VAN ECCELPOEL
Théo

Advanced Machine Learning
Lab 3 – Classification
Report & Analysis

In this report, I will analyze the lab I conducted on classification, a core task in machine learning that involves predicting binary or categorical outcomes. Classification models are broadly divided into two categories: discriminative models and generative models. Discriminative models focus on directly separating data into classes by modeling decision boundaries, while generative models learn the underlying distributions of each class to generate samples.

Generative models are particularly effective for smaller datasets, as they make stronger assumptions about the data's structure. This lab explores examples of both categories, including Logistic Regression and Support Vector Machines (SVM) as discriminative models, and Linear Discriminant Analysis (LDA) as a generative model.

1) First part : NLS_2.csv

The first thing interesting to experiment was the size of the test set (how to split our dataset).
I tried with 2 split methods: 33% of test sample and 20%. The idea was to try with 20% as well, to have more data to train our model and potentially come up with a better model.
I also tried to run several splits using a different random seed, in order to see the results on several splits (kind of k-fold cross-validation).
Based on my experiments, I noticed slight variations in model performance metrics (accuracy and AUC). A larger test set (33%) generally provides a more reliable estimate of the model's performance, as it better represents the diversity of the data. However, reducing the test set size (20%) increases the training data available, which can marginally improve performance on smaller datasets. This tradeoff highlights the importance of balancing the test and train split, depending on the dataset size and the need for robust evaluation.
I was interesting to notice that with some random seeds, results with 20% were better, whereas some others were worse... I learnt that we should always, when possible, try different seeds to not assert things that are not true on average (usefulness of k-fold methodology).

Then, I was interested in the results of the parameter optimizations.
The optimization of the regularization parameter shows a 2% increase in the performance metrics according to the parameter value used. We can also notice that at some point, the scores reach a kind of plateau, before decreasing sharply again. Thus, it is useful to plot the different scores according to the different parameter values, in order to better understand the behavior of the model regarding its regularization parameter.

With generative model such as LDA, we observe similar performances as the regularized logistic regression. It means that this model can perform quite well on our data. However, we will see just after that the scores can be significantly improved, which means that the assumptions on prior distribution of our data were not totally correct, leading to not best results.

Now, we are ineterested in another discriminative method, the SVM model. From a theoretical point of view, logistic regression with Ridge regularization (l2-norm) is more suited when probabilistic outputs are required, and handles noise in the data more smoothly due to its probabilistic nature.
However, SVM models can perform way better when there are a clear margin between classes.
With our dataset, SVM performs the best for now, which shows that our data contains margins that can be easily separable. However, looking at our data, it doesn't seem that there is a linear distinction between our classes, which also confirm that LDA, which is linear, did not perform the best with our data.
Indeed, we proved that SVM performed best with Radial Basis Function kernel, showing a non-linear margin between the classes.

Based on our previous conclusion, it would be interesting to implement as well a non-linear kernel to our Logistic regression model. As a result, using PolynomialFeatures to implement Logistic Regression with Polynomial kernel, we notice a significant increase in the model performance on our data, with results equivalent to the SVM. We may thus prefer logistic regression, which, in my opinion, have an additional advantage over SVM due to its probabilistic nature of its results.

I learnt the importance of kernelized models to better fit the shape of the data. Also, by plotting the dataset, I think we can already save some computation power by looking at the shape of the separation plane between the classes, and not compute models that we already know won't perform well on our data (linear models in our case).

2) Second part : Bank-full.csv

What I like about this lab is the "mini-project" with this second dataset.
Indeed, this second part is mainly about cleaning the data, which makes a lot of sense, as I believe that the reality of a data scientist is not just applying models, tweaking parameters and computing scores, but is mainly about getting the right data into the right model. As a data scientist, this data cleaning task will be dominant in my job, and

therefore it was interesting to do this part as well to highlight the importance of this reality.

First, we explore out dataset, looking at the first rows and at the main statistics.
It seems that the dataset is a bit unbalanced, as 75% quantiles and max are totally different for some columns.

Then we clean our data, first formatting it in a way we can analyze it, with hot-encoding.
We can try with drop_first=True or its opposite, to see if the removal of some variables to decrease colinearity boosts model performance or not. As a result, not removing extra variables leads to better results, which makes sense as multicolinearity is not as important as in linear regression tasks.

For numeric variables, we rescale them to not bias our model towards bigger variables.

We then try 8 different models (with different kernels) for the model selection step.
It seems that the best model for our data is the SVM with RBF kernel. The Logit with Polynomial Kernel is great as well, showing a non-linear boundary between our classes.
We can then optimize the regularization parameter to get our final best model. I learnt as well that regularization parameters differ a lot from a dataset to another one, so we should keep the range of testing values quite large.