Predicting the credit card default using machine learning algorithms

Team Members:
Md Rayhanul Islam
Eranda Sooriyarachchi

Department of Computer Science
Iowa State University
rayhanul@iastate.edu

Dec 09, 2022

Background of the project

- Banks are interested to know who among their customers would fail to do payment
- Provides early information about who is going to be defaulters
- Aims to classify customers defaults and non-defaults

Dataset [Yeh17]

- This credit dataset is collected from UCI machine learning repository
- 30000 customers data with 25 distinct features including gender, education, marital status, payment history, etc.

	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
0	1	20000.0	2	2	1	24	2	2	-1	-1	-2	-2	3913.0	3102.0	689.0	0.0	0.0	0.0	0.0	689.0	0.0
1	2	120000.0	2	2	2	26	-1	2	0	0	0	2	2682.0	1725.0	2682.0	3272.0	3455.0	3261.0	0.0	1000.0	1000.0
2	3	90000.0	2	2	2	34	0	0	0	0	0	0	29239.0	14027.0	13559.0	14331.0	14948.0	15549.0	1518.0	1500.0	1000.0
3	4	50000.0	2	2	1	37	0	0	0	0	0	0	46990.0	48233.0	49291.0	28314.0	28959.0	29547.0	2000.0	2019.0	1200.0
4	5	50000.0	1	2	1	57	-1	0	-1	0	0	0	8617.0	5670.0	35835.0	20940.0	19146.0	19131.0	2000.0	36681.0	10000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	0	0	188948.0	192815.0	208365.0	88004.0	31237.0	15980.0	8500.0	20000.0	5003.0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	0	0	1683.0	1828.0	3502.0	8979.0	5190.0	0.0	1837.0	3526.0	8998.0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	0	0	3565.0	3356.0	2758.0	20878.0	20582.0	19357.0	0.0	0.0	22000.0
29998	29999	80000.0	1	3	1	41	1	-1	0	0	0	-1	-1645.0	78379.0	76304.0	52774.0	11855.0	48944.0	85900.0	3409.0	1178.0
29999	30000	50000.0	1	2	1	46	0	0	0	0	0	0	47929.0	48905.0	49764.0	36535.0	32428.0	15313.0	2078.0	1800.0	1430.0

30000 rows x 25 columns

Data Pre-processing

Data Normalization

- Different features have a different range of values, which impacts the accuracy of models.
- The MinMaxScaler normalization technique [skl13] is used to normalize the dataset using the following Equation 1.

$$x_{normalized} = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

Data Balancing

- Data balancing is accomplished using SMOTE sklearn library with hybrid approach [skl13], which increases the minority class and decreases the majority class.

default.payment.next.month	Imbalanced dataset	Balanced dataset
0	23364	11010
1	6636	11981
Total	30000	22991

Data Pre-processing (cont.)

Feature engineering and new features generation

- Categorical features are converted binary-valued features with one hot encoding
- New features are introduced to provide more information to the models, for example, payment_score is calculated by summing all payments
- 30 new features were created, some of these features are provided below-
  - Payment_score : Summation of all Payment 1 to Payment 6.
  - Bill_amt_score : multiplication of payment score with average bill amount
  - BUSTED_LIMIT : Whether someone busts his/her limit.
  - BILL_TO_BALANCE : Ratio between the average bill and the balance limit.
  - LAST_BILL_TO_BALANCE : Ration between last bill and balance limit.
  - Pay_1 to 6 : Whether payment is duly or delay.
  - TOTAL_PAID_ON_TIME : Number of timely payments

Data Pre-processing (cont.)

Feature selection

- All features are not equally important and large number of features creates the curse of dimensionality issues.
- Features are reduced based on their impact to default.payment.next.month
- We used Pearson correlation analysis to identify the features positively contributing to default.payment.next.month using the following Equation 2.

$$r = \frac{\sum(x_i - x')(y_i - y')}{\sqrt{\sum(x_i - x')^2(y_i - y')^2}} \tag{2}$$

- Finally, 80% data for training and 20% for testing
- 10-fold cross-validation

Intuition behind selecting models

- **Logistic Regression:** works well when data are binary-valued and separable.
- **Linear Perceptron:** requires the dataset to be linear but we used it to compare results with Logistic Regression.
- **Support Vector Machine:** Even though the dataset is not linearly separable, the kernel function makes SVM workable in our data.
- **Naive Bayes:** performs well in cases of categorical input variables. As the dataset is categorical and small, we use Naive Bayes.
- **QDA:** makes the Quadratic decision boundary, which is why we selected this algorithm.
- **K-Nearest Neighbors:** As the data is numerical and plottable in N-dimensional space, that is the reason we choose knn models.

Intuition behind selecting models (cont.)

- **Decision Tree**: works well with binary features, which is why we considered the decision tree to apply to our dataset.
- **Random Forest**: applies the Bagging technique and shows how ensemble learning improves performance.
- **AdaBoot**: It is used to see how priorities to the misclassified sampled improved performances.
- **XgBoost**: How extreme boosting can give more priority to the misclassified samples and correct it in the next iteration.
- **Multi-layer Perceptron(MLP)**: A feedforward artificial neural network with at least three layers to classify our dataset

Hyper-parameters selection

- **No hyper-parameters:** Logistic regression, Linear Perceptron, Naïve Bayes and QDA
- **Elbow technique:** used to find the hyper-parameters of different machine learning models
- **SVM:** Kernel function: rbf, and error rate, c: 3
- **Knn:** k value: 31
- **Decision tree:** alpha: 8.110879909225043e-05
- **Random Forest:** # of estimators: 26 and depth:14
- **XgBoost:** Objective: binary:logistic, max depth:30, # of estimators: 100, learning rate:0.1
- **AdaBoost:** # of estimators: 90
- **MLP:** # of iterations: 90, and learing rate:0.01

Evaluation Metric

- Accuracy evaluates the performance that how many customers can be correctly labeled where $Accuracy = \frac{TP+TN}{P+N}$
- ROC value is calculated using TPR and FRP where $TPR = \frac{TP}{TP+FN}$ and $FPR = \frac{FP}{TP+FP}$
- For better understanding, confusion matrix is provided in the following table-

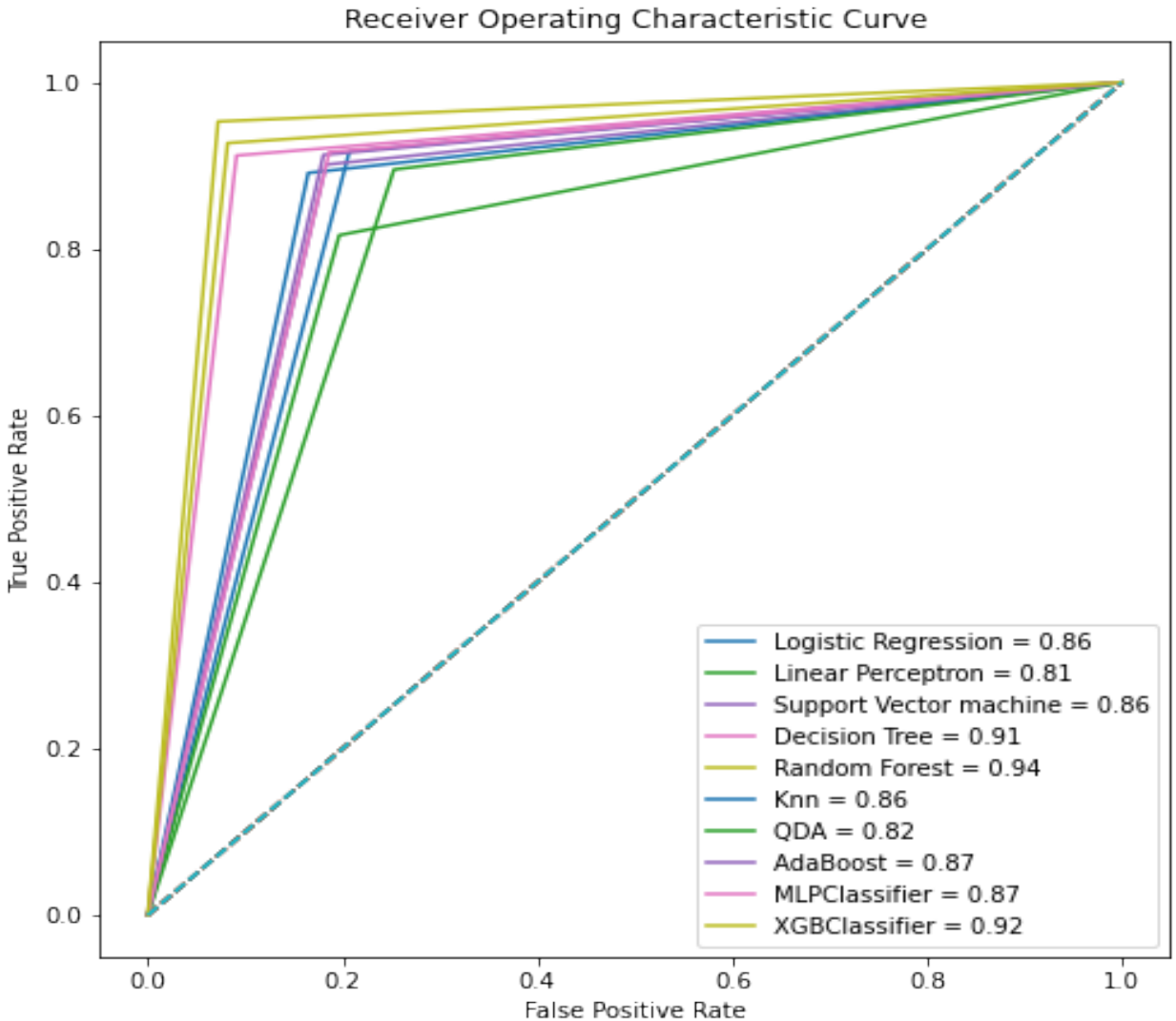
	Predicted True	Predicted False
Actual True	TP	FN
Actual False	FP	TN

Result Analysis

Model	Training accuracy	Testing accuracy
Logistic Regression	0.8498	0.8490
Linear Perceptron	0.7647	0.8103
SVM	0.8616	0.8573
Decision Tree	0.9188	0.9106
Random Forest	0.9392	0.9408
Knn	0.8621	0.8632
AdaBoost	0.8641	0.8638
MLP	0.8617	0.8617
XgBoost	0.9228	0.9228
Naive Bayes	0.8019	0.8021
QDA	0.8019	0.8106

The most outperforming models in testing-Decision Tree, Random Forest and XgBoost.

Result Analysis (cont.)



Result Comparison

Model	work A [worb]	work B [wora]	Our accuracy
Logistic Regression	0.8054	0.7807	0.8490
Decision Tree	-	0.7688	0.9106
Random Forest	0.8173	0.7918	0.9408
Knn	-	0.7752	0.8632
AdaBoost	-	0.7660	0.8638
XgBoost	0.8196	0.7087	0.9228
Naive Bayes	-	0.7874	0.8021

Conclusion

- Dataset is normalized and balanced before applying machine learning models
- Tree-based model with bagging and boosting technique outperforms the other models
- Decision Tree model, Random Forest model, and XGBoost have achieved above 90% accuracy

References

[skl13] API design for machine learning software: experiences from the scikit-learn project.
In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[wora] Kaggle.
`https://www.kaggle.com/code/samuelpaquettepar/default-payments-of-credit-card-clients-analysis#Model-Experimentation`.
Accessed: Dec 09, 2022.

[worb] Medium.
`https://ameyband.medium.com/default-of-credit-card-clients-dataset-classificati`

Accessed: Dec 09, 2022.

[Yeh17] I-Cheng Yeh.
UCI machine learning repository, 2017.