# Credit risk data analysis

Kevin Attila Hartyányi[C0S0RJ]

Eötvös Loránd University, Budapest, Egyetem tér 1-3, 1053, Hungary

**Abstract.** Classification and clustering on customer credit risk data.

**Keywords:** SVM · PCA · KMeans · Classification · Clustering · Frequent Itemsets.

## 1 Introduction

In this paper I'm going to describe how I solved the assignment for the Introduction to Data Science Lecture. We will go step-by-step on the process I used to analyze, classify and cluster the data, which is about customer credit risks, and I will show the results along the way with various plots to gain more insight.

## 2 Understanding the data

As a first step we need to understand our data. This is probably the most important step as it could give us a lot of information that we can use in later phases to improve our results and save time.

The data contains categorical and numerical column. Let's take a look at the numerical data first.

### 2.1 Numerical data analysis

We first check the data distribution of the numerical values. The plotted histograms can be seen on Figure 1. Here we can see that some of the histograms share a similar distribution, for example X05 and X13 or X08 and X11.

We can also study the correlation matrix of these values as seen on Figure 2. We are mainly interested in the Y column since that contains the information if a customer is good or bad, however even X02 which has the highest correlation with it's 0.214 value is not considered to be high.

### 2.2 Categorical data analysis

Now we take a look at the categorical data. We are interested in using the categorical data to find frequent itemsets in the dataset. To achieve this we use the apriori [7] algorithm with a minimum support of 0.7. The results can be seen on Figure 3. We see that almost every customer is a foreign worker in the dataset (A201). Also, 70% of the customers are labeled as good, we need to take this into account during the training as the models might have some bias towards predicting good customers, given that the data contains a lot more examples for them.

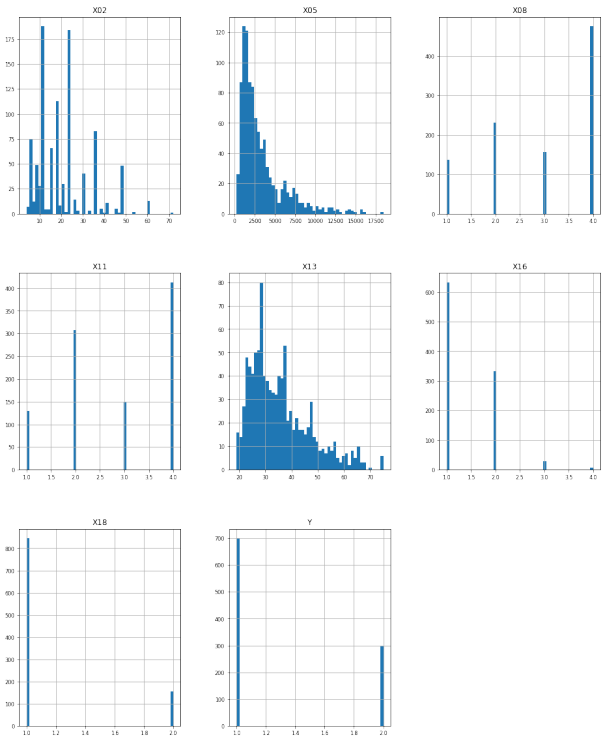



Fig. 1: Plots showing the distributions of the columns containing numerical data. We can see that some columns share a similar distribution.

## 3 Data Preprocessing

Now that we understand the data we need to prepare it for the models.

### 3.1 Normalization

Right now our numerical data have different scales which might end up creating a bias in the model, so we need to transform it to comparable scales that can prevent this problem. For this we can use sklearn's scale method [1], which center's the data to the mean and scales it to unit variance.

### 3.2 Encoding

We have dealt with the numerical columns of the data, but we still have the categorical columns left. We can't feed them to the models as they are, so we need to transform them. For this transformation we can use one hot encoding [2], which encodes our categories as one-hot numeric arrays.

The preprocessed data can be seen on Figure 4.

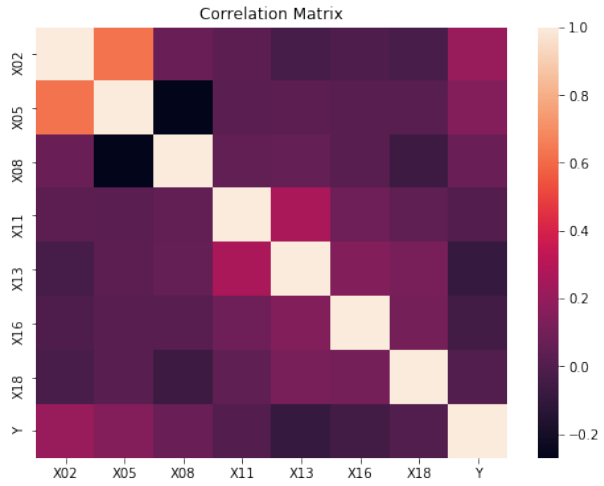



Fig. 2: Correlation matrix of the numerical data columns. We are mainly interested in the Y column as that contains the labels for the customers, it has the biggest correlation with the X02 column with a correlation number of 0.214.

| | items | support | ordered_statistics |
|---|---|---|---|
| **0** | (A101) | 0.907 | [(), (A101), 0.907, 1.0)] |
| **1** | (A143) | 0.814 | [(), (A143), 0.814, 1.0)] |
| **2** | (A152) | 0.713 | [(), (A152), 0.713, 1.0)] |
| **3** | (A201) | 0.963 | [(), (A201), 0.963, 1.0)] |
| **4** | (Good) | 0.700 | [(), (Good), 0.7, 1.0)] |
| **5** | (A101, A143) | 0.742 | [(), (A101, A143), 0.742, 1.0), ((A101), (A14... |
| **6** | (A101, A201) | 0.880 | [(), (A101, A201), 0.88, 1.0), ((A101), (A201... |
| **7** | (A201, A143) | 0.782 | [(), (A201, A143), 0.782, 1.0), ((A143), (A20... |
| **8** | (A101, A201, A143) | 0.718 | [(), (A101, A201, A143), 0.718, 1.0), ((A101)... |

Fig. 3: Results of the apriori algorithm on the categorical data columns with a minimum support of 0.7.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | ... | 1.0 | 1.0 | 0.0 | -1.236478 | -0.745131 | 0.918477 | 1.046987 | 2.766456 | 1.027079 | -0.428290 |
| **1** | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 1.0 | 0.0 | 2.248194 | 0.949817 | -0.870183 | -0.765977 | -1.191404 | -0.704926 | -0.428290 |
| **2** | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | ... | 0.0 | 1.0 | 0.0 | -0.738668 | -0.416562 | -0.870183 | 0.140505 | 1.183312 | -0.704926 | 2.334869 |
| **3** | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 1.0 | 0.0 | 1.750384 | 1.634247 | -0.870183 | 1.046987 | 0.831502 | -0.704926 | 2.334869 |
| **4** | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 1.0 | ... | 0.0 | 1.0 | 0.0 | 0.256953 | 0.566664 | 0.024147 | 1.046987 | 1.535122 | 1.027079 | 2.334869 |

Fig. 4: The customer credit data after the preprocessing. On the left side we see that the categorical columns have been transformed to one hot vectors, while on the right we see that the numerical columns have been normalized. It doesn't include the Y column that contains the labels, because that has to remain unchanged.

## 4 Classification

We first split the data into train and test set, then we can use support vector machines [3] for the classification of good/bad customers.

With this setup we get a score of **0.76363**. After doing cross-validation on this model with 10 folds the results show a mean of **0.763** and a standard deviation of **0.0494**.

Looking at the classification report 5a we can see that the model gets a high precision and recall for the good customer classification. However we have a constraint that we didn't take into account yet, that it is five times worse to classify customers as good when they are bad than it is to classify customers as bad when they are good.

To satisfy this constraint we need to improve the recall of class 2 (which represents the bad customers).

### 4.1 Grid Search

To improve the recall we can do a grid search where we specify a scoring based on the recall of class 2, so the search will find an optimal parameter combination that maximizes this recall.

We can see the results of the best model after the grid search on Figure 5b. There is also a comparison between the two models plotting the precision-recall curves on Figure 6. We see that the recall has improved by almost 10%, but we can do better than this.

```
[[138  17]
 [ 35  30]]
              precision    recall  f1-score   support

           1       0.80      0.89      0.84       155
           2       0.64      0.46      0.54        65

    accuracy                           0.76       220
   macro avg       0.72      0.68      0.69       220
weighted avg       0.75      0.76      0.75       220

Accuracy score:  0.7636363636363637
```

(a) Optimized for accuracy.

```
[[122  33]
 [ 29  36]]
              precision    recall  f1-score   support

           1       0.81      0.79      0.80       155
           2       0.52      0.55      0.54        65

    accuracy                           0.72       220
   macro avg       0.66      0.67      0.67       220
weighted avg       0.72      0.72      0.72       220

Accuracy score:  0.7181818181818181
```

(b) Optimized for class 2 recall.

Fig. 5: Classification report for the SVM model optimized with grid search. The recall optimized model improved the class 2 recall by 9% while losing 5% on average accuracy. In our case this is beneficial for us, given our constraint for the task.

### 4.2 Adjusting the probabilities of the model

Instead of retraining we can take the probabilities of the model that it predicts for each customer. When we have these probabilities we can adjust the amount that

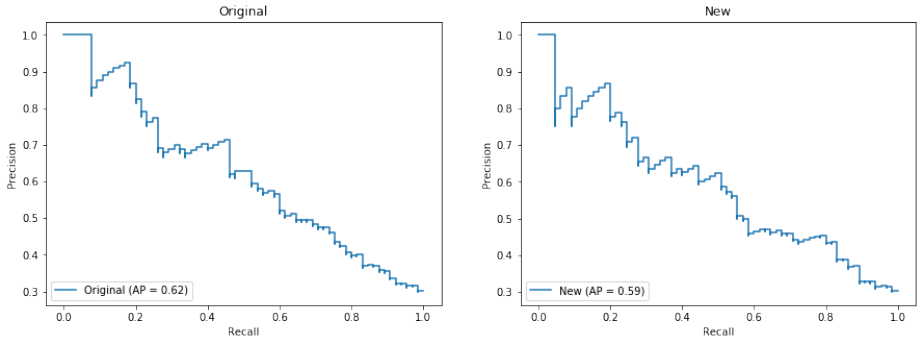



Fig. 6: Comparison of the precision-recall curves between the original (left) and grid search optimized for recall model (right). We see that the precision on the right side shrank in favour of a higher recall

we want for each class. In our case, we will adjust the probability of classifying a customer as good, so it requires a probability of at least 80%, if it's smaller than this, then we classify them as bad customers. The results of this adjusted model can be seen on Figure 7.

We can see that we improved the class 2 recall of the original model by 32% and the grid search model by 23%. However, we did lose more than 15% on average accuracy of the model, but given our constraint this model is more beneficial for us.

```
[[82 73]
 [14 51]]
              precision    recall  f1-score   support

           1       0.85      0.53      0.65       155
           2       0.41      0.78      0.54        65

    accuracy                           0.60       220
   macro avg       0.63      0.66      0.60       220
weighted avg       0.72      0.60      0.62       220

Accuracy score:  0.6045454545454545
```

Fig. 7: Classification report for the SVM model whose probabilities have been adjusted and requires at least 80% to classify a customer as good. The class 2 recall is greatly imrpoved compared to the previous SVM models.

## 5 Principal Component Analysis

Right now we can't easily plot our data, because it has a high dimensionality. To solve this problem we can use principal component analysis [4] to reduce the dimensionality of the data to 2 dimensions.

Now we can plot the data as seen on Figure 8. We don't see an easy way to separate the different labeled customers, but let's see if our model can find one.

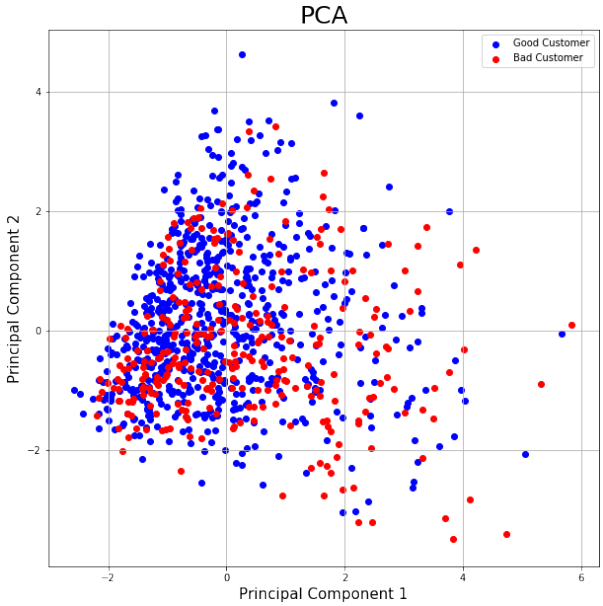
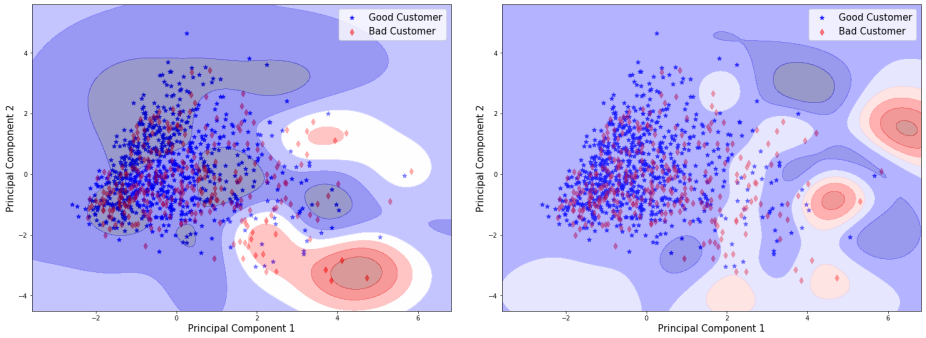



Fig. 8: Data after principal component analysis. We see that most of the points are cluttered around the left side of the plot and that we can't easily see a way to separate the differently labeled points.

### 5.1 SVM with PCA

Running our model with simple grid search (optimized for accuracy) on this data we get a score of **0.704** which seems nice, but after looking at the classification report Figure 10a we see that the model classified everyone as a good customer. Not really what we were looking for. We can also take a look at the plotted decision boundary of the model on Figure 9a where we see that the good customer examples overwhelm the decision of the model.

Changing the grid search optimization score to the class 2 recall we get an accuracy of **0.709**. The classification report for this model is on Figure 10b and the plot of the decision boundary is on Figure 9b. This achieves somewhat better results, but still far from what we have achieved on the original data.




(a) Optimized for accuracy. (b) Optimized for class 2 recall.

Fig. 9: Decision boundary of the svm model optimized with grid search on the dimensionality reduced data by pca. Optimizing for accuracy we see that the higher amount of good customer examples overwhelm the decision of the model predicting most customers as good. This effect can be reduced by optimizing for recall, but not to much extent.

```
[[155   0]
 [ 65   0]]
              precision    recall  f1-score   support

           1       0.70      1.00      0.83       155
           2       0.00      0.00      0.00        65

    accuracy                           0.70       220
   macro avg       0.35      0.50      0.41       220
weighted avg       0.50      0.70      0.58       220

Accuracy score:  0.7045454545454546
```

(a) Optimized for accuracy.

```
[[146   9]
 [ 55  10]]
              precision    recall  f1-score   support

           1       0.73      0.94      0.82       155
           2       0.53      0.15      0.24        65

    accuracy                           0.71       220
   macro avg       0.63      0.55      0.53       220
weighted avg       0.67      0.71      0.65       220

Accuracy score:  0.7090909090909091
```

(b) Optimized for class 2 recall.

Fig. 10: Classification report for the SVM model trained on the dimesnionality reduced data by pca and optimized with grid search. As discussed on Figure 9 the accuracy optimized model predicts all examples as good customers.

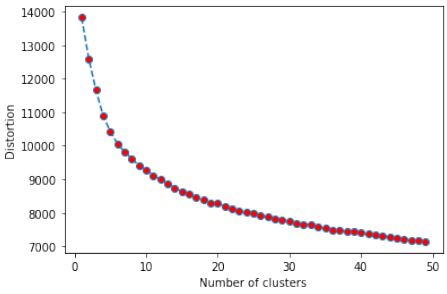
## 6 Clustering

### 6.1 KMeans

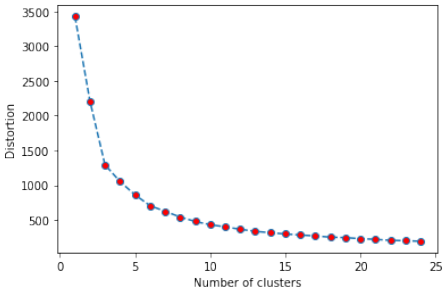
We can use KMeans for clustering the customers into various groups. Let's first see how well the model performs when we use a cluster number of 2 to classify the customers to be good/bad without the labels. The classification report Figure 12a showed an accuracy of **0.664**, which is not that bad given that it did all of this unsupervised.

If we didn't have the labels for the customers we wouldn't know how many clusters we should use, for this we can use the elbow method to estimate a good cluster number. A plot showing the elbow method on this data can be seen on Figure 11a. The y axis on this plot refers to the distortion which is the sum of

squared distances of samples to their closest cluster center. We don't see any plateau on this graph so it's not easy to choose a good cluster number.




(a) On the original data.




(b) On the dimensionality reduced data by pca.

Fig. 11: Elbow method using KMeans model. On the original data the curve keeps decreasing so we can't easily find a good cluster number, while on the dimensionality reduced data we see that it starts to plateau around 20 clusters.

### 6.2 KMeans with PCA

For better visualization we can try our KMeans model on the dimensionality reduced data. On this data we achieve an accuracy of **0.662** Figure 12b.

```
[[554 146]
 [190 110]]
              precision    recall  f1-score   support

           1       0.74      0.79      0.77       700
           2       0.43      0.37      0.40       300

    accuracy                           0.66      1000
   macro avg       0.59      0.58      0.58      1000
weighted avg       0.65      0.66      0.66      1000

Accuracy score:  0.664
```

(a) On the original data.

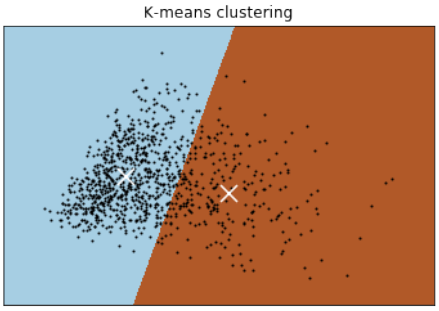
```
[[556 144]
 [194 106]]
              precision    recall  f1-score   support

           1       0.74      0.79      0.77       700
           2       0.42      0.35      0.39       300

    accuracy                           0.66      1000
   macro avg       0.58      0.57      0.58      1000
weighted avg       0.65      0.66      0.65      1000

Accuracy score:  0.662
```

(b) On the dimensionality reduced data by pca.

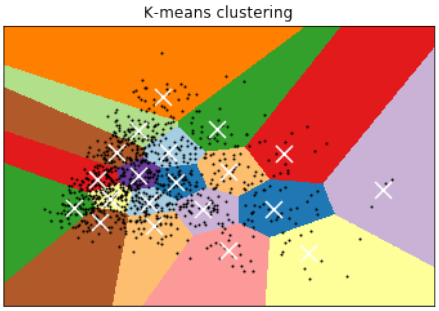
Fig. 12: Classification report for the KMeans model using 2 clusters. It receives similar results for the original and the dimesnionality reduced data.

A visualization of this clustering can be seen on Figure 13a.

Using the elbow method again on Figure 11b we see a nicer graph where the cluster number distortion starts to plateau around 20 clusters. The results of clustering with this number can be seen on Figure 13b.




(a) Using 2 clusters. Good/bad customers.




(b) Using 20 clusters. We use 20 clusters, because the elbow method on this data starts to plateau around this point.

Fig. 13: Visualization of the KMeans model with different clusters on the dimensionality reduced data by pca.

## 7 Conclusion

We have tried SVM, PCA and KMeans with different parameters on the customer credit data.

With the SVM we achieved an accuracy of **0.763** with a class 2 recall of **0.46**, which after taking the constraint of the task into account was reduced to **0.604**, but with the much better class 2 recall of **0.78**.

We used PCA for dimensionality reduction, so we could visualize the data. We also tried this data with SVM and KMeans, with the purpose of finding something interesting and to understand the data better.

For clustering the customers into various groups we used KMeans. We were able to estimate the accuracy of the clustering when we created 2 clusters from the data **0.662**, because we had the labels for them, however most of the time we don't have labels for our data, so we also checked other methods, like the elbow technique for deciding what is a nice cluster number for the data.

## References

1. Sklearn Scale, `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.scale.html`. Last accessed 4 Dec 2020
2. Sklearn OneHotEncoder, `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html`. Last accessed 4 Dec 2020
3. Sklearn SVM, `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`. Last accessed 4 Dec 2020
4. Sklearn PCA, `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`. Last accessed 4 Dec 2020

5. Sklearn KMeans, https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html. Last accessed 4 Dec 2020
6. PCA Boundary visualization, https://towardsdatascience.com/visualizing-support-vector-machine-decision-boundary-69e7591dacea. Last accessed 4 Dec 2020
7. Apriori algorithm implementation, https://pypi.org/project/apyori/. Last accessed 4 Dec 2020