Machine Learning for Credit Score Classification

Stoyan Stoyanov	**Varun Iyer**	**Partha Sarathi Dutta**
MIE Department	CS Department	MIE Department
UIC	UIC	UIC
663070504	664622523	658414357
sstoya3@uic.edu	*viyer9@uic.edu*	*pdutta3@uic.edu*

Abstract

This work aims to apply machine learning to the task of credit score classification, wherein a user's credit score may be considered Good, Standard, or Poor. We aim to apply these approaches to a Kaggle dataset for this task. We employ a variety of machine learning methods such as random forest, XGBoost, SVM, logistic regression, and MLP. The performance of all the models in the credit score classification is compared. The impact of the hyperparameters of the individual models on the accuracies is also investigated. We also gave a lucid explanation of how the models actually work. In our search for the best method, we find XGBoost achieves the best performance on the train, dev, and test splits of our dataset. This work also looks to analyze our results and see what results in XGBoost's superior performance in this task.

1 Introduction

Artificial Intelligence and in particular machine learning has become increasingly popular over the recent years. In our modern world it finds more and more areas of application ranging from image recognition, language translation, and stock trading to product recommendations and social media. Machine learning combines statistics with computer science to create models which have the capability to learn the optimal feature relationships within the data and then predict outcomes. These algorithms can be used for classification, such as deciding whether an email is spam or not, or regression, which has a continuous output and can be used to predict stock prices for example. There are 3 main groups of machine learning: Supervised learning, Unsupervised learning, and Reinforcement learning. In Supervised learning the input and output data is provided and separated into training data used to optimize the parameters of the model and testing data used for model validations. The model is then used to make predictions on previously unseen new data. Unsupervised learning is used to identify relationships in an unlabeled dataset by grouping data into clusters by association. Mainly it is used in clustering, anomaly detection, dimensionality reduction, and finding association rules. Reinforcement learning explores different actions and tries to maximize long term rewards. It has applications in robotics, autonomous machines and game playing.

A lot of financial institutions are employing different machine learning algorithms to monitor customer behavior, identify anomalies and prevent fraud, automate tasks, provide better customer experience, and assess risks. A tool for assessing risk that many banks use is the Credit score. It is a number that varies from 300 to 850 and indicates a person's creditworthiness. There are several different credit bureaus that maintain an individual's credit reports, and these include Equifax, Experian, and TransUnion. These bureaus collect data on the financial behavior of the individual and create a credit report that is used to determine the credit score. The factors that influence it are divided into 5 categories: Payment history, Outstanding balances, Length of credit history, Applications of new credit accounts, and Types of credit accounts. Banks use the credit score as a means to assess the risks associated with lending money and determining the interest rates of the loans. For example, if an individual with a high credit score is applying for a mortgage or a car loan, they are more likely to get approved for the loan and also get it at a lower interest rate. It is especially important for banks to be able to identify applicants who are very likely to default on their loans, and minimize the number of false positives since that is associated with a lot of cost that the bank needs to pay.

Our group focused on the application of machine learning algorithms in classifying credit scores. The motivation for the project is that the credit score is a widely used tool in banking, and also that credit report checks negatively impact the credit score, so it will be beneficial to be able to classify a credit score without actually requesting it from the credit bureau. The main challenges in this project lie in employing multiple suitable machine-learning algorithms for this task and validating the observations of this project with the techniques learned in this course. The machine learning algorithms that were employed in this project include Random Forest, XGBoost, Logistic regression, Support Vector Machines, and Multilayer Perceptron. The models are used to classify the credit score of an individual as poor, standard, or good. To evaluate the performance of these models a variety of metrics were employed such as Accuracy and F1 score and plots of the ROC curves.

2 Data analysis

The data that we used is from Kaggle, and it contains 100,000 rows (data points) and 28 columns (features). It contains both numerical and categorical features alike. Features include age, name, SSN, ID, Customer ID, Month, occupation, credit history age, outstanding debt, annual income, monthly balance, monthly in-hand salary, number of bank accounts, number of credit cards, interest rate, type of loan, number of loans, delay from due date, number of delayed payment, changed credit limit, number of credit inquiries, credit mix, outstanding debt, credit utilization ratio, payment of minimum amount, total EMI per month, amount invested monthly, payment behavior, and monthly balance. Our models will be tasked to use this information to predict the user's credit score. The dataset was split into a training set of 70,000 rows, a development set of 10,000 rows, and a test set of 20,000 rows. This train-dev-test split was performed with a fixed random seed so that our results are reproducible. The data is further preprocessed by concatenating numeric features with categorical 1-hot feature vectors. After converting the categorical variable to 1-hot-vectors the dataset has 44 features. The resulting unified input vector representation is then fed to the machine learning algorithms. Histograms of some features of the date are shown in Figure 1, Figure 2, and Figure 3.

Figure 1: Credit Score Histogram

As seen in Figure 1, the credit score is normally distributed with the majority of people having a standard credit score.

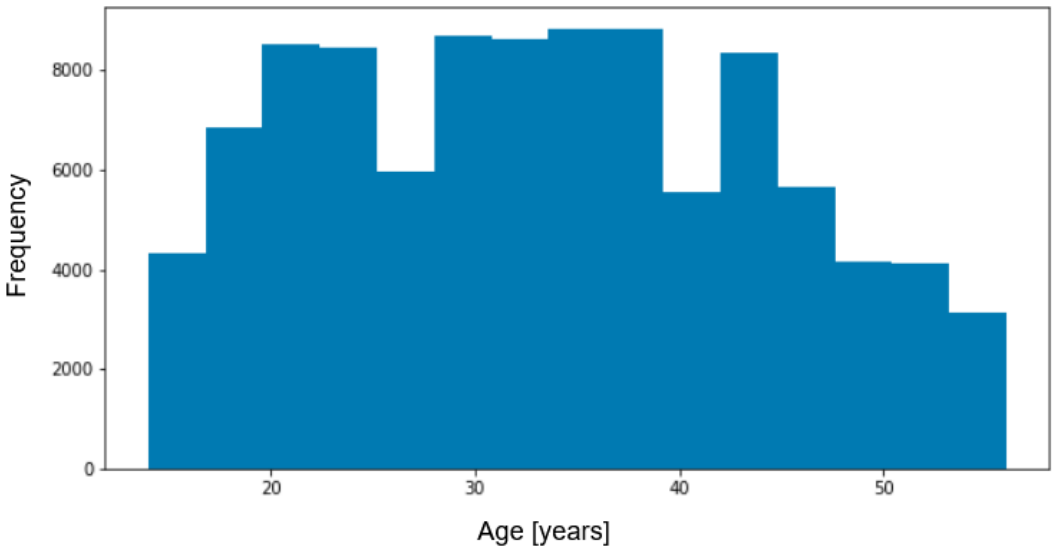

Figure 2: Age Score Histogram

From Figure 2 it can be seen that the samples are not uniformly distributed with respect to the age. This corresponds well with the real world.

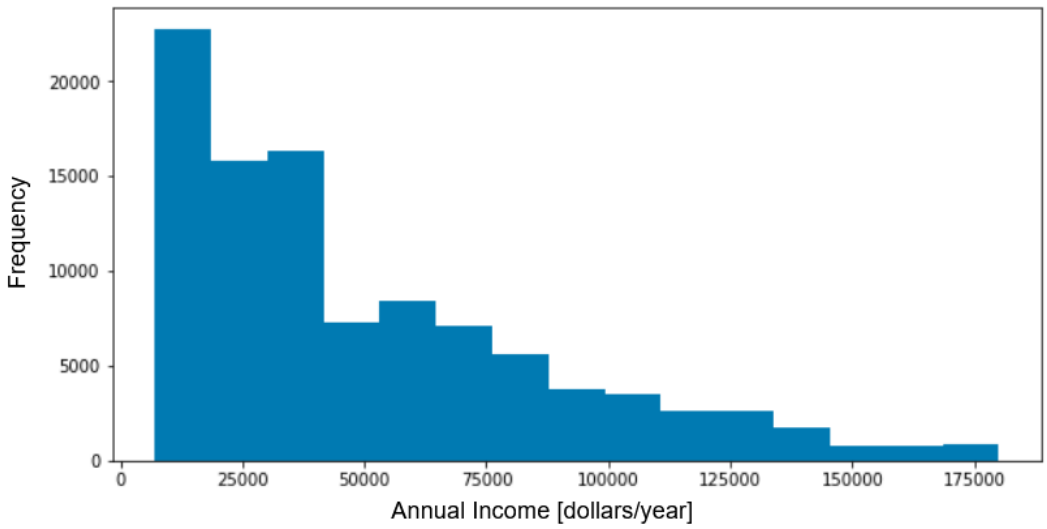

Figure 3: Annual Income Histogram

Figure 3 shows the histogram of the annual income. It can be seen that the data is skewed towards the lower income, which is consistent with the pay distribution in real life; the majority of the people are in the lower income bracket.

3 Methods

Random forest, XGBoost, Support vector machine, logistic regression, and multi-layer perceptron are the five discriminative models used for the classification of credit scores. Also from the existing literature, discriminative models are found to perform better than generative models for credit-score classification [1]-[2].

3.1 Random forest

Random forest works by forming several weak decision trees only trained using a limited number of feature splits. These weak decision trees are used in an ensemble by having them vote for a classification prediction or average predictions for regression [3]. Each individual decision tree split is based on a randomly selected feature. In the case of numerical features, the threshold for the feature is chosen to maximize information gain as with normal decision trees. However, since we represent categorical features as one-hot vectors, each component of the one-hot vector is treated as a numerical feature. We achieve acceptable performance using this approach and leave more sophisticated handling of categorical features to future work.

3.2 XGboost

XGBoost works using an ensemble of decision trees like random forest, but constructs trees and leverages their predictions in a different fashion. XGBoost aims to successively form decision trees where each decision tree repairs errors made by the previous trees. This process can be controlled using a learning rate which can result in more or fewer trees being needed for the model to converge on train data and generalize properly to test data. As with random forest, due to our encoding of categorical variables, we currently treat them as vectors of numerical variables for each tree in XGBoost.

3.3 Support Vector machine

In the Support vector machine, the discriminant is defined in terms of the support vectors [4]. And depending on the application we can choose an appropriate kernel for the measurement of similarity. In SVM, the optimization problem is to maximize the margin, which is defined as the distance between the separating hyperplane and the marginal hyperplanes. There are two types of SVM: hard margin and soft margin. In hard margin SVM, there are no outliers and all the data lies on the appropriate sides of the decision boundary. Soft SVM is used when the data is not linearly separable and we introduce an error (also called hinge loss), and how much error we introduce is a trade-off between regularization and loss. For non-linearly distributed data we can use non-linear kernels like sigmoid, radial basis function, polynomial kernel, etc.

3.4 Logistic regression

Linear discriminant analysis is another discriminant model used in this work and it provides the added advantage of lower space complexity and the complexity of the model is proportional to the number of classes [5]. The computation is further optimal when the distribution of the individual classes is gaussian with shared covariance or when the decision boundary is linear. It uses gradient descent-based optimization and maximum likelihood estimation. There is no closed-form solution but the optimization problem is convex and hence easy to solve. Also, we can get an idea of the importance of the features from the weights of the attributes in the discriminant function. There is also a generalized model which can account for any non-linearity in the data. Regularization techniques (L2 regularization/ L1 regularization) are used to regulate the complexity or overfitting of the model as it regularizes the number of features considered.

3.5 Multi-layer perceptron

Apart from SVM and linear discriminant analysis, neural networks are used in this work to model the non-linearity of the data [6]. It uses multiple layers, and different kinds of activation functions to model the non-linearity. The weights of the network are updated by the backpropagation method and regularization techniques can be used to control the overfitting/ complexity of the model. The usage of multiple layers in MLP makes the optimization problem non-convex and we have to carefully choose the hyperparameters which result in the least training and test losses. However, despite all these uncertainties, neural networks are still one of the most powerful methods both for classification and regression.

4 Experimental Results

4.1 Random Forest

The random forest classifier from sci-kit learn was used with a fixed random seed for reproducibility. Tunable hyperparameters for this model include the number of trees and the depth of each tree. For these, 100 trees with a max depth of 30 were found to be optimal hyperparameters when tuned on the dev set.

4.2 XGBoost

The XGBoost classifier was used from the xgboost Python library with a fixed random seed for reproducibility. Tunable hyperparameters for this model include the number of trees, depth of each tree, and learning rate. For these, 100 trees with a max depth of 30 and learning rate of 0.1 were found to be optimal hyperparameters when tuned on the dev set.

4.3 Support vector classifier

The SVC function of scikit-learn was used for the support vector classifier. As the distribution of the data was non-linear, we used non-linear kernels, namely polynomial and radial basis functions. For the polynomial kernel, we have chosen the degree as 3 which resulted in optimum accuracies. We have also experimented with different values of the regularization parameter C. The table 1 below summarizes three different values of the regularization parameter and the corresponding training and test accuracy for a polynomial kernel.

Table 1: Comparison of training accuracy and test accuracy for different values of the regularization parameter C for a polynomial kernel SVM.

C	Training Accuracy (%)	Test Accuracy (%)
0.1	54.19	54.26
1	63.05	62.93
10	67.66	67.64

The value of C=10 was used for both the polynomial kernel and the rbf kernel. Both the kernels perform similarly, and only the rbf kernel slightly overfits the training data. The training accuracy and the test accuracy for the polynomial kernel are around 67%. For the rbf kernel, the training accuracy is 71% and the test accuracy is 67%.

4.4 Logistic regression

For computing the logistic regression, the scikit-learn library was used, and to make the results reproducible, the random state was fixed. The lbfgs solver was used for optimization. And as it is a multi-class classification we chose the multinomial method. The one-versus-rest method of classification did not change the result significantly. The L2 regularization method was used for controlling the overfitting or complexity of the model. For the logistic regression, the training accuracy was 67%, the test accuracy was 61% and the F1 score was around 57 percent.

4.5 Neural network

The neural network was implemented using the Keras framework. The experimentation of the DNN consisted of various depth, width, and activation functions of the network. Adam optimizer was used for the training and 3 different learning rates 0.01, 0.001, and 0.0001 were used. The difference in accuracy in all three learning rates was less than 1 percent. As it is a multi-class classification problem (3 classes), the loss function was chosen as categorical cross-entropy. The batch size for the stochastic gradient descent step was chosen as 50. And for computational limitations, the number of epochs used for training was 50 but the curve of training accuracy/ test accuracy almost flattened after 50 epochs. For the activation function, different non-linear activation functions like sigmoid, relu, and tanh were experimented with but relu gave the best performance. The neural network was slightly overfitting and so a dropout of 0.2 was introduced between the layers as a regularization measure. The training accuracy obtained for the neural network was 76 percent and the test accuracy was 70. The performances of all the models are summarized in the table 2 below:

Table 2: Comparison of training and test performance for various models

Model	Train Acc. (%)	Test Acc. (%)	Test F1 Score (%)
Random Forest	99.91	77.96	75.91
XGBoost	100.00	79.12	78.29
Logistic Regression	66.31	61.61	57.12
SVM (Poly kernel)	63.05	62.95	62.17
SVM (RBF kernel)	71.67	67.79	64.42
MLP	76.65	70.80	69.23

5 Conclusion

Out of all the models, the decision tree-based approaches perform the best as it uses a series of if-else statements to model the non-linearity in the data. Gradient boosting further increases the accuracy. However, the gain in accuracy in the case of decision trees comes with slight overfitting. The future scope lies in controlling the overfitting of the decision tree by employing techniques like experimenting with pruning trees in the ensemble which are not directly supported by libraries like sci-kit learn.

The other observation is that the SVMs with non-linear kernels outperform the logistic regression model. Both the polynomial kernel and the rbf kernel perform similarly except the rbf kernel slightly overfits the training data.

Also, a sufficiently sophisticated neural network should perform close to XGBoost and so the future scope lies in experimenting with more model hyperparameters. Additionally, models with greater inductive bias than a fully-connected MLP may be leveraged for this specific type of tabular data.

The performance of any machine learning/deep learning model greatly depends on the individual features and how the features affect the accuracy. An extension of this work also lies in studying how the individual features affect the accuracies, proper feature engineering before training the model, detection of outliers and their replacement, etc.

Acknowledgments

We thank Prof. Xinhua Zhang for the machine learning concepts taught in the class. We also declare that there is no conflict of interest.

References

[1] Moscato, V., Picariello, A., & Sperlí, G. (2021). A benchmark of machine learning approaches for credit score prediction. Expert Syst. Appl., 165, 113986.

[2] Tripathi, D., Edla, D.R., Bablani, A., Shukla, A.K., & Reddy, B.R. (2021). Experimental analysis of machine learning methods for credit score classification. Prog. Artif. Intell., 10, 217-243.

[3] Yeh, Ching-Chiang, Fengyi Lin, and Chih-Yu Hsu. "A hybrid KMV model, random forests and rough set theory approach for credit rating." *Knowledge-Based Systems* 33 (2012): 166-172.

[4] Huang, Zan, Hsinchun Chen, Chia-Jung Hsu, Wun-Hwa Chen, and Soushan Wu. "Credit rating analysis with support vector machines and neural networks: a market comparative study." *Decision support systems* 37, no. 4 (2004): 543-558.

[5] Baesens, Bart, Tony Van Gestel, Stijn Viaene, Maria Stepanova, Johan Suykens, and Jan Vanthienen. "Benchmarking state-of-the-art classification algorithms for credit scoring." *Journal of the operational research society* 54, no. 6 (2003): 627-635.

[6] Zhao, Zongyuan, Shuxiang Xu, Byeong Ho Kang, Mir Md Jahangir Kabir, Yunling Liu, and Rainer Wasinger. "Investigation and improvement of multi-layer perceptron neural networks for credit scoring." *Expert Systems with Applications* 42, no. 7 (2015): 3508-3516.