PATTERN RECOGNITION AND MACHINE LEARNING

LAB 03

LAB REPORT

QUESTION 1.

- The data was first stored in 'df' through pandas and was preprocessed. The preprocessing includes normalisation, label encoding and splitting it into training and testing sets.

1)

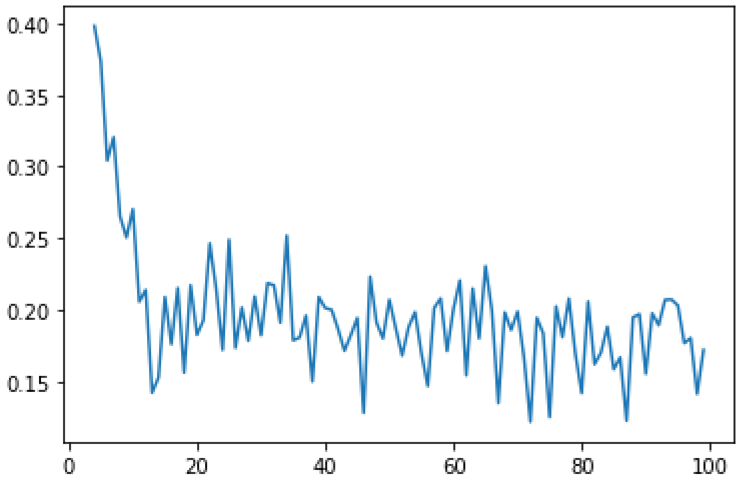
- A simple decision tree regressor was used and the reported r2_score value is 0.47 and Mean Squared Error is 0.02.
- Since the data is continuous, measures like accuracy cannot be used and r2_score is used as an alternative.

2)

- The x_train and x_test sets were split into 5 data frames and were stored in splitsX and splitsY.
- Decision tree model was trained iteratively over various max_depth values.
- On each iteration, one element from splitsX and splitsY was selected for training a decision tree model and remaining 4 for training it.
- This was repeated for each element in splitsX and splitsY and the model's accuracy was stored in terms of average r2_score.
- The best value of max_depth for highest accuracy was recorded as 4.

3)

- A plot was plotted for average r2_scores vs max_depth values.

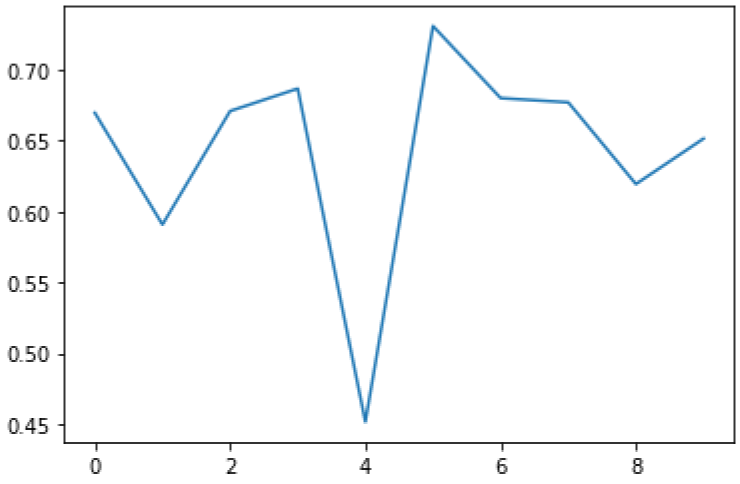


4)

- The core concept of bagging was applied where n_estimator number of dataframes were created.
- This was done by randomly selecting 277 rows from the main dataframe creating a new dataframe out of the 277 rows with replacement.
- The sub-data frames were stored in trainXList and trainYList.

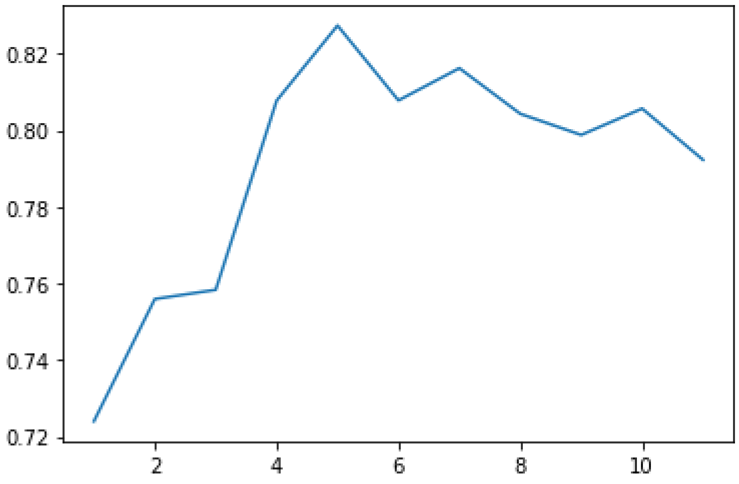
5) AND 6)

- n_estimators number of decision trees were trained on the basis of trainXList and trainYList.
- y_test and their prediction for x_test were compared and respective r2_score values were stored.
- A list 'ratio' is made with normalised values of stores r2_values.
- Ratio list is to determine the weight to be given to each decision tree.
- A plot r2_score values vs decision trees is plotted.



7) AND 8)

- Using the list ‘ratio’ as weights to the decision trees, predictions were made for x_test and compared to y_test.
- The r2_score values were analysed along with various max_depth.
- A plot r2_score vs max_depth was plotted for visualisation.



- r2_score values appear to peak at max_depth = 5, which is 1 more than bestDepth calculated earlier.

9)

- A Random Forest Regressor was made and trained for x_train and y_train.
- Predictions were made for x_test and were compared to y_test.
- The below values were obtained.
- `r2_score :             0.5042188651739021`
- `Mean Squared Error :   0.018784959598196864`
- `Mean Absolute Error :  0.10125250106993013`

10)

- A AdaBoost Regressor was made and trained for x_train and y_train.
- Predictions were made for x_test and were compared to y_test.
- The below values were obtained.
- `r2_score :             0.543601471860709`
- `Mean Squared Error :   0.017292767532956548`
- `Mean Absolute Error :  0.10161112490193527`

QUESTION 2.

- The data was first stored in ‘df’ through pandas and was preprocessed. The preprocessing includes normalisation, label encoding and splitting it into training and testing sets.

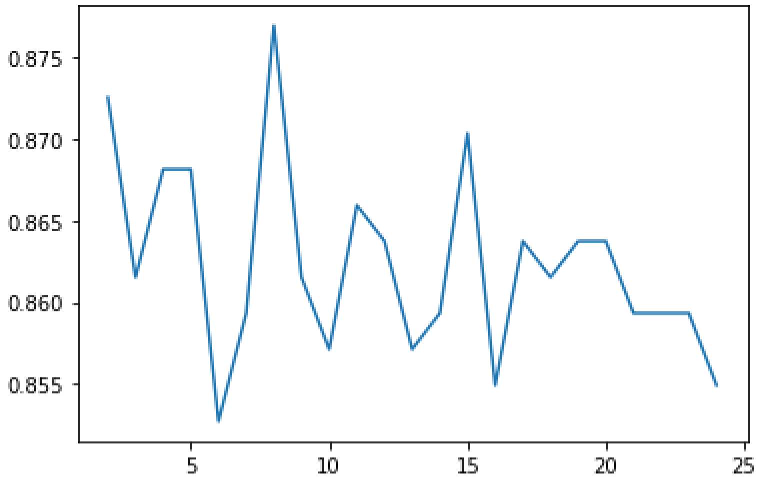
1)

- A simple decision tree regressor was used and the reported accuracy value is 0.895.

2)

- The x_train and x_test sets were split into 5 data frames and were stored in splitsX and splitsY.
- Decision tree model was trained iteratively over various max_depth values.
- On each iteration, one element from splitsX and splitsY was selected for training a decision tree model and remaining 4 for training it.
- This was repeated for each element in splitsX and splitsY and the model's accuracy was stored in terms of average accuracy.
- The best value of max_depth for highest accuracy was recorded as 8.

3)



4)

- Installed xgboost using `!pip install xgboost`
- Applied XGBClassifier and trained it on train data.

5)

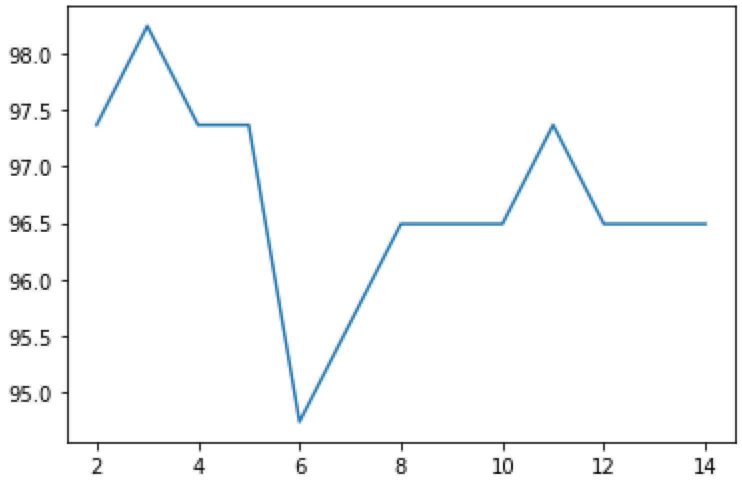
- Calculated the accuracy of the model by comparing its predictions of x_test to y_test and it was recorded as:
- `Accuracy on training data:  96.04395604395604 %`
- `Accuracy on testing data:  92.98245614035088 %`

6)

- Installed lightGBM using `!pip install lightgbm`
- Applied lightgbm and trained it on train data with appropriate parameters.
- Varied num_leaves for fixed max_depth of 3 and got best accuracy for num_leaves = 5.:
- `The value of num_leaves for best result whne tree depth is set to three:  5`

7)

- The model was trained for different values of max_depth and a plot between accuracies vs max_depth was plotted:
- A sharp decline in accuracy is seen at max_depth = 6 and the model does not recover to its original accuracy after.
- Hence, the model starts overfitting at max_depth = 6.



8)

- Pre-pruning and post-pruning techniques can be used to handle the problem of overfitting.
- Just by training, the Random Forest model with the default hyperparameters cannot completely attenuate the problem of overfitting.
- Hyperparameters such as **`max_depth`**, **`min_samples_leaf`**, **`min_samples_split`** can be tuned in order for a good model.
- max_depth is - as its name suggests - the maximum depth of the decision tree design.
- min_samples_leaf guarantees a minimum number of samples allowed in a leaf.
- min_samples_split specifies the minimum number of samples required to split an *internal node*