National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics

Ajay Krishnaa Ayyakutty Ramesh
Student ID: x21193649

School of Computing
National College of Ireland

Supervisor: Dr. Rejwanul Haque

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: …….…… Ajay Krishnaa Ayyakutty Ramesh ……

Student ID: ……x21193649…….……

Programme: …… MSc in Data Analytics…… **Year:** ……2023……..

Module: ……Research Project…….……

Lecturer: ……Dr. Rejwanul Haque…….……

Submission Due Date: ……14-08-2023…….……

Project Title: …… Online fraud detection using Machine learning Models…..……

Word Count: ……380…… **Page Count:** ……7…….……

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: …… Ajay Krishnaa Ayyakutty Ramesh……

Date: ……14-08-2023……

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

Ajay Krishnaa Ayyakutty Ramesh
Student ID: x21193649

1 Introduction

This manual provides the Implementation and configuration details required to setup the project from the scratch. The objective of this document is to provide the detail information for setup the project so that it can be replicate again if required.

2 System configuration

Following configuration is required for running the code:

Processor	AMD Ryzen 4000 series
Operating System	Windows 11
Memory	16 GB RAM
Hard Disk	512 MB SSD

3 Software and Tools

Programming Language	Python3
IDE	Jupyter Notebook
Web Browser	Google Chrome

4 Python Packages

The following python packages has been used in this project.

```
# Import necessary libraries
import pandas as pd
from sklearn.ensemble import VotingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
import numpy as np
from imblearn.over_sampling import SMOTE
from imblearn.combine import SMOTEENN
from imblearn.over_sampling import RandomOverSampler
from sklearn import metrics
from sklearn.cluster import KMeans
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import Dense

# Create the neural network model
```

```
from imblearn.combine import SMOTEENN
from imblearn.under_sampling import EditedNearestNeighbours
from imblearn.pipeline import Pipeline
from collections import Counter
```

5 Data Collection

The dataset was downloaded from kaggle and it is imported to colab file using Google drive

6 Data Loading

The dataset was loaded and converted into pandas dataframe.

```
# Load the dataset
data = pd.read_csv('/content/drive/MyDrive/online_fraud_with_mfa_enabled.csv')

data.dropna()
```

Out[1]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud	sms alert	MFA enabled	isFraudPrevented
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.00	0.00	0	0	1	1	1
1	1	PAYMENT	1864.28	C1666544295	21249.00	19384.72	M2044282225	0.00	0.00	0	0	1	1	1
2	1	TRANSFER	181.00	C1305486145	181.00	0.00	C553264065	0.00	0.00	1	0	1	0	0
3	1	CASH_OUT	181.00	C840083671	181.00	0.00	C38997010	21182.00	0.00	1	0	1	0	0
4	1	PAYMENT	11668.14	C2048537720	41554.00	29885.86	M1230701703	0.00	0.00	0	0	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
176645	95	CASH_OUT	56745.14	C526144262	56745.14	0.00	C79051264	51433.88	108179.02	1	0	1	0	0
176646	95	TRANSFER	33676.59	C732111322	33676.59	0.00	C1140210295	0.00	0.00	1	0	1	0	0
176647	95	CASH_OUT	33676.59	C1000086512	33676.59	0.00	C1759363094	0.00	33676.59	1	0	1	0	0
176648	95	TRANSFER	87999.25	C927181710	87999.25	0.00	C757947873	0.00	0.00	1	0	1	0	0
176649	95	CASH_OUT	87999.25	C409531429	87999.25	0.00	C1827219533	0.00	87999.25	1	0	1	0	0

176650 rows × 14 columns

7 Data Preprocessing

In this step data is pre-processed, null value was removed

```
data.dropna()
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud	sms alert	MFA enabled	isFraudPrevented
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.00	0.00	0	0	1	1	1
1	1	PAYMENT	1864.28	C1666544295	21249.00	19384.72	M2044282225	0.00	0.00	0	0	1	1	1
2	1	TRANSFER	181.00	C1305486145	181.00	0.00	C553264065	0.00	0.00	1	0	1	0	0
3	1	CASH_OUT	181.00	C840083671	181.00	0.00	C38997010	21182.00	0.00	1	0	1	0	0
4	1	PAYMENT	11668.14	C2048537720	41554.00	29885.86	M1230701703	0.00	0.00	0	0	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
176645	95	CASH_OUT	56745.14	C526144262	56745.14	0.00	C79051264	51433.88	108179.02	1	0	1	0	0
176646	95	TRANSFER	33676.59	C732111322	33676.59	0.00	C1140210295	0.00	0.00	1	0	1	0	0
176647	95	CASH_OUT	33676.59	C1000086512	33676.59	0.00	C1759363094	0.00	33676.59	1	0	1	0	0
176648	95	TRANSFER	87999.25	C927181710	87999.25	0.00	C757947873	0.00	0.00	1	0	1	0	0
176649	95	CASH_OUT	87999.25	C409531429	87999.25	0.00	C1827219533	0.00	87999.25	1	0	1	0	0

176650 rows × 14 columns

8 Data Preparation

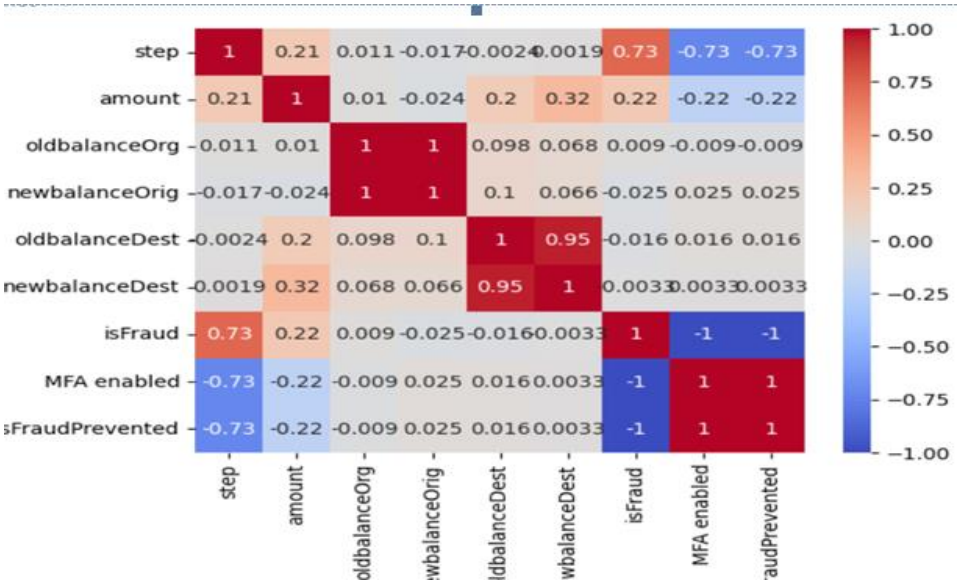
The datataset is converted into train and test set OF 70:30 RATIO.

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.metrics import roc_auc_score, mean_squared_error

X = data[['type','amount', 'oldbalanceOrg','newbalanceOrig','oldbalanceDest','newbalanceDest','sms alert','MFA enabled','isFraudPrevented']]
y = data[['isFraud']]
y = np.squeeze(y)

X = pd.get_dummies(data=X, prefix='type')

X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=0.70, random_state=42)
```



CORRELATION HEATMAP

8.1 Data resampling

```
# Applying SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, y_train.ravel())

# ROS
ros = RandomOverSampler()

# fit and apply the random oversampling
X_resampled_ros, y_resampled_ros = ros.fit_resample(X_train, y_train)
#print(X_resampled_ros)
```

```
from imblearn.combine import SMOTEENN
from imblearn.under_sampling import EditedNearestNeighbours
from imblearn.pipeline import Pipeline
from collections import Counter

smoteEnn = SMOTEENN(random_state=42)
X_resampled_1, y_resampled_1 = smoteEnn.fit_resample(X_train, y_train)

print("SMOTEENN class distribution:", Counter(y_resampled_1))

# Apply smote_enn pipeline
smote = SMOTE()
enn = EditedNearestNeighbours()
smote_enn = Pipeline([('smote', smote), ('enn', enn)])
X_resampled_2, y_resampled_2 = smote_enn.fit_resample(X_train, y_train)
print("smote_enn class distribution:", Counter(y_resampled_2))
print("smote_enn class distribution:", Counter(X_resampled_2))
```

```
SMOTEENN class distribution: Counter({1: 122066, 0: 120121})
smote_enn class distribution: Counter({0: 122840, 1: 122153})
smote_enn class distribution: Counter({'amount': 1, 'oldbalanceOrg': 1, 'newbalanceOrig': 1, 'oldbalanceDest': 1, 'newbalanceDest': 1, 'sms alert': 1, 'MFA enabled'
```

The dataset has 176650 records out of which only 1142 transactions are fraudulent. So to balance the imbalance of the dataset all the imbalance handling methods are need to be used.

9 Model Implementation

Implementing Neural network model

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import Dense

# Create the neural network model
model = Sequential()
model.add(Dense(16, input_dim=X_train.shape[1], activation='relu'))
model.add(Dense(24, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Train the model
model.fit(X_train, y_train, batch_size=32, epochs=10, validation_split=0.2)

# Evaluate the model
scores = model.evaluate(X_valid, y_valid)
print("\nAccuracy: %.2f%%" % (scores[1]*100))
```

Other models that has been applied

```
from sklearn.ensemble import RandomForestClassifier
#X_train, X_test, y_train, y_test = train_test_split(data, data['isFraud'], test_size=0.2, random_state=42)
#X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=0.70, random_state=1)

# Define individual models
model1 = DecisionTreeClassifier()
model2 = KNeighborsClassifier()
model3 = GaussianNB()
model4 = SVC()
model5 = RandomForestClassifier()
model6 = MLPClassifier()
model7 = LogisticRegression(random_state=42)
model8 = GradientBoostingClassifier(random_state=42)
```

9.1 IMPLEMENTATION OF ALL MODELS

```
from sklearn.ensemble import RandomForestClassifier
#X_train, X_test, y_train, y_test = train_test_split(data, data['isFraud'], test_size=0.
#X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=0.70, random_sta

# Define individual models
model1 = DecisionTreeClassifier()
model2 = KNeighborsClassifier()
model3 = GaussianNB()
model4 = SVC()
model5 = RandomForestClassifier()
model6 = MLPClassifier()
model7 = LogisticRegression(random_state=42)
model8 = GradientBoostingClassifier(random_state=42)
```

Selection of models is shown in above figure

Since the models developed are very high in number, one sample model, one ensemble model, one ensemble model with tuned hyperparameters will be covered in this document

```
log_model = model7.fit(X_train, y_train)
log_model_pred = model7.predict(X_valid)
```

```
print(metrics.confusion_matrix(log_model_pred, y_valid))
```

```
[[52527    37]
 [  141   291]]
```

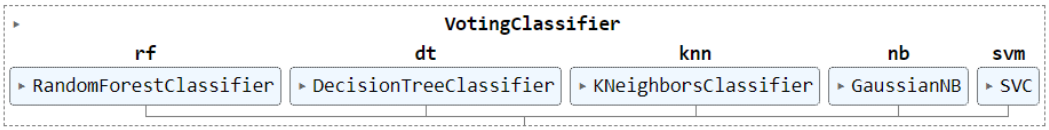
```
print(metrics.f1_score(log_model_pred, y_valid))
print(metrics.accuracy_score(log_model_pred, y_valid))
print(metrics.precision_score(log_model_pred, y_valid))
```

```
0.7657894736842106
0.9966412559438448
0.8871951219512195
```

Training and testing on the Logistic regression model

```
# Define the ensemble model
ensemble = VotingClassifier(estimators=[('rf', model5),('dt', model1), ('knn', model2), ('nb', model3),
                                        ('svm', model4)], voting='hard')

ensemble.fit(X_train, y_train)
```



Ensemble model developed on training dataset

```
from sklearn import metrics
confusion_matrix_ensemble = metrics.confusion_matrix(actual_ensemble, predicted_ensemble)
print(confusion_matrix_ensemble)
Accuracy_ensemble = metrics.accuracy_score(actual_ensemble, predicted_ensemble)
Accuracy_ensemble
```

```
[[52665     3]
 [  108   220]]
0.9979055023020605
```

Prediction on test data for above ensemble model

```
ensemble_new = VotingClassifier(estimators = [("MLP_NEW",model11),
                                              ('DT_NEW', model10),
                                              ('model9',model9)], voting='hard')
print(ensemble_new)
```

```
VotingClassifier(estimators=[('MLP_NEW',
                              MLPClassifier(hidden_layer_sizes=4,
                                            random_state=42)),
                             ('DT_NEW',
                              DecisionTreeClassifier(max_features='auto',
                                                     random_state=42)),
                             ('model9',
                              RandomForestClassifier(max_features='auto',
                                                     random_state=42))])
```

Ensemble_new customized model with tuned parameters

```
ensemble_new.fit(X_resampled, y_resampled)

ensemble_new_pred = ensemble_new.predict(X_valid)

print(metrics.confusion_matrix(y_valid,ensemble_new_pred))
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/tree/_classes.py:269: F
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/ensemble/_forest.py:424
  warn(
[[52668     0]
 [    0   328]]
```

Result on the test data for ensemble_new model

10. **EVALUATION AND RESULTS**

The models developed are evaluated on test datasets and using confusion matrix the perfomance metrics like accuracy, f1-score, precision, recall are derived and then results of models are evaluated and compared for finding the best performing model

11 Conclusion

This report outlines the complete steps for creating the project “Online fraud detection using machine learning algorithms” from scratch. In this python programming is used for the analysis and model building. The Google colab jupyter notebook is used as IDE for writing and executing the code.