**Predict Price of the computer**

A dataframe containing :

price : price in US dollars of 486 PCs

speed : clock speed in MHz

hd : size of hard drive in MB

ram : size of Ram in in MB

screen : size of screen in inches

cd : is a CD-ROM present ?

multi : is a multimedia kit (speakers, sound card) included ?

premium : is the manufacturer was a "premium" firm (IBM, COMPAQ) ?

ads : number of 486 price listings for each month

trend : time trend indicating month starting from January of 1993 to November of 1995.

Here I am using R programming language to solve this problem

**#Install ‘psych’ package to perform different performance metrics**

#importing dataset

comp\_datas <- read.csv(file.choose(),header = T)

View(comp\_datas)

# display first six data from a data set

head(comp\_datas)

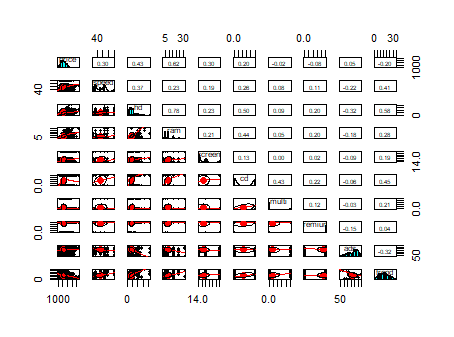
|  |
| --- |
| X price speed hd ram screen cd multi premium ads trend  1 1 1499 25 80 4 14 no no yes 94 1  2 2 1795 33 85 2 14 no no yes 94 1  3 3 1595 25 170 4 15 no no yes 94 1  4 4 1849 25 170 8 14 no no no 94 1  5 5 3295 33 340 16 14 no no yes 94 1  6 6 3695 66 340 16 14 no no yes 94 1 |
|  |
| |  | | --- | | > | |

#drop first column

comp\_datas<-comp\_datas[-1]

head(comp\_datas)

|  |
| --- |
| price speed hd ram screen cd multi premium ads trend  1 1499 25 80 4 14 no no yes 94 1  2 1795 33 85 2 14 no no yes 94 1  3 1595 25 170 4 15 no no yes 94 1  4 1849 25 170 8 14 no no no 94 1  5 3295 33 340 16 14 no no yes 94 1  6 3695 66 340 16 14 no no yes 94 1  #display the structure of the data |
| str(comp\_datas)   |  | | --- | | 'data.frame': 6259 obs. of 10 variables:  $ price : int 1499 1795 1595 1849 3295 3695 1720 1995 2225 2575 ...  $ speed : int 25 33 25 25 33 66 25 50 50 50 ...  $ hd : int 80 85 170 170 340 340 170 85 210 210 ...  $ ram : int 4 2 4 8 16 16 4 2 8 4 ...  $ screen : int 14 14 15 14 14 14 14 14 14 15 ...  $ cd : chr "no" "no" "no" "no" ...  $ multi : chr "no" "no" "no" "no" ...  $ premium: chr "yes" "yes" "yes" "no" ...  $ ads : int 94 94 94 94 94 94 94 94 94 94 ...  $ trend : int 1 1 1 1 1 1 1 1 1 1 ...  **#here we can see that three categorical variable in the data set. convert that variable into a numerical format**  **# here I am using label encoding**  install.packages("plyr")  library(plyr) | |  | | |  | | --- | | count(comp\_datas,"cd" )  cd freq  1 no 3351  2 yes 2908  > count(comp\_datas,"multi" )  multi freq  1 no 5386  2 yes 873  > count(comp\_datas,"premium" )  premium freq  1 no 612  2 yes 5647 | |  | | |  | | --- | | > | |   **#variable "cd", "multi" , "premium" are categorical variable, convert them into numerical variable**  comp\_datas$cd<-revalue(comp\_datas$cd,c("yes"="1", "no"="0"))  comp\_datas$multi<-revalue(comp\_datas$multi,c("yes"="1", "no"="0"))  comp\_datas$premium<-revalue(comp\_datas$premium,c("yes"="1", "no"="0"))  head(comp\_data)   |  | | --- | | price speed hd ram screen cd multi premium ads trend  1 1499 25 80 4 14 0 0 1 94 1  2 1795 33 85 2 14 0 0 1 94 1  3 1595 25 170 4 15 0 0 1 94 1  4 1849 25 170 8 14 0 0 0 94 1  5 3295 33 340 16 14 0 0 1 94 1  6 3695 66 340 16 14 0 0 1 94 1 | |  | | |  | | --- | | > | |   **#change the character variable datatypes to numerical data**  comp\_datas$cd<- as.numeric(comp\_datas$cd)  comp\_datas$multi<- as.numeric(comp\_datas$multi)  comp\_datas$premium<- as.numeric(comp\_datas$premium)  str(comp\_datas)     |  | | --- | | 'data.frame': 6259 obs. of 10 variables:  $ price : int 1499 1795 1595 1849 3295 3695 1720 1995 2225 2575 ...  $ speed : int 25 33 25 25 33 66 25 50 50 50 ...  $ hd : int 80 85 170 170 340 340 170 85 210 210 ...  $ ram : int 4 2 4 8 16 16 4 2 8 4 ...  $ screen : int 14 14 15 14 14 14 14 14 14 15 ...  $ cd : num 0 0 0 0 0 0 1 0 0 0 ...  $ multi : num 0 0 0 0 0 0 0 0 0 0 ...  $ premium: num 1 1 1 0 1 1 1 1 1 1 ...  $ ads : int 94 94 94 94 94 94 94 94 94 94 ...  $ trend : int 1 1 1 1 1 1 1 1 1 1 ... | |  | | |  | | --- | | > | | |   **#check whether the data set contain null values**  sum(is.na(comp\_datas))  **# data set contain zero null values**  # summary of dataset  summary(comp\_datas)  price speed hd ram screen  Min. : 949 Min. : 25.00 Min. : 80.0 Min. : 2.000 Min. :14.00  1st Qu.:1794 1st Qu.: 33.00 1st Qu.: 214.0 1st Qu.: 4.000 1st Qu.:14.00  Median :2144 Median : 50.00 Median : 340.0 Median : 8.000 Median :14.00  Mean :2220 Mean : 52.01 Mean : 416.6 Mean : 8.287 Mean :14.61  3rd Qu.:2595 3rd Qu.: 66.00 3rd Qu.: 528.0 3rd Qu.: 8.000 3rd Qu.:15.00  Max. :5399 Max. :100.00 Max. :2100.0 Max. :32.000 Max. :17.00  cd multi premium ads trend  Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. : 39.0 Min. : 1.00  1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:1.0000 1st Qu.:162.5 1st Qu.:10.00  Median :0.0000 Median :0.0000 Median :1.0000 Median :246.0 Median :16.00  Mean :0.4646 Mean :0.1395 Mean :0.9022 Mean :221.3 Mean :15.93  3rd Qu.:1.0000 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:275.0 3rd Qu.:21.50  Max. :1.0000 Max. :1.0000 Max. :1.0000 Max. :339.0 Max. :35.00  **#Visualization**  **#Here we are using three types of analysis techniques:**    **#Univariate analysis**  **#Bivariate ana**lysis  **#Univariate analysis**  **#Univariate analysis means one variable analysis.'Uni' means 'one' and 'variate' means'variable'. Univariate analysis is to analyse one variable or one features Univariate basically tells us how data in each feature is distributed and also tells us about central tendencies like mean, median, and mode.**  **#To do univariate data analysis we use following ploting mechanisms:**    **#Histograms**  **#Boxplot**  **#Histograms:**  **#following code shows histogram reprepresentation of variable in a dataset**  **hist(price)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\price**  **hist(speed)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\speed**  **hist(hd)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\hd**  **hist(ram)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\ram**  **hist(screen)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\screen**  **hist(cd)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\cd**  **hist(ads)**  **C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\ads**  **hist(trend)**  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\trend  **#boxplot**  **#box plot main used to identify the outliers**  boxplot(trend)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\1  boxplot(ads)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\2  boxplot(price) # many outliers  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\3  boxplot(speed)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\4  boxplot(ram)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\5  boxplot(screen)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\6  **#Bivariate representation**  **#scatter plot**  plot(trend,price)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\1  plot(ads,price)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\2  plot(speed,price)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\3  plot(screen,price)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\4  # in the scatter pot representation we can see that there is no linearity between dependant variable price and independent variable  **#pair plot**  pairs(comp\_datas)  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\2  **# in this pair plot also we can see that there is no linearity present in the data**  **#using pairs.panels to show the correlation between variables**  pairs.panels(comp\_datas) |
|  |



# using this pair plot we can conclude that there is no multicollinearity present in the data set

**# here price is our target variable, lets check the correlation between price and other variable**

cor(comp\_datas)

|  |
| --- |
| cor(comp\_datas)  price speed hd ram screen cd  price 1.00000000 0.30097646 0.43025779 0.62274824 0.296041474 0.19734334  speed 0.30097646 1.00000000 0.37230410 0.23476050 0.189074122 0.25825980  hd 0.43025779 0.37230410 1.00000000 0.77772630 0.232801530 0.50357041  ram 0.62274824 0.23476050 0.77772630 1.00000000 0.208953740 0.43850441  screen 0.29604147 0.18907412 0.23280153 0.20895374 1.000000000 0.12948766  cd 0.19734334 0.25825980 0.50357041 0.43850441 0.129487662 1.00000000  multi -0.01665139 0.08417193 0.09280483 0.04549689 -0.001740414 0.43217930  premium -0.08069636 0.11420791 0.19692359 0.19714459 0.018745223 0.21607660  ads 0.05454047 -0.21523206 -0.32322200 -0.18166971 -0.093919429 -0.06109108  trend -0.19998694 0.40543833 0.57779013 0.27684384 0.188614445 0.44578018  multi premium ads trend  price -0.016651388 -0.08069636 0.05454047 -0.19998694  speed 0.084171934 0.11420791 -0.21523206 0.40543833  hd 0.092804830 0.19692359 -0.32322200 0.57779013  ram 0.045496894 0.19714459 -0.18166971 0.27684384  screen -0.001740414 0.01874522 -0.09391943 0.18861444  cd 0.432179298 0.21607660 -0.06109108 0.44578018  multi 1.000000000 0.12477474 -0.03039426 0.21090743  premium 0.124774741 1.00000000 -0.15202274 0.04210738  ads -0.030394260 -0.15202274 1.00000000 -0.31855251  trend 0.210907431 0.04210738 -0.31855251 1.00000000 |
|  |
| |  | | --- | | > | |

**## create training and test data for price prediction**

**# here I am using 70% of data for training and 30% data for testing**

install.packages("caTools")

library(caTools)

**##use caTools function to split, SplitRatio for 70%:30% splitting**

data= sample.split(comp\_datas,SplitRatio = 0.3)

**#subsetting into Test data**

test\_data =subset(comp\_datas,data==TRUE)

**#subsetting into Train data**

train\_data =subset(comp\_datas,data==FALSE)

# check number of records present in the data set

nrow(test\_data)

1878

nrow(train\_data)

4381

head(train\_data) # display first 6 data from training data set

|  |
| --- |
| price speed hd ram screen cd multi premium ads trend  1 1499 25 80 4 14 0 0 1 94 1  2 1795 33 85 2 14 0 0 1 94 1  3 1595 25 170 4 15 0 0 1 94 1  5 3295 33 340 16 14 0 0 1 94 1  7 1720 25 170 4 14 1 0 1 94 1  8 1995 50 85 2 14 0 0 1 94 1 |
|  |
| |  | | --- | | > | |

head(test\_data) # display first 6 data from testing data set

|  |
| --- |
| price speed hd ram screen cd multi premium ads trend  4 1849 25 170 8 14 0 0 0 94 1  6 3695 66 340 16 14 0 0 1 94 1  9 2225 50 210 8 14 0 0 1 94 1  14 2295 25 245 8 14 0 0 1 94 1  16 2225 50 130 4 14 0 0 1 94 1  19 2095 33 250 4 15 0 0 1 94 1 |
|  |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | >  **# create first model**  model1=lm(price~speed+hd+ram+screen+cd+multi+premium+ads+trend,data=train\_data)  summary(model1)   |  | | --- | | Call:  lm(formula = price ~ speed + hd + ram + screen + cd + multi +  premium + ads + trend, data = train\_data)  Residuals:  Min 1Q Median 3Q Max  -1117.86 -171.81 -10.65 146.46 2008.77  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) 399.50107 72.34868 5.522 3.55e-08 \*\*\*  speed 9.25917 0.22150 41.802 < 2e-16 \*\*\*  hd 0.79228 0.03358 23.595 < 2e-16 \*\*\*  ram 47.28690 1.29094 36.630 < 2e-16 \*\*\*  screen 119.61613 4.81345 24.850 < 2e-16 \*\*\*  cd 69.02644 11.42463 6.042 1.65e-09 \*\*\*  multi 106.14233 13.85514 7.661 2.26e-14 \*\*\*  premium -509.78620 14.73849 -34.589 < 2e-16 \*\*\*  ads 0.55743 0.06170 9.035 < 2e-16 \*\*\*  trend -52.82593 0.75008 -70.427 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 277.8 on 4371 degrees of freedom  Multiple R-squared: 0.7755, Adjusted R-squared: 0.775  F-statistic: 1678 on 9 and 4371 DF, p-value: < 2.2e-16 | |  | | |  | | --- | | > | |   **# we can see that all variables are significant**  **#lets try to improve r squared value using different transformation methods**  **# check partial correlation**  **###Partial Correlation matrix**  install.packages("corpcor")  library(corpcor)  cor2pcor(cor(train\_data))   |  | | --- | | [,1] [,2] [,3] [,4] [,5] [,6] [,7]  [1,] 1.00000000 0.53441569 0.33611739 0.48463186 0.35184112 0.09100742 0.11510420  [2,] 0.53441569 1.00000000 -0.09892594 -0.26517922 -0.11293622 -0.02002226 -0.07447479  [3,] 0.33611739 -0.09892594 1.00000000 0.44361442 -0.09155438 0.09207451 -0.12014854  [4,] 0.48463186 -0.26517922 0.44361442 1.00000000 -0.11110102 0.09078843 -0.09958020  [5,] 0.35184112 -0.11293622 -0.09155438 -0.11110102 1.00000000 -0.03890860 -0.06146356  [6,] 0.09100742 -0.02002226 0.09207451 0.09078843 -0.03890860 1.00000000 0.39953826  [7,] 0.11510420 -0.07447479 -0.12014854 -0.09958020 -0.06146356 0.39953826 1.00000000  [8,] -0.46356371 0.28768902 0.20417428 0.25163754 0.14726947 0.13866643 0.11493062  [9,] 0.13539833 -0.11782020 -0.19892792 -0.02535071 -0.04919346 0.15044278 -0.03500741  [10,] -0.72908043 0.50605260 0.54201410 0.15755181 0.29633066 0.22109632 0.13628873  [,8] [,9] [,10]  [1,] -0.46356371 0.13539833 -0.72908043  [2,] 0.28768902 -0.11782020 0.50605260  [3,] 0.20417428 -0.19892792 0.54201410  [4,] 0.25163754 -0.02535071 0.15755181  [5,] 0.14726947 -0.04919346 0.29633066  [6,] 0.13866643 0.15044278 0.22109632  [7,] 0.11493062 -0.03500741 0.13628873  [8,] 1.00000000 -0.05020052 -0.41535405  [9,] -0.05020052 1.00000000 -0.01797063  [10,] -0.41535405 -0.01797063 1.00000000 | |  | | |  | | --- | | > | |   # use some diagnostic plot to detect outliers  install.packages("car")  library(car)  plot(model1) ### Residual Plots, QQ-Plos, Std. Residuals vs Fitted, Cook's distance  # Deletion Diagnostics for identifying influential variable  influence.measures(model1)  influenceIndexPlot(model1) # Index Plots of the influence measures  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\3  influencePlot(model1)  #records 5961,4478 considered as outliers, so remove these records from dataset and create new model |   #create second model  comp\_model2=lm(price~speed+hd+ram+screen+cd+multi+premium+ads+trend,data=train\_data[-c(5961,4478),])  summary(comp\_model2)   |  | | --- | | Call:  lm(formula = price ~ speed + hd + ram + screen + cd + multi +  premium + ads + trend, data = train\_data[-c(5961, 4478),  ])  Residuals:  Min 1Q Median 3Q Max  -1117.86 -171.81 -10.65 146.46 2008.77  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) 399.50107 72.34868 5.522 3.55e-08 \*\*\*  speed 9.25917 0.22150 41.802 < 2e-16 \*\*\*  hd 0.79228 0.03358 23.595 < 2e-16 \*\*\*  ram 47.28690 1.29094 36.630 < 2e-16 \*\*\*  screen 119.61613 4.81345 24.850 < 2e-16 \*\*\*  cd 69.02644 11.42463 6.042 1.65e-09 \*\*\*  multi 106.14233 13.85514 7.661 2.26e-14 \*\*\*  premium -509.78620 14.73849 -34.589 < 2e-16 \*\*\*  ads 0.55743 0.06170 9.035 < 2e-16 \*\*\*  trend -52.82593 0.75008 -70.427 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 277.8 on 4371 degrees of freedom  Multiple R-squared: 0.7755, Adjusted R-squared: 0.775  F-statistic: 1678 on 9 and 4371 DF, p-value: < 2.2e-16 | |  | | |  | | --- | | > | | |

#there is no difference between model 1 and model 2, lets try another methods

#check multicollinearity present in a data set

# here using Variance Inflation Factors(VIF) technique helps to identify the multicollinearity

vif(model2) # VIF is > 10 => collinearity

|  |
| --- |
| speed hd ram screen cd multi premium ads trend  1.268636 4.230803 3.005259 1.077700 1.839062 1.286137 1.098218 1.208633 1.978683 |
|  |
| |  | | --- | | > | |

# here we get VIF values of all variable less than 10, so we can say that there is no multicollinearity present in our data set

#The Akaike information criterion (AIC) is a mathematical method for evaluating how well a model fits the data it was generated from. In statistics, AIC is used to compare different possible models and determine which one is the best fit for the data.

install.packages("MASS")

library("MASS")

stepAIC(model2)

|  |
| --- |
| Start: AIC=49313.35  price ~ speed + hd + ram + screen + cd + multi + premium + ads +  trend  Df Sum of Sq RSS AIC  <none> 337357422 49313  - cd 1 2817448 340174870 49348  - multi 1 4529654 341887076 49370  - ads 1 6300174 343657596 49392  - hd 1 42967128 380324550 49837  - screen 1 47662446 385019868 49890  - premium 1 92337765 429695187 50371  - ram 1 103556616 440914037 50484  - speed 1 134867497 472224919 50785  - trend 1 382812022 720169443 52634  Call:  lm(formula = price ~ speed + hd + ram + screen + cd + multi +  premium + ads + trend, data = train\_data[-c(5961, 4478),  ])  Coefficients:  (Intercept) speed hd ram screen cd multi  399.5011 9.2592 0.7923 47.2869 119.6161 69.0264 106.1423  premium ads trend  -509.7862 0.5574 -52.8259 |
|  |
| |  | | --- | | > | |

## Lower the AIC (Akaike Information Criterion) value better is the model. AIC is used only if you build

# multiple models

# here we can see that only one AIC value, so we can take

lm(price~speed+hd+ram+screen+cd+multi+premium+ads+trend,data=train\_data)

# to improve the R-squared values, here I am using different transformation techniques

#exponential transformation

comp\_model3<-lm(log(price)~speed+hd+ram+screen+cd+multi+premium+ads+trend,data=train\_data[-c(5961,4478),])

summary(comp\_model3)

|  |
| --- |
| Call:  lm(formula = log(price) ~ speed + hd + ram + screen + cd + multi +  premium + ads + trend, data = train\_data[-c(5961, 4478),  ])  Residuals:  Min 1Q Median 3Q Max  -0.52570 -0.07457 0.00380 0.07535 0.50948  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) 6.867e+00 3.154e-02 217.745 < 2e-16 \*\*\*  speed 4.238e-03 9.656e-05 43.891 < 2e-16 \*\*\*  hd 3.433e-04 1.464e-05 23.451 < 2e-16 \*\*\*  ram 2.041e-02 5.628e-04 36.264 < 2e-16 \*\*\*  screen 5.327e-02 2.098e-03 25.387 < 2e-16 \*\*\*  cd 5.282e-02 4.980e-03 10.605 < 2e-16 \*\*\*  multi 4.764e-02 6.040e-03 7.887 3.87e-15 \*\*\*  premium -2.277e-01 6.425e-03 -35.436 < 2e-16 \*\*\*  ads 2.229e-04 2.690e-05 8.288 < 2e-16 \*\*\*  trend -2.399e-02 3.270e-04 -73.375 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 0.1211 on 4371 degrees of freedom  Multiple R-squared: 0.7836, Adjusted R-squared: 0.7831  F-**statistic**: 1759 on 9 and 4371 DF, p-value: < 2.2e-16 |
|  |
| |  | | --- | | > | |

#square root transformation

comp\_model4<-lm(sqrt(price)~speed+hd+ram+screen+cd+multi+premium+ads+trend,data=train\_data[-c(5961,4478),])

summary(comp\_model4)

|  |
| --- |
| Call:  lm(formula = sqrt(price) ~ speed + hd + ram + screen + cd + multi +  premium + ads + trend, data = train\_data[-c(5961, 4478),  ])  Residuals:  Min 1Q Median 3Q Max  -11.890 -1.756 -0.011 1.640 16.008  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) 27.7476654 0.7365568 37.672 < 2e-16 \*\*\*  speed 0.0982810 0.0022550 43.583 < 2e-16 \*\*\*  hd 0.0081554 0.0003419 23.856 < 2e-16 \*\*\*  ram 0.4897297 0.0131426 37.263 < 2e-16 \*\*\*  screen 1.2508236 0.0490040 25.525 < 2e-16 \*\*\*  cd 0.9766065 0.1163102 8.397 < 2e-16 \*\*\*  multi 1.1158940 0.1410543 7.911 3.21e-15 \*\*\*  premium -5.3456074 0.1500475 -35.626 < 2e-16 \*\*\*  ads 0.0055551 0.0006281 8.844 < 2e-16 \*\*\*  trend -0.5581410 0.0076363 -73.090 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 2.828 on 4371 degrees of freedom  Multiple R-squared: 0.7856, Adjusted R-squared: 0.7852  F-**statistic**: 1780 on 9 and 4371 DF, p-value: < 2.2e-16 |
|  |
| |  | | --- | | > | |

**#quadratic model**

comp\_model5<- lm((price)~speed+I(speed^2)+hd+I(hd^2)+ram+I(ram^2)+screen+I(screen^2)+cd+I(cd^2)+multi+I(multi^2)+premium+I(premium^2)+ads+I(ads^2)+trend+I(trend^2),data=train\_data[-c(5961,4478),])

summary(comp\_model5)

|  |
| --- |
| Call:  lm(formula = (price) ~ speed + I(speed^2) + hd + I(hd^2) + ram +  I(ram^2) + screen + I(screen^2) + cd + I(cd^2) + multi +  I(multi^2) + premium + I(premium^2) + ads + I(ads^2) + trend +  I(trend^2), data = train\_data[-c(5961, 4478), ])  Residuals:  Min 1Q Median 3Q Max  -1035.97 -170.63 -14.75 138.87 1859.98  Coefficients: (3 not defined because of singularities)  Estimate Std. Error t value Pr(>|t|)  (Intercept) 1.131e+04 1.081e+03 10.459 < 2e-16 \*\*\*  speed 2.114e+01 9.472e-01 22.316 < 2e-16 \*\*\*  I(speed^2) -1.008e-01 7.806e-03 -12.911 < 2e-16 \*\*\*  hd 1.575e+00 7.595e-02 20.738 < 2e-16 \*\*\*  I(hd^2) -5.764e-04 5.239e-05 -11.002 < 2e-16 \*\*\*  ram 5.099e+01 3.679e+00 13.860 < 2e-16 \*\*\*  I(ram^2) -1.611e-01 1.338e-01 -1.204 0.228512  screen -1.344e+03 1.417e+02 -9.484 < 2e-16 \*\*\*  I(screen^2) 4.724e+01 4.602e+00 10.264 < 2e-16 \*\*\*  cd 4.177e+01 1.083e+01 3.858 0.000116 \*\*\*  I(cd^2) NA NA NA NA  multi 1.028e+02 1.296e+01 7.930 2.77e-15 \*\*\*  I(multi^2) NA NA NA NA  premium -5.303e+02 1.387e+01 -38.228 < 2e-16 \*\*\*  I(premium^2) NA NA NA NA  ads -1.423e+00 4.999e-01 -2.846 0.004453 \*\*  I(ads^2) 2.699e-03 9.692e-04 2.785 0.005376 \*\*  trend -2.283e+01 5.012e+00 -4.556 5.36e-06 \*\*\*  I(trend^2) -9.868e-01 1.614e-01 -6.114 1.06e-09 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 259.4 on 4365 degrees of freedom  Multiple R-squared: 0.8046, Adjusted R-squared: 0.8039  F-statistic: 1198 on 15 and 4365 DF, p-value: < 2.2e-16 |
|  |
| |  | | --- | | > | |

#polynomial model

comp\_model6<lm((price)~speed+I(speed^2)+I(speed^3)+hd+I(hd^2)+I(hd^2)+ram+I(ram^2)+I(ram^3)+screen+I(screen^2)+I(screen^3)+cd+I(cd^2)+I(cd^3)+multi+I(multi^2)+I(multi^3)+premium+I(premium^2)+I(premium^3)+ads+I(ads^2)+I(ads^3)+trend+I(trend^2)+I(trend^3),data=train\_data[-c(5961,4478),])

summary(comp\_model6)

|  |
| --- |
| all:  lm(formula = (price) ~ speed + I(speed^2) + I(speed^3) + hd +  I(hd^2) + I(hd^2) + ram + I(ram^2) + I(ram^3) + screen +  I(screen^2) + I(screen^3) + cd + I(cd^2) + I(cd^3) + multi +  I(multi^2) + I(multi^3) + premium + I(premium^2) + I(premium^3) +  ads + I(ads^2) + I(ads^3) + trend + I(trend^2) + I(trend^3),  data = train\_data[-c(5961, 4478), ])  Residuals:  Min 1Q Median 3Q Max  -1054.95 -167.44 -17.15 137.99 1871.23  Coefficients: (7 not defined because of singularities)  Estimate Std. Error t value Pr(>|t|)  (Intercept) 1.045e+04 1.080e+03 9.674 < 2e-16 \*\*\*  speed 6.409e+01 5.681e+00 11.283 < 2e-16 \*\*\*  I(speed^2) -8.481e-01 9.772e-02 -8.679 < 2e-16 \*\*\*  I(speed^3) 3.929e-03 5.117e-04 7.679 1.97e-14 \*\*\*  hd 1.523e+00 7.639e-02 19.931 < 2e-16 \*\*\*  I(hd^2) -5.619e-04 5.235e-05 -10.733 < 2e-16 \*\*\*  ram 6.543e+01 7.171e+00 9.125 < 2e-16 \*\*\*  I(ram^2) -1.297e+00 5.352e-01 -2.422 0.015457 \*  I(ram^3) 2.612e-02 1.245e-02 2.098 0.035976 \*  screen -1.313e+03 1.407e+02 -9.330 < 2e-16 \*\*\*  I(screen^2) 4.626e+01 4.571e+00 10.119 < 2e-16 \*\*\*  I(screen^3) NA NA NA NA  cd 4.312e+01 1.084e+01 3.979 7.02e-05 \*\*\*  I(cd^2) NA NA NA NA  I(cd^3) NA NA NA NA  multi 1.024e+02 1.287e+01 7.957 2.23e-15 \*\*\*  I(multi^2) NA NA NA NA  I(multi^3) NA NA NA NA  premium -5.355e+02 1.388e+01 -38.585 < 2e-16 \*\*\*  I(premium^2) NA NA NA NA  I(premium^3) NA NA NA NA  ads -4.241e+00 1.176e+00 -3.606 0.000314 \*\*\*  I(ads^2) 1.788e-02 6.065e-03 2.949 0.003206 \*\*  I(ads^3) -2.497e-05 9.881e-06 -2.527 0.011527 \*  trend -1.964e+01 9.929e+00 -1.978 0.048043 \*  I(trend^2) -1.096e+00 5.687e-01 -1.928 0.053971 .  I(trend^3) 5.057e-04 9.684e-03 0.052 0.958357  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 257.4 on 4361 degrees of freedom  Multiple R-squared: 0.8077, Adjusted R-squared: 0.8069  F-statistic: 964.1 on 19 and 4361 DF, p-value: < 2.2e-16 |
|  |
| |  | | --- | | > | |

# here comp\_model6 has highest R-squared Adjusted R-squared values. so take comp\_model6 for prediction

#prediction of price

comp\_price\_prediction<-predict(comp\_model6,test\_data)

head(comp\_price\_prediction)

|  |
| --- |
| 4 6 9 14 16 19  2682.924 3257.339 2641.585 2244.085 2323.857 2266.520 |
|  |
| |  | | --- | | > | |

# compare the predicted value in test\_data

head(test\_data)

|  |
| --- |
| price speed hd ram screen cd multi premium ads trend  4 1849 25 170 8 14 0 0 0 94 1  6 3695 66 340 16 14 0 0 1 94 1  9 2225 50 210 8 14 0 0 1 94 1  14 2295 25 245 8 14 0 0 1 94 1  16 2225 50 130 4 14 0 0 1 94 1  19 2095 33 250 4 15 0 0 1 94 1 |
|  |
| |  | | --- | | > | |

test\_data$predicted\_price<-comp\_price\_prediction

head(test\_data)

|  |
| --- |
| price speed hd ram screen cd multi premium ads trend predicted\_price  4 1849 25 170 8 14 0 0 0 94 1 2682.924  6 3695 66 340 16 14 0 0 1 94 1 3257.339  9 2225 50 210 8 14 0 0 1 94 1 2641.585  14 2295 25 245 8 14 0 0 1 94 1 2244.085  16 2225 50 130 4 14 0 0 1 94 1 2323.857  19 2095 33 250 4 15 0 0 1 94 1 2266.520 |
|  |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | >  head(test\_data$predicted\_price)   |  | | --- | | 2682.924 3257.339 2641.585 2244.085 2323.857 2266.520 | |  | | |  | | --- | | > | |   # check the correlation of actual price and predicted price  r<-cor(test\_data$price,test\_data$predicted\_price)  0.8974  r\_squared<-cor(test\_data$price,test\_data$predicted\_price)^  0.8054  # using ggplot, grapgically represent the actual price and predicted price  install.packages("ggplot2")  library(ggplot2)  ggplot(data=test\_data,aes(x=price,color="actual price"))+  geom\_density(aes(x=price,color="actual price"))+  geom\_density(aes(x=predicted\_price,color="predicted price"))+  scale\_color\_manual(values=c("predicted price"="blue","actual price"="red"))+  labs(title="density plt between the actual price and predicted price",  x="price",y= "")  C:\Users\sathi\OneDrive\Desktop\DATASCIENCE\5 | |