# **HW 2 (584-Rangwala): Credit Risk Prediction**

# Pankaj Kumar Jatav – G10338769

**Goal:**

Develop predictive models that can determine someone’s credit risk 0 - high risk, 1-low risk.

**Approach:**

I follow the below diagram step to solve this assignment.



**Preprocessing of Data:**

Below are the steps taken to preprocess that data.

1. Perform feature selection based on different classification models
2. Remove low co-related features which I got from 1st step.

model = SelectFromModel(modelObj.model, prefit=True, threshold=threshold)  
X\_new = model.transform(X\_train)  
modelObj = ModelClass(X\_new, y\_train)

1. Remove duplicate data from test data

df = df.drop\_duplicates(df.columns[1:])

1. Oversampling for test data

oversample = RandomOverSampler(sampling\_strategy='minority')  
X\_over, y\_over = oversample.fit\_resample(X, y)

**Model Training:**

Below are the steps taken to create model from training data.

1. Perform the models test with and without features selection.
2. Train the models without oversampling data

X, y = shuffle(X, y)  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.3, random\_state = 0)  
  
print("Predict using feature selection")  
predict\_with\_fa(X\_train, X\_test, y\_train, y\_test)  
  
print("Predict using without feature selection")  
predict\_without\_fa(X\_train, X\_test, y\_train, y\_test)

1. Train the models with oversampling data

oversample = RandomOverSampler(sampling\_strategy='minority')  
X\_over, y\_over = oversample.fit\_resample(X, y)  
perform\_cross\_validation(X\_over, y\_over)

**Models:**

I have used the below classifiers to find the best for this assignment.

1. k-nearest Neighbors
2. Support Vector Machine
3. Random Forest
4. Logistic Regression
5. Decision Tree
6. Naive Bayes
7. Easy Ensemble

**Cross Validation:**

To Cross validate the model accuracy. I suffer the data split the document in 70-30 ratio. Testing was performed on each model and with/without feature selection.

Below are the f1 score evaluation metric for each models with combination of with/without oversampling and with/without feature selection.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classification | Cross Validation Without Oversampling | | Cross Validation With Oversampling | |
| feature selection | without feature selection | feature selection | without feature selection |
|  |  |  |  |  |
| k-nearest Neighbors | 66.053 | 65.611 | 84.224 | 83.705 |
| Support Vector Machine | 67.033 | 66.231 | 85.441 | 84.844 |
| Random Forest | 62.217 | 60.561 | 88.562 | 89.581 |
| Logistic Regression | 53.087 | 53.087 | 73.441 | 73.44 |
| Decision Tree | 58.926 | 56.882 | 86.146 | 87.301 |
| Naive Bayes | 54.313 | 55.129 | 56.916 | 59.975 |
| Easy Ensemble | 71.164 | 70.321 | 81.9 | 81.377 |

Below function used to visualize the prediction data.

ecc\_predict = predict\_model('eec', X\_train, y\_train, X\_test)  
print("EEC Accuracy")  
print(f1\_score(ecc\_predict, y\_test))

Test Validation:

Used EasyEnsembleClassifier for train and test for test data, Because it gave the best result on test data. Upload result on miner and got 67 percent accuracy.

**Step To Run the program:**

Run main.py file

>>python3 main.py

It will give 5 options as bellow.

Please select a options

1. Cross Validation without oversampling

2. Cross Validation with oversampling

3. Perform data prediction on test set

4. Above all in sequence

5. Exit

Enter your choice: 5

Cross Validation

Predict using feature selection

KNN Accuracy

0.6677128285602196

**Plots:**

**DT Importance Feature**

**Chart, bar chart

Description automatically generated**

**EEC Importance Feature**

**Chart, bar chart

Description automatically generated**

**Logistic regression Importance Feature**

**Chart, bar chart

Description automatically generated**

**NBayes Importance Feature**

**Chart

Description automatically generated**

**RF Importance Feature**

**Chart, bar chart

Description automatically generated**

**KNN Importance Feature**

**Chart

Description automatically generated**