What is the best classifier overall? What is the best regressor overall?

Baptiste Lemaire Averynder Singh Kyiv Edwards Samuel Cardin

Abstract

This report investigates two questions. First, for a given selection of data sets, can we say what is the 'best' classifier or the 'best' regressor in terms of good predictions? How much does the answer depend on the particular selection of data sets? How much does the answer depend on our computational constraints? We investigate these questions using data sets from the UCI repository. Second, we compare the performances of different Machine Learning Algorithms on a Computer Vision tasks. We then end on a performances comparison of a Deep Neural Network and a Convolutional Neural Network.

1. Introduction

Throughout the Fall 2021 semester, we studied many Machine Learning algorithms, such as Linear Regression and Decision Trees. Labs helped us learn to use them, however, we still don't know if one of them is more efficient than the others.

Indeed, we saw during the labs the performances of the algorithms vary a lot depending on the task. To create accurate algorithms, we can't study only one algorithm. Finding the appropriate model is the most important step in a Machine Learning problem's resolution.
Even if each algorithm has various performance depending on the task, we can still deduce the best algorithms for general tasks as Classification or Regression problems.

For this Final Project of COMP432 - Machine Learning, we compared multiple models on different datasets. First, we examined the Classification problem on eight data sets. We hoped to find which algorithms are suitable for Classification problems. Then, we studied similar algorithms on Regression problems. In the end, we hoped to find which algorithm to focus on for this second task.
Last, we studied Neural Networks' performances on a Computer Vision problem. We learned that for this kind of task, we favor Convolutional Neural Networks over Regular Deep Neural Networks. To prove this statement, we compared the performances of a CNN and a Deep NN to see which one performed better on a Computer Vision problem.

2. Methodology & Experimental Results

During this semester, we learned to use Machine Learning algorithms. We looked at the following eight:

- Linear Regression
- Support Vector Machine
- Decision Tree
- Random Forest
- K-Nearest Neighbours
- AdaBoost
- Gaussian Process / Naive Bayes
- Neural Network

2.1. Classification Experiments

For this first part, we are observed the different algorithms on Classification problems. For that, we have at our disposal 8 datasets:

- Diabetic Retinopathy
- Default of credit card clients
- Breast Cancer Wisconsin
- Statlog (German credit data)
- Adult
- Yeast
- Thoracic Surgery Data
- Seismic-Bumps

To know how well the model is doing, we used the accuracy. As it's a classification problem, we can simply count the number of right predictions and give an overall accuracy. We made the code to easily generate a table containing the accuracy of each pair of model and dataset. We stored those values in the Table 1.

From all the values, we got an average accuracy for each model.
The best three models were found to be the Random Forest, the k-nearest neighbours and the Gaussian Naive Bayes

as they were the only ones whose accuracy exceeded 80%. We can note that the Gaussian Process has the lowest average accuracy with 62%. Nevertheless, every algorithm has an acceptable accuracy. We don't have a model that outperformed the other models or underperformed.
From this study, we can't say that a certain algorithm is much more efficient than the others. All of them seem suitable for Classification problems, even if the Random Forest has the best performances and the Gaussian Naive Bayes classification.

2.2. Regression Experiments

In this new subsection, we compared different Machine Learning algorithms on Regression processes. To do that, we again used eight datasets to have an accurate idea of the performances of the models.

The datasets we used were :

- Wine Quality
- Communities and Crime
- QSAR aquatic toxicity
- Facebook metrics
- Bike Sharing
- Student Performance
- Concrete Compressive Strength
- SGEMM GPU kernel performance

For each data set, we used two evaluation metrics: the Mean Squared Error and the R^2 score. The Mean Squared Error represents the difference between the true value and the predicted one to the power two:

$$MSE(Y, \hat{Y}) = \frac{1}{n}\sum_{i=0}^{n}(Y_i - \hat{Y}_i)^2$$

However, we didn't use the MSE to compare the different models. Indeed, if a model has to predict 100 for a dataset and it actually predicts 90, the MSE is equal to 100 ($(100-90)^2 = 10^2$). For another dataset, this model needs to predict 1 and it predicts 0.9, the MSE is equal to 0.01. For these two dataset, we have the same percentage difference (10%) but the MSE is 10,000 times higher. If we want to have an average MSE, some datasets will have much more importance than the others. Therefore, we couldn't just use the MSE for our problem.
We decided to use the R^2 score (also known as Coefficient of determination). This metric represents the correlation between the predicted value and the true one to the power two. By definition, it has a range from 0 to 1.

$$R^2 = \left[\frac{1}{n}\sum_{i=0}^{n}\frac{(X_i - \bar{X}) \times (Y_i - \bar{Y})}{\sigma_x \times \sigma_y}\right]^2$$

Throughout the code, we save the score of the models for each data set to easily compare the performances. We sum up all those measures in the Table 2. As we can see, some R^2 scores are below 0. According to this thread, https://stats.stackexchange.com/questions/183265/what-does-negative-r-squared-mean, this means that the model used is not adapted for the problem, and so a simple line would have a higher accuracy than this model. This line can be generated thanks to Linear or Logistic Regression model.

As we can see in Table 2, the three best models have almost the same R^2 score average : the Linear Regression, the Random Forest and the AdaBoost, with an average higher than 0.6.

The most interesting measure of this table is the average for the Support Vector Machine. Indeed, the average R^2 score for the SVM is negative. As we said earlier, if the correlation score is negative, that means a simple line is more accurate than the model. Therefore, we can say that the SVM is not meant for Regression process as we have much more efficient models.

Also, we can see that Neural Network only took fourth place in this table. We used a Deep Neural Network so we expected it would have one of the highest scores but not at all. The reason seems to be that we don't have enough data. Indeed, Neural Networks become very efficient when they trained on thousands of samples. And some of the dataset we used only have 600 samples. We plot the accuracy of the model according to the size of the dataset :

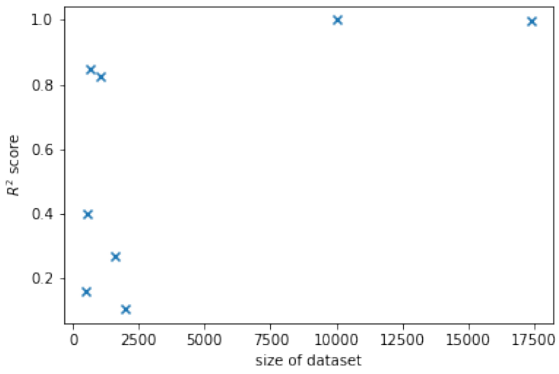

Figure 1. R^2 score depending on the size of the dataset

Even if two samples don't follow the tendency, we can see that the more samples the model has, the higher its R^2 score was. Knowing this, we are studied Neural Networks with a much bigger dataset.

2.3. Interpretability Experiments

For this part, we were going to study a Computer Vision problem. We would compare different models on the same dataset: the CIFAR-10 dataset. This dataset is composed of images of 10 differents categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). Each category has 6,000 images, so the whole dataset is composed of 60,000 images with a 5:1 split for Training/Test dataset (50,000 images to Train and 10,000 to Test).

First, we built a Decision Tree. We wanted to have a benchmark to compare the performances of different algorithms.
After training the model on the training set, we determined its accuracy on the test set. We saw that the model quickly converges to an accuracy of 100%. However, when we make prediction on the test set, we ended up with an accuracy of 22%, which is not a lot. We can conclude the model quickly overfits the data and so becomes inefficient.

Next, we built a Convolutional Neural Network. During the different labs we had, we saw that CNN have very great performances for most Computer Vision tasks.
We designed a simple CNN with the following architecture:

Layer Name	Layer Shape	# Params
Input_1	(3,32,32)	0
Conv2D_1	(16,32,32)	448
MaxPool2D_1	(32,16,16)	0
Conv2D_2	(32,16,16)	4,640
MaxPool2D_2	(32,8,8)	0
Conv2D_3	(64,8,8)	18,496
MaxPool2D_3	(64,4,4)	0
Flatten_1	(1024)	0
Dropout_1	(1024)	0
Linear_1	(500)	512,500
Dropout_2	(500)	0
Linear_2	(10)	5,010

This architecture has 541,094 trainable parameters. We trained the model on 20 epochs. Indeed, even if the architecture was not enormous, we still have 50,000 samples to compute on each epoch. Therefore, it took around 30 minutes to compute those 20 epochs. Finally, we ended up with an average accuracy of 75%. This performance is totally satisfactory knowing that the architecture is not the optimal one and that we only run the model for 20 epochs.
With this dataset, we could use Lenet-5. Indeed, this architecture is meant for the MNIST dataset for a Digit Recognition problem. The two datasets have samples with almost the same shape: (3,32,32) for CIFAR-10 and (1,32,32) for MNIST (counting the padding of Lenet-5). The two dataset have 10 categories. Lenet-5 is known to be very efficient on this task. Therefore, we could also make a study about using this architecture on the CIFAR-10 dataset. We could also try to use Transfer Learning from the MNIST dataset to the CIFAR-10 one.

2.4. Novelty component

In the finl part of this project, we used the CIFAR-10 dataset. We used this dataset with a Convolutional Neural Network. During the semester, we learnt that CNN are mostly meant for Computer Vision. We decided to see how it compares to a normal neural network for this task. We trained a simple Deep Neural Network and compared the performance of the two models.

First, the architecture of the model. We designed it iteratively, meaning we first created a small NN, trained it, and tested it. Then we added either neurons or layers. We once again trained and tested it. We kept doing that until we had a satisfactory accuracy. The model we ended with is probably not the most efficient, however for our study, its performances are sufficient to conclude on the comparison. Note that for this part, we didn't use a particular CNN Architecture. Indeed, many research papers are about the most efficient Architectures for certain problems (Lenet-5 for Digit Recognition, VGG-16 for Object Recognition,*etc.*). Therefore, the performances we obtained are not optimal.

Finally, we used the following architecture for our Deep Neural Network:

Layer Name	Layer Shape	# Params
Input_1	(3072)	0
Linear_1	(1536)	4,720,128
ReLU_1	(1536)	0
Linear_2	(768)	1,180,416
ReLU_2	(768)	0
Linear_3	(128)	98,432
ReLU_3	(128)	0
Linear_4	(64)	8,256
ReLU_4	(64)	0
Linear_5	(10)	650
LogSoftMax_1	(10)	0

We end up with 6,007,882 parameters, which is a lot compared to other models. For exemple, Lenet-5 has only 60,850 parameters and it has the same almost the same input (32x32x1 instead of 32x32x3).

After training the model for 100 epochs, we finally got an accuracy of around 50%. So we lost 15% of accuracy. If we would have trained the CNN for a hundred of epochs, we could have even better performances and so the difference would be much higher.

The number of parameters of the CNN we used represents 10% of the number of parameters for the Deep NN,

the CNN trained for only 20 epochs compared to 100 for the DNN and the CNN still end up with better performances. From this, we can conclude that, for Computer Vision, CNN seem to be a better option than Deep Neural Network.

3. Conclusions

Throughout this report, we studied the performances of different Machine Learning algorithms. We compared them on a Classification problems and saw that each algorithm does fairly well on this task. Therefore, for a real life problem, a Machine Learning scientist has to compare each one of them on his data to find the algorithm that suits it the best. Secondly, we compared those models on a Regression problems and saw that the Support Vector Machine has terrible performances for this kind of task. Therefore, we should prioritize other algorithms for Regression problems.
Last, we compared different algorithm on a Computer Vision problem. We mostly compared a regular Deep Neural Network and a Convolutional Neural Network. As expected, the CNN is much more efficient for this kind task. Even if our model was not perfect, it would still have great performances.

During this semester, we learnt the basic knowledge to do Machine Learning. We saw how to use those algorithms with the Python package Sci-kit Learn and PyTorch. The next step would be to study more profoundly Deep Learning as COMP-432 was mostly about the fundamentals of Machine Learning.

Dataset	Linear	SVM	DT	RF	KNN	AdaBoost	GNB	NN
Diabetic	0.7056	0.7316	0.7316	0.7619	0.7273	0.7186	0.7229	0.7229
Credit Card	0.7838	0.7828	0.7377	0.8217	0.7792	0.8260	0.3647	0.7840
Cancer	0.8643	0.9714	0.9357	0.9786	0.9786	0.9714	0.9571	0.9714
Statlog	0.7550	0.5100	0.6700	0.7350	0.7250	0.7600	0.7100	0.7450
Adult	0.7930	0.7858	0.8136	0.8485	0.7857	0.8601	0.7936	0.2839
Yeast	0.6532	0.6633	0.6869	0.7340	0.6970	0.7239	0.3535	0.6902
Surgery	0.7979	0.8085	0.7447	0.7979	0.8085	0.8085	0.2128	0.7979
Seismic	0.9130	0.1199	0.8956	0.9381	0.9400	0.9362	0.9052	0.8433
Average	0.7832	0.6717	0.7770	0.8269	0.8051	0.8256	0.6275	0.7298

Table 1. Accuracy of every models for each dataset, Classification

dataset	Linear	SVM	DT	RF	KNN	AdaBoost	GP	NN
Wine quality	0.3284	-0.2196	-0.0863	0.4335	0.1374	0.2900	0.2186	0.2680
Communities	0.1507	0.0817	-0.5069	0.2392	0.0405	0.1504	0.1705	0.1062
QSAR	0.4402	0.2247	0.0246	0.3710	0.2207	0.3480	0.3500	0.3983
Facebook	1.0	-0.0244	0.5668	0.5222	0.0265	0.5267	-0.0022	0.1608
Bike	1.0	-1.2549	0.9990	0.9996	0.9996	0.9765	0.1731	0.9988
Student	0.8487	0.5057	0.8280	0.8806	0.8135	0.8836	-0.0090	0.8480
Concrete	0.5620	0.3044	0.8063	0.9294	0.6967	0.8232	0.2341	0.8252
SGEMM	1.0	0.1444	0.9999	0.9999	0.9998	0.9953	0.4759	0.9995
Average	0.6662	-0.0297	0.4539	0.6719	0.4918	0.6242	0.2014	0.5756

Table 2. R^2 score of every models for each dataset, Regression