there are several explanation methods and strategies had proposed by researchers to make AI systems explainable. Among those techniques "Scoop Related Methods" and "Model related Methods" are commonly used for the explainable process. Scoop related methods are classified as global interpretability and local interpretability. Global interpretability facilitates the understanding of the whole mechanism of the model and the entire reasonings cause for the final output. Local interpretability focuses on explaining the reasons for a specific decision or a single prediction. Model-related methods are also classified into two categories as model specific interpretability (methods are limited to a specific model) and model-agnostic interpretability (methods are not tied to a specific ML model).

From the proposed techniques some have become more popular among the research community and new contributions are also done on top of these techniques. LIME (Local Interpretable Model-agnostic Explanation) [1], SHAP (SHapely Additive exPlanations) [5], XAI360 and Google XAI are examples for that. When we consider LIME, it interprets the model and explains the classification of the model in a faithful manner. It provides local optimum explanations which compute the important features by generating samples of the feature vector. Those samples are following a normal distribution. After getting the predictions from the samples it assigns weights to each of the rows to get an idea of how close they are from the original sample. Then LIME uses a feature selection technique to identify the most significant features.

SHAP is a unified approach that has been developed based on coalitional game theory. In that theory, they assign a reward to the game players according to their contributions to the game. SHAP assigns feature importance value for each feature that affects a particular output result.
It maps the input features with the output results based on that Sharpley value. The key difference between LIME and SHAP is in the way that assigns weights to the input features. LIME uses a cosine measurement while SHAP uses Shapley formula. In this review, rule-based explanation methods will be discussed.

XAI360 is a commercial explainable AI (XAI) tool that provides a model-agnostic explanation. Also, it provides both local and global explanations to enhance the model interpretability of black box models. XAI360 uses various techniques to do the model explainability task. Feature importance ranking, decision tree method, counterfactual explanation, and visualizations are some of those techniques. Through feature importance ranking, XAI360 can identify the features or variables that had the most influence on the model's output and understand which factors were most important in driving the model's predictions or recommendations. XAI360 can generate decision trees that visualize the decision-making process of the black box model. These trees show how the model arrived at its output by breaking down the decision process into a series of smaller, more understandable steps.

Google XAI (Explainable AI) is a research initiative focused on developing techniques and tools for making AI more transparent and interpretable. This is also a model-agnostic method that provides both local and global explanations. Testing with Concept Activation Vectors (TCAV), integrated gradients, and attention mechanisms are some of the explainable techniques developed by Google XAI. TCAV provides a technique for understanding how models make decisions by analyzing the activations of different neurons in the model's deep learning architecture. Integrated Gradients calculate the importance of input features by integrating the gradients of the model's output with respect to the input features. Furthermore, attention mechanisms are used in deep learning architectures to identify which parts of an input are most important for the model's output.

1.2.1 White-box Induction from SVM Models: Explainable AI with Logic Programming [6]

The author of [6] had come up with a model-specific explainable approach to SVM that focuses on inducing logic programs. Inductive Logic Programming (ILP) is a subfield of machine learning and here the models do the learning process in the form of logic programming rules (Horn Clauses) that are comprehensible to humans. They get used SHAP to calculate the feature importance value of the input features. This paper makes a novel contribution to introducing a novel ILP algorithm called SHAP-FOIL that iteratively learns a single clause for the most influential support vector. According to the paper, FOIL is a top-down sequential covering inductive logic programming algorithm and there were some issues with those types of algorithms. To overcome those problems, they proposed this new SHAP-FOIL algorithm.

This SHAP-FOIL method has been introduced as a statistical learning-based ILP method. Also, the SHAP-FOIL algorithm has the capability of learning the logic programs based on the global behavior of the SVM model. Explain the model as a logic program leads to greater comprehensibility for the users because of its well-defined declarative and operational semantics. The intuition behind the SHAP-FOIL algorithm is: If a subset of feature values explains the decision on a particular support vector, it explains the decision on data points that are "similar" to that support vector too. Similarity is measured using an equation.

1.2.2 Anchors: High-Precision Model-Agnostic Explanations [7]

In this research, they have introduced a novel model-agnostic methodology that explains the behavior of complex models with high-precision if-then rules called "anchors". Since this method is model agnostic it can be applied to any model that exists. This study shows how a model would behave on unseen data instances with less effort and higher precision, as compared to existing explanations. In that approach, once the model identifies the features that hold Anchors, the changes that happen for the rest of the features will not be considered. Therefore, the prediction will be almost the same all the time according to the anchors.

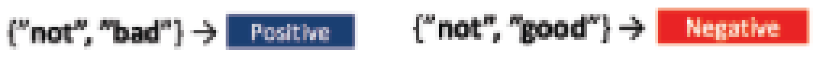


Figure1.2.2-1: Anchors Sample Explanation

Above example was given in the paper [7]. In this example, the method considers the words “not bad” and “not good” to virtually predict the sentiment of the statement (“not bad” as positive and “not good as negative”). This method has been applied to various machine learning scenarios and domains including Text Classification, Structured Prediction, Tabular Classification, Image Classification, and Visual Question Answering to prove the usefulness of the method. Further, the study shows that Anchors can effectively identify the most relevant parts of the input that contribute to the classifier's prediction.

1.2.3 Local rule-based explanations of black box decision systems [8]

In this study, the authors have proposed a method called "LORE" which is a local agnostic approach that provides interpretable explanations on black box models based on logic rules. First, it leans a local interpretable predictor on a synthetic neighbourhood generated by a genetic algorithm. Then it derives the logic from the local interpretable predictor to provide a meaningful explanation that caused the prediction. Apart from providing an explanation for decision making LORE provides a set of counterfactual rules, which propose the changes in the features that lead to a different outcome.

According to the study [8], it supports for relational, tabular data to generate explanation rules. The high-level idea of the process has been explained like this. There is a given black box predictor which used to perform binary predictions and a specific instance ‘x’ labeled with outcome ‘y’ by the binary predictor. First. It will develop an interpretable predictor using a balanced dataset of neighbourhood of the given instance ‘x’ through a genetic algorithm. Then the local explanation for the given instance ‘x’ is extracted using the obtained decision tree classifier. Furthermore, it generated a set of counterfactual rules, that explains which features and conditions should be changed in order to invert the given prediction ‘y’. As the authors of [8] say, the compass dataset, it may come up with the below explanations for both decision and counterfactuals.

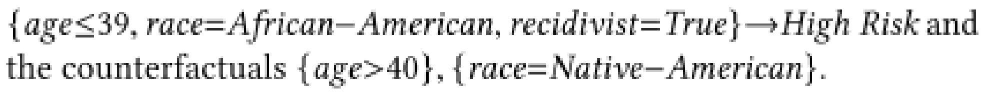

$\{age \leq 39,\ race{=}African{-}American,\ recidivist{=}True\} \rightarrow High\ Risk$ and the counterfactuals $\{age > 40\}$, $\{race{=}Native{-}American\}$.

Figure 1.2.3-1: Explanation rule and the counterfactual for compass dataset

The logic of this method is common as the methods discussed early in this review such as LIME [1], and Anchors [7]. The novelty of this method is it uses genetic algorithm to capture the decision boundary in the neighbourhood. Therefore, it produces high quality training dataset that will be used for learning the decision tree classifier in order to explore the decision rules.

1.2.4 Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations [9]

Concept of Counterfactuals requires imaginations of hypothetical realities that can be happened other than the existing situation. In this research, to understand the concept of Counterfactuals it has given a real-time scenario that can be happened in day-to-day life. It will be discussed below. In a situation where a loan has been rejected. Using interpretable models, the organization may give an explanation for that action. As an example, it can be because of "poor credit" history. When we consider the person's point of view who has been applied for the loan, the explanation does not help him to decide "what should he/she do next?" If the system is able to suggest some solutions to improve the chance to get the loan in future, that would be more effective for both sides.

However, there might be some situations where the most importance feature or features like gender, race may not be sufficient to flip or change the prediction. Therefore, it is pretty much important to provide alternative rules which are actionable. As the counterfactual explanation for the above loan example, the paper has provided below information [9].

"You would have received the load if your income was higher by $10000".

When the person gets that kind of explanation, he/she would be able to identify which features that need to be improved to be eligible for the load in future.

1.2.5 Nearest Instance Counterfactual Explanations [10]

NICE (Nearest Instance Counterfactual Explanations) is a counterfactual explanation method that aims to find the smallest and most meaningful changes to an instance that would alter the model's prediction. This method aims to make explanations more interpretable and practical compared to other counterfactual explanation methods. The algorithm considers the feasibility of the counterfactuals generated, ensuring that they are practically possible and not merely hypothetical. Further NICE produces explanations that are more interpretable and easier to understand for human users. The focus on the smallest and most meaningful changes ensures that users can comprehend why the model made a certain prediction and what would need to change to alter that prediction.

1.2.6 Sequentially Eliminating Discrediting Counterfactuals (SEDC) [12]

The SEDC method identifies counterfactual explanations in textual data. By treating each word as a distinct feature, the method iteratively removes features to shift in the model's predictions. Features that notably alter the predictions upon removal are considered highly influential for the initial classification. Expanding on this, Ramon et al. (2020) developed the SHAP-C method, blending the strengths of Shapley Additive Explanations (SHAP) and SEDC. Using SHAP's game theory foundation, it determines feature importance and subsequently, the SEDC method crafts counterfactual explanations by sequentially omitting crucial features

1.3 RESEARCH GAP

The main target of this research is to make the SVM model more interpretable related to the text classification domain. we can use a counterfactual rules generation mechanism, which proposes the changes that need to be done in the input features to flip the final predictions to achieve the SVM model explainable task.

The proposed local rule-based explanation methods are mainly focusing on binary predictors. Because of that LORE [8] and SHAP-FOIL algorithm [6] can't be used for text classification tasks. Also, those two approaches are not used a counterfactual explanation method for the explanation process. When we consider Anchors [7], it can be used for text classification tasks but it does not follow a counterfactual explanation approach. In LIME [1], SHAP [5] XAI360[4], and Google XAI [11] tools, they provide text classification explanation mechanisms. But only XAI360 provides a counterfactual explanation for explaining the internal behavior of black box models.

When we consider SHAP [5], its' computations are more complex, especially for models with a large number of features or complex interactions. This can make it unsuitable for real-time or large-scale applications. Also, SHAP assumes that features are independent, which is often not the case in real-world datasets. Furthermore, it does not directly provide counterfactual explanations, but it does provide a way to understand how each feature contributes to a specific prediction, either positively or negatively, in comparison to a baseline. SHAP values provide detailed insights into feature contributions for individual predictions, they do not directly answer the "what if" questions posed by counterfactual explanations. In LIME [1] also there are some weaknesses**.** LIME explanations can be unstable, which means small changes to the instance can lead to different explanations due to the random sampling process used in the local surrogate model fitting procedure.

Diverse Counterfactual Explanations [9] are designed to generate counterfactual explanations for any machine learning classifier but the process of generating diverse counterfactuals can be computationally demanding and it also assumes that the features are independent, which might not be the case in many real-world datasets. The NICE [10] method also provides a counterfactual explanation but it is a model agnostic and finding the nearest instance and ensuring feasibility can be computationally intensive.

Further, model explainability visualizations provide by most of the XAI tools are very complex and those visualizations can understand only by the subjects' experts. So, it is very important to provide a user-friendly more understandable visualization of the internal process of the black box models to the end users

The proposed method in this research study provides a model-specific counterfactual explanation using a custom word-flipping generator with a cosine similarity comparison mechanism to enhance the explainability of the SVM classifier related to the text classification domain.

1.4 RESEARCH PROBLEM

How to get a counterfactual rule generation-based explanation for the Support Vector Machine classifier, when it handles non-linear separable data in text classification?

When it comes to the applications of Machine Learning models, most of the time those models are used in decision classifier systems. Therefore, the importance of the XAI is highlighted in those situations. As discussed in the previous sections, multiple approaches have been taken by the researchers to provide a proper explainable mechanism to explain those models. Most of these proposed solutions are still in the research phase because of the immaturity of the XAI domain.

This research mainly focuses to enhance the model interpretability of the SVM model related to a text classification task. Text classification is a natural language processing (NLP) task that involves categorizing text documents into predefined categories or labels based on their content. In other words, it's the process of assigning a category or class to a piece of text automatically. As discussed in the background SVMs are popular in the text classification domain because of its ability to efficiently handle high-dimensional data, robustness to overfitting, and flexibility in kernel choice. When we consider about make the SVM model explainable, a counterfactual rule generation mechanism can use for it. Counterfactual explanations assist to make SVM models more interpretable by providing concrete examples of how a change in input variables would affect the output. Also, through these counterfactual rules, we can identify the key features that are driving those predictions and properly map the input features with the output result.

For Example, suppose there is a text classification problem that needs to classify customer reviews of a product as positive or negative. The dataset contains the customer reviews with their corresponding labels, and after training the SVM model it predicted that a particular customer review was positive. Here we can use counterfactual rule generation to generate a set of alternative reviews that would have been classified as negative by the model. These alternative reviews would differ from the original review in one or more key features, such as the presence or absence of certain words or phrases.

By these counterfactual explanations, we can identify which words or phrases were most important in driving the model's positive classification for the original review. That information can use to provide an XAI solution for the SVM model that makes the model more interpretable.

When we consider the issues in the text classification domain being biased for a particular topic is major. It may cause for happen a fairness issue as well. There were some situations in the model trained on news articles that disproportionately includes articles from certain news sources or about certain topics. For instance, if the training data contains mostly articles from conservative news sources or about conservative politicians, the resulting model may perform poorly on articles about liberal politicians or liberal-leaning news sources. Similarly, if the training data includes mostly articles about crime, the model may be biased towards classifying all text related to crime as negative, even if some articles are discussing positive developments in crime prevention or criminal justice reform. Such biases can lead to inaccurate and unfair text classification results. So, it is very important to identify the biased classifiers and avoid inaccurate results while improving the fairness of decisions. For that purpose, we can apply counterfactual rule-generation techniques.

Counterfactual analysis assists to detect and mitigate bias in SVM text classification models by generating hypothetical scenarios where a particular protected attribute (such as race or gender) is changed. Here we can observe whether the model's output is affected by that attribute. If the model's output changes significantly when the protected attribute is changed, it may be a sign that the model is biased. Also, this Counterfactual analysis can be used to understand why a particular SVM text classification model is making certain errors. As the previous bias detection solution, it generates hypothetical scenarios to identify where the input is slightly changed to cause model misclassification. By observing what aspects of the inputs are causing the error we can identify areas for improvement in the model. Furthermore, this counterfactual analysis can use to evaluate the fairness of SVM text classification models. The rules can identify where certain groups are overrepresented or underrepresented and if the model is affected by the group differences.

In the XAI research area, there are some proposed counterfactual rule generation-based SVM model explainable methods. But most of those counterfactual rule-based mechanisms cannot be used in text classification tasks because the text classification scenarios contain high dimensional data. Also as mentioned in the research gap section model explainability visualizations provided by proposed methods are not user-friendly and understandable for end users. To overcome above discussed problems, it is important to provide an XAI solution to make the SVM model more explainable related to the text classification tasks.

1.5 OBJECTIVES

1.5.1 Main Objective.

Provide a novel model-specific, local XAI solution to enhance the model interpretability of function-based classification models focus on SVM by developing a novel counterfactual rule generation mechanism related to the text classification domain.

The novel explanation method uses a custom word-flipping generator with a cosine similarity comparison mechanism for generating counterfactual rules. Word flipping generator returns a number of contradictory prompts related to the given original prompt. The best contradiction is selected by comparing the cosine similarity between the mirror point of the original prompt with contradictory prompts. The system can determine the most affecting words to provide a counterfactual explanation analyzing the most accurate contradictory prompt.

Eventually, the proposed solution will be visualized using the most appropriate Graphical User Interface (GUI) techniques that provide a better user experience to the end users. Through the visualization, the user can get a better understanding about the relationship between input features and how input features are mapped to an output.

1.5.2 Specific Objectives.

To achieve the main objective, there are few milestones to reach.

1. **Prepare the dataset and implement the SVM classifier.**

To perform the experiments, there should be identified dataset that aligns with the requirements. The study used the IMDb Movie Reviews Dataset dataset, consisting of 50,000 movie reviews. Each review, presented as a textual statement, encapsulates a user's opinion and sentiments regarding a particular film. Reviews are labeled either as 'positive' or 'negative', signifying the sentiment expressed in the review.

2. **Develop the novel counterfactual rule generation mechanism related to the text classification task.**

To generate counterfactual explanation rules, I used a custom word-flipping generator with a cosine similarity comparison mechanism. The custom word-flipping generator randomly flips words with defined POS tags to their antonyms and it returns contradictions. The end of the methodology system selects the most accurate contradictory prompt and provides counterfactual explanation.

3. **Test the output with existing explainable methods.**
4. **Do experiments to improve the XAI solution more.**
5. **Do the visualization using the most appropriate Graphical User Interface (GUI) technique.**

2.METHODOLOGY

In order to implement the proposed solution, there are few milestones to accomplish. In this section, it will be discussed the steps that need to be followed to carry out the study and the tools and technologies that are going to be required when accomplishing the sub-tasks in a more detailed manner.

2.1 Overall System Architecture

The project's primary goal is to provide a novel XAI solution to enhance the model interpretability of function-based classification models focused on Support Vector Machine by developing a novel counterfactual explanation related to the text classification domain.

When applying SVM for a text classification task, we have to do text pre-processing first. This involves tasks like lowercasing, tokenization (splitting text into words or tokens) and removing stop words and punctuations. Also, text data may undergo stemming or lemmatization to reduce words to their base forms, which can help improve feature representation. After pre-processing SVM required a numerical representation of text data for the feature extraction. Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word Embeddings are commonly used techniques for the above-mentioned feature extraction in text classification tasks. Bag of Words (BoW) is a matrix where each row corresponds to a document, and each column corresponds to a unique word in the corpus. The cell values represent word frequencies in the documents. The TFIDF technique calculates a numerical value for each word in a document based on its frequency in that document and its rarity across all documents in the corpus. when we consider word embedding techniques like Word2Vec, GloVe, or BERT embeddings, they can be used to convert words into dense vector representations.

As the third step of the text classification in SVM, each document is represented as a high-dimensional vector in a feature space of SVM. The dimensionality of this space depends on the number of unique features extracted from the text. Then involves the Label Encoding step that encodes the target labels into numerical values. For binary classification, this might be 0 and 1; for multiclass classification, it could be 0, 1, 2, etc.

Once the text data is preprocessed and represented numerically, the SVM model is trained on this data and finds the hyperplane that best separates the data points belonging to different classes while maximizing the margin between these points. After the training process, SVM can make the classification decisions in text classification tasks.

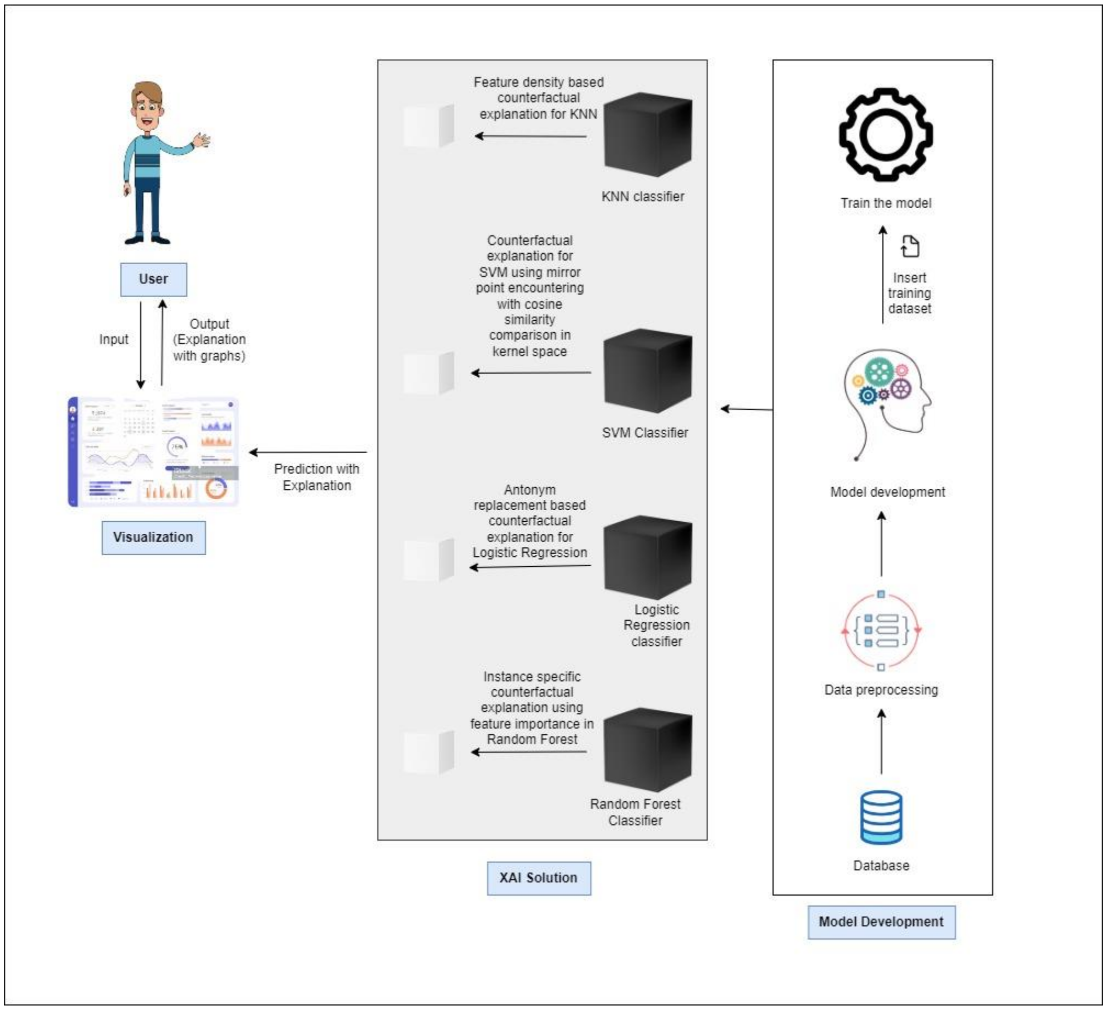

User
Input
Output (Explanation with graphs)
Visualization
Prediction with Explanation
Feature density based counterfactual explanation for KNN
KNN classifier
Counterfactual explanation for SVM using mirror point encountering with cosine similarity comparison in kernel space
SVM Classifier
Antonym replacement based counterfactual explanation for Logistic Regression
Logistic Regression classifier
Instance specific counterfactual explanation using feature importance in Random Forest
Random Forest Classifier
XAI Solution
Train the model
Insert training dataset
Model development
Data preprocessing
Database
Model Development

Figure 2.1-1: System Architecture Diagram

As the above system architecture diagram, the main objective is accomplished through the process which contains model development, encountering the novel XAI solution, and developing the GUI for end users. The operational workflow of the system entails a series of distinct steps; First, the user has to provide the relevant training dataset and the instance that needs to be predicted to the system through the user interface. Then the provided data will be applied to the support vector machine classifier to get the predictions.

To extract the counterfactual rules system will apply the novel counterfactual rule generation mechanism which contains a custom word flipping generator and mechanism of cosine similarity comparison to the SVM classifier. After extracting the counterfactual rules, it is important to evaluate the novel explanation mechanism by testing with existing explainable tools and analyzing experts' feedback. Finally, the outcomes of the process (Counterfactual Explanation) will be transferred to the GUI with appropriate visualizations to be more understandable for the users.

This research study used the "IMDb Movie Reviews Dataset" which is a popular dataset applied in natural language processing (NLP) and sentiment analysis. It typically consists of 50,000 movie reviews and each review, presented as a textual statement, encapsulates a user's opinion and sentiments regarding a particular film. These reviews can vary in length from a few words to several sentences. Reviews are labeled either as 'positive' or 'negative', signifying the overall sentiment expressed in the review. For example; if a review generally praises the movie and expresses a positive sentiment, it is labeled as 'positive,' whereas if the review is critical and expresses a negative sentiment, it is labeled as 'negative.' Movie reviews often contain various textual elements like quotations, character names, or plot points. So, it is essential to follow the necessary preprocessing steps before the model development process. As the preprocessing steps, initially, all characters in the text are converted to lowercase. Then remove any HTML content, any special characters, and stopwords present in the text. Finally did the words tokenization and lemmatization.

2.2 Counterfactual Explanation for Support Vector Machine

As described in the overall system architecture, SVM maps the text values into a vector space using a vectorizer in text classification task. Once the SVM learns the vector space of $\emptyset(x)$ it finds the best hyperplane that satisfies $\boldsymbol{\omega}^{\boldsymbol{T}}.\ \emptyset(x) + b = 0$. Note that $\boldsymbol{\omega}$ are the coefficients with size n.

$$\begin{tikzcd} \boxed{''prompt''} \arrow[r] & \boxed{Vector_{TFIDF}\ (i.e.,x)} \arrow[r] & \boxed{Vector_{SVM}\ (i.e.,\phi(x))} \end{tikzcd} x \in R^m,\ \phi(x) \in R^n.$$

Figure 2.2-1 Applying SVM for Text

Note that here m is the number of dimensions in the vector space of the TFIDF vectorizer and n is the number of dimensions in the vector space learnt by the SVM. $\emptyset(x)$ is known as the kernel function. Hence, the vector space learnt by the SVM is commonly known as the **output vector space of the kernel function.**

Through the research study, I provide a novel Counterfactual Explanation for the SVM classifier by applying a custom word-flipping generator with a cosine-similarity comparison mechanism. The mentioned novel SVM explainable method provides the counterfactual solution by going through the five steps that illustrate in the below figure 2.2-2

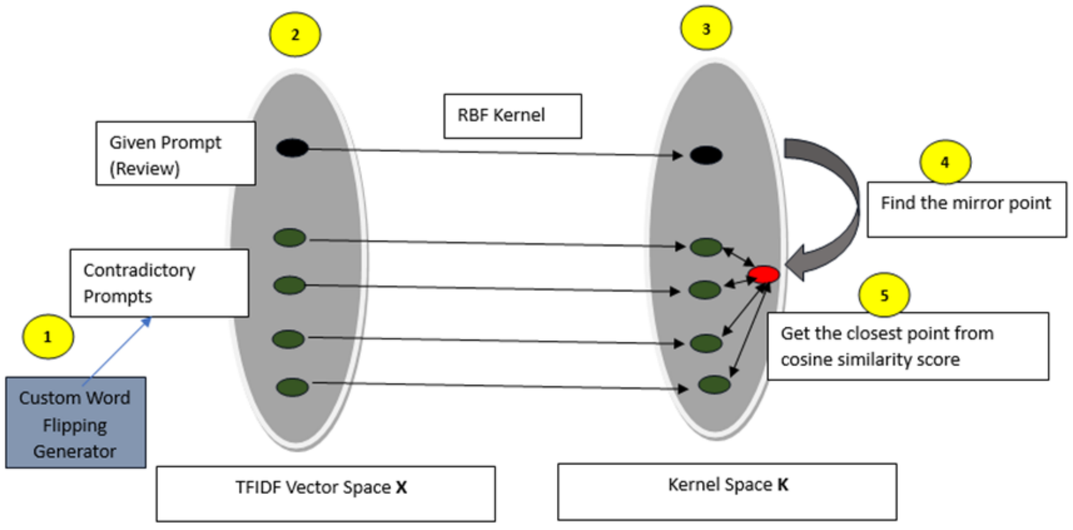


Figure 2.2-2 Counterfactual Explanation Methodology

First, it generates contradictory prompts for a given prompt using a finetuned T5 model or custom WordFlippingGenerator. These new prompts will be [$\boldsymbol{Contradictory\_prompt_i}$].

WordFlippingGenerator randomly flips the words with defined Part of Speech (POS) tags to their antonyms. POS tags are labels used to indicate the part of speech and other grammatical categories (case, tense etc.) of each token in a text corpus. Here the user should define the POS tags that are relevant to the words that must be flipped and invoke the functionality by specifying an original prompt with the number of variations needed. Then WordFlippingGenerator algorithm will tokenize the words and generate a mask list that corresponds to the tokens by referring to the POS tags previously defined by the user. The true value of the mask represents that the word must be flipped and false means otherwise. As the next step, the algorithm generates a set of lists that contain antonyms that are ordered according to the descending order of occurrence probability for the flipped words referring to the mask. Here the occurrence probabilities of antonyms follow the negative exponential distribution. That means the most commonly used antonyms return the beginning of the antonym list. Finally, the new sentences will be generated by merging the original words with the antonyms generated by the WordFlippingGenerator.

After the contradictory prompt generation, the TFIDF vectorizer vectorizes all the prompts into the vector space X. In there the $\boldsymbol{Prompt_0}$ will be mapped to $\boldsymbol{x_0}$ and the [$\boldsymbol{contradictionary\_prompt_i}$] will be mapped to $\boldsymbol{x_{c,i}}$ vector space using TFIDF vectorizer. TF-IDF vectorization assigns numerical values to words in a document based on their importance within that document relative to their frequency across a corpus of documents. It helps capture the significance of terms in a document while considering their distribution across the entire dataset. TFIDF values are calculated by multiplying the Term Frequency (TF) and Inverse Document Frequency (IDF) values for each term in a document. Mathematically, the TFIDF value for a term $\boldsymbol{t}$ in a document $\boldsymbol{d}$ is calculated as:

$$\boxed{TF - IDF(t, d) = TF(t, d) \times IDF\ (t)}$$

Term Frequency measures how frequently a term (word) appears in a document and is calculated as the ratio of the number of times a term appears in a document to the total number of terms in that document. Mathematically, for a term $\boldsymbol{t}$ in document $\boldsymbol{d}$, TF is calculated as:

$$\boxed{TF(t,d) = \frac{Number\ of\ times\ term\ t\ appears\ in\ document\ d}{Total\ number\ of\ terms\ in\ document\ d}}$$

Inverse Document Frequency measures the importance of a term across a corpus of documents and it is calculated as the logarithm of the ratio of the total number of documents in the corpus (N) to the number of documents containing the term (n_t). Mathematically, for a term $\boldsymbol{t}$ across a corpus of (N) documents, IDF is calculated as:

$$\boxed{IDF(t) = \log\left(\frac{N}{n_t}\right)}$$

The TF-IDF vector for a document is constructed by arranging the TF-IDF scores for all terms in the document in a vector format and each dimension of the vector corresponds to a unique term in the corpus. Most of the dimensions in the TF-IDF vector are typically zero (sparse) because most terms appear in only a subset of documents.

TFIDF vector offers more advantages than other feature extraction techniques for the vectorization in SVMs' text classification tasks. It helps in the creation of a high-dimensional feature space that can be transformed by non-linear SVM kernels. When using non-linear kernels, the model can find intricate patterns and decision boundaries in the TF-IDF feature space, potentially leading to improved classification accuracy. By preserving term importance, TF-IDF allows non-linear SVM models to focus on the most relevant terms for classification while accounting for complex relationships between them. Further Non-linear SVM kernels effectively apply the "kernel trick" to map data into a higher-dimensional space without explicitly calculating the transformation. This allows non-linear SVMs to work in the transformed space created by TF-IDF, potentially revealing non-linear separability patterns that may not be apparent in the original space.

As the third step of the method, algorithm project all the vectors ($\boldsymbol{x_0}$ and $\boldsymbol{x_{c,i}}$) into the SVM' s kernel space K.

$$\begin{tikzcd}\boxed{Vector_{TFIDF}\ (i.e., x_0)} \arrow[r] & \boxed{Vector_{SVM}\ (i.e., \phi(x_0) \in K)}\end{tikzcd}$$

Figure 2.2-3: Kernel Space Mapping

Here SVM use Radial Basis Function (RBF) as the kernel [$\emptyset(.)$]. By assuming a single TFIDF vector will have the size $\boldsymbol{n}$ and the number of support vectors will be $\boldsymbol{m}$ the RBF kernel given by

$$\boxed{K(\overrightarrow{x}, \overrightarrow{l^m}) = e^{-\gamma\|\overrightarrow{x}-\overrightarrow{l^m}\|^2}}$$

Note that $\boldsymbol{x}$ is a vector in the TFIDF vector space with size $\boldsymbol{m}$. $\boldsymbol{l^m}$is a collection of vectors (i.e., the support vectors) with each of size $\boldsymbol{n}$.

The RBF kernel is a powerful tool for capturing complex, non-linear relationships in the data. It introduces non-linearity to SVMs, allowing them to model and classify data that is not linearly separable in the original feature space. When considering the functionality of the RBF kernel it applies the "kernel trick," which implicitly maps data points from the input space into a higher-dimensional space without explicitly computing the transformation. This higher-dimensional space is where the SVM seeks to find a linear separation of data. Also, it is characterized by its radial symmetry, with the similarity between two data points diminishing as they move further. This makes it effective at capturing complex, non-linear patterns. Further, RBF kernels' flexibility and implicit feature mapping capabilities provide more advantages to the SVM classifier when it handles nonlinearly separable data. Flexibility makes RBF suitable for a wide range of applications by capturing complex patterns in the data and through implicit feature mapping capabilities the kernel trick allows it to work in a potentially infinite-dimensional feature space without explicitly computing the feature vectors.

Next have to find the mirror point of the given prompt's TFIDF vector on the hyperplane of the SVM ($\boldsymbol{C}$). The mirror point conceptually refers to a point on the opposite side of the decision boundary from a given support vector. This point has the same distance as the support vector but lies on the other side of the hyperplane. This concept assists in understanding the balance of the margin in SVM as well.

Once we have $\emptyset(\boldsymbol{x_0})$ for the given prompt, we find its opposite projection on the hyperplane of the SVM characterized by $\boldsymbol{w}$ and $\boldsymbol{b}$. For simplicity, we'll call $\emptyset(\boldsymbol{x_0})$ as $\boldsymbol{A}$ and $hyperplane$ as $\boldsymbol{h}$.

$$\boldsymbol{hyperplane = h \equiv (w_1, w_2, \dots w_n, b)}$$

$$\emptyset(\boldsymbol{x_0}) = \boldsymbol{A} = (\boldsymbol{a_1, a_2, \dots, a_n})$$

Any line $\boldsymbol{l}$ which is normal to the $\boldsymbol{h}$ through **A** will be given by the parametric equation (Line $\boldsymbol{l}$ goes across the given prompt as well as perpendicular to the hyperplane),

$$\boldsymbol{l \equiv A + tw = 0}$$

$$\boldsymbol{l \equiv (a_1 + tw_1,\ a_2 + tw_2, \dots,\ a_n\ + tw_n)}$$

Let $\boldsymbol{t}$ take the value $\boldsymbol{t_0}$at the point $\boldsymbol{B}$ that lies on this line and the hyperplane.

$$\boldsymbol{B\ \equiv (a_1 + t_0w_1, a_2 +\ t_0w_2, \dots, a_n + t_0w_n)}$$

Since this point would also satisfy the hyperplane,

$$\boldsymbol{w^T . l_0 + b = 0}$$

$$\boldsymbol{(w_1(a_1 + t_0w_1), w_2(a_2 +\ t_0w_2), \dots, w_n(a_n + t_0w_n)) + b = 0}$$

$$\boldsymbol{t_0 = -\frac{(b +\ w^T . A)}{\|w\|\frac{2}{2}}}$$

The mirror point $\boldsymbol{C}$ will exist where $\boldsymbol{t = 2t_0}$. Hence,

$$\boldsymbol{C\ \equiv (a_1 + 2t_0w_1, a_2 +\ 2t_0w_2, \dots, a_n + 2t_0w_n)}$$

As the final step, algorithm find the closest point to mirror point $(\boldsymbol{C})$ and retrieve the most accurate contradictory prompt as the output. Here the mirror point and contradictory points all are in the kernel space. Most accurate contradictonary prompt return using the cosine-similarity between the mirror point and the contradictonary prompts.

Cosine similarity is a measure of similarity between two non-zero vectors in an inner product space. It measures the cosine of the angle between the vectors and quantifies how similar or dissimilar they are. A cosine similarity of 1 indicates that the vectors are identical, while a value of 0 means they are orthogonal (completely dissimilar), and a value of -1 indicates they are diametrically opposed. The formula for calculating cosine similarity between two vectors, A and B, is as follows:

$$\boxed{Cosine\ Similarity(A, B) = \frac{A.B}{\|A\|.\|B\|}}$$

$A.B$ is the dot product of vectors A and B. The dot product is the sum of the element-wise products of the vectors.

$$\boxed{A.B = \sum_{i=1}^{n} A_i.B_i}$$

Where; A_i and B_i are the components (values) of vectors A and Bin dimension $\boldsymbol{i}$. $\boldsymbol{n}$ is the number of dimensions (or features) in the vectors.

$\|A\|.\|B\|$ represent the magnitudes (L2 norms) of vectors A and B, respectively. The magnitude of a vector is the square root of the sum of the squares of its components.

$$\boxed{\|A\| = \sqrt{\sum_{i=1}^{n} (A_i)^2}} \qquad \boxed{\|B\| = \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Cosine similarity offers several distinct advantages over other similarity metrics when used in text classification tasks with SVM. It is scale-invariant, which means it is not affected by the magnitude of the vectors and only focuses on the direction of the vectors, making it robust to varying document lengths. Also, cosine similarity is more suitable for measuring semantic similarity between documents because it focuses on the direction of vectors rather than their absolute values. When we consider the computations of cosine similarity those are very efficient, especially when dealing with sparse data representations like TF-IDF vectors. This is essential for processing large text corpora efficiently. In text data, most feature dimensions are zero (sparse) and cosine similarity inherently handles those sparse data efficiently because it only considers non-zero dimensions, reducing the impact of irrelevant terms. Further, the cosine similarity is tolerant to synonyms and variations as well. That means it is robust to synonymous terms and minor variations in language and because of that documents with similar meanings but different phrasing can still receive high similarity scores.

After returning the most accurate closest contradictory prompt by comparing the cosine similarity scores, system can determine the most affecting words to provide a counterfactual explanation by selecting the flipped words that refer to the antonyms. That means the original words that refer to the flipped antonyms are the most important features that caused to change the class label while providing the counterfactuals.

2.3 Testing and Implementation

2.3.1 Implementation

The implementation of the proposed counterfactual explanation for the SVM classifier is accomplished through the process which contains model development, encountering the novel XAI solution, and developing the GUI for end users.

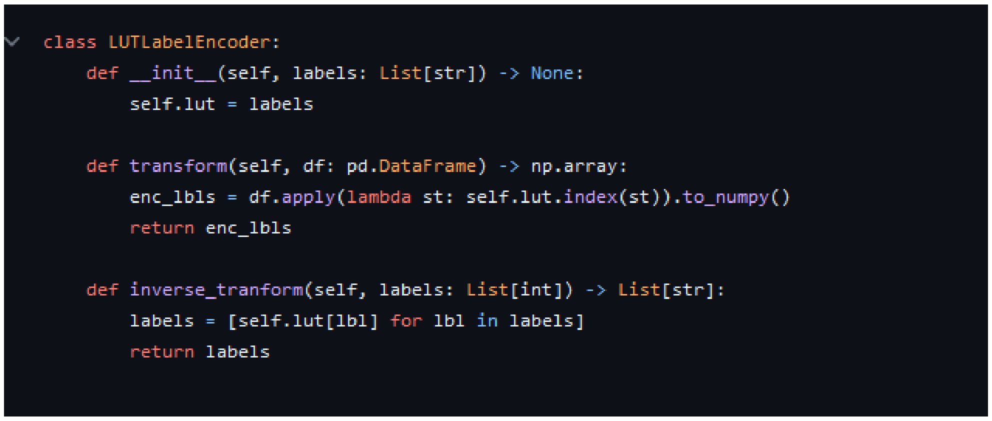
Here the backend developments were done using Google Colab and Visual Studio Code in Python. First, the support vector machine classifier was trained using the IMDB movie review dataset. The trained model can efficiently classify the input movie reviews as either positive or negative. Then developed the custom word flipping generator implementation and co-sine similarity comparison based counterfactual explanation in the VS Code. we used AWS to host the backend server of the web application and through that, we can ensure the reliability and scalability of the application.

Further, the front-end developments were done using NestJs and MantineUI. NestJs is a versatile framework for building efficient and scalable server-side applications and Mantine UI is a modern set of high-quality components and hooks for React and Next Js. It provides ready-made design elements and tools that make it easier to create interactive and visually appealing web interfaces. Together, NestJs and Mantine UI provided a solid foundation for building and styling the front end of the application.

Data Preprocessing:

I followed the necessary data preprocessing steps before the model development process. Initially used Look-Up Table Label Encoder (LUTLabelEncoder) to encode categorical labels into numerical values. This LUTLabelEncoder ensures consistency in the encoding process by mapping each unique category to a specific integer consistently across different datasets or instances of the model.

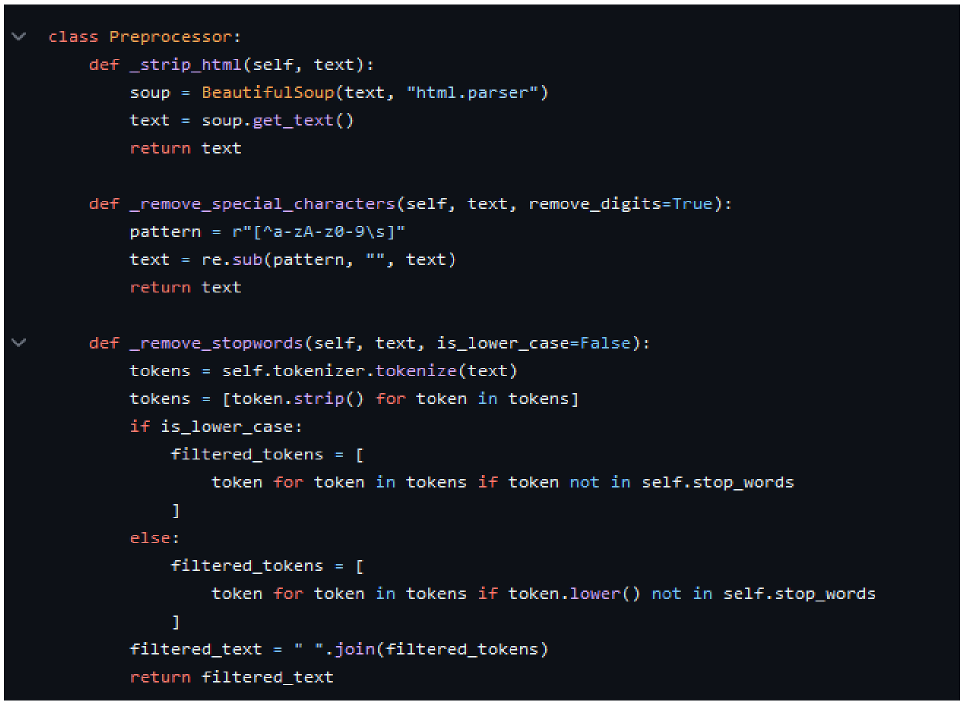
Figure 2.3.1-1 shows the implemented code that is used to handle categorical data in the data set using LUTLabelEncoder.



```python
class LUTLabelEncoder:
    def __init__(self, labels: List[str]) -> None:
        self.lut = labels

    def transform(self, df: pd.DataFrame) -> np.array:
        enc_lbls = df.apply(lambda st: self.lut.index(st)).to_numpy()
        return enc_lbls

    def inverse_tranform(self, labels: List[int]) -> List[str]:
        labels = [self.lut[lbl] for lbl in labels]
        return labels
```

Figure 2.3.1-1 Label Encoding using LUTLEncoder

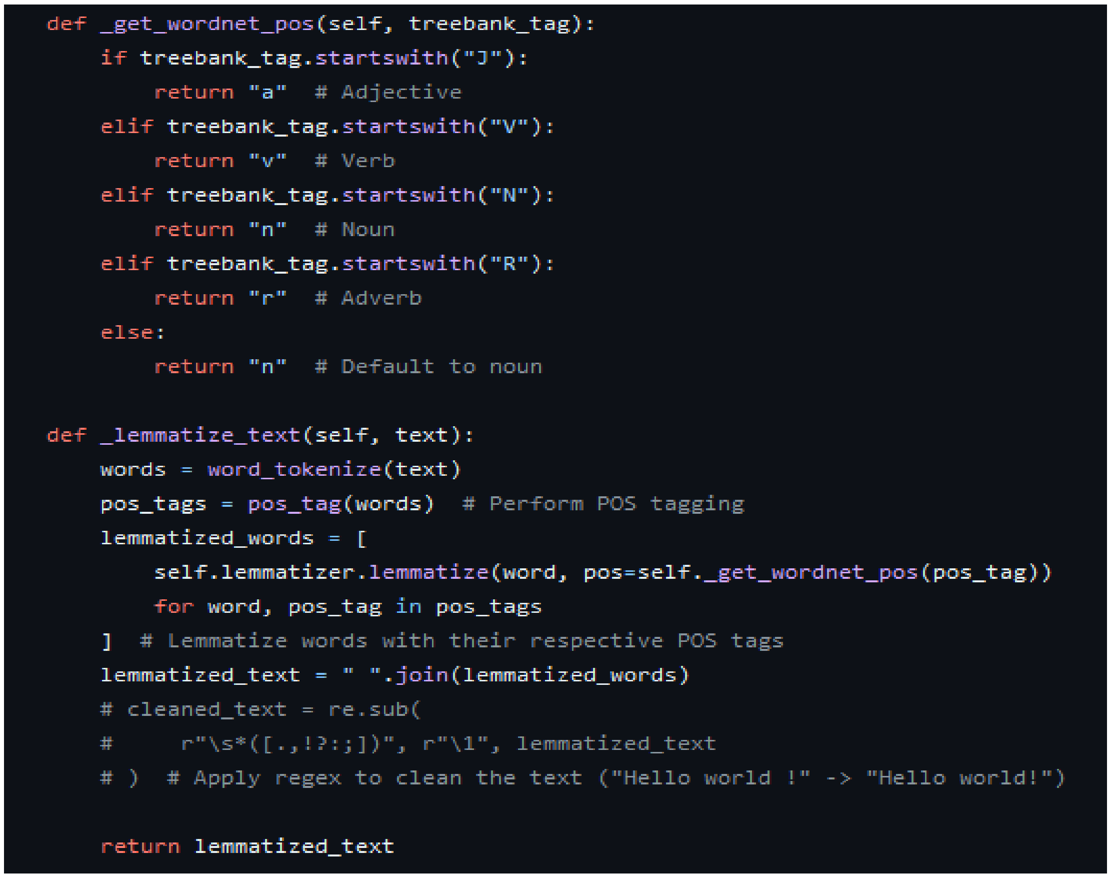
Next, I removed all the HTML tags from the text by using the BeautifulSoup library. Then removed the special characters and stopwords in the text. punctuation marks and symbols are some examples of special characters and they are irrelevant for text analysis tasks. Stopwords are the common words in a language (e.g., "the," "and," "is") and they also don't carry significant meaning for text analysis. The below figure 2.3.1-2 includes those pre-processing steps.



```python
class Preprocessor:
    def _strip_html(self, text):
        soup = BeautifulSoup(text, "html.parser")
        text = soup.get_text()
        return text

    def _remove_special_characters(self, text, remove_digits=True):
        pattern = r"[^a-zA-z0-9\s]"
        text = re.sub(pattern, "", text)
        return text

    def _remove_stopwords(self, text, is_lower_case=False):
        tokens = self.tokenizer.tokenize(text)
        tokens = [token.strip() for token in tokens]
        if is_lower_case:
            filtered_tokens = [
                token for token in tokens if token not in self.stop_words
            ]
        else:
            filtered_tokens = [
                token for token in tokens if token.lower() not in self.stop_words
            ]
        filtered_text = " ".join(filtered_tokens)
        return filtered_text
```

Figure 2.3.1-2 Data Preprocessing Steps

Finally, applied the lemmatization and tokenization to the text as well. Apart from that I implemented the WordNet POS Tagging method to map Treebank POS tags to WordNet POS tags within the lemmatization process. WordNet is a lexical database that associates words with their meanings and relationships. It uses specific POS tags (e.g., "n" for nouns, "v" for verbs) to perform lemmatization accurately. The figure 2.3.1-3 shows the implemented code for lemmatization and WordNet POS Tagging method.

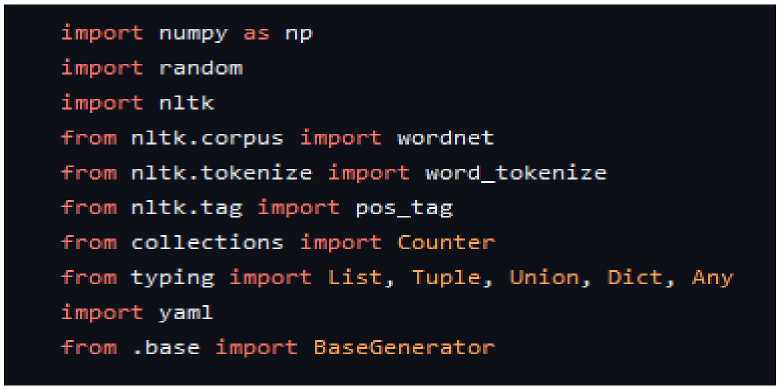


```python
def _get_wordnet_pos(self, treebank_tag):
    if treebank_tag.startswith("J"):
        return "a"  # Adjective
    elif treebank_tag.startswith("V"):
        return "v"  # Verb
    elif treebank_tag.startswith("N"):
        return "n"  # Noun
    elif treebank_tag.startswith("R"):
        return "r"  # Adverb
    else:
        return "n"  # Default to noun

def _lemmatize_text(self, text):
    words = word_tokenize(text)
    pos_tags = pos_tag(words)  # Perform POS tagging
    lemmatized_words = [
        self.lemmatizer.lemmatize(word, pos=self._get_wordnet_pos(pos_tag))
        for word, pos_tag in pos_tags
    ]  # Lemmatize words with their respective POS tags
    lemmatized_text = " ".join(lemmatized_words)
    # cleaned_text = re.sub(
    #     r"\s*([.,!?:;])", r"\1", lemmatized_text
    # )  # Apply regex to clean the text ("Hello world !" -> "Hello world!")

    return lemmatized_text
```

Figure 2.3.1-3 WordNet POS Tagging

Counterfactual Explanation for Support Vector Machine:

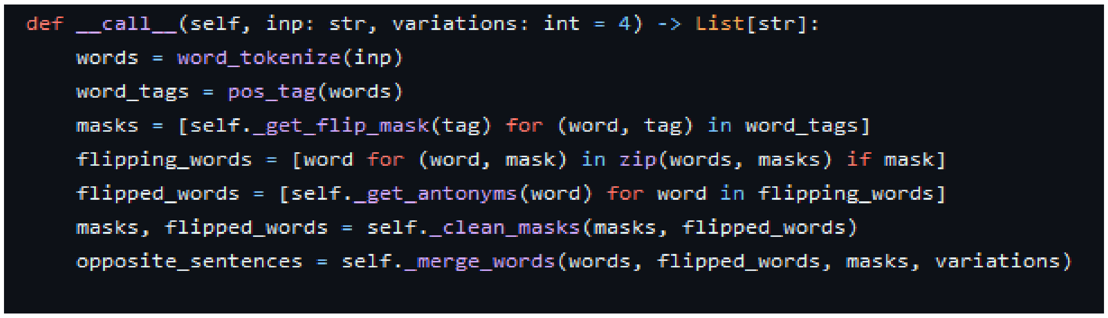
Before the development, required libraries were imported into the environment as shown in figure 2.3.1-4 Here Natural Language Toolkit (NLTK) is a powerful Python library for working with human language data, particularly in the field of natural language processing (NLP) and computational linguistics. It provides a wide range of tools and resources for tasks related to text processing and analysis.



```
import numpy as np
import random
import nltk
from nltk.corpus import wordnet
from nltk.tokenize import word_tokenize
from nltk.tag import pos_tag
from collections import Counter
from typing import List, Tuple, Union, Dict, Any
import yaml
from .base import BaseGenerator
```

Figure 2.3.1-4 Imported Necessary Libraries

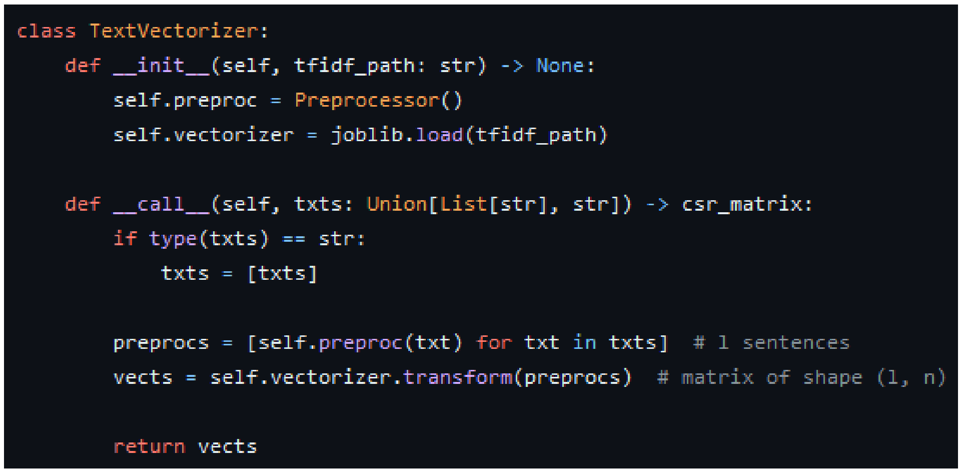
In the word flipping generator algorithm first, it assigns treebank tags for each tokenized word using the pos_tag method in the nltk library. Then generate a mask list by selecting the tags that need to be flipped by the _get_flip_mask method. It returns as true for the tags that need to be flipped and false for other tags. Next algorithm filters the words that are returned with true values referring to the mask list and the _get_antonyms method generates the sorted antonym list for each filtered word. Finally, _merge_words method generates the contradictory prompts using flipped words with original words. The algorithm returns the defined number of contradictory prompts relevant to the original prompt. Figure 2.3.1-5 shows the implemented code segment of word flipping generator algorithm.



```
def __call__(self, inp: str, variations: int = 4) -> List[str]:
    words = word_tokenize(inp)
    word_tags = pos_tag(words)
    masks = [self._get_flip_mask(tag) for (word, tag) in word_tags]
    flipping_words = [word for (word, mask) in zip(words, masks) if mask]
    flipped_words = [self._get_antonyms(word) for word in flipping_words]
    masks, flipped_words = self._clean_masks(masks, flipped_words)
    opposite_sentences = self._merge_words(words, flipped_words, masks, variations)
```

Figure 2.3.1-5 Implementation of Word Flipping Generator

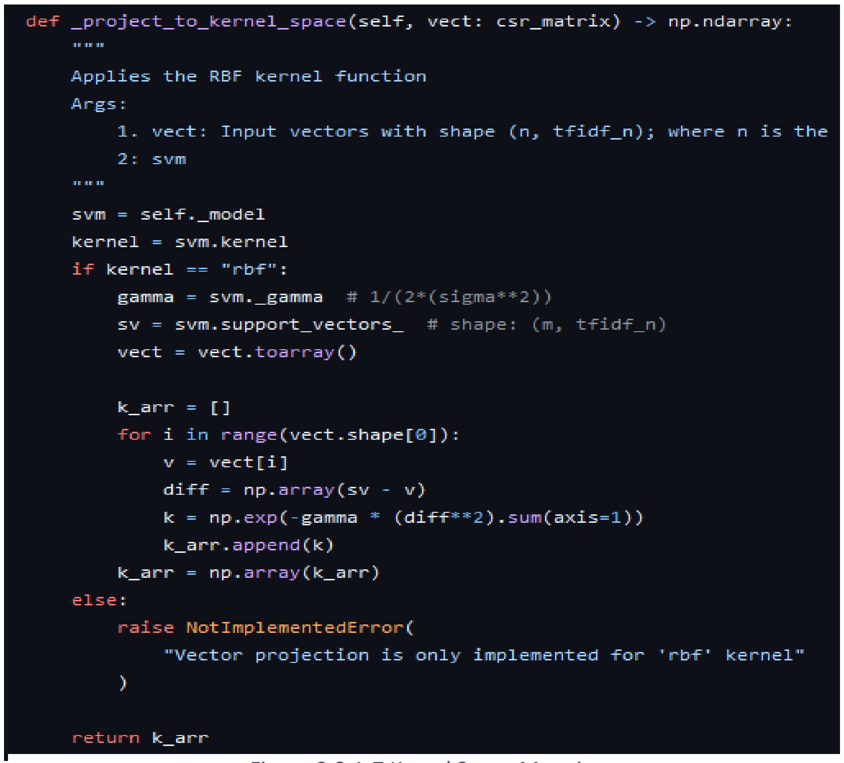
After getting contradictory prompts Text Vectorizer vectorizes all the prompts into the vector space. The code related to text vectorizer implementation is shown in figure 2.3.1-6



```python
class TextVectorizer:
    def __init__(self, tfidf_path: str) -> None:
        self.preproc = Preprocessor()
        self.vectorizer = joblib.load(tfidf_path)

    def __call__(self, txts: Union[List[str], str]) -> csr_matrix:
        if type(txts) == str:
            txts = [txts]

        preprocs = [self.preproc(txt) for txt in txts]  # 1 sentences
        vects = self.vectorizer.transform(preprocs)  # matrix of shape (1, n)

        return vects
```

Figure 2.3.1-6 Text Vectorization

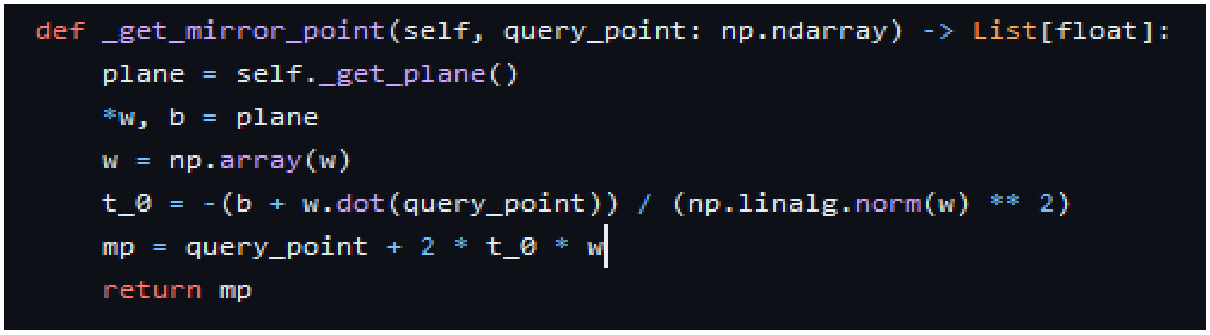
Next, I implemented the _project_to_kernel_space method to project all the vectors into the SVMs' kernel space. Here I applied the RBF kernel with the gamma and support vectors. The figure 2.3.1-7 shows the implementation of mapping to the kernel space.



```python
def _project_to_kernel_space(self, vect: csr_matrix) -> np.ndarray:
    """
    Applies the RBF kernel function
    Args:
        1. vect: Input vectors with shape (n, tfidf_n); where n is the
        2: svm
    """
    svm = self._model
    kernel = svm.kernel
    if kernel == "rbf":
        gamma = svm._gamma  # 1/(2*(sigma**2))
        sv = svm.support_vectors_  # shape: (m, tfidf_n)
        vect = vect.toarray()

        k_arr = []
        for i in range(vect.shape[0]):
            v = vect[i]
            diff = np.array(sv - v)
            k = np.exp(-gamma * (diff**2).sum(axis=1))
            k_arr.append(k)
        k_arr = np.array(k_arr)
    else:
        raise NotImplementedError(
            "Vector projection is only implemented for 'rbf' kernel"
        )

    return k_arr
```

Figure 2.3.1-7 Kernel Space Mapping

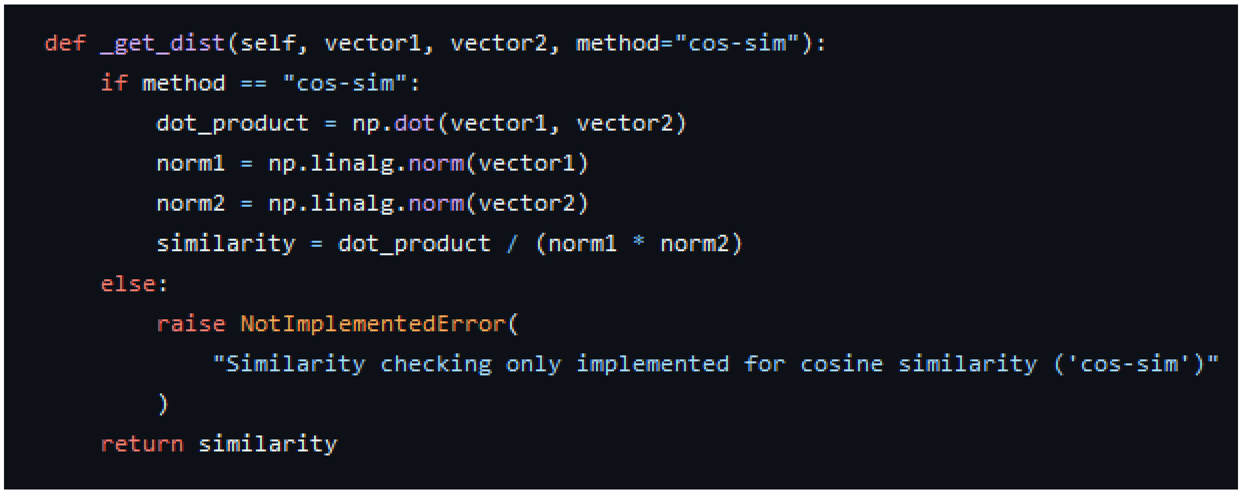
_get_plane method extracts the parameters of SVM from the linear function after transforming to the kernel space. Then _get_mirror_point method calculates the mirror point of the original prompt using the parameter of SVM. The implementation of mirror point calculation is shown in the figure 2.3.1-8



```python
def _get_mirror_point(self, query_point: np.ndarray) -> List[float]:
    plane = self._get_plane()
    *w, b = plane
    w = np.array(w)
    t_0 = -(b + w.dot(query_point)) / (np.linalg.norm(w) ** 2)
    mp = query_point + 2 * t_0 * w
    return mp
```

Figure2.3.1-8 Mirror Point Calculation

Finally, the _get_dist method calculates the distances between contradictory prompts and the mirror point of the original prompt using cosine similarity calculation. _get_dist method returns the contrastive review which has the minimum distance to the mirror point. The figure 2.3.1-9 shows the implementation of the _get_dist method.



```python
def _get_dist(self, vector1, vector2, method="cos-sim"):
    if method == "cos-sim":
        dot_product = np.dot(vector1, vector2)
        norm1 = np.linalg.norm(vector1)
        norm2 = np.linalg.norm(vector2)
        similarity = dot_product / (norm1 * norm2)
    else:
        raise NotImplementedError(
            "Similarity checking only implemented for cosine similarity ('cos-sim')"
        )
    return similarity
```

Figure 2.3.1-9 Co-Sine Similarity Calculation

2.3.2 Testing

As the final step of the development process, had to test the novel explanation method. The performance of the method is evaluated by testing each of the explanation steps by providing the necessary inputs to return the specified output. Afterward, the integrated system was tested separately by sending the inputs through the built user interface. Further, have to evaluate the accuracy of the novel method by comparing the outputs with existing methods as well. Through that testing process, we can identify the drawbacks of the novel method and improve the solution by fixing those drawbacks.

Test Plan and Test Strategy

Testing is carried out to ensure the effectiveness and accuracy of the novel method. For the testing process, it is important to have a proper test plan. A test plan consists of the scope of testing, objectives for tracking the progression of the project, and the list of the tasks that are to be tested. When we consider the testing strategies it provides a roadmap for how testing activities will be conducted, what will be tested, how testing resources will be allocated, and the goals that need to be achieved through the testing process.

Steps of test strategy:

- Define the items to be tested
- Select the functions based on the importance and risk to the user
- Design test cases as identified by the use case description
- Execute the testing
- Record the results of the conducted tests
- Identifying the bugs
- Correcting the bugs
- Repeat the test case until the output results are same as the expected results.

Test case Design

The test cases mentioned below were created to assure the accuracy of the novel counterfactual explanation related to the SVM classifier.

Table 2.3.2-1 Test case to check the performance of the word flipping generator with verbs and adjectives related POS tags

Test Case Id	01
Test Case	Check the performance of the word flipping generator with verbs and adjectives related POS tags
Test Scenario	Check the word flipping generator, flip the user-defined POS tags accurately. Here the user specified the adjectives and verbs related POS tags need to be flipped
Input	If you like original gut-wrenching laughter, you will like this movie. If you are young or old then you will love this movie, hell even my mom liked it.
Expected Output	If you don't like original gut-wrenching laughter, you will dislike this movie. If you are young or old then you will hate this movie, hell even my mom dislike it.
Actual Result	If you don't like original gut-wrenching laughter, you will dislike this movie. If you are young or old then you will hate this movie, hell even my mom dislike it.
Status (Pass/Fail)	Pass

Table 2.3.2-2 Test case for check the performance of the word flipping generator for different POS tags

Test Case Id	02
Test Case	Check the performance of the word flipping generator with nouns, verbs, adjectives and adverbs related POS tags
Test Scenario	Check the word flipping generator, flip the user-defined POS tags accurately. Here the user specified nouns, verbs, adjectives and adverbs related POS tags need to be flipped
Input	This was an interesting movie for the once who loves action movies.
Expected Output	This was a bore movie for the once who not loves action movies.
Actual Result	This was a bore movie for the once who not loves action movies.
Status (Pass/Fail)	Pass

Table 2.3.2-3 Test case for validate the cosine similarity calculation

Test Case Id	03
Test Case	Validate the cosine similarity calculations.
Test Scenario	Check the algorithm accurately calculates the cosine similarities between the original prompt and contradictory prompts.
Input	I like that movie because it provides more facts about world cultures. Also, surely it should be a better experience for the ones who are interested in finding historical facts. **1st contradiction prompt -** I **do not like** that movie because it provides **few** facts about world cultures. Also, surely it **should not** be a better experience for the ones who are interested in finding historical facts. **2nd contradiction prompt** - I **dislike** that movie because it provides **less** facts about world cultures. Also, surely it should be a **worse** experience for the ones who are interested in finding historical facts.
Expected Output	**1st contradiction prompt-** 0.999756352625204 **2nd contradiction prompt-** 0.9996472627085583
Actual Result	**1st contradiction prompt-** 0.999756352625204 **2nd contradiction prompt-** 0.9996472627085583
Status (Pass/Fail)	Pass

Table 2.3.2-4 Test case for Verify the overall final output of the explanation method

Test Case Id	04
Test Case	Verifying the overall final output of the explanation method.
Test Scenario	Check the algorithm return the contradictory prompt which has the minimum distance to the mirror point of the original prompt.
Input	I like that movie because it provides more facts about world cultures. Also, surely it should be a better experience for the ones who are interested in finding historical facts. Contradictory prompts – **1st contradiction prompt -** I **do not like** that movie because it provides **few** facts about world cultures. Also, surely it **should not** be a better experience for the ones who are interested in finding historical facts.(0.999756352625204) **2nd contradiction prompt** - I **dislike** that movie because it provides **less** facts about world cultures. Also, surely it should be a **worse** experience for the ones who are interested in finding historical facts.(0.9996472627085583)
Expected Output	**2nd contradiction prompt** - I **dislike** that movie because it provides **less** facts about world cultures. Also, surely it should be a **worse** experience for the ones who are interested in finding historical facts.
Actual Result	**2nd contradiction prompt** - I **dislike** that movie because it provides **less** facts about world cultures. Also, surely it should be a **worse** experience for the ones who are interested in finding historical facts.
Status (Pass/Fail)	Pass

Table 2.3.2-5 Test case for validate the flipping probability of word flipping generator

Test Case Id	05
Test Case	Validate the flipping probability of word flipping generator.
Test Scenario	Check the defined POS tags are entirely flipped when we assign 1 as the flipping probability.
Input	I don't like this movie because it has violence in the content. It is not suitable for small children.
Expected Output	I am happy with this movie because it has peacefulness in the content. It is good for small children
Actual Result	I am happy with this movie because it has peacefulness in the content. It is good for small children
Status (Pass/Fail)	Pass

3. RESULTS AND DISCUSSION

3.1 Results

The results evaluation process was conducted in two steps. First, evaluate the results of novel counterfactual explainable solution of support vector classifier. Then had to compare proposed novel solutions with the existing methods. For that, we selected the novel explanation method of the RF model from the implemented novel methods. To evaluate the performance of the novel RF explanation method, we compared it to the SEDC method that initiated the implementation of the novel RF explanation.

To provide novel explanation solutions we utilize the IMDB movie reviews dataset. When we consider the sentimental analysis of the above dataset, it classifies the reviews as positive and negative labels. Therefore, we evaluated the accuracy of the SVM explanation and the performance of the RF explanation through both positive and negative reviews.

3.1.1 Step 1 - Results evaluation of the SVM classifier counterfactual explanation.

Positive Review:

I like that movie because it provides more facts about world cultures. Also, surely it should be a better experience for the ones who are interested in finding historical facts. [class label – positive]

According to the novel counterfactual explanation of SVM first, we have to input the positive review to the custom word flipping generator by specifying the POS tags and the number of contradictory prompts needed. Here specify the verbs and adjectives as the POS tags and two as the variation count of needed contradictory prompts. Then custom word flipping generator returns the following two contradictory prompts related to the above positive review.

1st contradiction prompt - I **do not like** that movie because it provides **few** facts about world cultures. Also, surely it **should not** be a better experience for the ones who are interested in finding historical facts.

2nd contradiction prompt - I **dislike** that movie because it provides **less** facts about world cultures. Also, surely it should be a **worse** experience for the ones who are interested in finding historical facts.

After the contradictory prompt generation, the algorithm compares the co-sine similarity between each contradictory prompt with the mirror point of the original prompt. 0.99975, 0.99964 are the co-sine similarity scores for the above-mentioned contradictory prompts respectively. Finally, the algorithm returns the 2nd contradictory prompt which is the closest contradictory prompt to the mirror point of the original input review as the most accurate contradiction. From that, we can determine “like”, “more” and “better” are the most influential features in providing the counterfactual explanation.

<u>*Negative review:*</u>

I don't like this movie because it has violence in the content. It is not suitable for small children.

Same as the positive review first we have to input the review to the word flipping generator with mentioning verbs and adjectives as the pos tags and three as the variation count. The custom word-flipping generator returns the following three contradictions related to the positive review.

1st contradiction prompt - I **like** this movie because it has **no** violence in the content. It **is** suitable for small children.

2nd contradiction prompt - I **am satisfied** with this movie because it has **kindness i**n the content. It **is** suitable for small children.

3rd contradiction prompt - I **am happy** with this movie because it has **peacefulness** in the content. It is **good** for small children.

As the next step algorithm compares the cosine similarity scores and it returns 0.99933, 0.99944, 0.99929 as the cosine similarity scores of the above contradictions respectively. Finally, it returns the 3rd contradiction as the most accurate contradiction which is the closest contradiction to the mirror point. From that, we can determine “don’t like”, “violence” and “not suitable” are the most influential features in providing the counterfactual explanation

3.1.2 Step 2 - Results evaluation of the RF counterfactual explanation.

RF's novel methodology provides instance-specific counterfactual explanation based on the feature importance values that are calculated from the Gini impurity values. To evaluate the performance of the novel RF explanation method, we compared it to the SEDC method that initiated to the implementation of the novel RF explanation. The SEDC provides a method to identify counterfactual explanations in textual data based on prediction scores of the output.

Result Evaluation for the Negative reviews:

We applied the same negative review for both the novel method and the SEDC method. The class change of the two methods related to the applied negative review is visualized using the multiple-line graph. The y-axis of the graph shows the prediction score, and the x-axis shows the iteration number. The blue and green lines represent the class changes of the novel method and SEDC method respectively. The Orange line indicates the threshold value, 0.493. This threshold value is a predefined value that is used in both SEDC and novel methods. In both SEDC and novel methods to occur a class change to positive the prediction score must be greater than the threshold value for the negative reviews.

Negative Review 1:

“What can I say? I ignored the reviews and went to see it myself. Damn the reviews were so right. What a waste of money considering its budget. Good thing, I went to see Kill Bill after this one. To see a really scary movie, would be Crossroads! I like "Girl in Gold Boots" better than this crap.” [class label – negative]

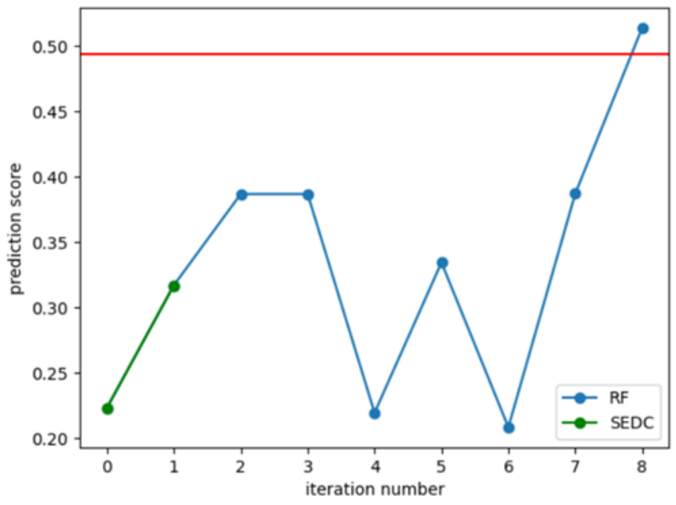


Figure 3.1.2 -1 Class change for negative review 1 RF explanation Vs SEDC

According to figure 3.1.2-1, the initial prediction score of the novel method is 0.223, which means it is a negative review (threshold > 0.22). First start the removal process according to the “instance-specific counterfactual explanation using feature importance in RF model”. Then continue this removal process by rechecking the prediction score each time, until the score goes up to the predefined threshold value. According to the above graph, there is a class change into positive in the 8th iteration and the final score is mentioned as 0.528. However, even after the maximum number of iterations, there is no class change from negative to positive in the SEDC method. In Figure 1, the green line ends between the 0.47 and 0.48 prediction scores and does not exceed the threshold value.

The instance-specific counterfactual explanation for random forest returns the most affecting words to provide a counterfactual explanation. For the above-mentioned review the explanation returns "ignore", "waste", "kill" and "crap" words. Therefore, those features mostly affected to the label change from negative to positive.

Negative Review 2:

This film might have bad production values, but that is also what makes it so good. The special effects are gross out and well done. Robert Prichard as the leader of the gang is hilarious, as are the other members. This film is actually trying to make a point, by saying that nuclear waste plants are bad. 4/10 Fair comedy, gross out film.

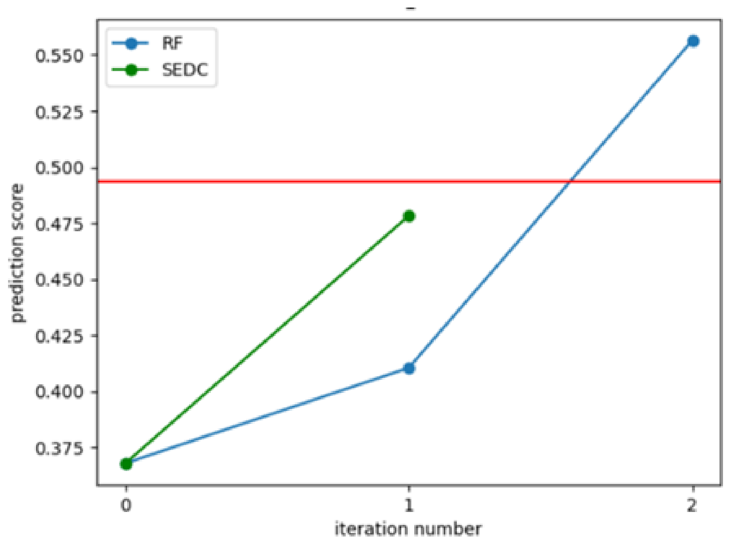


Figure 3.1.2-2 Class change for negative review 2 RF explanation Vs SEDC

According to figure 3.1.2-2, the initial prediction score of the novel method is 0.368 which means it is a negative review (threshold > 0.37). As shown in the above graph, there is a class change into positive in the 2nd iteration and the final score is mentioned as 0.556. In the SEDC method, like the previous negative review 1, even after the maximum number of iterations, there is no class change from negative to positive for the negative review 2. Also, the green line ends between the 0.46 and 0.48 prediction scores and does not exceed the threshold value as well. For the above-mentioned review, the explanation returns "bad" and "waste" words. Therefore, those features most affected to the label change from negative to positive.

Further, we compared the novel method with the SEDC method using another set of negative reviews, and the following graphs show the class label changes that occurred in the novel method and SEDC method related to those negative reviews. According to those below figures, we can realize the SEDC method does not provide any class label change from negative to positive. However, the novel RF explanation method always provides a class change from negative to positive.

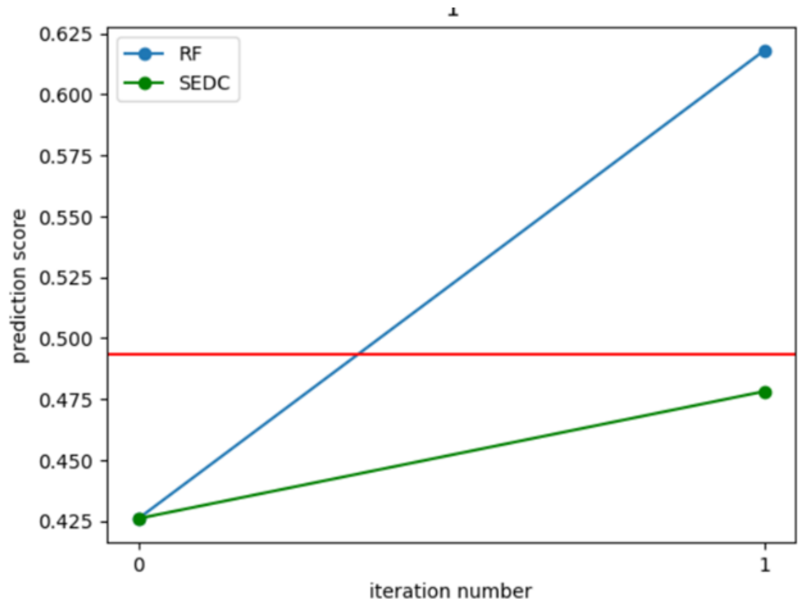


Figure 3.1.2-3 Class change of negative review 3 RF explanation Vs SEDC

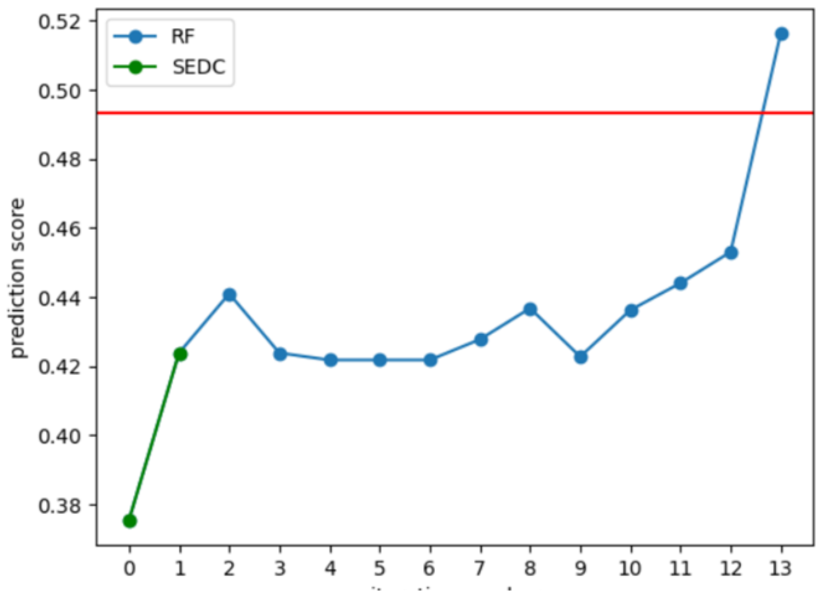


Figure 3.1.2-4 Class change of negative review 4 RF explanation Vs SEDC

Result Evaluation for the Positive reviews:

Here we applied the same positive review for both the novel method and the SEDC method. The class change of the two methods related to the applied positive review is visualized using the multiple-line graph. Same as the negative reviews' evaluation, we get the threshold value as 0.493. In both SEDC and novel methods to change the class label to negative the prediction score must be less than the threshold value for the positive reviews.

<u>*Positive Review 1:*</u>

This movie is one of the funniest, saddest and most accurate portrayals of the mentality that seems to have pervaded the Balkans yet again, 45 years after the time depicted. All the usual characters and conflicts are presented with such anger, sadness and love combined that it is impossible to decide whether crying or laughing would be the more appropriate response. The accuracy of portrayal and the timelessness of the types, however, make it for a great film to watch if one wants to understand a little bit of what drove ex-Yugoslavia to its madness. In fact, no diplomat dealing with the region should attempt anything until they saw this movie, and its twin, *Maratonci trce pocasni krug.* Did I mention it is one of the funniest movies I've ever seen?

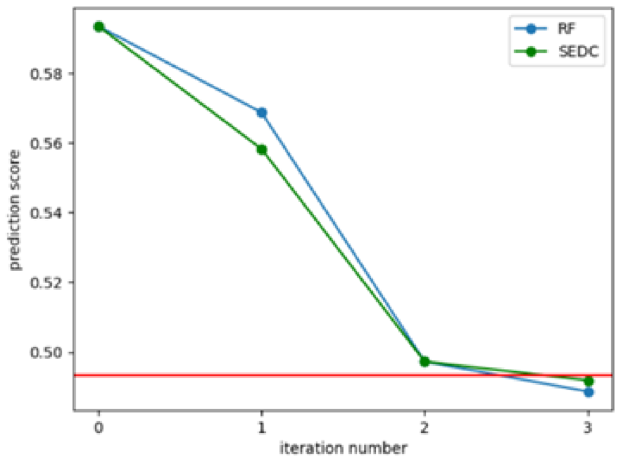


Figure 3.1.2-5 Class change of positive review 1 RF explanation Vs SEDC

According to figure 3.1.2-5, the initial prediction score of the novel method is 0.593 which means it is a positive review (threshold < 0.59). After input the positive review to the instance specific counterfactual explanation, it follows the feature removal process according to the feature importance.

This removal process is continued by rechecking the prediction score each time until the score is less than the predefined threshold value. According to the above graph, there is a class change into negative in the 3rd iteration and the final score is mentioned as 0.488. As shown in Figure 1, When we apply the SEDC method for the above positive reviews it also provides class change to negative after 3 iterations. The prediction score value is less than the threshold value,0. 493. For the above-mentioned review, the explanation returns "funny", "great", "love" and "laugh" words. Therefore, those features most affected to the label change from positive to negative.

Positive Review 2:

I like Steven Seagal but I have not a CLUE what this movie was about. I am not easily lost with movies but I had no idea who was on who's side. Hopefully some of his future movies will be a little better. My wife still thinks he looks good in black jeans, however.

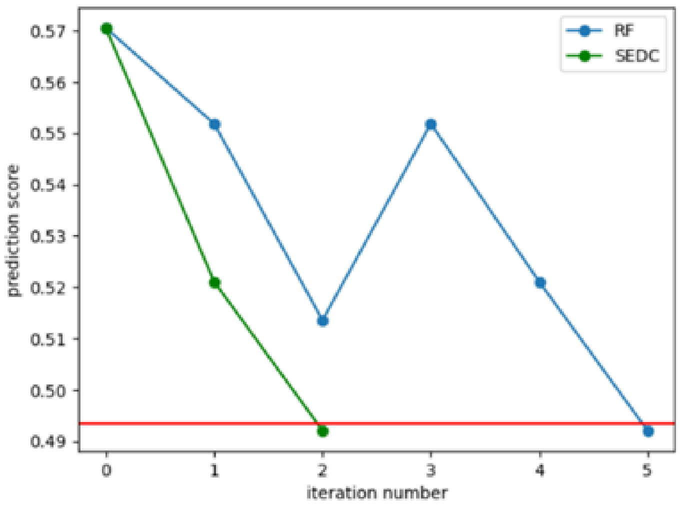


Figure 3.1.2-6 Class change of positive review 2 RF explanation Vs SEDC

According to figure 3.1.2-6, the initial prediction score of the novel method is 0.571 which means it is a positive review (threshold < 0.57). As shown in the above graph, there is a class change into the negative in the 5th iteration and the final score is mentioned as 0.4920. When we apply the SEDC method for the above positive reviews it also provides class change to negative after 2 iterations. The prediction score value is less than the threshold value,0. 4933. For the above-mentioned review, the explanation returns "better" and “good” words. Therefore, those features most affected to the label change from positive to negative.

Positive Review 3:

Garde Ã Vue has to be seen a number of times in order to understand the sub-plots it contains. If you're not used to french wordy films, based upon conversation and battle of wits rather than on action, don't even try to watch it. You'll only obtain boredom to death, and reassured opinion that french movies are not for you. Garde Ã Vue is a wordy film, essentially based upon dialogs (written by Audiard by the way) and it cruelly cuts the veil of appearances. Why does MaÃ®tre Martineau (Serrault) prefer to be unduly accused of being a child murderer rather than telling the truth ? Because at the time of the murder he was with an 18 years old girl with which he has a 8-years sexual relation. His wife knows it, she's jealous of it and he prefers to be executed (in 1980 in France, there was still death penalty) rather than unveiling the sole "pure and innocent" aspect of his pitiful life.

According to figure 3.1.2-7, the initial prediction score of the novel method is 0.595 which means it is a positive review (threshold < 0.59). After applying the counterfactual explanation for the above positive review, it does not provide any class change even 24 iterations occurred. However, the SEDC method changes the class label to negative after the 3rd iteration. In that, we can realize the novel counterfactual method of RF does not provide class change to negative in some situations.

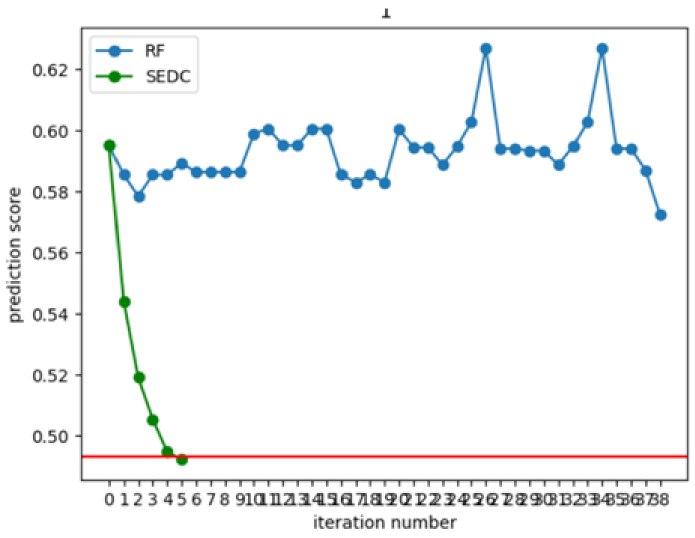


Figure 3.1.2-7 Class change for positive review 3 RF explanation Vs SEDC

3.2 Discussion

As discussed in detail in the introduction section, as a solution for the black box models explainability problem, the concept of explainable AI is being raised as a hot topic and it is the next step of AI-based decision-making systems. In this research study, the main objective is to provide a novel XAI solution to enhance the model interpretability of the function-based classification models focus on SVM by developing a novel counterfactual rule generation mechanism related to the text classification domain. The proposed method contains an algorithm that implements a custom word-flipping generator to get the specified number of contradictions for the input prompt and return the most accurate contradiction to provide the counterfactual explanation. To return the most accurate contradiction prompt algorithm calculates the mirror point of the original review and compares the cosine similarities of each contradiction with that mirror point.

However, this research study does not conclude here and it remains a host of possibilities for future enhancements and directions. One is the implementation of a dynamic selection of POS Tags and contradiction variations. Currently, the proposed method relies on users to define the necessary POS tags and the number of needed contradiction variations for generating contradiction prompts. Future work can focus on algorithmic improvements to automatically discern the most relevant POS tags and the optimal number of counterfactuals for comparison. This refinement is expected to enhance the accuracy of the proposed method, as the ultimately returned contradiction will be the most precise and tailored one to the specific instance.

Another future improvement is including advanced visualizations to provide counterfactual explanations for end users in a more understandable way. Presently, our XAI solutions lack comprehensive visualizations to facilitate user interaction. Future research can concentrate on delving into visualization techniques derived from the fields of User Interface (UI) and User Experience (UX) design. By incorporating effective visualization strategies, we can enable more accessible and user-friendly interactions with our XAI solutions, thereby further augmenting their practicality and user adoption.

The promising results achieved thus far serve as a stepping stone for above mentioned future refinements, promising more precise and user-friendly XAI solutions to meet the growing demands of an AI-driven world.

4. SUMMARY

Member	Component	Task
Warnasooriya S.D (IT20097660)	Provide a novel counterfactual explanation for Support Vector Machine classifier related to the text classification domain. (Custom word flipping generator implementation and cosine similarity comparison based counterfactual explanation)	• Prepare the data set and implement Support Vector Machine classifier. • Generate a novel counterfactual rule-based mechanism using text classification domain. ➢ Implement the custom word flipping generator. ➢ Vectorize all the prompts using a TFIDF vectorizer and map in to the kernel space. ➢ Calculate the mirror point of the original input review. ➢ Calculate the cosine similarity between each contradiction with the mirror point and return the closest contradiction to the mirror point. • Results evaluation conduct for the SVM's novel explanation and RF's novel explanation. Here RF's explanation compares with the SEDC method. • Frontend implementation related to the SVM's explanation.
Srinidee Methmal H.M (IT18161298)	Provide a novel counterfactual explanation for Logistic Regression related to the text classification domain. (Antonym replacement based counterfactual explanation)	• Prepare the data set and implement Logistic Regression classifier. • Develop a novel counterfactual rule-based mechanism using text classification domain. ➢ Vectorize all the text using TFIDF vectorizer ➢ Antonym selection and iteration replacement and removal

		➢ Get the prediction score of instances and classify the class label ➢ Get the feature importance of each feature and sort • Results evaluation conduct for the LR's novel explanation and RF's novel explanation. Here RF's explanation compares with the SEDC method Frontend • Frontend implementation related to the LR's explanation.
Britto T.A (IT201009698)	Provide a novel counterfactual explanation for Random Forest model related to the text classification domain. (Instance specific counterfactual explanation)	• Prepare the data set and implement Random Forest model. • Generate a novel counterfactual rule-based mechanism using text classification domain. ➢ Vectorize all the text using TFIDF vectorizer. ➢ Extract the feature importance and removed the features that are not related to the given instance. ➢ Get the prediction score and classify the class labels ➢ Get the feature importance and sort ➢ Remove the most impact features iteratively until get a class change. • Results evaluation conduct for the RF's novel explanation. Here RF's explanation compares with the SEDC method Frontend • implementation related to the RF's explanation

Lakshani N.V.M (IT20013950)	Provide a novel counterfactual explanation for K nearest neighbour model related to the text classification domain. (Feature density comparison based counterfactual explanation)	• Prepare the data set and implement K-nearest Neighbour classifier. • Generate a novel counterfactual rule-based mechanism using text classification domain. ➢ Implement the custom word flipping generator. ➢ Vectorize all the prompts using a TFIDF vectorizer and map in to the kernel space. ➢ Calculate the probability scores. ➢ Calculate the neighbour statistics and return the best counterfactual. • Results evaluation conduct for the KNN's novel explanation. Here RF's explanation compares with the SEDC method Frontend • implementation related to the KNN's explanation.

Table 4-1Summary of each member's contribution

5. CONCLUSION

This research has delved into the pressing challenge of enhancing model explainability within AI-driven decision-making systems, with a particular focus on KNN, LR, RF, and SVM models within the context of text classification. The impetus for this study has arisen from the rapidly growing utilization of AI in industries where transparency and explainability of AI-based decisions are very important. As a solution for the discussed explainability problem, this research study provides novel explainable methods for the above-mentioned ML models using a counterfactual rule generation mechanism. when we consider the proposed methods, it includes custom word flipping generator implementation and cosine similarity comparison-based mechanism for SVM, feature density comparison-based mechanism for KNN, antonym replacement-based explanations for LR, and instance-specific counterfactuals for RF. In the end, all of these explanations return the features that are mostly affected to provide the counterfactual explanation.
The implementation of the counterfactual explanations for each model is accomplished through the process which contains model development, encountering the novel XAI solution, and developing the GUI for end users. Backend implementations are done in the Google Colab and Visual Studio Code using Python and the frontend is implemented using NestJs with Mantine UI. Further backend is hosted using an AWS cloud.

When considering the results evaluation, have to evaluate novel proposed explainable methods with the existing methods to improve the novel XAI solutions more. Here we selected an instance-specific counterfactual explanation of the RF method to conduct the result evaluation process from the implemented novel methods. To evaluate the performance of the novel RF explanation method, we compared it to the SEDC method that initiated to the implementation of the novel RF explanation. Through the result evaluation process, identified the SEDC method does not provide any class change to positive but the proposed method provides that. Moreover, in some situations, the novel proposed method does not provide any class change to negative but the SEDC method provides it. Future work related to this research study includes the implementation of a dynamic selection of POS Tags and contradiction variations related to the SVM's novel explanation, the development of advanced visualizations to provide counterfactual explanations for end users in a more understandable way, and the improvement of the instance-specific explanation of RF to provide class changes to negative also in any situation.

6. REFERENCES

[1] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, vol. 13-17-Augu, pp. 1135–1144, doi: 10.1145/2939672.2939778.

[2] Miller, Tim. "Explanation in artificial intelligence: Insights from the social sciences." arXiv Preprint arXiv:1706.07269. (2017).

[3] Kim, Been, Rajiv Khanna, and Oluwasanmi O. Koyejo. "Examples are not enough, learn to criticize! Criticism for interpretability." Advances in Neural Information Processing Systems (2016).

[4] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, 2018, doi: 10.1109/ACCESS.2018.2870052.

[5] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Section 2, pp. 4766–4775, 2017.

[6] F. Shakerin and G. Gupta, "White-box Induction from SVM Models: Explainable AI with Logic Programming," *Theory Pract. Log. Program.*, vol. 20, no. 5, pp. 656–670, 2020, doi: 10.1017/S1471068420000356.

[7] M. T. Ribeiro and C. Guestrin, "Anchors : High-Precision Model-Agnostic Explanations," pp. 1527–1535.

[8] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local rule-based explanations of black box decision systems," *arXiv*, no. May, 2018.

[9] R. K. Mothilal and C. Tan, "Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations.", 2019

[10]Brughmans, D., Leyman, P., & Martens, D. (2023). Nice: an algorithm for nearest instance counterfactual explanations. Data Mining and Knowledge Discovery, 1-39.

[11] Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., ... & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion*, *58*, 82-115

[12] Ramon, Y., Martens, D., Provost, F., & Evgeniou, T. (2020). A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: SEDC, LIME-C and SHAP-C. Advances in Data Analysis and Classification, 14, 801-819.