Project Report

Table Of Contents

Introduction

1. Materials And Concepts
2. DataSets
3. Regression
4. Hypothesis testing

Data Processing

1. Artwork Value Dataset
2. Bus Station Dataset

Implementation Of Regression Model

1. Linear Regression
2. KNN Regression

Experimental Analysis And Results

1. Evaluation metrics of Bus Station DataSet
2. Evaluation metrics of Artwork Value DataSet

Conclusion

References

Introduction

Data science combines math and statistics, specialized programming, sophisticated analytics, artificial intelligence (AI), machine learning, and unique subject matter expertise with domain expertise to reveal useful insights hidden in the data of an organization. Decision-making and future planning can be influenced by these findings.

Analysts are able to derive useful insights because the data science lifecycle encompasses a variety of roles, technologies, and processes. A data science project often goes through the following stages:

- Data processing
- Machine Learning model implementation
- Experimental results and analysis
- Conclusion

Material And Concepts

Datasets:

Two datasets will be provided to you to analyze as part of this project:

1. Bus Station
2. Artwork Value

Bus Station:

The bus station dataset consists of hourly measurements of the number of commuters using different busses.

- Date = Date of current row. (Format = DD/MM/YYYY.)
- Hour = Hour of the day for the current row. (Unit = Character value.)
- Temperature = The temperature at each hour. (Unit = Deg. C.)
- Rainfall = The rainfall at each hour.(Unit = mm.)
- BusA2 = The number of commuters using train bus A2 at a given time.(Unit = integer value.)
- BusA3 = The number of commuters using train bus A3 at a given time. (Unit = Character value.)
- BusA1 = The number of commuters using train bus A1 at a given time.(Unit = integer value.)

```
> #observing the dataset
> head(tuna2)
  BusA1       Date Hour Temperature Rainfall BusA2 BusA3
1     9 01/01/2019    0    2.469638        0    11     0
2     5 01/01/2019    1    1.635323        0     5    46
3     5 01/01/2019    2    2.470651        0     5    15
4     2 01/01/2019    3    2.156461        0     3    56
5     0 01/01/2019    4    2.798946        0     0     0
6     0 01/01/2019    5    2.875518        0     0     0
```

Artwork Value:

The artwork dataset consists of artwork with an age in years (‘Age’), and an integer value of 0 or 1 indicating if the art was verified as authentic by an independent evaluator (‘Verified’).

- Art.price = The price of Artworks. (Unit = Integer value)
- Age = The age of the Artworks.(Unit = Character value)
- Verified = The Artwork's Verification status.(Unit = Character value

```
> #observing the dataset
> head(tuna1)
  Art.Price Age Verified
1 627505.46 487        1
2 205602.64 177        1
3 173949.20 168        1
4 236897.26 273        1
5 196058.40 179        1
6  89589.43 159        0
```

Regression

Regression is a method for examining the relationship between independent variables or features and a dependent variable or result. It serves as a technique for predictive modeling in machine learning, which employs an algorithm to predict continuous outcomes.We can majorly divide regression techniques into several parts but two most important are linear and polynomial regression.

Linear regression

When utilizing Simple Regression, two variables are connected by a straight line to establish their relationship. By identifying the slope and intercept that define the line and reduce regression errors, it aims to create a line that closely approximates the data. Only one x and one y variable are present in linear regression.

Polynomial Regression

The link between the independent variable x and the dependent variable y is described as an nth degree polynomial in polynomial regression, one of the kinds of linear regression. A nonlinear relationship between the value of x and the corresponding conditional mean of y, denoted E (y |x), can be fit by polynomial regression.

In this project we have used linear regression and knn regression as our regression models. One independent variable and one dependent variable are both estimated using a straight line in a regression model called linear regression while,through the use of a local average, KNN regression tries to estimate the value of the output variable.Now lets us understand each one of them in depth.

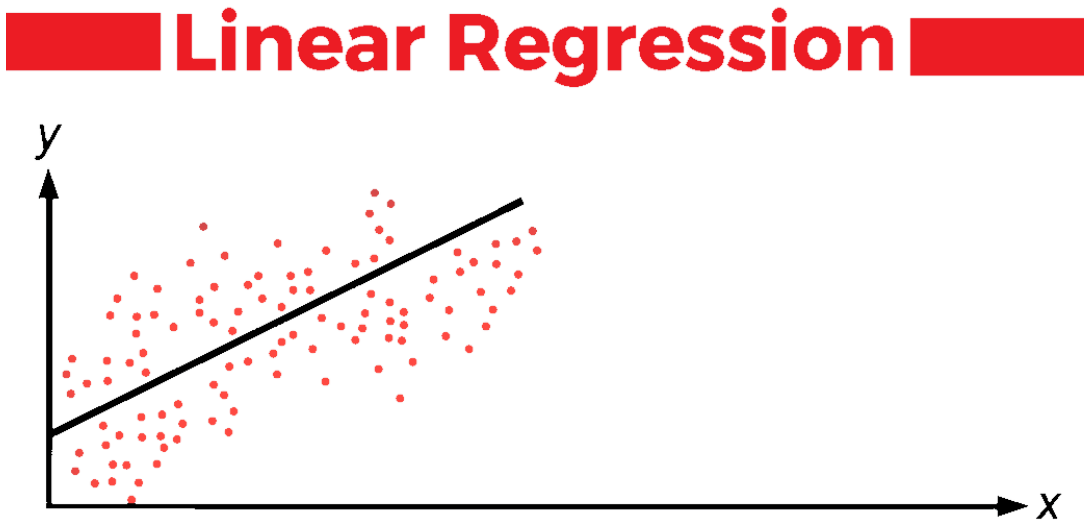
Linear Regression

A machine learning algorithm built on supervised learning is linear regression. It completes a regression task. Regression creates a goal prediction value based on independent variables. It is mostly used for determining how variables and predictions relate to one another.The linear regression model tries to find out the best fit line to determine the relationship between input variable (x) and output variable(y).The linear regression model uses the following function to find the relationship, as shown below.

$$y = \theta_1 + \theta_2.x$$

X: training data
Y: labeled data
θ_1:Slope
θ_2 :Coefficient of x

Our aim is to find the value of θ_1 and θ_2 in order to get the best fit line.



KNN Regression

The supervised learning technique K-nearest neighbors (KNN) is used for both regression and classification. By calculating the Euclidean distance between the test data and all of the training points, KNN tries to predict the proper class for the test data. Then choose the K spots

that are closest to the test data. The KNN algorithm estimates the likelihood that the test data belong to each of the "K" training data classes, and the class with the highest likelihood is chosen. The value in a regression situation is the average of the 'K' chosen training points.

The Knn Algorithm works in following ways:

- Choose the K neighbor.
- Determine the Euclidean distance between K neighbors.

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

- Pick the K closest neighbors based on the Euclidean distance estimate.
- Count how many data points there are in each category among these k neighbors.
- The category for which the number of neighbors is highest should receive the new data points.

Hypothesis Testing

A statistical technique called hypothesis testing is employed when generating statistical judgments based on experimental data. In essence, hypothesis testing involves making an assumption about the population parameter.In Statistics we define two parameters Null hypothesis (H_0) and Alternate hypothesis (H_a) for hypothesis testing.We have also established a Null Hypothesis for both the datasets in order to perform the hypothesis testing is as followed:-

Hypothesis Testing of Artwork Value dataset

Linear regression

The Null hypothesis defined in the case of linear regression is :

Null Hypothesis: True difference in means of actual and predicted values will be zero.

```
> #NULL hypothesis: True difference in means of actual and predicted values will be zero
> t.test(pred_kt, tuna1_new1$Art.Price,var.equal = TRUE,paired = FALSE)  #null hypothesis true

        Two Sample t-test

data:  pred_kt and tuna1_new1$Art.Price
t = 1.7912e-14, df = 188, p-value = 1
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -38461.88  38461.88
sample estimates:
mean of x mean of y
 270543.6  270543.6
```

As we can see from the output, our Null Hypothesis holds true .

KNN Regression

The Null Hypothesis defined in the case of KNN Regression model is:

Null Hypothesis: True difference in means of actual and predicted values will be zero.

```
> #NULL hypothesis: True difference in means of actual and predicted values will be zero
> t.test(pred1, tuna1_new1$Art.Price,var.equal = TRUE,paired = FALSE)  #null hypothesis False

        Two Sample t-test

data:  pred1 and tuna1_new1$Art.Price
t = 0.034635, df = 188, p-value = 0.9724
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -35774.57  37053.24
sample estimates:
mean of x mean of y
 271183.0  270543.6
```

As we can see from the output, our Null Hypothesis does not hold true . So, we have to opt for the Alternative hypothesis i.e,

Alternate Hypothesis: True difference in mean is not equal to zero.

Hypothesis Testing of Bus Station dataset

Linear regression

The Null hypothesis defined in the case of linear regression is :

Null Hypothesis: True difference in means of actual and predicted values will be zero.

```
> #NULL hypothesis: True difference in means of actual and predicted values will be zero
> t.test(pred_k_lm, tuna2$BusA1,var.equal = TRUE,paired = FALSE)  #null hypothesis true

        Two Sample t-test

data:  pred_k_lm and tuna2$BusA1
t = 0.088817, df = 4312, p-value = 0.9292
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.5739392  0.6284089
sample estimates:
mean of x mean of y
 19.47122  19.44398
```

As we can see from the output, our Null Hypothesis does not hold true . So, we have to opt for the Alternative hypothesis i.e,

Alternate Hypothesis: True difference in mean is not equal to zero.

KNN Regression

The Null Hypothesis defined in the case of KNN Regression model is:

Null Hypothesis: True difference in means of actual and predicted values will be zero.

```
> #NULL hypothesis: True difference in means of actual and predicted values will be zero
> t.test(pred2_k, tuna2$BusA1,var.equal = TRUE,paired = FALSE)  #null hypothesis false

        Two Sample t-test

data:  pred2_k and tuna2$BusA1
t = -0.053905, df = 4312, p-value = 0.957
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.7345999  0.6952850
sample estimates:
mean of x mean of y
 19.42432  19.44398
```

As we can see from the output, our Null Hypothesis does not hold true . So, we have to opt for the Alternative hypothesis i.e,

Alternate Hypothesis: True difference in mean is not equal to zero.

Data Processing

We processed both the Artwork value dataset and Bus Station Dataset separately , let's go through each dataset one by one.

Artwork Value Dataset

Step1:- Observing the Dataset to get the basic information about the dataset like, dimension of dataset,analyzing the head of dataset and then getting a summary and glimpse of the dataset.

```
> #reading the csv data file of artwork
> tuna1 <- read.csv("CT5163_RepeatProject_2021_22_Artwork_final.csv",TRUE ,",")
> #creatig the DataFrame of the csv file
> class(tuna1)
[1] "data.frame"
> #observing the dataset
> head(tuna1)
  Art.Price Age Verified
1 627505.46 487        1
2 205602.64 177        1
3 173949.20 168        1
4 236897.26 273        1
5 196058.40 179        1
6  89589.43 159        0
> plot(tuna1)
> # dimensions of dataset
> dim(tuna1)
[1] 99  3
> # list types for each attribute
> sapply(tuna1, class)
  Art.Price         Age    Verified
  "numeric" "character" "character"
> #Statistical summary and glimpse
> summary(tuna1)
   Art.Price           Age             Verified
 Min.   :-100780   Length:99          Length:99
 1st Qu.: 171822   Class :character   Class :character
 Median : 244991   Mode  :character   Mode  :character
 Mean   : 266027
 3rd Qu.: 354978
 Max.   : 627506
> glimpse(tuna1)
Rows: 99
Columns: 3
$ Art.Price <dbl> 627505.46, 205602.64, 173949.20, 236897.26, 196058.40, 89589.43, 407522.62, 199544.31, 358946.67, 127678.01, 978…
$ Age       <chr> "487", "177", "168", "273", "179", "159", "480", "192", "321", "261", "197", "159", "182", "479", "404", "446", …
$ Verified  <chr> "1", "1", "1", "1", "1", "0", "0", "1", "1", "0", "0", "0", "1", "1", "0", "0", "0", "0", "1", "1", "1", "0", "1…
> |
```

Step2:- Checking for Duplicate Values

```
> #checking for duplicate values
> dup_records <- duplicated(tuna1$verified)
> sum(dup_records)
[1] 0
> summary(tuna1$Verified)
   0    1  100 zero
  44   53    1    1
> summary(tuna1$Age)
106 110 118 121 123 131 136 138 141 143 147 153 154 159 162 164 168 172 176 177 179 182 185 191 192 193 197 201 205 207 222 227 241
  1   1   1   1   1   1   1   1   1   1   2   1   1   2   1   1   2   1   2   1   1   1   1   2   1   1   1   1   1   1   1   1   1
243 261 264 268 270 273 275 282 289 296 298 318 321 323 332 334 344 352 354 359 365 373 384 388 392 393 398 403 404 405 408 409 418
  1   1   1   2   1   1   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1
422 427 433 434 440 442 446 458 462 472 477 478 479 480 483 487 491 492 498 500 N/A
  2   1   1   1   1   1   2   1   1   1   1   1   1   3   1   1   1   1   1   1   1
> |
```

There were no duplicate values in the dataset

Step3:-Checking For NA Values

```
36
37  #checking for NA values
38  is.na(tuna1)
39  #removing row with na values
40  tuna1_new <- tuna1[-c(32),]
41  summary(tuna1_new$Age)
42  tuna1_new
43
```

There was a null value found at location 32 which was successfully removed.

```
48
49  #dropping NA value
50  tuna1_new1 <- tuna1[-c(38,32),]
51  #checking the incorrect values in verified
52  subset(tuna1_new,!(Verified %in% c(1,0)))
53
```

Step4:Replacing the Negative Values From Art.Price from NA and then removing all the incorrect values and storing the correct values at a new place.

```
48  #replacing negative value of Art.price with NA
49  tuna1_new$Art.Price[tuna1_new$Art.Price < 0] <- NA
50  #finding  NA value
51  is.na(tuna1_new$Art.Price)
52
53
54
55  #removing incorrect values
56  tuna1_new1 <- tuna1[-c(59,87,32,38),]
57  tuna1_new1
58  summary(tuna1_new1)
59  glimpse(tuna1_new1)
60  plot(tuna1_new1)
61  levels(tuna1_new1)
62
```

Now our data is Ready to feed into our mAchine learning model for training as we have pre-processed and cleaned the data to make it more suitable for training.

Bus Station Dataset

Step1:- Observing the Dataset to get the basic information about the dataset like, dimension of dataset,analyzing the head of dataset and then getting a summary and glimpse of the dataset.

```
> #reading the csv data file of bus station
> tuna2 <- read.csv("CT5163_RepeatProject_2021_22_BusStation_final.csv",TRUE,",")
> #creating the DataFrame of csv file
> class(tuna2)
[1] "data.frame"
> #observing the dataset
> head(tuna2)
  BusA1       Date Hour Temperature Rainfall BusA2 BusA3
1     9 01/01/2019    0    2.469638        0    11     0
2     5 01/01/2019    1    1.635323        0     5    46
3     5 01/01/2019    2    2.470651        0     5    15
4     2 01/01/2019    3    2.156461        0     3    56
5     0 01/01/2019    4    2.798946        0     0     0
6     0 01/01/2019    5    2.875518        0     0     0
> # dimensions of dataset
> dim(tuna2)
[1] 2160    7
> # list types for each attribute
> sapply(tuna2, class)
      BusA1        Date        Hour Temperature    Rainfall       BusA2       BusA3
  "integer" "character" "character"   "numeric"   "numeric"   "integer" "character"
> #statistical summary and glimpse
> summary(tuna2)
     BusA1           Date              Hour           Temperature        Rainfall         BusA2             BusA3
 Min.   : 0.00   Length:2160        Length:2160        Min.   :-4.337   Min.   : 0.00   Min.   :  -34.00   Length:2160
 1st Qu.: 9.00   Class :character   Class :character   1st Qu.: 4.070   1st Qu.: 0.00   1st Qu.:   11.00   Class :character
 Median :20.00   Mode  :character   Mode  :character   Median : 5.563   Median : 0.00   Median :   24.00   Mode  :character
 Mean   :19.44                                         Mean   : 5.788   Mean   : 1.50   Mean   :   38.92
 3rd Qu.:28.00                                         3rd Qu.: 7.415   3rd Qu.: 0.00   3rd Qu.:   33.00
 Max.   :61.00                                         Max.   :11.742   Max.   :19.96   Max.   :34000.00
> glimpse(tuna2)
Rows: 2,160
Columns: 7
$ BusA1       <int> 9, 5, 5, 2, 0, 0, 10, 30, 41, 39, 28, 29, 34, 22, 17, 19, 21, 40, 36, 27, 20, 19, 19, 13, 8, 5, 4, 2, 0, 0, 9,…
$ Date        <chr> "01/01/2019", "01/01/2019", "01/01/2019", "01/01/2019", "01/01/2019", "01/01/2019", "01/01/2019", "01/01/2019"…
$ Hour        <chr> "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14", "15", "16", "17", "18", "19", …
$ Temperature <dbl> 2.469638, 1.635323, 2.470651, 2.156461, 2.798946, 2.875518, 2.824573, 3.726758, 4.809061, 5.782161, 6.785449, …
$ Rainfall    <dbl> 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, …
$ BusA2       <int> 11, 5, 5, 3, 0, 0, 10, 31, 45, 43, 34, 36, 39, 26, 23, 20, 27, 53, 47, 33, 29, 23, 20, 16, 11, 5, 6, 3, 0, 0, …
$ BusA3       <chr> "0", "46", "15", "56", "0", "0", "23", "56", "47", "38", "55", "36", "60", "33", "25", "52", "43", "22", "8", …
>
```

Step2:- Checking for Duplicate Values

```
> #checking for duplicate values
> dup_records <- duplicated(tuna2)
> sum(dup_records)
[1] 0
>
```

There were no duplicate values present in the dataset.

Step3:-Checking For NA Values

```
30
31 #checking for NA values
32 is.na(tuna2)
33 # list rows of data that have missing values
34 tuna2[!complete.cases(tuna2),]
35
```

The row 923 had a NA value which was removed .

Step4:-Replacing the Negative Values From BUSA2 from NA and then removing all the incorrect values.

```
56
57 #replacing negative value of BUSA2 with NA
58 tuna2$BusA2[tuna2$BusA2 < 0] <- NA
59 is.na(tuna2$BusA2)
60 tuna2[!complete.cases(tuna2),]
61
62
```

The row 732 and 923 had NA values which were removed

```
> tuna2[!complete.cases(tuna2),]
    BusA1       Date Hour Temperature Rainfall BusA2 BusA3
732    29 31/01/2019   11    8.467258        0    NA    28
923    24 08/02/2019 <NA>    7.835228        0    32    10
> |
```

Now our data is Ready to feed into our mAchine learning model for training as we have pre-processed and cleaned the data to make it more suitable for training.

Regression Model Implementation

The linear regression and KNN regression algorithms were applied to both the dataset,let's observe each of them.

Dataset:Artwork Value

Linear Regression

Step1:- Splitting the dataset into training and test dataset.

```
63 #splitting the data
64 set.seed(2)
65 library(caTools)
66 split <- sample.split(tuna1_new1, SplitRatio = 0.8)
67 split
68 train <- subset(tuna1_new1,split="TRUE")
69 test  <- subset(tuna1_new1, split="FALSE")
70 train
71 test
72
73
```

Step2:- K-Fold Cross validation

```
117
118 #k fold cross validation linear regression
119 ctrl <- trainControl(method="cv" , number=10)
120
```

We choose k value to be 10 to resample the dataset.

Step3:- implementation of model

```
> #k fold cross validation linear regression
> ctrl <- trainControl(method="cv" , number=10)
> #model Implementation
> model <- train(Art.Price~.,data=train , method="lm" ,trControl=ctrl)
> print(model)
Linear Regression

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 84, 85, 86, 87, 86, 84, ...
Resampling results:

  RMSE      Rsquared   MAE
  53041.09  0.8655692  46274.1

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

KNN Regression

Step1:- Splitting the dataset into training and test dataset.

```
63 #splitting the data
64 set.seed(2)
65 library(caTools)
66 split <- sample.split(tuna1_new1, SplitRatio = 0.8)
67 split
68 train <- subset(tuna1_new1,split="TRUE")
69 test  <- subset(tuna1_new1, split="FALSE")
70 train
71 test
72
73
```

Step2:- K-Fold Cross validation

```
157
158
159 #k fold cross validation knn
160 trControl <- trainControl(method  = "cv",
161                           number  = 5)
162
```

We choose k value to be 10 to resample the dataset.

Step3:- implementation of model

We have selected all the features to train the ml model.

```
> #implementation of model
> model_knn <- train(Art.Price ~ .,
+                    method     = "knn",
+                    tuneGrid   = expand.grid(k = 1:10),
+                    trControl  = trControl,
+                    metric     = "RMSE",
+                    data       = train)
> print(model_knn)
k-Nearest Neighbors

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 75, 75, 76, 77, 77
Resampling results across tuning parameters:

  k   RMSE       Rsquared   MAE
   1  101511.25  0.5056746  76778.59
   2   98707.99  0.5235052  81951.40
   3   85288.83  0.6291709  70783.30
   4   84850.15  0.6348330  72919.46
   5   83942.59  0.6487896  72237.79
   6   86818.27  0.6221627  75671.49
   7   84359.59  0.6430770  74047.58
   8   83552.13  0.6440929  74008.90
   9   81408.47  0.6604770  72160.22
  10   80423.50  0.6700842  71828.63

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 10.
```

Dataset:Bus Station

Linear Regression

Step1:- Splitting the dataset into training and test dataset.

```
67
68  #splitting the data
69  set.seed(2)
70  library(caTools)
71  split2 <- sample.split(tuna2, SplitRatio = 0.8)
72  split2
73  train2 <- subset(tuna2,split="TRUE")
74  test2  <- subset(tuna2, split="FALSE")
75  train2
76  test2
77
```

Step2:- K-Fold Cross validation

```
123
124 #k fold cross validation linear regression
125 ctrl2 <- trainControl(method="cv" , number=10)
126
```

We choose k value to be 10 to resample the dataset.

Step3:- implementation of model.

We have dropped two features: Date and Rainfall having very high correlation with the data.

```
> #k fold cross validation linear regression
> ctrl2 <- trainControl(method="cv" , number=10)
> #implementation of model
> model2 <- train(BusA1~.,data=train2 , method="lm" ,trControl=ctrl)
> print(model2)
Linear Regression

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1939, 1938, 1938, 1939, 1939, 1938, ...
Resampling results:

  RMSE      Rsquared   MAE
  200.7145  0.3416063  20.02413

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

KNN Regression

Step1:- Splitting the dataset into training and test dataset.

```
67
68  #splitting the data
69  set.seed(2)
70  library(caTools)
71  split2 <- sample.split(tuna2, SplitRatio = 0.8)
72  split2
73  train2 <- subset(tuna2,split="TRUE")
74  test2  <- subset(tuna2, split="FALSE")
75  train2
76  test2
77
```

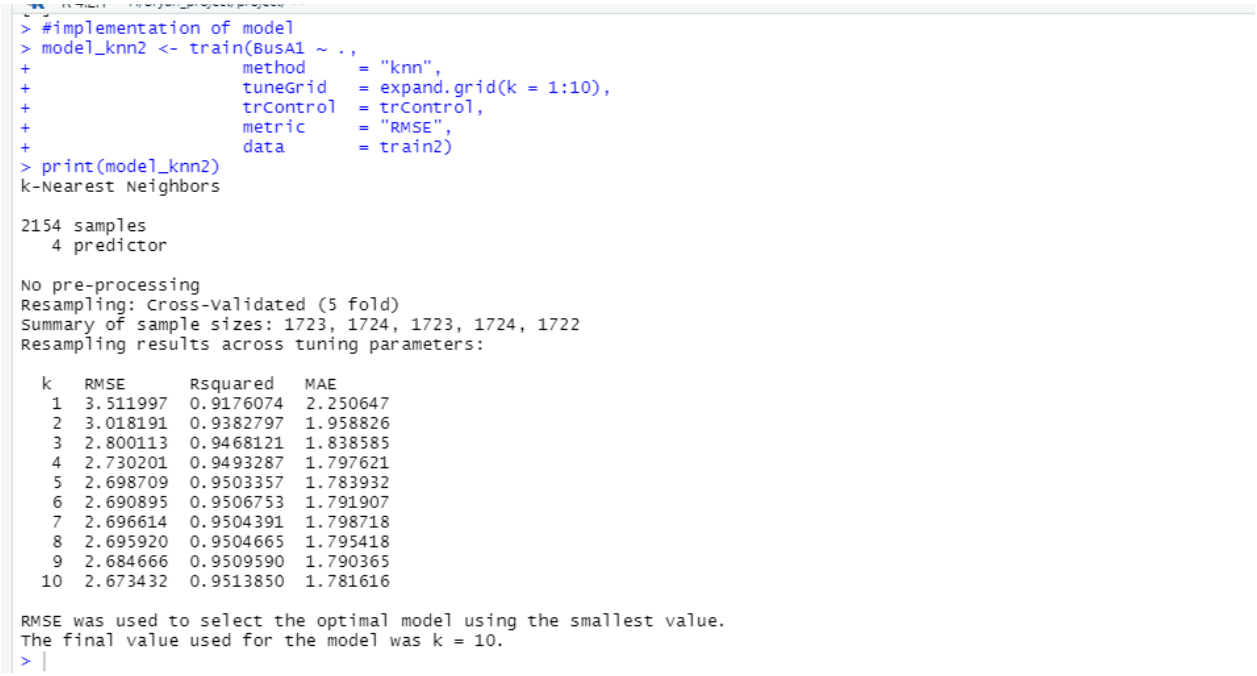
Step2:- K-Fold Cross validation

```
169
170
171  #k fold cross validation knn
172  trControl2 <- trainControl(method  = "cv",
173                             number  = 5)
174
```

We choose k value to be 10 to resample the dataset.

Step3:- implementation of model.

We have dropped two features: Date and Rainfall having very high correlation with the data.



```
> #implementation of model
> model_knn2 <- train(BusA1 ~ .,
+                     method     = "knn",
+                     tuneGrid   = expand.grid(k = 1:10),
+                     trControl  = trControl,
+                     metric     = "RMSE",
+                     data       = train2)
> print(model_knn2)
k-Nearest Neighbors

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 1723, 1724, 1723, 1724, 1722
Resampling results across tuning parameters:

  k   RMSE      Rsquared   MAE
   1  3.511997  0.9176074  2.250647
   2  3.018191  0.9382797  1.958826
   3  2.800113  0.9468121  1.838585
   4  2.730201  0.9493287  1.797621
   5  2.698709  0.9503357  1.783932
   6  2.690895  0.9506753  1.791907
   7  2.696614  0.9504391  1.798718
   8  2.695920  0.9504665  1.795418
   9  2.684666  0.9509590  1.790365
  10  2.673432  0.9513850  1.781616

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 10.
>
```

Experimental Results and Analysis

Now let's go through the results obtained in each Dataset one by one.

Dataset:Bus Station

Linear regression

Let's first calculate the MSE,MAE,MAPE,RMSE value on testing data

```
> #implementation of model
> model2 <- train(BusA1~.,data=train2 , method="lm" ,trControl=ctrl)
> print(model2)
Linear Regression

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1938, 1939, 1938, 1938, 1939, 1938, ...
Resampling results:

  RMSE     Rsquared   MAE
  200.858  0.3432875  19.97934
```

```
> #mse for linear regression
> mse2 <- mean(pred_k_lm-test2$BusA1)^2
> mse2
[1] 1.756022e-27
> #calculating mape
> mape(test2$BusA1,pred_k_lm)
[1] 8.390576
>
```

Now let's calculate the MSE,MAE,MAPE,RMSE value on training data.

```
> #evaluation metrics of training data
> ctrl2_t <- trainControl(method="cv" , number=10)
> model2_t <- train(BusA1~.,data=tuna2 , method="lm" ,trControl=ctrl)
> print(model2_t)
Linear Regression

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1939, 1939, 1938, 1938, 1938, 1939, ...
Resampling results:

  RMSE     Rsquared   MAE
  201.342  0.3383326  20.08637

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

```
> #mse for linear regression
> mse2 <- mean(tuna2$BusA1-pred_k_lmt)^2
> mse2
[1] 1.756022e-27
> #calculating mape
> mape(train2$BusA1,pred_k_lmt)
[1] 8.390576
>
```

Calculating the Residual Standard Error And F-Statistic for linear Regression Model.

```
Call:
lm(formula = BusA1 ~ ., data = train2)

Residuals:
    Min      1Q  Median      3Q     Max
-18.259  -6.857  -2.110   4.702  42.488

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.4176518  0.6864167  -3.522 0.000437 ***
Hour         0.7793743  0.0303256  25.700  < 2e-16 ***
Temperature  1.8713681  0.0922202  20.292  < 2e-16 ***
BusA2        0.0001831  0.0002824   0.648 0.516860
BusA3        0.0734510  0.0111774   6.571 6.24e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.59 on 2149 degrees of freedom
Multiple R-squared:  0.3758,    Adjusted R-squared:  0.3747
F-statistic: 323.5 on 4 and 2149 DF,  p-value: < 2.2e-16

> |
```

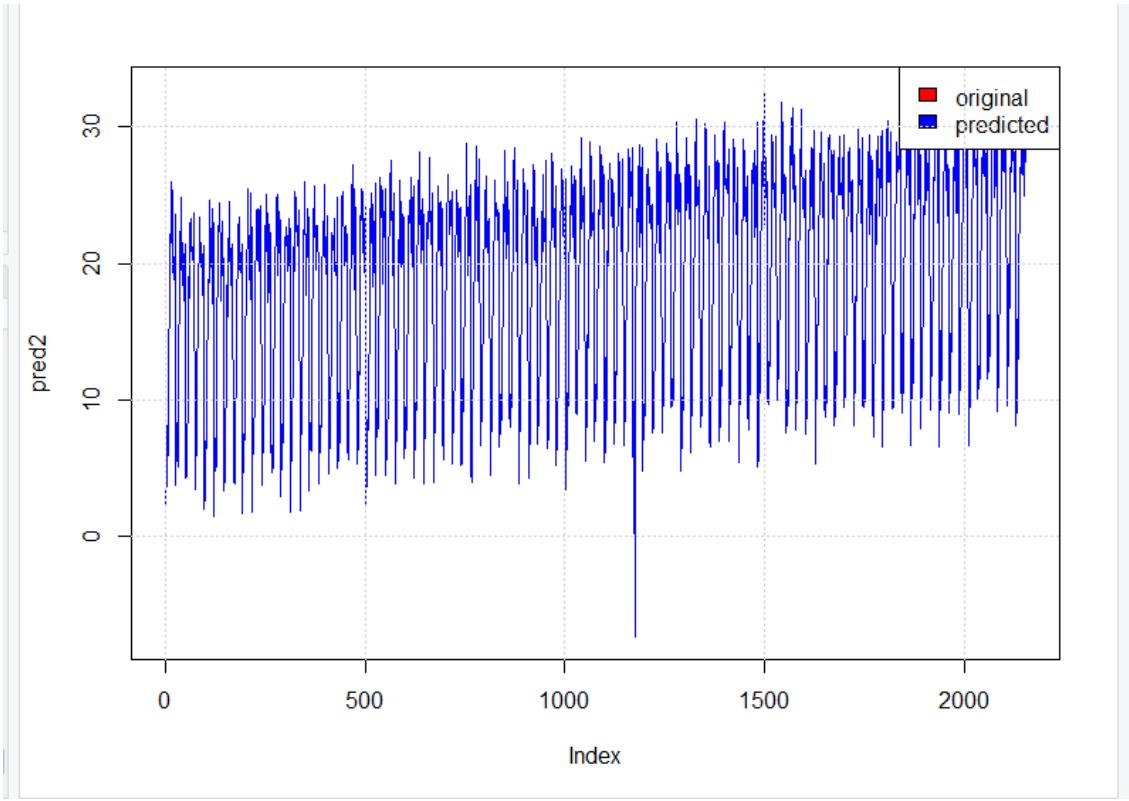
Dataset	RMSE	MAE	MSE	MAPE	RSE	F-STAT
Test data	200.8	19.9	1.7e-27	8.3	9.59	323.5
Train data	201.3	20.08	1.75e-27	8.3	9.59	323.5

Visualization OF Actual v/s Predicted Values

Dataset:Bus Station

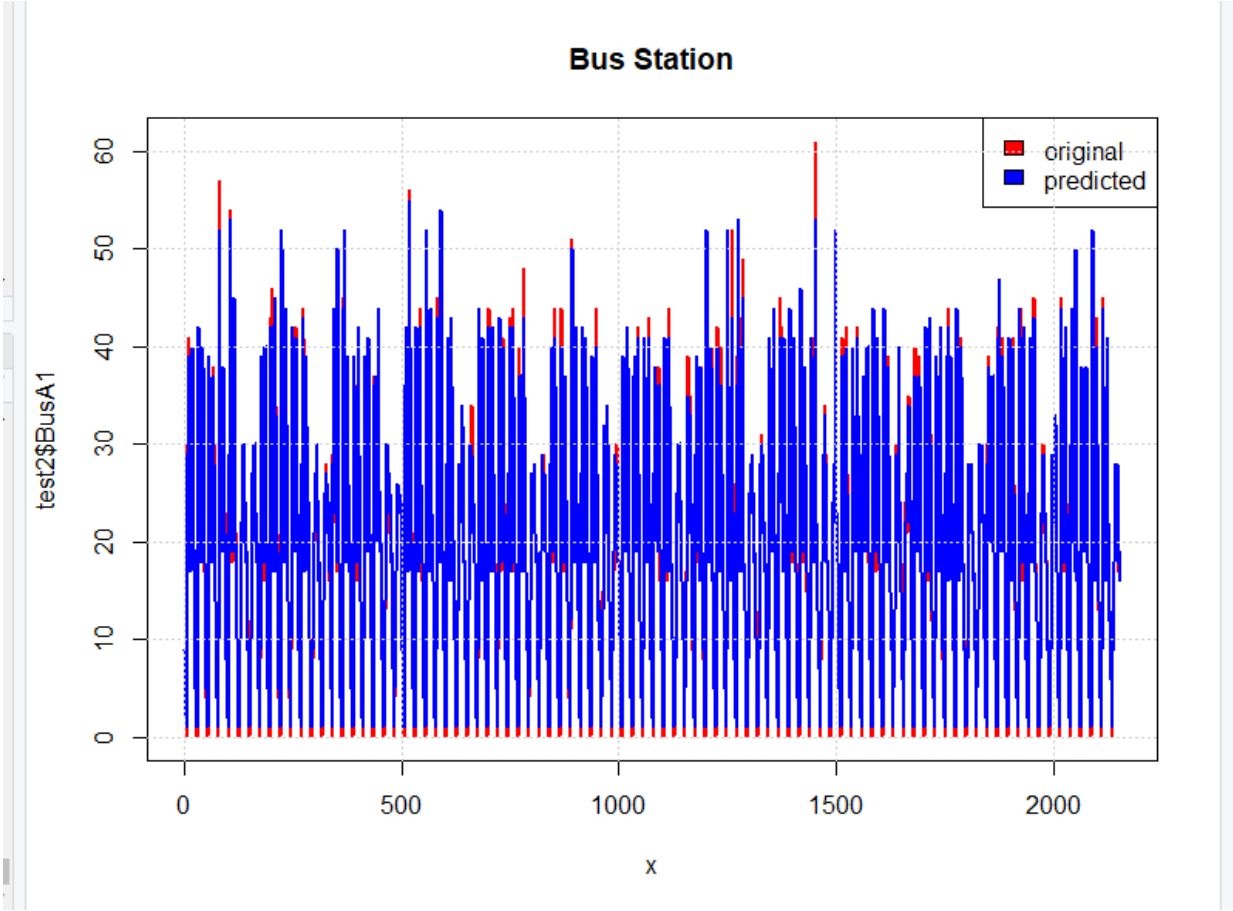
Linear Regression: Now we will visualize the actual v/s predicted value of the dataset trained using linear regression.As we can see from the graph obtained that the Actual Values are overlapping with Predicted values which shows that our Model was trained in a good fashion.

```
103
104 #comparing predicted vs actual values
105 plot(test2$BusA1 ,type="l", lty=1.8,col="red")
106 lines(pred2,type="l",col="blue")
107 plot(pred2,type="l", lty=1.8,col="blue")
108 legend("topright",  legend = c("original", "predicted"),
109        fill = c("red", "blue"), col = 2:3,  adj = c(0, 0.6))
110 grid()
111
```



KNN Regression: Now we will visualize the actual v/s predicted value of the dataset trained using linear regression.As we can see from the graph obtained that the Actual Values are not overlapping completely with Predicted values .

```
123
124 #comparing predicted vs actual values
125 x = 1:length(test2$BusA1)
126
127 plot(x, test2$BusA1, col = "red", type = "l", lwd=2,
128      main = "Bus Station")
129 lines(x, predictions2, col = "blue", lwd=2)
130 legend("topright",  legend = c("original", "predicted"),
131        fill = c("red", "blue"), col = 2:3,  adj = c(0, 0.6))
132 grid()
133
```



KNN Regression

Let's first calculate the MSE,MAE,MAPE,RMSE value on testing data

```
> #implementation of model
> model_knn2 <- train(BusA1 ~ .,
+                     method     = "knn",
+                     tuneGrid   = expand.grid(k = 1:10),
+                     trControl  = trControl,
+                     metric     = "RMSE",
+                     data       = train2)
> print(model_knn2)
k-Nearest Neighbors

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 1723, 1724, 1723, 1724, 1722
Resampling results across tuning parameters:

  k   RMSE      Rsquared   MAE
   1  3.511997  0.9176074  2.250647
   2  3.018191  0.9382797  1.958826
   3  2.800113  0.9468121  1.838585
   4  2.730201  0.9493287  1.797621
   5  2.698709  0.9503357  1.783932
   6  2.690895  0.9506753  1.791907
   7  2.696614  0.9504391  1.798718
   8  2.695920  0.9504665  1.795418
   9  2.684666  0.9509590  1.790365
  10  2.673432  0.9513850  1.781616

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 10.
>
```

```
> #calculating mse
> mse2_k <- mean(pred2_k-tuna2$BusA1)^2
> mse2_k
[1] 0.002198891
> #calculating mape
> mape(train2$BusA1,pred2_k)
[1] 0.08767981
>
```

Now let's calculate the MSE,MAE,MAPE,RMSE value on training data.

```
> #evaluation metrics of training data
> trControl2_t <- trainControl(method  = "cv",
+                              number  = 5)
> model_knn2_t <- train(BusA1 ~ .,
+                       method     = "knn",
+                       tuneGrid   = expand.grid(k = 1:10),
+                       trControl  = trControl,
+                       metric     = "RMSE",
+                       data       = tuna2)
> print(model_knn2_t)
k-Nearest Neighbors

2154 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 1723, 1723, 1723, 1722, 1725
Resampling results across tuning parameters:

  k   RMSE      Rsquared   MAE
   1  3.507184  0.9186598  2.223786
   2  3.005979  0.9393267  1.926460
   3  2.809596  0.9466411  1.829851
   4  2.757549  0.9485504  1.802455
   5  2.701288  0.9505560  1.764049
   6  2.693166  0.9509145  1.760479
   7  2.683528  0.9512708  1.761514
   8  2.692573  0.9508856  1.769592
   9  2.669651  0.9517290  1.763361
  10  2.668511  0.9517612  1.771412

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 10.
```

```
> #mse for linear regression
> mse2 <- mean(tuna2$BusA1-pred_k_lmtkn)^2
> mse2
[1] 0.002198891
> #calculating mape
> mape(train2$BusA1,pred_k_lmtkn)
[1] 0.08767981
>
```

Dataset	RMSE	MAE	MSE	MAPE
Test Data	2.67	1.78	0.002	0.08
Train Data	2.66	1.77	0.002	0.08

Dataset:Artwork Value

Linear Regression

Let’s first calculate the MSE,MAE,MAPE,RMSE value on testing data

```
> #k fold cross validation linear regression
> ctrl <- trainControl(method="cv" , number=10)
> #model Implementation
> model <- train(Art.Price~.,data=train , method="lm" ,trControl=ctrl)
> print(model)
Linear Regression

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 84, 85, 86, 87, 86, 84, ...
Resampling results:

  RMSE      Rsquared   MAE
  53041.09  0.8655692  46274.1

Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
> #mse for linear regression
> mse_k <- mean(pred_kt-test$Art.Price)^2
> mse_k
[1] 9.301393e-20
> #calculating mape
> mape(test$Art.Price,pred_kt)
[1] 0.2291241
>
```

Now let's calculate the MSE,MAE,MAPE,RMSE value on training data.

```
Linear Regression

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 85, 85, 86, 87, 85, 86, ...
Resampling results:

  RMSE      Rsquared   MAE
  154838.9  0.2975446  107375.8

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

```
> #mse for linear regression
> mse2_k <- mean(tuna1_new1$Art.Price-pred1_k_lmt)^2
> mse2_k
[1] 5.070259e-22
> #calculating mape
> mape(tuna1_new1$Art.Price,pred1_k_lmt)
[1] 0.01554863
>
```

Calculating the Residual Standard Error And F-Statistic for the linear regression model.

```
Residual standard error: 23270 on 11 degrees of freedom
Multiple R-squared:  0.9967,    Adjusted R-squared:  0.9721
F-statistic: 40.51 on 83 and 11 DF,  p-value: 6.912e-08

>
```

Dataset	RMSE	MSE	MAE	MAPE	RSE	F-STATISTIC
Testing data	53041.09	9.30e-20	46274.1	0.22	23270	40.51
Training data	154838.9	5.0	107375.8	0.01	23270	40.51

KNN Regression

Let's first calculate the MSE,MAE,MAPE,RMSE value on testing data

```
k-Nearest Neighbors

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 76, 76, 76, 75, 77
Resampling results across tuning parameters:

  k   RMSE      Rsquared   MAE
   1  92513.29  0.5859470  68430.26
   2  94160.14  0.5608610  77537.94
   3  86123.70  0.6327509  72423.59
   4  85507.96  0.6382520  72016.21
   5  82965.40  0.6608399  71276.24
   6  82388.67  0.6714568  71314.48
   7  83826.08  0.6552980  74169.35
   8  81442.04  0.6751667  71883.59
   9  82436.57  0.6651733  72917.90
  10  80175.95  0.6803283  72185.39
```

```
> #calculating mse
> mse <- mean(pred1-tuna1_new1$Art.Price)^2
> mse
[1] 1529801
> tuna1_new1$Age <- as.numeric(tuna1_new1$Age)
> tuna1_new1$Verified <- as.numeric(tuna1_new1$Verified)
> mape(train$Art.Price,pred1)
[1] 0.3322849
>
```

Now let's calculate the MSE,MAE,MAPE,RMSE value on training data.

```
k-Nearest Neighbors

95 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 75, 75, 76, 77, 77
Resampling results across tuning parameters:

  k   RMSE       Rsquared   MAE
   1  101511.25  0.5056746  76778.59
   2   98707.99  0.5235052  81951.40
   3   85288.83  0.6291709  70783.30
   4   84850.15  0.6348330  72919.46
   5   83942.59  0.6487896  72237.79
   6   86818.27  0.6221627  75671.49
   7   84359.59  0.6430770  74047.58
   8   83552.13  0.6440929  74008.90
   9   81408.47  0.6604770  72160.22
  10   80423.50  0.6700842  71828.63

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 10.
> 
```

```
> #mse for linear regression
> mse_ta <- mean(tuna1_new1$Art.Price-pred1_kn_t)^2
> mse_ta
[1] 1529801
> #calculating mape
> mape(tuna1_new1$Art.Price,pred1_kn_t)
[1] 0.3322849
> 
```

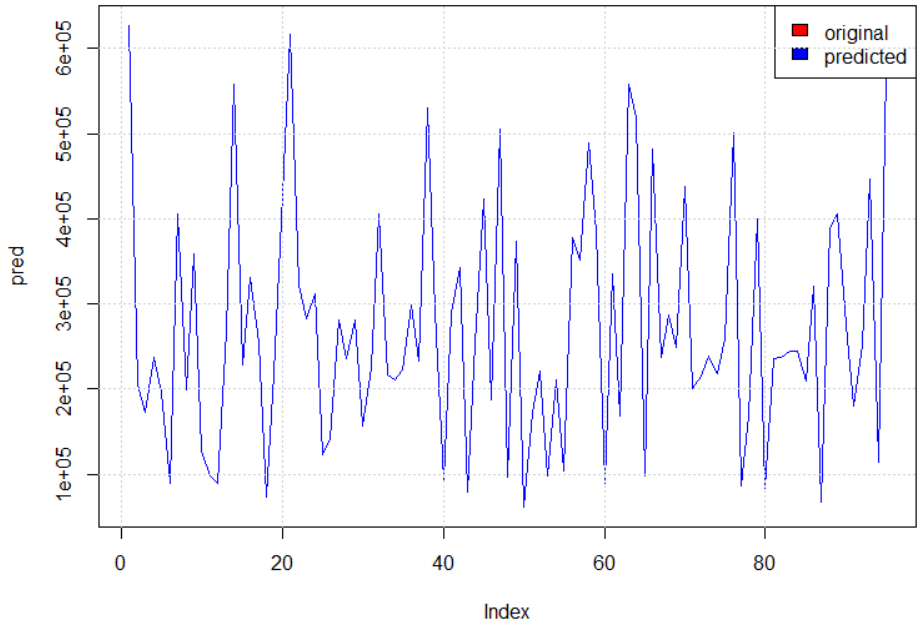
Dataset	RMSE	MSE	MAE	MAPE
Training Data	80423.5	1529801	71828.6	0.33
Testing data	8017.95	1529801	72185.39	0.33

Visualization OF Actual v/s Predicted Values

Dataset:Artwork Value

Linear Regression: Now we will visualize the actual v/s predicted value of the dataset trained using linear regression.As we can see from the graph obtained that the Actual Values are overlapping with Predicted values which shows that our Model was trained in a good fashion.

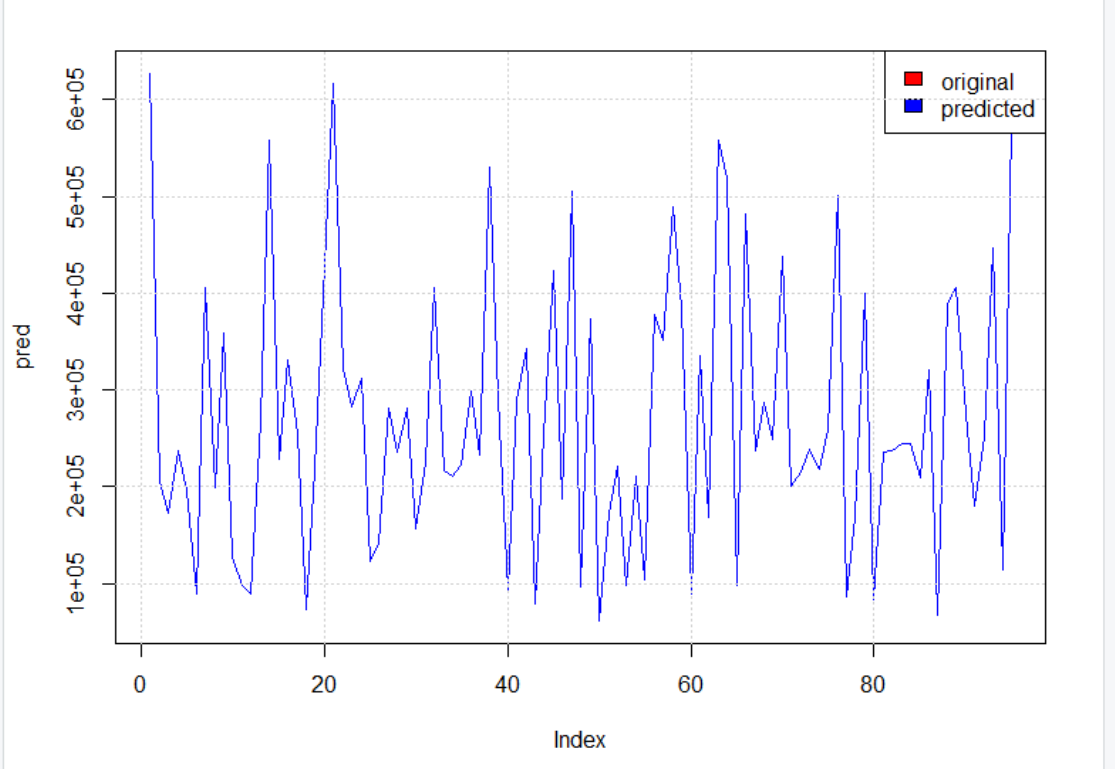
```
#comparing predicted vs actual values
plot(test$Art.Price ,type="l", lty=1.8,col="red")
lines(pred,type="l",col="blue")
plot(pred,type="l", lty=1.8,col="blue")
legend("topright",  legend = c("original", "predicted"),
       fill = c("red", "blue"), col = 2:3,  adj = c(0, 0.6))
grid()
```



KNN Regression: Now we will visualize the actual v/s predicted value of the dataset trained using KNN regression.As we can see from the graph obtained that the Actual Values are overlapping with

Predicted values which shows that our Model was trained in a good fashion.

```
121 #comparing predicted vs actual values
122 x = 1:length(test$Art.Price)
123 lines(predictions,type="l",col="blue")
124
125 plot(x, test$Art.Price, col = "red", type = "l", lwd=2,
126      main = "Artwork Value")
127 legend("topright",  legend = c("original", "predicted"),
128        fill = c("red", "blue"), col = 2:3,  adj = c(0, 0.6))
129 grid()
```



Conclusion

Dataset:Artwork Value

The Artwork Value dataset shows better performance with linear regression model as it has low RMSE value as compared to KNN regression model.As there were very low observations present in the dataset thus the KNN model wasn't able to fit on the dataset properly.Linear regression being less complex algorithm fits better on the dataset.

Dataset:Bus Station

The Bus Station dataset shows better performance with KNN regression model as it has Low RMSE value as compared to linear regression model . The following dataset had a good amount of observations thus the KNN model was able to predict the values with more accuracy as it was able to fit better on the dataset.

References

[1]https://www.statology.org/residual-standard-error-r/
[2]https://www.datatechnotes.com/2020/10/knn-regresion-example-in-r.html
[3]https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/
[4]https://www.r-project.org/about.html
[5]tutorialspoint.com/r/index.htm