

3252 – Big Data

German Credit Data: Predicting Loan Credit Risk using Apache Spark Machine Learning

Student: Isaac Aktam

Instructor: Katrin Shechtman

Table of Contents

1. Title. Page 1.
2. Table of Contents. Page 2.
3. Executive Summary. Page 3.
4. Introduction. Page 6.
5. Methodology. Page 7.
6. Preliminary findings. Page 12.
7. Models and the results. Page 15.
8. Recommendations. Page 19.
9. References. Page 19.

Code link:

<https://databricks-prod->

cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/5936482774525390/4446916431035602/5294232844401586/latest.html

Executive Summary

The purpose of this project was to build a classification Machine Learning in Scala using Apache Spark ML to predict whether a person is creditable or not creditable. We used a German Credit dataset:

- Type: Financial
- Year: 1994
- Number of instances: 1000
- Number of categorical features: 13
- Number of numerical features: 7
- Number of labels: 1 (700 not creditable, 300 creditable)

We used 6 classification models:

- Decision Trees
- Logistic Regression
- Random Forests
- LightGBM
- XGBoost
- Support Vector Machines

to determine Accuracy, Recall, and Precision for training and test sets before and after tuning. In the end, we determined that the Logistic Regression was the best model.

Categorical Features	Numerical Features
"balance" aka checking account; 1 : ... < 0 DM 2 : 0 <= ... < 200 DM 3 : ... >= 200 DM / salary assignments for at least 1 year 4 : no checking account	Amount

<p>"history" aka credit history</p> <p>0 : no credits taken/ all credits paid back duly</p> <p>1 : all credits at this bank paid back duly</p> <p>2 : existing credits paid back duly till now</p> <p>3 : delay in paying off in the past</p> <p>4 : critical account/ other credits existing (not at this bank)</p>	Age
<p>"purpose" aka purpose of a loan</p> <p>0 : car (new)</p> <p>1 : car (used)</p> <p>2 : furniture/equipment</p> <p>3 : radio/television</p> <p>4 : domestic appliances</p> <p>5 : repairs</p> <p>6 : education</p> <p>7 : (vacation - does not exist?)</p> <p>8 : retraining</p> <p>9 : business</p> <p>10 : others</p>	Duration
<p>"savings" aka savings in accounts/bounds in DM</p> <p>1 : ... < 100 DM</p> <p>2 : 100 <= ... < 500 DM</p> <p>3 : 500 <= ... < 1000 DM</p> <p>4 : .. >= 1000 DM</p> <p>5 : unknown/ no savings account</p>	instPercent
<p>"employment" aka present employment since</p> <p>1 : unemployed</p> <p>2 : ... < 1 year</p> <p>3 : 1 <= ... < 4 years</p> <p>4 : 4 <= ... < 7 years</p> <p>5 : .. >= 7 years</p>	residenceDuration
<p>"sexMarried"</p> <p>1 : male : divorced/separated</p> <p>2 : female : divorced/separated/married</p> <p>3 : male : single</p> <p>4 : male : married/widowed</p> <p>5 : female : single</p>	credits
<p>"guarantors"</p> <p>1 : none</p> <p>2 : co-applicant</p> <p>3 : guarantor</p>	dependents
<p>"assets"; QUALITATIVE</p> <p>1 : real estate</p> <p>2 : if not 1 : building society savings agreement/ life insurance</p>	

3 : if not 1/2 : car or other 4 : unknown / no property	
"concCredit" aka other installment plans 1 : bank 2 : stores 3 : none	
"apartment" 1 : rent 2 : own 3 : for free	
"occupation" 1 : unemployed/ unskilled - non-resident 2 : unskilled - resident 3 : skilled employee / official 4 : management/ self-employed/highly qualified employee/ officer	
"hasPhone" 1 : none 2 : yes, registered under the customers name	
"foreign" aka foreign worker 1 : yes 2 : no	

Introduction

Imagine you are a financial institution that loans funds to clients. As a lender, you care whether a client is eligible for a loan given client's socio-demographic features. Given that, you face two risks:

1. If applicant has a good credit risk (more likely to repay a loan), then not providing him or her with a loan may result in a loss of business to the bank.
2. If the applicant has a bad risk (less likely to repay a loan), then providing him or her with a loan may result in a financial loss to the bank.

Given that, it is in your interests to provide as much business as possible while minimizing the loss to the bank. Of course, being a financial institution, you would have security measures in place to deal with non-credible clients. Therefore, let us cover the Precision and Recall. First, we need to build a Confusion Matrix:

```

===== Confusion matrix =====
#####| Actual = 0                Actual = 1
-----+-----
Predicted = 0| True_Negative      False_Negative
Predicted = 1| False_Positive     True_Positive
=====

```

Next, let us take a look at Precision and Recall functions:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \qquad Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

At last, we can proceed. So, as a financial institution, it is in our interests to provide as much business as we can. This means we would also want to provide business to non-credible clients to some degree by ensuring that we have necessary credit default instruments in place. Therefore:

- We want higher False Positives and lower False Negatives
- We want Low Precision and High Recall

Methodology

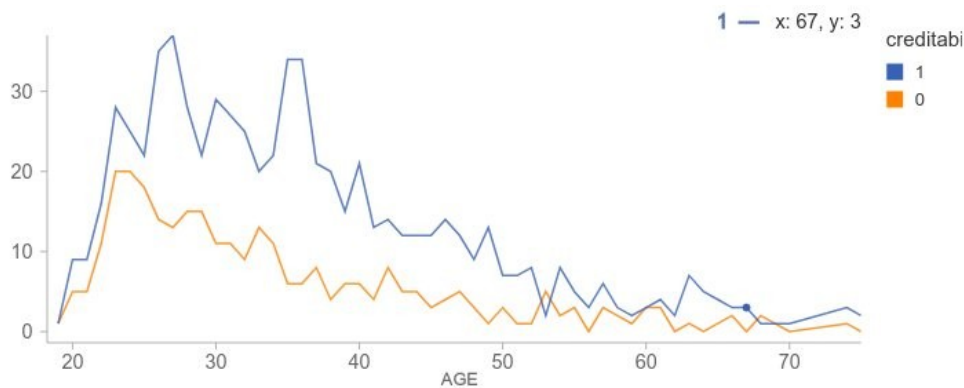
We divided coding into four parts:

1. Exploratory Analysis
2. Feature Engineering
3. Model Building and Preliminary Results
4. Results

Exploratory analysis deals with understanding the data by graphing the data set and making the necessary conclusions. For example, what are the average loan duration, average age, and average loan amount per creditability class?

creditability	avg(duration)	avg(age)	avg(amount)
1	19.207142857	36.22	2985.4428571429
0	24.86	33.96	3938.1266666667

Or, what's the average age per creditability?

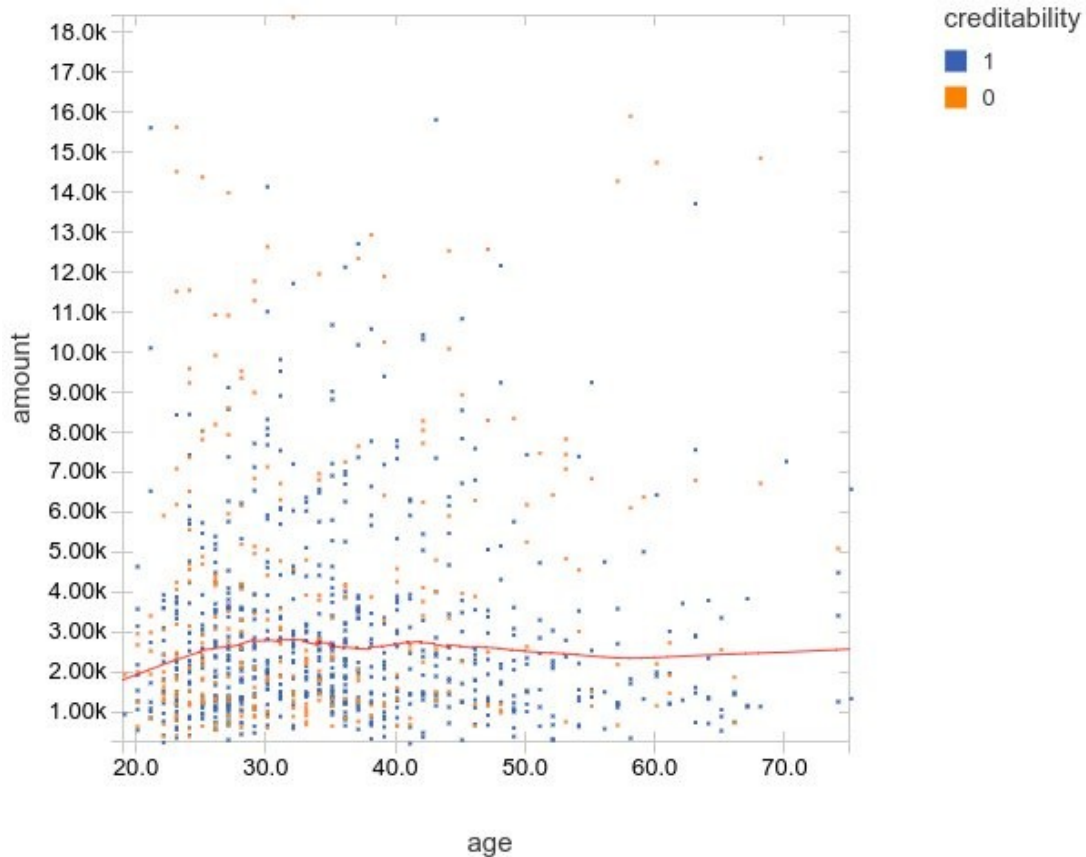


As we can see
from above,

- For creditable group, average age is higher than for the non-creditable group; and, it peaks at 27 and 35-36.
- For non-creditable group, the average peaks at 23-24.

Lastly, what is the relationship between Loan Amount and Age?

As we can see from above, there is a clear clustering 1,000 DM to 3'000 DM for the ages



between 20 and 40 both creditable and non-creditable classes.


Feature Engineering involves creating interaction variable for non-categorical features and applying one-hot encoder on categorical features.

Interaction features:

- Amount per Age
- Amount per Duration
- Duration per Age

The purpose of one-hot encoder was to turn nominal categorical features into features that have a value of 0 or

1. Let's take a look at the example below:

apartment		rent	own	free
rent		1	0	0
rent		1	0	0
own		0	1	0
free		0	0	1
free		0	0	1

For the above purpose and understanding, we turned nominal values of “1” to “Rent”, “2” to “Own”, and “3” to “Free”. As we can see, one-hot encoding deals with nominal values by turning them into features with values 0 or 1.

Model Building and Preliminary Results. Next:

- We created an Assembler in order to generate a vector of “features” for further analysis.
- We created a Feature Pipeline that we used to fit creditDF dataframe and transform the creditDF dataframe in order to produce dataframe with interaction variables and one-hot encoder.
- We split the resulting dataframe into 70% training set and 30% test.
- Next, we applied a **Standard Scaler** on training and test sets by making their features have mean 0 and variance 1. This ensures that such features as Amount, Age, or Duration do not have a dominant effect on the models and pull the results towards themselves. Standard Scaler ensures that features have “equal” effect on the model results.
- For the evaluator, we used a BinaryClassificationEvaluator with Area Under the Curve (AUC) as a metric. AUC measures the area under the Receiver Operator Curve (ROC) by plotting True Positive Rate against the False Positive Rate.
- On to the actual models. We used 6 classification models:
 - Decision Trees
 - Logistic Regression

- Random Forests
- LightGBM
- XGBoost
- Support Vector Machines
- For each of the 6 models, we calculated **Preliminary Results** such as Accuracy, Precision, Recall, and Confusion Matrix on training and test sets before and after tuning.
- To tune the models, we used the following:
 - ParameterGridBuilder
 - Pipeline
 - CrossValidator with 5 folds
 - Interesting finding. If our Parameter Grid is too sparse both in terms of number of hyper-parameters and values for those hyperparameters, DataBricks crashes by having cluster disconnected which results in a need to run the model again. To solve this, one would either need to buy a subscription or use PySpark or any other language that can be run locally.

Results. To display the results, we created a dataframe of :

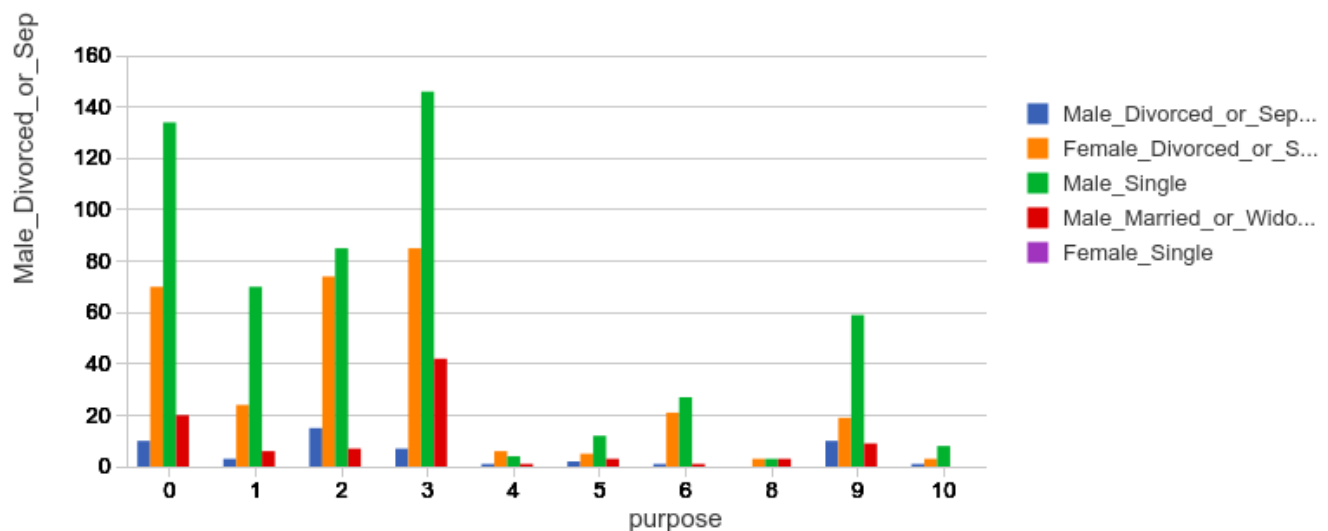
- Accuracy values for training and test sets before tuning
- Accuracy values for training and test sets after tuning
- Accuracy values for training set before and after tuning
- Accuracy values for test set before and after tuning
- Precision values for training and test sets before tuning
- Precision values for training and test sets after tuning
- Recall values for training and test sets before tuning
- Recall values for training and test sets after tuning

- Precision values for training and tests sets before and after tuning
- Recall values for training and tests sets before and after tuning
- Precision and recall values for training and tests before and after tuning

Preliminary Findings

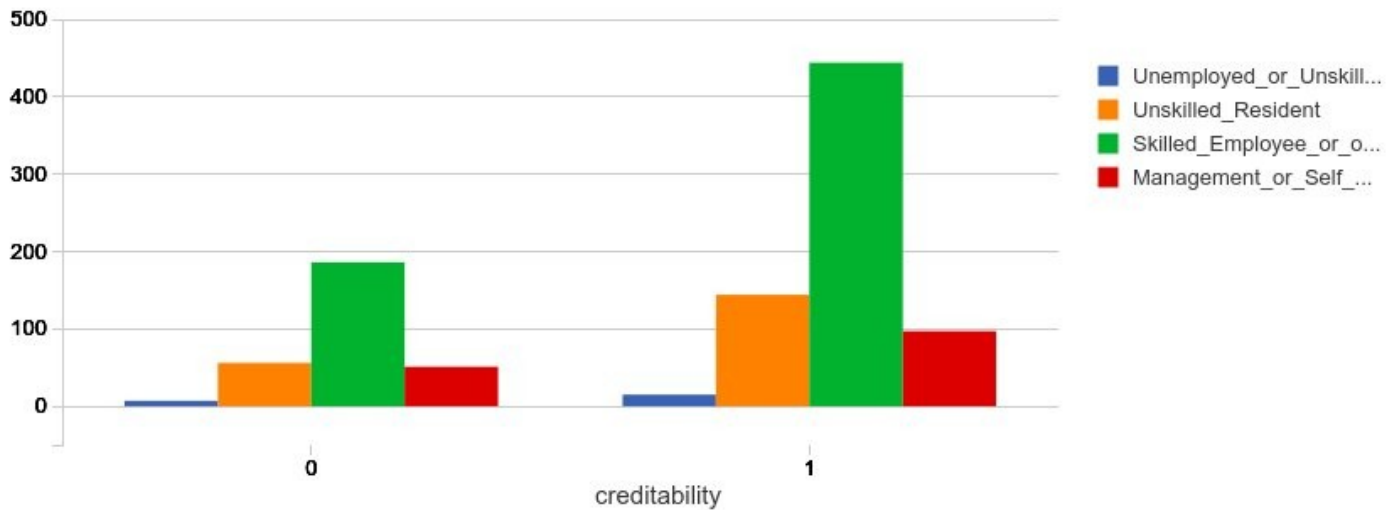
To make any financial decisions and properly meet client's expectations, organization must understand its data in terms of customer behaviour and its socio-demographic features. We need to remember that data on its own is meaningless. For this purpose, we need to prepare a graphycal analysis of the data in order to make the necessary inferences and conclusion that could help financial instutions make necessary decisions.

Example 1, let's take a look at the graph that plots SexMarried against the Purpose of a loan. This would help us understand for what purposes males or females of various marital statuses use the borrowed funds:



- 146 of single males borrow funds in order to purchase radio/television
- 134/70 of single males borrow funds in order to buy a new/old car respectively
- 85 of single males borrow funds to buy furniture/equipment
- 85 of divorced or separated females borrow funds to purchase radio/television
- 74 of divorced or separated females borrow funds to purchase furniture equipment
- And, 70 of divorced or separated females forrow funds in order to purchase a new car

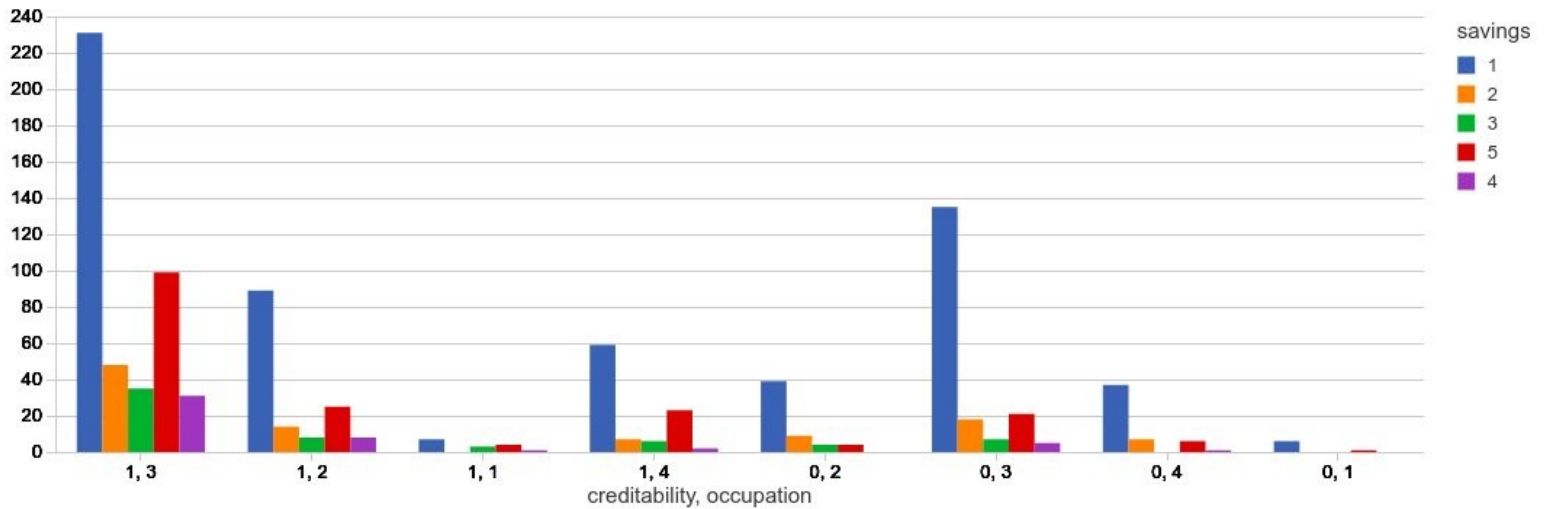
Example 2, let's take a look at the graph which displays the count of occupation type against the creditability in order to understand whether people who get the loan employed or not:



- a prevalent number (444) of borrowers are those who are those who are Skilled Employees or Officers. Interestingly, a prevalent number (186) of those who did not qualify for a loan are also Skilled Employees or Officers, perhaps they don't make as much to qualify for a loan.
- Interestingly, 144 of Unskilled Residents qualified for a loan whereas 56 of them did not.

Example 3, continuing from above, let's analyze the graph that plots the count of savings that a client has against the creditability and occupation. Please note that I was not able to come up with the necessary query to display category names on the graph and therefore we will need to resort to using the reference in regards to the categories:

	Occupation	Savings
1	unemployed/ unskilled - non-resident	... < 100 DM
2	unskilled - resident	100 <= ... < 500 DM
3	skilled employee / official	500 <= ... < 1000 DM
4	management/ self-employed/highly qualified employee/ officer	... >= 1000 DM
5		5 : unknown/ no savings account



- 231 clients who are creditable are skilled employees or officers and have less than 100 DM in savings
- 99 clients are creditable are skilled employees or officers who do not have any savings
- In regards to unskilled residents,
 - 89 of them are creditable and have less than 100 DM in savings. Interesting how they were qualified for a loan.
 - 39 of them are not creditable have less than 100 DM in savings.
- 135 clients who are skilled employees or officers are not qualified for a loan and have less than 100 DM in savings.

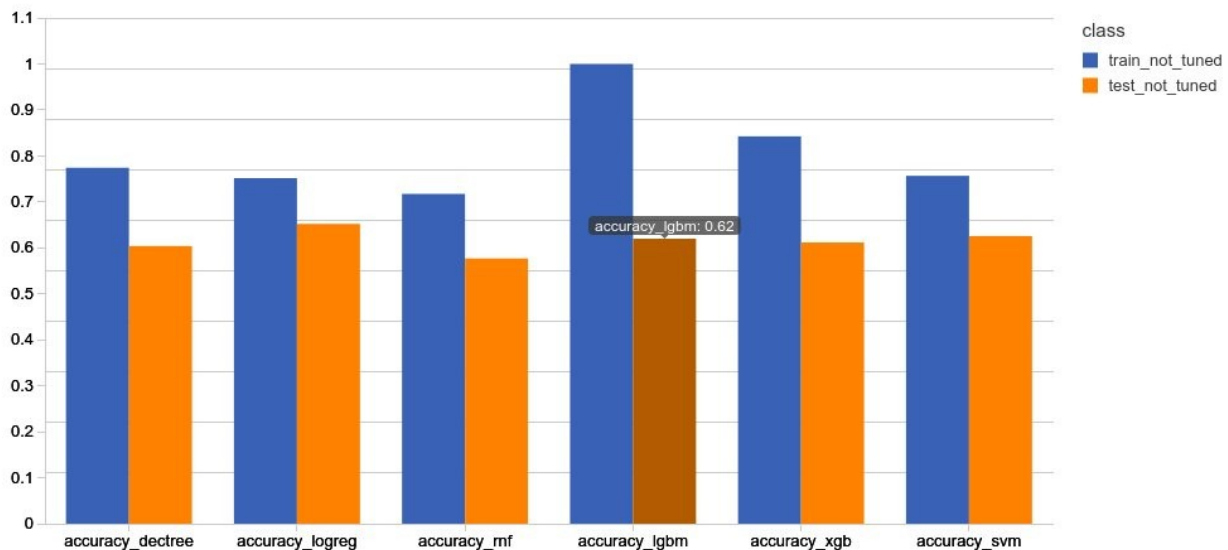
Models and Results

In order to classify clients as creditable or not creditable, we built 6 classification models:

- Decision Trees
- Logistic Regression
- Random Forests
- LightGBM
- XGBoost
- Linear SVM

and determined accuracy, precision, recall results and confusing matrix for before and after tuning.

Let's take a look at accuracy results for both training and test before tuning:

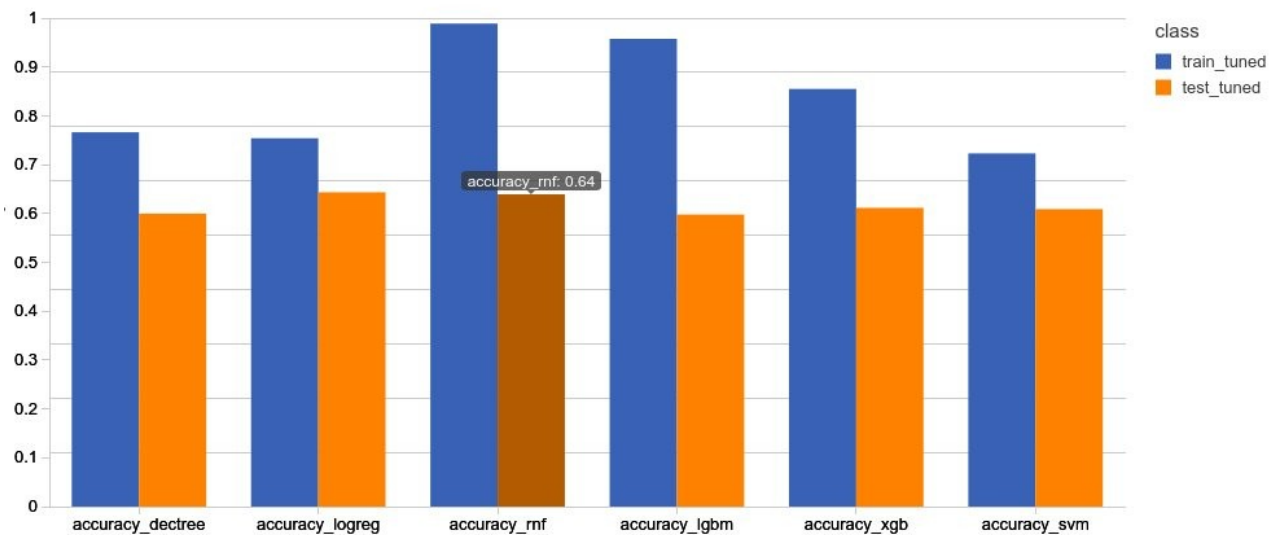


"model name, no tune"	train_not_tuned	test_not_tuned	Difference
accuracy_dectree	0.7742095703	0.6039535686	0.170256
accuracy_logreg	0.7516650584	0.6520514883	0.09961357
accuracy_mf	0.7173610717	0.5769164464	0.14044463
accuracy_lgbm	1	0.6198712792	0.38012872
accuracy_xgb	0.8425337269	0.6116538329	0.23087989
accuracy_svm	0.7566791546	0.6256464774	0.13103268

Before we make any conclusions, let us define “Overfitting”. Overfitting means that the model generalize the training data too well and test (unseen) data not so well. In short, one is really good at what one knows and really bad at what one does not know when encountered with something new.

As we can see from above, the Logistic Regression has the best performance since it overfits the least out of all the 6 models.

Next, let’s take a look at accuracy for training and test after tuning.

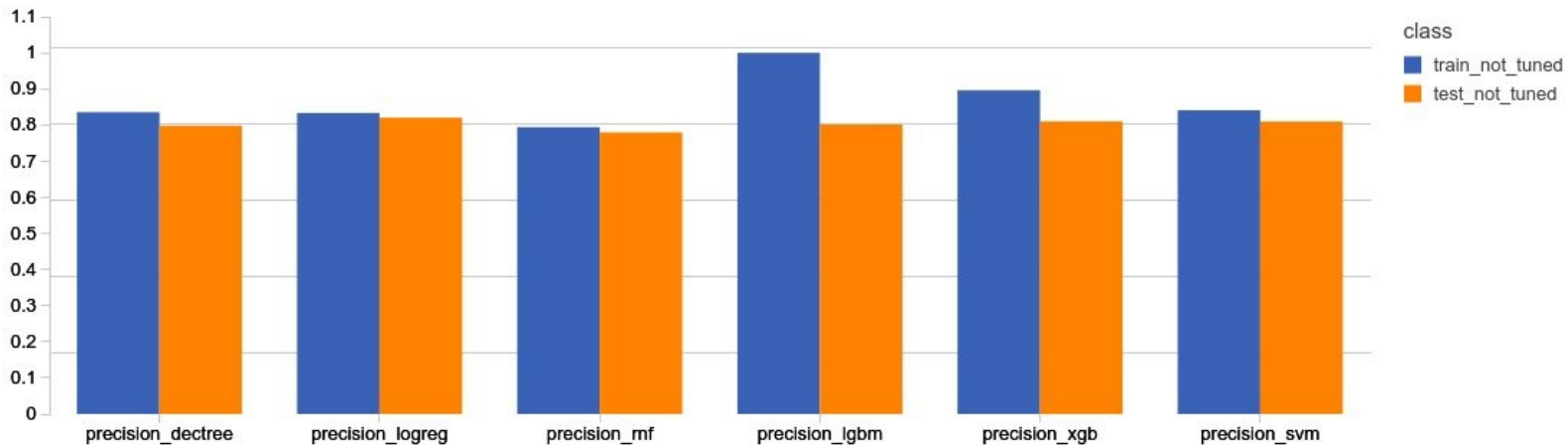


"model name, tune"	train_tuned	test_tuned	Difference
accuracy_dectree	0.7662958128	0.5998161131	0.1664796997
accuracy_logreg	0.7540396587	0.6433455925	0.1106940662
accuracy_rnf	0.9887892377	0.639208137	0.3495811007
accuracy_lgbm	0.957531551	0.597891047	0.359640504
accuracy_xgb	0.8549365196	0.6115101712	0.2434263484
accuracy_svm	0.7230421373	0.6089529939	0.1140891434

As we can see from above, Logistic Regression is the best model since it overfits the least. Interestingly, even after tuning, the performance for the Logistic Regression remains relatively the same.

Next, let’s analyze precision and recall. We need to remember that we want low precision and high recall since it is in the interests of the financial institutions to make loans to as many people as possible. Therefore, we need to have high False Positive (model predicted that a client is eligible for a loan but in reality is not) and low False Negatives (model predicted that a client is not eligible for a loan but in reality is).

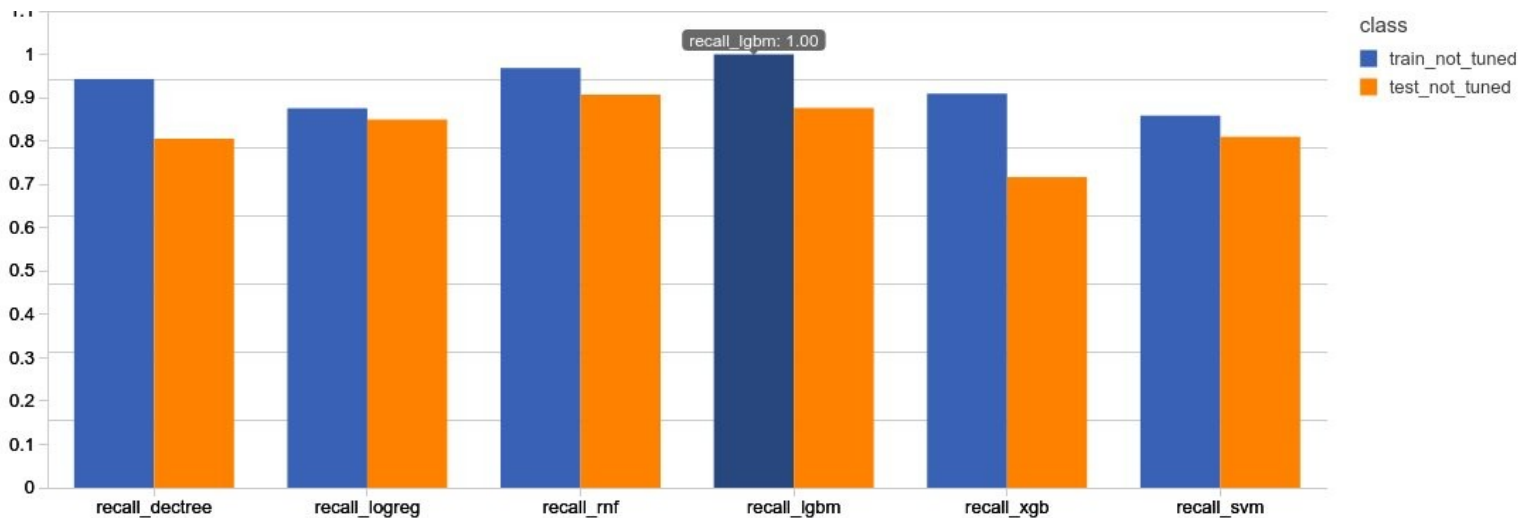
Precision for training and test sets before tuning.



"precision value, no tune"	"model name, no tune"	"class"	Precision diff
0.8355140187	precision_dectree	train_not_tuned	0.0372684047
0.8333333333	precision_logreg	train_not_tuned	0.0128205128
0.7941176471	precision_rnf	train_not_tuned	0.0146499665
1	precision_lgbm	train_not_tuned	0.1983805668
0.896049896	precision_xgb	train_not_tuned	0.086049896
0.8409090909	precision_svm	train_not_tuned	0.0311745776
0.798245614	precision_dectree	test_not_tuned	
0.8205128205	precision_logreg	test_not_tuned	
0.7794676806	precision_rnf	test_not_tuned	
0.8016194332	precision_lgbm	test_not_tuned	
0.81	precision_xgb	test_not_tuned	
0.8097345133	precision_svm	test_not_tuned	

From the above, we can see that Logistic Regression and Random Forests are closely matched. Logistic Regression has more consistent results with a difference of 0.0128 whereas Random Forest has lower precision values but a greater difference of 0.014. Therefore, both Logistic Regression and Random Forest are winning models in this case.

Recall for training and test sets before tuning (please see next page):



"recall value, no tune"	"mode name, no tune"	class	Recall diff
0.9430379747	recall_dectree	train_not_tuned	0.13772824
0.8755274262	recall_logreg	train_not_tuned	0.025969904
0.9683544304	recall_rnf	train_not_tuned	0.061274784
1	recall_lgbm	train_not_tuned	0.123893805
0.9092827004	recall_xgb	train_not_tuned	0.192468541
0.858649789	recall_svm	train_not_tuned	0.048915276
0.8053097345	recall_dectree	test_not_tuned	
0.8495575221	recall_logreg	test_not_tuned	
0.907079646	recall_rnf	test_not_tuned	
0.8761061947	recall_lgbm	test_not_tuned	
0.7168141593	recall_xgb	test_not_tuned	
0.8097345133	recall_svm	test_not_tuned	

As we can see from above, LightGBM has the highest Recall of 1 on the training set but much lower recall of 0.87 on the test set. Random Forest has the next highest Recall of 0.96 on the training set and highest Recall of 0.90 on the test set. Logistic Regression has lower Recall both for training and test sets with 0.87 and 0.84 respectively but with a smaller difference 0.03 between the two resulting the most consistent results out of all 6 models. Therefore, both Random Forest and Logistic Regression are contenders for the first place.

From the above analysis, we see that the Logistic Regression is a clear winner.

Recommendations

To make models more robust, we need to have larger dataset. As seen above, 1000 instances is simply not enough to build a robust model, avoid overfitting, and have high accuracy scores. Additionally, to better tune the models using DataBricks, it is evident that one would need to buy the license as a community edition is simply not enough to build a robust Parameter Space for Cross Validation as a sparse Parameter Space crashes the cluster. Lastly, better feature engineering is required to achieve better results.

References

1. German Credit Data Set, Clean:
 - <https://github.com/caroljmcDonald/spark-ml-randomforest-creditrisk/tree/master/data>
2. UCI Machine Learning Repository: Statlog (German Credit Data) Data Set:
 - [https://archive.ics.uci.edu/ml/datasets/Statlog+\(German+Credit+Data\)](https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data))
3. Predicting Loan Credit Risk using Apache Spark Machine Learning Random Forests:
 - <https://mapr.com/blog/predicting-loan-credit-risk-using-apache-spark-machine-learning-random-forests/>
4. PennState Eberly College of Science: Analysis of German Credit Data:
 - <https://onlinecourses.science.psu.edu/stat857/node/215/>