Homework 10 by Haritha Pulletikurti

2020-10-28

Question 14.1

The breast cancer data set breast-cancer-wisconsin.data.txt from http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/ (description at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29) has missing values.

1. Use the mean/mode imputation method to impute values for the missing data.
2. Use regression to impute values for the missing data.
3. Use regression with perturbation to impute values for the missing data.
4. (Optional) Compare the results and quality of classification models (e.g., SVM, KNN) build using
   (1) the data sets from questions 1,2,3;
   (2) the data that remains after data points with missing values are removed; and
   (3) the data set when a binary variable is introduced to indicate missing values.

Answer:

The breast-cancer-wisconsin data set had missing values.

Removing the missing value rows from the data set is the easiest to implement and does not introduce any errors. But the disadvantage of this approach is we lose many data points and introduce bias.

The ways to deal with missing data by estimating the missing values are called imputation approaches.

First Imputation Approach is using the midrange value – mean, median or mode.

Advantages of this approach is it is easy to compute while the disadvantage is that this method introduces Bias into the data. The mean/median will underestimate the missing value.

The Second Approach is imputing the missing value using Regression.

This approach reduces the problem of the bias, but it is complex to build, fit and validate, test data to estimate the missing value. Also, this does not capture all the variability.

The third approach is Imputation with variability by adding perturbation to each imputed value.

This gives estimates less accurate on average but they give more accurate variability.

Below are the steps I used to answer Question 14.1

Step 1: Read the Cancer data and Add the column names provided in the description of the data set.

Step 2: Printed the summary of the cancer data and found that the column “Bare_Nuclie” having 16 missing values.

Step 3: I divided the dataset into Training and Test Data sets. For this I wrote a Function - Split_data which takes the dataset as input and returns the Training and Test Data sets.

Step 4: Funtion `my_knn_func(Training data, Test data, list of k values)is written which uses` knn3() from the caret library to perform k nearest neighbor function to which predicts the values and gives the best k value and best accuracy of the predictions.

Step 5: I removed all the missing values from the data set and , split the data into training and test data sets and used my_knn_func to compute the best K and best prediction accuracy.

Step 6: Then used the mean value of the “Bare_Nuclie” column and imputed this value to all the missing rows for this column and. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

Step 7: Then used the mode value of the “Bare_Nuclie” column and imputed this value to all the missing rows for this column and. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

Step 8: I used “mice” library and used the method = “norm.predict” to perform regression to impute the missing data. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

Step 9: I used “mice” library and used the `method="norm.nob” to` perform regression with perturbation to impute the missing data. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

Step 10: I provided the results and analysis at the end.

Step 1: Read the Cancer data and Add the column names provided in the description of the data set.

```
rm(list=ls())
cancerdata <- read.table("breast-cancer-wisconsin.data.txt",stringsAsFactors
= FALSE,header= FALSE,
                         na.strings = "?", sep=",")

# give meaningful names to the data

colnames(cancerdata)<-c("ID", "Clump_Thickness", "Cell_Size",
                        "Cell_Shape", "Marginal_Adhesion", "Single_Epith_Cell
_Size",
                        "Bare_Nuclei",  "Bland_Chromatin","Normal_Nucleoli",
"Mitoses", "Class")

cancerdata$Class <- as.factor(cancerdata$Class)
levels(cancerdata$Class) <-c(0,1)
#Find the missing data
head(cancerdata)

##        ID Clump_Thickness Cell_Size Cell_Shape Marginal_Adhesion
## 1 1000025               5         1          1                 1
## 2 1002945               5         4          4                 5
## 3 1015425               3         1          1                 1
## 4 1016277               6         8          8                 1
## 5 1017023               4         1          1                 3
## 6 1017122               8        10         10                 8
##   Single_Epith_Cell_Size Bare_Nuclei Bland_Chromatin Normal_Nucleoli Mitos
es
## 1                      2           1               3               1
1
## 2                      7          10               3               2
1
## 3                      2           2               3               1
1
## 4                      3           4               3               7
1
## 5                      2           1               3               1
1
## 6                      7          10               9               7
1
##   Class
## 1     0
## 2     0
## 3     0
## 4      0
## 5      0
## 6      1
```

Step 2: Printed the summary of the cancer data and found that the column “Bare_Nuclie” having 16 missing values.

```
summary(cancerdata)

##        ID            Clump_Thickness    Cell_Size        Cell_Shape
##  Min.   :   61634   Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
##  1st Qu.:  870688   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 1.000
##  Median : 1171710   Median : 4.000   Median : 1.000   Median : 1.000
##  Mean   : 1071704   Mean   : 4.418   Mean   : 3.134   Mean   : 3.207
##  3rd Qu.: 1238298   3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 5.000
##  Max.   :13454352   Max.   :10.000   Max.   :10.000   Max.   :10.000
##
##  Marginal_Adhesion Single_Epith_Cell_Size  Bare_Nuclei     Bland_Chromatin
##  Min.   : 1.000    Min.   : 1.000         Min.   : 1.000   Min.   : 1.000
##  1st Qu.: 1.000    1st Qu.: 2.000         1st Qu.: 1.000   1st Qu.: 2.000
##  Median : 1.000    Median : 2.000         Median : 1.000   Median : 3.000
##  Mean   : 2.807    Mean   : 3.216         Mean   : 3.545   Mean   : 3.438
##  3rd Qu.: 4.000    3rd Qu.: 4.000         3rd Qu.: 6.000   3rd Qu.: 5.000
##  Max.   :10.000    Max.   :10.000         Max.   :10.000   Max.   :10.000
##                                           NA's   :16
##  Normal_Nucleoli     Mitoses       Class
##  Min.   : 1.000   Min.   : 1.000   0:458
##  1st Qu.: 1.000   1st Qu.: 1.000   1:241
##  Median : 1.000   Median : 1.000
##  Mean   : 2.867   Mean   : 1.589
##  3rd Qu.: 4.000   3rd Qu.: 1.000
##  Max.   :10.000   Max.   :10.000
##

# Bare Nuclie column has 16 missing values
```

Step 3: I divided the dataset into Training and Test Data sets. For this I wrote a Function - Split_data which takes the dataset as input and returns the Training and Test Data sets.

```
#Divide the data into training and testing sets
split_data <- function(cancerdata)
{
  set.seed(499)
  cancerdata <- subset(cancerdata,select=-c(ID))
  samplesize <- floor(0.75 * nrow(cancerdata))
  Train_indices <- sample(seq_len(nrow(cancerdata)),size = samplesize)
  trainingdata <- cancerdata[Train_indices,]
  testingdata <- cancerdata[-Train_indices,]

  return(list("Training" = trainingdata, "Test" = testingdata))
}
```

Step 4: Funtion my_knn_func(Training data, Test data, list of k values)is written which uses knn3() from the caret library to perform k nearest neighbor function to which predicts the values and gives the best k value and best accuracy of the predictions.

```
#Function to predict the best accuracy using
#K nearest neighbour fuction which takes input as Training set and Test Data
set and list of k values.
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

my_knn_func <- function(TrainingSet, TestSet,list_of_K)
{
  Best_Accuracy = 0; Best_K = 0

  for(k_value in list_of_K)
  {
    knnmodel <- knn3(Class~., data = TrainingSet, k = k_value)
    prediction_vals <- predict(knnmodel, TestSet[0:(length(TestSet)-1)], type
= "class")
    prediction_accuracy <- round((sum(prediction_vals == TestSet$Class)/lengt
h(TestSet$Class)),digits=3)

    if(prediction_accuracy > Best_Accuracy)
    {
      Best_K <- k_value
      Best_Accuracy <- prediction_accuracy
    }
  }
  return(list("Best_K" = Best_K, "Best_Accuracy"=Best_Accuracy))
}
```

Step 5: I removed all the missing values from the data set and , split the data into training and test data sets and used my_knn_func to compute the best K and best prediction accuracy.

```
#Case 1: Remove all the missing rows and run the knn fuction to find the accu
racy.
list_of_k <- seq(1,15)
```

```
drop_missing <- na.omit(cancerdata)
GetSplitData <- split_data(drop_missing)
GetK_And_Accuracy <- my_knn_func(GetSplitData$Training,GetSplitData$Test,list
_of_k )
print("Results after dropping Missing rows:")

## [1] "Results after dropping Missing rows:"

GetK_And_Accuracy$Best_K

## [1] 5

GetK_And_Accuracy$Best_Accuracy

## [1] 0.982
```

Step 6: Then used the mean value of the “Bare_Nuclie” column and imputed this value to all the missing rows for this column and. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

```
# Impute the mean of the Bare_Nuclei for the missing values
cancerdata_with_mean <- cancerdata
cancerdata_with_mean$Bare_Nuclei[is.na(cancerdata_with_mean$Bare_Nuclei)]<-me
an(cancerdata_with_mean$Bare_Nuclei,na.rm=TRUE)
rm(GetSplitData)
GetSplitData <- split_data(cancerdata_with_mean)
GetK_And_Accuracy <- my_knn_func(GetSplitData$Training,GetSplitData$Test,list
_of_k )
print("Results after imputing mean:")

## [1] "Results after imputing mean:"

GetK_And_Accuracy$Best_K

## [1] 5

GetK_And_Accuracy$Best_Accuracy

## [1] 0.977
```

Step 7: Then used the mode value of the “Bare_Nuclie” column and imputed this value to all the missing rows for this column and. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

```
# Impute the mode of Bare Nuclie
cancerdata_with_mode <- cancerdata
cancerdata_with_mode$Bare_Nuclei[is.na(cancerdata_with_mode$Bare_Nuclei)]<-mo
```

```
de(cancerdata_with_mode$Bare_Nuclei)
rm(GetSplitData)
GetSplitData <- split_data(cancerdata_with_mode)
#GetK_And_Accuracy <- my_knn_func(GetSplitData$Training,GetSplitData$Test,lis
t_of_k )
print("Results after imputing mode:")

## [1] "Results after imputing mode:"

GetK_And_Accuracy$Best_K

## [1] 5

GetK_And_Accuracy$Best_Accuracy

## [1] 0.965
```

Step 8: I used “mice” library and used the method = “norm.predict” to perform regression to impute the missing data. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

```
#Use Regression to impute the values for the missing data
library(mice)

## Warning: package 'mice' was built under R version 4.0.3

##
## Attaching package: 'mice'

## The following objects are masked from 'package:base':
##
##     cbind, rbind

impute <- mice(cancerdata,method="norm.predict",m=1)

##
##  iter imp variable
##   1   1  Bare_Nuclei
##   2   1  Bare_Nuclei
##   3   1  Bare_Nuclei
##   4   1  Bare_Nuclei
##   5   1  Bare_Nuclei

data_impute <- complete(impute)

rm(GetSplitData)
GetSplitData <- split_data(data_impute)
GetK_And_Accuracy <- my_knn_func(GetSplitData$Training,GetSplitData$Test,list
_of_k )
print("Results after imputing Regression:")
```

```
## [1] "Results after imputing Regression:"

GetK_And_Accuracy$Best_K

## [1] 5

GetK_And_Accuracy$Best_Accuracy

## [1] 0.977
```

Step 9: I used "mice" library and used the method="norm.nob" to perform regression with perturbation to impute the missing data. Then I split the data into training and test sets and computed the best K and best prediction accuracy using the k nearest neighbor method.

```
#use Regression with perturbation to impute the values for the missing data
# and run knn

impute <- mice(cancerdata,method="norm.nob",m=1)

##
##  iter imp variable
##   1   1  Bare_Nuclei
##   2   1  Bare_Nuclei
##   3   1  Bare_Nuclei
##   4   1  Bare_Nuclei
##   5   1  Bare_Nuclei

data_impute <- complete(impute)

rm(GetSplitData)
GetSplitData <- split_data(data_impute)
GetK_And_Accuracy <- my_knn_func(GetSplitData$Training,GetSplitData$Test,list
_of_k )
print("Results after imputing Regression with perturbation:")

## [1] "Results after imputing Regression with perturbation:"

GetK_And_Accuracy$Best_K

## [1] 3

GetK_And_Accuracy$Best_Accuracy

## [1] 0.977
```

Step 10: Conclusion and Analysis

Method	Prediction Accuracy	Best k (1 - 15)	Advantage	Disadvantage	Inference
Remove missing values and run k nearest neighbor model.	0.982	5	Easiest to implement	Lost 16 data points. Also, potential for biased missing data	The accuracy shows the model is overfit with more bias.
Impute missing values with **mean** and run k nearest neighbor model.	0.977	5	Easy to compute.	Underestimate the missing value.	The accuracy shows the model is overfit with more bias.
Impute missing values with **mode** and run k nearest neighbor model.	0.96	5	Easy to compute.	Underestimate the missing value.	The accuracy shows the model is overfit with more bias.
Impute missing values **with regression** and run k nearest neighbor model.	0.977	5	Reduced Bias.	Complex to build, fit and validate. Does not capture variability.	Linear regression is not a better option to be used for imputed values. Model is overfit.
Impute missing values **with regression with perturbation** and run k nearest neighbor model.	0.977	3	Overcomes Bias problem and gives better variability	Gives estimates that are less accurate	Model is better with better k value but is still overfits data

The above results show almost identical accuracy values > 96 which suggest that the Model is overfitting the data due to the imputation of the missing values.

The removal of missing data is highly overfit when compared to other model of imputation.

The mean/mode imputation method is models is less good than the regression imputation models as there is a higher chance of underestimating the missing values and overfits the model.

The linear regression imputation method is also overfitting. It is not so reasonable to expect the linear regression to give us a better model fit, we may be better off if we use some other advanced methods.

The "Imputation using regression with Perturbation" as we see the k value is less = 3 than the rest of the models (k=5) indicating that the data is not as tightly fit as in the other models. So is better.

Question 15.1

Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

Answer:

Optimization is the method of finding the most efficient solution from a set of possible solutions.

Suppose we are estabalishing a multinational company which will have its offices in many locations across the world and we need to pick the locations which will allow the company to serve its customers across the globe while being cost effective and all the locations need to be conveniently accessible by its employees.

Here we are trying to optimize the number of office locations. i.e minimize the number of offices also by meeting the required goals.

The Data that we need are customer bases across the world, airport location, real estate data, weather data, political conditions, location distance from head quarters and other potential offices.

The constrains could be locations should have international airport access and the real estate market prices should be reasonable. The weather and political conditions should be conducive for the business. No two locations should be very close by.