**Name: - UPENDRA DAMA**

**Batch: - 25/01/2020 (Weekend)**

**Module: - 7**

**Prepare a prediction model for price of Toyota Corolla**

**Problem Statement: -** Prepare a prediction model for price of Toyota Corolla.

**EDA: -** Let’s to Exploratory Data Analysis.

|  |
| --- |
| > library(readr)  > ToyotaCorolla <- read\_csv("Desktop/Digi 360/Module 7/ToyotaCorolla.csv")  Parsed with column specification:  cols(  .default = col\_double(),  Model = col\_character(),  Fuel\_Type = col\_character(),  Color = col\_character()  )  See spec(...) for full column specifications.  > View(ToyotaCorolla)  > ToyotaCorolla <- ToyotaCorolla[,c("Price","Age\_08\_04","KM","HP","cc","Doors","Gears","Quarterly\_Tax","Weight")]  > colnames(ToyotaCorolla) <- c("price","age","km","hp","cc","door","gear","qrt","wgt")  > attach(ToyotaCorolla)  The following object is masked from Computer\_Data (pos = 3):  price  The following object is masked from Computer\_Data (pos = 4):  price  The following object is masked from Computer\_Data (pos = 6):  price  The following object is masked from Computer\_Data (pos = 7):  price  ##Let’s see EDA with quick summary  > summary(ToyotaCorolla)  price age km hp cc  Min. : 4350 Min. : 1.00 Min. : 1 Min. : 69.0 Min. : 1300  1st Qu.: 8450 1st Qu.:44.00 1st Qu.: 43000 1st Qu.: 90.0 1st Qu.: 1400  Median : 9900 Median :61.00 Median : 63390 Median :110.0 Median : 1600  Mean :10731 Mean :55.95 Mean : 68533 Mean :101.5 Mean : 1577  3rd Qu.:11950 3rd Qu.:70.00 3rd Qu.: 87021 3rd Qu.:110.0 3rd Qu.: 1600  Max. :32500 Max. :80.00 Max. :243000 Max. :192.0 Max. :16000  door gear qrt wgt  Min. :2.000 Min. :3.000 Min. : 19.00 Min. :1000  1st Qu.:3.000 1st Qu.:5.000 1st Qu.: 69.00 1st Qu.:1040  Median :4.000 Median :5.000 Median : 85.00 Median :1070  Mean :4.033 Mean :5.026 Mean : 87.12 Mean :1072  3rd Qu.:5.000 3rd Qu.:5.000 3rd Qu.: 85.00 3rd Qu.:1085  Max. :5.000 Max. :6.000 Max. :283.00 Max. :1615 |

Here, the mean is less than the median for age, hp and cc so the distribution is **left skewed**.

Whereas for remaining variables, mean is greater than the median so the distribution is **right skewed**.

**Relationship: -**

Let’s draw scatter diagram to see the relationship among output variable Price and remaining all input variables.

|  |
| --- |
| ##Draw the scatter plot for entire dataset  > > plot(ToyotaCorolla)    ##Finding correlation coefficient  > cor(ToyotaCorolla)  price age km hp cc door  price 1.00000000 -0.876590497 -0.56996016 0.31498983 0.12638920 0.18532555  age -0.87659050 1.000000000 0.50567218 -0.15662202 -0.09808374 -0.14835921  km -0.56996016 0.505672180 1.00000000 -0.33353795 0.10268289 -0.03619661  hp 0.31498983 -0.156622020 -0.33353795 1.00000000 0.03585580 0.09242450  cc 0.12638920 -0.098083739 0.10268289 0.03585580 1.00000000 0.07990330  door 0.18532555 -0.148359215 -0.03619661 0.09242450 0.07990330 1.00000000  gear 0.06310386 -0.005363947 0.01502333 0.20947715 0.01462935 -0.16014143  qrt 0.21919691 -0.198430508 0.27816470 -0.29843172 0.30699580 0.10936323  wgt 0.58119759 -0.470253184 -0.02859846 0.08961406 0.33563740 0.30261764  gear qrt wgt  price 0.063103857 0.219196911 0.58119759  age -0.005363947 -0.198430508 -0.47025318  km 0.015023328 0.278164697 -0.02859846  hp 0.209477146 -0.298431717 0.08961406  cc 0.014629352 0.306995798 0.33563740  door -0.160141430 0.109363225 0.30261764  gear 1.000000000 -0.005451955 0.02061328  qrt -0.005451955 1.000000000 0.62613373  wgt 0.020613284 0.626133733 1.00000000 |

Here we observe the correlation coefficient is very far from 1 among all the variables. So, the correlation is not good among the variables.

**Model Building: -**

Now, let’s build the model.

|  |
| --- |
| ##Start building model  > model1 <- lm(price~age+km+hp+cc+door+gear+qrt+wgt)  > summary(model1)  Call:  lm(formula = price ~ age + km + hp + cc + door + gear + qrt +  wgt)  Residuals:  Min 1Q Median 3Q Max  -9366.4 -793.3 -21.3 799.7 6444.0  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) -5.573e+03 1.411e+03 -3.949 8.24e-05 \*\*\*  age -1.217e+02 2.616e+00 -46.512 < 2e-16 \*\*\*  km -2.082e-02 1.252e-03 -16.622 < 2e-16 \*\*\*  hp 3.168e+01 2.818e+00 11.241 < 2e-16 \*\*\*  cc -1.211e-01 9.009e-02 -1.344 0.17909  door -1.617e+00 4.001e+01 -0.040 0.96777  gear 5.943e+02 1.971e+02 3.016 0.00261 \*\*  qrt 3.949e+00 1.310e+00 3.015 0.00262 \*\*  wgt 1.696e+01 1.068e+00 15.880 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 1342 on 1427 degrees of freedom  Multiple R-squared: 0.8638, Adjusted R-squared: 0.863  F-statistic: 1131 on 8 and 1427 DF, p-value: < 2.2e-16 |

So, the linear equation with the above coefficients won’t be considered a good equation because the p value is greater than 0.05 for cc and Doors.

**Model Evolution: -**

Here p value is greater than 0.05 for cc and doors. So, let’s identify and drop the variable which is colinear with other input variables. After that let’s build the model again with remaining variables to see if we get correct equation with significance p value.

|  |
| --- |
| > #Variance Inflation Factor to identify which is most colinear variable with other input variables.  > library(car)  Loading required package: carData  > vif(model1)  age km hp cc door gear qrt wgt  1.884620 1.756905 1.419422 1.163894 1.156575 1.098723 2.311431 2.516420  ###Here the VFF is less than 5. So, no variable is collinear with others. Hence, we can’t decide which variable have to be removed to get the low p value.  > #Let's remove cc and doors since they have p value greater than 0.05 that means no significance of this value for our model.  ##Building the model after dropping the cc and doors variable  > model2 <- lm(price~age+km+hp+gear+qrt+wgt)  > summary(model2)  Call:  lm(formula = price ~ age + km + hp + gear + qrt + wgt)  Residuals:  Min 1Q Median 3Q Max  -9224.7 -792.3 -25.5 801.9 6436.6  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) -5.478e+03 1.409e+03 -3.889 0.000105 \*\*\*  age -1.217e+02 2.615e+00 -46.534 < 2e-16 \*\*\*  km -2.094e-02 1.249e-03 -16.769 < 2e-16 \*\*\*  hp 3.133e+01 2.799e+00 11.191 < 2e-16 \*\*\*  gear 5.990e+02 1.934e+02 3.096 0.001997 \*\*  qrt 3.737e+00 1.296e+00 2.883 0.003999 \*\*  wgt 1.673e+01 1.020e+00 16.393 < 2e-16 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 1342 on 1429 degrees of freedom  Multiple R-squared: 0.8636, Adjusted R-squared: 0.863  F-statistic: 1508 on 6 and 1429 DF, p-value: < 2.2e-16 |

Here p value is significant so we reject the null hypothesis. That means there is significant correlation between price vs other input variables.

Here we also can see R-squared value is 0.8636 which is greater than 0.85. **Hence our model is good** and we don’t need further transformations.

Since we are ready with equation, let’s **predict** the values and accuracy after **splitting** the dataset.

|  |
| --- |
| > n=nrow(ToyotaCorolla)  > n1=n\*0.7  > n2=n-n1  > train=sample(1:n,n1)  > test=ToyotaCorolla[-train,]  > ###Let’s predict the values with model2.  > pred=predict(model2,newdata = test)  > actual=test$price  > ##Finding the errors  > error=actual-pred  > ###Finding the accuracy  > test.rmse=sqrt(mean(error\*\*2))  > test.rmse  [1] 1361.433  > train.rmse=sqrt(mean(model2$residuals\*\*2))  > train.rmse  [1] 1339.106 |

Here we can see test rmse value is very close to train rmse value. So, our model is significantly fit.

**Table of R^2 value for two models: -**

|  |  |
| --- | --- |
| Model 1 | 0.8638 |
| Model 2 | 0.8636 |

**Let’s do the same in Python**

|  |
| --- |
| **Python Code** |
| ####################################################  ########Predict price of Toyota Corolla#############  ####################################################  import pandas as pd  import numpy as np  tc = pd.read\_csv ("~/desktop/Digi 360/Module 7/ToyotaCorolla.csv",encoding='mac\_roman')  tc.head(1)  ###Select required columns  tc1 = tc[['Price','Age\_08\_04','KM','HP','cc','Doors','Gears','Quarterly\_Tax','Weight']]  tc1.head()   |  | **Price** | **Age\_08\_04** | **KM** | **HP** | **cc** | **Doors** | **Gears** | **Quarterly\_Tax** | **Weight** | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 0 | 13500 | 23 | 46986 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 1 | 13750 | 23 | 72937 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 2 | 13950 | 24 | 41711 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 3 | 14950 | 26 | 48000 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 4 | 13750 | 30 | 38500 | 90 | 2000 | 3 | 5 | 210 | 1170 |   ###Renaming columns  tc1 = tc1.rename(columns = {'Price':'price','Age\_08\_04':'age','KM':'km','HP':'hp','Doors':'door','Gears':'gear','Quarterly\_Tax':'qrt','Weight':'wgt'})  tc1.head()   |  | **price** | **age** | **km** | **hp** | **cc** | **door** | **gear** | **qrt** | **wgt** | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 0 | 13500 | 23 | 46986 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 1 | 13750 | 23 | 72937 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 2 | 13950 | 24 | 41711 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 3 | 14950 | 26 | 48000 | 90 | 2000 | 3 | 5 | 210 | 1165 | | 4 | 13750 | 30 | 38500 | 90 | 2000 | 3 | 5 | 210 | 1170 |   ##Let's see EDA by summary  tc1.describe()   |  | **price** | **age** | **km** | **hp** | **cc** | **door** | **gear** | **qrt** | **wgt** | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | count | 1436.000000 | 1436.000000 | 1436.000000 | 1436.000000 | 1436.00000 | 1436.000000 | 1436.000000 | 1436.000000 | 1436.00000 | | mean | 10730.824513 | 55.947075 | 68533.259749 | 101.502089 | 1576.85585 | 4.033426 | 5.026462 | 87.122563 | 1072.45961 | | std | 3626.964585 | 18.599988 | 37506.448872 | 14.981080 | 424.38677 | 0.952677 | 0.188510 | 41.128611 | 52.64112 | | min | 4350.000000 | 1.000000 | 1.000000 | 69.000000 | 1300.00000 | 2.000000 | 3.000000 | 19.000000 | 1000.00000 | | 25% | 8450.000000 | 44.000000 | 43000.000000 | 90.000000 | 1400.00000 | 3.000000 | 5.000000 | 69.000000 | 1040.00000 | | 50% | 9900.000000 | 61.000000 | 63389.500000 | 110.000000 | 1600.00000 | 4.000000 | 5.000000 | 85.000000 | 1070.00000 | | 75% | 11950.000000 | 70.000000 | 87020.750000 | 110.000000 | 1600.00000 | 5.000000 | 5.000000 | 85.000000 | 1085.00000 | | max | 32500.000000 | 80.000000 | 243000.000000 | 192.000000 | 16000.00000 | 5.000000 | 6.000000 | 283.000000 | 1615.00000 |   ###let's draw a pirplot among all the input variables vs output variable.  import seaborn as sns  sns.pairplot(tc1.iloc[:,:])  /var/folders/kv/w79zffc14fd2hj518gqdhnmc0000gn/T/com.microsoft.Word/Content.MSO/839828F3.tmp  ##finding the correlation coefficient  tc1.corr()   |  | **price** | **age** | **km** | **hp** | **cc** | **door** | **gear** | **qrt** | **wgt** | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | price | 1.000000 | -0.876590 | -0.569960 | 0.314990 | 0.126389 | 0.185326 | 0.063104 | 0.219197 | 0.581198 | | age | -0.876590 | 1.000000 | 0.505672 | -0.156622 | -0.098084 | -0.148359 | -0.005364 | -0.198431 | -0.470253 | | km | -0.569960 | 0.505672 | 1.000000 | -0.333538 | 0.102683 | -0.036197 | 0.015023 | 0.278165 | -0.028598 | | hp | 0.314990 | -0.156622 | -0.333538 | 1.000000 | 0.035856 | 0.092424 | 0.209477 | -0.298432 | 0.089614 | | cc | 0.126389 | -0.098084 | 0.102683 | 0.035856 | 1.000000 | 0.079903 | 0.014629 | 0.306996 | 0.335637 | | door | 0.185326 | -0.148359 | -0.036197 | 0.092424 | 0.079903 | 1.000000 | -0.160141 | 0.109363 | 0.302618 | | gear | 0.063104 | -0.005364 | 0.015023 | 0.209477 | 0.014629 | -0.160141 | 1.000000 | -0.005452 | 0.020613 | | qrt | 0.219197 | -0.198431 | 0.278165 | -0.298432 | 0.306996 | 0.109363 | -0.005452 | 1.000000 | 0.626134 | | wgt | 0.581198 | -0.470253 | -0.028598 | 0.089614 | 0.335637 | 0.302618 | 0.020613 | 0.626134 | 1.000000 |   ###Preparing model with all variables  import statsmodels.formula.api as smf  model1 = smf.ols('price~age+km+hp+cc+door+gear+qrt+wgt',data=tc1).fit()  model1.summary()   |  |  |  |  | | --- | --- | --- | --- | | OLS Regression Results | | | | | **Dep. Variable:** | price | **R-squared:** | 0.864 | | **Model:** | OLS | **Adj. R-squared:** | 0.863 | | **Method:** | Least Squares | **F-statistic:** | 1131. | | **Date:** | Fri, 13 Mar 2020 | **Prob (F-statistic):** | 0.00 | | **Time:** | 10:48:52 | **Log-Likelihood:** | -12376. | | **No. Observations:** | 1436 | **AIC:** | 2.477e+04 | | **Df Residuals:** | 1427 | **BIC:** | 2.482e+04 | | **Df Model:** | 8 |  |  | | **Covariance Type:** | nonrobust |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | **coef** | **std err** | **t** | **P>|t|** | **[0.025** | **0.975]** | | **Intercept** | -5573.1064 | 1411.390 | -3.949 | 0.000 | -8341.728 | -2804.485 | | **age** | -121.6584 | 2.616 | -46.512 | 0.000 | -126.789 | -116.527 | | **km** | -0.0208 | 0.001 | -16.622 | 0.000 | -0.023 | -0.018 | | **hp** | 31.6809 | 2.818 | 11.241 | 0.000 | 26.152 | 37.209 | | **cc** | -0.1211 | 0.090 | -1.344 | 0.179 | -0.298 | 0.056 | | **door** | -1.6166 | 40.006 | -0.040 | 0.968 | -80.093 | 76.859 | | **gear** | 594.3199 | 197.055 | 3.016 | 0.003 | 207.771 | 980.869 | | **qrt** | 3.9491 | 1.310 | 3.015 | 0.003 | 1.379 | 6.519 | | **wgt** | 16.9586 | 1.068 | 15.880 | 0.000 | 14.864 | 19.054 |  |  |  |  |  | | --- | --- | --- | --- | | **Omnibus:** | 151.719 | **Durbin-Watson:** | 1.543 | | **Prob(Omnibus):** | 0.000 | **Jarque-Bera (JB):** | 1011.853 | | **Skew:** | -0.219 | **Prob(JB):** | 1.90e-220 | | **Kurtosis:** | 7.089 | **Cond. No.** | 3.13e+06 |   Warnings: [1] Standard Errors assume that the covariance matrix of the errors is correctly specified. [2] The condition number is large, 3.13e+06. This might indicate that there are strong multicollinearity or other numerical problems.  ###calculating VIF values for age  rsq\_ag = smf.ols('age~km+hp+cc+door+gear+qrt+wgt',data=tc1).fit().rsquared  vif\_ag= 1/(1-rsq\_ag)  print(vif\_ag)  1.8846198056602865  ###calculating VIF values for hp  rsq\_hp = smf.ols('hp~km+age+cc+door+gear+qrt+wgt',data=tc1).fit().rsquared  vif\_hp= 1/(1-rsq\_hp)  print(vif\_hp)  1.4194221086310979  ###calculating VIF values for km  rsq\_km = smf.ols('km~age+hp+cc+door+gear+qrt+wgt',data=tc1).fit().rsquared  vif\_km= 1/(1-rsq\_km)  print(vif\_km)  1.7569047782042881  ###calculating VIF values for age  rsq\_cc = smf.ols('cc~km+hp+age+door+gear+qrt+wgt',data=tc1).fit().rsquared  vif\_cc= 1/(1-rsq\_cc)  print(vif\_cc)  1.1638939849423795  ###calculating VIF values for door  rsq\_dr = smf.ols('door~km+hp+cc+age+gear+qrt+wgt',data=tc1).fit().rsquared  vif\_dr= 1/(1-rsq\_dr)  print(vif\_dr)  1.1565752070760438  ###calculating VIF values for gear  rsq\_gr = smf.ols('gear~km+hp+cc+door+age+qrt+wgt',data=tc1).fit().rsquared  vif\_gr= 1/(1-rsq\_gr)  print(vif\_gr)  1.0987230193470365  ###calculating VIF values for quarterly tax  rsq\_qr = smf.ols('qrt~km+hp+cc+door+gear+age+wgt',data=tc1).fit().rsquared  vif\_qr= 1/(1-rsq\_qr)  print(vif\_qr)  2.311430811531038  ###calculating VIF values for weight  rsq\_wt = smf.ols('wgt~km+hp+cc+door+gear+age+qrt',data=tc1).fit().rsquared  vif\_wt= 1/(1-rsq\_wt)  print(vif\_wt)  2.516419837445868  ##storing VIF values in a Dataframe  df1 = {'variables' :['age','hp','km','cc','door','gear','qrt','wgt'],'VIF' :[vif\_ag,vif\_hp,vif\_km,vif\_cc,vif\_dr,vif\_gr,vif\_qr,vif\_wt]}  vif\_df = pd.DataFrame(df1)  vif\_df   |  | **variables** | **VIF** | | --- | --- | --- | | 0 | age | 1.884620 | | 1 | hp | 1.419422 | | 2 | km | 1.756905 | | 3 | cc | 1.163894 | | 4 | door | 1.156575 | | 5 | gear | 1.098723 | | 6 | qrt | 2.311431 | | 7 | wgt | 2.516420 |   ###As cc and door are having p value high which is greater than 0.05 in model 1, we are going to drop these and build model with remaining variables  model2 = smf.ols('price~age+km+hp+gear+qrt+wgt',data=tc1).fit()  model2.summary()   |  |  |  |  | | --- | --- | --- | --- | | OLS Regression Results | | | | | **Dep. Variable:** | price | **R-squared:** | 0.864 | | **Model:** | OLS | **Adj. R-squared:** | 0.863 | | **Method:** | Least Squares | **F-statistic:** | 1508. | | **Date:** | Fri, 13 Mar 2020 | **Prob (F-statistic):** | 0.00 | | **Time:** | 11:01:48 | **Log-Likelihood:** | -12376. | | **No. Observations:** | 1436 | **AIC:** | 2.477e+04 | | **Df Residuals:** | 1429 | **BIC:** | 2.480e+04 | | **Df Model:** | 6 |  |  | | **Covariance Type:** | nonrobust |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | **coef** | **std err** | **t** | **P>|t|** | **[0.025** | **0.975]** | | **Intercept** | -5478.4778 | 1408.562 | -3.889 | 0.000 | -8241.548 | -2715.407 | | **age** | -121.6999 | 2.615 | -46.534 | 0.000 | -126.830 | -116.570 | | **km** | -0.0209 | 0.001 | -16.769 | 0.000 | -0.023 | -0.018 | | **hp** | 31.3269 | 2.799 | 11.191 | 0.000 | 25.836 | 36.818 | | **gear** | 598.9653 | 193.441 | 3.096 | 0.002 | 219.507 | 978.424 | | **qrt** | 3.7371 | 1.296 | 2.883 | 0.004 | 1.194 | 6.280 | | **wgt** | 16.7251 | 1.020 | 16.393 | 0.000 | 14.724 | 18.726 |  |  |  |  |  | | --- | --- | --- | --- | | **Omnibus:** | 147.966 | **Durbin-Watson:** | 1.540 | | **Prob(Omnibus):** | 0.000 | **Jarque-Bera (JB):** | 966.959 | | **Skew:** | -0.207 | **Prob(JB):** | 1.07e-210 | | **Kurtosis:** | 6.999 | **Cond. No.** | 3.12e+06 |   Warnings: [1] Standard Errors assume that the covariance matrix of the errors is correctly specified. [2] The condition number is large, 3.12e+06. This might indicate that there are strong multicollinearity or other numerical problems.  ##Splitting the data into train and test  from sklearn.model\_selection import train\_test\_split  tc\_train,tc\_test = train\_test\_split(tc1,test\_size=0.3) ##30% of test data  tc\_train.head()  ##Preparing the model on train data  model\_train = smf.ols('price~age+km+hp+gear+qrt+wgt', data=tc\_train).fit()  ###Train data prediction  train\_pred = model\_train.predict(tc\_train)  ###Finding train Risedual values  train\_resid = train\_pred - tc\_train.price  ###rmse value for train data  train\_rmse = np.sqrt (np.mean(train\_resid \* train\_resid))  train\_rmse  1333.667288915473  ###Prediction on test data  test\_pred = model\_train.predict(tc\_test)  ###Finding train Risedual values  test\_resid = test\_pred - tc\_test.price  ###rmse value for train data  test\_rmse = np.sqrt (np.mean(test\_resid \* test\_resid))  test\_rmse  1360.4137961072356 |