Milestone 2 Report

1. The used classification models:

A. Random Forest Classifier:

- ⇨ The random forest algorithm is a supervised classification algorithm. As the name suggests, this algorithm creates the forest with a number of trees. In general, the more trees in the forest the more robust the forest looks like. In the same way in the random forest classifier, the higher the number of trees in the forest gives the high the accuracy results. In Random Forest, each decision tree is made independent of each other. So, the order in which trees are made is not important. It uses parallel ensembling.

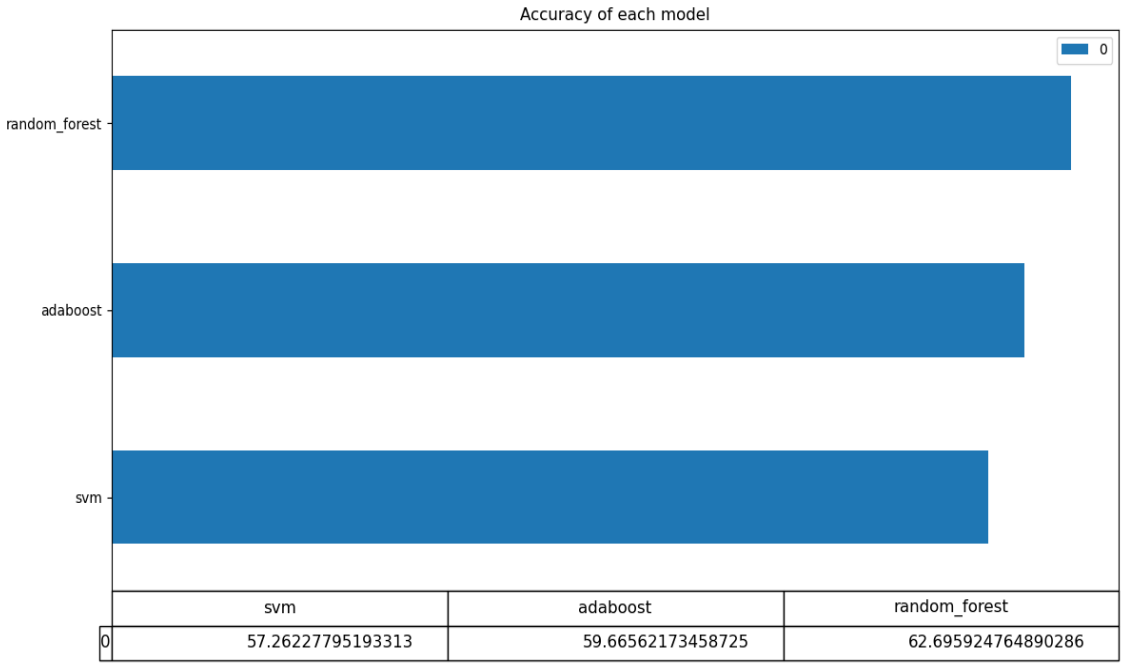
B. Adaboost with decision trees:

- ⇨ Boosting algorithms are a set of the low accurate classifier to create a highly accurate classifier. Adaboost, one of the common boosting algorithms, is an iterative ensemble method that build a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. It also uses sequential ensembling, as every iteration is depending on the previous one. It can use different types of classifiers “weak learners” to make a high accurate classifier. The default used classifier with adaboost is decision trees. The decision tree concept is more to the rule-based system. Given the training dataset with targets and features, the decision tree algorithm will come up with some set of rules. The same set rules can be used to perform the prediction on the test dataset.

C. SVM with Gaussian Kernel:

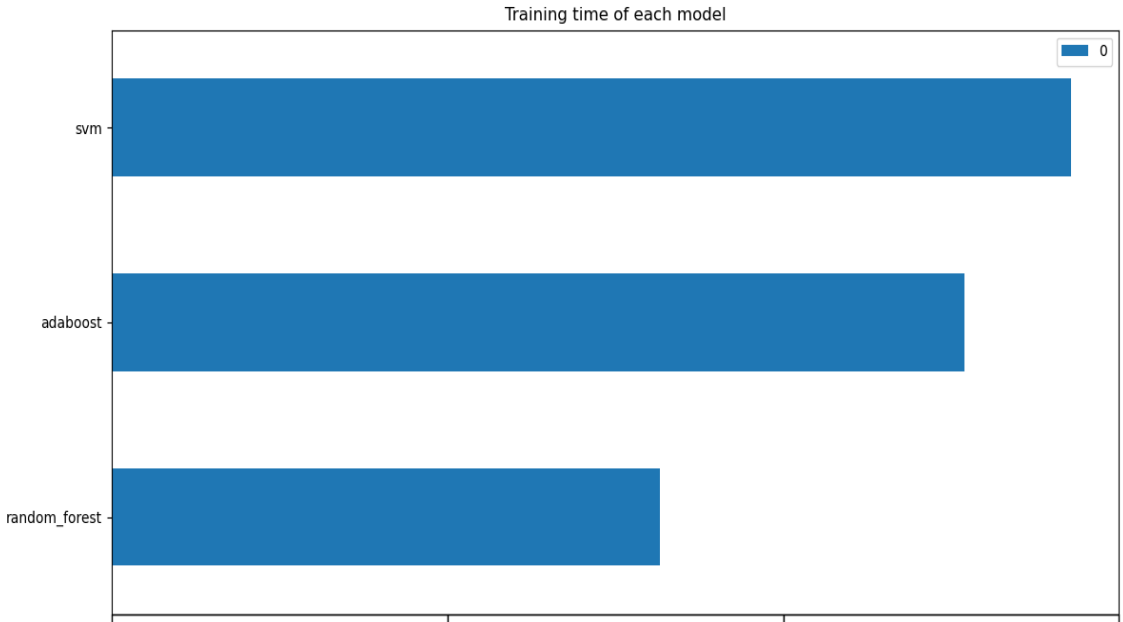
⇨ Support vector machine produces significant accuracy with less computation power. It can be used for both regression and classification tasks, but it is widely used in classification objectives. The objective of the support vector machine algorithm is to find a hyper-plane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. It differs from one model to another in the type of the used kernel function, that is responsible for transforming the dataset to a higher dimensional space.

2. Classification accuracies:



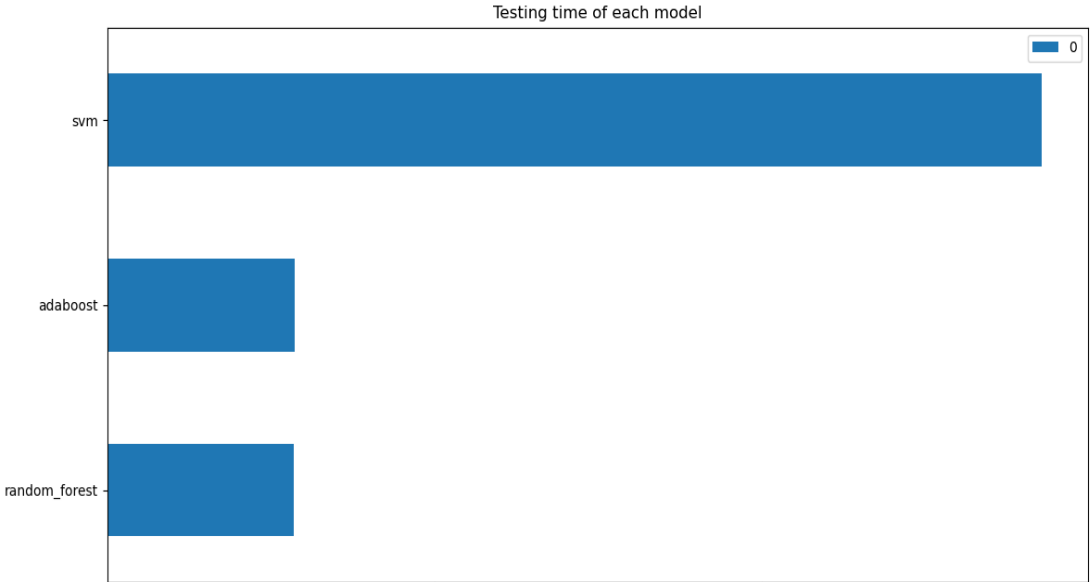
	svm	adaboost	random_forest
0	57.26227795193313	59.66562173458725	62.695924764890286

3. Total training time:



	random_forest	adaboost	svm
0	0.5625760555267334	0.8750791549682617	0.9844048023223877

4. Total testing time:



	random_forest	adaboost	svm
0	0.031250715255737305	0.03126406669616699	0.15623831748962402

5. Hyperparameters tuning:

In SVM Classifier, there are a lot of parameters to tune such as choosing the kernel function and selecting values for the C parameter and gamma. The C parameter (1 / λ) trades off correct classification of training examples against maximization of the decision function's margin. For larger values of C, a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin, therefore a simpler decision function, at the cost of training accuracy. In other words, C behaves as a regularization parameter in the SVM. The gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

A. Tunning "C" with fixed gamma = 0.0001:

The large C will cause lower bias and high variance "overfitting", while the small C will cause the opposite "underfitting". So, the C value should be balanced.

According to my trials:

- While C = 0.1, accuracy = 51.724%
- While C = 10, accuracy = 57.262%
- While C = 100, accuracy = 55.485%

B. Tunning gamma with fixed C = 10:

A small gamma value defines a Gaussian function with a large variance. In this case, two points can be considered similar even if are far from each other. In the other hand, a large gamma value defines a Gaussian function with a small variance and in this case, two points are considered similar just if they are close to each other.

According to my trials:

- While gamma = 0.0001, accuracy = 57.262%
- While gamma = 0.1, accuracy = 56.635%
- While gamma = 10, accuracy = 51.724%

So, I tried to use a small gamma value with a balanced C value, so that the high variance that is resulted from the low gamma value will be handled with the C value “Regularization”.

6. Conclusion:

In a nutshell, this phase in the project was very informative. Using new algorithms and models for classification was a good thing to differentiate between them. We knew how to start solving the classification problems. The dataset was pre-processed in a way that is similar to milestone 1 except some small differences. Knowing how to tune the hyperparameters is a very important skill as it helps to reach the desired accuracy and avoiding the future problems that might appear during the prediction phase such as overfitting and underfitting.