CS5331: ASSIGNMENT 2

Replacement of missing data with substitution is called data imputation. In R, there are lots of packages that provides us with the functionality of data imputation. 4 of those packages are:

mice() : Multivariate Imputation via Chained Equations (MICE) is one of the most commonly used packages in R, for data imputations. The assumption made by MICE is that missing data are Missing at Random (MAR). This means that the probability that a value is missing depends on observed value and can be predicted using them.

Mean Imputation : Mean imputation is the replacement of a missing observation with the mean of the non-missing observations for that variable. In R, this can be done by using the guess():

$imputedData < -guess(nonimputedData, type = "mean")$

missForest() : This is a non-parametric imputation method applicable to various variable types. A non-parametric method is such that does not make explicit assumptions about functional form of any arbitrary function, $'f'$. It tries to estimate $'f'$ such that it can be as close to the data points without seeming impractical. It builds a random forest model for each variable. Then it uses the model to predict missing values in the variable with the help of observed values.

k-NN : knnImputation function fills in all missing values (NA values) using the k Nearest Neighbour of each case. By default, it uses the values of the neighbors and obtains a weighted average of their values to fill in the unknowns. The definition of the function is:

$knnImputation(data, k = 10, scale = T.meth = weighAvg, distData = NULL)$

where,
data: A data frame with the data set.
k: The number of nearest neighbors to use (default value is 10).
scale: Boolean setting if the data should be scaled before finding the nearest neighbours (default value = T)
meth: String indicating the method used to calculate the value to full in each NA. Available values are median or weighAvg (default).
distData: This is optional; one can provide a data frame containing data set that should be used to find the neighbours.

OBSERVATIONS -

The function Rmse from the library imputeR was used to calculate the Root mean square error between imputed and true values. A demo for the function is as follows:

$Rmse(imp, mis, true, norm = FALSE)$

Where,

imp: the imputed data matrix. In this case, imp contains the dataset after data imputation is done.
mis: the missing data matrix. In this case, mis is the dataset containing randomly generated NA values.
true: the true data matrix with original values. In this case, true is the original IRIS dataset.

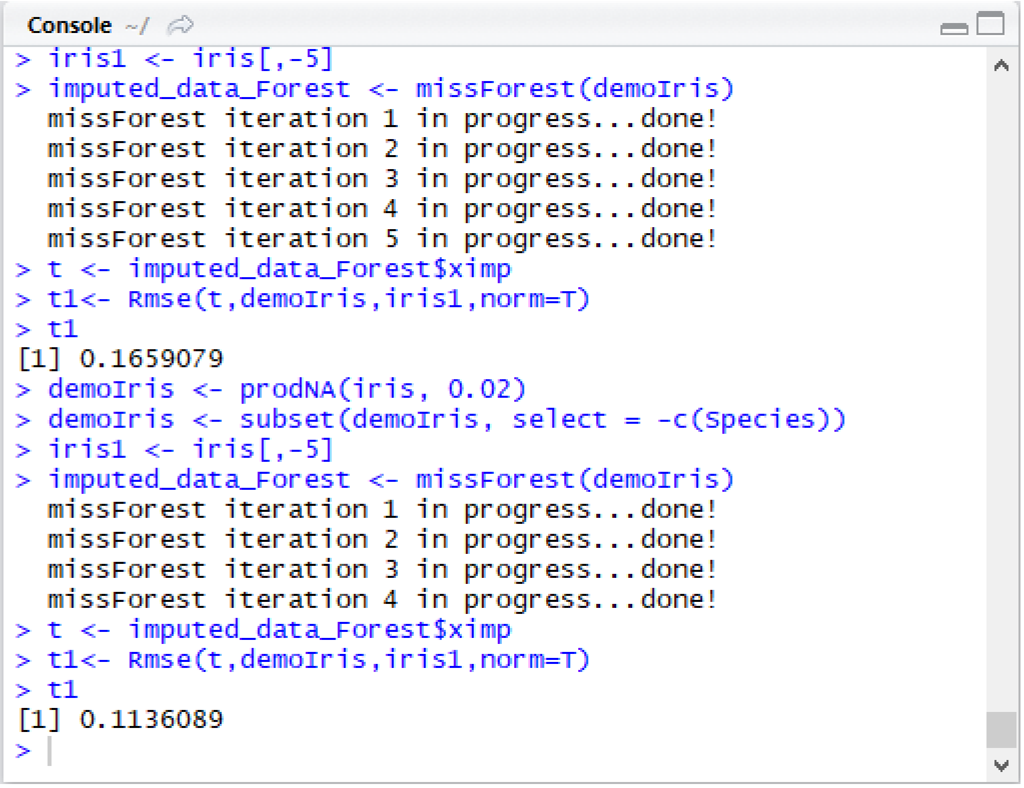
Missing values were randomly created to occupy x% of the data of the iris dataset. The values of x being 2, 5, 10, 15, 20, 25. Then three different data imputation techniques were used: missForest, mean imputation and k-NN imputation. The RMSE between the imputed values and the true values were then calculated. The code snippets and observations recorded are as follows:

missForest:

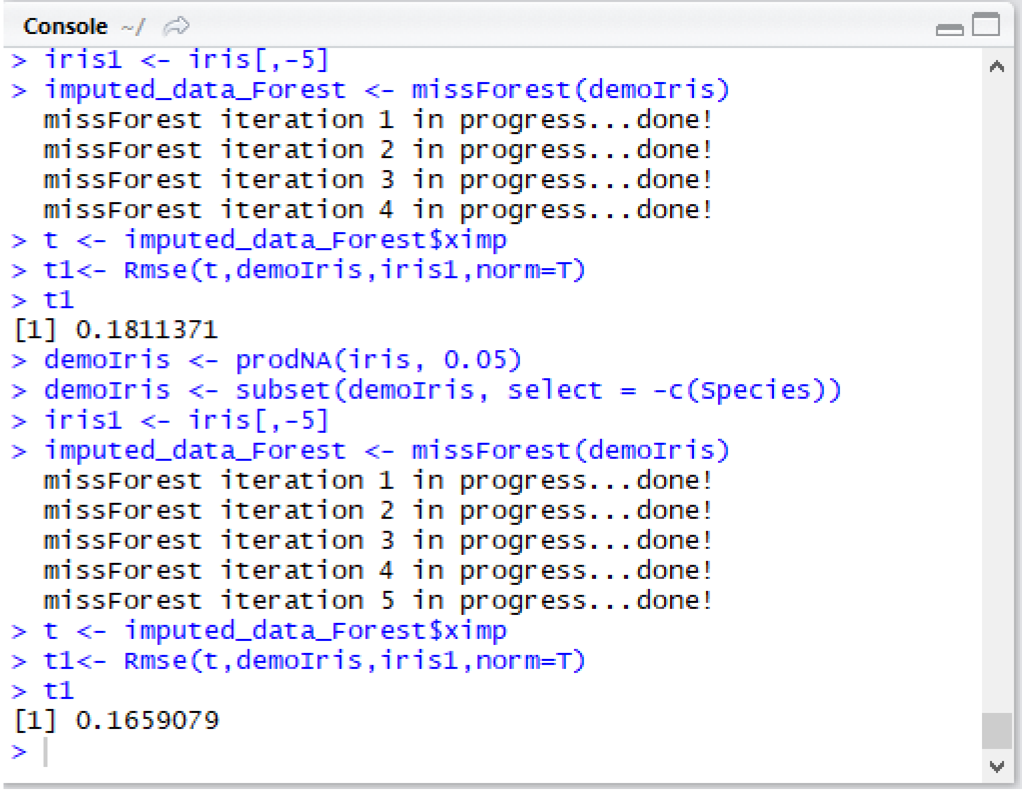
Code:

```
#missForest
imputed_data_Forest <- missForest(demoIris)
t <- imputed_data_Forest$ximp

#RMSE
t1<- Rmse(t, demoIris, iris1, norm=T)
t1
```

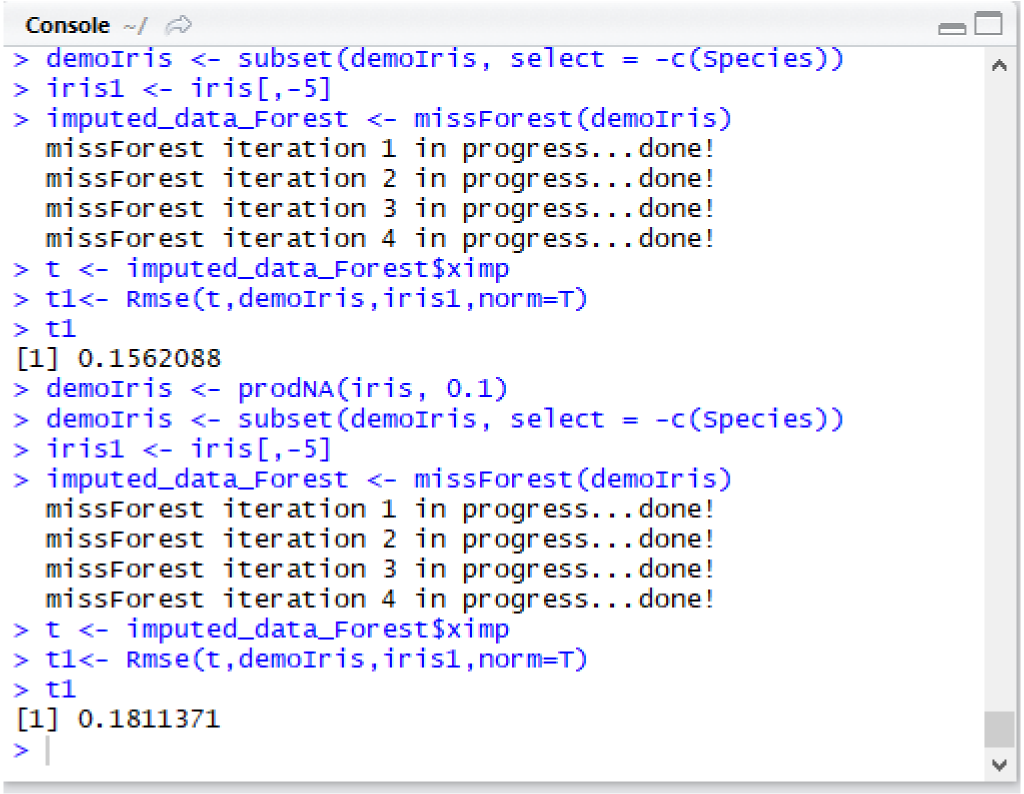
Screenshots:



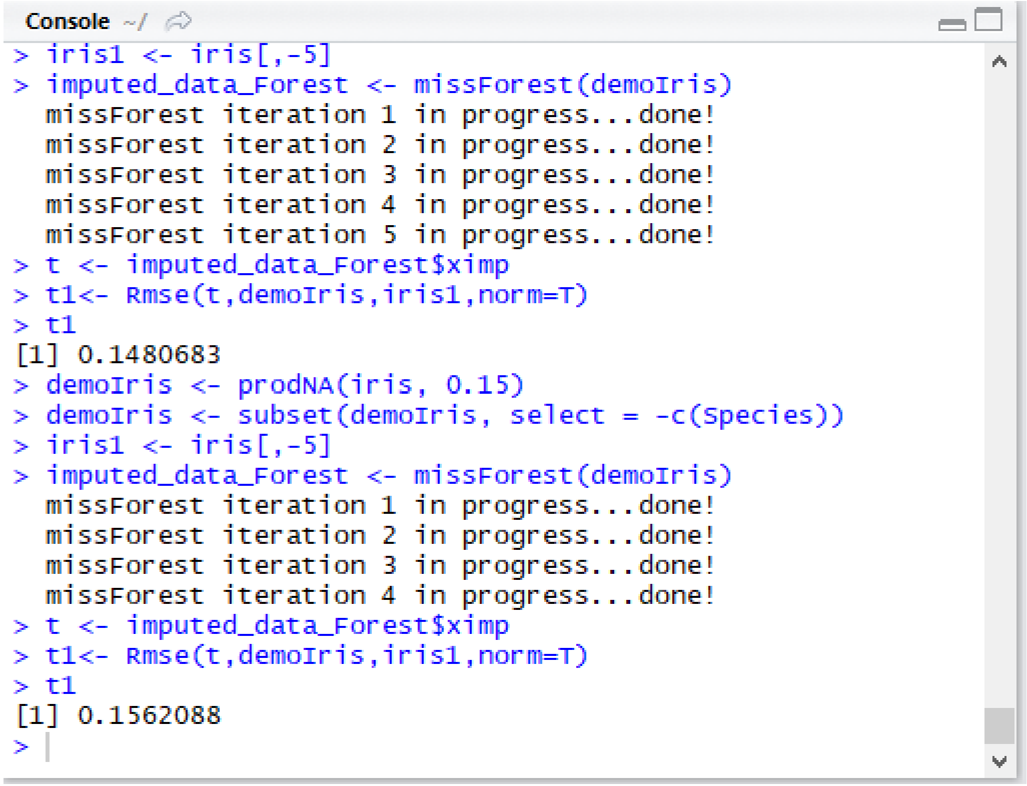
```
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1659079
> demoIris <- prodNA(iris, 0.02)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1136089
> 
```



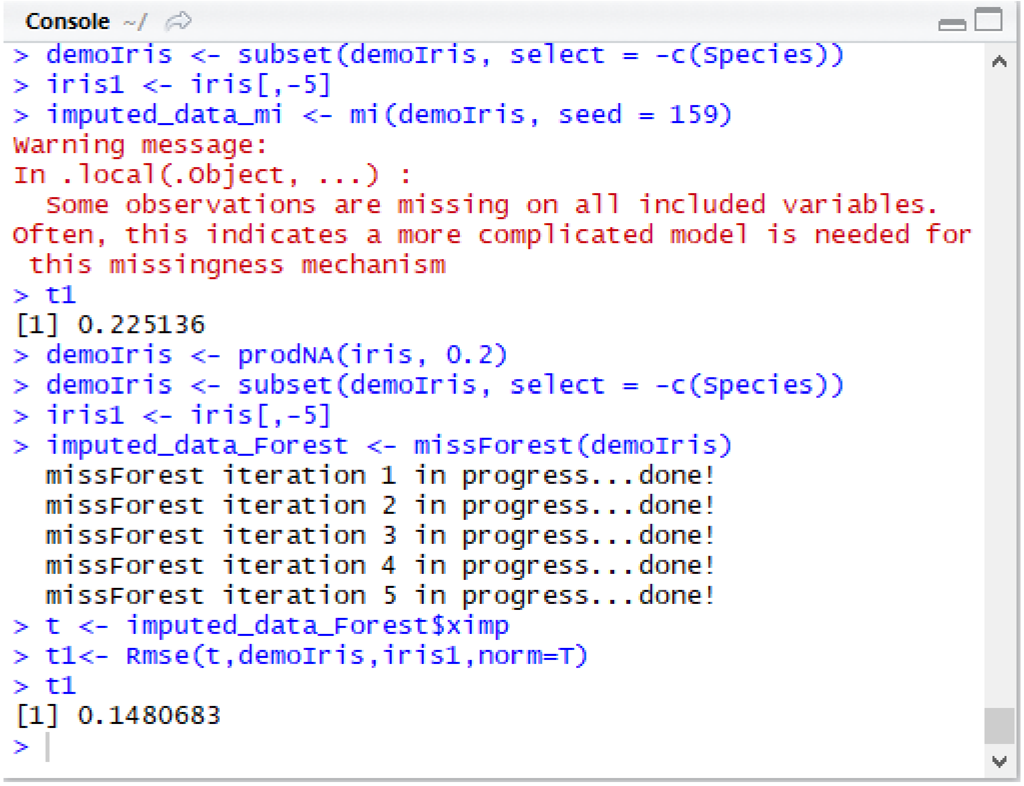
```
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1811371
> demoIris <- prodNA(iris, 0.05)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1659079
> 
```



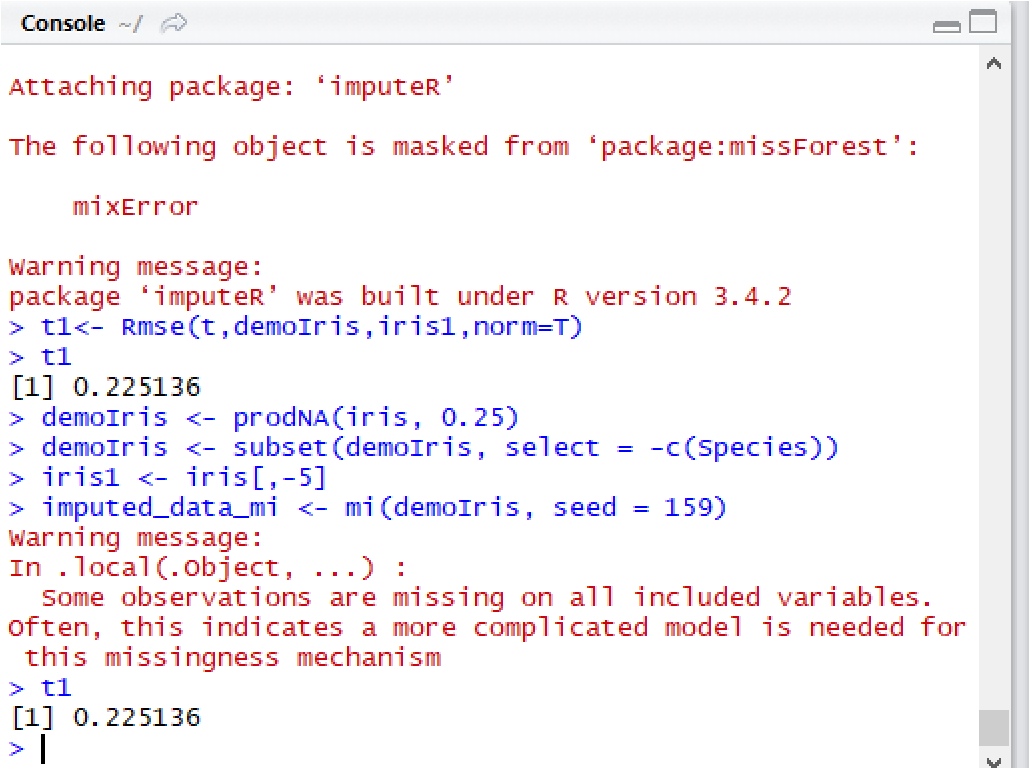
```
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1562088
> demoIris <- prodNA(iris, 0.1)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1811371
> 
```



```
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1480683
> demoIris <- prodNA(iris, 0.15)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1562088
> 
```



```
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mi <- mi(demoIris, seed = 159)
Warning message:
In .local(.Object, ...) :
  Some observations are missing on all included variables.
Often, this indicates a more complicated model is needed for
 this missingness mechanism
> t1
[1] 0.225136
> demoIris <- prodNA(iris, 0.2)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.1480683
> 
```



```
Attaching package: ‘imputeR’

The following object is masked from ‘package:missForest’:

    mixError

Warning message:
package ‘imputeR’ was built under R version 3.4.2
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 0.225136
> demoIris <- prodNA(iris, 0.25)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mi <- mi(demoIris, seed = 159)
Warning message:
In .local(.Object, ...) :
  Some observations are missing on all included variables.
Often, this indicates a more complicated model is needed for
 this missingness mechanism
> t1
[1] 0.225136
> 
```

Observation:

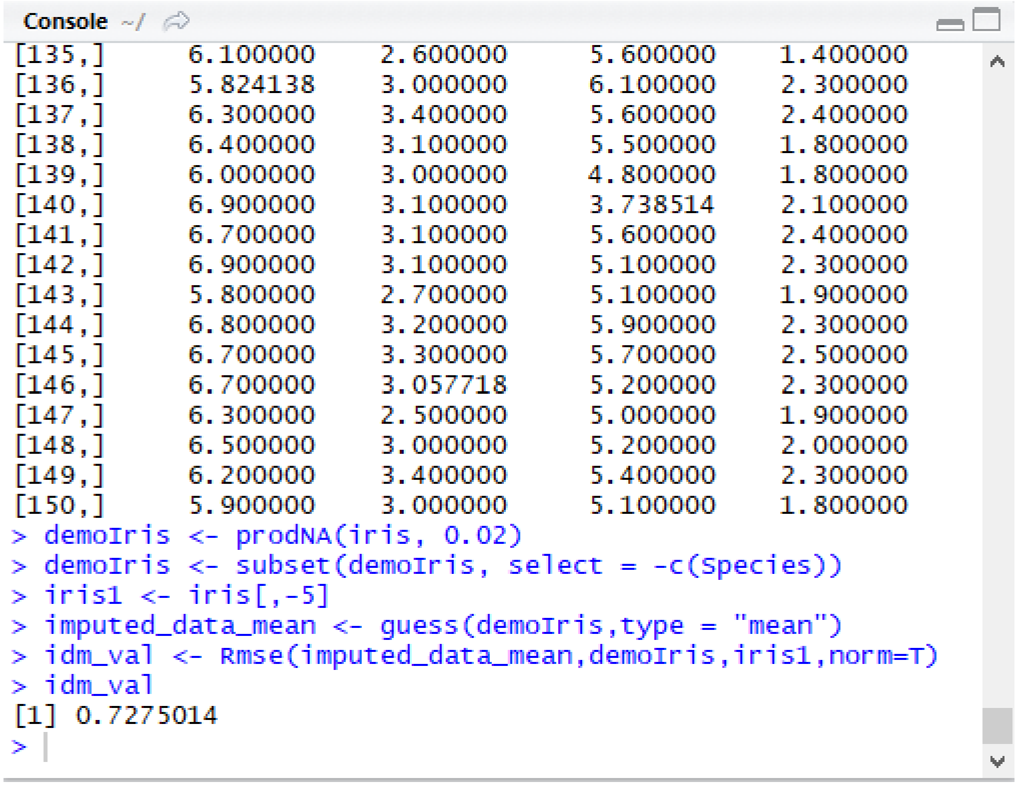
- X = 2%: 0.1136089
- X = 5%: 0.1659079
- X = 10%: 0.1811371
- X = 15%: 0.1562088
- X = 20%: 0.1480683
- X = 25%: 0.225136

Mean Imputation:

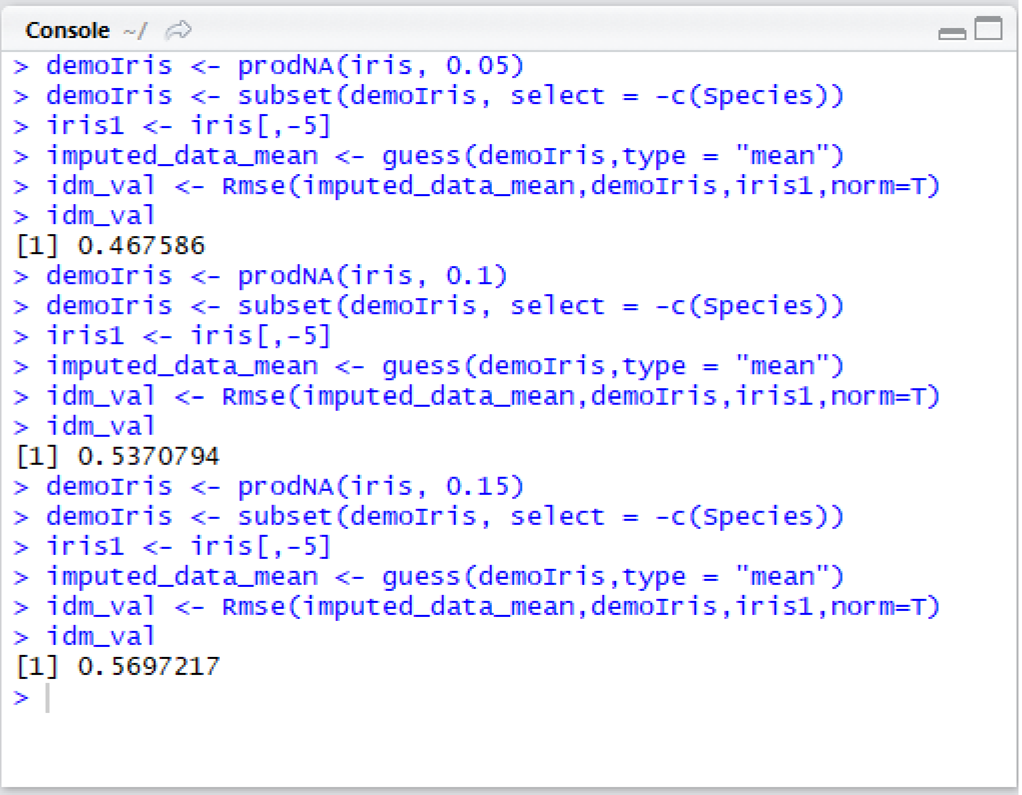
Code:

```
#Mean imputation
imputed_data_mean <- guess(demoIris,type = "mean")

#RMSE
idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
idm_val
```

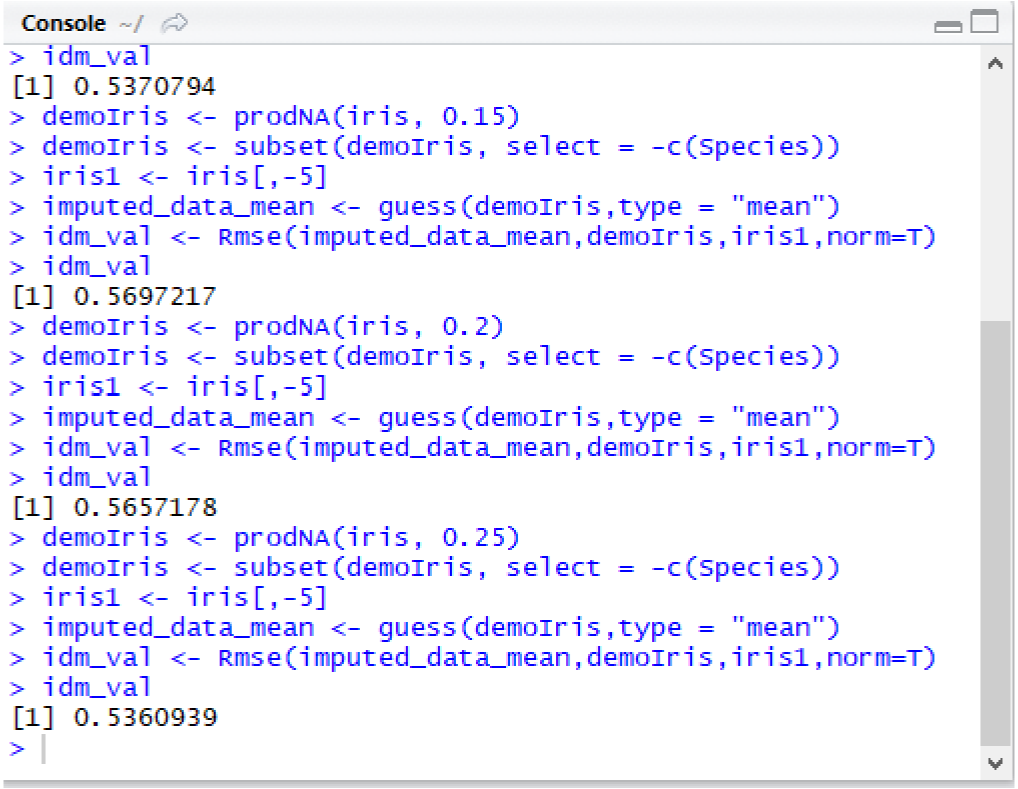
Screenshots:



```
[135,]      6.100000    2.600000     5.600000     1.400000
[136,]      5.824138    3.000000     6.100000     2.300000
[137,]      6.300000    3.400000     5.600000     2.400000
[138,]      6.400000    3.100000     5.500000     1.800000
[139,]      6.000000    3.000000     4.800000     1.800000
[140,]      6.900000    3.100000     3.738514     2.100000
[141,]      6.700000    3.100000     5.600000     2.400000
[142,]      6.900000    3.100000     5.100000     2.300000
[143,]      5.800000    2.700000     5.100000     1.900000
[144,]      6.800000    3.200000     5.900000     2.300000
[145,]      6.700000    3.300000     5.700000     2.500000
[146,]      6.700000    3.057718     5.200000     2.300000
[147,]      6.300000    2.500000     5.000000     1.900000
[148,]      6.500000    3.000000     5.200000     2.000000
[149,]      6.200000    3.400000     5.400000     2.300000
[150,]      5.900000    3.000000     5.100000     1.800000
> demoIris <- prodNA(iris, 0.02)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.7275014
> 
```



```
> demoIris <- prodNA(iris, 0.05)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.467586
> demoIris <- prodNA(iris, 0.1)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.5370794
> demoIris <- prodNA(iris, 0.15)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.5697217
> 
```



```
> idm_val
[1] 0.5370794
> demoIris <- prodNA(iris, 0.15)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.5697217
> demoIris <- prodNA(iris, 0.2)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.5657178
> demoIris <- prodNA(iris, 0.25)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 0.5360939
> 
```

Observation:

- X = 2%: 0.7275014
- X = 5%: 0.467586
- X = 10%: 0.5370794
- X = 15%: 0.5697217
- X = 20%: 0.5657178
- X = 25%: 0.5360939

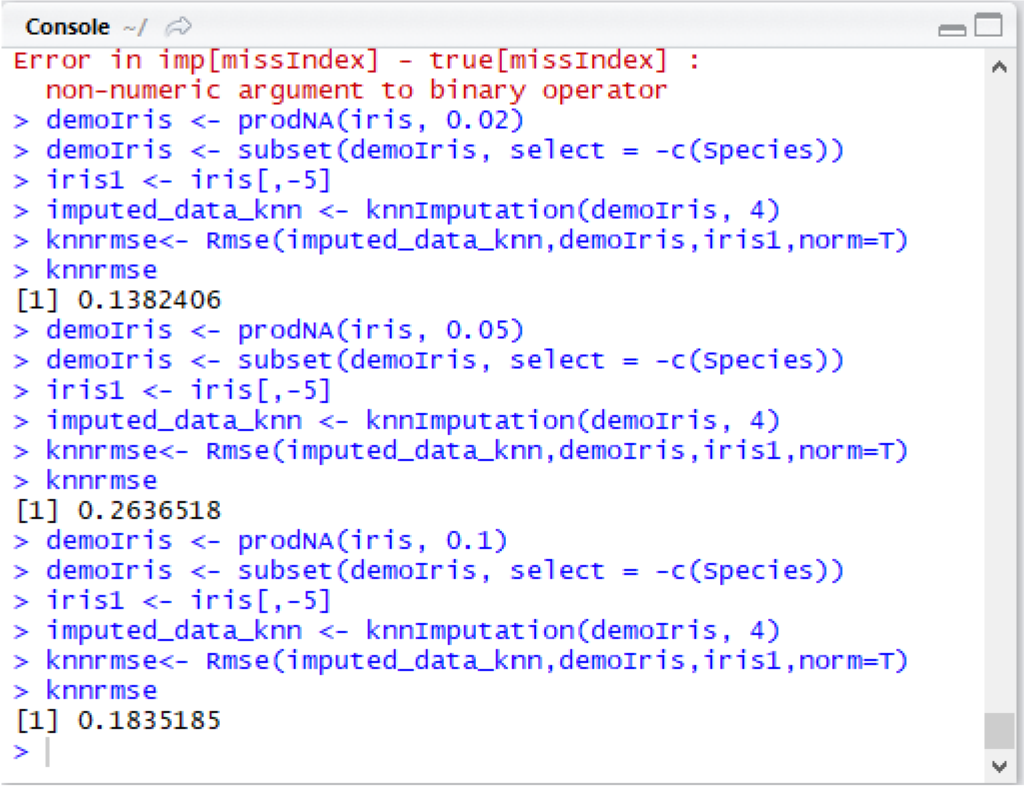
k-NN:

Code:

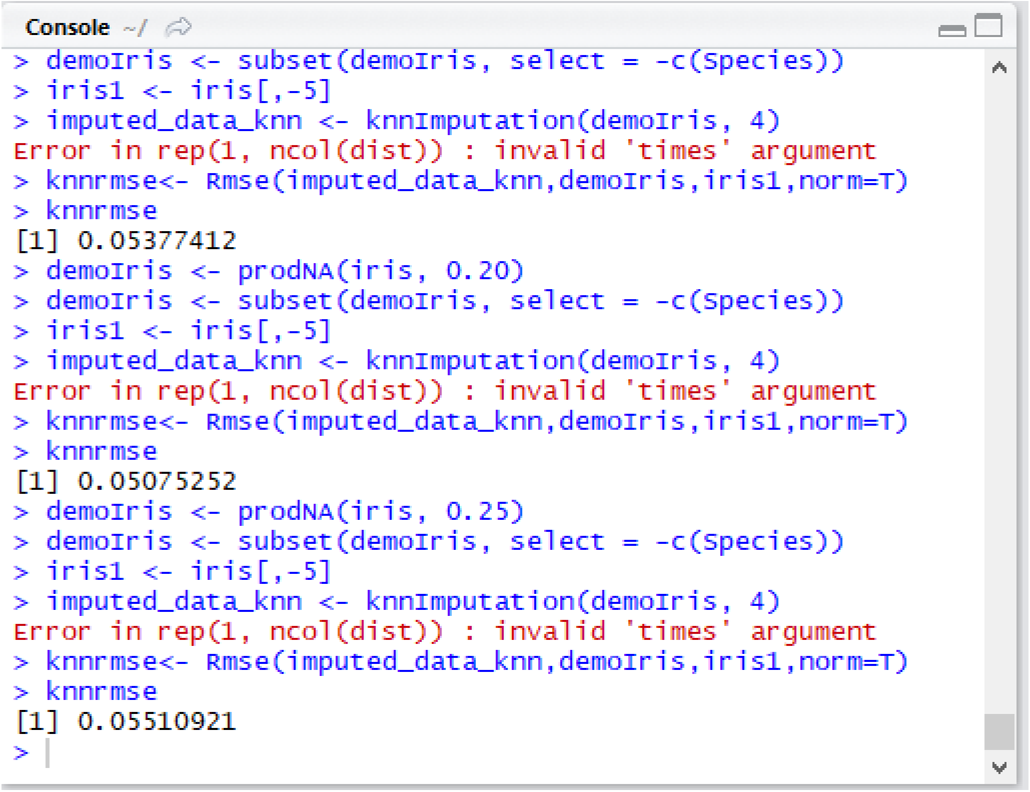
```
library(DMwR)
imputed_data_knn <- knnImputation(demoIris, 4)

knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
knnrmse
```

imputed_data_knn

Screenshots:



```
Error in imp[missIndex] - true[missIndex] :
  non-numeric argument to binary operator
> demoIris <- prodNA(iris, 0.02)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.1382406
> demoIris <- prodNA(iris, 0.05)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.2636518
> demoIris <- prodNA(iris, 0.1)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.1835185
> 
```



```
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
Error in rep(1, ncol(dist)) : invalid 'times' argument
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.05377412
> demoIris <- prodNA(iris, 0.20)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
Error in rep(1, ncol(dist)) : invalid 'times' argument
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.05075252
> demoIris <- prodNA(iris, 0.25)
> demoIris <- subset(demoIris, select = -c(Species))
> iris1 <- iris[,-5]
> imputed_data_knn <- knnImputation(demoIris, 4)
Error in rep(1, ncol(dist)) : invalid 'times' argument
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 0.05510921
> 
```

Observation:

- X = 2%: 0.1382406
- X = 5%: 0.2636518
- X = 10%: 0.1835185
- X = 15%: 0.05377412
- X = 20%: 0.05075252
- X = 25%: 0.05510921

From the above observations, we can see that k-NN imputation method is the most effective method amongst the three methods implemented, when the value of x is high, whereas missForest is more effective when the value of X is lower. We also see that mean imputation is the least accurate data imputation technique. We can also observe that when the amount of missing data is less, the imputations are somewhat less accurate than when the amount of missing data is more.

We used the function kNN() to find out the supervised classification error. We used the complete iris dataset as the training set and imputed dataset as the test set and recorded the observations. The supervised classification error is computed by the percentage of misclassified instances after applying kNN(). The observations are as follows:

missForest:

- 2% missing values : 2%
- 5% missing values : 2.67%
- 10% missing values : 1.33%
- 15% missing values : 0%
- 20% missing values : 0.67%
- 25% missing values : 1.33%

Mean:

- 2% missing values : 0%
- 5% missing values : 2.66%
- 10% missing values : 2.66%
- 15% missing values : 4.67%
- 20% missing values : 6.67%
- 25% missing values : 14.67%

kNN:

- 2% missing values : 0%
- 5% missing values : 0%
- 10% missing values : 0%
- 15% missing values : 2.67%
- 20% missing values : 2%
- 25% missing values : 2.67%

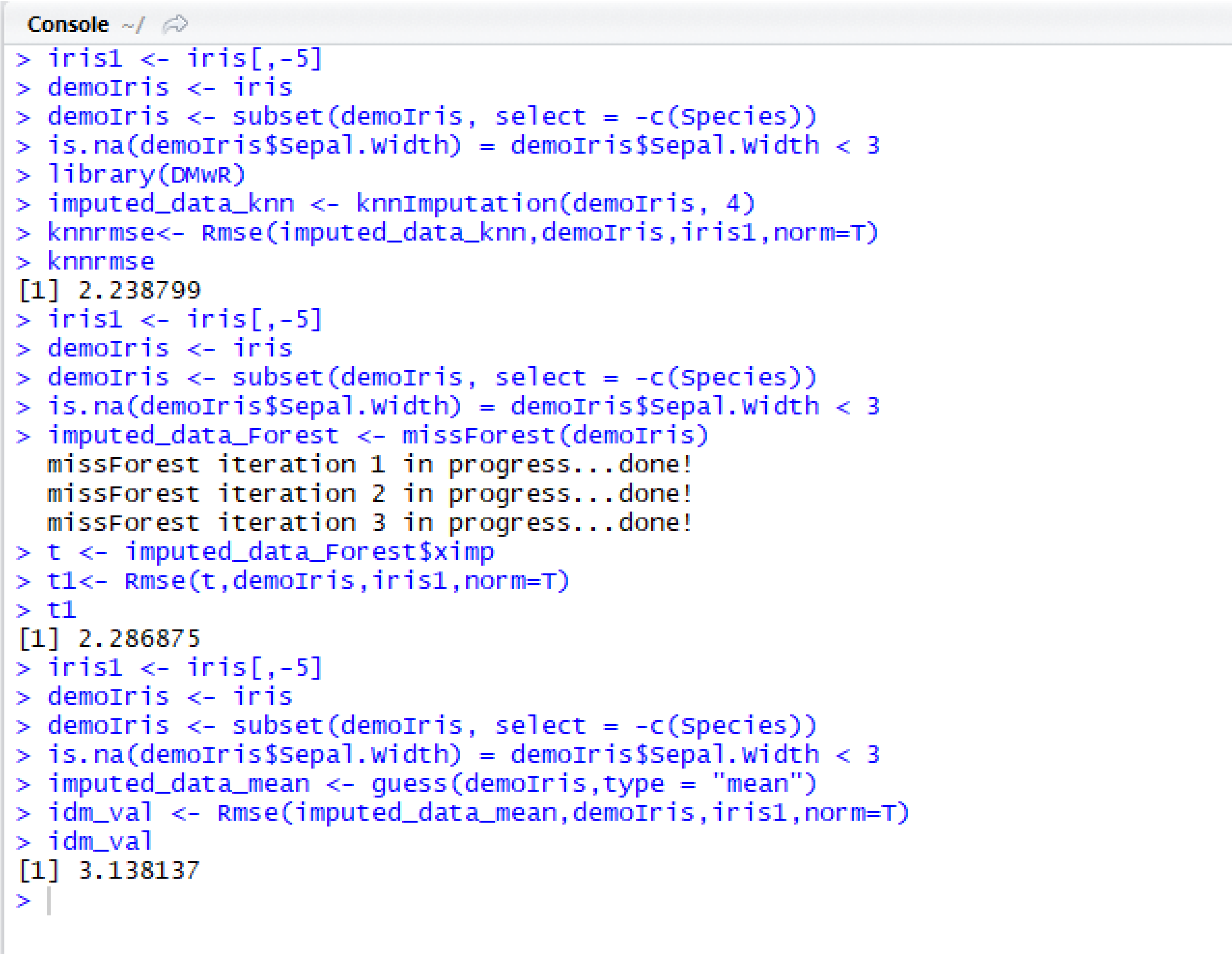
The screenshots for the observations are available in Github.

From the observations above, we see that kNN imputation is more effective than the other two methods and mean imputation is the least effective.

Now, the column of $Sepal_Width$ was taken into consideration, and missing values were assigned to those values which were less than 3. Then all the data imputation methods were repeated, and observation was recorded. The code snippet of assignment of missing values and the calculated RMSE values for each function are: **Code:**

```
#Changing one columns as NA

iris1 <- iris[,-5]
demoIris <- iris
demoIris <- subset(demoIris, select = -c(Species))
is.na(demoIris$Sepal.Width) = demoIris$Sepal.Width < 3
```

Screenshot



```
> iris1 <- iris[,-5]
> demoIris <- iris
> demoIris <- subset(demoIris, select = -c(Species))
> is.na(demoIris$Sepal.Width) = demoIris$Sepal.Width < 3
> library(DMwR)
> imputed_data_knn <- knnImputation(demoIris, 4)
> knnrmse<- Rmse(imputed_data_knn,demoIris,iris1,norm=T)
> knnrmse
[1] 2.238799
> iris1 <- iris[,-5]
> demoIris <- iris
> demoIris <- subset(demoIris, select = -c(Species))
> is.na(demoIris$Sepal.Width) = demoIris$Sepal.Width < 3
> imputed_data_Forest <- missForest(demoIris)
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
> t <- imputed_data_Forest$ximp
> t1<- Rmse(t,demoIris,iris1,norm=T)
> t1
[1] 2.286875
> iris1 <- iris[,-5]
> demoIris <- iris
> demoIris <- subset(demoIris, select = -c(Species))
> is.na(demoIris$Sepal.Width) = demoIris$Sepal.Width < 3
> imputed_data_mean <- guess(demoIris,type = "mean")
> idm_val <- Rmse(imputed_data_mean,demoIris,iris1,norm=T)
> idm_val
[1] 3.138137
> 
```

Observation:

- missForest: 2.286875
- Mean Imputation: 3.138137
- k-NN: 2.238799

We can see from the above observations that, missForest() performs a better than k-NN in this case, whereas mean imputation's performance is lower than the other two.

REFERENCE

`https://github.com/sm2k2010/Data_Imputation`