EAS 506 Statistical Data Mining — Jaideep Reddy Kommera

Homework 4 — Class no. 28

Question 1:

Introduction: The 'prostate' dataset has 97 rows and 9 variables. The response variable is 'lpsa'.

Pre-Processing: Dropped the trivial 'train' column from the data frame.

Train-Test Split: 80% Train data

20% Test data

Best (Exhaustive) Subset Selection:

y- 'lpsa'

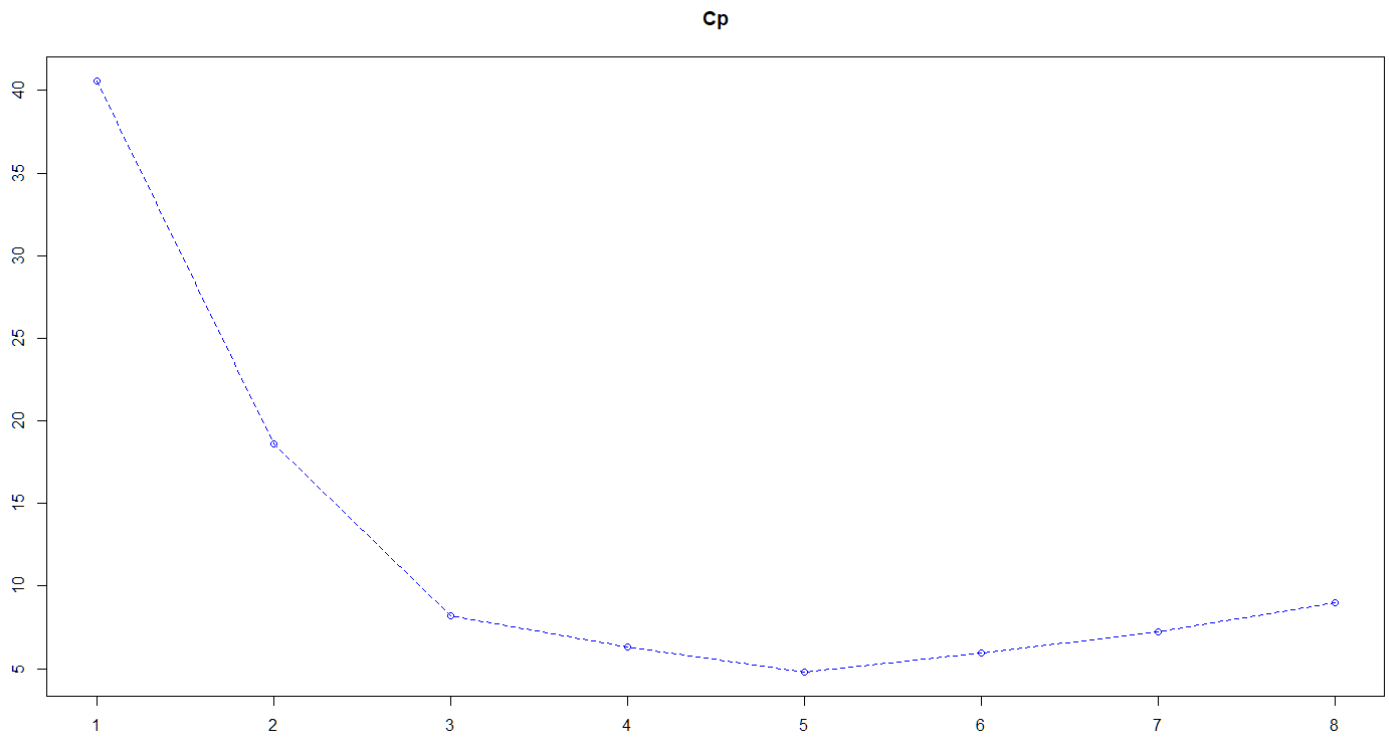
data- training data

nvmax=9

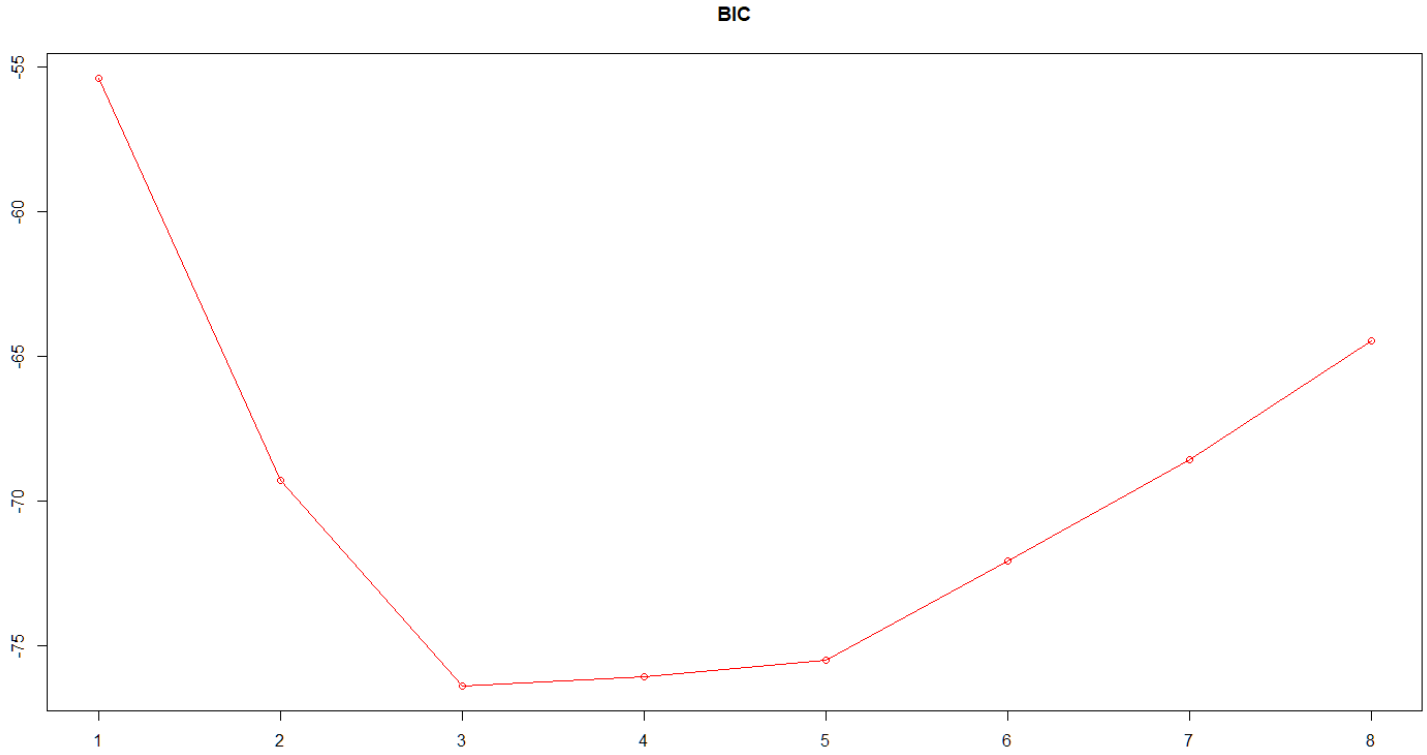
Minimum Cp- **5** variable subset

Minimum BIC- **3** variable subset

Plotting Cp:



Plotting BIC:



Prediction Error:

Cp- 0.8800252 (MSE)

BIC - 0.7951412 (MSE)

Cross Validation:

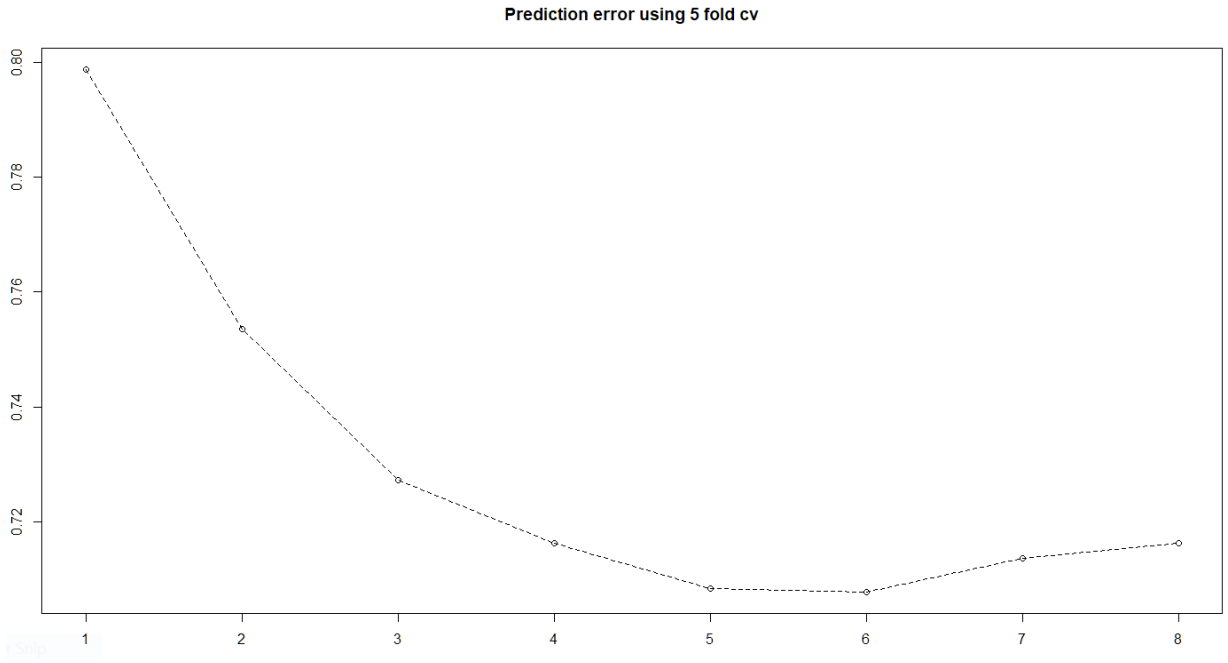
Step 1: Exhaustive subset selection

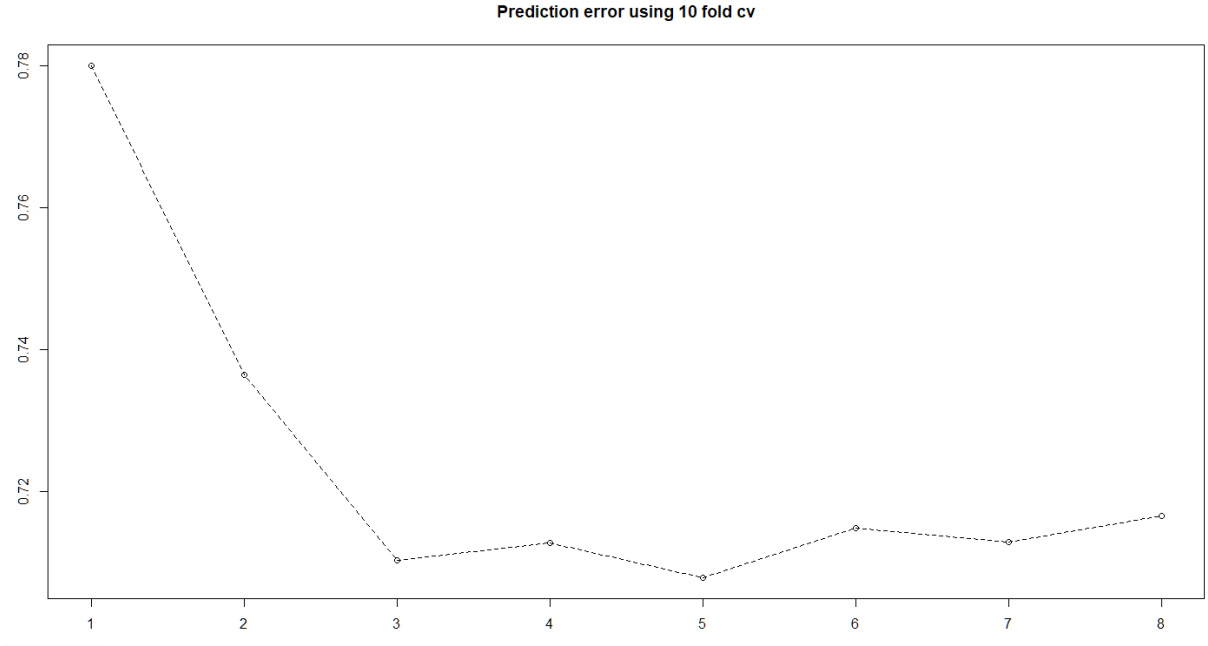
Step 2: Creating empty vectors to store 5-fold and 10-fold cross validation errors

Step 3: Extracting predictors one by one from summary of best subset model for all possible subsets

Step 4: Cross validation and prediction using Linear Regression for all possible subsets

Step 5: Plotting the prediction error





RMSE plot for Best Subset Linear Regression 10-fold CV

Bootstrap .632

Prediction error by using bootstrap method

```
> error_store
[1] 0.6390077 0.5628378 0.5227117 0.5261264 0.5332319
[6] 0.5268610 0.5257030 0.5426064
```

Question 2:

Introduction: The wine dataset has the results of a chemical analysis of 178 wines grown over the decade 1970-1979 in the same region of Italy. The dataset has 178 observations and 14 variables.

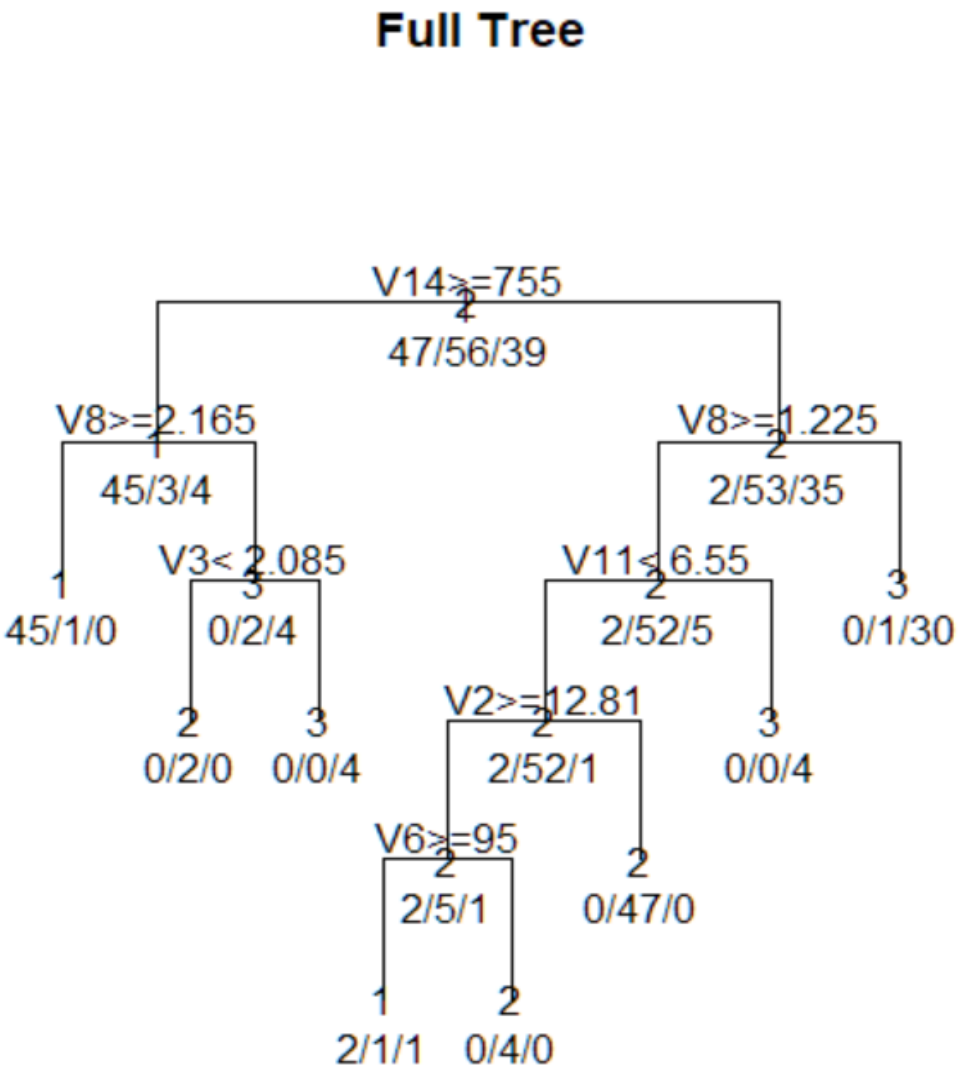
Train-Test split: The dataset has been split into train and test. 80% of the data has been allotted to training and 20% has been allotted to testing. set.seed() has been used before splitting data.

Growing Full Tree:

minsplit=5

initially Cp=0

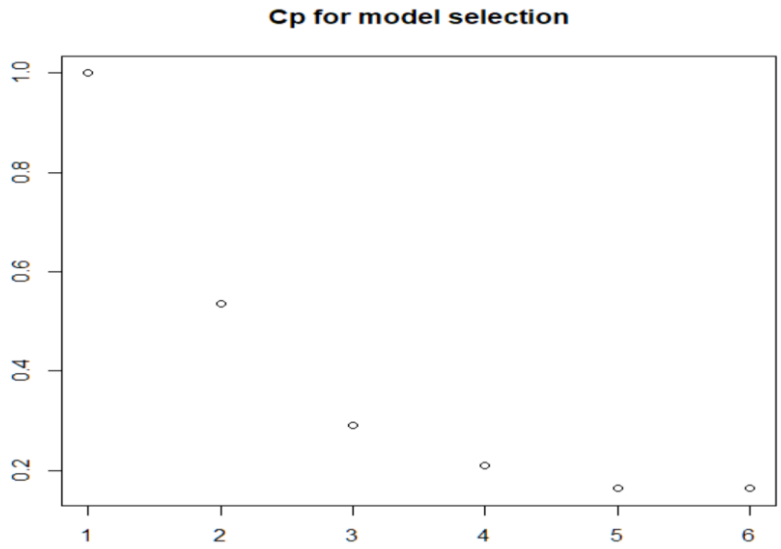
Target variable = 'V1'



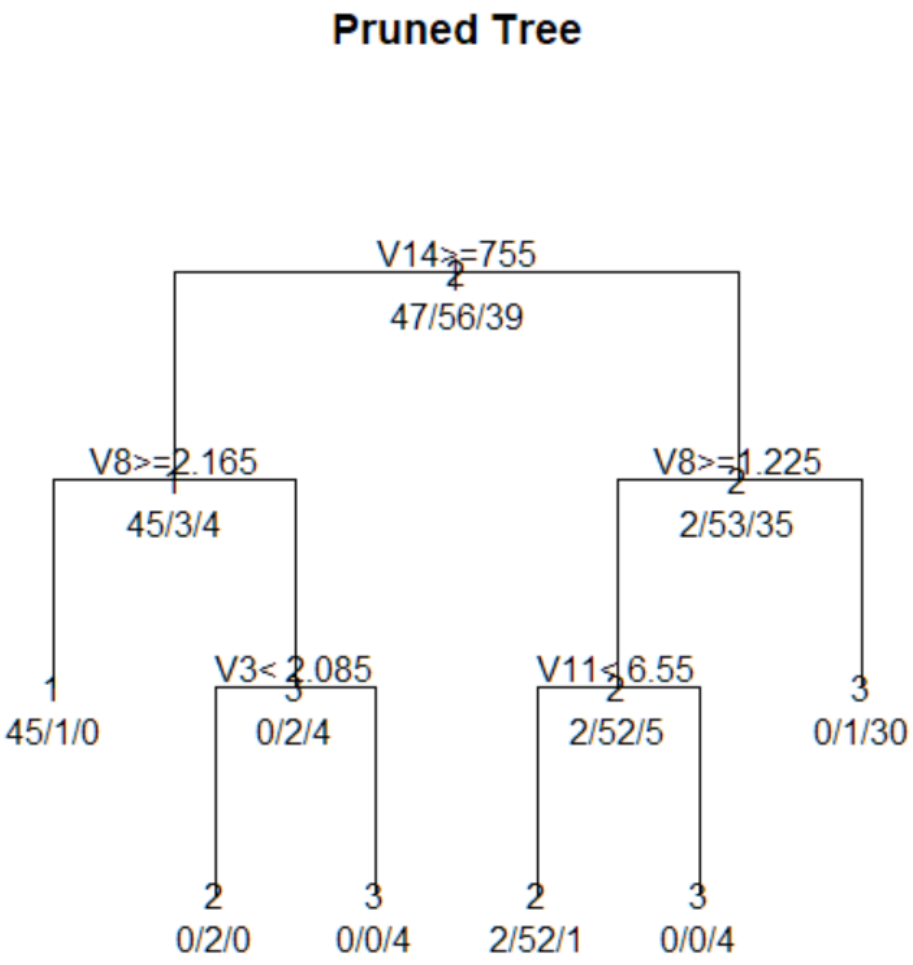
Pruning Tree:

Selecting the Cp value corresponding to the minimum cross validation error(xerror)

4th column of the cp table to get cross validation error



From the above plot we can see that Cp is minimum for a 5-variable subset.



Prediction Error of Pruned tree: 0.083333

Training samples falling at each node:

```
> trainnodes
 55  46  98  10  82  84 140  64  93  29 105 148  47  66 126 110  34  58 161 163  85 112 173 117  65  27 118 134  83  42  73 137
  4   4  12   4  12  12   7  12  12   4  12   7   4  12  12  12   4   4   7   7  12  12   7  12  12   4  12   7  12   4  12   7
 51 139 101 128  26  89 145  19 177 119 106 157  81 165 104 116  28  40  43 142  30  35  74  32  16 130  72 127 178  76 135  78
  4   7  12  12   4  12  11   4  11  12  12   7  12   7  12  12   4   4   4  11   4   4   4   4   4  12  12  12   7  12   7  12
146  41  52  50 153 155  45  36  61 102  69 154  88  79 156 175 138  59 141  94   4  54  68  23 121 133  80 172  31  39  77  33
 11   4   4   4  13   7   4   4   7  12  12   7  12  12   7   7   7   4   7  12   4   4  12   4  12   7  12   7   4   4  12   4
143  11   3 136 147 164  95  90 174 111  70 167   1  37  57 152   6 122  62 168  92 108  13  71  24  91 159 150  20   2 131  49
  7   4   4   7   7   7  12  12   7  12  12   7   4   4   4  13   4  12  12   7  12  12   4  10   4  12  13  13   4   4  12   4
 75   5  12 113 162 171 115 120 170  53  87  25  44 109
 10  12   4  12   7   7  12  12   7   4  12   4  12  12
```

Total sum of training samples at each node:

```
  4   7  10  11  12  13
 46  31   2   4  55   4
```

-node number

- number of train samples

Testing samples falling at each node:

```
> testnodes
  7   8   9  14  15  17  18  21  22  38  48  56  60  63  67  86  96  97  99 100 103 107 114 123 124 125 129 132 144 149 151 158
  4   4   4   4   4   4   4   4   4   4   4   4   7  12  12  12   4   7  12  12  12  12  12  12  12  12  12   7   7   7  13  11
160 166 169 176
  7   7   7  11
```

Total sum of testing samples at each node:

```
  4   7  11  12  13
 13   8   2  12   1
```

-node number

- number of test samples

Approach:

The full tree is first built and then pruning is done by selecting the Cp corresponding to the minimum cross validation error.

The tree is pruned to generalize the fitting and improve the prediction on the test data.

Pruning is done at node V2>=12.81.

The test prediction error on the pruned tree is 0.08333.

Question 3:

Introduction: The dataset I have chosen is the Pima Indian Diabetics dataset. This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. The dataset has 768 observations and 9 variables. Target variable is 'Outcome'.

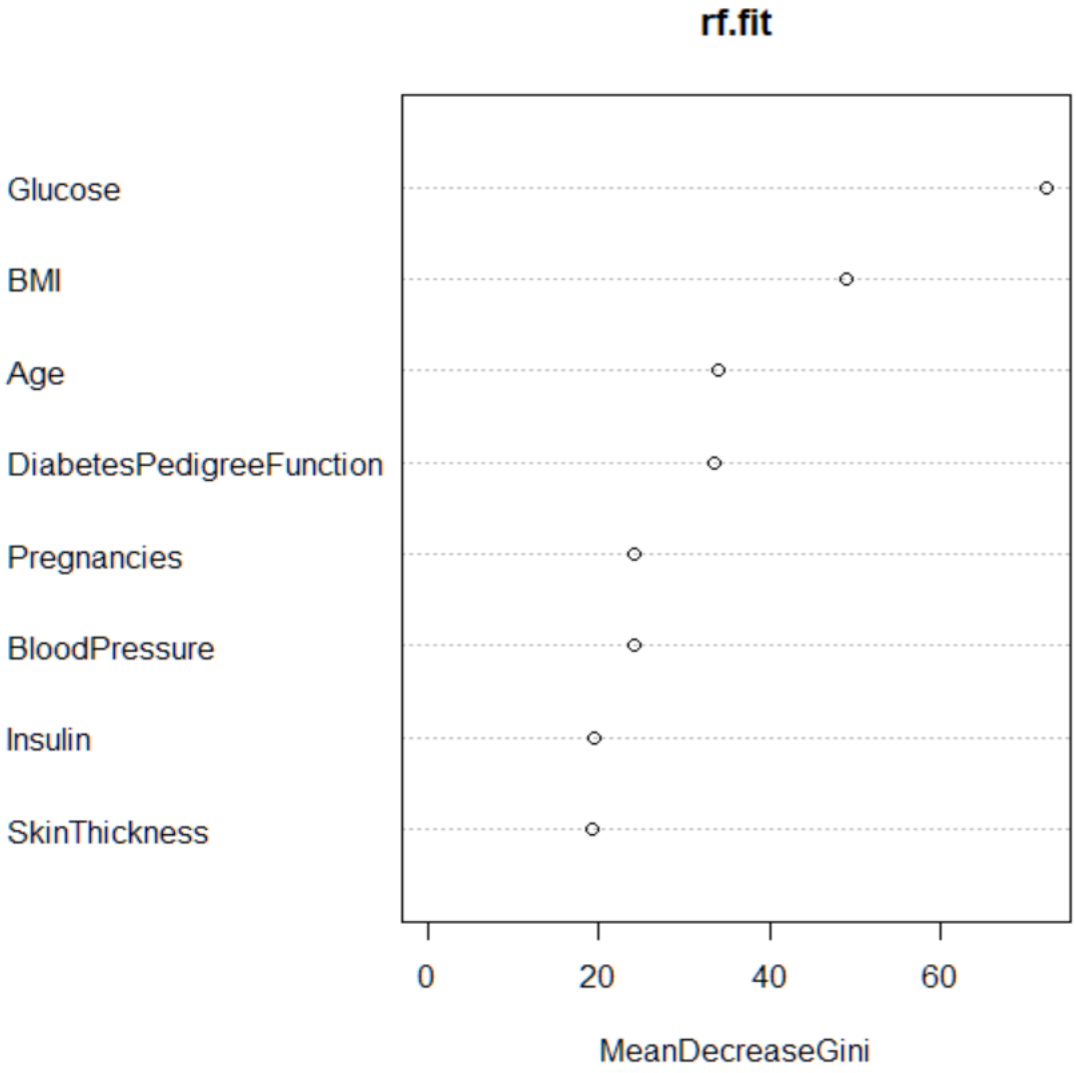
Train-Test split: The dataset has been split into train and test. 80% of the data has been allotted to training and 20% has been allotted to testing. set.seed() has been used before splitting data.

Random Forest:

Target- 'Outcome'

Data- Train data

Number of trees: 10000



Variable Importance Plot

Prediction Error Random Forest: 0.2922078

Bagging:

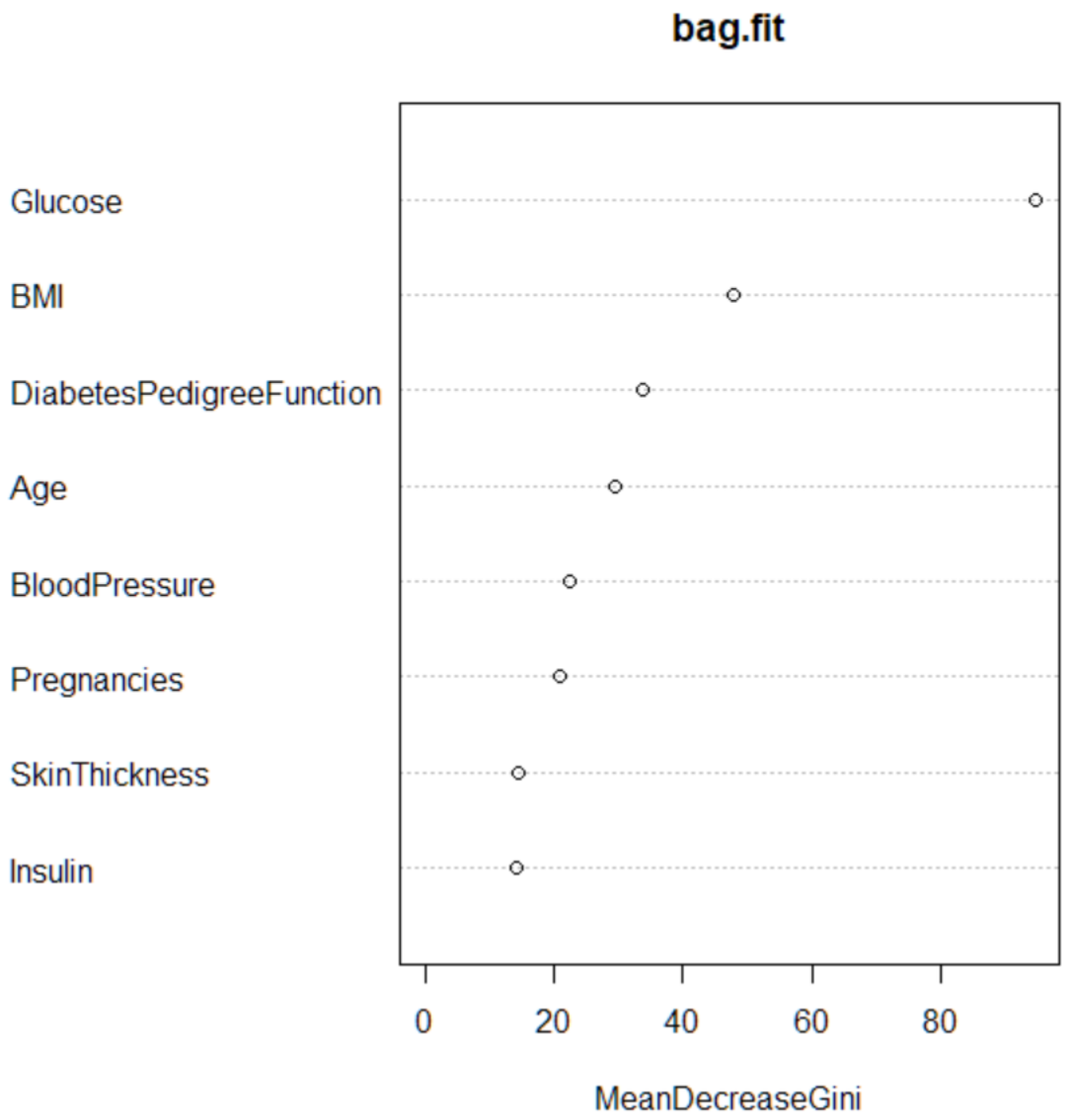
Target- ‘Outcome’

Data- Train data

Number of trees: 10000

Number of variables at each split(mtry): 8 (considering all the predictors)

By default, ‘mtry’ is square root of total number of predictors



Variable importance plot given by bagging

Prediction Error Bagging: 0.2792208

Boosting:

Target- 'Outcome'
Data- Train data
Number of trees: 100
Shrinkage: 0.1, 0.6
Interaction depth: 6
Method: Adaboost

Boosting Model 1:

Shrinkage 0.1

Prediction error: 0.288665

Boosting Model 2:

Shrinkage 0.6

Prediction error: 0.2940948

Logistic Regression:

```
Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)              -9.1680973  0.8539162 -10.737  < 2e-16 ***
Pregnancies               0.1353106  0.0364886   3.708 0.000209 ***
Glucose                   0.0359749  0.0042084   8.548  < 2e-16 ***
BloodPressure            -0.0107389  0.0057341  -1.873 0.061093 .
SkinThickness            -0.0049932  0.0075933  -0.658 0.510806
Insulin                  -0.0010606  0.0009782  -1.084 0.278295
BMI                       0.1067989  0.0178836   5.972 2.35e-09 ***
DiabetesPedigreeFunction  1.0779517  0.3420328   3.152 0.001624 **
Age                       0.0109070  0.0102919   1.060 0.289249
```

Significant variables from logistic regression can be seen from the table above

Prediction error Logistic Regression: 0.2532468

Model	Prediction Error
Random Forest	0.2922078
Bagging	0.2792208
Boosting (0.1 shrinkage)	0.288665
Logistic Regression	0.2532468

Inference**:** Logistic Regression is performing better than the ensemble models. It may be due to the data being linearly separable. When the independent variables are categorical, random forest tends to perform better than logistic regression. With continuous variables, logistic regression is usually better. In this dataset predictors are continuous therefore, logistic woks better.