Machine Learning
-
Supervised Learning

Brandon Alves

September 23, 2022

Contents

List of Figures

1 Introduction

In this article, I will present the results of various Supervised Learning methods applied on two classification problems. Those methods are:

- Decision Trees (with pruning);
- Neural Networks;
- Boosting;
- Support Vector Machines;
- K-Nearest Neighbors.

For each of them, I will discuss the results obtained, the pros and cons of using them, and the parameters that were used. I will also discuss the results when applying the methods on the two different datasets. At the end I will also compare the methods between them and highlight the best one for each dataset.

I will present many plots on this report to show the evolution of the accuracy depending on some parameters. Those plots will be presented in the following way : the hard line represents the median precision value for a given parameter value while the tint area goes from the minimum to the maximum precision value for the given parameter value.

The experiments on this report were performed on a computer with the following specifications:

- CPU : Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 6 cores, 12 threads
- RAM : 16GB DDR4
- GPU : NVIDIA Quadro P620

2 Datasets

In this article, I will learn over two datasets:

- Gender Classification;
- Credit-Card Fraud Detection.

Gender Classification

This dataset describes some appearence characteritics. It contains 7 features and one output. The features are: long hair, width of the forehead, height of the forehead, wide nose, long nose, thin lips and the distance between nose and lips. The output feature is gender and takes value 1 for male and 0 for female. That dataset is not the most complicated. I thought it would be a good idea to start with a simple dataset to get a better understanding of the methods. It is also a good idea to start with a dataset that is not too big to avoid long training times.

Credit-Card Fraud Detection

This dataset is a credit card fraud detection dataset. The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. It contains only numerical input variables which are the result of a PCA transformation. The only features which have not been transformed with PCA are *Time* and *Amount*. Feature *Time* contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature *Amount* is the transaction amount. Feature *Class* is the response variable and it takes value 1 in case of fraud and 0 otherwise. I chose that dataset because, unlike the previous one, it is a very unbalanced dataset with only a few positive outputs. I wanted to see how the methods would behave on such a dataset.

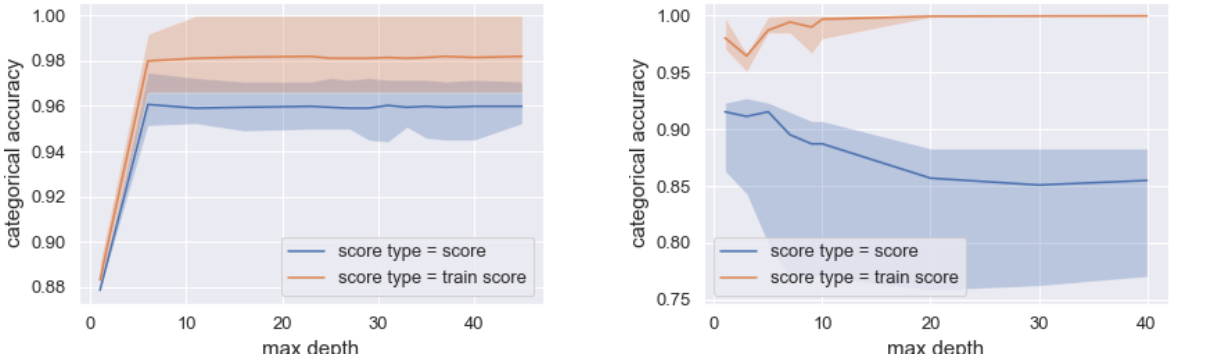
Normalisation

Some methods require the data to be scaled between 0 and 1. It also permits computation to be faster. Both the training and the testing datasets are scaled between 0 and 1. The scaling is done by dividing each value by the maximum value of the dataset. The scaling is done on the training dataset and then applied on the testing dataset.

3 Decision Trees

In that section I will present the results obtained when using Decision Trees with pruning. We will here discuss the influence of different parameters :

- The maximum depth of the tree;
- The minimum number of samples required to split an internal node;
- The size of the training set.



(a) Accuracy for train and test Gender datasets

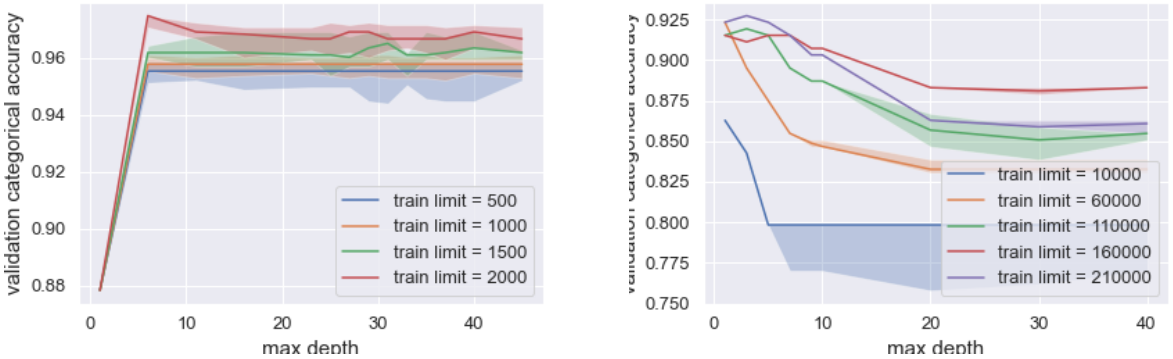
(b) Accuracy for train and test Credit-Card datasets

Figure 1: Accuracy of the Decision Tree Classifier according to the Maximum Depth of the Tree

We begin by finding the limit of our classifier. That limit can be of two natures : overfiting or a flat level on both training and test accuracy. Figure 1 represents those limits. On figure 1b, we can see that the Decision Tree method quickly overfits. Indeed the training accuracy quickly rise above 99% while the test accuracy decreases to reach around 0.87%. We can notice that the best score is obtained for a maximum depth value of 5. Also the score considerably decreases for a maximum depth value bigger than 5. Therefore we can conclude that only 5 out of the 28 dimensions of the input space is relevant for the classification output.

On figure 1a, we can notice that the classifier does not overfit. It is interesting to constat that the score for the testing dataset happens to be better than the score for the training dataset. This is probably due to the fact that the dataset is not too big. We can also notice that the best score is obtained for a maximum depth value of 7 which means that all the dimensions of the input space are relevant for the classification output.

Let's discuss now the influence of the size of the training set on the accuracy of the classifier. Figure 2 represents the accuracy of the classifier according to the size of the training set for both datasets. As expected the accuracy of the classifier increases when the size of the training set increases. The accuracy of the classifier is also impacted by the maximum depth of the tree. Indeed, the score for all the different sizes of the training set is almost the same when the maximum depth is around 1, whereas the score differs a lot when the maximum depth of the tree increases. We can also notice that the training set with 160 000 samples gives better results than the trainingset with 210 000 samples in figure 2b. This is probably due to the fact that the training set with 160 000 samples is less noisy than the training set with 210 000 samples, meaning that when adding those 50 000 samples, we probably introduce some noise in the training set.

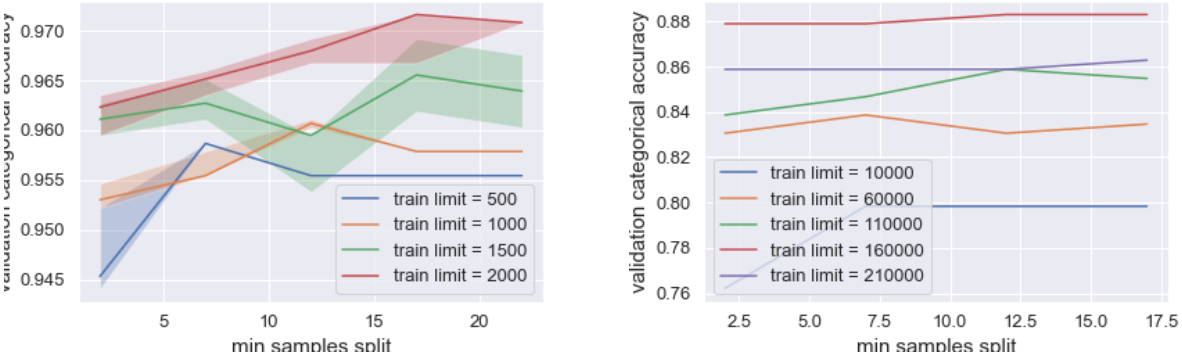


(a) Accuracy for train and test Gender datasets

(b) Accuracy for train and test Credit-Card datasets

Figure 2: Accuracy of the Decision Tree Classifier according to the Size of the Training Tet

On the Gender dataset, we can see on figure 2a that the accuracy of the classifier is also impacted by the size of the training set. The best score of the classifier is for a training set of 2 000 samples. We can also notice that the score is almost the same for a maximum depth between 1 and 5. This means that the training set size as more inpact on the classifier when the maximum depth of the tree is high.



(a) Accuracy for train and test Gender datasets

(b) Accuracy for train and test Credit-Card datasets

Figure 3: Accuracy of the Decision Tree Classifier according to Minimum Number of Samples required to Split a Node

The last parameter I will discuss is the minimum number of samples required to split an internal node. That variation is displayed on figure 3. In the Gender dataset, we see that this parameter has some influence. First of all we can see that this parameter lead to overfiting. In the case of the training set composed of 2 000 samples, the classifier overfits for a minimum sample split bigger than 17. But it also improves the results for all the training set sizes, when the minimum sample split is not too high.

In the case of the Credit-Card dataset, we can see that the accuracy of the classifier is not really impacted by the minimum number of samples required to split an internal node except for the smaller datatset. Which have sense. Indeed, the smaller dataset is the one with the biggest chance of improvment.

Decision Tree is the faster method compared with all the other ones. To get those results and generate the data associated with, it took around 10 minutes.

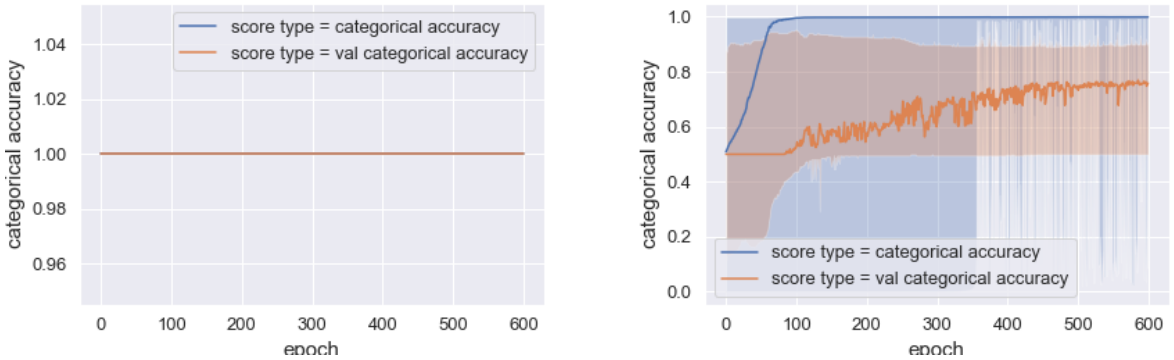
The decision tree classifier performs pretty well as seen above. But that method does not perform very well on highly dimensional datasets. To resolve that problem, Random Forrest can be used. Random Forrest is a method

that uses multiple decision trees to classify the data. The method is based on the idea that a large number of relatively uncorrelated classifiers (trees) operating as a committee will outperform any of the individual classifiers. That method is also known as bagging.

4 Neural Networks

Neural Network model is the most complicated in terms of parameters in comparison with the other models. We will discuss the influence of the followings:

- The number of epochs;
- The use of batch normalization;
- The number of layers and the number of neurons per layer;
- The activation function;
- The size of the training set.



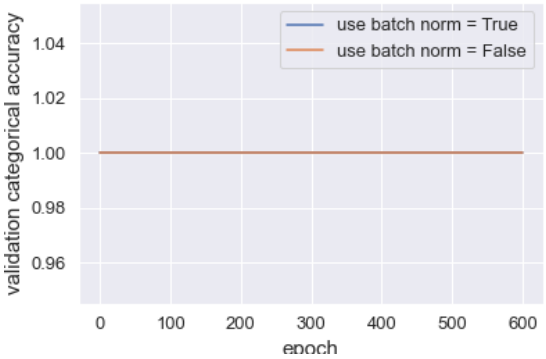
(a) Accuracy for the train and test Gender dataset (b) Accuracy for the train and test Credit-Card dataset

Figure 4: Accuracy of the Neural Network Classifier according to the Number of Epochs

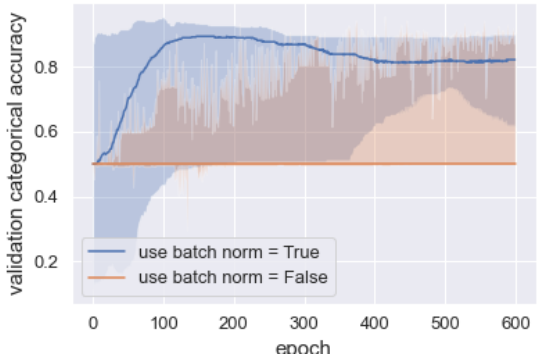
The Neural Network classifier has a perfect scores on the Gender dataset. That dataset is very easy to learn over. The results are more interesting on the Credit-Card dataset. We can see on figure 4b that the score of the classifier goes up with the number of epochs which seems totally fair. We notice that the classifier does not overfit the training set.

The parameter that affects the most the accuracy of the calssifier is the use of a batch normalization as we can see on figure 5. We know that the use of a batch normalization permits to decrease the number of training epochs required to train a neural network. We can see on figure 5b that not using batch normalisation does not even permit to reach a score of 0.5. The use of batch normalization is mandatory to get a good score on the Credit-Card dataset.

Let's discuss the impact of the activation function. The impact of the activation function is not as important as the batch normalization. We can see on figure 6 that the accuracy is not very different between the different activation functions. On the Credit-Card dataset, on figure 6b, the activation function that gets the worse results is the *sigmoid*.

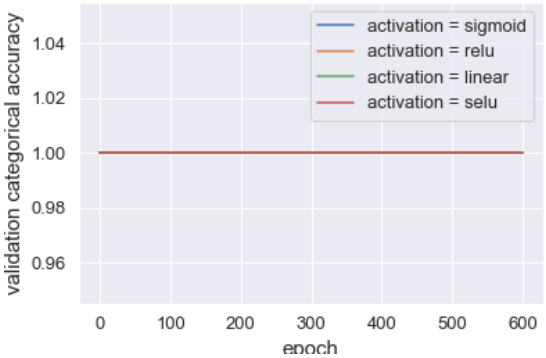


(a) Accuracy with and without batch normalization for Gender dataset

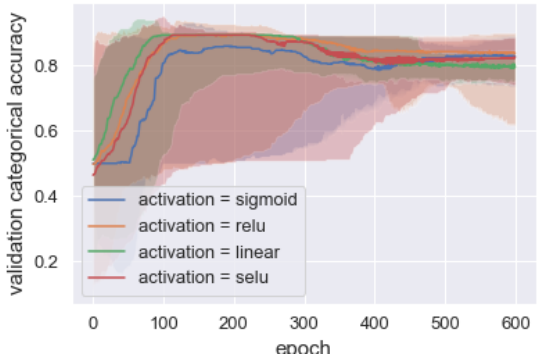


(b) Accuracy with and without batch normalization for Credit-Card dataset

Figure 5: Accuracy of the Neural Network Classifier according to the use of Batch Normalization



(a) Accuracy for different activation functions on Gender dataset



(b) Accuracy for different activation functions on Credit-Card dataset

Figure 6: Accuracy of the Neural Network Classifier according to the Activation Function

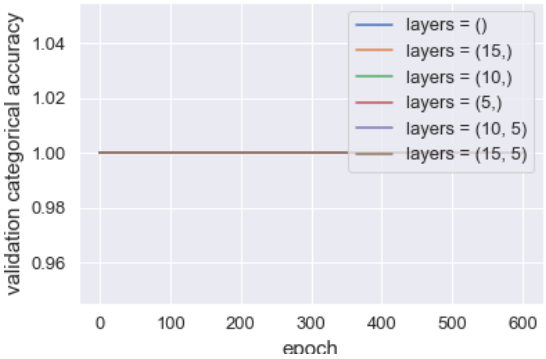
Another parameter that affected the score of the classifier is the number of layers and the number of nodes they contains. We can see on figure 7 that the accuracy depends on the layers. For the Gender dataset, results are the same for both layers we trained our network with. In fact not using layers (output directly connected to input) still gives a 100% fit as we can see on figure 7a.

On figure 7b we can see that the choose of layers does not have a big impact on the accuracy. The best results are obtained with 2 layers. The number of nodes in the layers does not have a big impact on the accuracy. The best results are obtained with 10 nodes in the first layer and 5 nodes in the second layer. But the results are not very different with only one layer with 5 nodes in it.

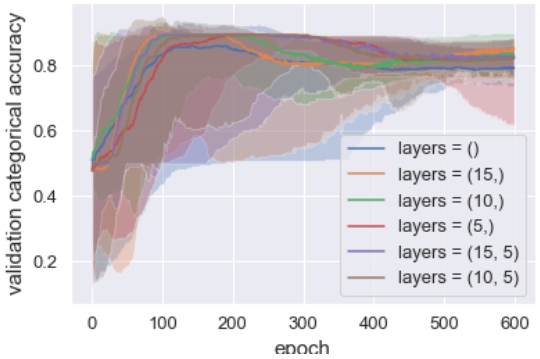
One interesting thing in that case is that the training size does not seem to have a big impact on the classification score. We can see on figure 8b that the accuracy is not very different between the different training sizes. For example we have pretty much the same accuracy for a 1 000 and a 160 000 training size.

In terms of computation time, our Neural Network classifier is the third slowest. In addition, the memory used in the learning process was also very high. It took 3h30m to train the neural network on both datasets.

I saw that, on the Credit-Card dataset, it is not easy to find a good tuning for the neural network for all these parameters to obtain good results.

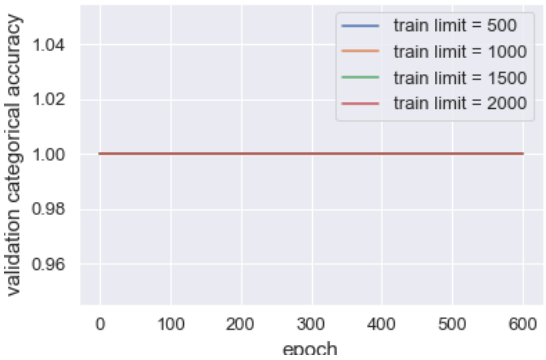


(a) Accuracy for different layers on Gender dataset

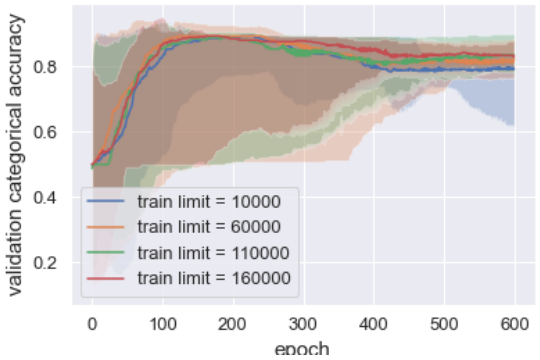


(b) Accuracy for different layers on Credit-Card dataset

Figure 7: Accuracy of the Neural Network Classifier according to the Number of Layers



(a) Accuracy for different training dataset sizes on Gender dataset



(b) Accuracy for different training dataset sizes on Credit-Card dataset

Figure 8: Accuracy of the Neural Network Classifier according to the Training Dataset Size

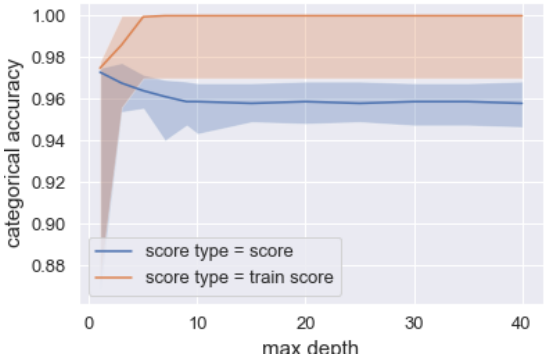
5 Boosting

Let's now discuss the influence of the followings parameters on the accuracy of the Boosting classifier:

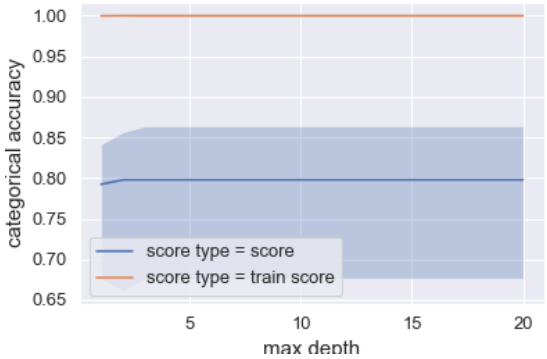
- The number of boosting stages to perform;
- The maximum depth of an individual regression estimator;
- The minimum number of sample needed to split a node of an estimator;
- The size of the training set.

First, we can see that the accuracy of the Boosting classifier is very good on the Gender dataset and pretty good on the Credit-Card dataset. On figure 9, we can see the accuracy of both classifier according to the maximum depth of an individual regression estimator. For the Gender dataset, it is interesting to notice that the score of the classifier is the best when the maximum depth is 1. That means that each tree of the Boosting classifier is a simple node. The Gender dataset is probably too simple to benefits from higher maximum depth values.

For the Credit-Card dataset, the accuracy is the best when the maximum depth is higher to 2. We also notice that the classifier does not overfit. Unlike in the case of the Gender dataset, the Credit-Card dataset benefits from higher maximum depth values.

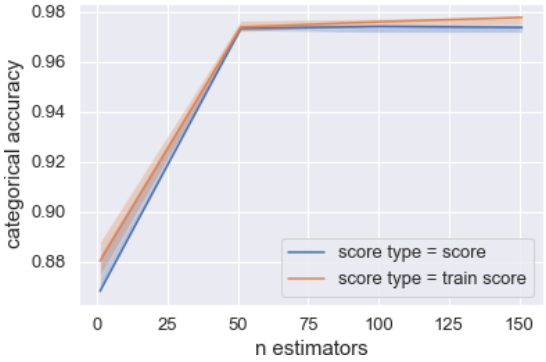


(a) Accuracy for the train and test Gender dataset

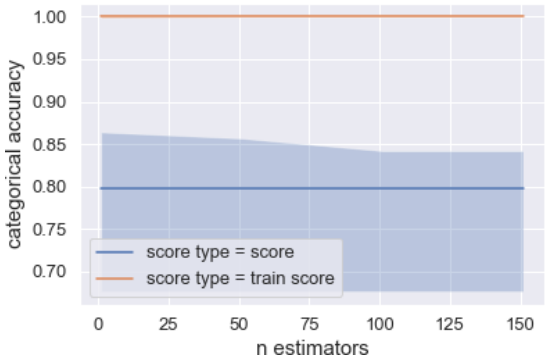


(b) Accuracy for the train and test Credit-Card dataset

Figure 9: Accuracy of the Boosting Classifier according to the Maximum Depth of an Estimator



(a) Accuracy for the train and test Gender dataset, for $maxdepth = 1$



(b) Accuracy for the train and test Credit-Card dataset, for $maxdepth = 3$

Figure 10: Accuracy of the Boosting Classifier according to the Number of Estimator used for a given maximum depth

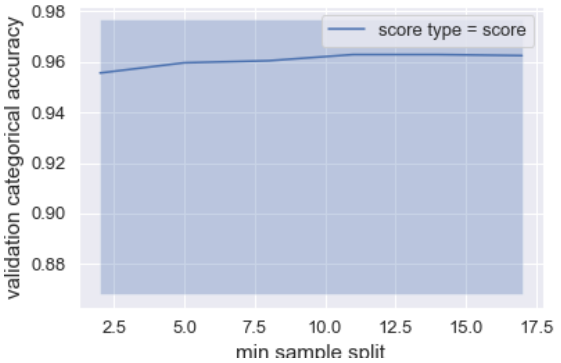
Let's now discuss the impluence of the number of estimators. Results are presented on figure 10. In the case of the Gender dataset, we can see that the accuracy is the higher for a number of estimators bigger than 50. This is interesting because we saw that the best fit for the Gender dataset was trees with only one node. So, it seems that the Boosting classifier is able to combine a lot of simple trees to obtain a good accuracy.

For the Credit-Card dataset, presented on figure 10b, we can see that the score is not dependent on the number of estimators, which means that a single Decision Tree was enough to obtain a good accuracy.

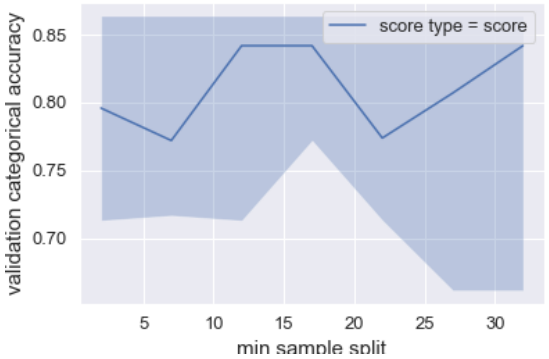
I present on figure 11 the influence of the parameter that set the minimum number of samples required to split an internal node. In the figure 11a, we can see that for the Gender dataset, that parameter increases a little bit the score of the classifier. But because we saw the perfect fit of the classifier with a maximum depth of 1, we wouldn't really care about that value.

For the Credit-Card dataset, presented on figure 11b, we can see that the best score is for a value between 12 and 17. It is interesting. The fact is I was not able to train as much data I would have liked for this dataset, because of computation time. Perhaps, with more data, result would have been more understanding.

In figure 12, I present the accuracy of the classifier according to the training set size. In the Gender dataset, we can see that the accuracy does not change a lot when the training set size is increased. That could mean that the data is not very complex and well balanced.

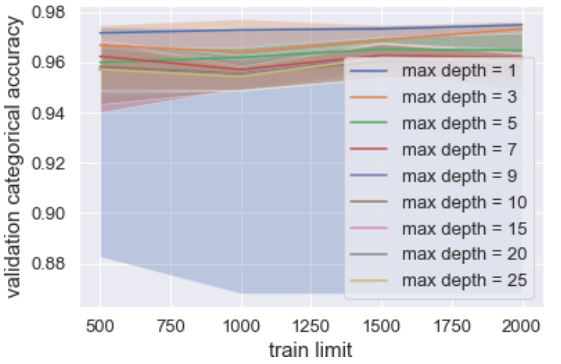


(a) Accuracy for the train and test Gender dataset, for $trainlimit = 2000$ and $maxdepth \geq 1$

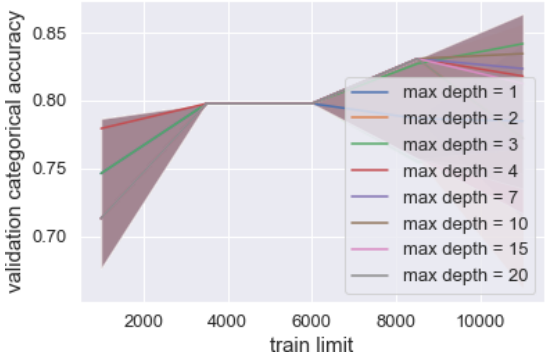


(b) Accuracy for the train and test Credit-Card dataset, for $trainlimit = 11000$ and $maxdepth \geq 1$

Figure 11: Accuracy of the Boosting Classifier according to the Minimum Number of Sample to Split an Estimator's node for a given size of training and maximum depth



(a) Accuracy for the train and test Gender dataset, for different train max depth values



(b) Accuracy for the train and test Credit-Card dataset, for different train max depth values

Figure 12: Accuracy of the Boosting Classifier according to the Training Set Size

In the Credit-Card dataset, presented on figure 12b, as expected, we can see that the accuracy increases when the training set size increases. There is that weird flat for a training set size between 3 500 and 6 000 were independent of the maximum depth, the score is the same. I could not explain why. I was not able to train more data due to the time computation of the Boosting classifier.

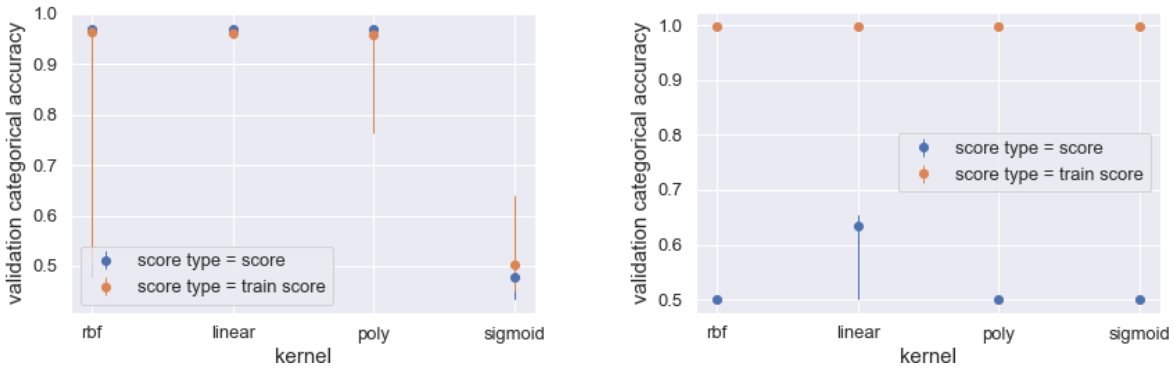
In terms of computation time, the Boosting Classifier is the slowest of all the classifiers I trained. To get the results and generate the data, it around 1h10m.

To conclude, we saw that Boosting classifier gets pretty good results. It seems to not tend to overfitting, which is a good thing. But it is the slowest of all the classifiers I trained. So, it would not really be a good choice for a real time application. For the Gender dataset, we saw that a Decision Tree was enough. For the Credit-Card dataset, I cannot really compare because I didn't trained it with enough data.

6 Support Vector Machines

In this section we will discuss about the Support Vector Machines classifier. We will see the influence of the followings parameters:

- The penalty of the error term (C value);
- The kernel function;
- The training set size.



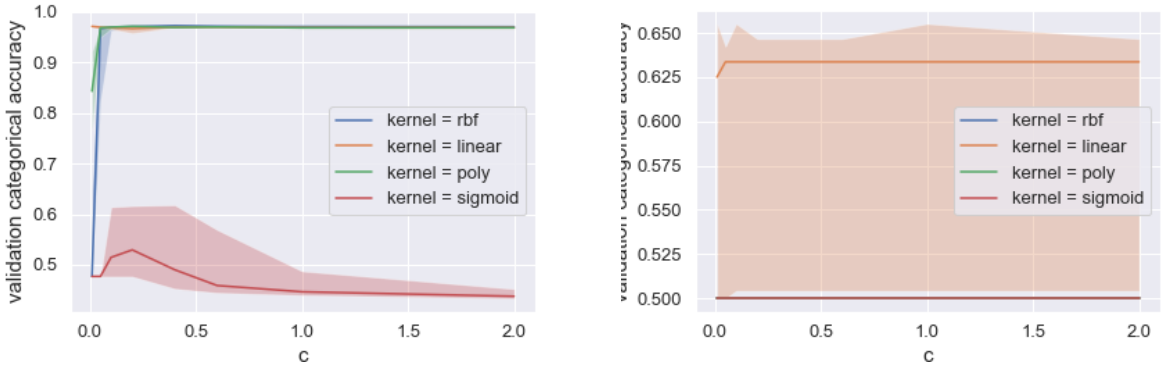
(a) Accuracy for the train and test Gender datasets

(b) Accuracy for the train and test Credit-Card datasets

Figure 13: Accuracy of the SVM Classifier according to the Kernel

The parameter that seems to have the biggest impact is the kernel choosed. In figure 13, we can see the impact on the score for both datasets. In the case of the Gender dataset, on figure 13a, we see that the accuracy is pretty much the same for all the kernels except for the *sigmoid* where the accuracy falls down to 0.45. The accuracy is around 0.98 for the other kernels : linear, polynomial (3rd degree) and the radial basis function. The score for the linear kernel clearly show that the data are linearly separable.

In the case of the Credit-Card dataset, on figure 13b, we see that the accuracy is better for the linear kernel, with an accuracy around 0.61. The other kernels don't perform very well.



(a) Accuracy for test Gender dataset according to different kernels

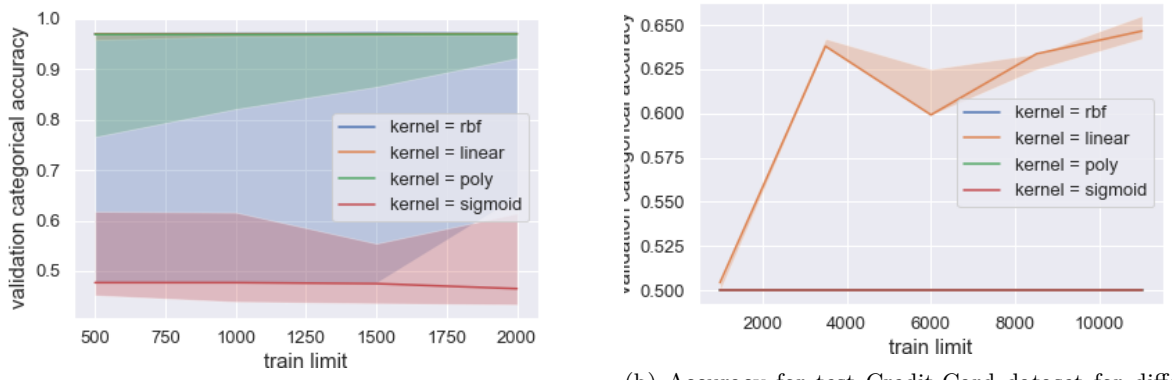
(b) Accuracy for test Credit-Card dataset according to different kernels

Figure 14: Accuracy of the SVM Classifier according to the C value

Let's now study the impact of the C parameter. In SVM, we are searching for a hyperplane with the largest minimum margin, and a hyperplane that correctly separates as many instances as possible. The problem is that it is not always possible to get both. The C parameter determines how great our desire is for the latter. Figure 14 presents the accuracy for the test dataset according to the C value.

For Gender dataset, we can see, on figure 14a that the C value doesn't have a big impact on the accuracy. The accuracy is around 0.98 for all the C values bigger than 0.01, except for the *sigmoid* kernel where the accuracy falls down under 0.5. Once again, because the data are linearly separable, the accuracy is pretty much the same for all the C values bigger than 0.01. The strength of the regularization is inversely proportional to C. So, the smaller C is, the stronger the regularization is. Having a C value too small means that we are considering the margin too much, and we are not allowing enough errors. So, we are underfitting the data.

In the case of the Credit-Card dataset, on figure 14b, we can see that the accuracy is neither really affected by the regularisation parameter for both kernels.



(a) Accuracy for test Gender dataset for different kernels

(b) Accuracy for test Credit-Card dataset for different kernels

Figure 15: Accuracy of the SVM Classifier according to the Size of the Training Set

The last parameter I will discuss for SVM is the training size. The results are presented on figure 15. For the Gender dataset, we can see on figure 15a that the accuracy is pretty much the same for all the training set sizes, with the working kernels. That dataset is indeed too easely separable.

For the Credit-Card dataset, we can see on figure 15b that the accuracy for the linear kernel is impacted by the training set size. There is that hollow betwenn 4 000 and 8 000 samples which means that we probably introduced outliers in the data. But the accuracy seems overall increasing with the training set size. I was not able to train more data due to the time computation of the SVM classifier.

Concerning time efficiency, the SVM classifier is one of the slowest. To get those results and generate the data associated with, it took around 25 minutes (knowing that I didn't train the SVM classifier with more than 11 000 samples for the Credit-Card dataset).

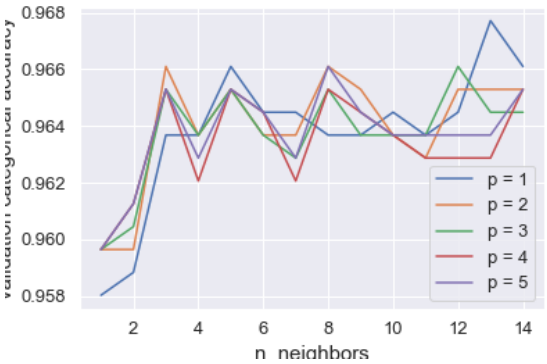
In conlusion, SVM does not perform very well on the most difficult dataset, the Credit-Card one. I think the most important for SVM working well is the choice of the kernel function as seen above. In the overhand, data are not always linearly separable. In the case of the Credit-Card dataset, it seems that it is not the case. So SVM does not seem to be a good option.

7 K-Nearest Neighbors

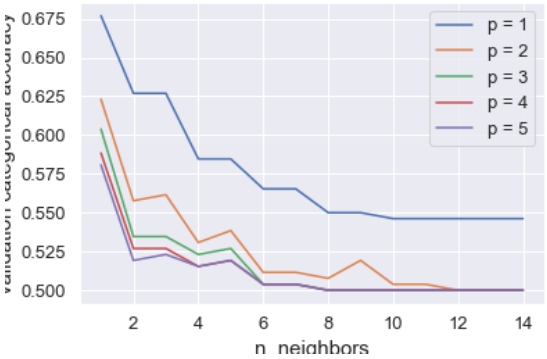
Here I will discuss the results obtained using a K-Nearest Neighbors classifier. We will see the influence on the accuracy of the classifier of:

- The number of neighbors;
- The distance metric (p-norm);
- The size of the training set.

By design, the accuracy on the training set is always 100% for KNN, so I will not display it on the plots.



(a) Evolution of the accuracy according to the number of neighbors on Gender dataset, with $train\_limit = 2000$



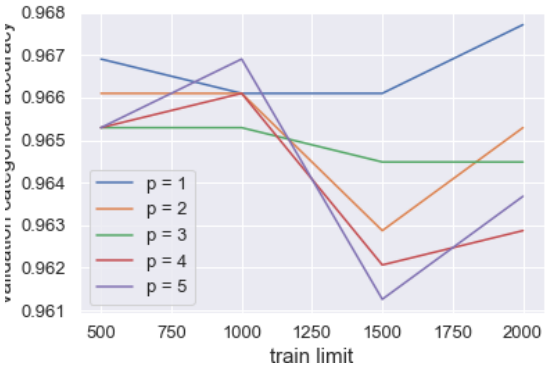
(b) Evolution of accuracy according to the number of neighbors on Credit-Card dataset, with $train\_limit = 90000$

Figure 16: Accuracy of the K-NN Classifier according to the Number of Neighbors

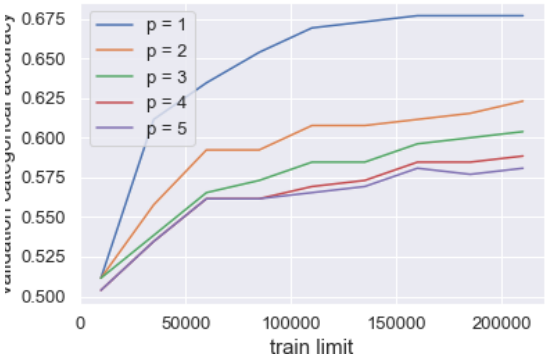
On figure 16, I present the effect of those parameters on the classification score. For the Credit-Card dataset, we can notice on figure 16b that the norm used for the distance metric has a consequent influence on the classification accuracy. The accuracy is better when using the Manhattan norm ($p = 1$). This is not a surprise as higher norm are generally inadequate to measure distances on highly dimensional spaces. In the case of the Gender datatset, the norm used doesn't seem to have a big influence on the accuracy.

On figures 16b we can see that the accuracy of the classifier is better for a small number of neighbors. K-NN is a non-parametric classifier. It is not able to generalize the data, so it is better to use a small number of neighbors. We usually increase the research perimeter to be more resilient to noise but here the data do not allow us such a liberty. This is particularly true in the Credit-Card dataset where the fraud are really close to legal transactions.

On figure 16a, we can see that the accuracy is better for a number of neighbors bigger than 3. In the case of the Gender dataset, the data are more separated and we can afford to increase the number of neighbors.



(a) Evolution of accuracy according to the number of training examples on Gender dataset, with $n\_neighbors = 13$



(b) Evolution of accuracy according to the number of training examples on Credit-Card dataset, with $n\_neighbors = 1$

Figure 17: Accuracy of the K-NN Classifier according to the Size of the Training Set

On figure 17, we can see that the accuracy of the classifier is better for an important number of training examples. Also the classifier doesn't overfit. On figure 17a, we can see that the accuracy decreases for a number of training

examples between 1 000 and 1 500. But it keeps having a very good score. I don't really know why. The samples introduced probably where not well representative of the data.

Concerning time efficiency, the K-NN classifier achieve a good performance. To get those results and generate the data associated with, it took around 25 minutes.

In comparison with the other methods, K-NN didn't perform well on the Credit-Card dataset. Having a discussion with an expert of the dataset would be a good idea for designing a distance metric that is better suited to the problem. Indeed, the distance metric is the key to success with K-NN. So it is important to choose the most appropriate one.

8 Conclusion

The best classifier for the Gender dataset was the Neural Network. The best classifier for the Credit-Card dataset was the Decision Tree. It is interesting to see that the most complex dataset got best score with one of the simpliest classifier, and the most simple dataset got best score with one of the most complex classifier in terms of parameters.

The most complicated in machine learning is without doubt the tuning of the parameters. It is a very time consuming task. It is also a very important task. The choice of the parameters is the key to success. That is why the domain knowledge is so important.

We also saw that the data is fundamental of machine learning. Every time we increased the training size we got better results. The cleanest data as possible, meaning the less outliers as possible, is also important. We saw that a simple dataset like the Gender one gaves us very good results with all classifiers.