Name: Abhijit Sinha

Project: Credit Card Fraud Detection

Email: sinhaabhijit12@yahoo.com

PROJECT OVERWIEW AND OBJECTIVE:

This is a Kaggle Credit Card Fraud Detection: Anonymized credit card transactions labelled as fraudulent or genuine - Credit Card Fraud Detection. The objective of the project is to perform data visualization techniques to understand the insight of the data.
Machine learning often required to getting the understanding of the data and its insights. This project aims apply various Python tools to get a visual understanding of the data and clean it to make it ready to apply machine learning operation on it.

PROJECT EXPLANATION:

In this notebook I will try to predict fraud transactions from a given data set. Given that the data is imbalanced, standard metrics for evaluating classification algorithm (such as accuracy) are invalid. I have focussed on the following metrics: Sensitivity (true positive rate) and Specificity (true negative rate). Of course, they are dependent on each other, so, I want to find optimal trade-off between them. Such trade-off usually depends on the application of the algorithm, and in case of fraud detection I would prefer to see high sensitivity (e.g. given that a transaction is fraud, I want to be able to detect it with high probability).

STEPS INVOLVED IN THE PROJECT:

1) **IMPORTING LIBRARIES**: NUMPY, PANDAS, MATPLOTLIB, SEABORN, WARNINGS etc.
2) **LOADING DATA:** Loads a CSV file into a Pandas DataFrame named `data`.
3) **CHECKING FOR MISSING VALUES:** `data.isnull().sum()` computes the total number of missing values in each column.
4) **CHECKING AND REMOVING DUPLICATES:** 'data.drop_duplicates(inplace = True)' removes the duplicate entries and 'print(data.duplicated().sum())' prints the number of duplicate entries
5) **GETTING DATA INFORMATION:** This displays information about the DataFrame, such as the number of non-null entries, the data types of each column, and memory usage.
6) **COUNTING FRAUD AND NON-FRAUD CASES:** Filters the rows where the `Class` column is `1`, indicating fraud cases. Similarly, it counts the number of rows where `Class` is `0` (non-fraud cases). **Only 0.17% fraudulent transaction out all the transactions. The data is highly Unbalanced.**
7) **SPLITTING DATA INTO FRAUD AND GENUINE:** These lines create two separate DataFrames: `fraud` for fraudulent transactions and `genuine` for non-fraudulent transactions.
8) **VISUALIZING DATA DISTRIBUTION:** Creates histograms for each numeric column in the DataFrame.
9) **VISUALIZING SCATTER PLOT FOR FRAUDULENT AND GENUINE TRANSACTION**

10) **SETTING FIGURE SIZE FOR CALCULATING CORRELATION MATRIX AND CREATING THE HEATMAP USING SEABORN:**

IN THE HEATMAP WE CAN CLEARLY SEE THAT MOST OF THE FEATURES DO NOT CORRELATE TO OTHER FEATURES BUT THERE ARE SOME FEATURES THAT EITHER HAS A POSITIVE OR A NEGATIVE CORRELATION WITH EACH OTHER. FOR EXAMPLE, V2 AND V5 ARE HIGHLY NEGATIVELY CORRELATED WITH THE FEATURE CALLED AMOUNT. WE ALSO SEE SOME CORRELATION WITH V20 AND AMOUNT. THIS GIVES US A DEEPER UNDERSTANDING OF THE DATA AVAILABLE TO US.

11) **PREPARING DATA SET FOR CREATING MACHINE LEARNING MODEL**

a) **IMPORTING REQUIRED LIBRARIES:**
   `train_test_split`, 'RandomForestClassifier`, `accuracy_score`, `numpy`
b) **SPLITTING THE DATA INTO TRAIN AND TEST**
c) **CREATING THE RANDOM FOREST CLASSIFIER**
d) **TRAINING THE MODEL**
e) **MAKING PREDICTIONS**: `model_1.predict (x_Test) generates predictions for the test data (x_Test) using the trained model.
f) **EVALUATING THE MODEL:** `accuracy_score(y_Test, y_Pred)` calculates the accuracy of the model by comparing the true labels (`y_Test`) with the predicted labels (`y_Pred`).

12) **EVALUATING THE PERFORMANCE OF THE RANDOM FOREST CLASSIFIER USING SEVERAL METRICS.**

a) **IMPORTING METRICS:**
   `classification_report` , `accuracy_score`,`precision_score`, `recall_score`
   `f1_score`, `confusion_matrix`
b) **CALCULATING NUMBER OF OUTLIERS AND ERRORS:**
   - `n_outliers` is the total number of fraudulent transactions in the dataset
   - `n_errors` calculates the number of misclassified samples
c) **PRINTING MODEL INFORMATION**
d) **CALCULATING AND PRINTING METRICS:** `accuracy_score', `precision_score', `recall_score', `f1_score'

13) **GENERATING AND DISPLAYING A CONFUSION MATRIX AS A HEATMAP:**

a) **Defining Labels**
b) **Calculating the Confusion Matrix**
c) **Creating and Displaying the Heatmap**

14) **APPLING LOGISTIC REGRESSION TO A DATASET:**

a) **Preparing the Data**
b) **Splitting the Data**
c) **K-Fold Cross Validation and Grid Search was applied**
d) **Predictions were done on Training and Test set**
e) **ROC curve for both Training and Test set was plotted**
f) **Creating and Training the Logistic Regression Model**

15) **EVALUATING THE PERFORMANCE OF A LOGISTIC REGRESSION MODEL WITH HANDLING CLASS IMBALANCE BY SMOTE:**

a) **IMPORTING EVALUATION METRICS:**
`accuracy_score`, `precision_score`, `recall_score`, `f1_score`,`confusion_matrix`

b) **CALCULATING NUMBER OF OUTLIERS AND ERRORS:**
`n_outliers` represents the number of fraudulent transactions
`n_errors` counts the number of misclassified predictions

c) **PRINTING MODEL INFORMATION**

d) **CALCULATING AND PRINTING METRICS:**
`accuracy_score', `precision_score',`recall_score',`f1_score,

16) **GENERATING AND VISUALIZING THE CONFUSION MATRIX FOR THE LOGISTIC REGRESSION MODEL:**

a) **DEFINING LABELS**

b) **CALCULATING THE CONFUSION MATRIX:**
- True Positives (TP): Fraud predicted as Fraud
- True Negatives (TN): Normal predicted as Normal
- False Positives (FP): Normal predicted as Fraud
- False Negatives (FN): Fraud predicted as Normal

c) **CREATING AND DISPLAYING THE HEATMAP**

17) **DEMONSTRATION TO USE A DECISION TREE CLASSIFIER MODEL AND EVALUATING THE DATA:**

a**) Preparing the Data;** b**) Initializing the Decision Tree Classifier**
c**) Splitting the Data;** d**) Training the Model**
e**) Making Predictions;** f) **Evaluating Accuracy**

18) EVALUATING THE PERFORMANCE OF `DECISION TREE CLASSIFIER` MODEL,

a) **IMPORTING EVALUATION METRICS:** `accuracy_score`,`precision_score`, `recall_score`,`f1_score`

b) **CALCULATING NUMBER OF OUTLIERS AND ERRORS:**
`n_outliers` gives the number of fraudulent transactions from the dataset.
`n_errors` counts the number of misclassified instances

c) **CALCULATING AND PRINTING METRICS:**
`accuracy_score', `precision_score', `recall_score', `f1_score'

19) VISUALIZING THE CONFUSION MATRIX FOR THE MODEL'S PREDICTIONS:

a) **DEFINING LABELS**

b) **CALCULATING THE CONFUSION MATRIX:**
- True Positives (TP): Fraud predicted as Fraud
- True Negatives (TN): Normal predicted as Normal
- False Positives (FP): Normal predicted as Fraud
- False Negatives (FN): Fraud predicted as Normal

c) **CREATING AND DISPLAYING THE HEATMAP**