# COS40007- Artificial Intelligence for Engineering

## Portfolio Assessment – 2

**Name –** Rupayan Banerjee

**Student Number –** 103538229

**Studio Class –** 1-3

After performing the tasks, as outlined in Studio 3, the following is a summary table of the accuracy value of all the SVM models developed so far. This is a representation of Activity 6 mentioned in the studio tasks.

|  |  |  |
| --- | --- | --- |
| **SVM Model** | **Train-Test Split** | **Cross Validation** |
| **Original Features** | 89.71% | 89.17% |
| **With Hyperparameter Tuning** | 84.04% | 83.60% |
| **With Feature Selection and Hyperparameter Tuning** | 84.29% | 83.80% |
| **With PCA and Hyperparameter Tuning** | 84.15% | 83.64% |

After performing the train-test split and cross-validation on the original set of data with other available forms of classifiers, as listed in the provided document, these are the observations in the form of a table below. This is a representation of Activity 7 mentioned in the studio tasks.

|  |  |  |
| --- | --- | --- |
| **Model** | **Train-Test Split** | **Cross Validation** |
| **SVM** | 89.08% | 89.20% |
| **SGD** | 75.47% | 88.06% |
| **RandomForest** | 92.32% | 92.61% |
| **MLP** | 88.48% | 88.39% |

This week, I have also gone step-by-step through the portfolio tasks and have performed them successfully. A link to the GitHub repository consisting of my portfolio tasks is provided later, along with the link to the studio tasks. Step 4 in the portfolio tasks consisted of training an SVM models with different classifiers. The following table outlines their accuracy.

|  |  |  |
| --- | --- | --- |
| **SVM Model** | **Train-Test Split** | **Cross Validation** |
| **Original Features** | 77.84% | 75.02% |
| **With Hyperparameter Tuning** | 77.56% | 75.19% |
| **With Feature Selection and Hyperparameter Tuning** | 77.56% | 75.19% |
| **With PCA and Hyperparameter Tuning** | 77.56% | 75.19% |

I have also conducted training with different ML models, a comparison of whose accuracies is also listed in the following table in accordance with the step 4 of the portfolio tasks.

|  |  |  |
| --- | --- | --- |
| **Model** | **Train-Test Split** | **Cross Validation** |
| **SVM** | 77.84% | 75.02% |
| **SGD** | 75.42% | 75.19% |
| **RandomForest** | 73.75% | 73.71% |
| **MLP** | 76.44% | 75.91% |

Now from both the tables above we can draw some conclusions as asked off us in step 5 of the portfolio task. They are summarised below.

As for the best SVM model to use in our case, it is very tight among all the options, especially since, due to the flattening out of the peaks, all the 3 models, except the one with the original features, exhibit same accuracy levels. However, if I were to choose one it has to be the first one with original parameters, at least for its train-test split training structure, where it compromises a bit in its cross-validation structure when compared to the others.

Now for the best ML model in this case, SVM stands out as the best when it comes to the train-test split structure. However, it is outperformed by MLP in the cross-validation structure. Overall though I would still pick SVM to be the best model since the compromise is less than the improvements.

## Appendix

Link to all the code files - <https://github.com/rupayan-banerjee/CodeBase.git>