

INT2X: EXPLANATION FOR THE CAUSES OF A PREDICTION

2023 - 42



Thesis

Lakshani N.V.M.

IT20013950

B.Sc. (Hons) Degree in Information Technology Specialized in Software
Engineering


Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

March 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.

Name	Student ID	Signature
Lakshani N.V.M.	IT20013950	

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

.....
Signature of the Supervisor

.....
Date

Abstract

In recent years, as machine learning technologies have advanced, more decision-making processes are becoming automated and AI-based. This has raised concerns in communities about the lack of transparency and understanding behind these AI-driven decisions. The issue of explainability in AI decision-making became particularly important following the introduction of the General Data Protection Regulation (GDPR) by the European Union in 2016. GDPR emphasizes the right of consumers to understand how black box models make predictions. To address this explainability problem, there is a growing interest in the concept of Model Interpretability, which is seen as the next step in AI development. Despite its popularity, there have not been as many studies in this area as one might expect. This research aims to tackle the issue by introducing a new method for improving the interpretability of black box models, specifically in the context of text classification. The proposed approach will focus on enhancing our understanding of function-based classification models, with a particular emphasis on the k Nearest Neighbour (KNN). The goal is to create a unique mechanism for generating counterfactual rules, which will shed light on how these models work and make decisions. The K-Nearest Neighbors (KNN) classifier is widely used for text classification tasks due to its versatility in handling textual data. However, KNN often becomes a black-box model in the context of text classification, primarily due to the curse of dimensionality. The curse of dimensionality refers to the exponential increase in computational complexity as the number of features (dimensions) in the dataset grows. In text classification, where each word or term can be considered a feature, the high-dimensional nature of the data can make KNN's decision-making process difficult to interpret. This complexity is not limited to linear or non-linear separability but is inherent in the high-dimensional feature space of text data. In our research, our primary goal is to address the challenges posed by the curse of dimensionality in text classification with KNN. We are actively working on developing an innovative eXplainable Artificial Intelligence (XAI) solution tailored to textual data, specifically designed to make KNN more interpretable in high-dimensional feature spaces. This research aims to provide insights into KNN's decision-making process, especially in the context of text classification, where the curse of dimensionality is a critical concern for model interpretability.

ACKNOWLEDGEMENT

Regarding the final year project involving an explanation for the causes of a prediction (INT2X), Lakshani 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.

TABLE OF CONTENTS

DECLARATION	2
Abstract	3
ACKNOWLEDGEMENT	4
List of Figures	6
List of Tables	7
List of Abbreviations	8
1. INTRODUCTION	9
1.1. Background	11
1.2. Literature Review.....	13
1.2.1. White-box Induction from SVM Models: Explainable AI with Logic Programming [6].....	15
1.2.2 Anchors: High-Precision Model-Agnostic Explanations [7].....	16
1.2.3 Local rule-based explanations of black box decision systems [8].....	17
1.2.4 Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations [9]	18
1.2.5 Nearest Instance Counterfactual Explanations [10]	19
1.2.6 Sequentially Eliminating Discrediting Counterfactuals (SEDC) [12].....	19
1.2.7 Explaining and improving model behavior with k nearest neighbor representations [13] 20	
1.3. Research Gap	21
1.4. Research Problem	23
1.4.1. High-Dimensional Data	23
1.4.2. Curse of Dimensionality	23
1.5. Research Objectives.....	25
1.5.1. Main Objectives	25
1.5.2. Specific Objectives	25
2. METHODOLOGY	27
2.1. Counterfactual Explanation for K-Nearest Neighbor.....	28
2.1.1. Generating Counterfactuals.....	28
2.1.2. Tokenizing the original review and the generated counterfactuals	29
2.1.2.1. Preprocessing	30
2.1.2.2. Term Frequency (TF).....	31
2.1.2.3. Inverse Document Frequency (IDF)	31
2.1.2.4. TF-IDF Score Calculation.....	31
2.2.3. Get the probabilities of getting positive and negative classification.....	32

2.2.4.	Get Neighbour statistics.....	34
2.2.5.	Define the best Counterfactual.....	36
2.3.	Testing and Implementation.....	40
2.3.1.	Implementation	40
2.3.2.	Testing.....	45
3.	RESULTS AND DISCUSSION	47
3.1.	Results.....	47
3.1.1.	Results evaluation of the KNN classifier counterfactual explanation.....	47
3.1.2.	Results evaluation of the RF counterfactual explanation.	49
	Result Evaluation for the Negative reviews:	49
	Result Evaluation for the Positive reviews:	53
4.	SUMMARY	56
5.	CONCLUSION.....	59
6.	REFERENCES	61

List of Figures

Figure 1: Label Encoding using LUTLEncoder.....	41
Figure 2: Data Preprocessing Steps.....	41
Figure 3: WordNet POS Tagging.....	42
Figure 4: Imported Necessary Libraries	43
Figure 5: Implementation of Word Flipping Generator	43
Figure 6: Text Vectorization	44
Figure 7: Class change for negative review 1 RF explanation Vs SEDC.....	50
Figure 8: Class change for negative review 2 RF explanation Vs SEDC.....	51
Figure 9: Class change of negative review 3 RF explanation Vs SEDC	52
Figure 10: Class change of negative review 4 RF explanation Vs SEDC	52
Figure 11: Class change of positive review 1 RF explanation Vs SEDC	53
Figure 12: Class change of positive review 2 RF explanation Vs SEDC	54
Figure 13: Class change for positive review 3 RF explanation Vs SEDC.....	55

List of Tables

Table 1: Table of Abbreviations 8

Table 2: Test Case 0145

Table 3: Test Case 02 46

List of Abbreviations

Abbreviation	Definition
XAI	Explainable AI
AI	Artificial Intelligence
KNN	K-Nearest Neighbour

Table 1: Table of Abbreviations

1. INTRODUCTION

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 are used for the decision-making process. Decisions made by the AI model are extremely critical in a situation like a medical diagnosis of a disease. It's crucial that both clinicians and patients can understand and validate the reasoning behind AI-driven diagnoses. Here explainable AI (XAI) provides clear, understandable insights into how an AI model arrives at its conclusions. After an AI model makes a prediction (e.g., the likelihood of a patient having a certain disease), it can present the relative importance of each input feature (e.g., patient symptoms, test results) that contributed to the prediction. Also, through the explanation, doctors can get an idea of how the outcome would be different when input features slightly changed. For example, 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.

There is no exact definition for Model Explain ability/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" and "Interpretability is the degree to which a human can consistently predict the model's result". 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 was a particular answer obtained" and "when a particular AI-based system can fail?".

This research mainly focuses on enhancing the model explainability of selected Black-Box type classification models, K Nearest Neighbor. It provides a novel explainable method that related to a text classification The black box model refers to a complex computational algorithm or system that processes inputs to produce outputs, but the internal workings or mechanisms that are responsible for the transformation are not readily understandable or explainable from an external perspective. K-Nearest Neighbors (KNN) can become a black box due to its inherent nature of relying on local similarity measures for prediction. As the dimensionality of data increases, it becomes harder to interpret which specific features are driving the predictions, obscuring the

underlying relationships. Additionally, the choice of the number of neighbors (k) and the distance metric can greatly influence outcomes but understanding their impact might not be straightforward. In complex datasets, KNN's decision boundaries can become intricate, making it challenging to explain why a certain point was classified a particular way. Lastly, KNN lacks a clear model representation, unlike linear regression or decision trees, further contributing to its "black box" perception.

There are many explainable techniques that can be used to provide explainable solutions for Black-Box models. This is where the idea of "counterfactual explanation" comes in. A counterfactual explanation provides insight into a machine learning model's prediction by illustrating a hypothetical scenario wherein specific feature values are altered to achieve a different, desired outcome. In essence, it answers the "what if" questions by demonstrating the minimal changes needed to reverse a prediction.

Counterfactual analysis assists to detect and mitigate bias in 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 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 text classification models. The rules can identify where certain groups are overrepresented or underrepresented and if the model is affected by the group differences.

1.1. Background

In recent years, the field of eXplainable Artificial Intelligence (XAI) has gained significant attention from researchers and the wider community, largely driven by the rapid growth of AI models and their real-world applications. The need for and importance of XAI have become evident as people increasingly seek trustworthy and transparent AI-based decision-making processes. Over the last few years, researchers have proposed various model-explainable methods to address these concerns. However, it's important to note that the knowledge in this area is still evolving, and most techniques are in the research phase. With advancements in XAI, researchers have categorized explanation methods into different approaches, including Local-Global, Model Specific-Model Agnostic, and Post hoc-Intrinsic methods. In the context of K-Nearest Neighbors (KNN) applied to text classification, we will discuss how these approaches are relevant.

When it comes to the Local-Global explanation approach, Local explanation methods aim to provide insights into how the KNN model makes predictions for individual instances or data points. In contrast, Global explanation methods offer a broader understanding of the model's overall functionality and highlight the factors or features that are most crucial for its predictions. For KNN in text classification, examples of Local explanation methods could include techniques like LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations), while Global explanation methods might involve feature importance ranking, Partial dependence plots, and decision trees. In terms of the Model Specific-Model Agnostic approach, Model Specific methods are tailored to explain a specific machine-learning model like KNN. On the other hand, Model Agnostic methods are designed to be applicable across various machine-learning models, irrespective of their type or complexity. These methods aim to provide transparency and insights into model predictions. Furthermore, explanation methods can be categorized as either intrinsic or post-hoc. Intrinsic methods focus on embedding explainability directly into the machine learning model itself, considering the internal structure and parameters of the model. Post-hoc methods, on the other hand, analyze the model after it has been trained. Intrinsic methods are typically suitable for simpler models that can be designed to be inherently interpretable, while post-hoc methods are used for more complex models.

As mentioned earlier, the concept of XAI has been primarily directed towards addressing challenges in explaining black-box models. In the case of KNN, which is known for its effectiveness in text classification, it can also be considered a black box due to certain inherent

complexities and low interpretability. KNN is a popular supervised learning technique used for classification tasks, including text classification, where its ability to handle high-dimensional data efficiently is highly advantageous.

In a typical KNN text classification scenario, the algorithm operates by considering the similarity between data points in a feature space. Each feature represents a term, word, or phrase present in the text data. When classifying a new text document, KNN identifies the k nearest neighbors in this high-dimensional space, where " k " is a user-defined parameter. These neighbors are determined based on the similarity metric, often using methods like cosine similarity or Euclidean distance, which measure how closely the new document aligns with existing documents in the dataset. KNN then assigns a class label to the new document based on the majority class among its k nearest neighbors. This approach works well when the data is relatively low-dimensional and can be effectively separated in the feature space.

However, when dealing with text data and many features (words or phrases), a challenge known as the "curse of dimensionality" emerges. This curse occurs because as the number of features (dimensions) increases, the volume of the feature space grows exponentially. This expansion results in a sparsity issue where data points become increasingly spread out, making it more challenging to find meaningful neighbors.

In the context of KNN for text classification, this means that as the number of unique words or terms in the dataset increases, the feature space becomes extremely large and sparse. Consequently, finding nearest neighbors becomes less effective because most data points are far apart in this high-dimensional space. This sparsity leads to a degradation in the quality of the k nearest neighbors, making it harder for KNN to accurately classify text documents.

To mitigate the curse of dimensionality in KNN for text classification, various techniques are employed. Feature selection or dimensionality reduction methods, such as term frequency-inverse document frequency (TF-IDF) weighting or principal component analysis (PCA), can help reduce the number of features while retaining essential information. These approaches aim to strike a balance between preserving relevant information and reducing the computational complexity of KNN in high-dimensional text data.

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.

("not", "bad") → Positive {"not", "good"} → Negative

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 learns 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.

$\{age \leq 39, race = \text{African-American}, recidivist = \text{True}\} \rightarrow \text{High Risk and}$

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

1.2.7 Explaining and improving model behavior with k nearest neighbor representations [13]

The author of [13] introduces a novel approach for NLP model interpretability and performance enhancement using k Nearest Neighbors (kNN) representations. By employing kNN, it uncovers insights at both individual and dataset levels, identifying influential training examples and exposing spurious associations. This technique improves predictions in tasks like Natural Language Inference (NLI) and bolsters model robustness against adversarial inputs. By leveraging kNN as a backoff strategy, it offers a holistic solution to tackle opacity, artifact identification, and performance issues in deep NLP models. The method involves mapping sequences to normalized hidden representations using a neural network like BERT or RoBERTa, computing kNN based on L2 distances, and utilizing a weighted softmax function to integrate kNN scores with base model predictions. This comprehensive approach enhances both understanding and efficacy in NLP models.

1.3. Research Gap

The main target of this research is to make the KNN 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 KNN 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 feature density comparison mechanism to enhance the explainability of the KNN classifier related to the text classification domain.

1.4. Research Problem

How to get a counterfactual rule generation-based explanation for the k-NN classifier, when it handles Curse of Dimensionality problem in text classification?

K-Nearest Neighbors (KNN) is a simple yet effective machine learning algorithm that has been widely used for classification and regression tasks. One of its key advantages is its transparency, as it is inherently interpretable. However, in certain scenarios, KNN can become a black box model, where its decision-making process becomes complex and challenging to interpret. This transformation of KNN into a black box can occur due to the curse of dimensionality problem. High-dimensional data and the curse of dimensionality are related concepts that describe challenges and issues that arise when dealing with data in spaces with a large number of dimensions.

1.4.1. High-Dimensional Data

High-dimensional data refers to data sets where each data point is described by a substantial number of features or attributes. In such data, the dimensionality is high because there are many dimensions, and each dimension corresponds to a distinct feature or variable. For example, in a dataset of images, each pixel's color intensity might be considered a dimension. In text data, each unique word or term can be treated as a dimension. High-dimensional data can be found in various fields, including image processing, natural language processing, genomics, and more.

1.4.2. Curse of Dimensionality

The curse of dimensionality is a term coined to describe the detrimental effects of having a high number of dimensions in a data set. It encompasses several challenges and consequences:

1.4.2.1. Sparsity of Data

As the number of dimensions increases, the available data points become increasingly sparse in the high-dimensional space. This means that data points are spread out, and most of the space remains unoccupied by data. Consequently, it becomes harder to find meaningful patterns or relationships between data points.

1.4.2.2. Increased Computational Complexity

High dimensionality leads to increased computational complexity in various data analysis tasks, such as distance calculations, clustering, and machine learning. Operations that were efficient in lower-dimensional spaces become significantly more resource-intensive.

1.4.2.3. Overfitting

In high-dimensional spaces, machine learning models, including classifiers and regressors, are more prone to overfitting. This means that models can capture noise in the data rather than the underlying patterns, leading to poor generalization to unseen data.

1.4.2.4. Reduced Model Interpretability

With many dimensions, it becomes challenging to visualize and interpret the data or model results. Human comprehension of data relationships and model behavior becomes increasingly difficult as the dimensionality grows.

1.4.2.5. Data Collection and Storage Costs

High-dimensional data can be expensive to collect, store, and process. The sheer volume of data can strain computational resources and storage capacity.

Addressing the curse of dimensionality often involves techniques such as feature selection, dimensionality reduction, and regularization. These methods aim to reduce the number of dimensions while preserving essential information and mitigating the adverse effects of high dimensionality. Techniques like Principal Component Analysis (PCA), t-SNE, and L1 regularization have proven valuable in managing high-dimensional data and making it more tractable for analysis and modeling.

1.5. Research Objectives

1.5.1. Main Objectives

Provide a novel post-hoc, model-specific, local XAI solution to enhance the model explainability of distance based classification models focus on KNN 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 feature density 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 feature densities between the contradictory and the original text(review) 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 KNN classifier.

To perform the experiments, there should be identified dataset that aligns with the requirements. The study used the IMDb Movie Reviews 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 feature density 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

This research is dedicated to the development of an Explainable Artificial Intelligence (XAI) solution tailored specifically to the K-Nearest Neighbors (KNN) model. The central objective is to enhance the interpretability of the KNN model, particularly in the realm of text classification based on reviews, by introducing a novel approach centered around counterfactual rule generation.

The core innovation of this system lies in its ability to provide comprehensive explanations for the predictions made by the KNN model, offering insights into the rationale behind its classifications. In the context of text classification, especially when dealing with reviews, this becomes exceptionally valuable. Users and stakeholders can gain a deeper understanding of why a particular review received a certain classification, whether positive, negative, or neutral.

The system's operation involves the systematic generation of counterfactual instances, which are alternative data points designed to illustrate how slight modifications to the input features would lead to different model outcomes. These counterfactuals serve as a bridge between the opaque, black-box nature of the KNN model and the need for transparency and interpretability. They enable users to comprehend why a specific review received a particular classification, offering actionable insights for model improvement and decision-making.

In essence, this research aims to empower users, analysts, and developers with the tools they need to navigate and trust the KNN model's outputs in the intricate world of text classification. By unraveling the inner workings of the model and providing intuitive explanations, it fosters transparency and confidence, thereby paving the way for more effective and responsible AI-driven decision-making in the realm of reviews and beyond.

To achieve its objective, this research meticulously follows five key steps. These steps are carefully structured to enhance the interpretability and transparency of the K-Nearest Neighbors (KNN) model, particularly in text classification based on reviews. Each step contributes to the development of a specialized Explainable Artificial Intelligence (XAI) solution, facilitating a clearer understanding of the model's decision-making process and promoting responsible AI applications.

2.1. Counterfactual Explanation for K-Nearest Neighbor

2.1.1. Generating Counterfactuals

In our pursuit of enhancing the interpretability of the K-Nearest Neighbors (KNN) model for text classification, particularly in the domain of reviews, a crucial step in our research methodology involves the generation of counterfactuals. These counterfactuals play a pivotal role in our mission to provide users with transparent and comprehensible insights into the decision-making process of the KNN model.

The process of counterfactual generation is multi-faceted and meticulously designed to deliver meaningful explanations to users. It begins with the tokenization of the original review. Tokenization involves breaking down the text into its constituent words or tokens. This initial step is fundamental in preparing the review for subsequent analysis and manipulation.

Once the review is tokenized, the system proceeds to recognize specific words within the text based on their Part-of-Speech (POS) tags. POS tags represent the grammatical category of a word, encompassing categories such as adjectives, nouns, verbs, and more. This recognition of POS tags is a critical aspect of the process, as it allows us to pinpoint words that can be flipped to their opposites, thereby altering the overall meaning of the review.

To make the process even more user-centric and customizable, we provide users with the ability to select the POS tags they wish to focus on. This customization ensures that the counterfactual generation aligns with the specific aspects of the review that users find most relevant or intriguing. For instance, a user interested in comprehending how adjectives influence the model's decision-making can choose to emphasize adjectives as their selected POS tags.

The heart of the counterfactual generation process lies in the "flipping" of words based on the selected POS tags. "Flipping" here refers to the transformation of words into their antonyms or opposites. This transformation has a profound impact on the overall sentiment and meaning of the review. For example, a word like "good" can be flipped to "bad," or "best" can be transformed into "worst." By employing this technique, we aim to demonstrate how subtle changes in the choice of words can significantly influence the KNN model's predictions.

To ensure that the generated counterfactuals provide diverse and insightful explanations, we introduce an element of randomization into the process. When creating multiple counterfactuals for a given review, using the same flip word repeatedly could limit the depth of insight. To address this, the system randomly selects opposite words from a pool of antonyms associated with the chosen POS tags. This randomness ensures that each counterfactual offers a unique perspective, shedding light on various facets of the model's decision process.

The output of this meticulous process is the creation of multiple counterfactual instances based on the selected POS tags. Each counterfactual represents a modified version of the original review, where words have been flipped according to the chosen grammatical categories. These counterfactuals serve as powerful tools for users and analysts alike, enabling them to grasp how specific word choices and their antonyms influence the KNN model's predictions.

In conclusion, this step in our research methodology underscores our commitment to developing a transparent and user-centric Explainable Artificial Intelligence (XAI) solution for the KNN model. By generating counterfactuals based on user-selected POS tags and incorporating randomization for diversity, we provide valuable insights into the model's decision-making process. These counterfactuals empower users to understand the nuanced factors that contribute to the model's predictions, fostering trust and responsible AI utilization in the domain of review-based text classification.

2.1.2. Tokenizing the original review and the generated counterfactuals

In the context of our research endeavor, the second step plays a pivotal role in preparing the text data for further analysis and interpretation. This crucial stage involves the tokenization of both the original review and the generated counterfactuals, enabling us to transform unstructured textual information into a structured format amenable to computational analysis. A fundamental tool we employ in this process is TF-IDF (Term Frequency-Inverse Document Frequency), an essential natural language processing technique.

TF-IDF, or Term Frequency-Inverse Document Frequency, stands as a cornerstone of natural language processing (NLP). Its primary function is to quantify the importance of individual

words or terms within a document relative to a larger collection of documents, often referred to as a corpus. By doing so, TF-IDF assists in the transformation of textual data into a structured format suitable for computational analysis.

At its core, TF-IDF operates on a simple yet powerful premise – words that appear frequently within a specific document but sparingly across the entire corpus carry significant semantic weight. These words, therefore, are indicative of the document's content and its unique characteristics.

Tokenization, within the framework of TF-IDF, is the process by which we break down the text into individual words or tokens, often referred to as terms. Each term is treated as a distinct unit and is assigned a weight based on its TF-IDF score, which reflects both its importance within the document and its significance across the entire corpus.

To elucidate the tokenization process further, the following steps are important.

2.1.2.1. Preprocessing

Before tokenization, text data typically undergoes preprocessing. Preprocessing is an essential preliminary step in text data analysis, including the tokenization process with TF-IDF. It involves a series of transformations and operations aimed at cleaning and structuring unprocessed text data to make it suitable for computational analysis. In essence, preprocessing acts as a critical data preparation stage, ensuring that the subsequent analyses yield meaningful and accurate results.

Lowercasing: One of the initial steps is converting all text to lowercase. This uniformity ensures that words with different cases (e.g., "The" and "the") are treated as the same term, eliminating inconsistencies in the data.

Tokenization: Tokenization involves splitting the text into individual words or tokens. Each token becomes a distinct unit of analysis. This step is crucial for understanding the composition of the text and is a prerequisite for TF-IDF analysis.

Stop Word Removal: Stop words are frequently occurring words that don't carry significant meaning on their own, such as "and," "the," and "is." Removing these words from the text can improve the efficiency and accuracy of subsequent analyses.

Punctuation Removal: Punctuation marks like periods, commas, and exclamation points are often removed to ensure that they do not interfere with the interpretation of individual terms.

Special Character Handling: Depending on the nature of the text data, special characters or symbols may need to be addressed. For instance, email addresses, URLs, or non-standard characters may require specific handling.

Preprocessing is essential because unprocessed text data is often noisy and unstructured. It may contain irregularities, inconsistencies, or extraneous information that can negatively impact analytical tasks. By applying preprocessing techniques, we ensure that the text data is clean, consistent, and well-structured, laying a solid foundation for further analysis.

For instance, in a review analysis context, preprocessing helps remove irrelevant information, such as punctuation or common stop words, while preserving essential terms. This not only improves the efficiency of the TF-IDF process but also ensures that the derived TF-IDF scores accurately reflect the significance of each term within the document and across the corpus.

2.1.2.2. Term Frequency (TF)

The TF component calculates the frequency of each term within the document. In simpler terms, it determines how often a term appears. For example, if the term "excellent" appears three times in a review, it will have a higher TF weight assigned to it.

2.1.2.3. Inverse Document Frequency (IDF)

The IDF component assesses the uniqueness of each term across the entire corpus. It identifies terms that are relatively rare and specific to particular documents. For instance, a specialized term like "gourmet" may have a high IDF score if it appears in only a limited set of reviews.

2.1.2.4. TF-IDF Score Calculation

The TF-IDF score for each term is computed by multiplying its TF and IDF scores. This score reflects both the local importance of the term within the document and its global significance across the corpus.

The output of the tokenization process with TF-IDF is a structured representation of the original text data. In this representation, each term is associated with its TF-IDF score. This

structured format is instrumental in enabling computational analysis, such as text classification, sentiment analysis, and clustering. It quantifies the importance of words within each document, facilitating deeper insights into textual data.

To illustrate the concept, let's consider an example review about a restaurant: "The food at this restaurant is excellent, and the service is top-notch."

After tokenization with TF-IDF, the representation might look something like this:

- "food" (TF-IDF Score: 0.5)
- "restaurant" (TF-IDF Score: 0.2)
- "excellent" (TF-IDF Score: 1.0)
- "service" (TF-IDF Score: 0.7)
- "top-notch" (TF-IDF Score: 1.2)

In this example, terms like "excellent" and "top-notch" receive high TF-IDF scores, indicating their significance in describing the positive sentiment of the review. Conversely, more common words like "food" and "restaurant" receive lower scores, as they are not as distinctive and appear more frequently across the corpus.

In conclusion, the second step of our research process is pivotal in the transformation of unstructured textual data into a structured and analyzable format. Through the utilization of TF-IDF, we assign weighted scores to individual terms, which quantify their importance within documents and across a corpus. This structured representation empowers us to gain profound insights into textual data, particularly in the context of reviews and text classification. It lays a robust foundation for subsequent analytical processes, contributing significantly to our overarching goal of enhancing interpretability and transparency in AI-driven decision-making.

2.2.3. Get the probabilities of getting positive and negative classification

In the intricate landscape of text analysis, the third step of our research methodology emerges as a pivotal stage in our mission to achieve enhanced interpretability and transparency

within the realm of text classification. This crucial phase involves the calculation of probabilities, specifically aimed at determining the sentiment of textual data – whether it be an original review or one of the meticulously generated counterfactuals.

Our methodology for probability calculation is designed to provide a consistent and well-defined approach, ensuring that both the original review and the counterfactuals are subjected to the same analytical process. To achieve this, we rely on a methodology that leverages insights from the train data vector set, allowing us to assess the sentiment of textual data effectively.

The first step involves representing both the original review and the counterfactual instances as vectors, denoted as X . This standardized representation facilitates a uniform and comparative analysis of the text data. The crux of the methodology lies in the systematic calculation of distances between vector X and the vectors contained within the train data set. This distance calculation serves as a fundamental measure of similarity or dissimilarity between the text data instances.

Following the computation of distances, the system proceeds to select the nearest neighbors from the train data set. The parameter ' k ' determines the number of neighbors to consider in the analysis. For instance, if ' k ' is set to 100, the system meticulously selects the 100 vectors from the train data set with the shortest distances to vector X . These 100 vectors collectively form the nearest neighbors to our text data instance.

Among the carefully selected 100 nearest neighbors, the system places significant emphasis on the labels associated with these neighbors. These labels effectively indicate whether the neighbors predominantly belong to the positive or negative class.

The probabilities pertaining to the sentiment of the text data instance (whether it leans toward positivity or negativity) are meticulously calculated based on the labels of the selected neighbors. If a substantial majority of the selected neighbors bear negative labels, the system makes a prediction that the text is likely to be classified as negative. Conversely, if the majority of the neighbors are positively labeled, the prediction swings towards positivity.

To provide a tangible example, suppose that among the selected 100 neighbors, a significant 70% are labeled as negative. In this scenario, the probability of the text data instance

being classified as negative is calculated as $70/100$, which simplifies to $7/10$. Simultaneously, the probability of the text being classified as positive would be calculated as $30/100$, or $3/10$.

The output of this meticulously crafted step is a robust final prediction concerning the sentiment of the text data instance. This prediction is not presented in isolation; rather, it is accompanied by the calculated probabilities of the text's likelihood of being classified as positive or negative. These probabilities hold immense value, as they not only contribute to the confidence level of the sentiment prediction but also significantly enhance the interpretability of the model's decision-making process.

In conclusion, the third step in our research methodology is pivotal in unearthing the probabilities that underpin text classification, whether it pertains to an original review or one of the meticulously generated counterfactuals. Through a systematic analysis that consistently applies the method to both data categories, we arrive at sentiment probabilities that not only bolster the transparency but also enrich the interpretability of our AI-driven decision-making process. These probabilities stand as a testament to our commitment to achieving meaningful insights in the complex realm of text analysis, fostering trust and responsible AI utilization.

2.2.4. [Get Neighbour statistics](#)

In the intricate web of text analysis, the fourth step of our research methodology emerges as a pivotal stage, where we venture deeper into the realm of neighbor statistics. This phase is instrumental in our mission to provide comprehensive insights into text data, whether it takes the form of the original review or the meticulously generated counterfactuals. In this step, we embark on a journey to calculate density, a key metric that serves as a window into the distribution of features and their profound influence on the model's predictions.

Density, in the context of our research, serves as a metric of paramount importance. It encapsulates the concentration and distribution of features that underpin a particular output or prediction. By quantifying the density of these features, we gain a nuanced understanding of how they impact the model's decision-making process.

Traditionally, the formula for density involves the ratio of mass (M) to volume (V), denoted as $d = M/V$. However, our approach adapts this concept to align specifically with the features that contribute to the model's prediction.

To embark on the journey of density calculation, we begin by considering the mass (M). This mass represents the number of features that wield influence over the model's prediction. Crucially, the calculation of M is tailored to the nature of the prediction being made. For instance, if a counterfactual is classified as positive, we diligently focus on the features that have contributed to this optimistic prediction. Conversely, we discreetly disregard the features associated with negativity. This selective consideration results in the quantification of the features that are pivotal to positivity, and we denote this quantification as M.

Volume (V), in the context of our research, is not a spatial dimension but rather an abstract representation of feature distribution. To calculate V, the system engages in a meticulous process. It involves computing the average distance (d) for each point (or feature) that has played a role in shaping the prediction. The formula that guides this calculation is $d = (q/d) = q / \sum (d/q)$, where q signifies the total number of features affected by the positivity prediction, and $\sum (d/q)$ denotes the sum of distances associated with these features.

A deeper exploration of the formula further refines it to $d = q^2 / \sum d$, where q^2 captures the squared number of features contributing to positivity, and $\sum d$ represents the sum of distances traversed by these features. This refined formulation encapsulates the concentration and distribution of features that shape the positivity prediction.

With the groundwork laid for density calculation, we proceed to apply this process to every instance within our dataset. This entails identifying the features intrinsically tied to the respective prediction—whether it leans towards positivity or negativity—and evaluating their influence in light of the distances between them.

By methodically performing these density calculations, we gain a quantitative grasp of the concentration of features that underlie the model's predictions. This information is invaluable for enhancing interpretability and transparency, as it brings to the forefront the features that play pivotal roles in shaping these predictions while highlighting those that wield lesser influence.

The density metrics derived for each instance, whether it's the original review or one of the counterfactuals, provide profound insights into the significance of features contributing to the predictions. Elevated density values signify a more concentrated influence of features on the prediction, whereas lower values indicate a more dispersed impact.

Moreover, the comparison of density metrics across multiple instances permits the discernment of patterns and trends in feature distribution. If specific counterfactuals consistently exhibit higher density than others, it signifies that certain feature consistently wield substantial influence in driving those predictions.

The calculation of density metrics in this step significantly enriches the interpretability and transparency of our AI-driven decision-making process. It allows us to pinpoint with precision the features that hold the greatest sway in determining a particular sentiment classification—be it positive or negative. By quantifying the concentration of these features, we empower users and analysts not only to trust the model's predictions but also to grasp the nuanced factors that contribute to these predictions. In doing so, we foster a realm of more informed and responsible AI utilization, particularly in the domain of text analysis and sentiment classification.

In sum, the fourth step in our research methodology is dedicated to the meticulous calculation of density metrics for both the original review and the meticulously generated counterfactuals. This process is designed to unearth the quantitative concentration of features that propel the model's sentiment predictions, offering invaluable insights into the influence of specific features on the decision-making process. By tailoring the concept of density to our unique context, we elevate the interpretability and transparency of AI-driven text classification, nurturing trust and comprehension among users and analysts alike.

2.2.5. Define the best Counterfactual

In the final step of our research methodology, known as " Define the Best Counterfactual," we embark on a critical journey to select the most appropriate counterfactual from a pool of generated alternatives, along with the original review. This step is pivotal as it involves a comprehensive assessment of the densities associated with each counterfactual, followed by a comparative analysis against the density of the original review. The primary objective is to identify the counterfactual that closely aligns with the density of the original review, making it the optimal

choice. In this detailed process, we navigate the complexities of density comparison, acknowledge its significance in decision-making, and emphasize its vital role in ensuring the transparency and interpretability of AI-driven text analysis.

At the heart of selecting the best counterfactual lies the concept of density. Density is a quantitative metric that encapsulates how similar or aligned a given text is to another in terms of sentiment. In our context, it serves as a pivotal element for decision-making.

The process of determining the best counterfactual is methodical and comprises several key techniques begins by gathering and considering all the counterfactuals meticulously generated throughout our research. Each counterfactual represents a potential variation of the original review, offering distinct perspectives on how the text can be altered while retaining the same sentiment prediction. Next, The crux of this step revolves around the calculation of densities. We determine the density value for each of the generated counterfactuals, as well as for the original review. These density values provide us with a quantifiable understanding of the concentration of influential features within each text variant.

The crux of our technique lies in pinpointing the counterfactual whose density most closely aligns with that of the original review. This alignment is achieved by computing the nearest densities among all the available options. In essence, we assess the proximity of each counterfactual's density to that of the original review.

The counterfactual with the nearest density to that of the original review emerges as the best counterfactual. It is the choice that most faithfully mirrors the sentiment distribution of the original review while introducing specific modifications.

Understanding the importance of choosing the best counterfactual is important. The original review often allows for multiple interpretations and alterations. Different phrasings, word choices, or sentence structures can lead to varied sentiments. Selecting the best counterfactual ensures that the modified text remains meaningful and adheres to the intended sentiment.

Opting for a counterfactual that closely aligns with the density of the original review enhances the interpretability of AI-driven decisions. This choice provides a clear understanding of how minor textual adjustments can impact sentiment predictions, thus promoting transparency in the decision-making process.

Beyond theoretical research, these techniques hold practical relevance. In real-world scenarios like content moderation, maintaining the desired tone while addressing specific issues is crucial. The best counterfactual can serve as a model for generating revised text that upholds the intended sentiment.

The output of this final step comprises a set of density values, each associated with a counterfactual and the original review. Additionally, it includes the density of the best counterfactual, chosen for its close alignment with the density of the original review. These density values provide a comprehensive snapshot of how various text modifications influence sentiment distribution.

In summary, Step 05 of our research methodology revolves around defining the best counterfactual—a pivotal decision achieved through a meticulous comparison of densities. By considering all generated counterfactuals, calculating their densities, and selecting the counterfactual with the nearest density to that of the original review, we ensure transparency in AI text analysis. This process empowers users and analysts to comprehend and trust the underlying decision-making mechanisms. The output of this step, with its density values, sheds light on the multifaceted nature of sentiment within text data, underscoring the significance of selecting the best counterfactual in AI-driven decision-making scenarios.

If we go through an example that encompasses all five steps of our research methodology for enhancing the interpretability of a K-Nearest Neighbors (KNN) model in text classification, particularly in the domain of online reviews.

Imagine we have an online review: "The food at this restaurant is excellent, and the service is top-notch." We tokenize this review and identify Part-of-Speech (POS) tags, focusing on adjectives. Counterfactuals are generated by flipping adjectives, transforming "excellent" into "poor" and "top-notch" into "mediocre." Randomization introduces variety among counterfactuals, creating alternative versions of the review.

In the second step, both the original review and generated counterfactuals are tokenized using TF-IDF. The word "excellent" receives a high TF-IDF score due to its importance in the review. Next, the system will calculate probabilities for each text instance (original review and counterfactuals) by comparing them to the labeled data. If the majority of nearest neighbors for a

counterfactual have positive labels, it's likely to be classified as positive. After those densities are calculated for each instance. Features related to positivity have higher density, showcasing their influence on sentiment predictions. Finally, by comparing densities, we select the counterfactual closest in density to the original review. Let's say Counterfactual A has the nearest density.

In this example, Counterfactual A, with slight modifications to the review, is identified as the best counterfactual. It maintains interpretability, demonstrates the impact of word choice, and aligns with the original review's sentiment, thus enhancing transparency in AI-driven text analysis.

2.3. Testing and Implementation

2.3.1. Implementation

The implementation of the proposed counterfactual explanation for the KNN 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.


```

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 1: Label Encoding using LUTLabelEncoder

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.

```

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: 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.

```
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 3: WordNet POS Tagging

Counterfactual Explanation for K-Nearest Neighbour

Before the development, required libraries were imported into the environment as shown in figure 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 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 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 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 6.

```
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 6: Text Vectorization

2.3.2. Testing

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

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: Test Case 01

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 a bore movie for the once who not 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 3: Test Case 02

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 K Nearest Neighbour classifier. Then had to compare proposed novel solutions with the existing methods. For that, we selected the novel explanation method of the Random Forest 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 KNN explanation and the performance of the Random Forest explanation through both positive and negative reviews.

3.1.1. Results evaluation of the KNN 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 KNN 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 feature densities between each contradictory prompt with the original prompt. 40.504, 46.403 are the feature densities for the above-mentioned contradictory prompts respectively. Finally, the algorithm returns the 2nd contradictory prompt which is the closest contradictory prompt to the feature density of the original input review (48.419) as the most accurate contradiction.

Negative review:

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** in the content. It **is** suitable for small children.

After the contradictory prompt generation, the algorithm compares the feature densities between each contradictory prompt with the original prompt. 35.406, 35.424 are the feature densities for the above-mentioned contradictory prompts respectively. Finally, the algorithm returns the 2nd contradictory prompt which is the closest contradictory prompt to the feature density of the original input review (38.518) as the most accurate contradiction.

3.1.2. 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]

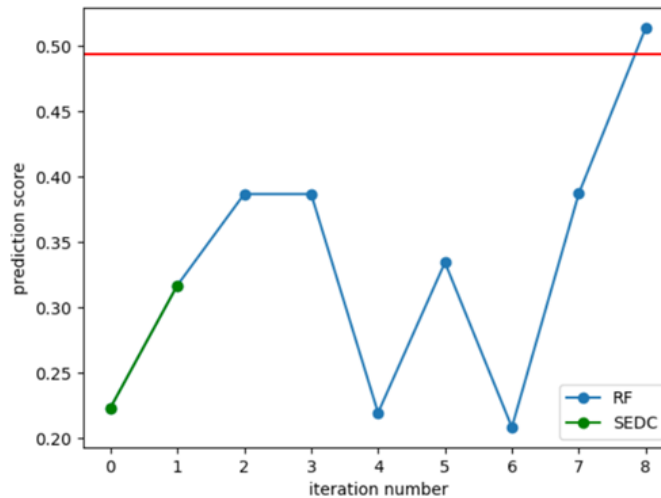


Figure 7: Class change for negative review 1 RF explanation Vs SEDC

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.

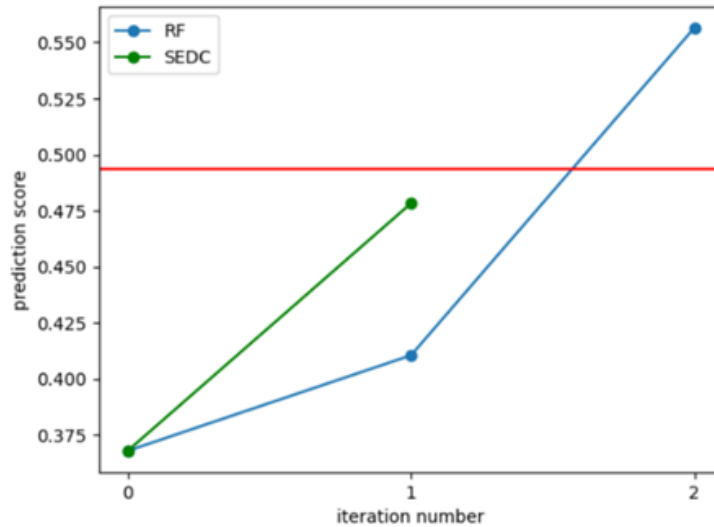


Figure 8: Class change for negative review 2 RF explanation Vs SEDC

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.

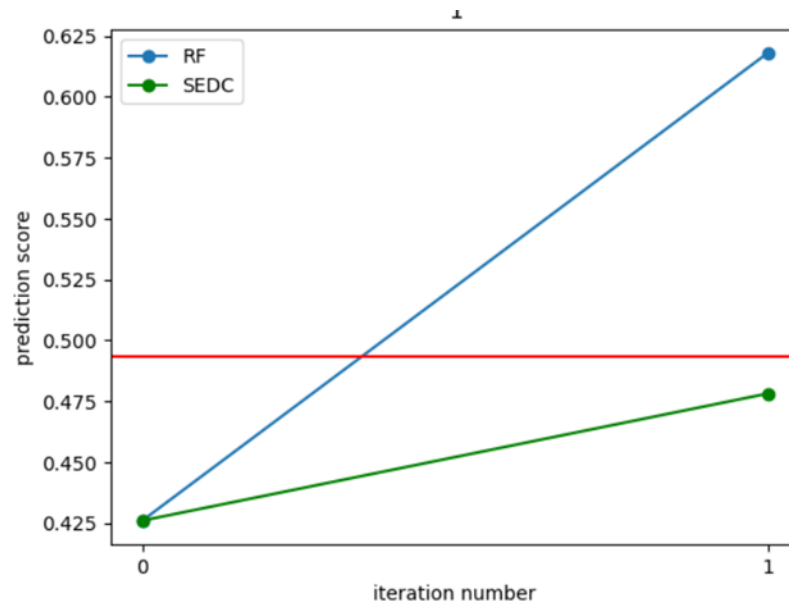


Figure 9: Class change of negative review 3 RF explanation Vs SEDC

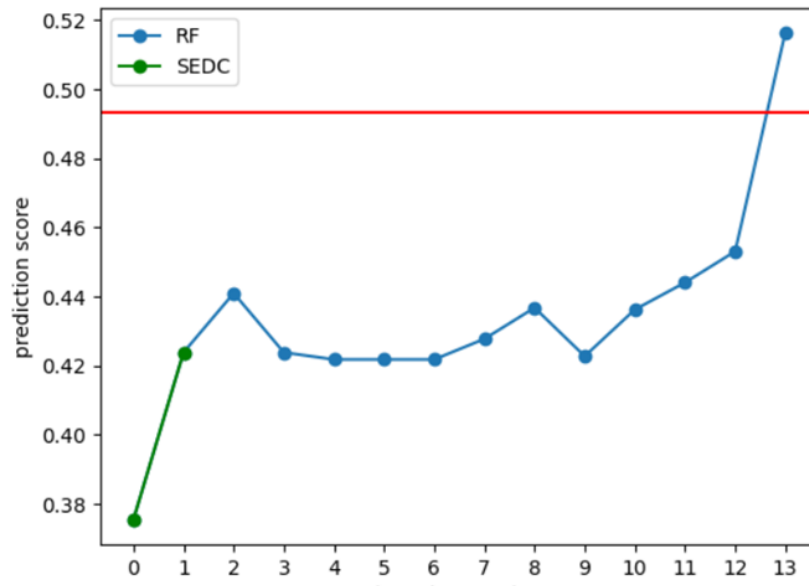


Figure 10: Class change of negative review 4 RF explanation Vs SEDC

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.

Positive Review 1:

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 počasni krug.* Did I mention it is one of the funniest movies I've ever seen?

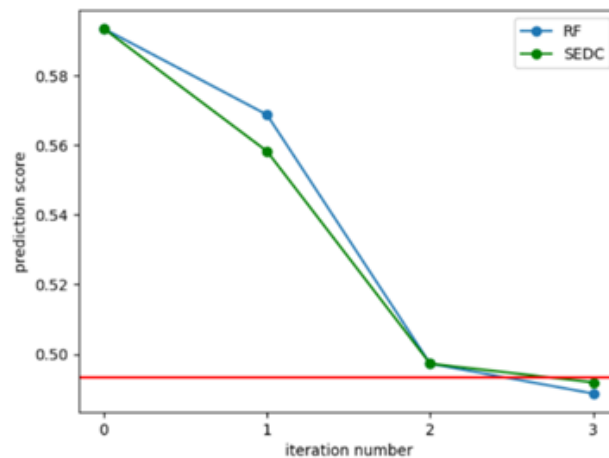


Figure 11: Class change of positive review 1 RF explanation Vs SEDC

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.

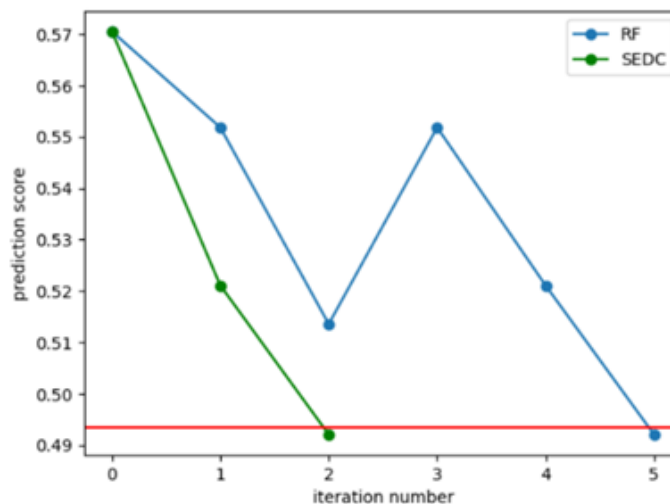


Figure 12: Class change of positive review 2 RF explanation Vs SEDC

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 À 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 À 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 **Maître 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.

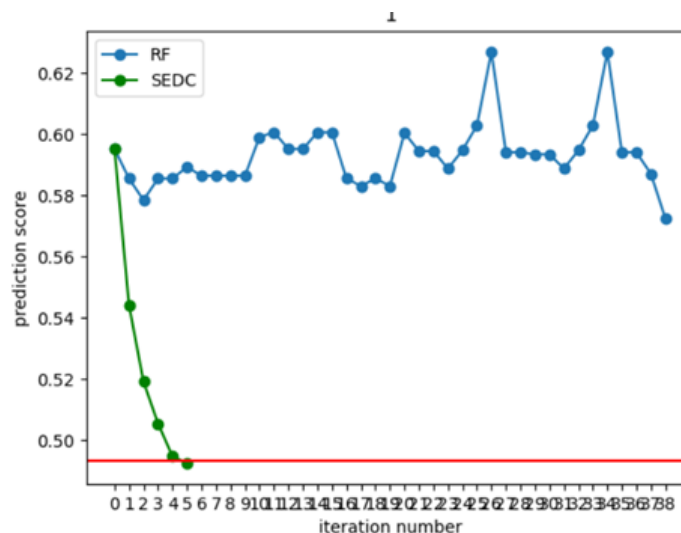


Figure 13: Class change for positive review 3 RF explanation Vs SEDC

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)	<ul style="list-style-type: none"> • Prepare the data set and implement Support Vector Machine classifier. • Generate a novel counterfactual rule-based mechanism using text classification domain. <ul style="list-style-type: none"> ➤ 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)	<ul style="list-style-type: none"> • Prepare the data set and implement Logistic Regression classifier. • Develop a novel counterfactual rule-based mechanism using text classification domain. <ul style="list-style-type: none"> ➤ Vectorize all the text using TFIDF vectorizer ➤ Antonym selection and iteration replacement and removal

		<ul style="list-style-type: none"> ➤ 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)	<ul style="list-style-type: none"> • Prepare the data set and implement Random Forest model. • Generate a novel counterfactual rule-based mechanism using text classification domain. <ul style="list-style-type: none"> ➤ 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

<p>Lakshani N.V.M (IT20013950)</p>	<p>Provide a novel counterfactual explanation for K nearest neighbour model related to the text classification domain. (Feature density comparison based counterfactual explanation)</p>	<ul style="list-style-type: none"> • Prepare the data set and implement K-nearest Neighbour classifier. • Generate a novel counterfactual rule-based mechanism using text classification domain. <ul style="list-style-type: none"> ➤ 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.
--	--	--

5. CONCLUSION

In the dynamic and ever-evolving landscape of artificial intelligence, the quest for Explainable Artificial Intelligence (XAI) is of paramount significance. In this context, our comprehensive research endeavors have been meticulously tailored to enhance the interpretability and transparency of the K-Nearest Neighbors (KNN) model, particularly within the intricate domain of text classification based on reviews. The journey we embarked upon was driven by an unwavering commitment to empower users, analysts, and developers with the tools they need to navigate and trust AI-driven decision-making processes.

Our research's central objective was to create a specialized XAI solution for the KNN model, a model known for its effectiveness yet often criticized for its opacity. Through five key steps, we embarked on this mission, meticulously crafting a methodology that unveils the inner workings of the model, provides insights into its decision-making rationale, and fosters trust in the realm of text classification.

Our research began with the systematic generation of counterfactual instances, a novel approach designed to illustrate how slight modifications to input features would lead to different model outcomes. This approach serves as a bridge between the opaque nature of the KNN model and the need for transparency. By flipping words based on user-selected Part-of-Speech (POS) tags and introducing randomization, we offer valuable insights into how specific word choices influence the model's predictions.

The second step involved transforming unstructured textual data into a structured format through tokenization with TF-IDF. This process assigns weighted scores to individual terms, quantifying their importance within documents and across a corpus. Preprocessing, lowercasing, stop word removal, and punctuation removal were integral to this phase, ensuring that text data was clean and well-structured for computational analysis.

In the third step, we calculated probabilities to determine sentiment, whether positive or negative, for both the original review and counterfactuals. By assessing distances and selecting nearest neighbors from the train data set, we provided users with probabilities that not only

bolstered prediction confidence but also enriched the interpretability of the model's decision-making process.

The fourth step delved into the realm of neighbor statistics, quantifying the density of features influencing predictions. Density, a key metric, revealed the concentration and distribution of these features. By calculating density metrics for both original reviews and counterfactuals, we highlighted the significance of specific features and their influence on predictions.

In the final step, we defined the best counterfactual by meticulously comparing the density of generated alternatives to that of the original review. This critical decision ensured that the chosen counterfactual closely mirrored the sentiment distribution of the original text while introducing specific modifications.

In conclusion, our research into XAI for the KNN model within the context of text classification based on reviews has far-reaching implications. By systematically following these five key steps, we have succeeded in enhancing the interpretability and transparency of the KNN model. Our approach has empowered users, analysts, and developers to gain a deeper understanding of why the model makes specific classifications, whether positive, negative, or neutral.

The significance of our research extends beyond academia; it holds practical relevance in real-world applications. In content moderation, for instance, our methodology can be employed to maintain a desired tone while addressing specific issues. Moreover, our approach stands as a testament to the potential of XAI in fostering responsible AI applications. It paves the way for more effective, accountable, and trustworthy AI-driven decision-making, particularly in the intricate domain of text analysis and sentiment classification.

Our commitment to transparency, interpretability, and trust in AI is at the forefront of this research. We have not only unraveled the complexities of the KNN model but also provided a roadmap for future research endeavors in XAI. As AI continues to permeate various aspects of our lives, understanding and trusting these technologies become paramount. With our XAI solution tailored to KNN, we contribute to this understanding, empowering individuals and organizations to harness the power of AI while maintaining control and comprehension.

In the grand tapestry of AI innovation, our research represents a significant thread, one that bridges the gap between the intricacies of machine learning models and the needs of humanity. As we look to the future, we envision a world where AI is not a black box but a transparent and accountable partner, and we are proud to have played a role in shaping that future through our dedication to Explainable Artificial Intelligence.

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-Aug, 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.
- [13] Rajani, N. F., Krause, B., Yin, W., Niu, T., Socher, R., & Xiong, C. (2020). Explaining and improving model behavior with k nearest neighbor representations. *arXiv preprint arXiv:2010.09030*.