Assignment 1 Report : 15D170013

Data Preprocessing:

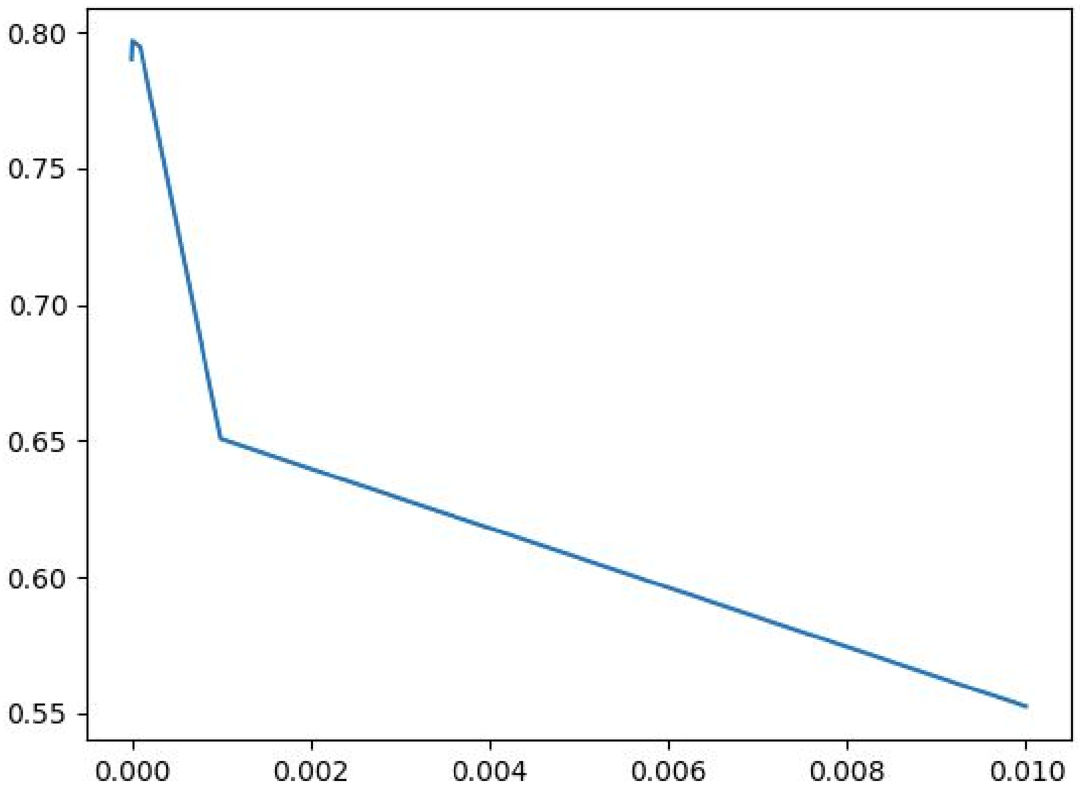
1) Used one hot encoding since the data values did not have an inherent notion of closeness if we just used the numbers 1 to n for the classes. Hence one hot which assigns separate columns / features to each of the following is a better method to deal with such data points. This in turn also did the conversion of nominal to numeric variables. There was this issue with missing variables in the case of one hot encoding which was resolved by concatenating the train and test data and then applying one hot encoding and then splitting them again.
2) Given the accuracy of around 50% using the SVM classifier I added a few modified features like the square of the square foot area and all such features which had the notion of some number representing something attached to them. But even after adding such features there was no significant increase in the model performance.
3) Normalization: Normalizing the feature columns which had numeric input led to increase in the accuracy of all the models (0.49-0.55 for SVM, 0.45-0.75 for Naive Bayes, etc). Normalisation performed by selecting the columns that had numeric values and then applying the normalisation function.

Cross Validation Methodology:

Also splitted the train data into two sets train and test into roughly around (70%:30%) and choosing the hyper-parameters based on the score on this test data set.

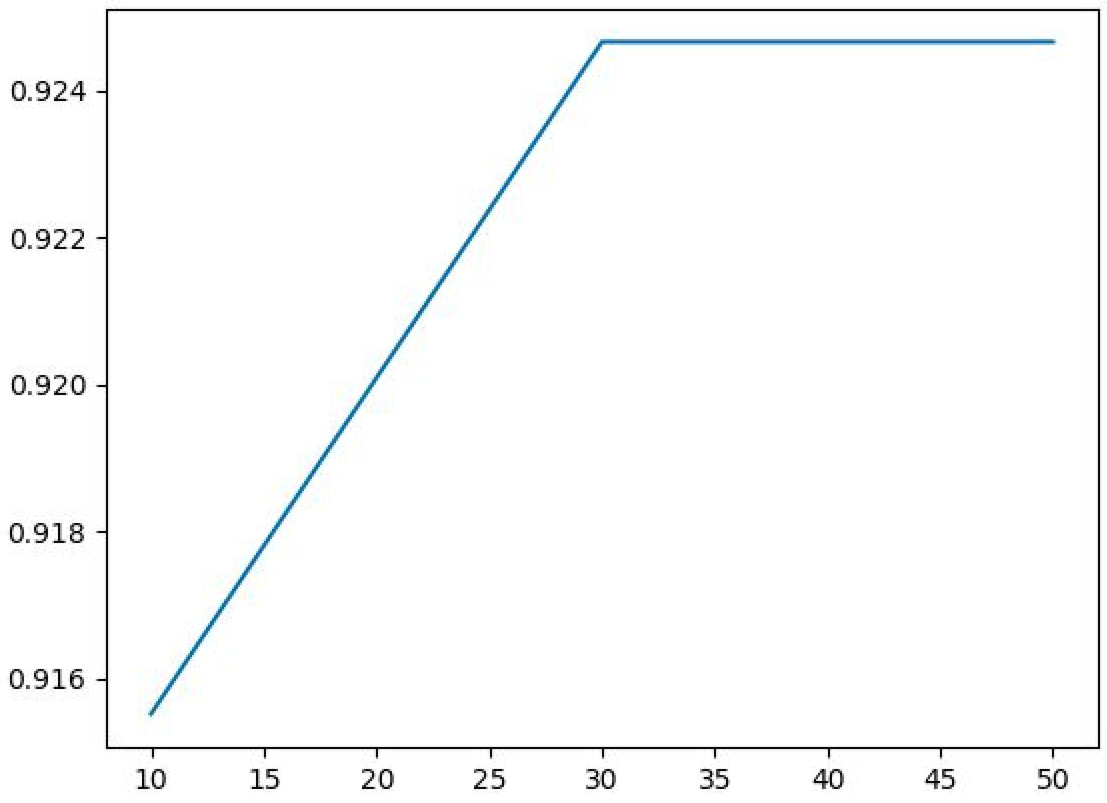
Classification Models and Tuning of Hyperparameters:

1) SVM: Used the SVM classifier model from the sklearn library.
   Hyperparameters Tuned:
   Iterated for different values of gamma: Iterated over gamma values range from 1e-1 to 1e-6. The best score/dec in gamma ratio was found to be close to around 0.001. Also changed the values for C, not much variation in the score, so keeping it set as default. Also tried changing the kernel to different ones available. It was taking a lot of time to train the model for the different kernels, and found an article on the net stating that not much change is observed after changing the kernels so skipped it. Score increased after improving gamma from 0.65 to 0.79.



2) Decision Tree Classifier (Not taught in class):
Iterated over the hyper parameter of random_seed and used the one with the max value to train the given model. Provides the best score amongst all the models.

3) Random Forest Classifier (Not taught in class):
Used the Random Forest Classifier with tuning done on the n_estimators to find that the best score/ time ratio comes with n_estimators of 30. This works best because Random Forests do not have many hyperparameters to train and are the best at avoiding overfitting too which probably is the case in our example.



4) Naive Bayes Estimator :

Just to compare. This is of the least power. Score on the test part of the train data of about 0.8

5) Logistic Regression Classification:

Used the normal logistic regression classification based approach to get an accuracy of about 0.82. Didn't tune the parameters since this also was a basic model and didn't have expectations from this model.

6) Gaussian Process Classifier:

This classifier took too long to train giving a score of about 0.8. Hence did not pursue it any further.