Machine Learning

5th lab assignment

Instituto Superior Técnico

Evaluation and Generalization

Group 5 – Wed 17:00

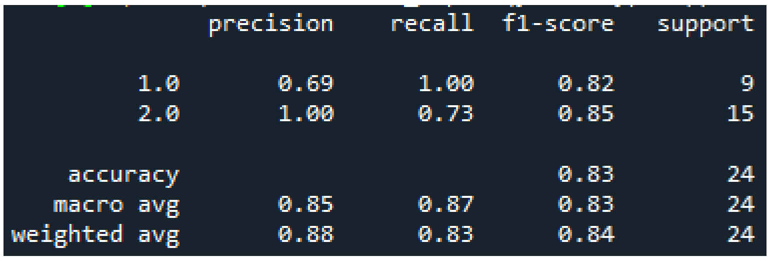
José Diogo Teixeira Cunha Castro – 84401
André Luís - 98638

2.1 Classification Task

For the classification task, we started by importing the Cancer dataset, to get the train and test data.

First, we implemented **support vector machines**, for which we needed to decide between a linear and non linear SVM. For this, we decided to automate the search for the best method and parameters, using the GridSearchCV() function. As input for this function, we gave one linear SVM, Linear, and one nonlinear SVM, Radial Basis Function, as well as several values for the margin, C, varying from 0.1 to 10000, and for gamma, varying in the same scale.

The solution of this optimization problem was a linear SVM with C=100, for which we computed the classification report and the confusion matrix:



```
              precision    recall  f1-score   support

         1.0       0.69      1.00      0.82         9
         2.0       1.00      0.73      0.85        15

    accuracy                           0.83        24
   macro avg       0.85      0.87      0.83        24
weighted avg       0.88      0.83      0.84        24
```

Figure 1 – Classification Report for SVM

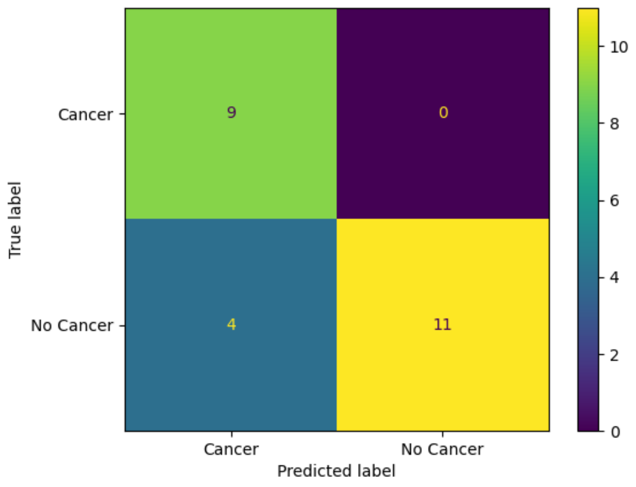


Figure 2 – Confusion Matrix for SVM

Based on this, we can conclude that the chosen SVM got an accuracy of 83%, with a precision of 100% identifying patients that did not have cancer, and 69% predicting patients with cancer, with 4 individuals being classified as having cancer, when they had not, as we can see in the confusion matrix.

The second classification algorithm used was **Decision Trees**. In order to avoid the overfit of our decision tree, we decided to prune it, that is to find the smallest tree with the minimum number of errors. For this, we need to find the right alpha, that controls how much pruning is going to happen.

We divided our training data into training and validation, in order to know which alpha to choose:

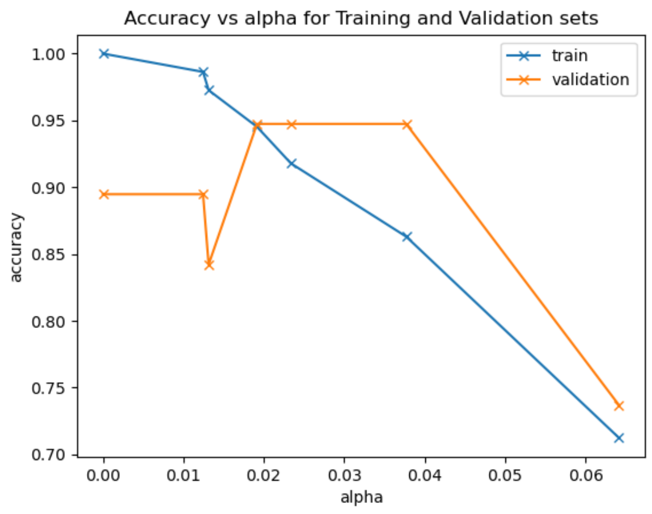


Figure 3 – Accuracy vs Alpha

Bases on this chart, we choose an alpha of 0.02 for our final decision tree:

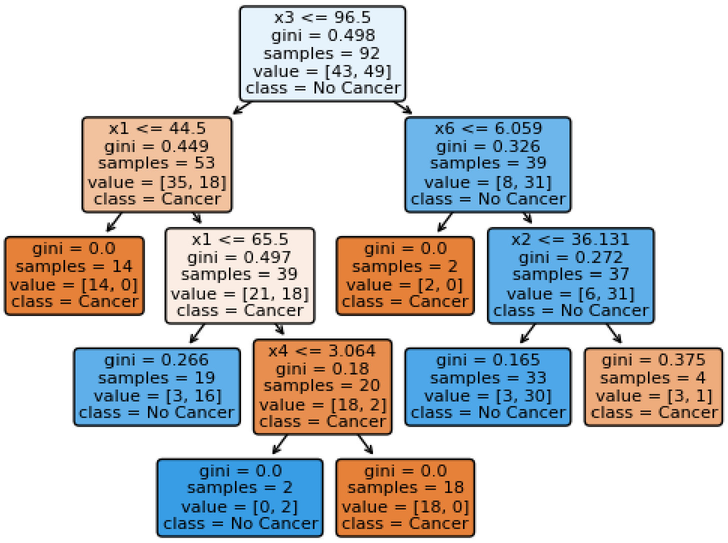


Figure 4 – Decision Tree for alpha = 0.02

This classification method presented an accuracy of just 54%, and the confusion matrix looks as follows:

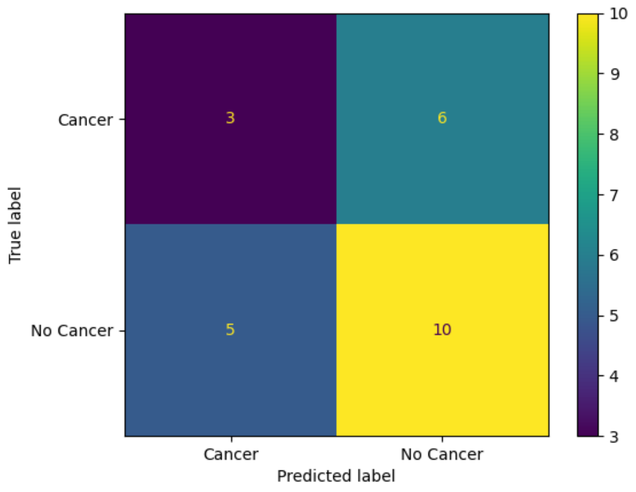


Figure 5 – Confusion Matrix for Decision Trees

Comparing both methods, we can see that support vector machines performed better than decision trees for the Cancer dataset. The poor performance of the decision trees can be explained by tendence for overfitting form this algorithm, and can be seen in our example, because due to the nature of our dataset, with few records, the results from the chart in figure 4 varied a lot, almost always producing overfitted results that when applied to the test set, had a low performance.

Support vector machines, on the other hand, give decent results even with a small dataset like ours, and the fact that it was linearly separable also helped in the good results achieved.

2.2 Regression Task

For the second part of the project, we have a regression prediction task.

We were given a Real Estate dataset already divided in two parts:

1) **Real_Estate_Xtrain/…ytrain:** Composed of 404 Houses, 13 properties of the houses and the respective price of the houses.
2) **Real_Estate_Xtrain/…ytrain:** Composed of 102 Houses, 13 properties of the houses and the respective price of the houses.

To better understand the data here is an example of some houses from the training data:

House Number	CRIM	ZN	INDUS	Pool	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	Price
1	0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
2	0.06127	40	6.41	1	0.447	6.826	27.6	4.8628	4	254	17.6	393.45	4.16	33.1

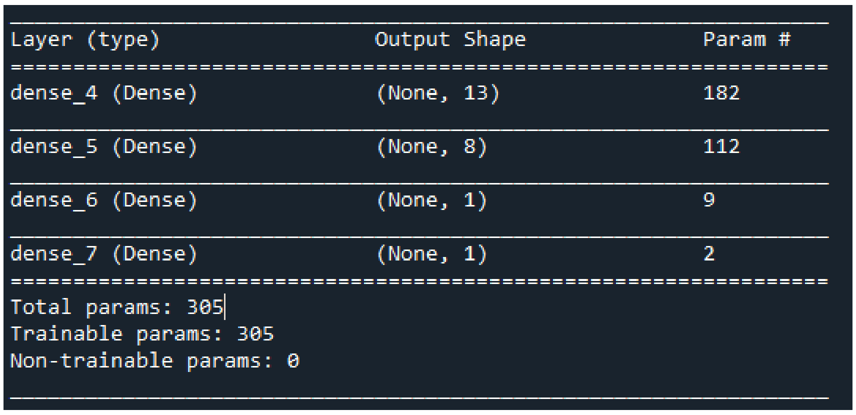
*Properties were assigned by us to give real context to the problem.

The dataset was already good, and there were no missing values, or any significant outliers. We could remove some properties after the principal component analysis, however, to test our MLP with all the possible details we consider maintain all the variables.

Since some values from the properties were binary, and most of them have a lot of variance we executed a standardization from the variables using the library *sklearn.preprocessing*.

We split the training in subsets, 30% for validation and rest for training and performed an MLP with early stopping to prevent overfit to the training data, that would lead to bad results when predicting unseen data.

The MLP model selected was:



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 13)                182
_________________________________________________________________
dense_5 (Dense)              (None, 8)                 112
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 9
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 2
=================================================================
Total params: 305
Trainable params: 305
Non-trainable params: 0
_________________________________________________________________
```

After training the MLP with early stop we get this plot comparing the training vs validation loss:

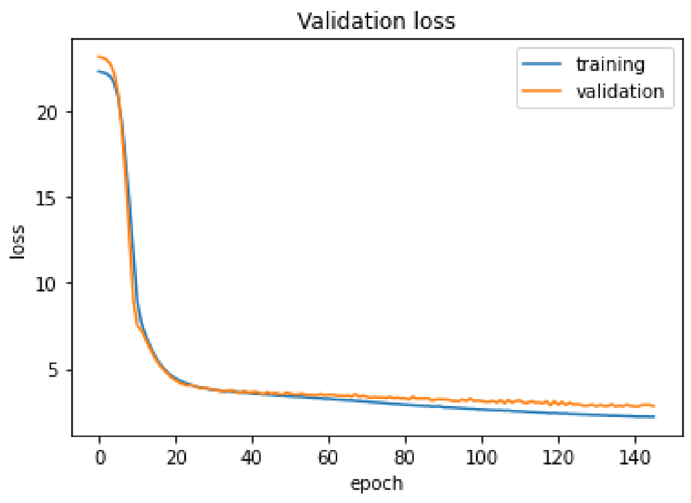


Figure 6 – Training vs Validation Loss

- Mean Squared Error = 10.03
- Mean Absolute error = 2.415
- Maximum Error = 9.73

Using Linear Regression Lasso:

For the second method we choose the linear regression Lasso but this time we did not divide the training set, so that we have more data to train our linear regression. After some iterations we found that alpha equal to 0.5 would give us the best results.

We obtained the following values of error:

- Mean Squared Error = 19.83
- Mean Absolute error = 3.38
- Maximum Error = 16.23