Applying Machine Learning Algorithms to Predicting Loan Default Risk

DATS 6202 Group 5 Final Project

Summer 2018

Kimberly Kreiss, Bijiao Shen & Zhoudan Xie

1. Introduction

Our project comes from a kaggle competition and uses data from a company called Home Credit. Home Credit is a company that operates in several different countries including several countries in Europe and Asia in addition to the United States. The company's business model focuses on making loans to people with little or no credit. Their loan products primarily include point of sale loans (POS), cash loans, credit cards, debit cards, revolving loans, or car loans.

Our project uses supervised learning to predict a client's ability to repay a loan. Given the company's loan products, business model, and regions of operation, there are several unique challenges with this task. First, given that the company operates primarily by making loans to people with little or no credit, the ability to which we can use default on previous loans or credit history is limited. Moreover, since there is variation in which the company operates in, there is variation in measurement in various credit bureaus in each country. For example, lenders in the United States often use FICO scores as a large predictor in a borrower's ability to repay a loan. Given these challenges, we make use of a loan dataset provided by Home Credit.

The dataset covers a large number of loan features such as the loan type (i.e. cash or revolving loan), the gender of the client, the number of children the client has, and the annual income of the client. The dataset also has the target—whether the client had payment difficulties for the loan (i.e. default or not default). We employed three supervised machine learning algorithms to estimate the relationship between the target and key loan features, including the Support Vector Machine (SVM), Naïve Bayes, and Random Forest.

This report is structured as follows. The next section describes the data source, the structure, and the target and features in the data. Section 3 includes background information and principles of the three algorithms we use. Section 4 specifically discusses how we use the loan data to train and test the networks. Section 5 provides detailed discussions of the results of each network. Finally, section 6 summarizes the project.

2. Description of the data set

There is one dataset in CSV format named "application_train.csv," which is involved in the Home Credit borrower default analysis. The data is sourced from Kaggle "Home Credit Default Risk" competition[1]. The dataset used for SVM and Naive Bayes is slightly different from the one for Random Forest.

[1] Detailed information can be accessed through the link: https://www.kaggle.com/c/home-credit-default-risk

(1) Data Set Information

The dataset is originally from Home Credit, which is a non-banking financial institution, founded in 1997 in Czech Republic. The objective of the dataset is to diagnostically predict whether or not the borrowers repay the loan, based on certain measurements described in the data. The dataset consists of two parts: loan application information collected by Home Credit and previous loan perforation information collected by credit bureau. The dataset consists of several financial predictor variables and one target variable.

<u>Support Vector Machine and Naïve Bayes</u>

(2) Number of Instances: 316
(3) Number of Attributes: 16
(4) Attribute Information:

Target

- Target Class Variable (0 or 1) - Categorical

Predictor

- Loan Type - Categorical
- Gender - Categorical
- Car Ownership - Categorical
- Realty Ownership - Categorical
- Number of Children - Numerical
- Education Type - Categorical
- Age - Numerical
- Income - Numerical
- Regional Rating based on Credit Home - Categorical
- Number of Requires to Credit Bureau about the Client - Numerical
- Normalized Score from External Data Source 1 - Numerical
- Normalized Score from External Data Source 2 - Numerical
- Normalized Score from External Data Source 3 - Numerical
- Address Mismatch Indicator - Categorical

<u>Random Forest</u>

(2) Number of Instances: 2581
(3) Number of Attributes: 17
(4) Attribute Information:

Target

- Target Class Variable (0 or 1) - Categorical

Predictor

- Loan Type - Categorical
- Gender - Categorical
- Car Ownership - Categorical
- Realty Ownership - Categorical

- Companion during Loan Application - Categorical
- Clients Income Type - Categorical
- Level of Highest Education - Categorical
- Family Status - Categorical
- Housing Situation of the Client - Categorical
- Occupation Type - Categorical
- Application Weekday - Categorical
- Occupation Type - Categorical
- Residential Apartment Building Type - Categorical
- Residential House Type - Categorical
- Residential Wall Type - Categorical
- Residential Emergency Type - Categorical

3. Machine Learning Networks

In this section, we describe the fundamental principle of the three training algorithms we use in the project: Support Vector Machine, Naïve Bayes, and Random Forest. We briefly go through some background information and the mathematical concepts of these algorithms.

A. Support Vector Machine

The Support Vector Machine (SVM) is a discriminative classifier that identifies an optimal decision boundary between data, which can be used for classification and regression problems. The SVM algorithm was first introduced by Vladimir Vapnik in 1963, and further developed by Alexey Chervonenkis as the *VC theory* (Kowalczyk 2017). Since then, it has become one of the most popular supervised machine learning algorithms, especially in the areas of text categorization, image recognition, and bioinformatics (Cristianini & Shawe-Taylor 2000).

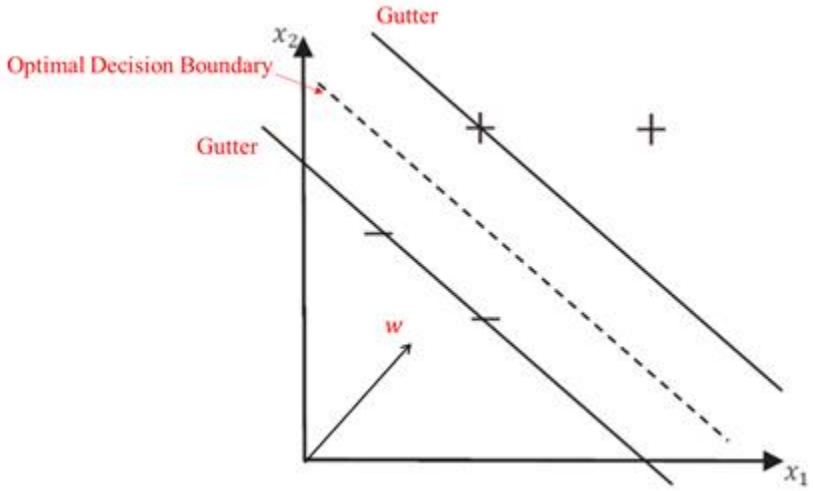
The SVM is superior to an earlier simple learning algorithm--Perceptron, as it not only attempts to find a decision boundary that classifies the data correctly, but also to find the optimal one which separates the data in the best way. Hence, contrary to the Perceptron which might find a different decision boundary every time, the SVM will always produce the same result. In a simplified linearly separable example, the optimal decision boundary sits in the middle of the "widest street" between the data in two different classes (as shown in Figure 1).

In the mathematical term, the optimal boundary can be found by maximizing the street width function ($2/||w||$), or simply minimizing $||w||$--the weight orthogonal to the decision boundary. At the same time, it is also subject to the constraint: $y(w \cdot x + b) - 1 = 0$, where w is the weight, x is a support vector, b is a bias, and y is an adjusting term such that $y = 1$ for the positive samples and $y = -1$ for the negative samples. Using the Lagrange Multiplier method, the optimization can be solved by letting

$$w = \sum \alpha_i y_i x_i$$

where a is the learning rate.

Figure 1: A simplified linearly separable example of SVM



Source: Amir Jafari. Lecture 7. Edited by the authors.

We consider that the SVM might be an appropriate classifier for our loan dataset because we have a binary classification problem (i.e. default or not default). Further, it is less clear to us whether the boundary between the loan data is linear or non-linear, the SVM provides efficient and flexible options to transform the data through the kernel trick. In the following training, we use various kernels to test the performance of the SVM classifier, including linear, polynomial, sigmoid, and RBF.

B. Naïve Bayes

Naïve Bayes is a simple classification technique that relies on conditional probability, and predicts the most probable class given a set of inputs. It is also a simple technique for predicting the most probable class/label given a set of features/inputs. Naïve Bayes Classifiers are extremely fast and surprisingly accurate given their "naïve assumptions". It can be thought of as a "Purely Statistical" model involving joint density function, marginal density function, Bayes rule and maximum likelihood.

The Naïve Bayes equation is below with Y denotes class and X denotes features:

$$\underset{Y}{\operatorname{argmax}} P(Y|X) = \underset{Y}{\operatorname{argmax}} P(x_1|Y)P(x_2|Y)\cdots P(x_n|Y)|$$

The Naïve Bayes classifier relies on three things:

1) Independence assumption;
2) The notion of conditional probability;
3) Bayesian Inference - a method of statistical inference in which Bayes Theorem is used to update the probability for a hypothesis as more evidence becomes available.

Bayes rule is merely the mathematical relation between the prior allocations of credibility and the posterior reallocation of credibility (conditional on data).

Solving a classification problem, we need three basic steps for Naïve Bayes classifier:

1) Convert the data set into frequency tables and Use the frequency tables to calculate likelihood tables;
2) Use the product rule to obtain a joint conditional probability for the attributes;
3) Use Bayes Rule to calculate the posterior probability for each class variable them once this has been done for all classes, output the class with the highest probability.

C. Random Forest

Random Forest is an ensemble method that makes use of several decision trees and can be used for both regression and classification. Since our problem is a classification method (a person either defaults or does not), we use it in this context.

The basic principle behind Random Forest is to create several different decision trees from random pulls of the dataset, and then average these decision trees. The general idea is to average several unbiased decision trees to reduce the variance. The random forest algorithm corrects for overfitting that often happens with decision trees, and it also allows for us to more easily observe feature importance. Since Random Forest pulls several random samples of the data to create several decision trees, we do not need to worry as much about dimensionality reduction or correlation across variables as much with this model. Please see the image below for a more in-depth description of the random forest model, which comes from *The Elements of Statistical Learning*:

Algorithm 15.1 *Random Forest for Regression or Classification.*

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\mathrm{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\mathrm{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

4. Experimental Setup

In this section, we describe how we use the data to train and test the networks mentioned above. Specifically, we explain the data preprocessing, training and testing, and performance metrics we have performed. Given the different natures of the algorithms, the data preprocessing and training follow slightly different approaches.

A. Support Vector Machine

First, we notice that the original training dataset is highly imbalanced, with 8,076 observations in class 0 (not default) and only 526 observations in class 1 (default). We have attempted to perform the trainings using the imbalanced dataset, but no input in class 1 is correctly classified. To tackle this issue, we resample the data by randomly deleting the observations in the over-represented class (i.e. under-sampling class 0). As a result, we create a new balanced training dataset with 526 observations in each class.

In the data preprocessing for the SVM training, we encode all the categorical features, and normalize the numerical ones. We select and test 15 important features based on our knowledge and experience on loans. We then split the training dataset into Train and Test using the `train_test_split` function in the `sklearn.model_selection` module, where the Test data are set to be randomly selected 30% of the training data. The `random_state` option is set to be 100 to define the seed used by the random number generator.

Next, we perform the SVM training by using the `sklearn.svm.svc` package. In doing so, we define a function to allow for tests using various kernels including linear, RBF, polynomial, and sigmoid. We examine and compare the performance of each kernel by looking at the classification report, confusion matrix, and receiver operating characteristic (ROC) curve.

B. Naïve Bayes

In order to applying Naïve Bayes classifier, we start with variable summary statistical analysis on key variables selected based on the opinion of industry experts. Then we do the pre-processing on the dataset, which includes treatment of missing value, data normalization and treatment of imbalanced data.

Next, we split the dataset between training set and test set with 70/30 ratio and then we set the Naïve Bayes classifier to be GaussianNB. We fit the model using training set and test the model performance using test set.

Finally, we conduct the performance analysis on the model using confusion matrix, ROC and classification report.

C. Random Forest

In this specification, we begin by using all the variables in the balanced training dataset to train and test the network. We convert all the categorical variables into numeric variables; however, we do not need to normalize since we are using random forest, that is, a series of decision trees. We perform the rest of the necessary preprocessing which includes dropping variables with missing observations.

Next, we divide the dataset into test and training datasets, using the standard 70/30 split and initialize the random forest classifier to use 100 trees. Using the test and training data, we use the model to fit the data. Now that we have fit the data, we can observe which features are the most important in this model. Finally, we rerun the model, limiting to only the 25 most important features to improve model performance.

5. Results

We present and interpret the results of the three training algorithms mentioned above in this section.

A. Support Vector Machine

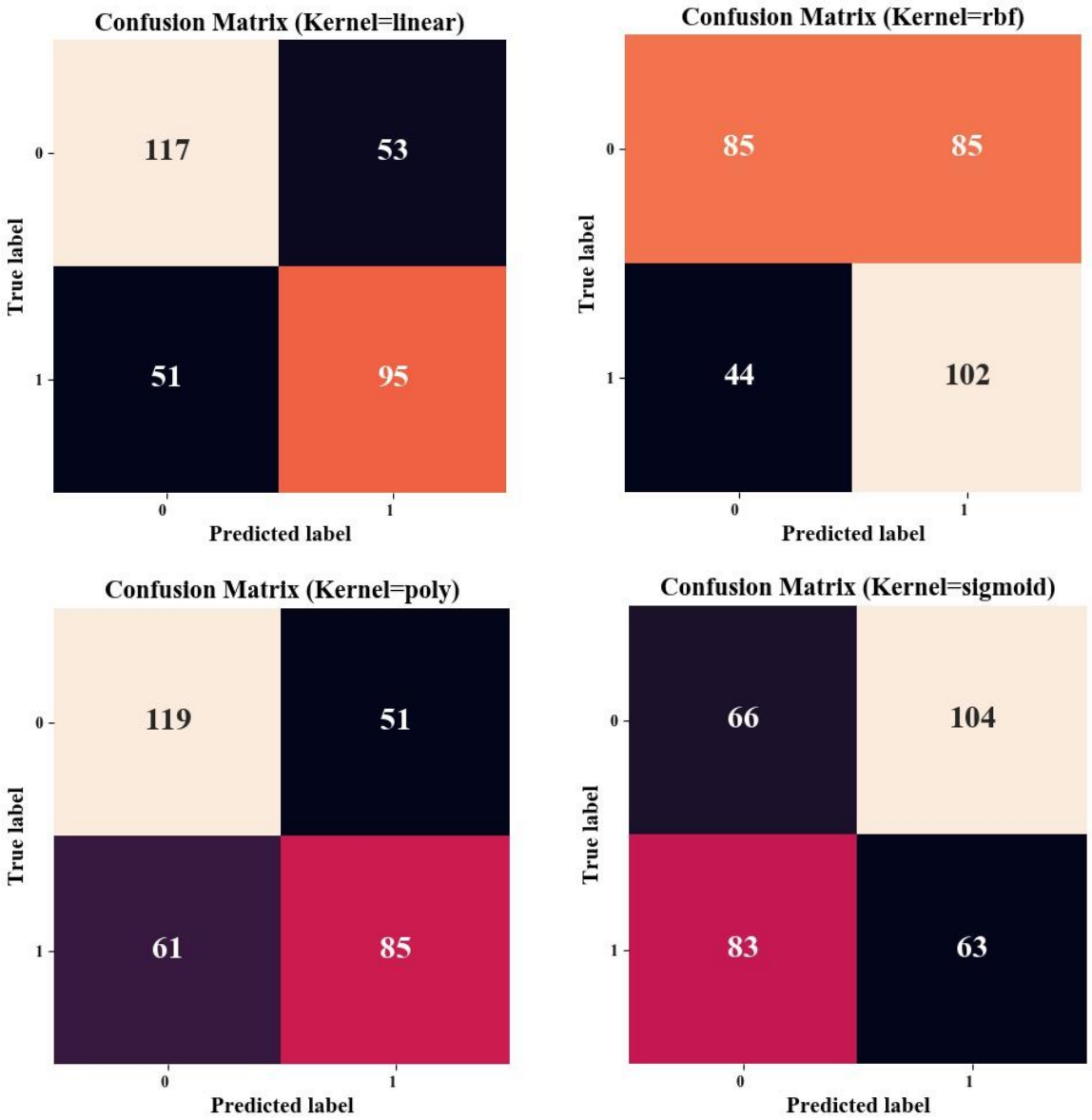
The SVM classifiers using different kernels produce different results. As shown in Table 1, the classification reports shows that the linear and polynomial kernels produce a relatively higher level of accuracy. Their average F1 scores are respectively 0.67 and 0.64. The RBF kernel has a slightly lower score of 0.59, and the sigmoid kernel performs the worst with the lowest score of 0.41.

Table 1: SVM Classification Reports

	Precision	Recall	F1-score	Support
Kernel=linear				
0	0.70	0.69	0.69	170
1	0.64	0.65	0.65	146
Avg/ total	0.67	0.67	0.67	316
Kernel=rbf				
0	0.66	0.50	0.57	170
1	0.55	0.70	0.61	146
Avg/ total	0.61	0.59	0.59	316
Kernel=poly				
0	0.66	0.70	0.68	170
1	0.62	0.58	0.60	146
Avg/ total	0.64	0.65	0.64	316
Kernel=sigmoid				
0	0.44	0.39	0.41	170
1	0.38	0.43	0.40	146
Avg/ total	0.41	0.41	0.41	316

The confusion matrix presents similar results. As shown in Figure 2, the linear SVM produces 117 true positives, 95 true negatives, 53 false positives and 51 false negatives. In other words, the algorithm is able to correctly predict 117 defaults and 95 non-defaults in the test, while incorrectly predicts 53 defaults in the loans which actually have no defaults, and 51 non-defaults in the loans which actually have defaults. Comparatively, the polynomial SVM produces a few less false positives but more false negatives. Although the RBF SVM has less false negatives, it generates much more false positives. Still, the sigmoid SVM performs the worst by producing more false positives and false negatives.

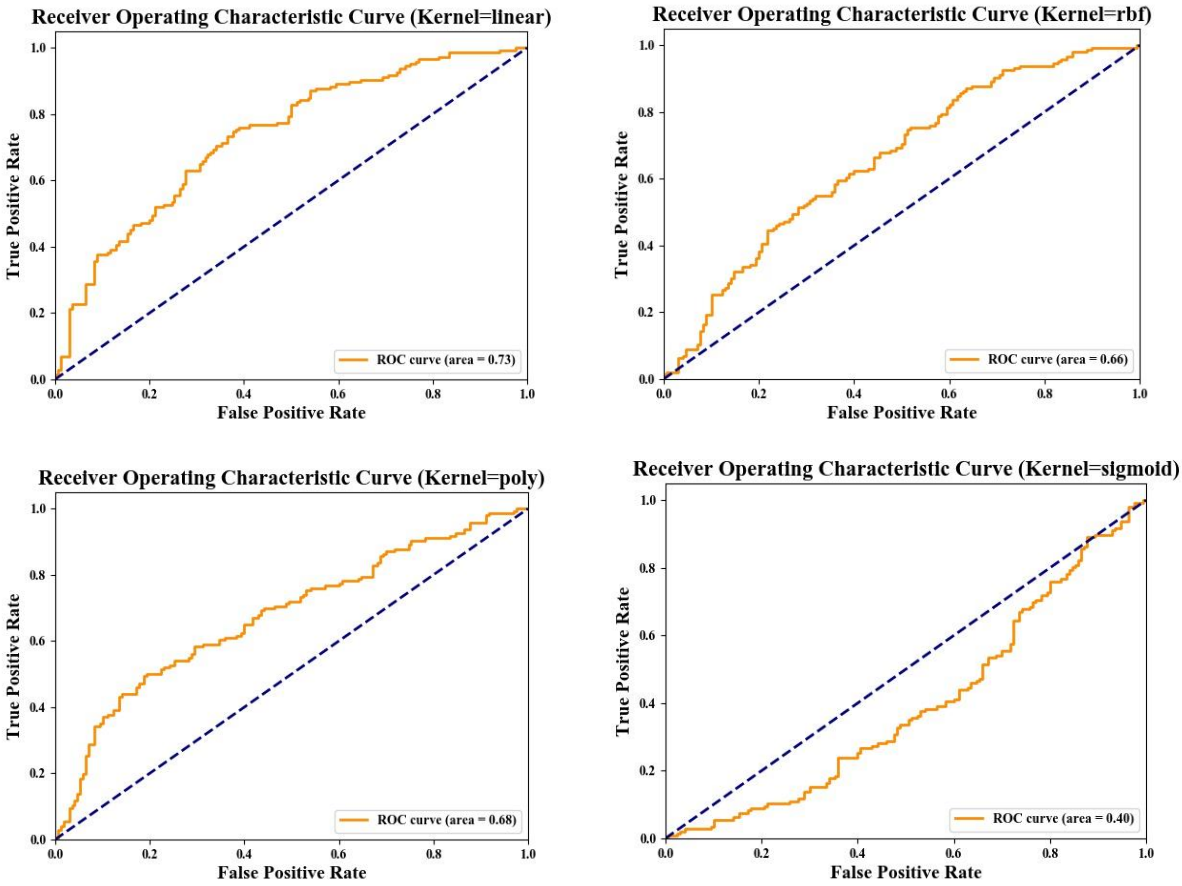
Figure 2: SVM Performance - Confusion Matrix



The area under the ROC curve gives us another measure of test performance. As shown in Figure 3, the linear SVM has the largest area under the ROC curve (0.73), which can be

considered to be “fair” performance. The polynomial and RBF classifiers present slightly smaller area, 0.68 and 0.66 respectively, and the sigmoid classifier only has an area of 0.40.

Figure 3: SVM Performance - ROC Curves



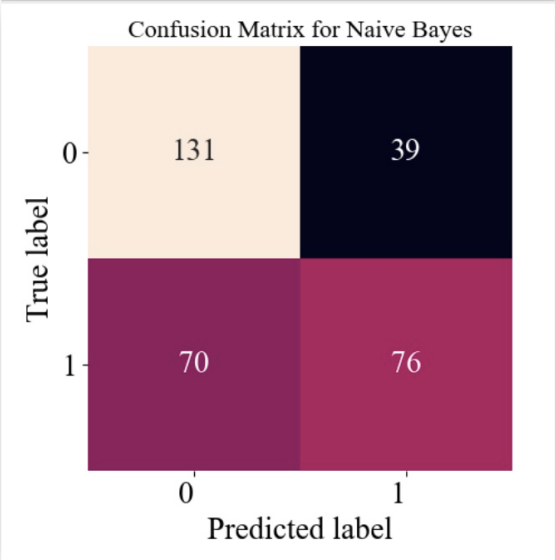
In sum, the above three performance metrics have conveyed similar messages about the SVM classifiers. In general, the linear and polynomial SVM perform relatively better in terms of accuracy, as indicated by the F1 score and the area under the ROC curve. The RBF has a lower level of accuracy, whereas it might also be a reasonable option to consider for loan companies like Home Credit, because it produces less false negatives and more false positives. Given a similar level of accuracy, lenders may prefer a more “secure” algorithm that produces more false positives than more false negatives. Put it simply, lenders may rather prefer to falsely reject 100 good applications than falsely approve one bad application. From this perspective, the RBF has its advantages over the other kernels. Nevertheless, neither of the SVM classifiers above actually achieve a very good performance, so we test the other algorithms as well.

B. Naïve Bayes

In the section, we examine the three common performance assessment methods. We have two classes in the analysis: 1 represents client with payment difficulties while 0 represents all other

cases. The Confusion Matrix shows that we correctly classify 131 loans and 76 loans about its repayment performance. We classify 39 actual performing loans as loans with repayment difficulties and classify 70 actual loans which have repayment difficulties as performing loans.

Figure 5: **Naïve Bayes - Confusion Matrix**



The ROC and classification report show a similar result. We have a 65.51% (rounding to two decimal places) accuracy ratio and it is reasonable because of the credit risk estimation difficulty of alternative lending in developing countries. According to the classification report, we have 0.65 precision for estimating class 0 which is clients without payment difficulties and 0.66 precision for classifying class 1 which is clients with payment difficulties. The credit risk classification used in our analysis is more conservative towards default and it matches the purpose of the analysis.

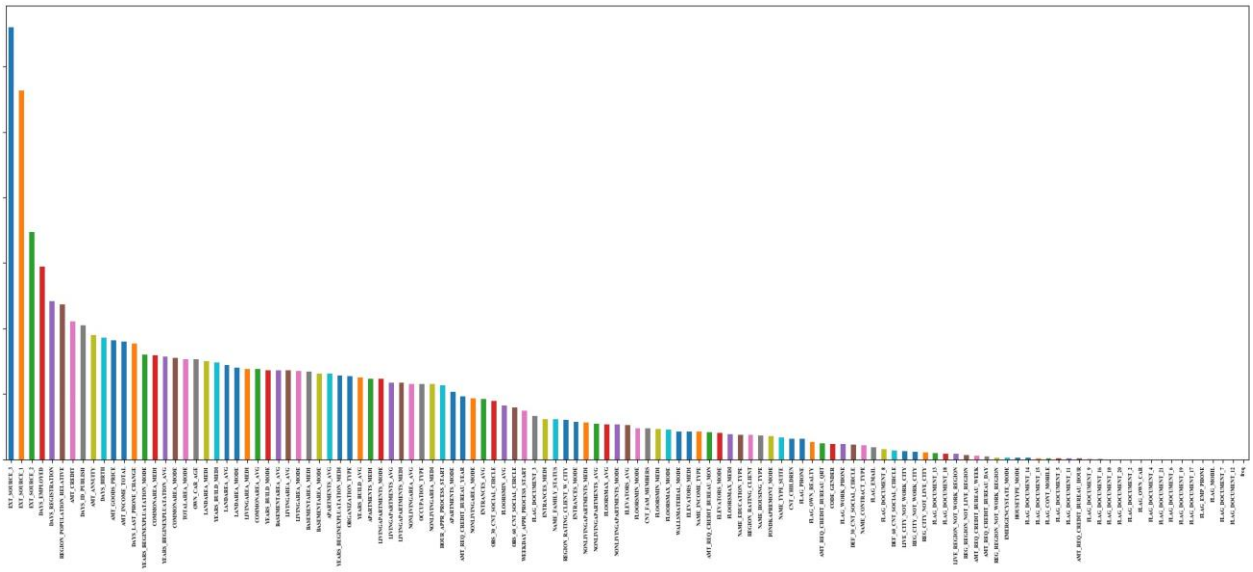
```
Classification Report:
             precision    recall  f1-score   support

        0.0       0.65      0.77      0.71       170
        1.0       0.66      0.52      0.58       146

avg / total       0.66      0.66      0.65       316


Accuracy :  65.50632911392405


ROC_AUC :  71.55519742143433
```

In general, the algorithm performance assessment using the methods discussed above shows the difficulty of estimating credit risk for alternative lending in developing countries.

C. Random Forest

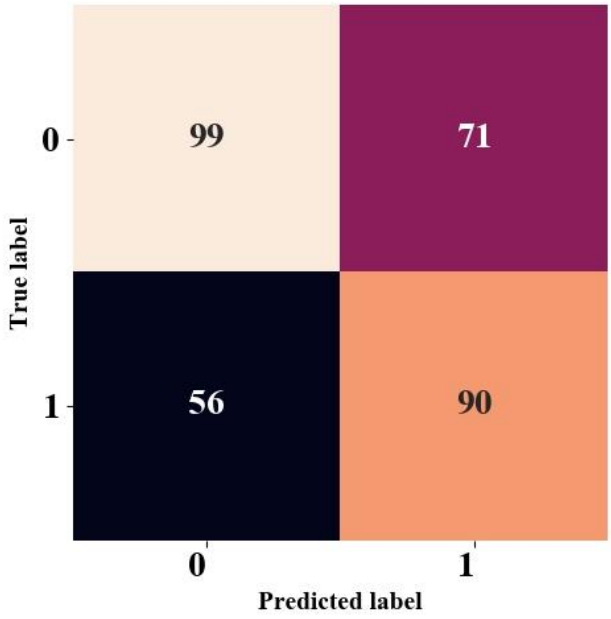
We first start by analyzing the results of the random forest model using all the features and begin by analyzing feature importance, as shown in Figure 5.

Figure 5: Random Forest Feature Importance



As we can see from the graph, the features with the highest importance are shown on the left side of the x-axis. Some of the most important features seem to be the normalized scores from external sources (EXT_SOURCE_1-3), days employed, and the number of days since registration. For a full description of each variable, please see "HomeCredit_columns_description.csv". Now we can observe the classification report, ROC and AUC scores, and confusion matrix from running this model:

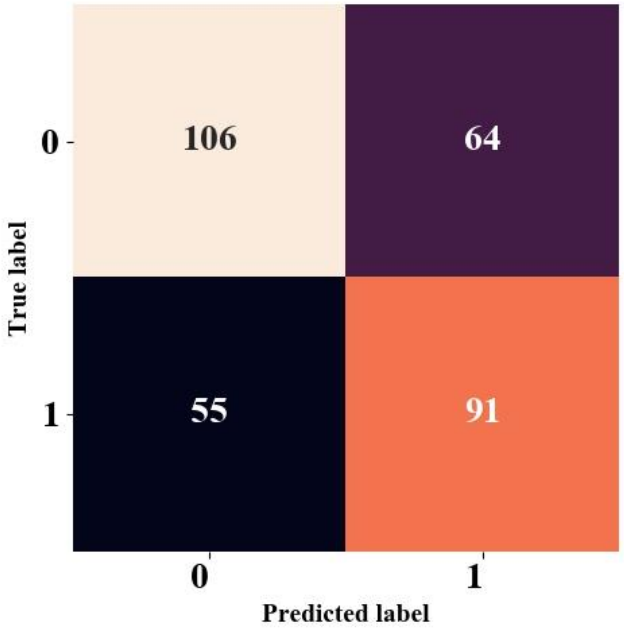
```
Results Using All Features:

Classification Report:
             precision    recall  f1-score   support

        0.0       0.64      0.58      0.61       170
        1.0       0.56      0.62      0.59       146

avg / total       0.60      0.60      0.60       316


Accuracy :  59.81012658227848


ROC_AUC :  65.8239323126511
```



We can see from both the classification report and the confusion matrix, that the model is unable to achieve high accuracy in predicting whether a client will default or not using the given features in the dataset. One way to try to correct for this is to include a higher number of trees in the random forest model. However, when implementing this with 200, 400, 800, and 1000 trees, the model does not have see any increase in accurately predicting if a person will default or not.

In an attempt to improve the model's performance, we can limit the features to a subset of those that are the most important features in the first model. Therefore, we rerun the same model using the 25 most important features. The classification report, ROC_AUC score, and confusion matrix are shown below:

```
Results Using K features:

Classification Report:
             precision    recall  f1-score   support

        0.0       0.66      0.62      0.64       170
        1.0       0.59      0.62      0.60       146

avg / total       0.63      0.62      0.62       316


Accuracy :  62.34177215189873


ROC_AUC :  67.64907332796132
```



So we can see that the performance is slightly improved, but still the model incorrectly predicts many cases. This model follows the same pattern of having similar accuracy when including more trees as well. However, we can see that the ROC_AUC score improves slightly compared with the model that uses all features. At first glance, the results from the random forest specifications suggest that the features in the dataset are simply unable to sufficiently predict if a person will default. Intuitively, this makes sense since we are lacking important information about credit history due to the nature of the customers that Home Credit lends to.

6. Summary and Conclusions

In this project, we perform supervised machine learning to predict a borrower's default risk on a loan. Using a dataset provided by Home Credit, we employ three learning algorithms, including the Support Vector Machine, Naïve Bayes, and Random Forest. We test the performance of the three algorithms by examining their accuracy scores, confusion matrix, and ROC curves. As a result, we attempt to identify a superior algorithm that can best predict a borrower's default risk.

In the SVM, we perform the learning using various kernels including linear, RBF, polynomial, and sigmoid. We find that the linear and polynomial SVM perform relatively better than the other two in terms of accuracy in the test. The linear SVM achieves the highest F1 score and the largest area under the ROC curve, indicating a fair performance. Although with a lower level of accuracy, the RBF SVM may have more practical advantages because it produces less false negatives than the others.

In the Naïve Bayes, we use GaussianNB as classifier to estimate the model based on 15 features selected based on industry best practice with consideration of the observation numbers. We get an accuracy score of 0.66, which is similar to the performance of the SVM.

In the Random Forest, we perform the learning by using both all the features in the dataset and the most important 25 features. The performance of the model using only the important features is slightly better, but still unable to achieve a high level of accuracy.

Comparatively, the linear SVM and Random Forest using ket important features are able to predict the default risk more accurately. However, the performance in general can only be considered to be fair. To improve the learning performance, we think that the key is less related to algorithms but more about features. The currently available features still lack important information about a client's default risk. Issuing loans to clients with insufficient or non-existent credit histories remains to be a significant challenge for loan companies. However, we believe that analysis could be improved by using a larger dataset with higher data quality in the future.

7. References

Cristianini, Nello, and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines: and Other Kernel-based Learning Methods*. Cambridge, U.K: Cambridge University Press.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Springer Series in Statistics Springer New York Inc., New York, NY, USA, (2001)

Kowalczyk, Alexandre. 2017. *Support Vector Machines Succinctly*. Morrisville, NC: Syncfusion, Inc.