Text

Description automatically generated with low confidence

A STUDY ON INTERPRETABILITY OF ARTIFICIAL INTELLIGENCE MODELS USING LOCAL MODEL AGNOSTIC TECHNIQUES

by

ADITYA PANCHWAGH (16084325)

September 2021

MSc Computer Science – Data Science and Analytics

Module: MSc Project (7COM1039-0509-2021)

Supervisor

Prof. Epaminondas Kapetanios

School of Physics, Engineering and Computer Science

Word Count: 11170

Declaration

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Data Science and Analytics at the University of Hertfordshire (UH).

It is my own work except where indicated in the report.

*I did not use human participants in my MSc Project.*

I hereby give permission for the report to be made available on the university website provided the source is acknowledged.

Aditya Panchwagh

Date: 23 September 2022

ACKNOWLEGMENT

I hereby wish to acknowledge and express my sincere gratitude to everyone who helped me during this process, and because of which I could do a successful completion of the dissertation.

I am very lucky and appreciative of my professor Prof. Epaminondas Kapetanios support, assistance, ongoing advice, and encouragement throughout my dissertation as I had the fantastic opportunity and chance to study under his supervision.

Throughout my master's degree program at the University of Hertfordshire, I received vital support, advice, suggestions, and encouragement from my faculty and staff, without whose help I could not have progressed to this point and finished my dissertation study and hence, I would like to thank all of them. Their extensive knowledge of the data science and analytics field, as well as related coding and small group activities during the course, has given me the possibility to expand my knowledge and really assisted me in moving forward.

Aditya Panchwagh

Date: 23 September 2022

ABSTRACT

While there has been an increase in the usage of deep learning algorithms in many different fields like the medical fields or in business sectors, as the complexity of model increases, the less interpretable it becomes. Therefore, to still use the high-level machine learning models, and still make it interpretable for humans, the methods of Explainable Artificial Intelligence are used. Therefore, in this project I have implemented the local model agnostic techniques namely LIME and SHAP for explaining the traits observed in employee attrition and implemented these techniques in image classification.

After using a different number of machine learning algorithms, the best accuracies were obtained by using the XG Boost algorithm and the Random Forest algorithm, however both these algorithms are high level decision tree algorithms, making them very less interpretable for humans. Therefore, using LIME and SHAP on both machine learning algorithms yielded in the contribution of each attribute to the final decision made, thus making the machine learning model more interpretable and transparent.

For the image classification, CNN model is used in this project, with a total of 2 Max Pooling layers and 4 Conv2D layers. The model was concluded with the Dense layer. Using LIME and SHAP on image classification helps identify the most important pixels in an image, which are taken into consideration by the CNN model, to make the classification.

For the tabular dataset, the results obtained by LIME and SHAP are in terms of probability values. After storing these probability values in two different data frames, for the entire test dataset, which was a total of 617 records, I compared them with each other. The probability values obtained by LIME and SHAP were the same, thus stating that both LIME and SHAP methods gave the same conclusion. Similar results were obtained for the image dataset as well, where the same pixels were highlighted by both LIME as SHAP.

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Declaration | | 2 |
|  | Acknowledgement | | 3 |
|  | The Abstract | | 4 |
|  | List of Graphs | | 8 |
|  | List of Figures and Tables | | 8 |
| I | ***CHAPTER 1: INTRODUCTION*** | | 9-13 |
|  | 1.1 | Introduction Summary | 9 |
|  | 1.2 | Aim | 12 |
|  | 1.3 | Objectives | 12 |
|  | 1.4 | Research Questions and Hypothesis | 13 |
| II | ***CHAPTER 2: LITERATURE REVIEW*** | | 14 – 17 |
|  | 2.1 | Literature review - Introduction | 14 |
|  | 2.2 | Advantages and Dis-Advantages of LIME and SHAP  2.2.1 LIME Advantages  2.2.2 LIME Dis-Advantages  2.2.3 SHAP Advantages  2.2.4 SHAP Dis-Advantages | 15-16 |
|  | 2.3. | Consideration of Legal, Ethical and Social Issues | 17 |
| III | ***CHAPTER 3: RESEARCH METHODOLOGY*** | | 18 - 38 |
|  | 3.1. | Tabular Dataset: - HR Attrition Dataset. | 18 |
|  | 3.1.1. | Description of the Dataset | 18 |
|  | 3.1.2. | Processing of the Data | 19-20 |
|  | 3.1.3. | Analysing the Data  3.1.3.1. Exploratory Data Analysis | 20 - 25 |
|  | 3.1.4. | Model Training and Testing  3.1.4.1. Model Evaluation Techniques  3.1.4.2. Baseline model- Linear Regression  3.1.4.3. K Nearest Neighbor Algorithm  3.1.4.4. Random Forest Algorithm  3.1.4.5. XG Boost Algorithm  3.1.4.6. Final Accuracies | 26-33 |
|  | 3.1.5. | Final Model Selection | 33 |
|  | 3.1.6. | Local Model Agnostic Techniques  3.6.1 LIME  3.6.2. SHAP | 33 – 36 |
|  | 3.2. | Image Dataset Analysis | 36 |
|  | 3.2.1. | Selecting the Dataset | 36 |
|  | 3.2.2. | Processing of the Data and Modelling | 36-38 |
| IV | ***CHAPTER 4: RESULTS AND EVALUATION*** | | 39 - 53 |
|  | 4.1. | Results for IBM HR Attrition Dataset – Structured Data  4.1.1. LIME Results: Example 1: Using the Algorithm - Random Forest  4.1.2. SHAP Results: Example 1: Using the Algorithm - Random Forest  4.1.3. LIME Results: Example 1: Using XG Boost Algorithm  4.1.4. SHAP Results: Example 1: Using XG Boost Algorithm  4.1.5. LIME Results: Example 2: Using the Algorithm - Random Forest  4.1.6. SHAP Results: Example 2: Using the Algorithm - Random Forest  4.1.7. LIME Results: Example 2: Using XG Boost Algorithm  4.1.8. SHAP Results: Example 2: Using XG Boost Algorithm | 39-47 |
|  | 4.2. | Hypothesis and its Evaluation | 48-49 |
|  | 4.3. | Results for Image Classification  4.3.1. Example 1: Using LIME  4.3.2. Example 1: Using SHAP  4.3.3. Example 2: Using LIME  4.3.4. Example 2: Using SHAP | 50-53 |
| V | ***CHAPTER 5: CONCLUSION*** | | 54 |
| VI | ***CHAPTER 6: REFERENCES*** | | 55 – 56 |
| VII | ***APPENDIX***   1. Source Code 2. Project Plan 3. Supervisory Meetings | | 57 – 74 |

List of Graphs:

*Graph 1:*  Sampling Results - before and after using SMOTE *Source:* Self-made Source Code Output

*Graph 2.* Correlation Matrix Source*:* self-made programming code

*Graph 3.* Density Plot of Age with Attrition Rate *Source:* self-made programming code

*Graph 4.* Attrition vs Gender *Source:* self-made programming cod

*Graph 5.* Attrition vs Overtime *Source:* self-made programming code

*Graph 6.* Attrition vs Job satisfaction *Source:* self-made programming code

*Graph 7.* Attrition vs Job level *Source:* self-made programming code

*Graph 8.* Attrition vs Stock Option Level *Source:* self-made programming code

*Graph 9.* Attrition vs Years at company *Source:* self-made programming code

*Graph 10.* Attrition vs Job Involvement *Source:* self-made programming code

List of Figures and Tables:

Figure 1 Explainable AI Architecture Source: Self- drawn .

Figure 2. Explainable AI Types *Source:* Self- drawn

Figure 3 Flowchart of Methodology *Source:* Self- drawn

Figure 4: CNN Model Summary *Source:* source-code output

Figure 5: LIME result ex 1 output *Source:* source-code output

Figure 6: SHaP result ex 1 output *Source:* source-code output

Figure 7: LIME result ex 1 using XG Boost algorithm *Source:* source-code output

Figure 7: SHAP result ex 1 using XG Boost algorithm *Source:* source-code output

Figure 8: SHAP result ex 2 using Random Forest algorithm *Source:* source-code output

Figure 9: Probability Percentage using LIME *Source:* source-code output

Figure 10: Probability Percentage using SHAP *Source:* source-code output

Figure 11: Using LIME for Image Classification *Source:* source-code output

Figure 12: Using SHAP for Image Classification *Source:* source-code output

Table 4.1. Accuracies of Machine Learning Models *Source:* self-made programming code

# CHAPTER 1: INTRODUCTION

1.1. INTRODUCTION SUMMARY:

It has been noticed that there is a significant increase and usage of the Artificial Intelligence models since the past few years. The more complex model we use, for example: Deep Learning models, the more efficient they are in making predictions since, they outperform the traditional models in every aspect. However, there is a restriction in using such type of models. The mode we make a model complex, the less interpretable the model is to humans. For example, there are thousands of parameters in a Deep Learning model and the model finally returns just a decision without any explanation of the result.

This is the biggest drawback of machine learning models, understanding how a model arrived at a particular prediction. For this task, the concept of **Explainable Artificial Intelligence (XAI)** was introduced. (Xu, F. *et al*., 2019)

Any machine learning model follows certain steps for prediction:

* First, we take into consideration the dataset for training.
* Secondly, we apply the pre-processing steps for data analysis.
* Next, we make a machine learning model.
* Finally, the model predicts the outcome.

The process of Explainable AI starts after this final model prediction which is very helpful for the user to see and understand how the decision was made by the model.

The following is the process for Explainable Artificial Intelligence (XAI):

***Figure 1:* Explainable AI Architecture *Source:* Self-made diagram**

With the use of XAI, the users can know, the reason behind a particular decision made by the model. The machine learning models are basically used for solving real-live problems. However, even if the model comes up with a solution, the question arises is that how much the users can trust the model’s decision? Therefore, to make the model more interpretable Explainable AI techniques are used.

Making the model more interpretable has its own perks, like:

* **Trust**: If an explanation is given for every decision made by the model, it becomes easy for an individual to trust the results given by the model.
* **Fairness**: Since the interpretations of the model are available, it helps ensuring that the model is completely unbiased, and all the results obtained are not based on any kinds of discriminations or biases.
* **Reliability**: This is the most important parameter when it comes to using a machine learning model’s outcome. If the reasons behind the model’s outcome are explained, then it becomes easy to judge the outcomes predicted by the model.

**Therefore, the Explainable AI techniques can be divided as:**

Global

Local

Model Agnostic

Model Specific

Post-hoc Methods

Intrinsic Methods

**Figure 2. Explainable AI Types *Source:* Self-made diagram**

This project focuses on using the Model-Agnostic Local methods to make the model more transparent for the users and understand the individual predictions made by the model.

* **Depending on Model: -**
* **Model Specific: -**
  + As the name suggests, these methods are specifically restricted or limited to a distinct model.
  + All the intrinsic methods are model specific by nature.
* **Model Agnostic: -**
  + These methods can be used and applied to several models.
  + They come under the post-hoc techniques.
  + The use of these techniques is that we can use and compare many different models on the same evaluations matrices rather than creating different evaluations for each model.
* **Varying as per the Individual Case or for the Complete Model: -**
  + **Local Methods: -**
    - The explanation of a particular case or instance occurring in the data frame is given by local model.
    - Some methods of Local techniques are: -
      1. **LIME (Local Interpretable Model-agnostic Explanations):**
         * LIME techniques are used to get an interpretation on specific instances for the black box models.
         * LIME shows the probability of why the result was given by the model by giving the contribution of each column into the prediction result.
         * Another usefulness of LIME is that anyone, irrespective of his knowledge in the of Artificial Intelligence pr Machine learning is able to understand the reasons behind the model’s outcomes.
      2. **SHAP (SHapley Additive exPlanations):**
         * SHAP is like LIME, which is also used for providing interpretations for single instances as well as for the overall model.
         * The goal of SHAP is to provide a user-friendly interpretation of the model’s results and make the black box model more transparent.
  + **Global methods: -**
    - As opposed to the local methods, the global techniques are used to give the overall representation of a machine learning model.
    - The global techniques showcase the contribution of all the attributes towards the behaviour of the model, i.e., finding if there is a positive or negative correlation of the attributes with the resultant attribute and many more.

Therefore, this project focuses on the implementation of these above-mentioned Local Model Agnostic Techniques on the use-cases of Explainable AI, to make the black box models more transparent and interpretable.

**For this project, I have considered two use-cases of Explainable AI which are: -**

* **XAI on IBM HR Attrition Dataset - Business Perspective (Part 1):**
  + The most important issue that a company faces is to tackle the problem of employee attrition.
  + Since employees are the most important assets of a company, using the Local Model agnostic techniques on the above-mentioned dataset would help in highlighting the important traits leading to employee attrition.
  + Therefore, companies can try to mitigate this issue by targeting such employees.
* **XAI on Image Classification (Part 2):** 
  + Often when the model classifies an image, the user has no knowledge of how the model was able to classify the image or using which attributes did the model reach at that decision.
  + Therefore, the Local Model Agnostic techniques like LIME or SHAP are used in such cases, to acquire a better insight of how the model was able to arrive at a decision.

**1.2. AIMS:**

* + The implementation of different algorithms of machine leaning and find the most optimal algorithm among them.
  + To understand different Local model agnostic techniques(LIME and SHAP) used for explaining the outcome given the machine learning models.
  + To interpret the important attributes used by any machine learning model to make a classification, using the Local Model Agnostic Explainable AI techniques.

**1.3 OBJECTIVES: -**

* To successfully execute the methods of Explainable AI.
* To implement different model-agnostic techniques(Local techniques)
* To compare different types of supervised learning algorithms on the given dataset and the results produced by the Local Model Agnostic techniques on them.

**1.4 RESEARCH QUESTION AND HYPOTHESIS:**

This project takes into consideration the following two Research Questions:

1. **Do the Local Model Agnostic techniques(LIME and SHAP) give detailed justification on the attributes used by the machine learning models for the individual predictions upon the produced outcome?**
   1. Let the hypothesis be formed for the above research question, therefore, let,

**1. H0 (Null Hypothesis): -**

The Local Model agnostics techniques (LIME and SHAP) do not provide any explanation on the attributes used by the machine learning models to arrive at the final classification.

**2. H1 (Alternate Hypothesis): -**

The Local Model agnostics techniques (LIME and SHAP) do provide the explanation on the attributes used by the machine learning models to arrive at the final classification.

1. **Do both the local model agnostic methods, LIME and SHAP produce the same outcome for the classification results given by the machine learning models, or the results obtained are different?**
   1. Let the hypothesis be formed for the above research question, therefore, let,

**1. H0 (Null Hypothesis): -**

Both the Local Model agnostics techniques (LIME and SHAP) provide different results for the same machine learning algorithms to arrive at the final classification result.

**2. H1 (Alternate Hypothesis): -**

Both the Local Model agnostics techniques (LIME and SHAP) provide the same results for the same machine learning algorithms to arrive at the final classification result.

# CHAPTER 2: BACKGROUND RESEARCH LITERATURE REVIEW

# 2.1 Literature Review Outline: -

Qutub*, A*. et al.in hispaper focuses on the analysis of data for IBM-HR attrition dataset. They have performed various machine learning algorithms; however, they have used imbalanced dataset for their classification. Also, no Explainable AI techniques were used to explain the outcome produced by the model. (Qutub*, A*. *et al*., 2021)

The authors, Venkatadri, T. and Kakollu*,*V.intheir paper focused on training a random forest classification model for prediction of the reasons behind the employee Attrition. They also used under-sampling techniques to deal with the problem of data unbalancing. However, in this paper, they have not used any Explainable AI methods to showcase which factors contribute mostly towards the attrition of employees. (Venkatadri, T. and Kakollu*,*V. , 2022)

The authors Gunning, D. et al. in their research paper focuses on the reasons behind the introduction of the Explainable AI field of study. Because of the increase on the use cases of black box models, this paper explains the importance and uses of using the Explainable AI technology while also showcasing some drawbacks of this technology. However, this paper does not consist of any methods of Explainable AI, it focuses only on the introduction to Explainable AI and the potential challenges in it. (Gunning, D. et al., 2019)

This research Paper has briefly explained about the Explainable Artificial Intelligence Using Convolutional Neural Network (CNN) algorithm helping the Humans to Support their decision systems in Medical Field. The aim of this project is to study on Explainable Artificial Intelligence Methods Mainly on both SHAP and LIME models. (Knapiˇc, S. et al., 2021)

This research paper tells us about the study of the LIME model. The main aim of the research paper is studying the detailed factor related to LIME and how it can be helpful for Black-Box Models to create more transparency for the output received. Using different Algorithms and techniques this paper has clearly explained that based on the sampling process’s randomness, model credibility explanations and changes of the sampling proximity, the Deep learning Accuracy can be easily explained (Zhang, Y. et al., 2019)

The primary purpose of the paper is headed towards studying about the framework in the field of Explainable Artificial Intelligence based on SHAP technique. This paper has explained about how Machine learning Models can be explained in more detailed way using SHAP XAI technique. Based on Framework of SHAP, the Machine Learning models can be explained in more detailed and in Accurate way.(Chromik, M. 2020)

This research paper explains about the Explainable artificial intelligence compiled on the dataset of Employee attrition. An important objective of this research paper is to analyse a factor that determine the employee attrition rate and the factors that influence the LIME and SHAP model. After the analysis of both the models on Employee attrition dataset, it is observed that both the model deeply studies the factors, and both the model has mainly focused on the same target labels. However, class Imbalance is the limitation observed in this research. (Knapiˇc, S. et al., 2021)

The authorsSamek, W. and Müller, M.inthis research paper started with the explanation of the requirements of Explainable AI while using the deep machine learning algorithms. They also explained the legal and ethical sides of explainable AI. Furthermore, they displayed the methods of Explainable AI, explaining LIME for local instances, and explaining propagation-based approaches. Finally, they concluded with the challenges faced while working in this field. (Samek, W. and Müller, M. , 2019)

The authorsDas, A. and Rad, P in this paper focused on the use of Explainable AI techniques on certain deep learning algorithms and black box models. In this, they have used LIME method on Image Classification, where in they have explained in a step-by-step fashion, why their model is labelling a particular image. Along with local interpretations, they have also provided global models providing the limitations for Explainable AI visualizations as well.

(Das, A. and Rad, P. , 2020)

The authors Sekaran, K. and Shanmugam, S. in their paper had discussed different deep learning models approaches for identifying the most important features relating to the attrition of employees. They have used the explainable AI techniques like LIME and SHAP to explain the results given by model. However, they have not balanced the data, which may lead to changes in the features that lead to employee Attrition. (Sekaran, K. and Shanmugam, S., 2022).

**2.2. ADVANTAGES AND DISADVANTAGES OF LIME AND SHAP:**

**2.2.1. LIME Advantages: -**

Using LIME makes any given machine learning model very transparent and intuitive to explain to both machine learning users and non-machine learning users. The explanations which are generated by LIME contain the probability values for final classification decision made. LIME gives the contribution made by each of the attributes for both the classes, therefore giving the information for the most important parameters as well as the least important parameters. LIME being model agnostic, is suitable for a large variety of machine learning algorithms, and LIME also works with structured and un-structured datatypes(images, tabular data).

**2.2.2. LIME Drawbacks: -**

LIME, in a way is very effective to get a closer look to get to know in understanding the working of black box models. However, there is just a drawback for LIME, i.e., the analysis for LIME can only be useful for justifying the Local instances by the model, and not the entire model. Since LIME uses linear models to make classifications, it can be useful when explaining only a local instance or a small region around it, as only that can be linearly classified. However, when this region is expanded, the LIME is not that powerful, to explain the behaviour of the overall model, thus having this as its drawback.

**2.2.3. SHAP Advantages: -**

SHAP uses the game theory approach to give the predict the outcome, thus guaranteeing that the classification made is fairly distributed among all its features. The explanations which are generated by SHAP contain the probability values for final classification decision made. SHAP implementation for tree-based algorithms is very fast compared to the other methods, and it comes as an overall package. This means, SHAP can be used for both local and global instances, therefore SHAP can be used to display the overall trends in a dataset and explain the local instances for the same. Also, it can be used for both structures and un-structures datasets.

**2.2.4.SHAP Dis-Advantages: -**

Using SHAP has its own advantages, however setting that aside, a dis-advantage of SHAP is that when the computational speed is considered with LIME and SHAP, SHAP is at a dis-advantage (except for decision trees algorithms), as it takes more time for SHAP to obtain results, as SHAP’s computation speed is strongly associated to the number of features that are in that dataset. Therefore, in a dataset, higher the number of elements, the more time SHAP will take to provide the final classification results.

**2.3. CONSIDERATION OF ETHICAL, LEGAL AND SOCIAL ISSUE:**

**2.3.1. Ethical Issues: -**

This project has assessed two chosen XAI techniques with regards to logical reasoning for the conducted research. The chosen XAI technique won't just make sense of why the employee is reviewed and why certain actions need to take place like promotion, removal or retainment of the employees and likewise the managers can decide if this forecast is accurate or inaccurate by making a sense of it.

Another ethical aspect is the level to which the managers can rely on the results. The machines have their own limitations and do not understand the behavioural aspect of the employee. Sometimes, necessity of the job, money. household status and position for an employee can be stressful and due to which they need to continue working at the job even though they have a less satisfaction level of that job. These ethical points need to be considered as well by the employers and with the help of XAI techniques, the employer can figure of the employees who are less motivated or highly intellectual but lacking motivation and find out the reasons behind their behaviour and help them to increase their job satisfaction level thereby increasing the motivation level of the organisation.

**2.3.2. Legal Issue: -**

General Data Protection Regulation guidelines mentioned by the EU, on decision making by Artificial Intelligence, has the right to clarification and Transparency as per Articles 13 and 14 state that the process of making decisions should be done in a transparent manner, by providing an explanation, wherever necessary.

**2.3.3. Social Issue: -**

The Results found from the XAI methods should just be used as a recommendation guidance by the companies. The managers of the organisation should keep the final decisions in their hands and use this information as a guiding tool for the same. Hence, the intention of retaining, promoting, or removing an employee should be done by the managers taking into consideration factors outside the survey as well that is social, ethical and behavioural aspects of the employee along with the XAI model results for better results and outcomes.

# CHAPTER 3: METHODOLOGY

3.1. Tabular Dataset: - HR Attrition Dataset.

The following flowchart represents the flow of the analysis performed on the HR Attrition Dataset and the use of local model agnostics Explainable AI techniques on the dataset:

Data Selection

Data Cleaning

Data

Pre-processing

Encoding Dataset

Data

Analysis

Model Training

And Testing

Final

Model Selection

LIME

Local Model

Agnostic Methods

SHAP

***Figure 3:*  Flowchart of Methodology *Source:* Self-made diagram**

**3.1.1. Dataset Description: -**

The dataset selected is IBM HR-Attrition, where the main aim is to check if an employee may leave the company, and because of what factors. Since there are a total of 35 attributes, in this phase, the important attributes are selected for the training of dataset.

The dataset consists of a total of 1471 records. The Attrition Attribute is the dependent variable whereas remaining of the 34 elements or attributes are used as variables which are independent in nature to identify traits that may lead to the attrition of employee.

After importing the dataset, it looks as follows: -

Table

Description automatically generated with low confidence ***Figure 4:*  Dataset Description *Source:* Self-made Source Code Output**

**3.1.2. Data Pre-Processing: -**

It is a process in machine learning, where data is converted from its raw nature to a usable, machine encoded data format. Since the data that we have is in many different data types, for example it is numerical as well as categorical, it first requires to be transformed into a single data format.

Since categorical values cannot be interpreted by the machine, it needs to be encoded into machine interpretable language by using the encoding techniques. In total there are 9 attributes which are categorical, amongst which some attributes such as ‘Over18’ is not as important attribute and contributes very less, which results in its exclusion in the final attribute selection.

**The process of Data Pre-processing is as follows:**

* + - **Data Cleaning: -**

It is very important for any dataset that it does not contain any null values or incorrect format data. Therefore, in data cleaning stage, first the entire dataset is scanned for having any null values or any duplicate records, and then process those records.

* + - **Encoding Dataset: -**

When it comes to the machine learning models, the dataset can only be in the numerical format. However, the IBM-HR Attrition dataset had many categorical features, such as Attrition, Gender, Department and many more. Therefore, before implementing any machine learning algorithm on the dataset, first the Data must be converted from the categorical format to numeric format. There are many ways in which encoding of data can be done, of which the ones used in the project are:

**1. Label Encoding:**

* Label encoding is the process of converting all the categorical values into numeric values.
  + For Example, in our dataset, Gender is a categorical attribute. Therefore, applying categorical encoding on it(cat.codes), Gender(‘M’, ‘F’) is converted into numerical values Gender(0,1).
    - **Balancing Dataset: -**

The attribute of Attrition, which is the dependent variable consist of two categories: ‘Yes’ and ‘No’. Amongst the dataset’s total quantity of records, the category of ‘Yes’ is approximately 200, whereas the category of ‘No’ is approximately 1200. This leads to the imbalance of the dataset, with a greater number of ‘No’ categories than ‘Yes’ and can lead to a biased output.

#### Therefore, to balance the dataset, a data-balancing techniques known as SMOTE is used, which helps in balancing the dataset. SMOTE stands for Synthetic Minority Oversampling Technique, which is used to Oversample the minority class. Therefore, in this case, since the class of ‘Yes’ is less than ‘No’, the SMOTE technique fills in artificial data points which are built on the original points.

#### Using this algorithm helps in finding more cases of the smaller minor class as well, which leads to the unbiased deep learning model. Following are the number of records before using SMOTE and after using SMOTE: -

Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

***Graph 1:*  Sampling Results - before and after using SMOTE *Source:* Self-made Source Code Output**

**3.1.3. Data Analysis: -**

In the process of Machine learning, this is the most important and significant step to be considered. In this process, the entire dataset is visualised using different machine learning libraries like matplotlib or seaborn. In this step, the formation of research questions take place which must be analysed thoroughly through the project.

**3.1.3.1. Exploratory Data Analysis: -**

**I. Visualizations: -**

The Attrition rate of employees is the decrease in the number of employees in a company which can be because of many factors. Therefore, following are the graphs of Attrition with the other attributes:

**Correlation Matrix: -**

The below correlation matrix is between all the attributes with each other. From this, most prominently it can be observed that, with the target attribute i.e., Attrition, the attributes such as Age, Stock\_Option\_Level, JobLevel, JobInvolment are negatively proportional to the target attribute. This means that, for example, if we consider Age as an attribute, then as the Age of employees increases, the attrition rate of employees lowers. Similar is the case for the above-mentioned attributes.

However, some of the other attributes such as DistanceFromHome or Overtime are some of the attributes which are directly proportional to the target attribute. This means that, for example, for the attribute such as DistanceFromHome, as the distance between the home and office starts increasing, the attrition level of the employees also increases.

Chart

Description automatically generated

***Graph 1.* Correlation Matrix Source*:* self-made programming code**

**Attrition with Age: -**

The following figure shows the relationship between Age and Attrition rate of employees. It is clearly visible that the attrition rate of employees increases till the age 30, and after that a constant decrease in the graph can be seen. This implies that, employees mostly in the age group till 30 are most likely to change the company, whereas after age 30, the attrition rate lowers.

Chart, histogram

Description automatically generated

***Graph 2.* Density Plot of Age with Attrition Rate *Source:* self-made programming code**

**Attrition with Gender: -**

In the graph below, when the attrition rate is compared with the gender, it is observed that the male gender has a greater number of attrition rate than the female.

**Chart, bar chart

Description automatically generated**

***Graph 3.* Attrition vs Gender graph *Source:* self-made programming code**

**Attrition with Overtime: -**

Overtime is one of the most important factors which leads to employee attrition. It is observed that the employees who are working overtime are most likely to leave that company than the people who do not overtime.

**Chart, bar chart

Description automatically generated**

***Graph 4.* Attrition vs Overtime graph *Source:* self-made programming code**

**Attrition with Job Satisfaction: -**

In the graph below, the Job Satisfaction ranges from 1 to 4, where 1 is very less job satisfaction and 4 has the highest satisfaction. Therefore, it is observed that the employees having a job satisfaction less than 3 are likely to leave the company, because of the less satisfaction they get from that job, whereas the employees having a job satisfaction of 4(very satisfied) are very less likely to leave the job.

**Chart, histogram

Description automatically generated**

***Graph 4.* Attrition vs Job satisfaction graph *Source:* self-made programming code**

**Attrition with Job Level: -**

The below density plot is between Job Level and Attrition rate. The Job Level ranges from 1 to 5, where 1 being very low level and 5 being the highest level. From the above density plot, it is observed that if the employee is having a low Job level (between 1-3), the attrition rate is highest, whereas when the job level increases, the Attrition rate decreases.

**Chart, histogram

Description automatically generated**

***Graph 5.* Attrition vs Job level graph *Source:* self-made programming code**

**Attrition vs Stock Option Level**: -

The Stock\_Option\_Level was another most important parameter in the analysis of finding attrition rate in employees. The Stock\_Option\_Level included values ranging from 0-4 where 0 meant that the employee had no stock options of the company, whereas 4 meaning having a considerable amount of stock option for that respective company. Therefore, from the analysis, it was observed that the attrition rate in employees is much higher when the Stock\_Option\_Level was less, i.e., less than 2. Whereas the employees having a higher Stock\_Option\_Level has a very less attrition rate.  
 Chart, histogram

Description automatically generated

***Graph 6.* Attrition vs Stock Option Level Graph *Source:* self-made programming code**

**Attrition vs Number of Years in a Company:**

The below graph shows the attrition rate of employees with the number of years they are associated with that company. It is observed that, employees with less duration in a company have the highest attrition rate, i.e., employees working less than 4 years in a company have the highest attrition rate than other employees who stay longer in a company.

Chart, histogram

Description automatically generated

***Graph 7.* Attrition vs Years at company graph *Source:* self-made programming code**

**Attrition vs JobInvolvement: -**

The graph stated below displays the attrition rate against the employee’s Job Involvement in a company. It is observed from the above graph, that the attrition rate increases till the job involvement 3 is observed in the company. This means that the low the Job Involvement, the attrition rate increases.

**Chart, histogram

Description automatically generated** ***Graph 8.* Attrition vs Job Involvement *Source:* self-made programming code**

**3.1.4. Model Training and Testing: -**

After completing the exploratory data analysis of the data, the dataset is in the proper format for implementation of the different machine learning models. Here, in this project I have performed Supervised machine learning classification on the dataset, performing algorithms such as Logistic Regression, Random Forest Classification and some more. For the training of these models, I took into consideration 20 different parameters and then performed classification for the Attrition column.

**3.1.4.1. Model Evaluation Techniques: -**

1. **Accuracy Score: -** It tells how many correct predictions have been made by the model out of 100. For example, for Logistic Regression model, the accuracy score for test data is 75, which implies that, out of 100 predictions, 75 of them were correctly classified.
2. **Precision:** - It is calculated as:
3. **True Positive: -** It is the total records where employees have Attrition as Yes, and our model has also predicted them to be Yes.
4. **True Negative: -** It is the number of cases where the employees have Attrition as No, and our model has also predicted them to be No.
5. **False Positive: -** It is the total records where employees have Attrition as Yes, however, our model has predicted them to be No.
6. **False Negative: -** It is the number of cases where the employees have Attrition as No, however, our model has predicted them to be Yes.

Therefore, Precision contains all the Positives (True Positive and False Positives) in the model i.e., in our case, Precision is the measurement of how many employees that our model identified to have an attrition as ‘Yes’, out of all the total employees having an Attrition ‘Yes’.

1. **Recall: -** It is calculated as:

Therefore, Recall is a measurement of identifying all the correctly identified positives out of all the positives. For instance, in our model, Recall tells us that out of all the employees who have an Attrition ‘Yes’, how many records did our model correctly identify who have an Attrition ‘Yes’.

1. **F1 Score: -**

It is the result obtained by using both the Recall and Precision values. Therefore, a trade of between them must be maintained, where a high Precision and Recall case is very useful.

F1 Score is calculated as: -

1. **Confusion matrix: -**

It consists of all the actual versus the predicted Positive and Negative values used to define how well a classification model classifies the data.

**3.1.4.2**. **Baseline Model – Logistic Regression: -**

The baseline model refers to the basic model which gives the lowest accuracy compared to the other algorithms implemented. Therefore, for my project, Logistic Regression is the baseline model. It is used for classifying instances, for example, in this project is it used to classify the employee Attrition.

The parameters which I have taken into consideration for implementing Logistic regression are: -

1. Solver = LBFGS
2. Random State=42, so that the results obtained can remain constant and not change every time.
3. Max\_iter = 500
4. Verbose=5.

**With the above parameters, the accuracies obtained by Logistic Regression Algorithm are: -**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Algorithm | Test Accuracy | Train Accuracy |
| 1. | Logistic Regression | 72.12% | 74. 47% |

**The classification report and the Confusion matrix is as follows: -**

Chart, treemap chart

Description automatically generated A picture containing text, receipt

Description automatically generated

***Source:* source-code output**

Therefore, from the above evaluation matrix logistic Regression provides a Precision of 74 and a Recall of 69. The correctly made predictions are 214+231 = 445 records out of the 617 records.

These 617 records are a result of the test data, which is obtained because of the train-test-split method, in which the test data is made by 25% of total records. Therefore, the accuracy obtained by the Logistic regression model is decent, however the other algorithms used such as Random Forest or XG Boost produce more stable results.

**3.1.4.3. K-Nearest Neighbors Algorithm: -**

This is a classification algorithm. It uses principle of similarity, i.e., similar things exist close to each other. First, we need to initialize the number of k neighbors, then for every point, the distances between the data points and the cluster centroids are calculated and then classified into a particular neighbor.

The parameters used for KNN algorithm are: -

1. **n\_neighbors = 5**, which means that the knn algorithm considers 5 neighbors at a time for every data point.
2. **P=1,** It implies that, the Manhattan distance is being used to calculate the distances for each point.
3. **Algorithm: Auto,** out of the 4 algorithms, the Auto mode selects the best suited algorithm for the model.

**With the above parameters, the accuracies obtained by K Nearest Neighbor (KNN) Algorithm are: -**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Algorithm | Test Accuracy | Train Accuracy |
|  | KNN Algorithm | 72.12% | 83.39 % |

The classification report and the Confusion matrix is as follows: -‘

Chart, treemap chart

Description automatically generated A picture containing text, receipt

Description automatically generated

***Source:* source-code output**

From the above evaluation matrix KNN algorithm provides a Precision of 79 but a Recall of 61, which is quite poor performance.

The correctly made predictions are 190+255 = 445 records out of the 617 records.As compared to the Logistic Regression algorithm, the precision is very good for the KNN algorithm, however, the recall is very poor, with just 61 correctly identified cases for employee attrition. The overall accuracy obtained by the KNN model is 72%, which is a decent accuracy, however since the recall is very low, the model fails to identify the attrition rate of employees, therefore not making it a suitable algorithm for this classification task.

**3.1.4.4. Random Forest Algorithm: -**

It is an ensemble algorithm in deep learning, used for performing classification tasks. It comprises of various single Decision trees, and from this, best result is selected by means of voting. Also, the random forest classification algorithm provides a list of all the attributes which are most important for the classification result.

Here, the factors used are: -

1. **n\_estimators=300**, It estimates the quantity of decisions trees to be made before final voting.
2. **max\_depth=5,** It specifies the depth of each decision tree.
3. **Entropy= ‘gini’**, The Gini entropy is used for calculating the decision trees.

**With the above parameters, the accuracies obtained by Random Forest Classification Algorithm are: -**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Algorithm | Test Accuracy | Train Accuracy |
|  | Random Forest Classification | 82.82% | 86.20% |

The classification report and the Confusion matrix is as follows: -‘

Chart, treemap chart

Description automatically generated A picture containing text, receipt

Description automatically generated

***Source:* source-code output**

Therefore, from the above evaluation matrix Random Forest Classification provides a Precision of 82 and a Recall of 85. The correctly made predictions are 264+247 = 511 records out of the 617 records.

As compared to the Logistic Regression algorithm, both the precision, recall and the overall accuracy obtained by the Random Forest model is more improved and better, therefore making it a more suitable and reliable deep learning model for classification.

Also, the Random Forest model provides an advantage, by displaying the most important attributes which contributed the most in making the decisions used for classification of Attrition. The following figure shows the attributes which contributed the most: -

Chart, bar chart

Description automatically generated***Source:* source-code output**

Therefore, according to Random Forest Algorithm, the Top-5 attributes which contribute most to the attrition of employees are:

1. StockOptionLevel
2. JobSatisfaction
3. JobLevel
4. JobInvolvement
5. MonthlyIncome

**3.1.4.5. XG Boost Algorithm: -**

It is a high-level decision tree-based algorithm. XG Boost stands for Extreme Gradient Boosting. The model performance and speed of XG Boost makes it one of the best algorithms of machine learning. In the process of boosting, the model is focused on a single weak tree, and then tries to improve it by combining it with other weak models, which in turn generates a strong model.

The parameters used for XG Boost Algorithm are: -

1. **Learning rate=0.09,** this field decides the speed of the model for learning as per the use of different number of decision trees.
2. **N\_estimators:100,** It decides the total number of decision trees used in the XG Boost model.
3. **Random\_state=42,** used to get constant results over the course of running model.
4. **Loss = log\_loss,** the loss used in this case is log\_loss, the same as Logistic regression for classification.
5. **Max\_depth=4**, in a decision tree, it results in the maximum amount of nodes.

**With the above parameters, the accuracies obtained by Random Forest Classification Algorithm are: -**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Algorithm | Test Accuracy | Train Accuracy |
|  | XG Boost Classification | 85.08% | 91.02% |

**The classification report and the Confusion matrix is as follows: -**

Chart, treemap chart

Description automatically generated A picture containing text, receipt

Description automatically generated

***Source:* source-code output**

Therefore, from the above evaluation matrix XG Boost Classification provides a Precision of 86 and a Recall of 84, by far providing the best results of all. The correctly made predictions are 262+263 = 525 records out of the 617 records. As compared to the Logistic Regression algorithm, both the precision, recall and the overall accuracy obtained by the XG Boost model is better in all the aspects, therefore making it a more suitable and reliable deep learning model for classification.

Also, when compared with the Random Forest algorithm, it can be observed that the overall Precision and Recall is also increased, making the model more accurate in identifying the attrition rate of employees. Like the Random Forest model, even the XG Boost model provides an advantage, by displaying the most important attributes which contributed the most in making the decisions used for classification of Attrition. The following figure shows the attributes which contributed the most: -

Chart, bar chart

Description automatically generated

***Source:* source-code output**

Therefore, according to XG Boost Algorithm, the Top-5 attributes which contribute most to the attrition of employees are:

1. JobLevel
2. JobSatisfaction
3. StockOptionLevel
4. BusinessTravel
5. JobInvolvement

**3.1.4.6. Final Accuracies Table: -**

The following table displays the training and testing of various algorithms used: -

|  |  |  |  |
| --- | --- | --- | --- |
| Sr No. | Machine Learning Algorithm | Train Accuracy | Test Accuracy |
| 1. | Logistic Regression | 74. 47% | 72.12% |
| 2. | K Nearest Neighbor Classifier | 83.39 % | 72.12% |
| 3. | Random Forest Classifier | 86.20% | 82.82% |
| 4. | XG Boost Classifier | 91.02% | 85.08% |

**Table 4.1. Accuracies of Machine Learning Models**

**3.1.5. Final Model Selection: -**

Based on the above different accuracies, the best model for classification of attrition of employees is the Random Forest Classification algorithm and the XG Boost. Since this model has the best testing accuracy and lowest RMSE score, the classifications made by these models are better than the other models used.

However, even with a good accuracy, it is difficult to get an idea of how this model is giving better results than the other model, as this random forest model algorithm is a complex model by nature, and when it comes to the XG Boost, there is no way of knowing, how the model arrived at the classification decision.

Thus, to make the model more transparent and get more user-friendly interpretations of the results, in this project, we go ahead a step and use the Local Model Agnostic Techniques under Explainable AI, to explain the outcome given by the models.

**3.1.6. Local Model Agnostic Methods: -**

After finalizing the machine learning algorithm and getting the results for the classification, it is time to test those results. Even though with the Random Forest algorithm, the test accuracy is approximately 90%, still the question arises, **how do general technical/non-technical users trust the model, if it is not known how the model arrived at that outcome?** Therefore, to make our machine learning model more transparent and easier for users to comprehend the thoughts behind the model’s decision making, the Explainable AI part comes into picture.

Using the local model agnostics methods give the explanation for individual results, explaining what attributes contributed to the classification result. Therefore, in this project, I have used two such local model agnostic methods: -

**3.1.6.1. LIME**:

As the name suggests, the Local Interpretable Model-agnostic Explanations method is used for the explanation of the individual instances in a model classification.

**Working of LIME: -**

* **Step 1: -** Install and import Lime library. Since our dataset is tabular, we need to import the LIME Tabular Explainer.
* **Step 2: -** Defining the parameters of LIME Tabular Explainer. The parameters to be passed into the explainer are as follows: -

**a) Data**: In this field, the data should be passed on which the classification can be made.

* + - **Feature Names:** Feature names are an important parameter. In this, all the attributes should be entered, based on which the final classification would be made.

In my dataset, the following attributes are used for Feature Names:

Text, letter

Description automatically generated

**Figure 4.2. Feature Names**

Therefore, based on the above attributes, the final classification will be made.

**b) Mode:** This field contains the type of algorithm to be made, i.e., if we want to perform Regression model, or a Classification model.Thus, in our project, this field contains Classification. The following image explain the parameters given to the interpreter.



**Figure 4.3. LIME Explainer Function**

* **Step 3: -** Explaining a particular Local Instance.

After defining the interpreter, now the final part is to explain local instances. Therefore, following parameters are to be given in the explain\_instance function: -

**a) Data:** In this field, we input a particular local instance for which our LIME model provides a detailed explanation. Therefore, in our function, we select a particular instance from the test dataset, which is the new, unseen data, for which we want to classify if that employee would have an Attrition of 1 (‘Yes’) or 0 (‘No’).

**b) Predict Function:** This field containsthe final machine learning model algorithm function as an input, so that the classifications can be made on the test data and a probability value for the final classification will be provided on the test data instance provided.

So basically, what LIME does is, it fits a simple linear interpretable model, also called as a surrogate model, in the local instance area which is taken into consideration. With this, the black box model becomes more understandable to the user and helps make the model more transparent.

**3.1.6.2. SHAP (SHapley Additive exPlanations):**

Like LIME, the purpose of SHAP is also to clarify and describe the projections given by the black box models, and to make them more transparent and user friendly. SHAP method is basically created on Game theory, where:

a) The machine learning model used for prediction is referred as game.

b) The players refer to the attributes used by the machine learning model for making the predictions.

When considering the case study of our dataset, i.e., HR-Attrition, there are many factors which can affect the attrition of an employee. What SHAP does is, it takes all the features into consideration and then it gives an estimate of how significant each factor is for getting the final classification.

**Working of SHAP:**

* **Step 1: -** Install and import SHAP library and call the initjs() function.
* **Step 2: -** Calculate the SHAP explainer**.** In this, we need to pass the final model selected for making the classifications on the test data record.
* **Step 3: -** Select the appropriate interpreter for SHAP. In our project, the SHAP.TreeExplainer() is used, which is used to approximate the values of SHAP for single tree models or the collections of trees. The following are the parameters passed in the Explainer function ():

**a) Data:** The data that is passed here, is the test data, which is the unseen data. Since SHAP is used for individual predictions, the record for which the explanation is required is also mentioned in the data taken form the test dataset.????

**b) Model\_Output:** Thisattributehas options regarding what output is expected from the model, as in raw output or probability output.

* **Step 4: -** After passing all the above parameters,now the remaining step is to plot the visualizations of SHAP values. In this, SHAP uses function called force\_plot, where in all the features used by SHAP to arrive at the final classification are displayed showing how much each feature contributed towards target attribute.

In this way, SHAP is used for analysing the Employee Attrition and finding which are the top features resulting in the churning of employees.

3.2. Image Dataset Analysis: - Animals-10 Dataset.

**3.2.1. Data Selection: -**

There is a total of 28K images of animals in 10 different classes of this dataset, namely: ‘cat, elephant, spider, butterfly, dog, horse, sheep, squirrel, chicken, cow,’. These are the ‘rgb’ images of size: 224 x 244.

**3.2.2. Data Pre-Processing and Modelling: -**

For the data pre-processing stage, all the given images must be converted into the standard image size, which in this case is 224x224x3. After the images are pre-processed, there is a divided in the dataset. It is separated as Training and Testing with 20% data test size. Therefore, after splitting the dataset, comes the modelling of the data. For this, the process of convolution neural network algorithm is used.

* **Convolution Neural Networks Algorithm (CNN):**

The CNN algorithm is used for image classification and processing images by identifying features of an image. The CNN architecture consists of the following: -

* + **The Convolution Layer: -** The convolution layer is the first and the most important layer of the CNN algorithm. It extracts the characteristics from the image inputted by employing various filters like edge detection of image, blurring the image and many more on the image. The output of this is a feature map which consist of a reduced image size with only the most important features of the image.
  + **Stride: -** A stride decides the movement of pixels of the filter on the image. For example, with a stride=1, the filter is shifted 1 pixel at a time on the image.
  + **Padding:** - Not always does a filter is perfectly applied on an image, especially while detecting the edges of the image. Therefore, to get a more accurate result, padding is done om the image, either by padding the image by zeros, or by dropping the part of image which is not important for feature selection.
  + **Pooling Layer: -** The clearer the image, the more is it bigger in size, and to process such images in large quantity takes a long time. Therefore, what pooling does is, it reduces the number of parameters from the image. It can be achieved in three different ways, which are or average pooling, Sum pooling or max pooling.
  + **Fully Connected Layer: -** This is the final layer in a CNN architecture, where after using all these above layers, we finally convert the feature map into vectors by using the SoftMax activation function to successfully classify these given images.

The structure of the CNN model used for classification of images provided is as follows: - It consist of 2 convolution layers of size 32, with filters of (5x5) and an activation function of Relu. The input shape provided is of 224x224x3.

After that, a layer of size 2x2 using Max Pooling is added. Next, another convolution layer of size 64 is added to the model, however this containing the filter size (3x3), again continued by the 2x2 Max Pooling layer. The activation function used is ‘Relu’.

Finally, a Dense layer containing 1024 units and 10 units concluding with ‘Softmax’ activation Function, is preceded by a Flatten layer. After this, model is trained for a total of 5 epochs, with a batch size of 75 per epoch. Therefore, each iteration is trained over size of 280 parameters, needing a time of approximately 1 hour/epoch.

**The structure of the model is as follows: -**

Table

Description automatically generated

**Figure 4: CNN Model Summary *Source:* source-code output**

# CHAPTER 4: RESULTS AND EVALUATION

# 4.1. Results for IBM HR Attrition Data - Tabular Data: -

# After implementing various deep learning algorithms of like, KNN algorithm, Random Forest, Logistic Regression algorithm and XG Boost, the best results were achieved using the XG Boost algorithm followed by using the Random Forest model for the tabular data. However, these simulations, being black box in nature, it is hard to interpret the model’s conclusions and their ways to reach these conclusions.

Therefore, to understand this process and to get to know the justifications behind the individual model projections, the usage of local model agnostic techniques is done. Using the Local model agnostic techniques like LIME and SHAP on the IBM HR Attrition dataset, helps us understand the various attributes that contribute to the attrition rate of employees. This helps the companies in general to focus on such traits and thus helping them reduce the attrition rate of employees.

**4.1.1. LIME Results for Example 1: -**

The attrition rate of employees is a binary classification as ‘Yes’ denoted by ‘1’

and ‘No’ denoted by ‘0’. Therefore, consider the following local instance selected below: -

**Example 1: Classified Attrition – ‘No’ (denoted as: 0) using Random Forest Model**

The above mentioned are the attributes used for generating the results for LIME model. As per the previously done Exploratory Data Analysis, and when considered the correlation matrix, the following analysis can be observed:

As per LIME analysis, the outcome given by LIME is Attrition: ‘No’ and LIME makes the model transparent by proving the details as to exactly which attributes contributed to the outcome provided by the model. This analysis provided also helps the users to build a trust on the model, as the users get to know the steps taken by the model to reach this final decision.

**Following are the analysis observed by the attributes in LIME: -**

* **Attributes contributing to outcome as Attrition= ‘No’: given a 66% probability**
  + **StockOptionLevel=2**
    - * As observed in the StockOptionLevel graph vs Attrition graph, it was observed that if the StockOptionLevel is greater than 0, then the attrition rate of employees is very less. Since the StockOptionLevel for the given instance is 2, this attribute favours for the employee to stay in the company.
  + **Over Time=0**
    - * Overtime is an attribute where, if the employees are asked not to do overtime, their attrition rate is very less, then the employees who work overtime. In this instance, this employee is not doing any overtime work, therefore, the attribute contributes to not attrition.

Table

Description automatically generated Table

Description automatically generated with medium confidence

**Figure 5: LIME result ex 1 output *Source:* source-code output**

* + - **Job Involvement: 4**
      * The Job Involvement attribute specifies how much an employee is involved in the tasks he has been assigned. Therefore, as per the analysis performed in the EDA, since the job involvement of the employee is very high, this parameter contributes to the attrition rate as ‘No.
    - **JobLevel= 2**
      * The JobLevel=2 implied that the tasks given to the employee in the job are bot low level, which therefore making the employee stay with the company. Therefore, it contributes to attrition rate ‘No’.
    - **Age=42**
      * Since as per the trend observed in the Age vs Attrition graph, the attrition rate increases till the age 30, whereas after the age 30, the attrition rate reduces. Here, since the age is greater than 30 (i.e., 42), this attribute contributes to Attrition= ‘No’.
* **Attributes contributing to the outcome as Attrition= ‘Yes’**
  + - **Job Satisfaction= 1**
      * The next important parameter is the Job Satisfaction. If the employee is dissatisfied by the work his attrition rate increases. As is the current case, where since the JobSatisfaction is 1, it is very low, which therefore contributes to Attrition rate as ‘Yes’.
    - **DistanceFromHome = 13**
      * The DistanceFromHome is a parameter that specifies the job distance from home, and as per the trend observed, if the DistanceFromHome is more, the employees tend to have a high Attrition rate. Since this parameter has a high DistanceFromHome, it contributes towards Attrition= ‘Yes’.

These were some of the most important attributes among the 20 attributes which the LIME considered to make the Random Forest model more transparent to the user to understand the working behind the local outcomes produced by the model.

The above example was using the Random Forest Model, however, the XG Boost model provides a more stable accuracy with less incorrect classifications than the Random Forest model.

**4.1.2. SHAP Results for Example 1 (same instance): -**

**Example 1: Classified Attrition – ‘No’ using Random Forest Model**

****

**Figure 6: SHAP result ex 1 output *Source:* source-code output**

Table

Description automatically generated

Similar to LIME, SHAP is another method used to get local explanations to understand the working of the black box models. Therefore, comparing LIME and SHAP, for the same examples as above, the following output is received: -

The results obtained by SHAP are same as LIME, with the same probabilities assigned to the individual classes of Attrition and the models used.

For example, in this given local instance using the Random Forest model, SHAP assigns 66% probability that the employee would not leave the company, same as LIME, with the similar parameters giving their contributions to each of the probabilities.

The parameters such as StockOptionLevel, JobInvolvement, Age support the class with ‘No’ Attrition, whereas the attributes such as JobSatisfaction, DistanceFromHome support the class with Attrition ‘Yes’.

However, in the end, a greater number of features support the Attrition= ‘No’ class, therefore receiving the final outcome as Attrition No.

***Source:* source-code output**

**4.1.3. LIME Results: Example 1: Using XG Boost Algorithm: -**

As compared to the Random Forest Algorithm, it gives a 66% probability that the employee would not leave the company, explaining based on which parameters the model arrived at that conclusion.

However, when compared with the XG Boost model, it gives a probability that 75% that the

given employee would not leave the company. This implies that the probability values are better assigned to all the attributes using the XG Boost algorithm than the Random Forest Algorithm.

For example, with random Forest algorithm, the most attribute was StockOptionLevel with a probability value of 0.13 and then was JobInvolvement with a value of 0.07. However, here in XG Boost algorithm, the highest probability was given to the attribute JobInvolvement which is 0.17, and then for StockOptionLevel, which is 0.12, thus increasing the overall percentage from 66% to 75%.

Table

Description automatically generated A picture containing graphical user interface

Description automatically generated

**Figure 7: LIME result ex 1 using XG Boost algorithm *Source:* source-code output**

**4.1.4. SHAP Analysis using XG Boost Algorithm: Example 1: -**

The same is observed when using the XG Boost Algorithm, as the accuracy tends to increase than the Random Forest algorithm where the probability increases from 66% to 75%, with more accurate results. The following can be seen in the below image.



**Figure 8: SHAP result ex 1 using XG Boost algorithm *Source:* source-code output**

**4.1.5. Example 2: - Classified Attrition: ‘Yes’ (denoted as: 1) for Random Forest Model**

The following is an instance where the Random Forest algorithm classified the attrition rate to be ‘Yes’ and the LIME is used to identify the attributes contributing to that decision.

Following are the analysis observed by the attributes in LIME: -

* **Attributes contributing to outcome as Attrition= ‘Yes’, given a total of 89%**
  + - **StockOtionLevel= 0**
      * It is the most important parameter, which is observed to contribute greatly to the attrition rate of employees. The less the StockOptionLevel, the more is the attrition rate observed. Since the StockOptionLevel for the given instance is 0, which is the least, this attribute contributed to the Attrition as ‘Yes’.

Table

Description automatically generated with low confidence Graphical user interface

Description automatically generated with low confidence

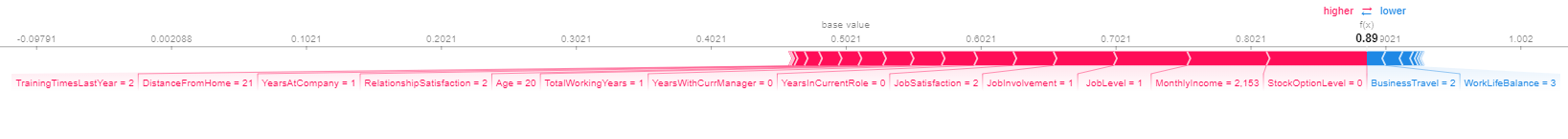
***Source:* source-code output**

* + - **Job Involvement: 1**
      * The Job Involvement attribute specifies how much an employee is involved in the tasks he has been assigned. Therefore, as per the analysis performed in the EDA, since the job involvement of the employee is very low, this parameter contributes to the attrition rate as ‘Yes’.
    - **JobSatisfaction: 2**
      * + The JobSatisfaction parameter specifies if an employee is satisfied while doing his work. As per the exploratory data analysis performed, since the JobSatisfaction is less than 4, which means the employee is not satisfied in his work, this attribute contributes 0.06% of its probability to Attrition ‘Yes’.
    - **PercentSalaryHike: 12**
      * The Percent Salary Hike attribute specifies how much an employee’s salary is hiked. Therefore, as per the analysis performed in the EDA, since the salary hike of the employee is low i.e., 12, this parameter contributes to the attrition rate as ‘Yes’.
* **Attributes contributing to outcome as Attrition= ‘No’**
  + - **WorkLifeBalance=3**
      * Since. as per the trend observed in the WorkLifeBalance vs Attrition graph, the attrition rate decreases as it increases in a company. Here, since the WorkLifeBalance is 3, which is rather greater figure, this attribute contributes to Attrition= ‘No’.
  + **Over Time=0**
    - * Overtime is an attribute where, if the employees are asked not to do overtime, their attrition rate is very less, then the employees who work overtime. In this instance, this employee is not doing any overtime work, therefore, the attribute contributes to not attrition.

However, since among all the total parameters, a greater number of attributes contribute towards the attrition rate as ‘Yes’ than ‘No’, the overall classification for this employee is ‘Yes’. This was the analysis for the Random Forest model.

**4.1.6. SHAP Analysis: - Example 2 Using Random Forest Algorithm**

**Classified Attrition – ‘Yes’ using Random Forest Model**

****

**Figure 8: SHAP result ex 2 using Random Forest algorithm *Source:* source-code output**

Table

Description automatically generated with low confidence

For this example, in this given local instance using the Random Forest model, SHAP assigns 89% probability that the employee would leave the company, same as LIME, with the similar parameters giving their contributions to each of the probabilities.

The parameters such as StockOptionLevel, JobInvolvement, JobSatisfaction and some more, support the class with ‘Yes’ Attrition class, whereas the attributes such as WorkLifeBalance, Overtime support the class with Attrition ‘No’.

However, in the end, a greater number of features support the Attrition= ‘Yes’ class, therefore receiving the outcome as Attrition Yes.

**4.1.7. LIME Analysis using XG Boost Algorithm: -Example 2:**

**Classified Attrition: ‘Yes’ for XG Boost Model**

The attributes such as Overtime, JobLevel, StockOptionLevel, JobSatisfaction contribute to the attrition ‘Yes’, as specified by the random forest algorithm. However, the difference lies in the probabilities assigned to each parameter by both the Random Forest Algorithm and the XG Boost algorithm, thus increasing the probability of the result given by the model and making it more stable to use for the users.

The probabilities given by the Random Forest Model were 89% for Attrition as ‘Yes’ and 11% as Attrition ‘No’. Although, the XG Boost provides a more stable and upgraded probability outcome such as: 94% for Attrition= ‘Yes’ and just 6% as Attrition= ‘No’. This is because, the probability distribution given to each attribute is different as of Random Forest algorithm.

Thus, the XG Boost model gives a better outcome than the Random Forest model. The following figures show the probabilities given by each attribute for both the classes of Attrition for that given employee.

In this way, LIME is used to provide an explanation as to exactly which attributes contributed to what extent towards each class, making the model more transparent to users and providing a justification behind the model’s outcome.

Table

Description automatically generated with low confidenceA picture containing timeline

Description automatically generated

***Source:* source-code output**

**4.1.8. SHAP Analysis using XG Boost Algorithm: Example 2: -**

****

Similar analysis as LIME is observed in SHAP as well, with a probability of 94% for Attrition rate classified as ‘Yes’ by the XG Boost model.

4.2. EVALUATION OF THE HYPOTHESIS

**4.2.1. Hypothesis 1: -**

1. **Do the Local Model Agnostic techniques (LIME and SHAP) give detailed justification on the attributes used by the machine learning models for the individual predictions upon the produced outcome?**

From the above results, it is observed that both LIME and SHAP methods provide a detailed analysis, proving a probability distribution for all the attributes and their contribution to the final classification result made. With this analysis, the identification of the most important parameters as well as the least important parameters can be identified, making the black box model transparent enough. Using the LIME and SHAP methods provide an explanation on how the black box models arrive at a particular classification result.

Therefore, with this, the null hypothesis can be rejected**,** and the alternate Hypothesis can be confirmed, which is: **The Local Model agnostics techniques (LIME and SHAP) does provide the explanation on the attributes used by the machine learning models to arrive at the final classification.**

**4.2.2. Hypothesis 2: -**

1. **Do both the local model agnostic methods, LIME and SHAP produce the same outcome for the classification results given by the machine learning models, or the results obtained are different?**

For checking the current hypothesis, I took all the samples of the test dataset that is 617 samples and used a for loop to get the probability values obtained through the LIME method for all of them.

Therefore, the output received was the probability values for both Attrition rate ‘Yes’ and ‘No’, which I then stored in a data frame.

Table

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

**Figure 9: Probability Percentage using LIME *Source:* source-code output**

For example, in the above figure, the picture on the left shows the probability values for the instances of the test dataset, i.e., the 8th instance shows the probability as 66% for Attrition ‘No’ and 34% for Attrition ‘Yes’.

For SHAP, the following figures provide the results for all the test data probabilities: -

Table

Description automatically generated **Graphical user interface, text, application, email

Description automatically generated**

**Figure 10: Probability Percentage using SHAP *Source:* source-code output**

As per the above results, both the Local Model agnostics techniques (LIME and SHAP) provide the same results for the same machine learning algorithms to arrive at the final classification result. Also, the following code proves that for all the records(617 cases), LIME and SHAP provide the same probabilities for Attrition.

Graphical user interface, text, application

Description automatically generated

Therefore, for the 2nd Hypothesis, H0 (Null Hypothesis) should be excluded and rejected and the H1 (Alternate Hypothesis) can be accepted, which states that: **both the Local Model agnostics techniques (LIME and SHAP) provide the same results for the same machine learning algorithms to arrive at the final classification result, can be accepted.**

4.3. RESULTS FOR IMAGE CLASSIFICATION:

Both LIME and SHAP provide a very useful analysis when it comes to the image classification model. For image Classification, the Convolution Neural networks are implemented, with very complex model structures, like the VGG-16 architecture. Therefore, very accurate classification results are obtained, however, the working of the model is not clear to the users implementing the model.

Local Model Agnostic methods such as SHAP and LIME are used to make this process clearer and transparent to the user. Also identifying which pixels in the image contributed most for the classification can be understood in simple terms by using these methods.

**4.3.1. Example 1: Using LIME for Image Classification**

The First example taken is of a Dog. When given this image to the model, the CNN architecture correctly classifies it as a dog. However, the user in this case has no idea as to which pixels were taken into consideration by the model to classify the image as a Dog.

To solve this, LIME is used. The following output is given by LIME showing which pixel it took into consideration for classifying the image as a Dog.

Original Image LIME Classification

Graphical user interface

Description automatically generated A person wearing a garment

Description automatically generated with low confidence

**Figure 11: Ex1. Using LIME for Image Classification Source: source-code**

**Analysis using LIME for Classification: -**

In the above LIME Classification, the pixels highlighted in green colour are taken by the LIME classifier, to identify the given image as a Dog. Therefore, it can be clearly observed from the above image that LIME uses most of the pixels focused on the dog and removing all the unnecessary pixels, which are highlighted in ‘Red’ colour like the ‘Human hands’ shown in the picture, thus, correctly identifying the image as a Dog.

The Red highlighted pixels are the ones which are completely ignored by the LIME for classification. Since most the highlighted green pixels are of the dog, CNN the model has been trained well to identify the animals correctly.

This is the use of LIME for classifying the image and to check that even if the classification made by the model is correct, is the model able to identify the animals apart in any provided image.

**4.3.2. Using SHAP for Analysis: -**

Using SHAP for analysis mentioned below of the image, SHAP method highlights the image pixels into the red and blue colours.

The Red colour signifies the important pixels used for image classification, also as per the colour bar shown below, the pixels in the image highlighted in Red are taken by SHAP for classifying the image as a dog. It correctly identifies the pixels of dog, while significantly ignoring the pixels in Blue, which are of the human present in the picture, and thus correctly identifying the image as a dog.

CNN Classification SHAP Explanation

Graphical user interface

Description automatically generatedA picture containing square

Description automatically generatedGraphical user interface

Description automatically generated with low confidence

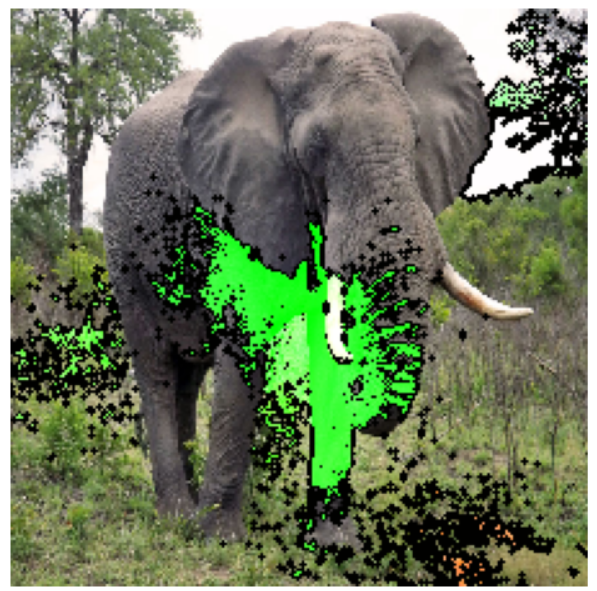
**Figure 12: Ex1. Using SHAP for Image Classification Source: source-code**

**4.3.3. Example 2: LIME Analysis**

The 2nd example taken is of an elephant. When applied LIME on the image, the output of the image is as follows: -

CNN Classification LIME Explanation

A group of elephants walking

Description automatically generated with medium confidence

**Ex2. Using LIME for Image Classification Source: source-code**

As per the above shown LIME explanation for the given image, LIME correctly identifies the pixels associated with the elephant which are highlighted in green colour. As per the LIME explanation, the CNN model is accurately trained to identify the pixels supporting the elephant (like the legs and tusks of elephant) and blurring all the rest of the unused pixels, to classify the image correctly.

Also, the pixels marked in the red colour, are the pixels which are avoided by the classifying CNN model which in this case are the pixels of grass, which are not at all associated with the elephant. This is how LIME explains the image.

**4.3.4. Example 2: SHAP Analysis**

CNN Classification SHAP Explanation

A group of elephants walking

Description automatically generated with medium confidenceA picture containing text

Description automatically generated

**Ex2. Using SHAP for Image Classification Source: source-code**

A picture containing graphical user interface

Description automatically generated

As per the above shown SHAP explanation for the given image, SHAP identifies the pixels associated with the elephant which are highlighted in red colour. As per the SHAP explanation, the CNN model is accurately trained to identify the pixels supporting the elephant (like the legs and tusks of elephant), also identifying some of the elephants in the background and blurring all the rest of the unused pixels, to classify the image correctly.

Also, the pixels marked in the blue colour, are the pixels which are avoided by the classifying CNN model which in this case are the pixels of grass and the surface of road, which are not at all associated with the elephant. This is how SHAP explains the image. The colour bar shown by SHAP explains that the blue colour is used for negative pixels, which are the least important pixels, and the red colour are for the most important pixels.

# CHAPTER 5: CONCLUSION

This entire project was based on the implementation of local Explainable AI techniques on its different applications, like the tabular datasets and on Image Classification, to get a better understanding for the users, both having a technical expertise and non-technical people, for understanding the working behind the model. For this, two local model agnostic techniques were used to make the machine learning black box models transparent, namely LIME and SHAP.

Starting with the tabular datasets, first an exploratory data analysis was performed to obtain a clearer interpretation of the dataset and its attributes. Since the dataset was imbalanced, using SMOTE oversampling technique, it was balanced to get more accurate and unbiased results. Furthermore, the data had to be cleaned and the categorical data had to be encoded by using encoding techniques. After the data was completely processed, different machine learning algorithms were applied on the dataset for the classification of Attrition in employees.

It was observed that the best accuracy was obtained by the XG Boost model, followed by the Random Forest model. However as both are black box prototypes, interpretation of how the model arrived at that prediction is a bit complex. The XG Boost algorithm had an accuracy of 85.08%, whereas the accuracy obtained by the random forest model was 82.82%. However, with this, it was not clear how the model arrived at the decision, thus making it very less transparent to the users. Therefore, using LIME and SHAP helped understand the important features contributing to the model’s prediction for local instances.

Similarly, for the image classification dataset as well, using CNN the model becomes very complex, since the model was configured with many different trials of convolution layers, their respective kernel sizes, using different activation functions for respective convolution layers. Therefore, the working and structure of the CNN model cannot be visualized by users since it contains thousands of trainable parameters. Using LIME and SHAP methods helps identify the important pixels taken by the CNN model to arrive at the result for the classification. LIME highlights the pixels in green colour to showcase the important pixels used for the image classification, whereas the red coloured pixels showcase the least important pixels for classification. On the other hand, SHAP uses

Both the methods, LIME and SHAP have their respective advantages and dis-advantages, however there is no proper conclusion as to which of both techniques is better than the other. LIME is technique which can only be used on Local instances, whereas, SHAP can be used on both local and global techniques, to get a better understanding of the entire model. However, LIME is faster than SHAP, as SHAP’s speed is directly dependent on the dataset’s size. The further analysis which can be performed is to analyse these model agnostic methods on different high-level datasets, to get a better understanding of the working of these two methods.

# CHAPTER 6: REFERENCES

Wang, D., Vermeulen, J., Abdul, A., Kankanhalli, M. and Lim, B., 2018. Trends and Trajectories for Explainable, Accountable and Intelligible Systems: An HCI Research Agenda. Association for Computing Machinery New York, NY, United States, pp.pp. 1-18.

Barua, S., Islam, M., Begum, S. and Ahmed, M., 2022. A Systematic Review of Explainable Artificial Intelligence in Terms of Different Application Domains and Tasks. Applied Sciences, 12(3), p.1353.

  Biau, G., 2012. Analysis of a Random Forests Model. Journal of Machine Learning Research 13 (2012), pp.1063-1095.

Chromic, M., 2020. reSHAPe: A Framework for Interactive Explanations in XAI Based on SHAP.

Dataset Link: [IBM - HR Attrition Dataset](https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset)

Jarrahi, M.H. Artificial intelligence and the future of work: HumanAI symbiosis in organizational decision making. Bus. Horiz. 2018, 61, 577–586

Kim, J., Kim, H. and Jung, H., 2018. Convolutional Neural Network Based Image Processing System. Journal of Information and communication convergence engineering, (2234-8883), pp.160-165.

Knapič, S., Saluja, R., Främling, K. and Malhi, A., 2021. Explainable Artificial Intelligence for Human Decision Support System in the Medical Domain. Machine Learning and Knowledge Extraction, 3(3), pp.740-770.

Legal Considerations: [GDPR legal considerations for using AI](https://ico.org.uk/media/for-organisations/guide-to-the-general-data-protection-regulation-gdpr/the-right-to-be-informed-1-0.pdf)

La Rocca, M. and Perna, C., 2022. Opening the Black Box: Bootstrapping Sensitivity Measures in Neural Networks for Interpretable Machine Learning. *Stats*, 5(2), pp.440-457.

La Rocca, M. and Perna, C., 2015. Model Selection for Neural Network Models: A Statistical Perspective. *Computational Network Theory: Theoretical Foundations and Applications, First Edition*, pp.1-26.

Mardaoui, D. and Garreau, D., 2021. What does LIME really see in images. Proceedings of the 38th International Conference on Machine Learning, (PMLR 139:3620-3629, 2021.).

Mardaoui, D. and Garreau, D., 2021. An Analysis of LIME for Text Data. arXiv:2010.12487, v2(PMLR 130:3493-3501, 2021).

Plaia, A. and Buscemi, S., 2019. Model selection in linear mixed-effect models. AStA Advances in Statistical Analysis, 104(4), pp.529-575.

Qutub, A. et al. 2021, “Prediction of Employee Attrition Using Machine Learning and Ensemble Methods”, International Journal of Machine Learning and Computing, Vol. 11, No. 2.

Rad, P. and Das, A.(2020), “Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey”, IEEE.

Sachin, R., Govind, D., Soman, K.P., Sowmya, V. (2018). Dependency of Various Color and Intensity Planes on CNN Based Image Classification. In: Thampi, S., Krishnan, S., Corchado Rodriguez, J., Das, S., Wozniak, M., Al-Jumeily, D. (eds) Advances in Signal Processing and Intelligent Recognition Systems. SIRS 2017. Advances in Intelligent Systems and Computing, vol 678. Springer, Cham. https://doi.org/10.1007/978-3-319-67934-1\_15

Samek, W. and Müller, K. (2019), “Towards Explainable Artificial Intelligence”, Springer,11700, pp. 5-22.

# Sekaran, K. and Shanmugam, S. (2022), “Interpreting the Factors of Employee Attrition using Explainable AI”, International Conference on Decision Aid Sciences and Applications (DASA), pp. 932-936.

Siegrist, J., Wahrendorf, M., von dem Knesebeck, O., Jurges, H. and Borsch-Supan, A., 2006. Quality of work, well-being, and intended early retirement of older employees--baseline results from the SHARE Study. The European Journal of Public Health, 17(1), pp.62-68.

Tulio Ribeiro, M., Guestrin, C. and Singh, S., 2016. Model-Agnostic Interpretability of Machine Learning. arXiv:1606.05386v1, pp.pp. 91-95.

Venkatadri, T. and Kakollu,V. (2022), “A random forest classification on employee attrition analysis handling of imbalanced data” International Research Journal of Modernization in Engineering Technology and Science, Volume:04/Issue:04.

Vermeire, T., Goethals, S., de Oliveira, R., Brughmans, D. and Martens, D., 2022. Explainable image classification with evidence counterfactual. Pattern Analysis and Applications, 25(2), pp.315-335.

# Xu, F. et al. 2019, “Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges”, Natural Language Processing and Chinese Computing, Springer, vol 11839.

Zhang, Y., Sun, Y., Tan, S., Song, K. and Udell, M., 2022. "Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations.

APPENDIX

1. **Source Code: -**

**Part 1: HR Attrition Dataset – Structured Data: -**

**### Import Libraries**

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.metrics import confusion\_matrix

from sklearn import metrics

from imblearn.over\_sampling import SMOTE

import warnings

warnings.filterwarnings("ignore")

**### Import Dataset**

df = pd.read\_csv("HR-Employee-Attrition.csv")

df1=df.copy()

**### Display Dataset and Properties:**

df.head(5)

df.shape

df.columns

**### Encoding Categorical Values to Numeric values**

df['MaritalStatus'] = df['MaritalStatus'].astype('category')

df['Gender'] = df['Gender'].astype('category')

df['OverTime'] = df['OverTime'].astype('category')

df['BusinessTravel'] = df['BusinessTravel'].astype('category')

df['Attrition'] = df['Attrition'].astype('category')

df['Attrition'] = df['Attrition'].cat.codes

df['MaritalStatus'] = df['MaritalStatus'].cat.codes

df['Gender'] = df['Gender'].cat.codes

df['OverTime'] = df['OverTime'].cat.codes

df['BusinessTravel'] = df['BusinessTravel'].cat.codes

**### Splitting Dataset into X and Y**

X = df[['Age','OverTime','Gender',

       'DistanceFromHome', 'Education', 'HourlyRate',

       'JobInvolvement', 'JobLevel' ,'JobSatisfaction',

        'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',

        'PercentSalaryHike', 'PerformanceRating',

       'RelationshipSatisfaction', 'StockOptionLevel',

       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',

       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',

       'YearsWithCurrManager','MaritalStatus','BusinessTravel']]

Y =  df['Attrition']

class\_names = Y.unique()

class\_names

**### Before Sampling**

sns.countplot(x=df["Attrition"])

plt.title("Before Sampling")

labels = ['No', 'Yes']

x=[0,1]

plt.xticks(x, labels)

**### Using SMOTE to Sample Dataset**

X, Y = SMOTE().fit\_resample(X, Y)

sns.countplot(x=Y)

plt.title("After Sampling")

labels = ['No', 'Yes']

x=[0,1]

plt.xticks(x, labels)

### Performing EDA on Dataset

### Correlation Matrix

corr = X.corr()

plt.figure(figsize=(24,14))

sns.heatmap(corr, annot=True)

plt.show()

**###1. Age vs Attrition**

attr = ['Yes', 'No']

plt.figure(figsize=(8,6))

for a in attr:

    # Subset to the airline

    subset = df1[df1['Attrition'] == a]

    # Draw the density plot

    sns.distplot(subset['Age'], hist = False, kde = True,

                 kde\_kws = {'shade': True, 'linewidth': 2},

                  label = a)

plt.legend(prop={'size': 16}, title = 'Age vs Attrition')

plt.title('Density Plot of Age with Attrition Rate')

plt.xlabel('Age')

plt.ylabel('Density')

**### JobSatisfaction vs Attrition**

plt.figure(figsize=(8,6))

sns.kdeplot(df.JobSatisfaction[(df['Attrition']==0)], color='green', shade=True)

sns.kdeplot(df.JobSatisfaction[(df['Attrition']==1)], color='red', shade=True)

plt.legend(["Attrition-No","Attrition-Yes"])

plt.title("Attrition vs Job Satisfaction")

plt.show()

**### 3. Gender Vs Attrition**

plt.figure(figsize=(8,6))

sns.countplot(x='Gender', hue='Attrition', data=df1,palette="Set1")

plt.legend(["Attrition-Yes","Attrition-No"])

plt.title("Attrition vs Gender")

plt.show()

**### Overtime vs Attrition**

plt.figure(figsize=(8,6))

sns.countplot(x='OverTime', hue='Attrition', data=df1, palette="gist\_heat")

plt.legend(["Attrition-Yes","Attrition-No"])

plt.title("Attrition vs Overtime")

plt.show()

**###JobLevel vs Attrition**

plt.figure(figsize=(8,6))

sns.kdeplot(df.JobLevel[(df['Attrition']==0)], color='Red', shade=True)

sns.kdeplot(df.JobLevel[(df['Attrition']==1)], color='Blue', shade=True)

plt.legend(["Attrition-No","Attrition-Yes"])

plt.title("Attrition vs JobLevel")

plt.show()

**### StockOptionLevel vs Attrition**

plt.figure(figsize=(8,6))

sns.kdeplot(df.StockOptionLevel[(df['Attrition']==0)], color='Green', shade=True)

sns.kdeplot(df.StockOptionLevel[(df['Attrition']==1)], color='Blue', shade=True)

plt.legend(["Attrition-No","Attrition-Yes"])

plt.title("Attrition vs StockOptionLevel")

plt.show()

**###NoofYearsAtCompany vs Attrition**

plt.figure(figsize=(8,6))

x\_ticks = [0,5,10,15,20,25,30,35,40]

sns.kdeplot(df.YearsAtCompany[(df['Attrition']==0)], color='Red', shade=True)

sns.kdeplot(df.YearsAtCompany[(df['Attrition']==1)], color='Green', shade=True)

plt.legend(["Attrition-No","Attrition-Yes"])

plt.title("Attrition vs Years at Company")

plt.xticks(ticks=x\_ticks)

plt.show()

**###JobInvolvement vs Attrition**

plt.figure(figsize=(8,6))

sns.kdeplot(df.JobInvolvement[(df['Attrition']==0)], color='Orange', shade=True)

sns.kdeplot(df.JobInvolvement[(df['Attrition']==1)], color='Blue', shade=True)

plt.legend(["Attrition-No","Attrition-Yes"])

plt.title("Attrition vs Job Involvement")

plt.show()

**### PercentageSalaryHike vs Attrition**

plt.figure(figsize=(8,6))

sns.kdeplot(df.PercentSalaryHike[(df['Attrition']==0)], color='Green', shade=True)

sns.kdeplot(df.PercentSalaryHike[(df['Attrition']==1)], color='Blue', shade=True)

plt.legend(["Attrition-No", "Attrition-Yes"])

plt.title("Attrition vs Percentage Salary Hike")

plt.show()

**###Import Machine Learning libraries**

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

from sklearn.model\_selection import train\_test\_split

**### TrainTestSplit Dataset**

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y,test\_size=.25, random\_state = 42)

print("X\_Train Shape: ",X\_train.shape,"\nX\_Test Shape: ",X\_test.shape,"\nY\_Train Shape: ",Y\_train.shape,"\nY\_Test Shape: ",Y\_test.shape)

**### Logistic Regression**

model\_logreg = LogisticRegression(random\_state = 42, solver='lbfgs',max\_iter=500,verbose=5)

model\_logreg.fit(X\_train,Y\_train)

y\_pred\_lr = model\_logreg.predict(X\_test)

model\_logreg.score(X\_train, Y\_train)\*100

accuracy\_score(Y\_test, y\_pred\_lr)\*100

**### Logistic Regression Confusion Matrix**

confusion\_matrix\_lr = confusion\_matrix(Y\_test, y\_pred\_lr)

confusion\_matrix\_lr

plt.figure(figsize=(6,6))

cm\_display\_lr = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix\_lr, display\_labels = [False, True])

cm\_display\_lr

cm\_display\_lr.plot()

plt.title("Confusion matrix for Logistic Regression")

plt.show()

**### Classification report**

print(classification\_report(Y\_test, y\_pred\_lr))

**### Random Forest Algorithm**

from sklearn.ensemble import RandomForestClassifier

model\_rf = RandomForestClassifier(n\_estimators=300,oob\_score= True,bootstrap=True,max\_depth=5,criterion='gini', random\_state = 42)

model\_rf.fit(X\_train,Y\_train)

y\_pred\_rf = model\_rf.predict(X\_test)

model\_rf.score(X\_train, Y\_train)\*100

print("Accuracy: {}".format(accuracy\_score(Y\_test, y\_pred\_rf)\*100))

**### Plotting Important Features**

plt.figure(figsize=(12,10))

sort\_rf = model\_rf.feature\_importances\_.argsort()

plt.barh(X.columns[sort\_rf], model\_rf.feature\_importances\_[sort\_rf])

plt.title("Most Important features")

**### Confusion matrix Random Forest**

confusion\_matrix\_rf = confusion\_matrix(Y\_test, y\_pred\_rf)

confusion\_matrix\_rf

cm\_display = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix\_rf, display\_labels = [False, True])

cm\_display

cm\_display.plot()

plt.title("Confusion matrix for Random Forest Classification")

plt.show()

**### Classification report**

print(classification\_report(Y\_test, y\_pred\_rf))

**### Plotting Single Decision Tree**

from sklearn import tree

len(model\_rf.estimators\_)

plt.figure(figsize=(20,20))

\_ = tree.plot\_tree(model\_rf.estimators\_[0], feature\_names=X.columns, filled=True)

**### K Nearest Neighbor Algorithm**

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors=5,p=1)

np.random.seed(42)

knn.fit(X\_train, Y\_train)

y\_pred\_knn = knn.predict(X\_test)

knn.score(X\_train, Y\_train)\*100

print("Accuracy: {}".format(accuracy\_score(Y\_test, y\_pred\_knn)\*100))

**### Confusion matrix: KNN Algorithm**

confusion\_matrix\_knn = confusion\_matrix(Y\_test, y\_pred\_knn)

confusion\_matrix\_knn

cm\_display = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix\_knn, display\_labels = [False, True])

cm\_display

cm\_display.plot()

plt.title("Confusion matrix for KNN Classification")

plt.show()

**### Classification Report**

print(classification\_report(Y\_test, y\_pred\_knn))

**### XG Boost Algorithm**

from xgboost import XGBClassifier

model\_xgb = XGBClassifier(loss='log\_loss',base\_score=0.9,learning\_rate=0.09,colsample\_bytree=0.1,max\_depth=4,verbose=3, random\_state = 42)

model\_xgb.fit(X\_train.values, Y\_train.values)

y\_pred\_xgb = model\_xgb.predict(X\_test.values)

model\_xgb.score(X\_train.values, Y\_train.values)\*100

accuracy = accuracy\_score(Y\_test, y\_pred\_xgb)

accuracy\*100

**### Confusion Matrix: XG Boost Algorithm**

confusion\_matrix\_xgb = confusion\_matrix(Y\_test, y\_pred\_xgb)

confusion\_matrix\_xgb

cm\_display = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix\_xgb, display\_labels = [False, True])

cm\_display

cm\_display.plot()

plt.title("Confusion matrix for XG Boost Classification")

plt.show()

**### Classification report**

print(classification\_report(Y\_test, y\_pred\_xgb))

**### Most Important Parameters as per XG Boost**

plt.figure(figsize=(10,10))

sort\_xgb = model\_xgb.feature\_importances\_.argsort()

plt.barh(X.columns[sort\_xgb], model\_xgb.feature\_importances\_[sort\_xgb])

**###Comparing XG Boost and Random Forest important Parameters**

f = plt.figure(figsize=(15,10))

ax = f.add\_subplot(1,2,1)

ax = plt.gca()

plt.barh(X.columns[sort\_rf], model\_rf.feature\_importances\_[sort\_rf])

ax = f.add\_subplot(1,2,2)

ax = plt.gca()

plt.barh(X.columns[sort\_xgb], model\_xgb.feature\_importances\_[sort\_xgb])

**### LIME Analysis**

import lime

import lime.lime\_tabular

X.columns

**### Providing Feature Names for LIME Explainer**

feature\_names = ['Age','OverTime','Gender',

       'DistanceFromHome', 'Education', 'HourlyRate',

       'JobInvolvement', 'JobLevel' ,'JobSatisfaction',

        'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',

        'PercentSalaryHike', 'PerformanceRating',

       'RelationshipSatisfaction', 'StockOptionLevel',

       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',

       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',

       'YearsWithCurrManager','MaritalStatus','BusinessTravel']

**### Displaying 1st Instance**

print(X\_test.iloc[8])

print("\n\n Attrition is: ",Y\_test.iloc[8])

**### Test Results**

Y\_test.head(10)

**### Passing the 1st instance and Feature names to the LIME Explainer**

explainer = lime.lime\_tabular.LimeTabularExplainer(X\_train.values, feature\_names=feature\_names,training\_labels=df['Attrition'], mode='classification')

**### Using Random Forest Model for prediction of result**

exp\_rf1 = explainer.explain\_instance(X\_test.iloc[8], model\_rf.predict\_proba, num\_features=25)

exp\_rf1

###Display Result in Jupiter Notebook

exp\_rf1.show\_in\_notebook(show\_table=False, show\_all=False)

**### SHAP Analysis for the same example using Random Forest Algorithm**

import shap

shap.initjs()

**### Passing the same instance to SHAP Explainer**

explainer\_shaprf1 = shap.TreeExplainer(model\_rf)

shap\_values\_shaprf1 = explainer\_shaprf1.shap\_values(X\_test.iloc[8])

**### Displaying SHAP Analysis Result**

shap.force\_plot(explainer\_shaprf1.expected\_value[0], shap\_values\_shaprf1[0], X\_test.iloc[8])

**### Using XG Boost Algorithm for the same instance and LIME Analysis: -**

exp\_xgb = explainer.explain\_instance(X\_test.iloc[8], model\_xgb.predict\_proba, num\_features=25)

exp\_xgb.show\_in\_notebook(show\_table=True, show\_all=True)

**### Using XG Boost Algorithm for the same instance and SHAP Analysis: -**

explainer\_shapxgb1 = shap.KernelExplainer(model\_xgb.predict\_proba, X\_train)

shap\_values\_shapxgb1 = explainer\_shapxgb1.shap\_values(X\_test.iloc[8])

shap.force\_plot(explainer\_shapxgb1.expected\_value[0], shap\_values\_shapxgb1[0], X\_test.iloc[8])

**### 2nd Example Description**

print(X\_test.iloc[3])

print("\n\n Attrition is: ",Y\_test.iloc[3])

**### LIME Explanation of 2nd example using Random Forest Algorithm**

exp\_rf2 = explainer.explain\_instance(X\_test.iloc[3], model\_rf.predict\_proba, num\_features=25)

exp\_rf2.show\_in\_notebook(show\_table=True, show\_all=True)

**### SHAP Explanation of 2nd example using Random Forest Algorithm**

explainer\_shaprf2 = shap.TreeExplainer(model\_rf)

shap\_values\_shaprf2 = explainer\_shaprf2.shap\_values(X\_test.iloc[3])

shap.force\_plot(explainer\_shaprf2.expected\_value[1], shap\_values\_shaprf2[1], X\_test.iloc[3])

**### LIME Explanation of 2nd example using XG Boost Algorithm**

exp\_xgb2 = explainer.explain\_instance(X\_test.iloc[3].values, model\_xgb.predict\_proba, num\_features=25, num\_samples=2500)

exp\_xgb2.show\_in\_notebook(show\_table=True, show\_all=True)

**### SHAP Explanation of 2nd example using XG Boost Algorithm**

explainer\_shapxgb2 = shap.KernelExplainer(model\_xgb.predict\_proba, X\_train)

shap\_values\_shapxgb2 = explainer\_shapxgb2.shap\_values(X\_test.iloc[3])

shap.force\_plot(explainer\_shapxgb2.expected\_value[1], shap\_values\_shapxgb2[1], X\_test.iloc[3])

**### Analysis of Hypothesis**

**### Storing LIME results for entire Test dataset**

l=[]

for n in range(0,X\_test.shape[0]):

     exp = explainer.explain\_instance(X\_test.values[n], model\_rf.predict\_proba, num\_features=20)

     a=exp.predict\_proba.tolist()

     l.append(a)

dff = pd.DataFrame(l)

dff1 = np.round(dff,2)

dff1 = dff1.rename(columns={0:"No", 1:"Yes"})

dff1.head(9)

**### Storing SHAP results for entire Test dataset**

m=[]

for n in range(0,X\_test.shape[0]):

    expshap = explainer\_shaprf1.shap\_values(X\_test.values[n])

    b=np.round(explainer\_shaprf1.expected\_value[0]+expshap[0].sum(),2).tolist()

    m.append(b)

p=[]

for n in range(0,X\_test.shape[0]):

    expshap = explainer\_shaprf1.shap\_values(X\_test.values[n])

    c=np.round(explainer\_shaprf1.expected\_value[1]+expshap[1].sum(),2).tolist()

    p.append(c)

dff2 = pd.DataFrame({'No': m, 'Yes': p})

dff2.head(9)

**### Comparing both datasets (dff1 and dff2) to check if LIME and SHAP produce same results**

print(dff1.equals(dff2))

**B) Image Analysis Code**

**### Importing Libraries**

import numpy as np

import matplotlib.pyplot as plt

import os

import random

import cv2

import warnings

warnings.filterwarnings("ignore")

**### Assigning path to the dataset**

init\_path = 'C:\\Users\\Aditya\\Downloads\\Dataset\\archive (7)\\raw-img'

**### defining class names for all 10 Animal Classes**

cls = ['cane', 'cavallo', 'elefante', 'farfalla', 'gallina', 'gatto', 'mucca', 'pecora', 'ragno', 'scoiattolo']

**### Assigning each image in dataset to its respective class name**

for c in cls:

    path = os.path.join(init\_path, c)

    for im\_path in os.listdir(path):

        img = cv2.imread(os.path.join(path, im\_path))

        break

    break

**###Reshaping the image to standard size: 224x224**

df = []

for cl in cls:

    cls\_num = cls.index(cl)

    path = os.path.join(init\_path, cl)

    for img in os.listdir(path):

        img = cv2.imread(os.path.join(path, img))

        resized\_img = cv2.resize(img, (224,224))

        df.append([resized\_img, cls\_num])

**# Appending labels and pictures to an array**

a =  []

b = []

for pic, label in df:

    a.append(pic)

    b.append(label)

c = np.array(a)

d = np.array(b)

print(c.shape)

print(d.shape)

**### Importing rest of the libraries**

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.layers import AveragePooling2D

from tensorflow.keras.models import Model

from tensorflow.keras.preprocessing.image import img\_to\_array

from tensorflow.keras.preprocessing.image import load\_img

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report

**### Converting Scientific Class names to normal common names**

new\_var = ["butterfly", "elephant", "dog", "chicken", "cat", "sheep", "squirrel", "cow" , "spider", "horse"]

**###TrainTestSplit of dataset**

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(x, y, test\_size=0.20)

print(X\_train.shape, X\_test.shape, Y\_train.shape, Y\_test.shape)

**### Building the CNN Model**

import tensorflow as tf

from tensorflow.keras import layers, models

from keras.models import Sequential

from keras.layers import Dropout

from keras.layers import Flatten

from keras.layers import Dense

model = models.Sequential()

model.add(layers.Conv2D(32, (5, 5), activation='relu', input\_shape=(224, 224, 3)))

model.add(layers.Conv2D(32, (5, 5), activation='relu', input\_shape=(224, 224, 3)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(Dropout(0.25))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())

model.add(layers.Dense(1024, activation='relu'))

model.add(Dropout(0.5))

model.add(layers.Dense(10, activation='softmax'))

model.summary()

**### Compiling the model**

model.compile(optimizer="adam",loss="sparse\_categorical\_crossentropy",metrics=["accuracy"])

**### Fitting the model: total time taken: 8.75 hours**

history=model.fit(X\_train,Y\_train,epochs=9, batch\_size=50)

p = model.predict(X\_test)

**### Saving the HDF5 File**

model.save('new\_animal-10.hdf5')

**###Testing Outputs:**

**### Taking the image and reshaping it as per required size, then using the above trained model for Classifying the image.**

**1.**

from tensorflow.keras.preprocessing import image

from tensorflow.keras.models import load\_model

img1 = image.load\_img('C:\\Users\\Aditya\\Downloads\\Dataset\\archive (7)\Test Data\\dg1.jpg', target\_size= (224, 224))

plt.figure(figsize=(14,6))

plt.imshow(img1)

img1 = image.img\_to\_array(img1)

img1 = np.expand\_dims(img1,axis=0)

model = load\_model('new\_animal-10.hdf5')

result\_b = model.predict(img1)

new\_var[np.argmax(result\_b[0])]

**A dog sitting in the grass

Description automatically generated with medium confidence**

**2.**

img2 = image.load\_img("C:\\Users\\Aditya\\Downloads\\Dataset\\archive (7)\\Test Data\\dg3.jpg", target\_size= (224, 224))

plt.imshow(img2)

img2 = image.img\_to\_array(img2)

img2 = np.expand\_dims(img2,axis=0)

model = load\_model('new\_animal-10.hdf5')

result\_d = model.predict(img2)

new\_var[np.argmax(result\_d[0])]

**A picture containing graphical user interface

Description automatically generated**

**3.**

img3 = image.load\_img("C:\\Users\\Aditya\\Downloads\\Dataset\\archive (7)\\Test Data\\elephant.jpg", target\_size= (224, 224))

plt.imshow(img3)

img3 = image.img\_to\_array(img3)

img3 = np.expand\_dims(img3,axis=0)

model = load\_model('new\_animal-10.hdf5')

result\_d = model.predict(img3)

new\_var[np.argmax(result\_d[0])]

**A group of elephants walking

Description automatically generated with low confidence**

**4.**

img5 = image.load\_img("C:\\Users\\Aditya\\Downloads\\Dataset\\archive (7)\\Test Data\\she1.jpg", target\_size= (224, 224))

plt.imshow(img5)

img5 = image.img\_to\_array(img5)

img5 = np.expand\_dims(img5,axis=0)

model = load\_model('new\_animal-10.hdf5')

result\_d = model.predict(img5)

new\_var[np.argmax(result\_d[0])]

**A sheep standing in a field

Description automatically generated with medium confidence**

**### LIME Analysis on the above images: -**

import lime

from lime import lime\_image

explainer = lime\_image.LimeImageExplainer()

**### Example 1:**

explanation1 = explainer.explain\_instance(img2[0].astype('double'), model.predict,top\_labels=10, hide\_color=0, num\_samples=1500)

Shape

Description automatically generated with medium confidence

**### Segmenting the Image and marking Boundaries for LIME**

from skimage.segmentation import mark\_boundaries

c,masker = explanation1.get\_image\_and\_mask(explanation1.top\_labels[0], positive\_only=False, num\_features=10, hide\_rest=False)

fig = plt.figure(figsize=(14,6))

ax2 = fig.gca()

ax2.imshow(mark\_boundaries(c, masker).astype('uint8'))

ax2.axis('off')

**### SHAP Analysis for the Same image**

import shap

shap.initjs()

**### Importing the blur masker**

masker = shap.maskers.Image("blur(224,224)", (224,224,3))

shap\_explainer1 = shap.Explainer(model, masker, output\_names=new\_var)

shap\_explainer1

**### Generating the Output for Given Image, with a total of 3000 evaluations**

shap\_values = shap\_explainer1(img2, batch\_size=50, max\_evals=3000, outputs=shap.Explanation.argsort.flip[:1])

**### Plotting SHAP Analysis**

shap.image\_plot(shap\_values)

**### Example 2: Using LIME**

explanation2 = explainer.explain\_instance(img6[0].astype('double'), model.predict, top\_labels=10, hide\_color=0, num\_samples=700)

**### Importing the 2nd Image and Marking boundaries for same**

from skimage.segmentation import mark\_boundaries

c,masker = explanation2.get\_image\_and\_mask(explanation2.top\_labels[0], positive\_only=False, num\_features=10, hide\_rest=False)

fig = plt.figure(figsize=(14,6))

ax2 = fig.gca()

ax2.imshow(mark\_boundaries(c, masker).astype('uint8'))

ax2.axis('off')

**### SHAP Analysis for 2nd Image**

shap\_values = shap\_explainer1(img6, batch\_size=50, max\_evals=3000, outputs=shap.Explanation.argsort.flip[:1])

shap.image\_plot(shap\_values)

**2. Project Plan: -**

|  |  |  |
| --- | --- | --- |
| **Project Plan:- Phase 1** | | |
| **Sr No** | **Topic** | **Date** |
| 1. | Basic Study/ Topic Selection | 20th May – 1st June |
| 2. | Discussion of Scope | 2nd June – 5th June |
| 3. | Finalizing Parameters | 6th June – 12th June |
| 4. | Discussion with Supervisor | 13th June – 18thth June |
| 5. | DPP Preparation | 19th June – 25th June |
| 6. | Model Preparation | 26th June – 2nd July |
| 7. | Fine-Tuning Results | 3rd July – 9th July |
| 8. | Preparation of Interim Project Report | 10th July – 15th July |
| 9. | Discussion of Results and Updations of Phase 1 results as per Supervisor suggestions. | 16th July – 24th July |
| **Project Plan:- Phase 2** | | |
| **Sr No** | **Topic** | **Date** |
| 1. | Study on the 2nd Part of Project | 25th July – 30th  July |
| 2. | Finalizing Parameters | 31st July to 3rdAugust |
| 3. | Model Preparation and Getting Preliminary Results | 4th August to 14th August |
| 4. | Fine-Tuning Results as per Supervisor Suggestions | 15th August to 25th August |
| 5. | Preparation of Final Project Report | 26th August to 10th September |
| 6. | Discussion with Supervisor for the entire master’s Project | 12th September |

**Table 5.1- Project Plan**

# 3. Supervisor Meeting till Date: -

|  |  |  |
| --- | --- | --- |
| Sr No | Date | Reason |
| 1. | 31/05/2022 | Discussion MSc Project Topic Selection. |
| 2. | 10/06/2022 | Discussion Regarding the Proposal Draft |
| 3. | 24/06/2022 | Discussion Regarding making of Detailed Project Proposal |
| 4. | 12/09/2022 | Discussion regarding Final Results obtained |