**DSBA/MBAD 6211 Assignment 2**

Due: 11:59pm @ 07/14/2020

We will analyze the Boston housing dataset to classify and predict ***whether a property exceeds the median value of owner-occupied home in Boston***. Please only use the binary DV, and discard the continuous DV. Please apply both Random Forest and Support Vector Machines to analyze the dataset.

* **Variable and model naming requirements:**
  + Please include your ***name initials*** to the data frame names as well as model names in your R coding.
  + Please instance, in my coding, I would name the data frames as ***dfKZ, dfKZ.train***, and ***dfKZ.valid.*** I would also name the models as ***RFKZ, SVMKZ***, etc.

Please submit a Word document including answers to the following questions.

1. Summarize the random forest model results

* **What is the optimal parameter mtry?**
  + mtry=4 with 96% of accuracy.
* **Please provide the variable importance ranking. Do they make sense to you?**
* > varImp(rf\_mtry)
* rf variable importance
* Importance
* RM 100.0000
* LSTAT 84.1473
* PTRATIO 33.5089
* INDUS 28.8898
* TAX 20.4746
* ZN 13.3084
* DIS 12.6687
* RAD 11.6174
* CRIM 6.8860
* AGE 3.3567
* NOX 0.2566
* CHAS 0.0000
* >

Yes RM(avg no of rooms) is much more important to decide whether property value exceeds median value. Also other variables like LSTAT, PTRATIO, INDUS are significant too to predict the outcome variable.

* **Please provide the confusion matrix for the test dataset**
* > confusionMatrix(prediction,dfNK.test$CAT..MEDV)
* Confusion Matrix and Statistics
* Reference
* Prediction 0 1
* 0 123 3
* 1 3 22
* Accuracy : 0.9603
* 95% CI : (0.9155, 0.9853)
* No Information Rate : 0.8344
* P-Value [Acc > NIR] : 1.531e-06
* Kappa : 0.8562
* Mcnemar's Test P-Value : 1
* Sensitivity : 0.9762
* Specificity : 0.8800
* Pos Pred Value : 0.9762
* Neg Pred Value : 0.8800
* Prevalence : 0.8344
* Detection Rate : 0.8146
* Detection Prevalence : 0.8344
* Balanced Accuracy : 0.9281
* 'Positive' Class : 0

1. For SVM, please compare results for two kernel functions, linear vs. radial. Which kernel function does lead to a better model performance in the default setting?

* Below is the confusionMatrix for Linear Vs Radial with default setting. Based on the accuracy Radial is better than Linear.

|  |  |
| --- | --- |
| Linear | Radial |
| > confusionMatrix(linear\_pred, dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 119 5  1 7 20    Accuracy : 0.9205  95% CI : (0.8653, 0.9583)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 0.001589    Kappa : 0.7213    Mcnemar's Test P-Value : 0.772830    Sensitivity : 0.9444  Specificity : 0.8000  Pos Pred Value : 0.9597  Neg Pred Value : 0.7407  Prevalence : 0.8344  Detection Rate : 0.7881  Detection Prevalence : 0.8212  Balanced Accuracy : 0.8722    'Positive' Class : 0 | > confusionMatrix(radial\_pred, dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 122 4  1 4 21    Accuracy : 0.947  95% CI : (0.8983, 0.9769)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 2.458e-05    Kappa : 0.8083    Mcnemar's Test P-Value : 1    Sensitivity : 0.9683  Specificity : 0.8400  Pos Pred Value : 0.9683  Neg Pred Value : 0.8400  Prevalence : 0.8344  Detection Rate : 0.8079  Detection Prevalence : 0.8344  Balanced Accuracy : 0.9041    'Positive' Class : 0 |
|  |  |

1. **Please tune the SVM model with selected kernel function, and identify best parameters.**

* **Please provide the confusion matrix for the test dataset**

> confusionMatrix(radial\_tune\_pred,dfNK.test$CAT..MEDV)

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 122 10

1 4 15

Accuracy : 0.9073

95% CI : (0.8493, 0.9484)

No Information Rate : 0.8344

P-Value [Acc > NIR] : 0.007493

Kappa : 0.6287

Mcnemar's Test P-Value : 0.181449

Sensitivity : 0.9683

Specificity : 0.6000

Pos Pred Value : 0.9242

Neg Pred Value : 0.7895

Prevalence : 0.8344

Detection Rate : 0.8079

Detection Prevalence : 0.8742

Balanced Accuracy : 0.7841

'Positive' Class : 0

* **What is the optimal parameter(s) for the selected kernel function**
* sigma = 0.5 and C = 1.

1. **Overall, which method (random forest vs. SVM) is performing better?**

* Random forest is performing better with 96% accuracy while SVM with default setting for both kernel (Linear and Radial) accuracy level are 92% and 94% resp. Even with tuning Radial model the accuracy is 90%

1. **Attach your R codes at the end of the WORD document.**

* **Code attached below**

Data dictionary:

* CRIM per capita crime rate by town
* ZN proportion of residential land zoned for lots over 25,000 sq.ft.
* INDUS proportion of non-retail business acres per town.
* CHAS Charles River dummy variable (1 if tract bounds river; 0 otherwise)
* NOX nitric oxides concentration (parts per 10 million)
* RM average number of rooms per dwelling
* AGE proportion of owner-occupied units built prior to 1940
* DIS weighted distances to five Boston employment centers
* RAD index of accessibility to radial highways
* TAX full-value property-tax rate per $10,000
* PTRATIO pupil-teacher ratio by town
* LSTAT % lower status of the population
* MEDV Median value of owner-occupied homes in $1000
* CAT.MEDV binary indicator whether a home’s value exceeds the median, 1 yes and 0 no

|  |
| --- |
| #################### RANDOM FOREST #############################################  #install.packages('ISLR')  #install.packages('randomForest')  library(ISLR)  library(randomForest)  library(caret)  setwd("C:\\Users\\P2190101\\Desktop\\NK Personal\\NK Study\\UNCC\\6211 - Advanced BI\\Assignment 2")  getwd()  dfNK <- read.csv('BostonHousing.csv',na.strings = c('NA','')) # Read the csv file  summary(dfNK)  str(dfNK)  dfNK <- dfNK[-c(13)]  head(dfNK)  dfNK$CAT..MEDV <- factor(dfNK$CAT..MEDV)  set.seed(101)  trainIndex <- createDataPartition(dfNK$CAT..MEDV,  p=0.7,  list=FALSE,  times=1)  dfNK.train <-dfNK[trainIndex,]  dfNK.test <- dfNK[-trainIndex,]  rf\_NK <- train(CAT..MEDV~., # Default Model  data=dfNK.train,  method='rf',  metric='Accuracy',  ntree=100)  print(rf\_NK)  tuneGrid <- expand.grid(.mtry=c(1:12)) # mtry is number of variables used in each tree. Only max total independent variables  rf\_mtry <- train(CAT..MEDV~.,  data=dfNK.train,  method='rf',  metric='Accuracy',  tuneGrid=tuneGrid,  importance=TRUE,  ntree=100)  print(rf\_mtry)  varImp(rf\_mtry)  prediction <- predict(rf\_mtry,dfNK.test)  confusionMatrix(prediction,dfNK.test$CAT..MEDV) |

|  |
| --- |
| > #################### RANDOM FOREST #############################################  >  > #install.packages('ISLR')  > #install.packages('randomForest')  >  >  > library(ISLR)  > library(randomForest)  > library(caret)  >  > setwd("C:\\Users\\P2190101\\Desktop\\NK Personal\\NK Study\\UNCC\\6211 - Advanced BI\\Assignment 2")  >  > getwd()  [1] "C:/Users/P2190101/Desktop/NK Personal/NK Study/UNCC/6211 - Advanced BI/Assignment 2"  >  > dfNK <- read.csv('BostonHousing.csv',na.strings = c('NA','')) # Read the csv file  >  > summary(dfNK)  CRIM ZN INDUS CHAS NOX RM AGE DIS  Min. : 0.00632 Min. : 0.00 Min. : 0.46 Min. :0.00000 Min. :0.3850 Min. :3.561 Min. : 2.90 Min. : 1.130  1st Qu.: 0.08205 1st Qu.: 0.00 1st Qu.: 5.19 1st Qu.:0.00000 1st Qu.:0.4490 1st Qu.:5.886 1st Qu.: 45.02 1st Qu.: 2.100  Median : 0.25651 Median : 0.00 Median : 9.69 Median :0.00000 Median :0.5380 Median :6.208 Median : 77.50 Median : 3.207  Mean : 3.61352 Mean : 11.36 Mean :11.14 Mean :0.06917 Mean :0.5547 Mean :6.285 Mean : 68.57 Mean : 3.795  3rd Qu.: 3.67708 3rd Qu.: 12.50 3rd Qu.:18.10 3rd Qu.:0.00000 3rd Qu.:0.6240 3rd Qu.:6.623 3rd Qu.: 94.08 3rd Qu.: 5.188  Max. :88.97620 Max. :100.00 Max. :27.74 Max. :1.00000 Max. :0.8710 Max. :8.780 Max. :100.00 Max. :12.127  RAD TAX PTRATIO LSTAT MEDV CAT..MEDV  Min. : 1.000 Min. :187.0 Min. :12.60 Min. : 1.73 Min. : 5.00 Min. :0.000  1st Qu.: 4.000 1st Qu.:279.0 1st Qu.:17.40 1st Qu.: 6.95 1st Qu.:17.02 1st Qu.:0.000  Median : 5.000 Median :330.0 Median :19.05 Median :11.36 Median :21.20 Median :0.000  Mean : 9.549 Mean :408.2 Mean :18.46 Mean :12.65 Mean :22.53 Mean :0.166  3rd Qu.:24.000 3rd Qu.:666.0 3rd Qu.:20.20 3rd Qu.:16.95 3rd Qu.:25.00 3rd Qu.:0.000  Max. :24.000 Max. :711.0 Max. :22.00 Max. :37.97 Max. :50.00 Max. :1.000  > str(dfNK)  'data.frame': 506 obs. of 14 variables:  $ CRIM : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...  $ ZN : num 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...  $ INDUS : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...  $ CHAS : int 0 0 0 0 0 0 0 0 0 0 ...  $ NOX : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...  $ RM : num 6.58 6.42 7.18 7 7.15 ...  $ AGE : num 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...  $ DIS : num 4.09 4.97 4.97 6.06 6.06 ...  $ RAD : int 1 2 2 3 3 3 5 5 5 5 ...  $ TAX : int 296 242 242 222 222 222 311 311 311 311 ...  $ PTRATIO : num 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...  $ LSTAT : num 4.98 9.14 4.03 2.94 5.33 ...  $ MEDV : num 24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...  $ CAT..MEDV: int 0 0 1 1 1 0 0 0 0 0 ...  >  > dfNK <- dfNK[-c(13)]    > head(dfNK)  CRIM ZN INDUS CHAS NOX RM AGE DIS RAD TAX PTRATIO LSTAT CAT..MEDV  1 0.00632 18 2.31 0 0.538 6.575 65.2 4.0900 1 296 15.3 4.98 0  2 0.02731 0 7.07 0 0.469 6.421 78.9 4.9671 2 242 17.8 9.14 0  3 0.02729 0 7.07 0 0.469 7.185 61.1 4.9671 2 242 17.8 4.03 1  4 0.03237 0 2.18 0 0.458 6.998 45.8 6.0622 3 222 18.7 2.94 1  5 0.06905 0 2.18 0 0.458 7.147 54.2 6.0622 3 222 18.7 5.33 1  6 0.02985 0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 5.21 0  >  > dfNK$CAT..MEDV <- factor(dfNK$CAT..MEDV)  >  >  > set.seed(101)  >  > trainIndex <- createDataPartition(dfNK$CAT..MEDV,  + p=0.7,  + list=FALSE,  + times=1)  >  > dfNK.train <-dfNK[trainIndex,]  > dfNK.test <- dfNK[-trainIndex,]  >  >  > rf\_NK <- train(CAT..MEDV~., # Default Model  + data=dfNK.train,  + method='rf',  + metric='Accuracy',  + ntree=100)  > print(rf\_NK)  Random Forest  355 samples  12 predictor  2 classes: '0', '1'  No pre-processing  Resampling: Bootstrapped (25 reps)  Summary of sample sizes: 355, 355, 355, 355, 355, 355, ...  Resampling results across tuning parameters:  mtry Accuracy Kappa  2 0.9524000 0.8164894  7 0.9511090 0.8130799  12 0.9447922 0.7873665  Accuracy was used to select the optimal model using the largest value.  The final value used for the model was mtry = 2.  >  >  > tuneGrid <- expand.grid(.mtry=c(1:12)) # mtry is number of variables used in each tree. Only max total independent variables  >  > rf\_mtry <- train(CAT..MEDV~.,  + data=dfNK.train,  + method='rf',  + metric='Accuracy',  + tuneGrid=tuneGrid,  + importance=TRUE,  + ntree=100)  > print(rf\_mtry)  Random Forest  355 samples  12 predictor  2 classes: '0', '1'  No pre-processing  Resampling: Bootstrapped (25 reps)  Summary of sample sizes: 355, 355, 355, 355, 355, 355, ...  Resampling results across tuning parameters:  mtry Accuracy Kappa  1 0.9391519 0.7574420  2 0.9509436 0.8157460  3 0.9525893 0.8249330  4 0.9540867 0.8309783  5 0.9523132 0.8239666  6 0.9503774 0.8152400  7 0.9486603 0.8084765  8 0.9465340 0.8015361  9 0.9448684 0.7932848  10 0.9430421 0.7863696  11 0.9415371 0.7800662  12 0.9390515 0.7691342  Accuracy was used to select the optimal model using the largest value.  The final value used for the model was mtry = 4.  >  > varImp(rf\_mtry)  rf variable importance  Importance  RM 100.0000  LSTAT 84.1473  PTRATIO 33.5089  INDUS 28.8898  TAX 20.4746  ZN 13.3084  DIS 12.6687  RAD 11.6174  CRIM 6.8860  AGE 3.3567  NOX 0.2566  CHAS 0.0000  >  > prediction <- predict(rf\_mtry,dfNK.test)  > confusionMatrix(prediction,dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 123 3  1 3 22    Accuracy : 0.9603  95% CI : (0.9155, 0.9853)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 1.531e-06    Kappa : 0.8562    Mcnemar's Test P-Value : 1    Sensitivity : 0.9762  Specificity : 0.8800  Pos Pred Value : 0.9762  Neg Pred Value : 0.8800  Prevalence : 0.8344  Detection Rate : 0.8146  Detection Prevalence : 0.8344  Balanced Accuracy : 0.9281    'Positive' Class : 0 |

|  |
| --- |
| #################### SVM #############################################  library(caret)  library(ggplot2)  setwd("C:\\Users\\P2190101\\Desktop\\NK Personal\\NK Study\\UNCC\\6211 - Advanced BI\\Assignment 2")  getwd()  dfNK <- read.csv('BostonHousing.csv',na.strings = c('NA','')) # Read the csv file  summary(dfNK)  str(dfNK)  dfNK <- dfNK[-c(13)]  dfNK$CAT..MEDV <- factor(dfNK$CAT..MEDV)  str(dfNK)  set.seed(101)  trainIndex <- createDataPartition(dfNK$CAT..MEDV,  p=0.7,  list=FALSE,  times=1)  dfNK.train <-dfNK[trainIndex,]  dfNK.test <- dfNK[-trainIndex,]  trControl <- trainControl(method='cv', # Cross Validation - 10 fold here  number=10,  search='grid',  )  #######################################################################################################  svm\_linear <- train(CAT..MEDV~.,  data=dfNK.train,  method='svmLinear',  trControl=trControl,  preProcess=c('center','scale'))  print(svm\_linear)  linear\_pred <- predict(svm\_linear,dfNK.test)  confusionMatrix(linear\_pred, dfNK.test$CAT..MEDV)  #######################################################################################################  svm\_radial <- train(CAT..MEDV~.,  data=dfNK.train,  method='svmRadial',  trControl=trControl,  preProcess=c('center','scale'))  print(svm\_radial)  radial\_pred <- predict(svm\_radial,dfNK.test)  confusionMatrix(radial\_pred, dfNK.test$CAT..MEDV)  grid\_radial <- expand.grid(sigma=c(0,0.5,0.75,1,1.3,1.5),  C=c(0,0.05,0.25,0.5,0.75,1))  svm\_radial\_tune <- train(CAT..MEDV~.,  data=dfNK.train,  method='svmRadial',  trControl=trControl,  preProcess=c('center','scale'),  tuneGrid=grid\_radial)  print(svm\_radial\_tune)  radial\_tune\_pred <- predict(svm\_radial\_tune,dfNK.test)  confusionMatrix(radial\_tune\_pred,dfNK.test$CAT..MEDV)  ######################################################################################################### |

|  |
| --- |
| > #################### SVM #############################################  > library(caret)  > library(ggplot2)  >  >  > setwd("C:\\Users\\P2190101\\Desktop\\NK Personal\\NK Study\\UNCC\\6211 - Advanced BI\\Assignment 2")  >  > getwd()  [1] "C:/Users/P2190101/Desktop/NK Personal/NK Study/UNCC/6211 - Advanced BI/Assignment 2"  >  > dfNK <- read.csv('BostonHousing.csv',na.strings = c('NA','')) # Read the csv file  >  > summary(dfNK)  CRIM ZN INDUS CHAS NOX RM AGE DIS  Min. : 0.00632 Min. : 0.00 Min. : 0.46 Min. :0.00000 Min. :0.3850 Min. :3.561 Min. : 2.90 Min. : 1.130  1st Qu.: 0.08205 1st Qu.: 0.00 1st Qu.: 5.19 1st Qu.:0.00000 1st Qu.:0.4490 1st Qu.:5.886 1st Qu.: 45.02 1st Qu.: 2.100  Median : 0.25651 Median : 0.00 Median : 9.69 Median :0.00000 Median :0.5380 Median :6.208 Median : 77.50 Median : 3.207  Mean : 3.61352 Mean : 11.36 Mean :11.14 Mean :0.06917 Mean :0.5547 Mean :6.285 Mean : 68.57 Mean : 3.795  3rd Qu.: 3.67708 3rd Qu.: 12.50 3rd Qu.:18.10 3rd Qu.:0.00000 3rd Qu.:0.6240 3rd Qu.:6.623 3rd Qu.: 94.08 3rd Qu.: 5.188  Max. :88.97620 Max. :100.00 Max. :27.74 Max. :1.00000 Max. :0.8710 Max. :8.780 Max. :100.00 Max. :12.127  RAD TAX PTRATIO LSTAT MEDV CAT..MEDV  Min. : 1.000 Min. :187.0 Min. :12.60 Min. : 1.73 Min. : 5.00 Min. :0.000  1st Qu.: 4.000 1st Qu.:279.0 1st Qu.:17.40 1st Qu.: 6.95 1st Qu.:17.02 1st Qu.:0.000  Median : 5.000 Median :330.0 Median :19.05 Median :11.36 Median :21.20 Median :0.000  Mean : 9.549 Mean :408.2 Mean :18.46 Mean :12.65 Mean :22.53 Mean :0.166  3rd Qu.:24.000 3rd Qu.:666.0 3rd Qu.:20.20 3rd Qu.:16.95 3rd Qu.:25.00 3rd Qu.:0.000  Max. :24.000 Max. :711.0 Max. :22.00 Max. :37.97 Max. :50.00 Max. :1.000  > str(dfNK)  'data.frame': 506 obs. of 14 variables:  $ CRIM : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...  $ ZN : num 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...  $ INDUS : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...  $ CHAS : int 0 0 0 0 0 0 0 0 0 0 ...  $ NOX : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...  $ RM : num 6.58 6.42 7.18 7 7.15 ...  $ AGE : num 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...  $ DIS : num 4.09 4.97 4.97 6.06 6.06 ...  $ RAD : int 1 2 2 3 3 3 5 5 5 5 ...  $ TAX : int 296 242 242 222 222 222 311 311 311 311 ...  $ PTRATIO : num 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...  $ LSTAT : num 4.98 9.14 4.03 2.94 5.33 ...  $ MEDV : num 24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...  $ CAT..MEDV: int 0 0 1 1 1 0 0 0 0 0 ...  >  > dfNK <- dfNK[-c(13)]  >    > dfNK$CAT..MEDV <- factor(dfNK$CAT..MEDV)  >  > str(dfNK)  'data.frame': 506 obs. of 13 variables:  $ CRIM : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...  $ ZN : num 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...  $ INDUS : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...  $ CHAS : int 0 0 0 0 0 0 0 0 0 0 ...  $ NOX : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...  $ RM : num 6.58 6.42 7.18 7 7.15 ...  $ AGE : num 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...  $ DIS : num 4.09 4.97 4.97 6.06 6.06 ...  $ RAD : int 1 2 2 3 3 3 5 5 5 5 ...  $ TAX : int 296 242 242 222 222 222 311 311 311 311 ...  $ PTRATIO: num 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...  $ LSTAT : num 4.98 9.14 4.03 2.94 5.33 ...  $ CAT..MEDV : Factor w/ 2 levels "0","1": 1 1 2 2 2 1 1 1 1 1 ...  >  > set.seed(101)  >  > trainIndex <- createDataPartition(dfNK$CAT..MEDV,  + p=0.7,  + list=FALSE,  + times=1)  >  > dfNK.train <-dfNK[trainIndex,]  > dfNK.test <- dfNK[-trainIndex,]  >  >  > trControl <- trainControl(method='cv', # Cross Validation - 10 fold here  + number=10,  + search='grid',  + )  > #######################################################################################################  > svm\_linear <- train(CAT..MEDV~.,  + data=dfNK.train,  + method='svmLinear',  + trControl=trControl,  + preProcess=c('center','scale'))  >  > print(svm\_linear)  Support Vector Machines with Linear Kernel  355 samples  12 predictor  2 classes: '0', '1'  Pre-processing: centered (12), scaled (12)  Resampling: Cross-Validated (10 fold)  Summary of sample sizes: 319, 320, 320, 320, 319, 320, ...  Resampling results:  Accuracy Kappa  0.9462698 0.8020586  Tuning parameter 'C' was held constant at a value of 1  >  > linear\_pred <- predict(svm\_linear,dfNK.test)  > confusionMatrix(linear\_pred, dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 119 5  1 7 20    Accuracy : 0.9205  95% CI : (0.8653, 0.9583)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 0.001589    Kappa : 0.7213    Mcnemar's Test P-Value : 0.772830    Sensitivity : 0.9444  Specificity : 0.8000  Pos Pred Value : 0.9597  Neg Pred Value : 0.7407  Prevalence : 0.8344  Detection Rate : 0.7881  Detection Prevalence : 0.8212  Balanced Accuracy : 0.8722    'Positive' Class : 0    > #######################################################################################################  >  > svm\_radial <- train(CAT..MEDV~.,  + data=dfNK.train,  + method='svmRadial',  + trControl=trControl,  + preProcess=c('center','scale'))  >  > print(svm\_radial)  Support Vector Machines with Radial Basis Function Kernel  355 samples  12 predictor  2 classes: '0', '1'  Pre-processing: centered (12), scaled (12)  Resampling: Cross-Validated (10 fold)  Summary of sample sizes: 319, 320, 320, 319, 319, 319, ...  Resampling results across tuning parameters:  C Accuracy Kappa  0.25 0.9041083 0.5262316  0.50 0.9349813 0.7146518  1.00 0.9687208 0.8779963  Tuning parameter 'sigma' was held constant at a value of 0.1323706  Accuracy was used to select the optimal model using the largest value.  The final values used for the model were sigma = 0.1323706 and C = 1.  >  > radial\_pred <- predict(svm\_radial,dfNK.test)  > confusionMatrix(radial\_pred, dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 122 4  1 4 21    Accuracy : 0.947  95% CI : (0.8983, 0.9769)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 2.458e-05    Kappa : 0.8083    Mcnemar's Test P-Value : 1    Sensitivity : 0.9683  Specificity : 0.8400  Pos Pred Value : 0.9683  Neg Pred Value : 0.8400  Prevalence : 0.8344  Detection Rate : 0.8079  Detection Prevalence : 0.8344  Balanced Accuracy : 0.9041    'Positive' Class : 0    >  > grid\_radial <- expand.grid(sigma=c(0,0.5,0.75,1,1.3,1.5),  + C=c(0,0.05,0.25,0.5,0.75,1))  >  > svm\_radial\_tune <- train(CAT..MEDV~.,  + data=dfNK.train,  + method='svmRadial',  + trControl=trControl,  + preProcess=c('center','scale'),  + tuneGrid=grid\_radial)  There were 50 or more warnings (use warnings() to see the first 50)  > print(svm\_radial\_tune)  Support Vector Machines with Radial Basis Function Kernel  355 samples  12 predictor  2 classes: '0', '1'  Pre-processing: centered (12), scaled (12)  Resampling: Cross-Validated (10 fold)  Summary of sample sizes: 319, 320, 321, 319, 319, 319, ...  Resampling results across tuning parameters:  sigma C Accuracy Kappa  0.00 0.00 NaN NaN  0.00 0.05 0.8338655 0.0000000  0.00 0.25 0.8338655 0.0000000  0.00 0.50 0.8338655 0.0000000  0.00 0.75 0.8338655 0.0000000  0.00 1.00 0.8338655 0.0000000  0.50 0.00 NaN NaN  0.50 0.05 0.8338655 0.0000000  0.50 0.25 0.8533100 0.1659091  0.50 0.50 0.9099953 0.5690671  0.50 0.75 0.9381653 0.7460389  0.50 1.00 0.9436368 0.7778467  0.75 0.00 NaN NaN  0.75 0.05 0.8338655 0.0000000  0.75 0.25 0.8338655 0.0000000  0.75 0.50 0.8844304 0.4162484  0.75 0.75 0.9184827 0.6349792  0.75 1.00 0.9267320 0.6849567  1.00 0.00 NaN NaN  1.00 0.05 0.8338655 0.0000000  1.00 0.25 0.8338655 0.0000000  1.00 0.50 0.8618067 0.2514878  1.00 0.75 0.9041130 0.5506745  1.00 1.00 0.9212605 0.6520246  1.30 0.00 NaN NaN  1.30 0.05 0.8338655 0.0000000  1.30 0.25 0.8338655 0.0000000  1.30 0.50 0.8479178 0.1253515  1.30 0.75 0.8673623 0.2889878  1.30 1.00 0.9126050 0.6018261  1.50 0.00 NaN NaN  1.50 0.05 0.8338655 0.0000000  1.50 0.25 0.8338655 0.0000000  1.50 0.50 0.8421989 0.0750000  1.50 0.75 0.8618067 0.2514878  1.50 1.00 0.8956162 0.5003624  Accuracy was used to select the optimal model using the largest value.  The final values used for the model were sigma = 0.5 and C = 1.  >  > radial\_tune\_pred <- predict(svm\_radial\_tune,dfNK.test)  > confusionMatrix(radial\_tune\_pred,dfNK.test$CAT..MEDV)  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 122 10  1 4 15    Accuracy : 0.9073  95% CI : (0.8493, 0.9484)  No Information Rate : 0.8344  P-Value [Acc > NIR] : 0.007493    Kappa : 0.6287    Mcnemar's Test P-Value : 0.181449    Sensitivity : 0.9683  Specificity : 0.6000  Pos Pred Value : 0.9242  Neg Pred Value : 0.7895  Prevalence : 0.8344  Detection Rate : 0.8079  Detection Prevalence : 0.8742  Balanced Accuracy : 0.7841    'Positive' Class : 0    > ######################################################################################################### |