Predict Credit Default

By:

Varsha Wainganka r

Overview

- Data source
- Data cleaning and preprocessing
- Code
- Scaling normalizing data
- Handling imbalanced data
- Feature engineering
- Predictive modeling
- Accuracy and best model

Data source and Problem Statement

- **Data source UCI Machine Learning Repository**

https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

- This research aimed at the case of customers default payments in Taiwan and compares the predictive accuracy of probability of default using various methods

Data Frame

```python
#Reading the data using pandas

df = pd.read_excel("default of credit card clients.xls")

#default of credit card clients.xls
```

```python
df.head()
```

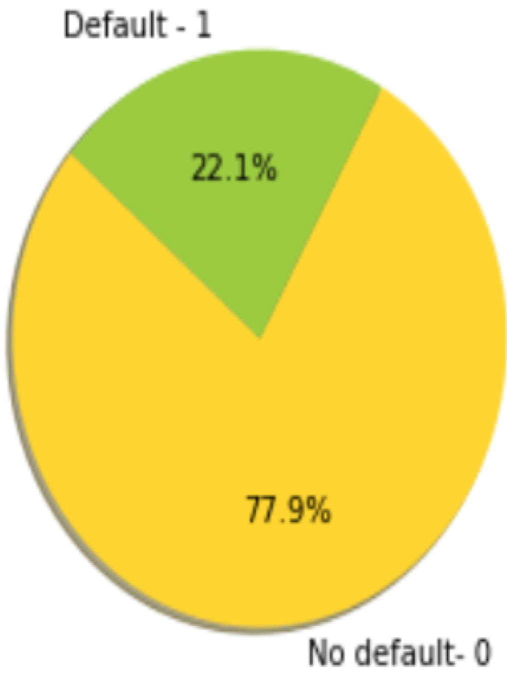
AGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default payment next month
1	24	2	2	-1	-1	...	0	0	0	0	689	0	0	0	0	1
2	26	-1	2	0	0	...	3272	3455	3261	0	1000	1000	1000	0	2000	1
2	34	0	0	0	0	...	14331	14948	15549	1518	1500	1000	1000	1000	5000	0
1	37	0	0	0	0	...	28314	28959	29547	2000	2019	1200	1100	1069	1000	0
1	57	-1	0	-1	0	...	20940	19146	19131	2000	36681	10000	9000	689	679	0

```python
#Replacing the column name for convenience

df.rename(columns={"default payment next month": "default"}, inplace = True)
```

```python
#checking the columns

df.columns
```

```
Index(['ID', 'LIMIT_BAL', 'SEX', 'EDUCATION', 'MARRIAGE', 'AGE', 'PAY_0',
       'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6', 'BILL_AMT1', 'BILL_AMT2',
       'BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1',
       'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6', 'default'],
      dtype='object')
```

Distribution of data (2 categories)



Data is highly imbalanced

78 percent is credit card amount payed duly

22 percent is credit card default

Data Manipulation

Data Manipulation:

Reduced unknown values to category 4 (Education)

```
df['EDUCATION'].unique()
```

```
array([2, 1, 3, 5, 4, 6, 0])
```

```
#Change values for education  (1 = graduate school; 2 = university; 3 = high school; 4 = others)
#Anything other than 4 will be changed to 4

fil = (df['EDUCATION'] == 5) | (df['EDUCATION'] == 6) | (df['EDUCATION']== 0)
df.loc[fil, 'EDUCATION'] = 4
df['EDUCATION'].value_counts()
```

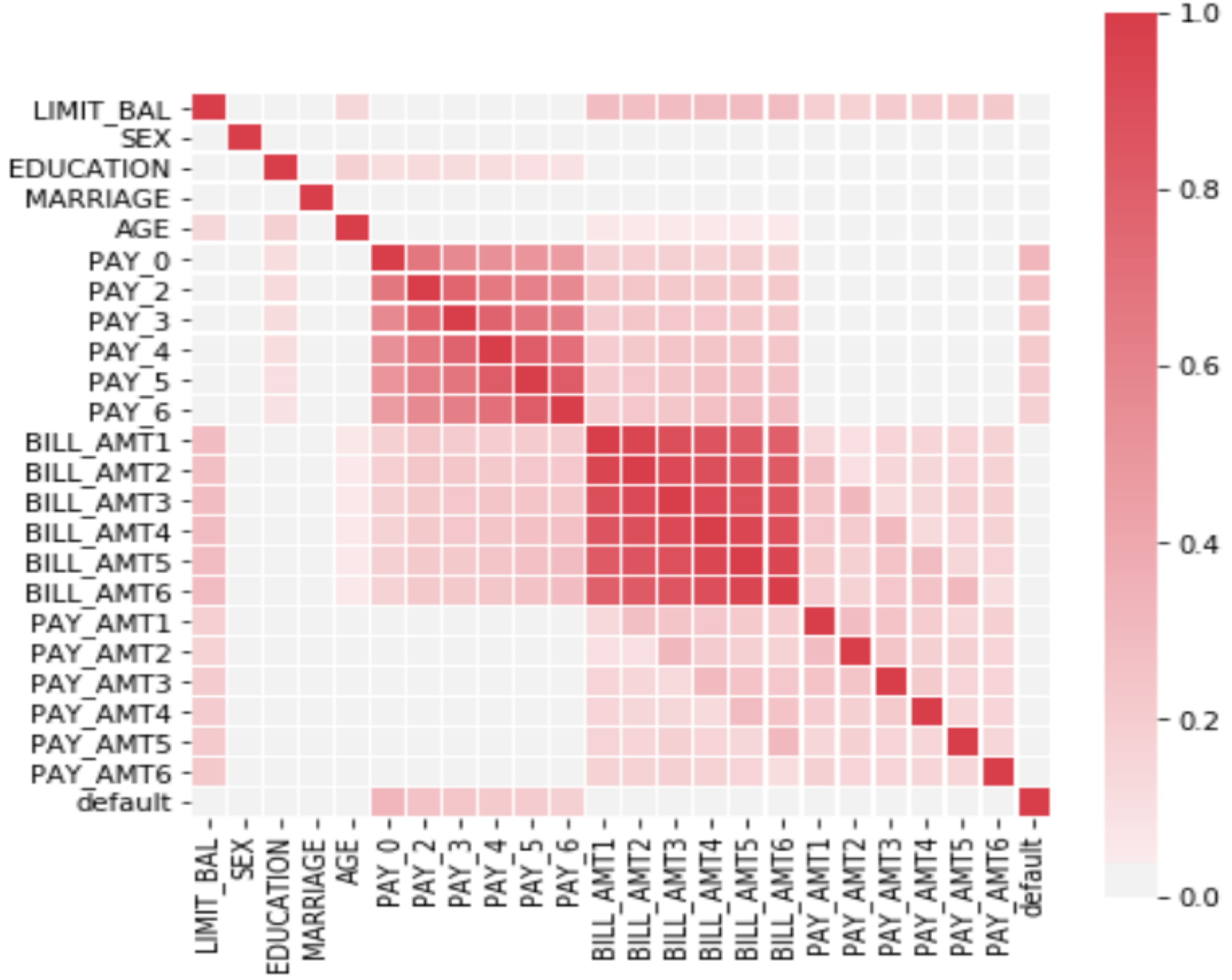
```
2    14030
1    10585
3     4917
4      468
Name: EDUCATION, dtype: int64
```

```
df['MARRIAGE'].unique()
```

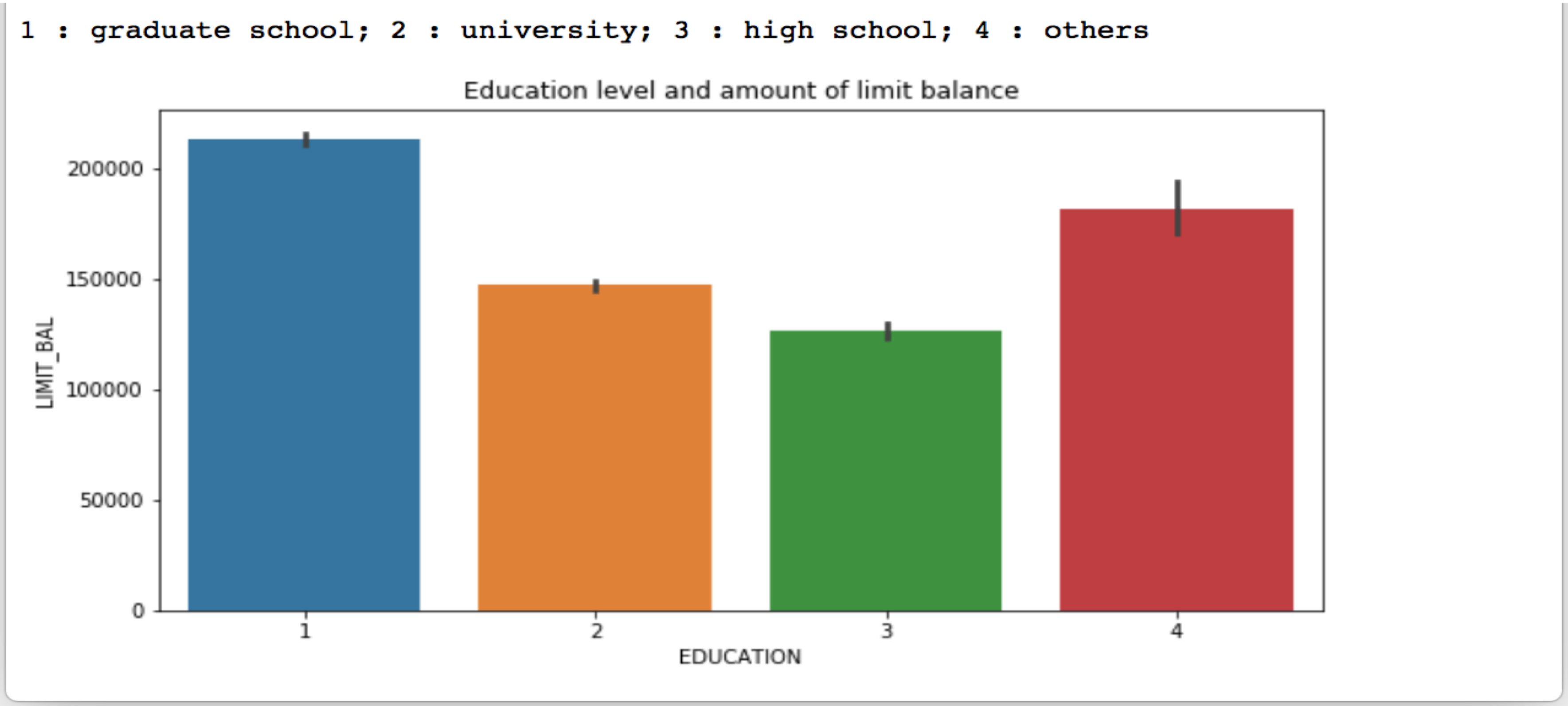
```
array([1, 2, 3, 0])
```

Similar manipulation was performed on various other fields, that had different values other than the ones defined.

Correlation coefficient of variables



Distribution of Education field



Applying Minmax Scaler

```
minmax_scale = preprocessing.MinMaxScaler().fit(df)
df_minmax = minmax_scale.transform(df)
df_minmax = pd.DataFrame(df_minmax, columns= list(df))
df_minmax.hist(figsize=(20,20))
plt.show()
```

Models :(sklearn)

- Logistic Regression

Most widely used for Binary classification problem. The sigmoid   function snaps values to 0 and 1, and we predict a class value.

- K Nearest Neighbors

For a data point to be classified into two different categories, We find the k nearest neighbors (k is any odd value) Then we use majority voting on the labels. The majority class label is assigned to the data point If k is even then distance is calculated. The shorted distance is used

- Decision Tree Classifier

DTC will segregate the data points based on all values of variables and identify the variable, which creates the best homogeneous sets of data points (which are heterogeneous to each other)

- Random Forest Classifier

Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance

Predictive Modeling on Imbalanced Data

```
from sklearn import linear_model
logreg = linear_model.LogisticRegression(C=1e5)
logreg.fit(X_train, y_train)
prediction = logreg.predict(X_test)
print("accuaracy of model")
a= accuracy_score(y_test, prediction)
a=a*100
print(a)
```

```
accuaracy of model
81.01666666666667
```

Conclusion of running model on imbalanced data:

Since distribution is 78:22 ratio, so running a model yeilds an 80 percent accuarcy. So it makes no sense to run a model on imbalanced data. Even random guess will give this result.

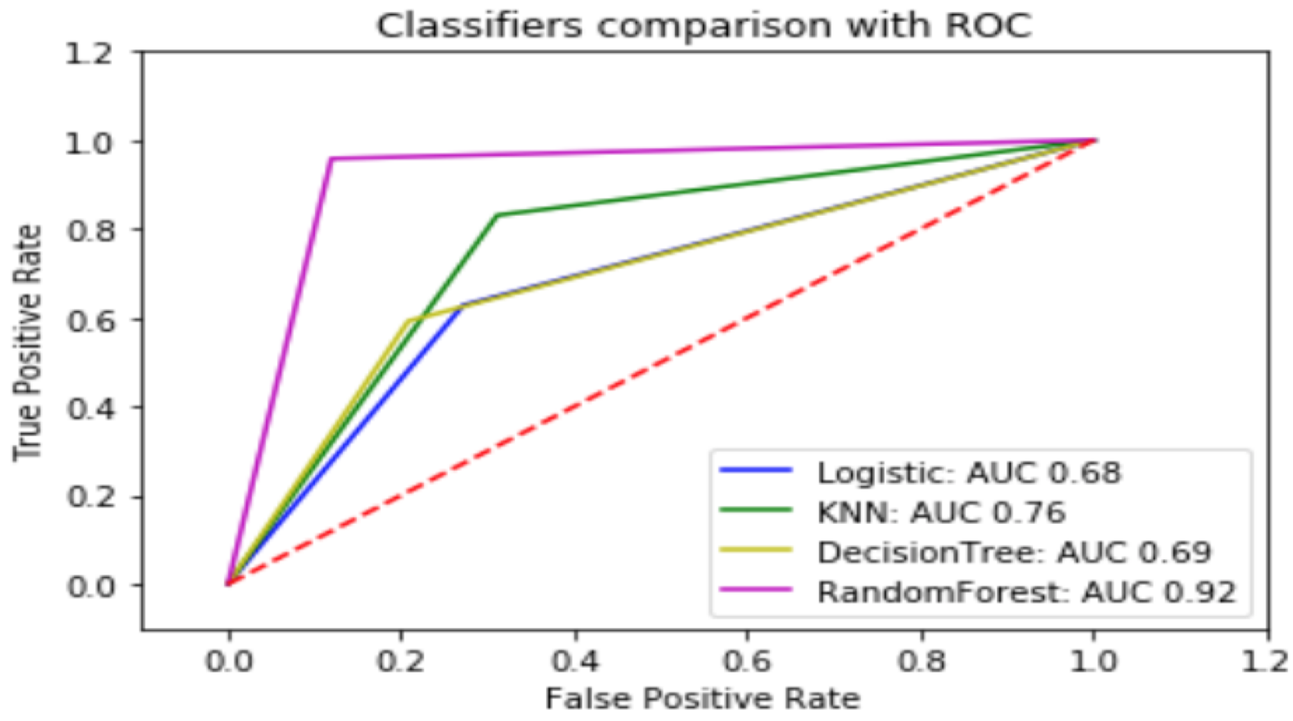
No other model was tried, cause running model on imbalanced data doesn't serve the purpose.

Running model on Randomly Oversampled Data

```
from sklearn.metrics import accuracy_score

cmp = 0
for model, predicted in prediction.items():
    accuracy = accuracy_score(y_test, predicted)
    accuracy
    print(model, accuracy*100)
    cmp += 1
```

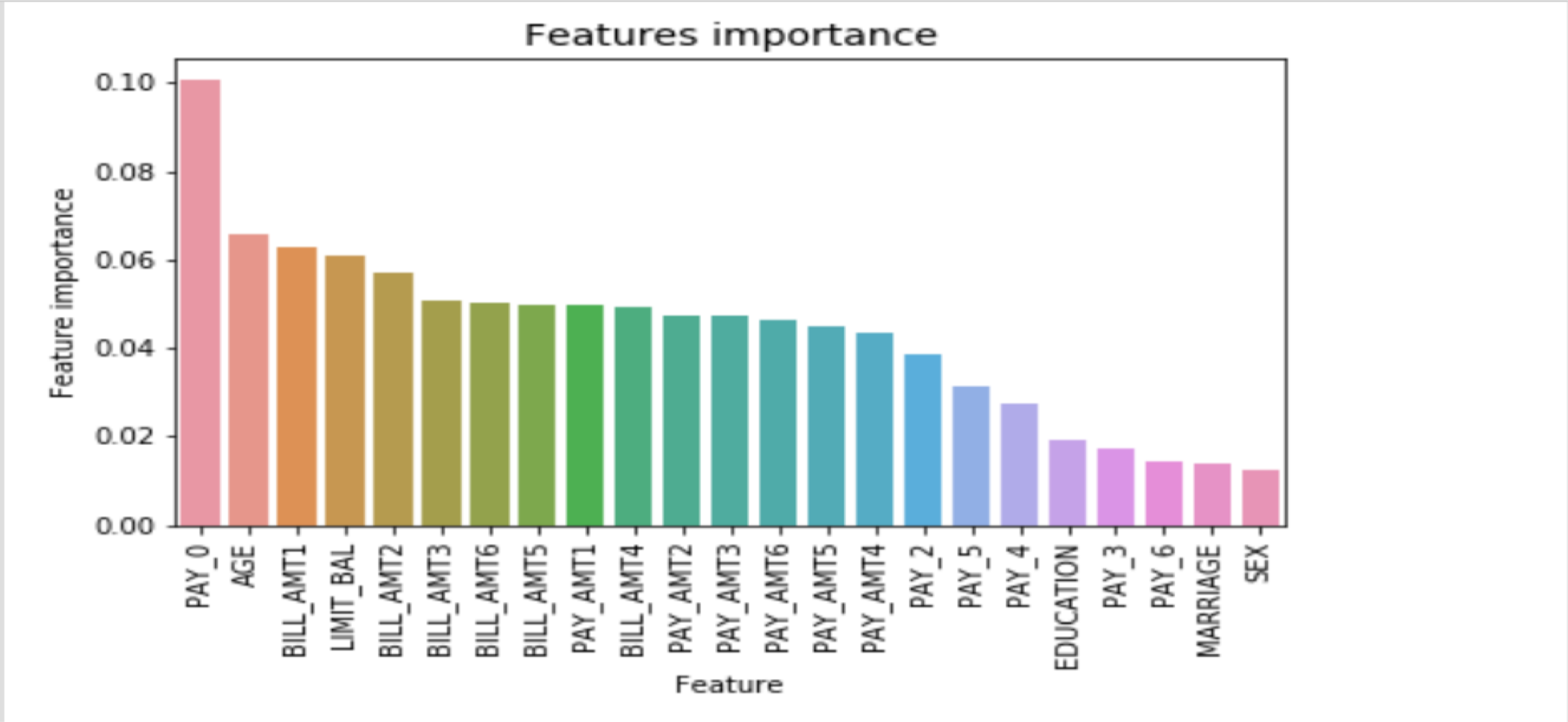
```
Logistic 67.86838961885113
KNN 75.93658377674014
DecisionTree 69.26919318058421
RandomForest 91.91008795743295
```

Receiver Operator Characteristic (Area under curve)



Feature Selection

- We can select the best features based on the correlation coefficient of predictors with target . (forward ,backward, automated(random forest)



SMOTE

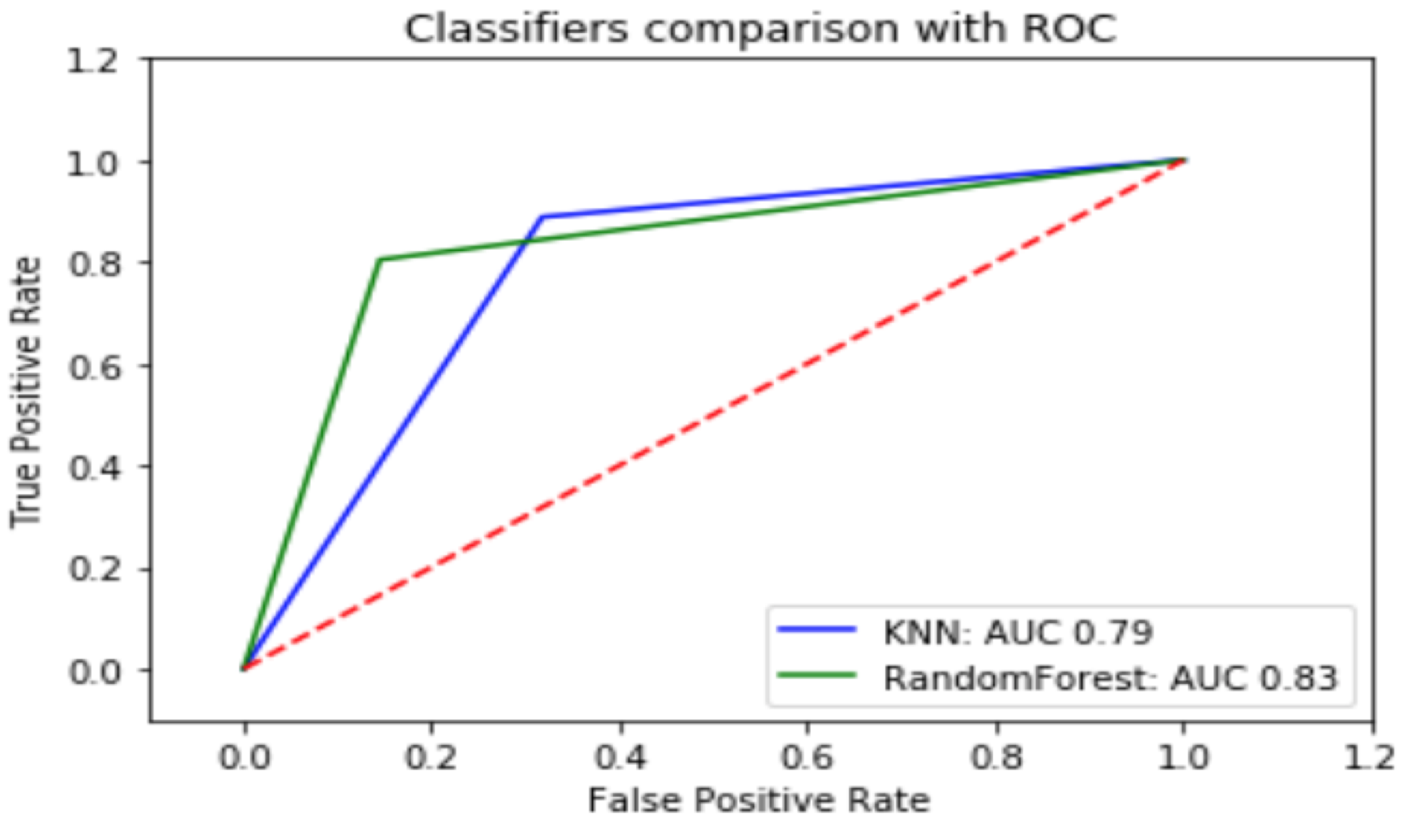
- **Using SMOTE - Synthetic Minority Oversampling Technique**
- over-sampling approach in which the minority class is over-sampled by creating synthetic" examples rather than by over-sampling with replacement.
- Reduces the chance of overfitting

Accuracy

KNN 78.45067408517012

Random Forest -83.59726086026107

Receiver Operator Characteristic (Area under curve)



Learning Processes

- Understanding various new concepts
- Using MinMax Scaler to Normalize data.
- Understanding the effect of unbalanced data
- Random oversampling of minority class, under sampling of majority class, SMOTE.
- Using Sklearn library for running various models.
- Using feature engineering.
- Understanding confusion matrix , accuracy and Reciever Operator characteristic concepts.

Conclusion

- The most important parameters in determining default of credit cards are the repayment status variable.
- With Random oversampling of data and Random Forest classifier, achieves the best accuracy of 91 percent , with precision – recall score of 0.87 and area under curve of 0.92
- With , SMOTE Random Forest Classifier , achieves the best accuracy of 82 percent , with precision – recall score of 0.79 and area under curve of 0.83
- KNN is the next best model.
- Random Under sampling didn’t yield great results because of less data points.