CREDIT CARD FRAUD DETECTION

ABOUT THE STUDY

The aim of the study is to create a model that can flag Fraudulent transaction

- DATA FROM KAGGLE: Credit Card Fraud Detection | Kaggle
- EDA IN PYTHON
- MODEL DEVELOPMENT IN PYTHON
- PACKAGES USED: numpy , pandas, matplotlib.pyplot ,seaborn,sklearn.preprocessing,sklearn.model_selection,statsmodels.api,sklearn.tree,statsmodels.stats.outliers_influence, sklearn.ensemble,sklearn.linear_model,sklearn.svm,sklearn.neighbors,sklearn.tree,sklearn.ensemble,collections, sklearn.metrics,sklearn.pipeline,imblearn.pipeline,imblearn.over_sampling,imblearn

THE DATA STRUCTURE

Before correction

- The data was heavily imbalanced: 492 Frauds among 284807 datapoints
- Had around thirty-one variables: thirty of them PCA and rest normal.
- Skewed distributions
- No Null Values or any prominent anomalies.
- Y= 'Class' 1=FRAUD,0=NON-FRAUD ,X= rest of the variables

After correction

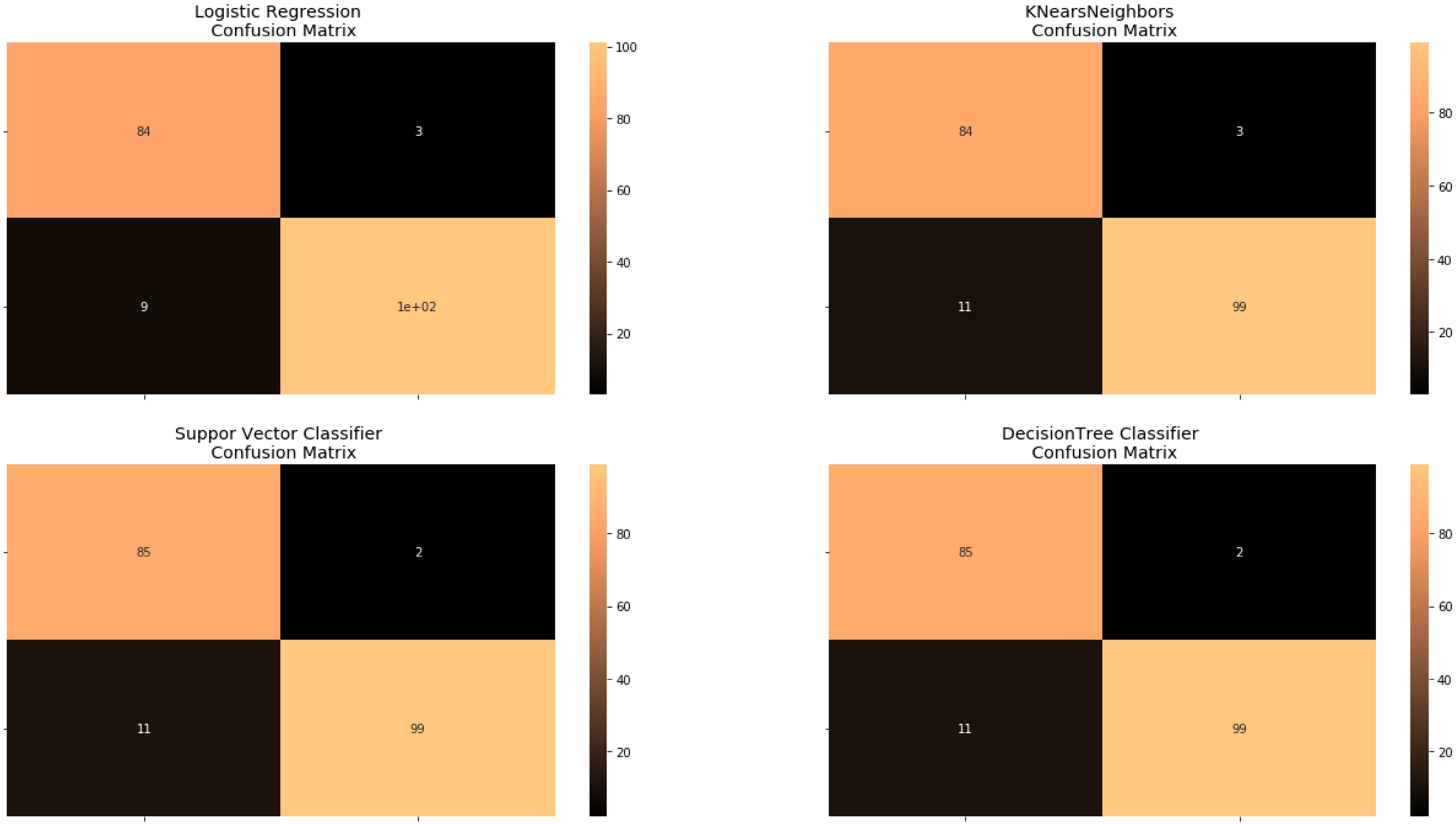
- Created a subset of data with 492 Frauds and 492 Non-Frauds.
- Scale Time and Amount to make it normally distributed.
- Use Stratified Kfold to split the original data to train and test
- Created a correlation heatmap on the balanced dataset.
- Run VIF and decision Forrest to identify the most important predictors.

INSIGHTS FROM BALANCED DATA

- By performing EDA of subset-data i.e. New_df, the relation between the variables became more legible.
- The dimension was reduced in different iterations using VIF, Decision Tree and Random Forrest Classification.
- Based on the dimension reduction techniques, the variables 'Class','V4','V10','V11','V12','V14','V16','V17','V19','V20','V27','scaled_amount','scaled_time' are selected for the formulating the model

Model on Balanced Data

Classifier	CROSS VALIDATION SCORE	TESTING SCORE
LogisticRegression	94.0 %	93.90%
KNeighborsClassifier	93.0 %	92.89%
Support Vector Classifier	93.0 %	93.4%
DecisionTreeClassifier	90.0 %	88.83%


Logistic Regression
Confusion Matrix
84
3
9
1e+02
100
80
60
40
20
KNearsNeighbors
Confusion Matrix
84
3
11
99
80
60
40
20
Suppor Vector Classifier
Confusion Matrix
85
2
11
99
80
60
40
20
DecisionTree Classifier
Confusion Matrix
85
2
11
99
80
60
40
20

```
Logistic Regression:
              precision    recall  f1-score   support

           0       0.90      0.97      0.93        87
           1       0.97      0.92      0.94       110

    accuracy                           0.94       197
   macro avg       0.94      0.94      0.94       197
weighted avg       0.94      0.94      0.94       197

KNears Neighbors:
              precision    recall  f1-score   support

           0       0.88      0.97      0.92        87
           1       0.97      0.90      0.93       110

    accuracy                           0.93       197
   macro avg       0.93      0.93      0.93       197
weighted avg       0.93      0.93      0.93       197

Support Vector Classifier:
              precision    recall  f1-score   support

           0       0.89      0.98      0.93        87
           1       0.98      0.90      0.94       110

    accuracy                           0.93       197
   macro avg       0.93      0.94      0.93       197
weighted avg       0.94      0.93      0.93       197

Decision tree classifier:
              precision    recall  f1-score   support

           0       0.89      0.98      0.93        87
           1       0.98      0.90      0.94       110

    accuracy                           0.93       197
   macro avg       0.93      0.94      0.93       197
weighted avg       0.94      0.93      0.93       197
```

Model on Balanced Data

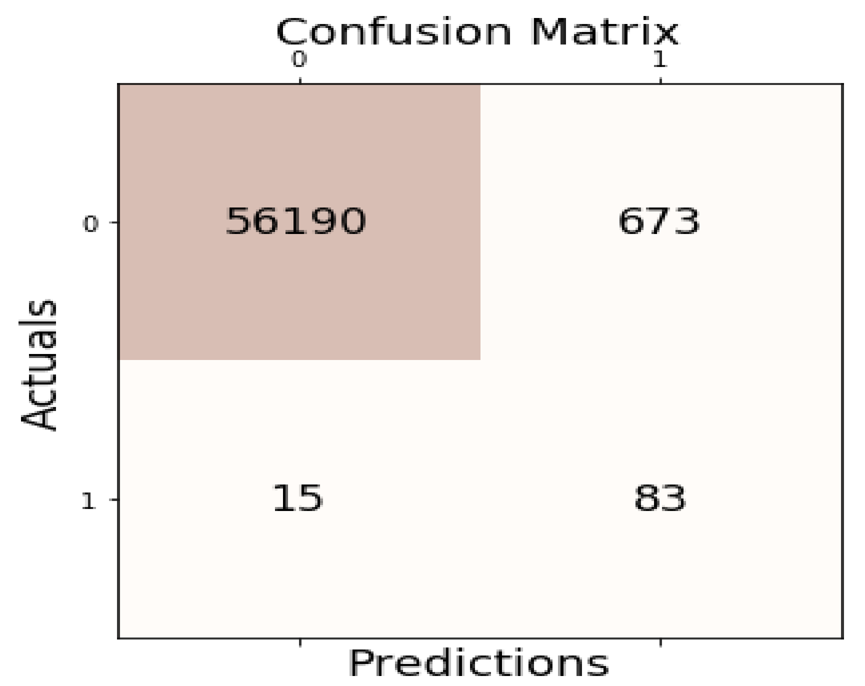
- We decided to go with Logistic regression.
- For fitting the model on Original data SMOTE Technique was adopted.
- SMOTE creates synthetic points from the minority class in order to reach an equal balance between the minority and majority class. Location of the synthetic points: SMOTE picks the distance between the closest neighbors of the minority class, in between these distances it creates synthetic points. Final Effect: More information is retained since we didn't have to delete any rows unlike in random under sampling

IMPLEMENTING SMOTE IN LOGISTIC REGRESSION

```
#implementing SMOTE in Logistic regression

from imblearn.over_sampling import SMOTE
accuracy_lst = []
precision_lst = []
recall_lst = []
f1_lst = []
auc_lst = []

for train, test in stkf.split(original_Xtrain, original_Ytrain):
    pipeline = imbalanced_make_pipeline(SMOTE(sampling_strategy='minority'),Log_regg) # SMOTE happens during Cross Validat
e..
    model = pipeline.fit(original_Xtrain[train], original_Ytrain[train])
    prediction = Log_regg.predict(original_Xtrain[test])
    accuracy_lst.append(pipeline.score(original_Xtrain[test], original_Ytrain[test]))
    precision_lst.append(precision_score(original_Ytrain[test], prediction))
    recall_lst.append(recall_score(original_Ytrain[test], prediction))
    f1_lst.append(f1_score(original_Ytrain[test], prediction))
    auc_lst.append(roc_auc_score(original_Ytrain[test], prediction))
```




```
---------------------------------

accuracy: 0.9425100695767785
precision: 0.061068551743605
recall: 0.9162934112301201
f1: 0.11271073613773068
```

REPORT

```
labels = ['No Fraud', 'Fraud']
smote_prediction = Log_regg.predict(original_Xtest)
print(classification_report(original_Ytest, smote_prediction, target_names=labels))
```

	precision	recall	f1-score	support
No Fraud	1.00	0.99	0.99	56863
Fraud	0.11	0.85	0.19	98
accuracy			0.99	56961
macro avg	0.55	0.92	0.59	56961
weighted avg	1.00	0.99	0.99	56961

Model is too picky: it flags some genuine transaction if they exhibit some anomaly

What did we learn?

- learned how to fix an Imbalance data
- learned about SMOTE.
- learned pick out important predictor variables.

THANK YOU!