HW2: Credit Risk Prediction

Details:

Name: Aniket Pandey
Miner2 Username: 414
GMU Id: apandey7
Rank: 63
Public Score: 0.66

Steps to run the code:

1. Download CreditRiskPrediction.ipynb file
2. Open JupyterLab on Anaconda Navigator or Google Colab Notebook
3. On the menu bar click File -> Open -> CreditRiskPrediction.ipynb
4. Keep the credit_test.dat and credit_train.dat file in the same directory
5. One by one run each cell in the notebook

Introduction:

To predict the Credit Risk, we tried training multiple models and then the best of them was chosen to predict the risk for the test data. The algorithms used in this assignments are: Support Vector Machine, K Nearest Neighbor, Random Forest, Decision Tree, and Naive Bayes.

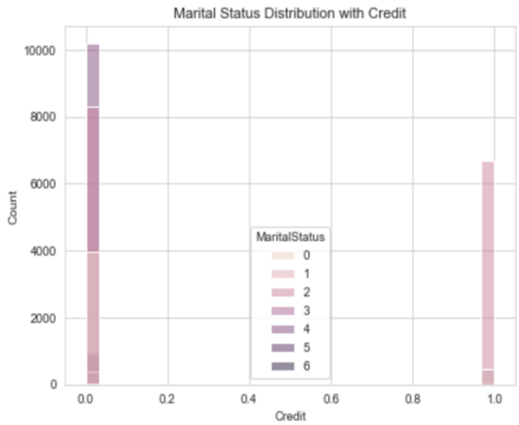
Solution:

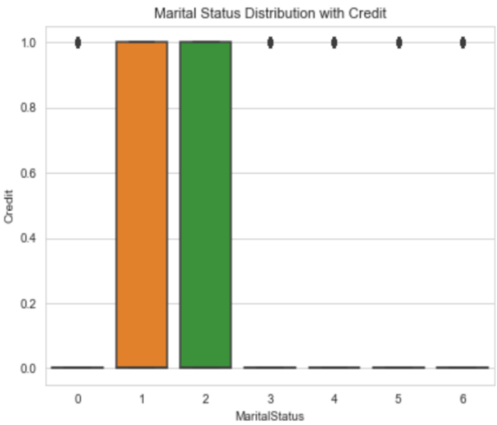
After training the algorithms with the training data and by cross validation we found out that Random Forest is giving the best F1 Score of – and therefore we choose Random Forest for predicting the risk of the test data.

Process/Approach:

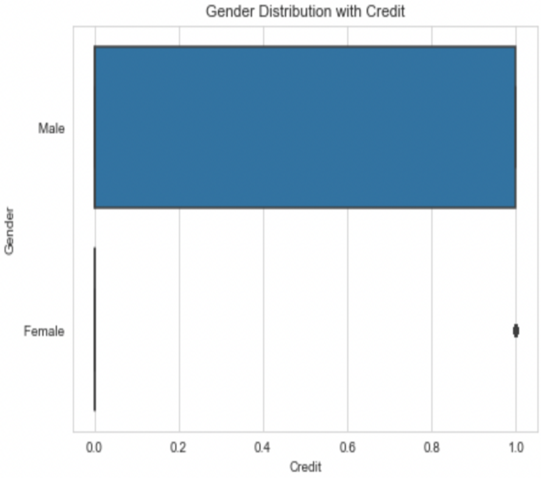
1. Import necessary packages and libraries. Following libraries are used in the classifier code:
   - Numpy for working mathematical operations on arrays
   - Seaborn for Pearson’s Correlations
   - Pandas for Dataframe
   - RandomUnderSampler for Sampling
   - Sklearn for various algorithms
   - Matplotlib for plotting graphs
2. Read the training and test data using Pandas.

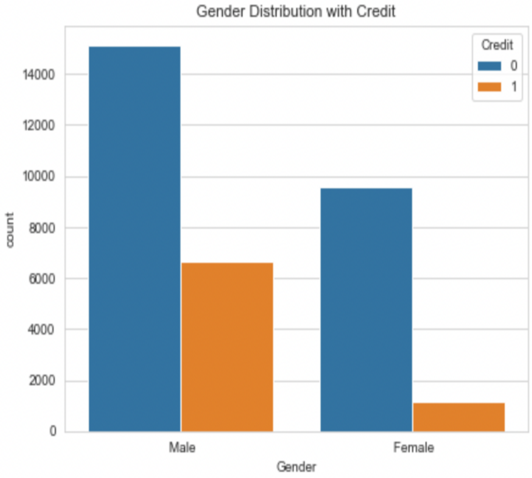
3. Replace column name with actual heading for easy visualization
4. Check the categorical and continuous data in the train file
5. Credit distribution with respect to Marital Status.



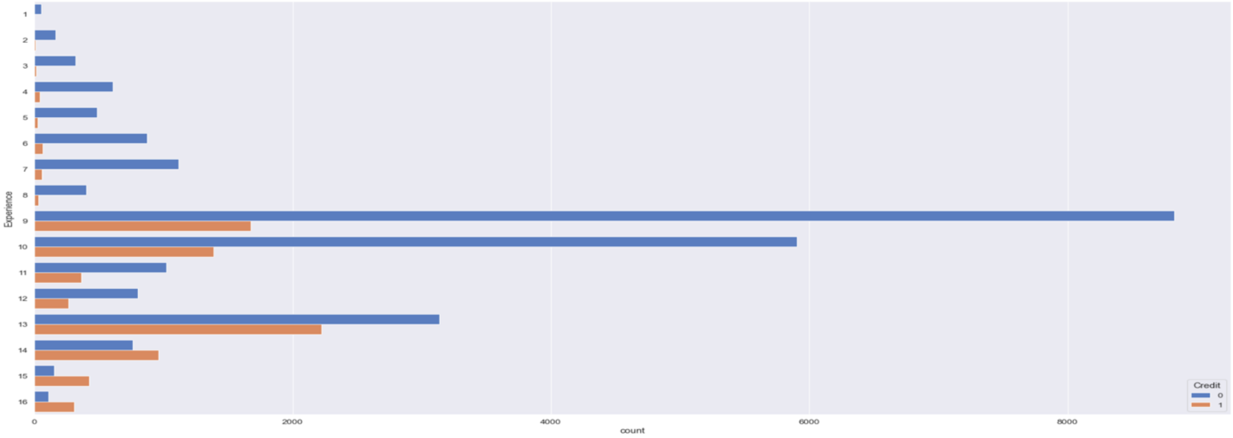


6. Credit distribution with respect to Gender.



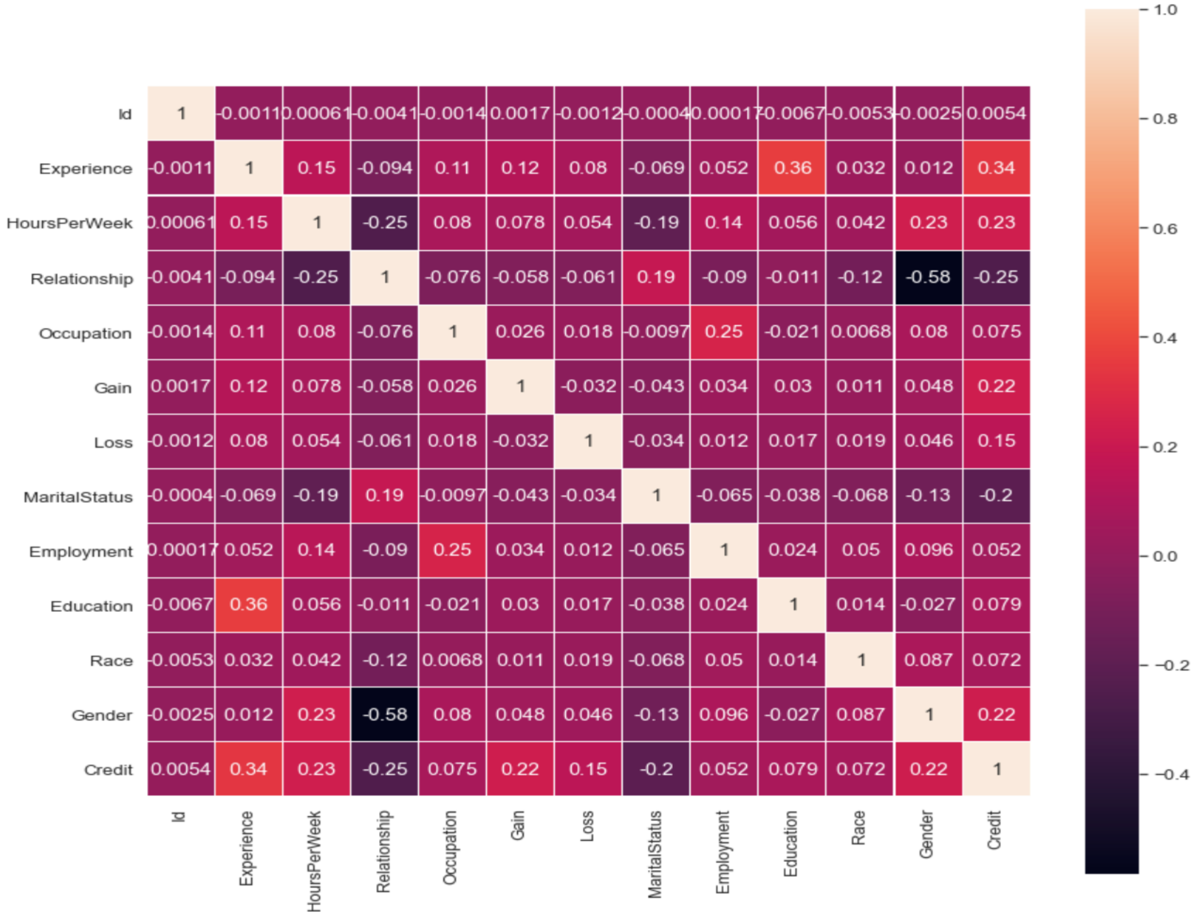


7. Credit distribution with respect to experience



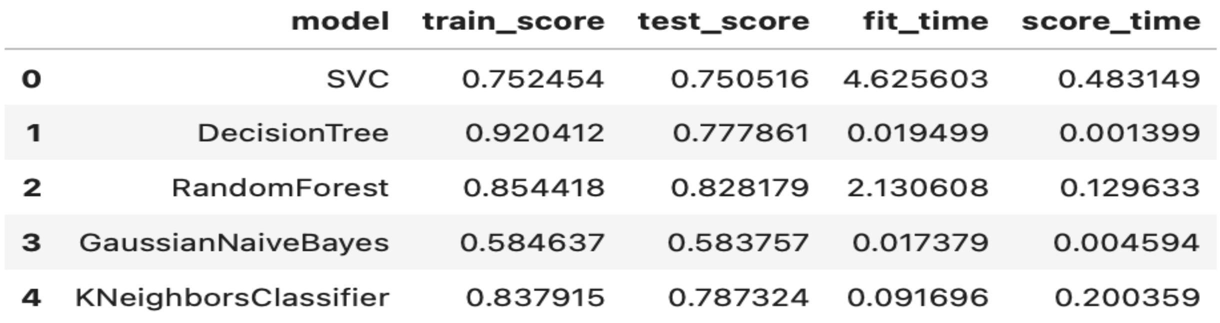
8. Using LabelEncoder transform Race and Gender column.

9. Find the Pearson’s correlations between Credit and other columns to determine which columns are best suited for training the model.



	Id	Experience	HoursPerWeek	Relationship	Occupation	Gain	Loss	MaritalStatus	Employment	Education	Race	Gender	Credit
Id	1	-0.0011	0.00061	-0.0041	-0.0014	0.0017	-0.0012	-0.0004	0.00017	-0.0067	-0.0053	-0.0025	0.0054
Experience	-0.0011	1	0.15	-0.094	0.11	0.12	0.08	-0.069	0.052	0.36	0.032	0.012	0.34
HoursPerWeek	0.00061	0.15	1	-0.25	0.08	0.078	0.054	-0.19	0.14	0.056	0.042	0.23	0.23
Relationship	-0.0041	-0.094	-0.25	1	-0.076	-0.058	-0.061	0.19	-0.09	-0.011	-0.12	-0.58	-0.25
Occupation	-0.0014	0.11	0.08	-0.076	1	0.026	0.018	-0.0097	0.25	-0.021	0.0068	0.08	0.075
Gain	0.0017	0.12	0.078	-0.058	0.026	1	-0.032	-0.043	0.034	0.03	0.011	0.048	0.22
Loss	-0.0012	0.08	0.054	-0.061	0.018	-0.032	1	-0.034	0.012	0.017	0.019	0.046	0.15
MaritalStatus	-0.0004	-0.069	-0.19	0.19	-0.0097	-0.043	-0.034	1	-0.065	-0.038	-0.068	-0.13	-0.2
Employment	0.00017	0.052	0.14	-0.09	0.25	0.034	0.012	-0.065	1	0.024	0.05	0.096	0.052
Education	-0.0067	0.36	0.056	-0.011	-0.021	0.03	0.017	-0.038	0.024	1	0.014	-0.027	0.079
Race	-0.0053	0.032	0.042	-0.12	0.0068	0.011	0.019	-0.068	0.05	0.014	1	0.087	0.072
Gender	-0.0025	0.012	0.23	-0.58	0.08	0.048	0.046	-0.13	0.096	-0.027	0.087	1	0.22
Credit	0.0054	0.34	0.23	-0.25	0.075	0.22	0.15	-0.2	0.052	0.079	0.072	0.22	1

10. Using RandomUnderSampler resample the data and equalize them.
11. Preprocess the data and normalize them using StandardScaler.
12. Split the training data into train and test with test data being 20% for cross validation.
13. Cross Validate the data for each algorithm to determine which algorithm has the best F1 Score and Accuracy.



	model	train_score	test_score	fit_time	score_time
0	SVC	0.752454	0.750516	4.625603	0.483149
1	DecisionTree	0.920412	0.777861	0.019499	0.001399
2	RandomForest	0.854418	0.828179	2.130608	0.129633
3	GaussianNaiveBayes	0.584637	0.583757	0.017379	0.004594
4	KNeighborsClassifier	0.837915	0.787324	0.091696	0.200359

14. By cross validation we found that Random Forest has the best F1 Score and therefore we'll use Random Forest model to predict the credit risk for actual test data.
15. Result of actual test data Credit prediction is Public Score: 0.66
16. Write the predicted output in a file using FileWriter.

Models:

SVM - In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is several features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well

K-NN - The K-Nearest Neighbor classification algorithm assumes that similar things exist in proximity. In other words, similar things are near to each other. Using this approach, we will write our own kNN classification algorithm in Python and predict the sentiments based on the review

Decision Tree - It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The result is a tree with decision nodes and leaf nodes

Random Forest - Random Forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

Naive Bayes - Based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Conclusion:

Successfully implemented five different classification algorithm using Python and found Random Forest to be the most accurate. The model can predict credit risk either good or bad based on the input features.