Configuration Manual

MSc Research Project
Data Analytics

Oluwaseun Ayokunbi Ogunbowale
Student ID: X21193355

School of Computing
National College of Ireland

Supervisor: Vitor Horta.

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Oluwaseun Ayokunbi Ogunbowale
Student ID:	X21193355
Programme:	Data Analytics
Year:	2022-2023
Module:	MSc Research Project
Supervisor:	Vitor Horta
Submission Due Date	18 / 09 / 2023
Project Title:	Explaining Neural Networks and Random Forests for Employee Retention
Word Count:	1217
Page count:	15

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ………………Oluwaseun Ayokunbi Ogunbowale……..……………………………………

Date: ………18/09/2023………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Oluwaseun Ayokunbi Ogunbowale
X21193355

1 Introduction

This is the configuration manual that serves as a directive required to reproduce the research project with step-by-step guidelines of code implementation spanning from Hardware and software requirements, data selection to execution of the project. (with complete implementation procedure clearly stated)

2 System Requirement

The table below contains the hardware requirements for the research.

Table 1: Hardware requirements

S/NO

1	Device name	LAPTOP-Q52S7GLR
2	Processor	11th Gen Intel(R) Core (TM) i5-1135G7 @ 2.40GHz 2.42 GHz
3	RAM	16.0 GB (15.8 GB usable)
4	Type	64-bit operating system, x64-based processor

Table 1.

Software Requirements.
The software specifications are listed below.

- Anaconda Navigator 3 for Windows
- Jupyter Notebook (Version 6.4.12)
- Python (Version 3.9)

3.0 Data selection.

The datasets used were obtained from Kaggle's repository and two different datasets were considered for the research work with the link listed below.

I. https://www.kaggle.com/datasets/giripujar/hr-analytics

II. https://www.kaggle.com/datasets/jpmiller/employee-attrition-for-healthcare

Click on the links above to download the datasets and stored it as CSV file on your system.

3.1 Importing of libraries and loading of Dataset on Jupyter Notebook.
Open Jupyter Notebook to select a new Python file, import and install all the necessary libraries that are needed by this research using pip install ('Packages name') to install the required libraries, and continue with the installation as the need arises.

```
[1]: import numpy as np
     from sklearn.preprocessing import LabelEncoder
     import pandas as pd
     import itertools
     import random
     import matplotlib
     import matplotlib.pyplot as plt
     import seaborn as sns
     from lime import lime_tabular
     from sklearn.metrics import classification_report
     from sklearn.utils import resample
     from sklearn.preprocessing import MaxAbsScaler
     from sklearn.model_selection import train_test_split
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.neural_network import MLPClassifier
     from sklearn.metrics import accuracy_score
     from sklearn.metrics import confusion_matrix
     from sklearn import metrics
     import lime
     import shap
```

Figure 2: Libraries installation.

```
#Import the dataset.
hremployee = pd.read_csv("C:\\Users\\oluwa\\Desktop\\HR_comma_sep.csv")
```

```
hremployee.head(5)#viewing the five rows in the dataset
```

satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	Department	salary
0.38	0.53	2	157	3	0	1	0	sales	low
0.80	0.86	5	262	6	0	1	0	sales	medium
0.11	0.88	7	272	4	0	1	0	sales	medium
0.72	0.87	5	223	5	0	1	0	sales	low
0.37	0.52	2	159	3	0	1	0	sales	low

```
hratrition = pd.read_csv("C:\\Users\\oluwa\\desktop\\watson_healthcare_modified.csv")
```

```
hratrition.head(5)
```

	EmployeeID	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	RelationshipSatisfac
0	1313919	41	No	Travel_Rarely	1102	Cardiology	1	2	Life Sciences	1	...	
1	1200302	49	No	Travel_Frequently	279	Maternity	8	1	Life Sciences	1	...	
2	1060315	37	Yes	Travel_Rarely	1373	Maternity	2	2	Other	1	...	
3	1272912	33	No	Travel_Frequently	1392	Maternity	3	4	Life Sciences	1	...	
4	1414939	27	No	Travel_Rarely	591	Maternity	2	1	Medical	1	...	

5 rows × 35 columns

Figure 3: Dataset and attributes checked.

Import the dataset that is already saved CSV file into the Python environment on the jupyter notebook with the pandas library and view the first five rows in the dataset as shown in Figure 3 to see the labeling and naming of the attributes in the columns on the data frame and the attributes of the dataset as the first dataset Hr analytic dataset consists of 14999 rows with 10 columns and the second dataset employee attrition for healthcare dataset consist of 1676 rows with 35 columns.

```
hremployee.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   satisfaction_level     14999 non-null  float64
 1   last_evaluation        14999 non-null  float64
 2   number_project         14999 non-null  int64
 3   average_montly_hours   14999 non-null  int64
 4   time_spend_company     14999 non-null  int64
 5   work_accident          14999 non-null  int64
 6   left                   14999 non-null  int64
 7   promotion_last_5years  14999 non-null  int64
 8   department             14999 non-null  object
 9   salary                 14999 non-null  object
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

Figure 4a. Hr analytic dataset attributes.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1676 entries, 0 to 1675
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   EmployeeID                1676 non-null   int64
 1   Age                       1676 non-null   int64
 2   Attrition                 1676 non-null   object
 3   BusinessTravel            1676 non-null   object
 4   DailyRate                 1676 non-null   int64
 5   Department                1676 non-null   object
 6   DistanceFromHome          1676 non-null   int64
 7   Education                 1676 non-null   int64
 8   EducationField            1676 non-null   object
 9   EmployeeCount             1676 non-null   int64
 10  EnvironmentSatisfaction   1676 non-null   int64
 11  Gender                    1676 non-null   object
 12  HourlyRate                1676 non-null   int64
 13  JobInvolvement            1676 non-null   int64
 14  JobLevel                  1676 non-null   int64
 15  JobRole                   1676 non-null   object
 16  JobSatisfaction           1676 non-null   int64
 17  MaritalStatus             1676 non-null   object
 18  MonthlyIncome             1676 non-null   int64
 19  MonthlyRate               1676 non-null   int64
 20  NumCompaniesWorked        1676 non-null   int64
 21  Over18                    1676 non-null   object
 22  OverTime                  1676 non-null   object
 23  PercentSalaryHike         1676 non-null   int64
 24  PerformanceRating         1676 non-null   int64
 25  RelationshipSatisfaction  1676 non-null   int64
 26  StandardHours             1676 non-null   int64
 27  Shift                     1676 non-null   int64
 28  TotalWorkingYears         1676 non-null   int64
 29  TrainingTimesLastYear     1676 non-null   int64
 30  WorkLifeBalance           1676 non-null   int64
 31  YearsAtCompany            1676 non-null   int64
 32  YearsInCurrentRole        1676 non-null   int64
 33  YearsSinceLastPromotion   1676 non-null   int64
 34  YearsWithCurrManager      1676 non-null   int64
dtypes: int64(26), object(9)
memory usage: 458.4+ KB
```

Figure 4b. Employee attrition for healthcare attributes.

4. Dataset Exploration.

The attributes name on the columns consists of both lower case and upper case, it is better to stick to one case out of the two cases used for easy reading of the dataset and for easy reading and computation on the Python platform. The uppercase attributes in the dataset were changed to lowercase.

```
hremployee=hremployee.rename(columns=lambda x: x.lower())#change capital letter of department and work_accident to small letter
```

```
hratrition=hratrition.rename(columns=lambda x: x.lower())
```

```
hratrition.columns
```

```
Index(['employeeid', 'age', 'attrition', 'businesstravel', 'dailyrate',
       'department', 'distancefromhome', 'education', 'educationfield',
       'employeecount', 'environmentsatisfaction', 'gender', 'hourlyrate',
       'jobinvolvement', 'joblevel', 'jobrole', 'jobsatisfaction',
       'maritalstatus', 'monthlyincome', 'monthlyrate', 'numcompaniesworked',
       'over18', 'overtime', 'percentsalaryhike', 'performancerating',
       'relationshipsatisfaction', 'standardhours', 'shift',
       'totalworkingyears', 'trainingtimeslastyear', 'worklifebalance',
       'yearsatcompany', 'yearsincurrentrole', 'yearssincelastpromotion',
       'yearswithcurrmanager'],
      dtype='object')
```

Figure 4: Renaming the column's name.

The statistical value of the dataset is displayed below with mean, standard deviation, minimum, maximum and quantile.

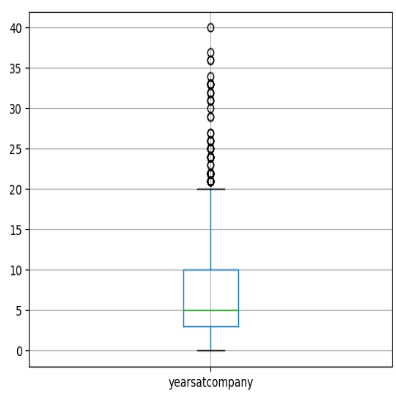
```
hremployee.describe()# statistical description of the dataset
```

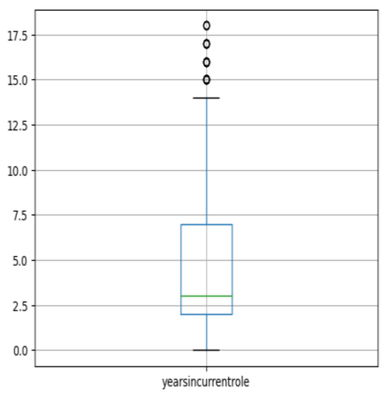
	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.612834	0.716102	3.803054	201.050337	3.498233	0.144610	0.238083	0.021268
std	0.248631	0.171169	1.232592	49.943099	1.460136	0.351719	0.425924	0.144281
min	0.090000	0.360000	2.000000	96.000000	2.000000	0.000000	0.000000	0.000000
25%	0.440000	0.560000	3.000000	156.000000	3.000000	0.000000	0.000000	0.000000
50%	0.640000	0.720000	4.000000	200.000000	3.000000	0.000000	0.000000	0.000000
75%	0.820000	0.870000	5.000000	245.000000	4.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	7.000000	310.000000	10.000000	1.000000	1.000000	1.000000

Figure 5: Statistical description and values of the dataset.

2. The two dataset was checked for null values and duplicated values. There was no missing value in the datasets and no duplicate in the second dataset, but the first dataset contains 3008 duplicated values.

The dataset was visualized with a boxplot to check if there are outliers in the dataset, utilizing the seaborn library, boxplots are used for outlier analysis and there are several outliers that need to be dealt with as shown below.





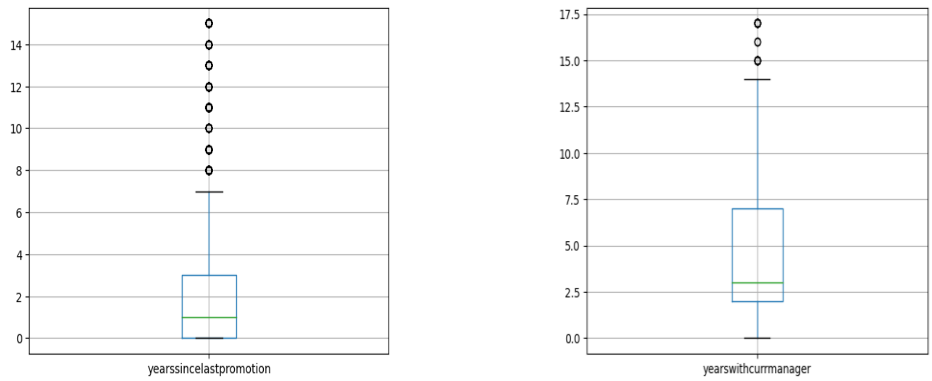
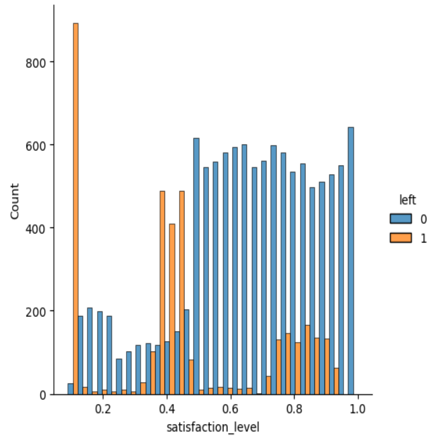


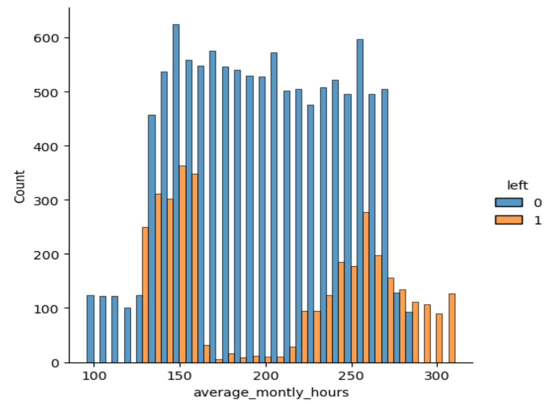
Figure 6: Box plots for the variables

5 Data visualization.

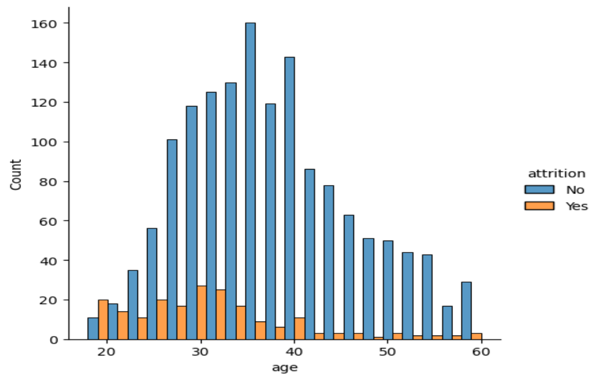
Visualization was done on the dataset with a histogram and pie chart to check the interaction of the target variable with other variables to observe the trend and pattern in the dataset that are not visible to the eyes.



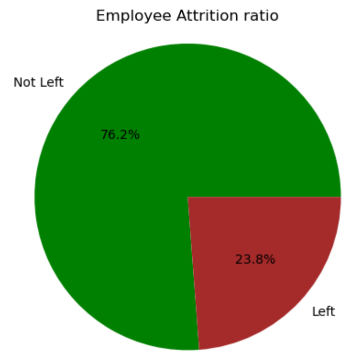
7a. Histogram of satisfaction level vs left



7b. Histogram of average_montly vs left



7c. Histogram of age vs left



7d. Target Variable

Figure7: Attrition rate visualizations with histogram and pie chart.

From Figure 7, the relationship between the target variable and the independent variable is visible and the pie chart shows that the class in the target variable is imbalanced because the class is not represented.

6. Correlation of numerical variables

```
numeric_columns = hremployee.select_dtypes(include=["int64", "float64"])
corr_matrix = numeric_columns.corr()
```

```
#correlation plot
import matplotlib.pyplot as plt
import seaborn as sns
fig = plt.figure(figsize=(8,7))
sns.heatmap(corr_matrix, annot=True)
plt.show()
```

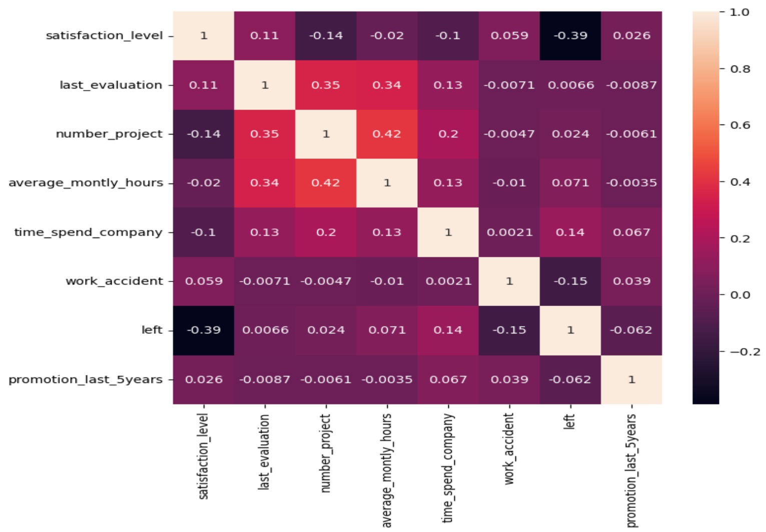


Figure 8: Heap map for correlation of variables

7. Label encoding.

```
import numpy as np
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
```

```
hratrition['businesstravel'] = labelencoder.fit_transform(hratrition['businesstravel'])
hratrition['department'] = labelencoder.fit_transform(hratrition['department'])
hratrition['educationfield'] = labelencoder.fit_transform(hratrition['educationfield'])
hratrition['gender'] = labelencoder.fit_transform(hratrition['gender'])
```

```
hratrition['jobrole'] = labelencoder.fit_transform(hratrition['jobrole'])
hratrition['maritalstatus'] = labelencoder.fit_transform(hratrition['maritalstatus'])
#hratrition['over18'] = labelencoder.fit_transform(hratrition['over18'])
hratrition['overtime'] = labelencoder.fit_transform(hratrition['overtime'])
hratrition['attrition'] = labelencoder.fit_transform(hratrition['attrition'])
```

Figure 9: Label encoding of some attributes.

The proportion of the target variable is imbalance as shown in the pie chart above so the target variable was upsampled in figure 10 below to remove bias in the class and improve the performance of the model and scaling was done to normalize all the variables as differences in feature scale can affect the result of the model present as shown in the figure below.

```
hremployee['left'].value_counts()
```

```
0    7566
1    1775
Name: left, dtype: int64
```

```
hremployee_majority=hremployee[hremployee['left']==0]
hremployee_minority=hremployee[hremployee['left']==1]
```

```
print("Majority class {}".format(hremployee_majority.shape))
print("Minority class {}".format(hremployee_minority.shape))
```

```
Majority class (7566, 10)
Minority class (1775, 10)
```

```
#Upsampling is done to equalize the class or  balance the imbalance class
from sklearn.utils import resample
#Upsample the minority class
hremployee_minority_upsampled = resample(hremployee_minority, replace=True, n_samples=len(hremployee_majority), random_state=42)
from sklearn.utils import resample
```

```
# Concatenate the upsampled minority class with the majority class
hremployee = pd.concat([hremployee_majority,hremployee_minority_upsampled])
```

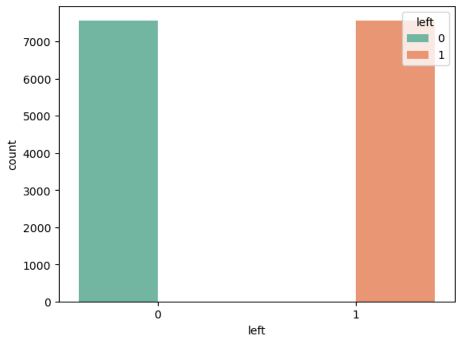
```
# Concatenate the upsampled minority class with the majority class
hremployee = pd.concat([hremployee_majority,hremployee_minority_upsampled])
```

```
hremployee['left'].value_counts()
```

```
0    7566
1    7566
Name: left, dtype: int64
```

```
#application of scaling of dataset
from sklearn.preprocessing import MaxAbsScaler
import pandas as pd
scaler = MaxAbsScaler() #to standardize the figures for each column before modelling
scaler.fit(hremployee)
scaled = scaler.transform(hremployee)
scaled_data = pd.DataFrame(scaled, columns=hremployee.columns)
hremployee=scaled_data
```

Figure 10: upsampling and scaling of the variables



Balanced class for the target variable.

8. Training and splitting of the dataset

The dataset was splited into test and train sets using the train _test _split function from the scikit-learn library in Python this is mostly used to split the dataset into subsets purposely done to check the performance of the model as training set is used to train and the testing test is used to evaluate or validate the performance of the model. The dataset is partitioned into ratios of 80% and 20% respectively as the test part is 20 % as shown in the figure below.

```
from sklearn.model_selection import train_test_split

X = hremployee.drop('left', axis=1)
y = hremployee['left']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 11: Dataset splitting in Ration 80% AND 20%

Models Application

Random Forest.

Two models were applied to the dataset, these are Random forests and Neural network model classification was done and the result evaluated.

```
emp.fit(X_train, y_train)
```

```
RandomForestClassifier(n_estimators=10, random_state=42)
```

```
y_predict = emp.predict(X_test)
threshold = 0.5
y_predict_binary = np.where(y_predict >= threshold, 1, 0)
```

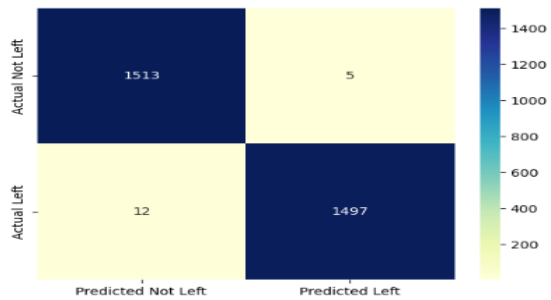
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
print("Train Accuracy = ", emp.score(X_train, y_train))
print("Test Accuracy = ", emp.score(X_test, y_test))
```

```
Train Accuracy =  0.996191282268303
Test Accuracy =  0.9864636209813875
```

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix

# Create the confusion matrix
cm = confusion_matrix(y_test, y_predict_binary)
cm
```

```
sns.heatmap(conf_mat, annot = True, fmt='d', cmap="YlGnBu")
plt.show()
```




```
78]: from sklearn.metrics import classification_report
print('Classification Report: ', classification_report(y_test, y_predict_binary))
```

```
Classification Report:                precision    recall  f1-score   support

         0.0       0.99      1.00      0.99      1518
         1.0       1.00      0.99      0.99      1509

    accuracy                           0.99      3027
   macro avg       0.99      0.99      0.99      3027
weighted avg       0.99      0.99      0.99      3027
```

```
79]: from sklearn import metrics
#calculate AUC of model
auc = metrics.roc_auc_score(y_test, y_predict_binary)
#print AUC score
print(auc)# printing the Auc score
```

```
0.9943769530380299
```

Figure 12: Random Forests Model.

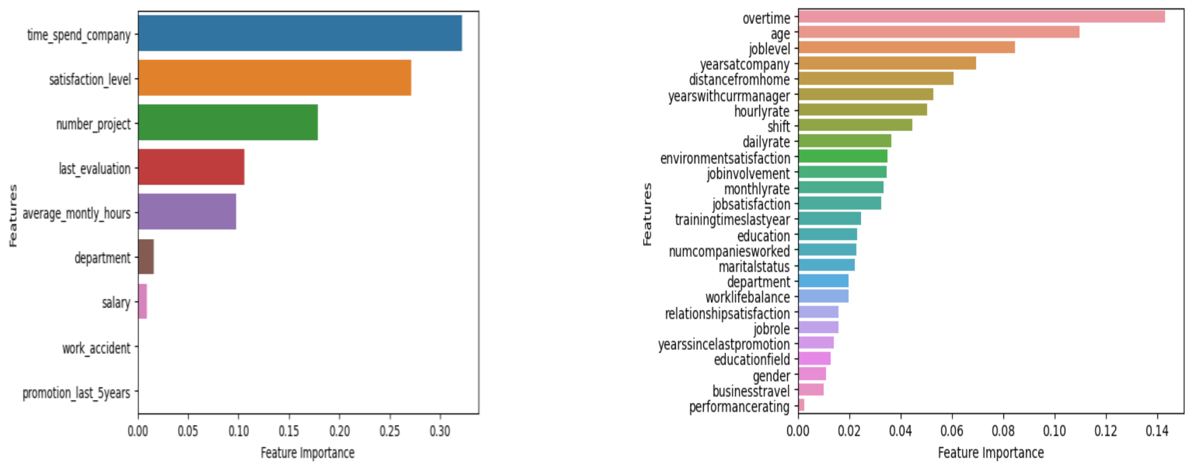
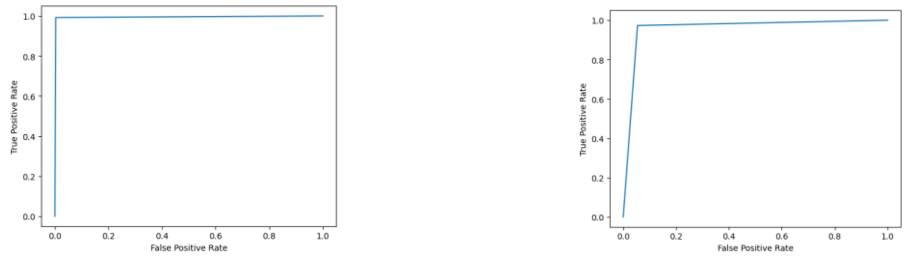


Figure 13: Features importance in the HR analytics and employee attrition healthcare dataset

```
fpr, tpr, _ = metrics.roc_curve(y_test,  y_predict_binary)# creating the curve
plt.plot(fpr,tpr)
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



Roc Curve in Random Forest and Neural Network.

Neural Network.

```
# scaling the data before being use for modelling.
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)
```

```
# building the model
from sklearn.neural_network import MLPClassifier
Nmodel = MLPClassifier(hidden_layer_sizes=(64, 32, X.shape[1]), max_iter=500, random_state=105)
```

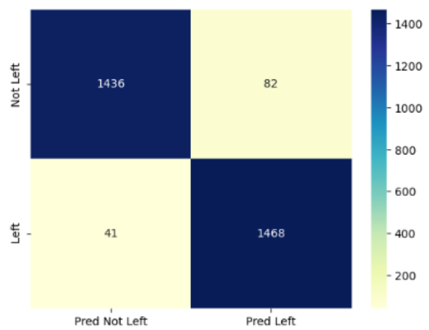
```
Nmodel.fit(X_train, y_train)
y_pred = Nmodel.predict(X_test)
```

```
#from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
print("Train Accuracy = ", Nmodel.score(X_train, y_train))
print("Test Accuracy = ", Nmodel.score(X_test, y_test))
#same as the accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Train Accuracy =  0.9703428335398596
Test Accuracy =  0.9593657086223984
Accuracy: 0.9593657086223984
```

```
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)
```

```
Confusion Matrix:
[[1436   82]
 [  41 1468]]
```




```
from sklearn.metrics import classification_report
print(classification_report(y_test,Nmodel.predict(X_test)))
```

```
              precision    recall  f1-score   support

         0.0       0.97      0.95      0.96      1518
         1.0       0.95      0.97      0.96      1509

    accuracy                           0.96      3027
   macro avg       0.96      0.96      0.96      3027
weighted avg       0.96      0.96      0.96      3027
```

Figure 14: Neural Network Model

Explainable AI Implementation
Experiment 1.

Lime in Random Forest in Hr analytic data

The lime model was applied Random forest model to give interpretations to the output of the result in other to explain factors that contributed to employee retention in a company by giving a local interpretation to the instances of the variable being explained.

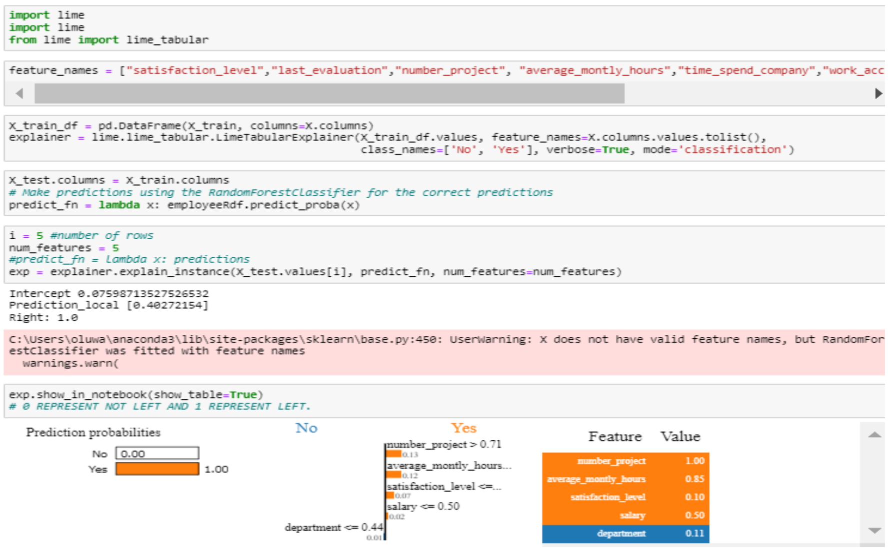


Figure 15a: Lime interpretation of Random Forest Model

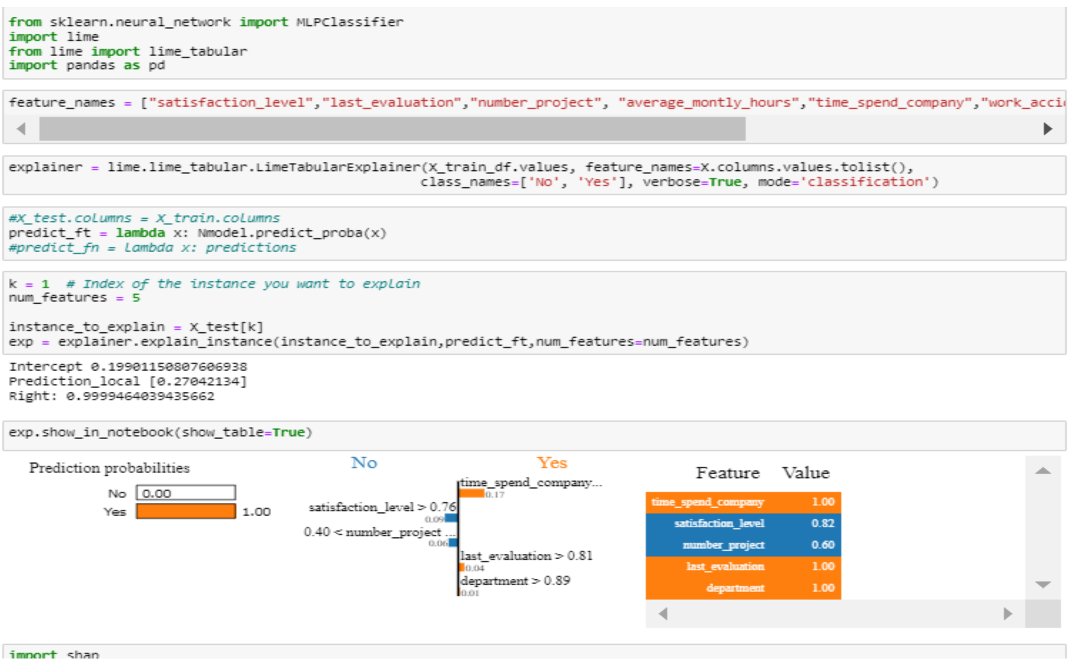
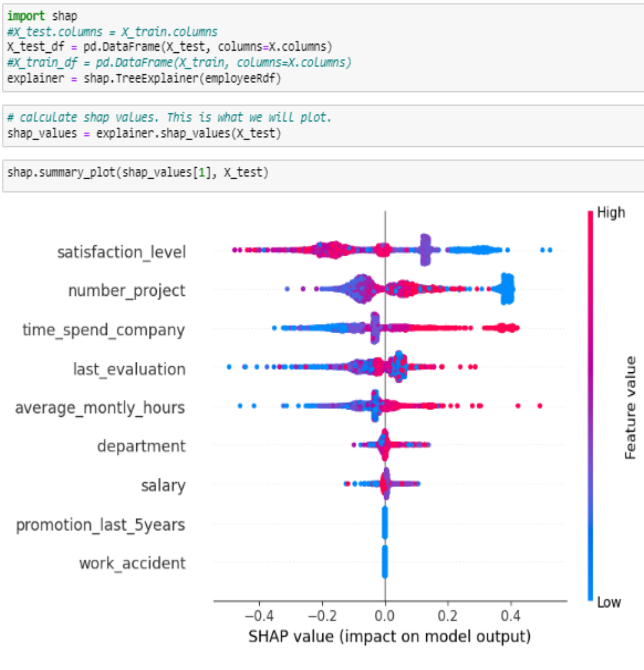


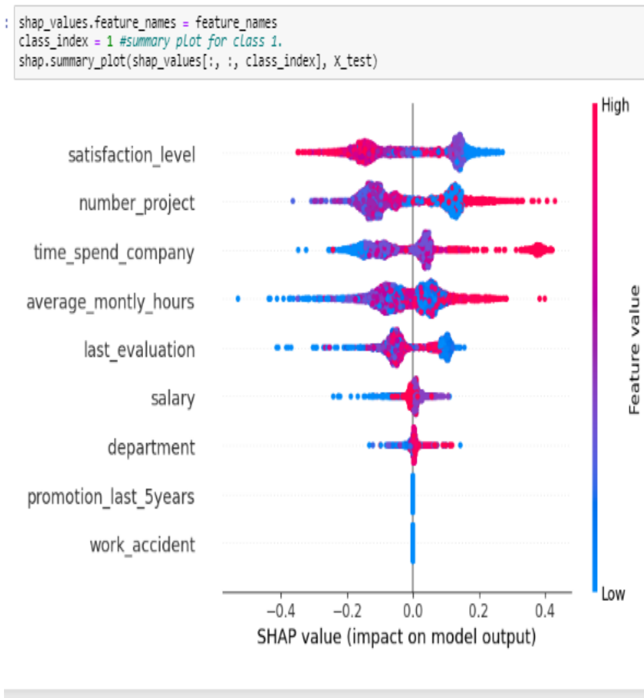
Figure 15b:Lime in Neural network Model

Shap was also applied to the same model to interpret the global impact of the factors on the model prediction as shown in figure 16.

The summary plot shows the significant impact of the features on employee attrition in order of their impact and effect as the feature with high shap values has a significant effect on model prediction.

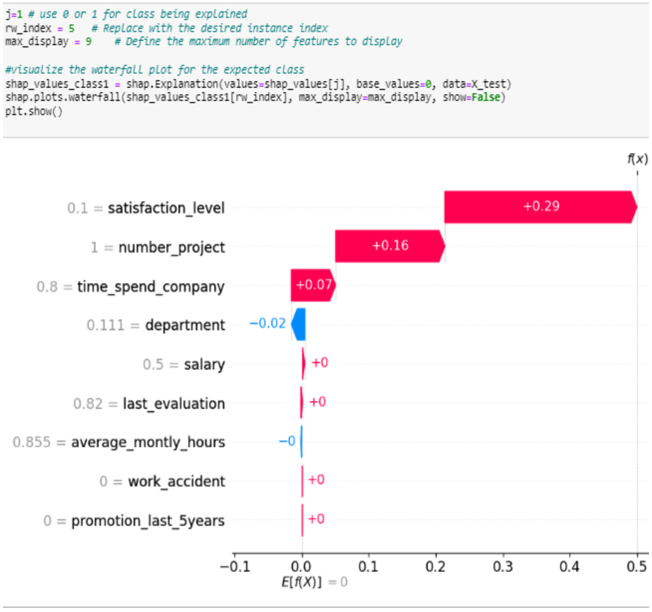


Summary plot of Random Forest

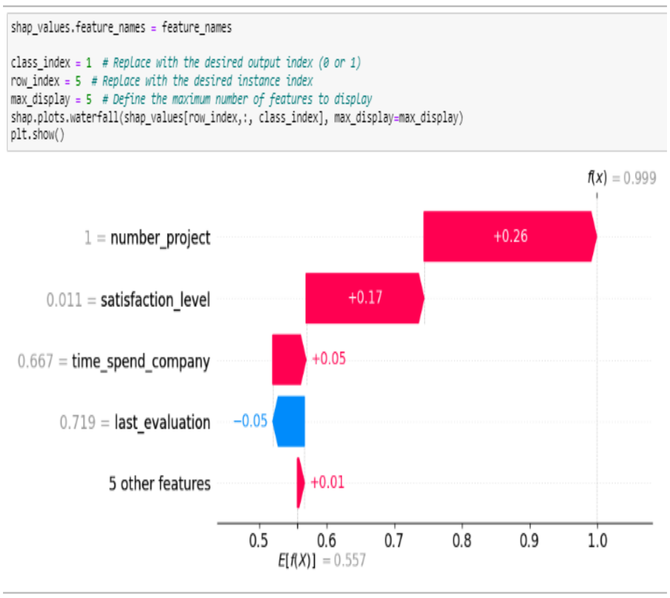


Summary plot of Neural Network models

Figure 16: Summary plot of Random Forest and Neural Network models



Waterfall plot in Random forest



Waterfall plot in Neural network

Model Figure 17 : Waterfall plot in Random forest and Neural network Model

The waterfall plot in figure 16 shows the contribution of each of the factors on the model prediction and interactions that occur within the factors to arrive at the final output given by the model.

Figure 18 Force plot of Random Forests Model.

The forceplot in figure 17 above shows the effect of factors on the model output this works just like in waterfall plots which shows the impact f each of the factors on the model output.

Experiment 2: Lime on Random Forest for employee attrition for healthcare

```
import lime
import lime.lime_tabular
```

```
class_names=['No', 'Yes']
```

```
feature_names= ['age','businesstravel','dailyrate','department','distancefromhome','education','educationfield','environmentsati
```

```
X_train_df = pd.DataFrame(X_train, columns=X.columns)
explainer = lime.lime_tabular.LimeTabularExplainer(X_train_df.values, feature_names=X.columns.values.tolist(),
                                                   class_names=['No', 'Yes'], verbose=True, mode='classification')
```

```
X_test.columns = X_train.columns

# Make predictions using the RandomForestClassifier
predict_fn = lambda x: emp.predict_proba(x)
```

```
k = 2 # number of rows  2$9
num_features = 9# number of features
#predict_fn = lambda x: predictions
exp = explainer.explain_instance(X_test.values[k], predict_fn, num_features=num_features)
```

```
Intercept 0.4820591454818457
Prediction_local [-0.04028488]
Right: 0.1
```

```
X does not have valid feature names, but RandomForestClassifier was fitted with feature names
```

```
exp.show_in_notebook(show_table=True) #showing lime result
```

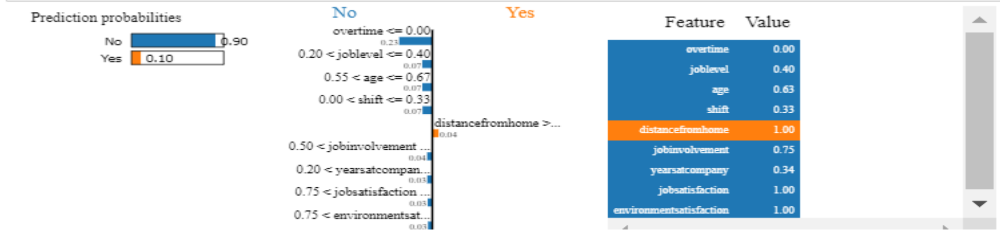


Figure 19: Lime on Random Forest for employee attrition dataset

SHAP

```
import shap #import the shap library
```

```
X_test_df = pd.DataFrame(X_test, columns=X.columns)
explainer = shap.TreeExplainer(emp)
```

```
# calculate shap values to plot the variables
shap_values = explainer.shap_values(X_test)
```

```
shap.summary_plot(shap_values[0], X_test)# summary plot for class no
```

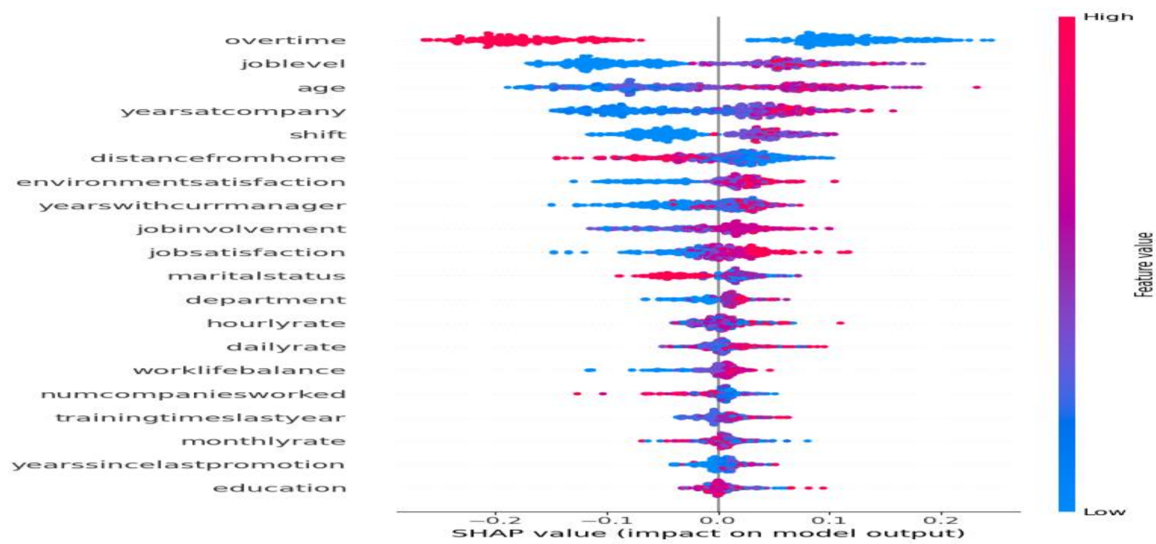


Figure 20 : shap on Random forest Model.

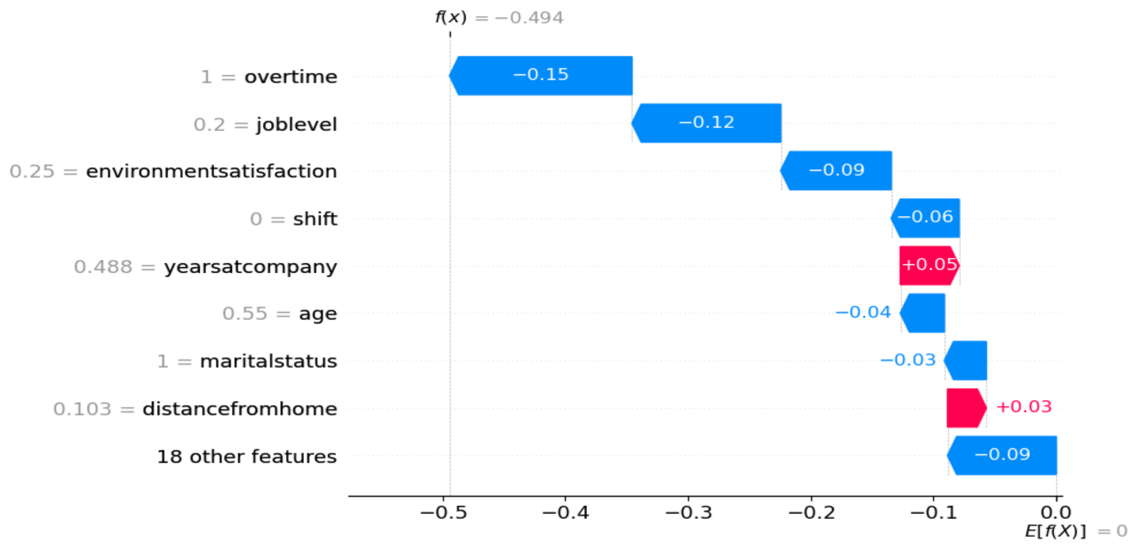


Figure 21: Waterfall on Random forest Model.

Experiment 2: Lime on Neural Network for employee attrition for healthcare

Lime was applied to the Neural network model as seen in the figure below.

```
import lime #import the lime bibrary
from lime import lime_tabular
```

```
feature_names= ['age','businesstravel','dailyrate','department','distancefromhome','education','educationfield','environmentsati
```

```
explainer = lime.lime_tabular.LimeTabularExplainer(X_train_df.values, feature_names=X.columns.values.tolist(),
                                                   class_names=['No', 'Yes'], verbose=True, mode='classification')
```

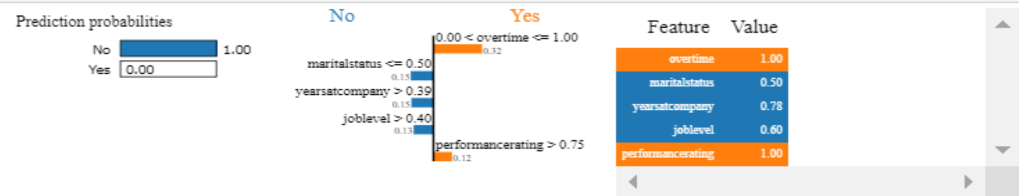
```
X_test.columns = X_train.columns
predict_f = lambda x: Nnmodel.predict_proba(x)
```

```
i = 9 #number of instances or rows being explained
num_features = 5
exp = explainer.explain_instance(X_test.values[1], predict_f, num_features=num_features)
```

```
X does not have valid feature names, but MLPClassifier was fitted with feature names
```

```
Intercept 0.2146884414708377
Prediction_local [0.23267693]
Right: 3.50795141512931e-06
```

```
exp.show_in_notebook(show_table=True)
```




```
import shap
#X_test.columns = X_train.columns
X_test_df = pd.DataFrame(X_test, columns=X.columns)
#X_train_df = pd.DataFrame(X_train, columns=X.columns)
explainer = shap.Explainer(Nnmodel.predict_proba, X_train)
#explainer = shap.Explainer(Nmodel.predict_proba,X_train_df)
```

```
shap_values = explainer(X_test)
```

```
Permutation explainer: 592it [01:32,  6.07it/s]
```

```
shap_values.shape
```

```
(591, 26, 2)
```

Shap
The library was imported and shap was applied on the Neural network model to interprete the features that contributed to model predictions

```
import shap
#X_test.columns = X_train.columns
X_test_df = pd.DataFrame(X_test, columns=X.columns)
#X_train_df = pd.DataFrame(X_train, columns=X.columns)
explainer = shap.Explainer(Nnmodel.predict_proba, X_train)
#explainer = shap.Explainer(Nmodel.predict_proba,X_train_df)
```

```
shap_values = explainer(X_test)
```

```
Permutation explainer: 592it [01:42,  5.50it/s]
```

```
shap_values.shape
```

```
(591, 26, 2)
```

```
classe = 0  # Replace with the desire class (0 or 1)
obser_index = 2  # Replace with the desired rows to explained
max_display = 27  # maximum number of features to display
shap.plots.waterfall(shap_values[obser_index,:, classe], max_display=max_display)
plt.show()
```

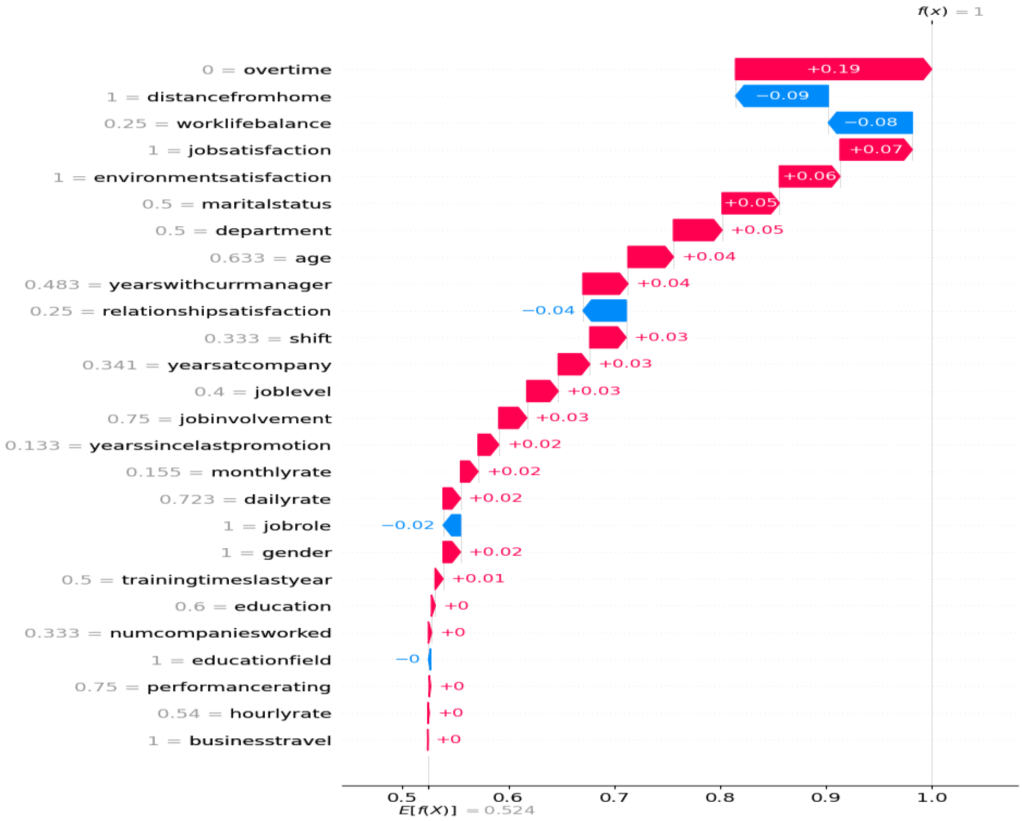


Figure 22: Waterfall on Neural Network Model

```
output_index = 1# getting summary plot for class 1
shap.summary_plot(shap_values[:, :, output_index], X_test)
```

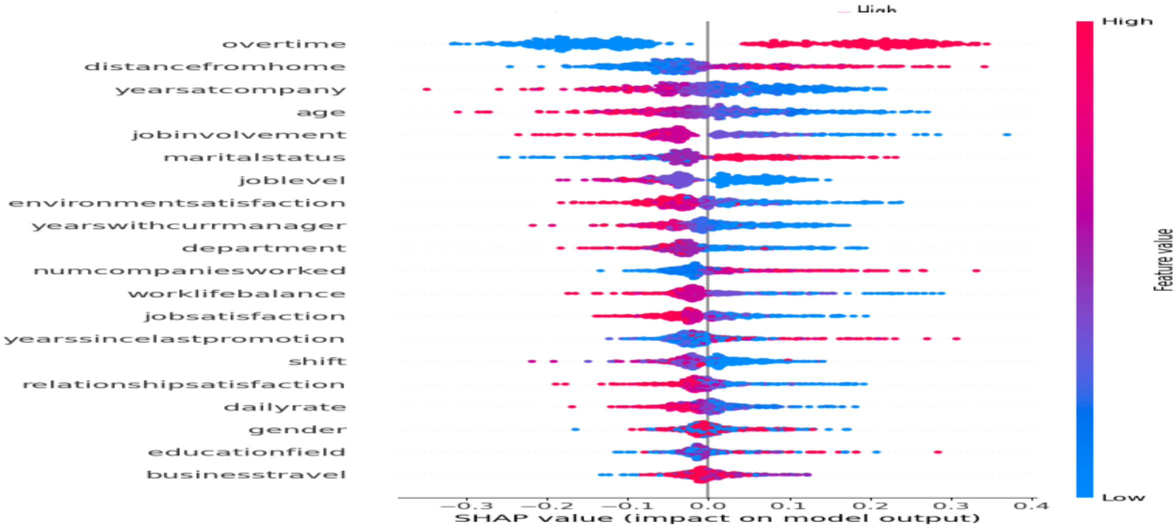


Figure 23: Summary Plot Neural Network Model

References

Guleria, P. and Sood, M., 2023. Explainable AI and machine learning: performance evaluation and explainability of classifiers on educational data mining inspired career counseling. *Education and Information Technologies*, *28*(1), pp.1081-1116.

Huang, Q., Yamada, M., Tian, Y., Singh, D. and Chang, Y., 2022. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*.

Idit Cohen (2021)Explainable AI (XAI) with SHAP -Multi-Class Classification Problem .Available at:URL:https://towardsdatascience.com/explainable-ai-xai-with-shap-multi-class-classification-problem-64dd30f97cea [Accessed July 2023].

Shilpi Bhattacharyya(2020) Explaining Black Box Models: Ensemble and Deep Learning Using LIME and SHAP. Available at: URL:https://www.kdnuggets.com/2020/01/explaining-black-box-models-ensemble-deep-learning-lime-shap.html.

Ma, X., Hou, M., Zhan, J. and Liu, Z., 2023. Interpretable Predictive Modeling of Tight Gas Well Productivity with SHAP and LIME Techniques. *Energies*, *16*(9), p.3653