Components Influence Car's Price

Rabina Padhy
dept. School of Computing, Engineering and Mathematics
Western Sydney University
Victoria Rd, Rydalmere, NSW, 2116
19884657@student.westernsydney.edu.au

***Abstract*— This paper discusses about the feature impacting the car price in Australia using different data modeling and analyzing technique. As well, investigate what are the important components impacting prominently and how we explore those features in our dataset. After analyzing the data, try to explore the significant impact of it on each of the car models.**

Keywords—Car's Price, EngineSize and CurbWeight

I. INTRODUCTION

Australians are more adventures and journey oriented people. Here maximum people prefer car over motorbike because of its comfort, safety, and ability to carry more luggage. In Australia when it comes to auto mobile industry then the vexed question is that "why people are paying too much price for a car in Australia". There are many reasons behind it one the peculiar reason is there are so many components using to manufacture a car that's impacting their price. So, let's figure out following features impact car price or not.

II. DATA EXPLORE AND DATA PROCESSING

A. *Data Explore*

In this dataset we have total 203 observation and 16 variables. In which Price will be target variable and rest are independent variables. There are 12 quantitative (such as WheelBase, Length, Width, Height, CurbWeight, EngineSize, Bore, Stroke, CompressionRatio, HorsePower, PeakRPM, MPG)and 3 qualitative (such as Make, FuelType, Cylinder) (kolandasamy, 2019).

B. *Data Cleaning*

Before processing the data for the regression analysis, we are changing Make, FuelType and Cylinder from factor variable to numeric variable as MakeCode, FuelTypeCode and Cylinder respectively. We need to remove all missing values, remove all NAs and remove all negative values. So that we can accurately measures the pricing details of car.

C. *Decision Tree*

Before selecting a particular independent variable, we need to visualize prediction by decision tree method and it helps to visualize the nature of partitioning carried out by a Regression Tree. We need to split our data set into training and testing dataset. So that we can draw model by training dataset and then verify that model by testing the data set.

By observing the tree model, we conclude that Prices are below 8 or above 20 thousands, when EngineSize is less than 182 and CurbWeight is less than 2557.5. When Prices are higher viz. above 40 or below 30 thousand, then MPG is less than 20. So, we conclude that those cars's having low MPG (miles per gallon) their Prices are high. Thus, demonstrating negative linear relationship of MPG with Price.

From the summary of regression tree on training dataset we come to know there are 6 nodes terminals and highly significant variables to make decision for car's Price viz. EngineSize, CurbWeight, HorsePower, MakeCode and MPG. The residual mean deviance is 8.89 which is smaller value. Below is a demonstration of identifying the mean square error on this training dataset by using cross validation method.

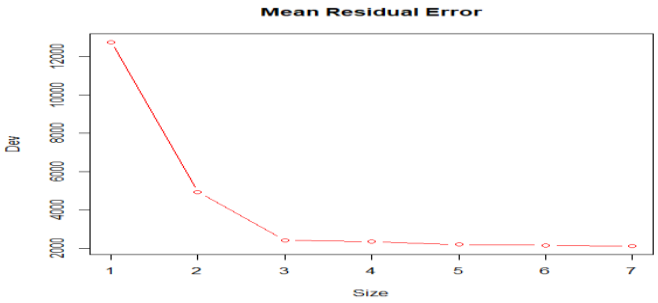


FIGURE:1 RESIDUAL ERROR ON TRAINING DATASET

From Figure 1, we can summarize that when size is 1 then deviation is more but it drastically change when size is 2 and 3. It slightly decreases when size is 4 and 5. There is no change in deviation when size is 6 or 7. So the best size of the tree model is 3 or 4 from figure:1.

Below is the illustration of using pruning technique to find the best nodes and their tree structure, in Figure:2. From this pruning method we get means square error as 9.76 and standard deviation as 3.12. We get 4 terminal nodes and when the EngineSize is less than 182, CurbWeight is less than 2557.5 then, Price will be minimum.

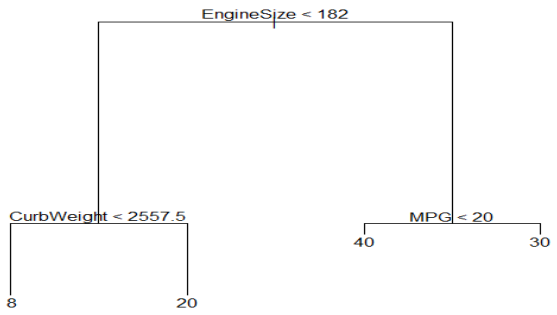


FIGURE:2 BEST REGRESSION TREE STRUCTURE

D. Principle Component Analysis

Principle Component Analysis technique is useful for exploratory data analysis and to visualize the variation present in the dataset. It helps to create new variable which will be the combination of original variables and has maximum variance.

From Figure:3 we can perceive that there is 45^0 angle variance between CurbWeight and PeakRPM. The highest variance is CurbWeight where as the lowest variance is EngineSize.

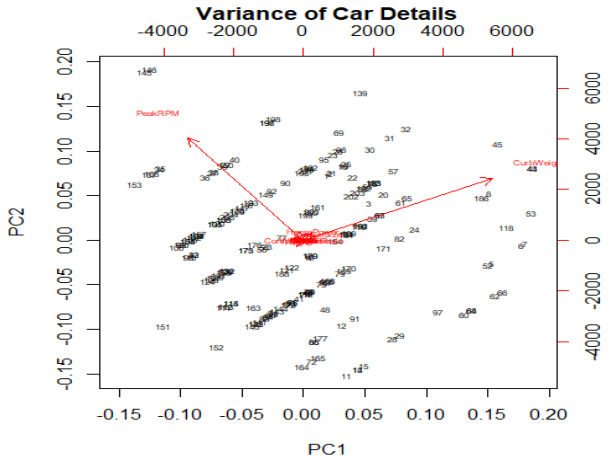


FIGURE:3 VARIANCE OF CAR DETAILS DATA SET

From the summary of PCA, we note that cumulative proportion of PC1 and PC2 is 0.998 which is quite close to 1. Using this process, we can get 2 independent variables with Price, whereas if we consider PC1, PC2, PC3, PC4, PC5, PC6, PC7, PC8, PC9 then we are getting 1 as cumulative proportion. Thus, the structure will be more complex and less expressive.

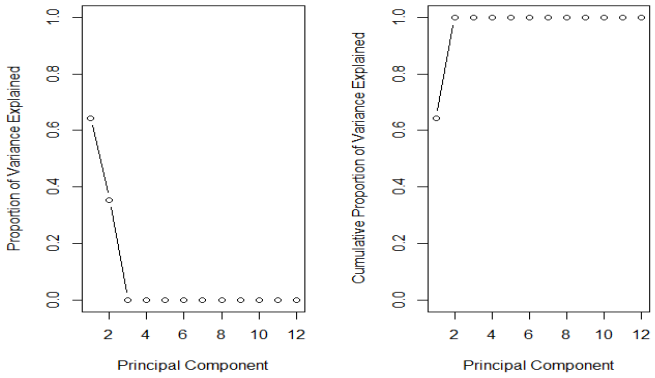


FIGURE:4 PRINCIPLE COMPONENTS OF CAR DETAILS DATA SET

From Figure 4, we confirmed that PC1 and PC2 provide maximum variance than other principle components. The first principal component explains about 64% of the variation in the data, the next principal component explains about 35% of the variation in the dataset.

III. LINEAR REGRESSION

Referring to Figure 5, we derive that from graph 2 (Length Vs Price), graph 5 (CurbWeight Vs Price), graph 6(EngineSize Vs Price), graph 10(HorsePower Vs Price) and graph 12(MPG Vs Price) shows strong linear relationship. The strength of linear relation is high then others. But some variables doesn't have any relationship with Price such as graph 4 (Height Vs Price), graph 8 (Stroke Vs Price), graph 9 (CompressionRatio Vs Price), graph 11 (PeakRPM Vs Price) as we can see if we increase or decrease independent variable there is no much impact in Price value.

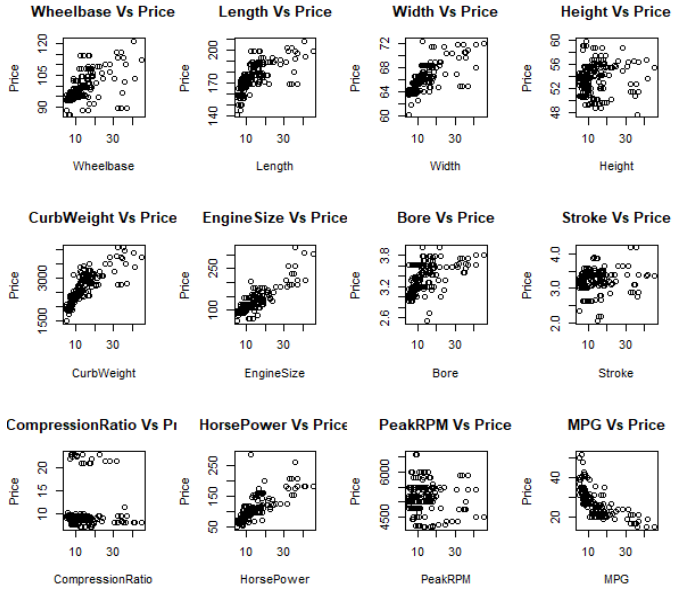


FIGURE:5 PRICE VERSE OTHER INDEPENDENT VARIABLES RELATIONSHIP

A. *Correlation Coefficient*

The correlation coefficient r measures the strength and direction of a linear relationship between two variables on a scatter plot. The value of r is always between +1 and –1. With Price the highest correlation coefficient is 0.865 from EngineSize followed by CurbWeight 0.829. Third and fourth position hold by HorsePower and MPG respectively.

B. *Single Independent Variable*

We are considering EngineSize with Price to figure out the linearity. From the summary of LinearModel, we observe that coefficient of determination (R^2) is 76.6%, p-value is < 2e-16 which is extremely small, residual is 3.7 and '***' shows that significance is high which indicates the strong linear relation between EngineSize and Price.

The Linear model equation will be

$$\mathrm{E}(\hat{Prices}) = \hat{\alpha} + \hat{\beta}\text{EngineSize} \quad (1)$$

The least square estimates of the linear model are

Estimate of Intercept = $\hat{\alpha}$ = -6.76

Estimate of the Slope of EngineSize (β) = 0.156

$$\mathrm{E}(\hat{Price}) = -6.76 + 0.156\text{EngineSize}$$

C. *Hypothesis Testing for Linear Model*

H_0:$\beta_0 = 0$

If the slope parameter is not significantly different from 0, then there are no relationship between Price and EngineSize. As here β is more than 0 so we can reject null hypothesis.

Vs

H_1:$\beta_1 \neq 0$

If The slope parameter is significantly different from 0, then there is a relationship between Price and EngineSize.

The p-value is 2.2e-16 which is less than 0.05. We have strong evidence to reject the null hypothesis at 5% level of significance and support the alternative hypothesis H_1: $\beta_1 \neq 0$. Therefore, there is strong evidence to support that there is a significant linear relationship between Price and EngineSize.

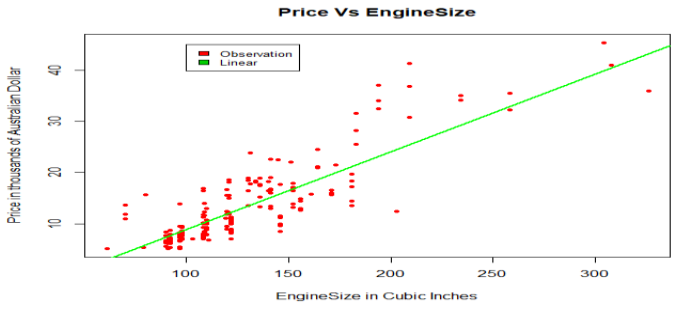


FIGURE:6 LINEAR MODEL WITH ENGINESIZE AND PRICE

In Figure 6, red dots are represented as observation and green line shows the linearity of EngineSize with Price.

IV. NON-LINEAR REGRESSION

A. *Polynomial Regression*

We have just one continuous explanatory variable, CurbWeight. In addition to show CurbWeight in a curvature format with Price, we modified CurbWeight to their poly orders viz. $(\text{CurbWeight})^2$, $(\text{CurbWeight})^3$.

Model equation for polynomial is

$$\mathrm{E}(\hat{Prices}) = \hat{\alpha} + \hat{\beta}_1\text{CurbWeight} + \hat{\beta}_2(CurbWeight)^2 + \hat{\beta}_3(CurbWeight)^3 \quad (2)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = 12.96

Estimate of slope of CurbWeight (β_1) = 91.72

Estimate of slope $(\text{CurbWeight})^2$ (β_2) =20.64

Estimate of slope $(\text{CurbWeight})^3$ (β_3) =5.18

Then the equation 2 will be

$$\mathrm{E}(\hat{Prices}) = 12.96 + 91.72\text{CurbWeight} + 20.64(CurbWeight)^2 + 5.18(CurbWeight)^3$$

From the summary of Polynomial model, we can conclude that coefficient of determination (R^2) is 78.86%. For order 1, p-value is < 2e-16, for order 2 and 3 p-value will be $1.66e^{-06}$ and 0.118 respectively. The smaller values of p-value and residual of 3.57, indicates the significant polynomial relation between CurbWeight and Price. Poly orders considerably increases when we move from 3 to 1. The degree of freedom varies from 3 to 146.

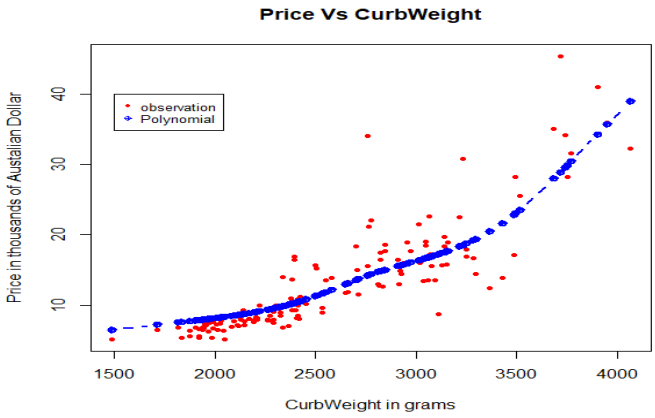


FIGURE:7 POLYNOMIAL MODEL FOR PRICE VS CURBWEIGHT

Conclusively, Polynomial model is efficient with R^2 with a factor of 0.05% with residual value lesser than 0.06, compared to Linear Model.

B. *Interactions Modeling*

We must take two continuous explanatory variables such as Width and CurbWeight, but we need to fit the interaction of these two variables with Price.

The Interaction Model equation will be

$$\hat{E}(Prices) = \hat{\alpha} + \hat{\beta}_1 \text{Width} + \hat{\beta}_2 \text{CurbWeight} + \hat{\beta}_3 \text{CurbWeight*Width} \quad (3)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = 1.628e+02

Estimate of slope of Width (β_1)= -2.625e+00

Estimate of slope of CurbWeight (β_2)=-7.364e-02

Estimate of slope of CurbWeight*Width (β_3) =1.248e-03

Then the equation 3 will be

$$\hat{E}(Prices) = 1.628e+02 \ -2.625e+00\text{Width} \ -7.364e-02\text{CurbWeight} \ +1.248e-03\text{CurbWeight*Width}$$

From the summary of Interaction model, we can conclude that coefficient of determination (R^2) is 82.04%, p-value is fixed at Width value of 8.25e-05, CurbWeight value of 1.59e-07 and Width*CurbWeight value of 6.54e-09. This is considering a residual standard error of 3.29 indicating the significant interaction relation between Width, CurbWeight and Price. Additionally we get intercepts and slope in which negative slope indicates that, an increase in price will result in decrease of independent variable such as Width and CurbWeight.

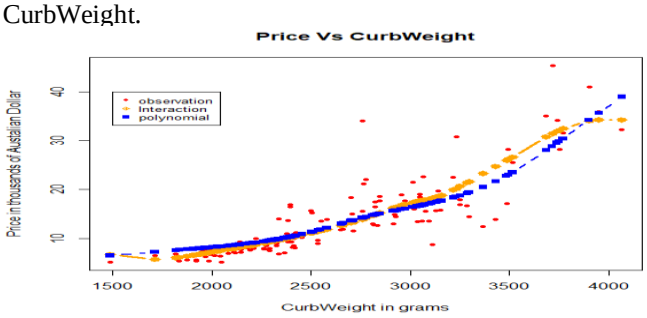


FIGURE:8 INTERACTION MODEL WITH PRICE

In Figure8, red dot represent observation, orange line indicates interaction whereas blue line indicates polynomial. Interaction model provides slightly better and more accurate result than other two.

C. *Multiple Linear Model*

We are considering 4 independent variables such as EngineSize, CurbWeight, Width and MPG, to check the relationship with Price.

The Multiple Model equation will be

$$\hat{E}(Prices) = \hat{\alpha} + \hat{\beta}_1 \text{EngineSize} + \hat{\beta}_2 \text{CurbWeight} + \hat{\beta}_3 \text{Width} + \hat{\beta}_4 \text{MPG} \quad (4)$$

The Least square estimate of Multi linear model is

Estimate of Intercept = $\hat{\alpha}$ = -60.26

Estimate of slope of EngineSize (β_1) = -0.091

Estimate of slope of CurbWeight (β_2) = 0.002

Estimate of slope of Width (β_3) =0.881

Estimate of slope of MPG (β_4) =-0.045

Then the equation 4 will be

$$\hat{E}(Prices) = -60.26 \ -0.091\text{EngineSize} + \ 0.002\text{CurbWeight} \ +0.881\text{Width} \ -0.045\text{MPG}$$

From the summary of multiple regression model we can conclude that coefficient of determination (R^2) is 84.46%. P-value of EngineSize is 4.08e-13, CurbWeight is 0.10 and Width is 0.0002. These P-values are extremely small which indicates it is 0.05 significant. But for MPG, P-value don't have any significance with Price where residual standard error is 3.07. This illustrates significant multiple relationship exists between EngineSize, CurbWeight, Width, MPG and Price.

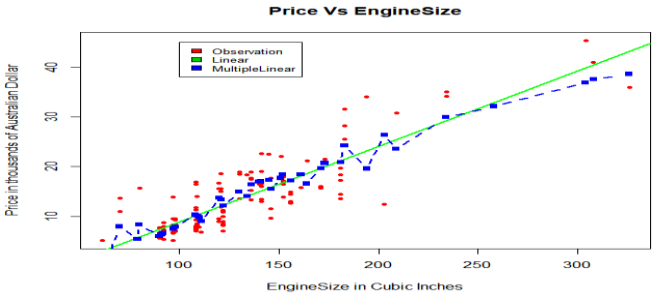


FIGURE:9 COMPARISON OF MULTIPLE LINEAR MODEL WITH OTHER MODELS

In Figure 9, blue line provides slightly better and more accurate results than green line. Red dots are records in green and blue line, indicating Linear and multi linear models. It exemplifies impact of other parameters will reduce the error and increase the coefficient of determination (R^2).

D. *Hybrid Model*

We are taking those independent variables which has high significance with Price, to derive an efficient model. In this case, we are considering factors EngineSize, CurbWeight, Width, CurbWeight*Width and order 2 of CurbWeight with Price.

The hybrid mmodel equation will be

$$\hat{E}(Prices) = \hat{\alpha} + \hat{\beta}_1 \text{EngineSize} + \hat{\beta}_2 \text{Width} + \hat{\beta}_3 \text{CurbWeight} + \hat{\beta}_4 \text{I(CurbWeight} * \text{CurbWeight)} + \hat{\beta}_5 (\text{CurbWeight} * \text{Width}) \quad (5)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = 2.364e+02

Estimate of slope of EngineSize (β_1) = 7.590e-02

Estimate of slope of Width (β_2) = -4.200e+00

Estimate of slope of CurbWeight (β_3) =-8.783e-02

Estimate of slope of CurbWeight*CurbWeight (β_4) = -4.623e-06

Estimate of slope of CurbWeight*Width (β_5) =1.769e-03

Then the equation 5 will be

$$\hat{E(Prices)} = 2.364e+02 + 7.590e\text{-}02EngineSize - 4.200e+00Width - 8.783e\text{-}02CurbWeight - 4.623e\text{-}06I(CurbWeight*CurbWeight) + 1.769e\text{-}03(CurbWeight*Width)$$

From the summary of the hybrid model we notice that the coefficient of determination (R^2) is 86.28% and residual standard error is 3.21. P-value corresponding to the coefficient parameter of EngineSize is 2.27e-09, Width is 0.0015, CurbWeight is 5.95e-05, Width * CurbWeight interaction is 0.01 and $(CurbWeight)^2$ is 0.0001. If we relate all p-values then all the findings contradict the null hypothesis(H_0). It demonstrates that hybrid model is highly significant with Width, CurbWeight, EngineSize, CurbWeight*Width, $(CurbWeight)^2$ and Price. This ssupports non-linear relationship with Price.

V. Selection Model

My basis of model selection is shouldered upon high coefficient of determination value, smaller p-value and minimum residual error. Let's compare the entire models and their respective values.

From the table 1, we can be clearly identifying the parameters for a competent model are R^2, p-value, residual standard error, F-statistic and degree of freedom. From the table we notice that hybrid model is more suitable for this dataset because, it has high coefficient of determination (R^2) than other models (86.28%), smaller p-value (<2.2e-16) and minimal residual standard error of 2.9. Although the structure looks more complex but provides better result.

TABLE I. Comparing different models and their parameters

Different models	Parameters				
	Correlation coefficient (R^2) in percent	*P-value*	*residual error*	*F-statistic*	*Degree of freedom*
Linear Model	*78.81*	*< 2e-16*	*3.55*	*550.4*	*1 o 148*
Polynomial Model	*78.86*	*< 2.2e-16*	*3.57*	*181.5*	*3 to 146*
Interaction Model	*82.04*	*< 2.2e-16*	*3.29*	*222.3*	*3 to 146*
Multilinear Model	*84.46*	*< 2.2e-16*	*3.07*	*197*	*4 to 145*
Hybrid Model	*86.28*	*< 2.2e-16*	*2.9*	*181.1*	*5 to 144*

VI. Testing and Validating the SelectedModel

A. *Overall accuracy of the model*

The overall accuracy can be measure from ANOVA (Analysis of Variance) model. ANOVA helps you test differences between mean value of two or more group. ANOVA test is centred around different sources of variation (variation between and within group) in a typical variable. Primarily ANOVA test provides evidence of the existence of the equality of mean value amongst the groups.

The following provides results from the ANOVA model.

1. P-value of 2.2e-16 is extremely small.
2. Estimated Mean Square error (MSE) is 8.4
3. Coefficient of determination(R^2) is 0.8628

A variation of 86.28% in Price which is explained by regression. The model supports moderate non-linear relationship.

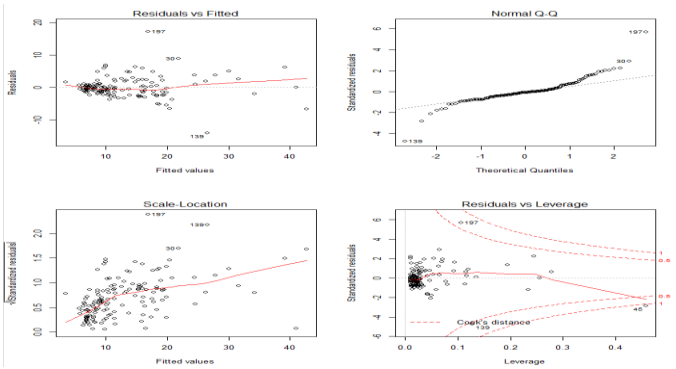


FIGURE : 10 RESIDUALS AND FITTED VALUES

The residual is the difference between the observed value of the target variable (Price) and the predicted value (E(Price)). There are a few common residual plots.

Graph 1 represents residuals vs. fitted plot. This plot tests the assumptions of whether the relationship between variables is linear (i.e. linearity) or there are equal variance along the regression line (Medium, 2019). The predicted line (red line) is very much near to zero line. Values are not much scattered but didn't follow any particular pattern. The plot is okay and relatively shapeless. Graph 1 shows a non-linear pattern and hence the linearity assumption is violated.

Graph 2 represents that the theoretical Quantiles Vs Standardized residuals which is known as Normal Q-Q plot. This plot tests that our dependent variable is normally distributed by plotting quantiles from distribution against a theoretical distribution. If it is normally distributed, then it should be plotted in a generally straight line on the Q-Q plot (Medium, 2019). The plot looks pretty good as it is almost linear to the dotted line but slightly curved in the beginning and the end. It can be concluded from the Graph 2 that the standardized residuals deviate from the normal distribution since the data points do not lie on the straight line.

Graph 3 represents that plot between fitted value and standardized residuals which is known as scale-location. This plot shows whether the residuals are spread equally along the

predictor range. We want the line on this plot to be horizontal with randomly spread points on the plot (Medium, 2019). We have noticed that our predictor range is somewhat heteroscedastic as the red line started with a steep gradient until 20 and from 30 to 45. In the intermediate it's slightly horizontal. Graph 3 shows that the residual variance is not constant

Graph 4 represents the plot between Leverage and standardized residuals. The Residuals vs. Leverage plots helps us to identify influential data points in our model. Outliers can be influential, though they don't necessarily have to be included. Few points within a normal range in the model could be very influential (Medium, 2019). The points we're looking or not looking for are values in the upper right or lower right corners, which are outside the red dashed Cook's distance line. These are points that would be influential in the model and removing them would likely noticeably alter the regression results. Our model shows influential points such as 197 with in upper right corner and 139 and 45 in the lower right corner. If we remove those cases then the model will be provide better result.

B. Verification using K-fold cross validation with TestData

To verify the dataset we need to perform cross validations on the testing data sets. For testing datasets, we use k-fold cross-validation, whose samples are randomly partitioned into k sets i.e. known as k-fold which almost equally size. Applying the first sample subset to out selected model, we could figure out the prediction error of a fitted model. Same operation will be repeated for each of the fold. The model performance is calculated by averaging error across different folds (Venturini, 2019). The k value can be 5, 10 etc and by averaging we could derive the mean square error.

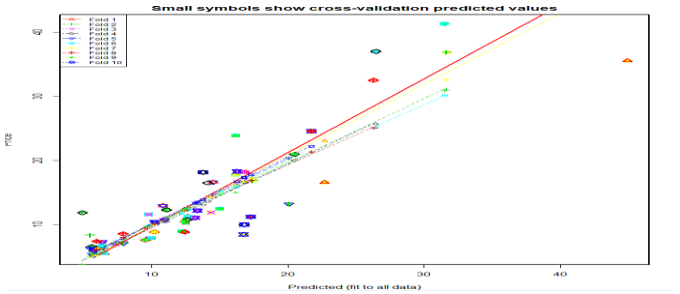


FIGURE:11 K-FOLD CROSS VALIDATION

From figure 11, one can easily note that upper part is slightly scattered but lower parts are all together. The lowest mean square error is 3.06 in fold-3 and highest mean square error is 32.2 in fold-7. So the overall mean square error is 20.8 which is considerably minor.

VII. LOGISTIC REGRESSION

To determine the classification method we change the target variable from numeric to factor variable. The average price is $10,295 in Australia. Anything higher than $10,295 is a HighPrice. Others are considered as low priced. We have added new columns in the dataset as HighPrice, whose values are either Yes or No.

A. Linear Model

To build the Linear logistic model we will consider EngineSize as continuous variable and HighPrice as Target variable.

The Equation of Linear logistic Model will be

$$\text{logit}(\text{E}(\hat{Prices})) = \hat{\alpha} + \hat{\beta}\text{EngineSize} \quad (6)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = -9.21

Estimate of slope of EngineSize ($\hat{\beta}$) = 0.06

Then the equation 6 will be become

$$logit(\text{E}(\hat{Price})) = -9.21 + 0.06\text{EngineSize}$$

From the summary of Linear logistic model, we can observe that p-value is 3.4e-12 which is very small and highly significant. The misclassification rate on train dataset is 17%.

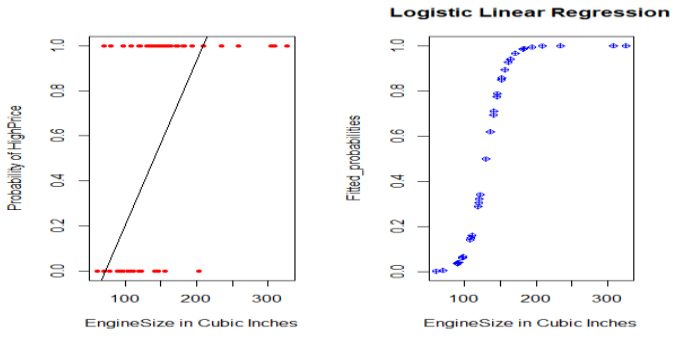


FIGURE:12 LOGISTIC LINEAR REGRESSION

B. Polynomial Model

To build non linear logistic polynomial regression, we are considering one continuous variable as EngineSize and HighPrice as Target variable.

The Equation of polynomial logistic Model is

$$\text{logit}(\text{E}(\hat{Prices})) = \hat{\alpha} + \hat{\beta}_1\text{EngineSize} + \hat{\beta}_2(EngineSize)^2 + \hat{\beta}_3(EngineSize)^3 \quad (7)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = -0.526

Estimate of slope of EngineSize ($\hat{\beta}_1$) = 24.538

Estimate of slope of (EngineSize)^2 ($\hat{\beta}_2$) = -7.07

Estimate of slope of (EngineSize) ^{23} ($\hat{\beta}_3$) = -4.46

Then the equation 7 will be become

$$logit(\text{E}(\hat{Price})) = -0.526 + 24.538\text{EngineSize} - 7.07(EngineSize)^2 - 4.46(EngineSize)^3$$

From the summary of polynomial logistic model, we can observe that the p-value corresponding to the coefficient parameter of EngineSize is 0.027, $(\text{EngineSize})^2$ as 0.41 and $(\text{EngineSize})^3$ as 0.44. These P-values are considerable small and highly significant. The misclassification rate on train dataset is 18%.

C. *Interaction Model*

To build non linear logistic interaction regression, we are considering two continuous variable as CurbWeight and Width with HighPrice as Target variable.

The Equation of interaction logistic Model is

$$\text{logit}(\text{E}(\hat{Prices})) = \hat{\alpha} + \hat{\beta}_1\text{Width} + \hat{\beta}_2\text{CurbWeight} + \hat{\beta}_3\text{CurbWeight*Width} \quad (8)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = 1.19e+02

Estimate of slope of Width (β_1) = -2.07e+00

Estimate of slope of CurbWeight (β_2) = -5.30e-02

Estimate of slope of CurbWeight*Width (β_3) = 9.03e-04

Then the equation 8 will be become

$$logit(\text{E}(\hat{Price})) = 1.19\text{e+}02 \ -2.07\text{e+}00\text{Width} \ -5.30\text{e-}02\text{CurbWeight} \ +9.03\text{e-}04\text{CurbWeight*Width}$$

From the summary of interaction logistic model, we can observe that the corresponding p-value are high which specifies that there is not enough evidence to reject null hypothesis H_0.The misclassification rate on train dataset is 18%.

D. *MultipleLinear Model*

To build non linear logistic multiple linear regression, we are considering four continuous variable as EngineSize, MPG, CurbWeight and Width with HighPrice as Target variable.

The Equation of multiple linear logistic Model is

$$\text{logit}(\text{E}(\hat{Prices})) = \hat{\alpha} + \hat{\beta}_1\text{EngineSize} + \hat{\beta}_2\text{CurbWeight} + \hat{\beta}_3\text{Width} + \hat{\beta}_4\text{MPG} \quad (9)$$

The Least square estimate of Polynomial model is

Estimate of Intercept = $\hat{\alpha}$ = -24.79

Estimate of slope of EngineSize (β_1) = -0.008

Estimate of slope of CurbWeight (β_2) = 0.004

Estimate of slope of Width (β_3) = -0.374

Estimate of slope of MPG (β_4) = -0.286

Then the equation 9 will be become

$$logit(\text{E}(\hat{Price})) = -24.79 \ -0.008\text{EngineSize} + \ 0.004\text{Curb-Weight} \ -0.374\text{Width} \ -0.286\text{MPG}$$

From the summary of multilinear logistic model, we can observe that the corresponding p-value are high as compare to other models. Thus, it doesn't have strong evidence to reject null hypothesis H_0, but to contradict the p-value of MPG is 0.0024. The misclassification rate on train dataset is 8% which is comparatively lower than other models.

E. *Hybrid Model*

To build non linear logistic hybrid linear regression, we are considering two continuous variable as EngineSize and MPG with HighPrice as Target variable.

The Equation of hybrid linear logistic Model is

$$\text{logit}(\text{E}(\hat{Prices})) = \hat{\alpha} + \hat{\beta}_1\text{EngineSize} + \hat{\beta}_2\text{MPG} \quad (10)$$

From summary, we received the α and β_1, β_2 values and then the equation will be

$$logit(\text{E}(\hat{Price})) = 5.902 + 0.03\text{EngineSize} \ -0.29\text{MPG}$$

From the summary of hybrid linear logistic model, we can observe that the corresponding p-values are low for both EngineSize and MPG i.e. 0.022 and 0.0002 respectively. As per 5% significant we can have strong evidence to reject the null hypothesis H_0.The misclassification rate on train dataset is 8%.

VIII. Selection Model for Logistic Regression

My basis of model selection is shouldered upon few parameters as smaller p-value and minimum misclassification rate. Let's compare the entire models and their respective values.

TABLE II. COMPARING DIFFERENT CLASSIFICATION MODELS AND THEIR PARAMETERS

	Parameters	
Different models	***p-value***	***misclassification rate(percentage)***
Linear Model	***6.8e-07***	***17***
Polynomial Model	***0.027***	***18***
Interaction Model	***0.39***	***18***
Multilinear Model	***0.0024***	***8***
Hybrid Model	***0.0002***	***8***

As per Table: II, we concluded that hybrid logistic regression model is our selection model as the p-value is 0.0002 which provides 5% strong evidence to reject null hypothesis and the residual error is 8%.

IX. Testing and Validating the SelectionModel

A. *Overall accuracy of the model*

To test or validate the selected model we use ANOVA method. This method derives the deviance of residual and degree of freedom in a group.

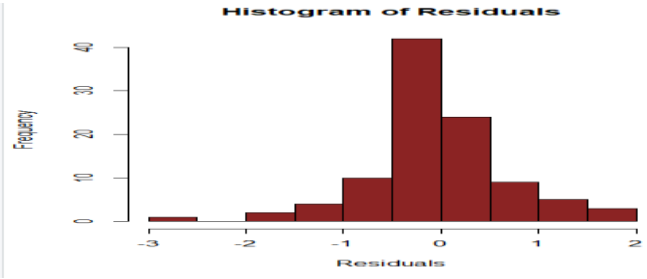


FIGURE:13 HISTOGRAM OF RESIDUALS

From Figure 13, represents the histogram plot of residual Vs frequency. It also depicted that residuals not followed normal distributions. If we observe clearly then mode is close to zero. However, the residuals follow a negatively skewed as well as positive skewed.

B. *Testing and Validating by test dataset*

There are 100 observations in the testing dataset. In a table we consider first predicted HighPrice Vs the actual HighPrice. From which we can calculate the misclassification rate is o.45. From SVM technique, we easily find out the best model. When using polynomial kernel function, the best parameters are cost = 1 and degree = 3 since the error is minimum at this value. The optimal model thus fitted when using a polynomial kernel function has 150 support vectors.

X. Verify Selection Model using different car's Model

In our dataset we have Toyota, Nissan, Mitsubishi as different brands of car with different manufacturer. From our dataset we extract that subset as per different car's brand. So applying selected model to each subset we can compare the results . In Figure 14, it depicted that the predicted Price for various continuous variable as EngineSize, CurbWeight and Width.

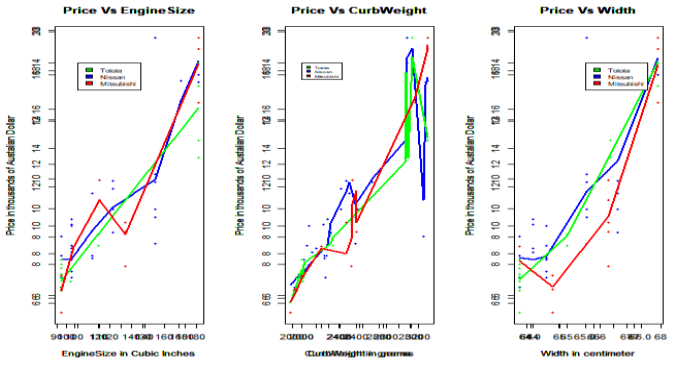


FIGURE : 14 DIFFERENT CAR MODELS FIT INTO SELECTION MODEL

From summary, we can clearly seen that for Toyota car model(green colour) the coefficient of determination (R^2) is 87.19%, overall p-value is 8.378e-11 which is very small and residual error is 1.253 which is minimum. Similarly, for Nissan model(blue colour) the coefficient of determination (R^2) is 99.04%, overall p-value is 1.126e-11 which is very small, so we reject null hypothesis and residual error is 0.521 which is very minimum or negligible. For Mitsubishi model(red colour) the coefficient of determination (R^2) is 96.93%, overall p-value is 3.795e-05 which is very small, so we reject null hypothesis and residual error is 0.698 which is very minimum or negligible.

XI. Conclusion

We conclude that car Price is depending on few features such as EngineSize, Width, CurbWeight, MPG. From regression test we observe that not only individual features but also their poly order of 2 like (CurbWeight)2 and interaction of variables like (CurbWeight*Width) plays vital role in their Price. After comparing various car models we conclude that model Nissan is highly suitable. The rationality of that is because its R2 is 99.04%, residual standard error is 0.521 and the p-value 1.126e-11. Those being considerably small, we can reject the null hypothesis(H_0).

Acknowledgment

Thanks Dr. Liwan Liyanage gives me constructive comments and warm encouragement. Without her guidance and persistent help this paper dissertation would not have been possible.

References

[1] kolandasamy, D. (2019). Deepikakolandasamy/Car-insurance-prediction. [online] GitHub. Available at: https://github.com/Deepikakolandasamy/Car-insurance-prediction [Accessed 30 Sep. 2019].

[2] Montefiore.ulg.ac.be. (2019). [online] Available at: http://www.montefiore.ulg.ac.be/~kvansteen/GBIO0009-1/ac20092010/Class8/Using%20R%20for%20linear%20regression.pdf [Accessed 30 Sep. 2019].

[3] Medium. (2019). Residual Plots Part 1 — Residuals vs. Fitted Plot. [online] Available at: https://medium.com/data-distilled/residual-plots-part-1-residuals-vs-fitted-plot-f069849616b1 [Accessed 30 Sep. 2019].

[4] Medium. (2019). Residual Plots Part 3— Scale-Location Plot. [online] Available at: https://medium.com/data-distilled/residual-plots-part-3-scale-location-plot-113e469b99c [Accessed 30 Sep. 2019].

[5] Medium. (2019). Residual Plots Part 4— Residuals vs. Leverage Plot. [online] Available at: https://medium.com/data-distilled/residual-plots-part-4-residuals-vs-leverage-plot-14aeed009ef7 [Accessed 30 Sep. 2019].

[6] Venturini, S. (2019). Cross-Validation for Predictive Analytics Using R - MilanoR. [online] MilanoR. Available at: http://www.milanor.net/blog/cross-validation-for-predictive-analytics-using-r/ [Accessed 30 Sep. 2019].

[7] DataCamp Community. (2019). Logistic Regression in R Tutorial. [online] Available at: https://www.datacamp.com/community/tutorials/logistic-regression-R [Accessed 1 Oct. 2019].

XII. Appendix

packaging

```
library(readr)
library(lattice)
library(tibble)
library(DAAG)
```

```
## [1] 205  16
```

CleanProcess

```
carDetails <- subset(data1, data1$HorsePowe
dim(carDetails)
```

```
## [1] 203  16
```

Classification

```
sapply(carDetails, class)
```

```
##            Make        FuelType       WheelBase          Length
##        "factor"        "factor"       "numeric"       "numeric"
##           Width          Height       CurbWeight        Cylinder
##       "numeric"       "numeric"       "integer"        "factor"
##      EngineSize            Bore          Stroke CompressionRatio
##       "integer"       "numeric"       "numeric"       "numeric"
##      HorsePower         PeakRPM             MPG           Price
##       "integer"       "integer"       "numeric"       "numeric"
```

```
dim(carDetails)
```

```
## [1] 203  16
```

Converting FuelType to FuelTypeCode

```
FuelTypeCode = carDetails$FuelType
levels(FuelTypeCode)
```

```
## [1] "diesel" "gas"
```

```
levels(FuelTypeCode)[levels(FuelTypeCode)=="diesel"]=0
levels(FuelTypeCode)[levels(FuelTypeCode)=="gas"]=1
levels(FuelTypeCode)
```

```
## [1] "0" "1"
```

```
FuelTypeCode = as.numeric(levels(FuelTypeCode))[FuelTypeCode]
class(FuelTypeCode)
```

```
## [1] "numeric"
```

```
table(FuelTypeCode)
```

```
## FuelTypeCode
##   0   1
##  20 183
```

COnverting Make to MakeCode

```
MakeCode = carDetails$Make
levels(MakeCode)
```

```
##  [1] "alfa"         "audi"         "bmw"          "chevrolet"
##  [5] "dodge"        "honda"        "isuzu"        "jaguar"
##  [9] "mazda"        "mercedes-benz" "mercury"     "mitsubishi"
## [13] "nissan"       "peugot"       "plymouth"     "porsche"
## [17] "renault"      "saab"         "subaru"       "toyota"
## [21] "volkswagen"   "volvo"
```

```
levels(MakeCode)[levels(MakeCode)=="alfa"]=1
levels(MakeCode)[levels(MakeCode)=="audi"]=2
levels(MakeCode)[levels(MakeCode)=="bmw"]=3
levels(MakeCode)[levels(MakeCode)=="chevrolet"]=4
levels(MakeCode)[levels(MakeCode)=="dodge"]=5
levels(MakeCode)[levels(MakeCode)=="honda"]=6
levels(MakeCode)[levels(MakeCode)=="isuzu"]=7
levels(MakeCode)[levels(MakeCode)=="jaguar"]=8
levels(MakeCode)[levels(MakeCode)=="mazda"]=9
levels(MakeCode)[levels(MakeCode)=="mercedes-benz"]=10
levels(MakeCode)[levels(MakeCode)=="mercury"]=11
levels(MakeCode)[levels(MakeCode)=="mitsubishi"]=12
levels(MakeCode)[levels(MakeCode)=="nissan"]=13
levels(MakeCode)[levels(MakeCode)=="peugot"]=14
levels(MakeCode)[levels(MakeCode)=="plymouth"]=15
levels(MakeCode)[levels(MakeCode)=="porsche"]=16
levels(MakeCode)[levels(MakeCode)=="renault"]=17
levels(MakeCode)[levels(MakeCode)=="saab"]=18
levels(MakeCode)[levels(MakeCode)=="subaru"]=19
levels(MakeCode)[levels(MakeCode)=="toyota"]=20
levels(MakeCode)[levels(MakeCode)=="volkswagen"]=21
levels(MakeCode)[levels(MakeCode)=="volvo"]=22

levels(MakeCode)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22"
```

```
table(MakeCode)
```

```
## MakeCode
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
##  3  7  8  3  9 13  4  3 17  8  1 13 18 11  7  5  0  6 12 32 12 11
```

```
MakeCode = as.numeric(MakeCode)
class(MakeCode)
```

```
## [1] "numeric"
```

Reformat Dataset

```
carDetails <- add_column(carDetails, MakeCode,FuelTypeCode, .after = 1)
carDetails = carDetails[,-1]
carDetails = carDetails[,-3]
head(carDetails)
```

```
##   MakeCode FuelTypeCode WheelBase Length Width Height CurbWeight Cylinder
## 1       22            1     104.3  188.8  67.2   56.2       2912     four
## 2       22            1     104.3  188.8  67.2   56.2       2935     four
## 3       22            1     104.3  188.8  67.2   56.2       3045     four
## 4        5            1     103.3  174.6  64.6   59.8       2535     four
## 5       10            0     110.0  190.9  70.3   56.5       3515     five
## 6       10            0     110.0  190.9  70.3   58.7       3750     five
##   EngineSize Bore Stroke CompressionRatio HorsePower PeakRPM  MPG  Price
## 1        141 3.78   3.15              9.5        114    5400 37.0 12.950
## 2        141 3.78   3.15              9.5        114    5400 38.0 15.985
## 3        130 3.62   3.15              7.5        162    5100 28.0 18.420
## 4        122 3.34   3.46              8.5         88    5000 39.0  8.921
## 5        183 3.58   3.64             21.5        123    4350 34.5 25.552
## 6        183 3.58   3.64             21.5        123    4350 34.5 28.248
```

```
set.seed(9001)
train<-sample(1:nrow(carDetails),150)

head(train)
```

```
## [1]  78 197  66   9 192  53
```

```
CarDetrailstrain=carDetails[train, ]
dim(CarDetrailstrain)
```

```
## [1] 150  16
```

Linear Model

```
LinearModel = lm(Price~EngineSize, data = carDetails, subset = train)
summary(LinearModel)
```

```
##
## Call:
## lm(formula = Price ~ EngineSize, data = carDetails, subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.1055 -2.0664 -0.5858  1.2440 10.5242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.769363   0.894291   -7.57 3.71e-12 ***
## EngineSize   0.156058   0.006652   23.46  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.555 on 148 degrees of freedom
## Multiple R-squared:  0.7881, Adjusted R-squared:  0.7867
## F-statistic: 550.4 on 1 and 148 DF,  p-value: < 2.2e-16
```

```
confint(LinearModel)
```

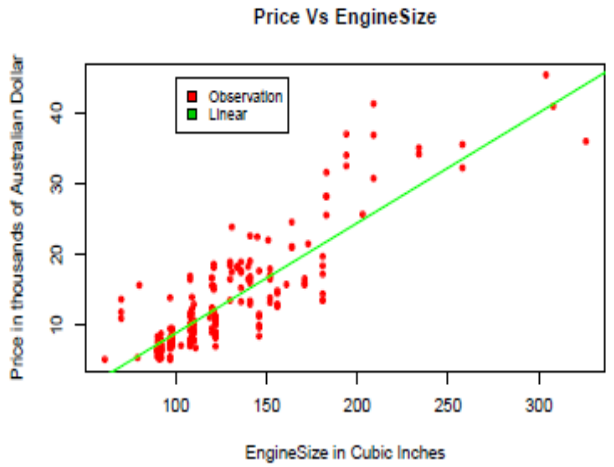
```
##                 2.5 %     97.5 %
## (Intercept) -8.5365917 -5.0021348
## EngineSize   0.1429126  0.1692029
```

Linear Model Anova

```
anova(LinearModel)
```

```
## Analysis of Variance Table
##
## Response: Price
##             Df Sum Sq Mean Sq F value    Pr(>F)
## EngineSize   1 6956.6  6956.6  550.39 < 2.2e-16 ***
## Residuals  148 1870.7    12.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

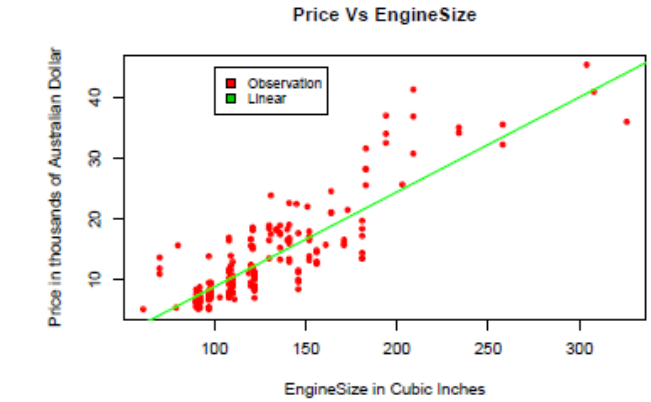
```
plot(Price~EngineSize, data = carDetails, xlab="EngineSize in Cubic Inches",ylab = "Pr
abline(a= -6.76, b=0.156, col = "green", lwd = 2)
legend(100, 45, legend=c("Observation", "Linear"),
       col=c("red", "green"), c( 2, 3),cex=0.8)
```



PolyNomial Model

```
polyModel1=lm(Price~poly(CurbWeight,3), data = carDetails, subset = train)
summary(polyModel1)
```

```
##
## Call:
## lm(formula = Price ~ poly(CurbWeight, 3), data = carDetails,
##     subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.2819  -1.2685  -0.3126   0.6554  19.9443
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            12.9691     0.2922  44.379  < 2e-16 ***
## poly(CurbWeight, 3)1   91.7246     4.0473  22.663  < 2e-16 ***
## poly(CurbWeight, 3)2   20.6438     4.1334   4.994 1.66e-06 ***
## poly(CurbWeight, 3)3    5.1854     4.0999   1.265    0.208
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.575 on 146 degrees of freedom
## Multiple R-squared:  0.7886, Adjusted R-squared:  0.7842
## F-statistic: 181.5 on 3 and 146 DF,  p-value: < 2.2e-16
```



PolyNomial Model

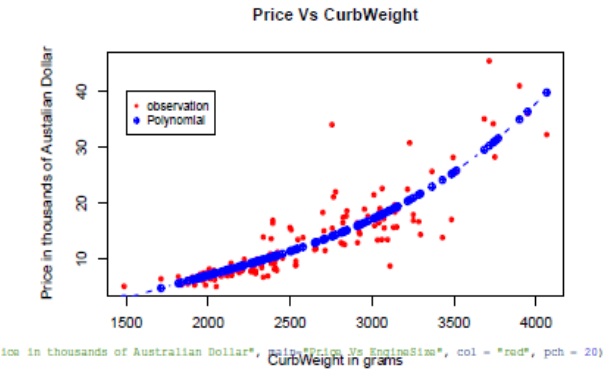
```
polyModel1=lm(Price~poly(CurbWeight,3), data = carDetails, subset = train)
summary(polyModel1)
```

```
##
## Call:
## lm(formula = Price ~ poly(CurbWeight, 3), data = carDetails,
##     subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.2819  -1.2685  -0.3126   0.6554  19.9443
##
## Coefficients:
```

```
anova(polyModel1)
```

```
## Analysis of Variance Table
##
## Response: Price
##                     Df Sum Sq Mean Sq F value    Pr(>F)
## poly(CurbWeight, 3)  3 6960.8 2320.27   181.5 < 2.2e-16 ***
## Residuals          146 1866.5   12.78
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(Price~CurbWeight, data = CarDetrailstrain, xlab="CurbWeight in grams",ylab = "Price in thousa
lines(smooth.spline(CarDetrailstrain$CurbWeight, predict(polyModel1)), col= "blue",lwd=2, lty=2,pc
legend(1500,40, legend=c("observation","Polynomial"), col=c("red","blue"), pch = c(20, 10),cex=0.8
```



Interaction Model

```
InteractionModel = lm(Price~Width+CurbWeight+Width*CurbWeight, data = carDetails, subset = train)
summary(InteractionModel)
```

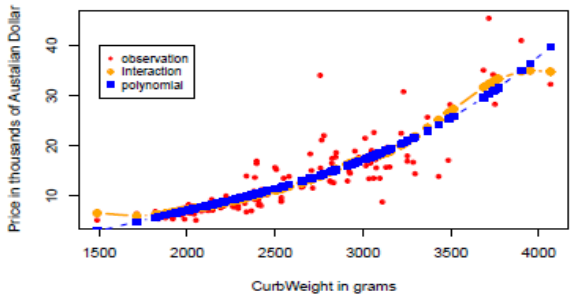
```
##
## Call:
## lm(formula = Price ~ Width + CurbWeight + Width * CurbWeight,
##     data = carDetails, subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5498 -1.3506 -0.4230  0.4523 21.2910
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.628e+02  4.166e+01   3.906 0.000143 ***
## Width             -2.625e+00  6.480e-01  -4.051 8.25e-05 ***
## CurbWeight        -7.364e-02  1.337e-02  -5.508 1.59e-07 ***
## Width:CurbWeight   1.248e-03  2.024e-04   6.165 6.54e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.295 on 146 degrees of freedom
## Multiple R-squared:  0.8204, Adjusted R-squared:  0.8167
## F-statistic: 222.3 on 3 and 146 DF,  p-value: < 2.2e-16
```

```
anova(InteractionModel)
```

```
## Analysis of Variance Table
##
## Response: Price
##                Df Sum Sq Mean Sq F value    Pr(>F)
## Width           1 6085.1  6085.1 560.405 < 2.2e-16 ***
## CurbWeight      1  744.1   744.1  68.530 7.244e-14 ***
## Width:CurbWeight  1  412.7   412.7  38.007 6.541e-09 ***
## Residuals     146 1585.3    10.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(Price~CurbWeight, data = CarDetrailstrain,xlab="CurbWeight in grams",ylab = "Price in thousanc
lines(smooth.spline(CarDetrailstrain$CurbWeight,predict(InteractionModel)), col= "orange",lwd=3, pc
lines(smooth.spline(CarDetrailstrain$CurbWeight, predict(polyModel1)), col= "blue",lwd=2, lty=2,pch
legend(1500,40, legend=c("observation","Interaction", "polynomial"), col=c("red","orange", "blue"),
```

Price Vs CurbWeight



MultiLinear Model Summary

```
MultiLinearModel = lm(Price~EngineSize+CurbWeight+Width+MPG , data = carDetails, subset
summary(MultiLinearModel)
```

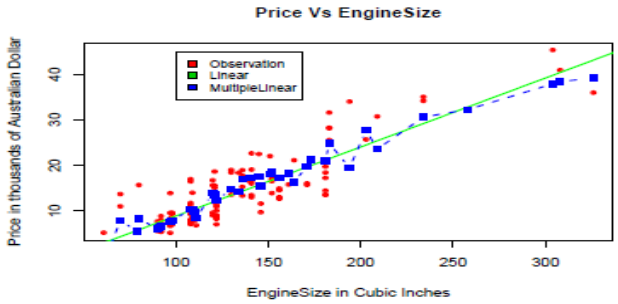
```
##
## Call:
## lm(formula = Price ~ EngineSize + CurbWeight + Width + MPG, data = carDetails,
##     subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.7601 -1.9529  0.0039  1.1397 14.7113
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -60.260481  13.546287  -4.448 1.71e-05 ***
## EngineSize    0.091055   0.011412   7.979 4.08e-13 ***
## CurbWeight    0.002169   0.001323   1.639 0.103292
## Width         0.881143   0.233450   3.774 0.000233 ***
## MPG          -0.045363   0.039939  -1.136 0.257911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.076 on 145 degrees of freedom
## Multiple R-squared:  0.8446, Adjusted R-squared:  0.8403
## F-statistic:   197 on 4 and 145 DF,  p-value: < 2.2e-16
```

```
anova(MultiLinearModel)
```

```
## Analysis of Variance Table
##
## Response: Price
##              Df Sum Sq Mean Sq  F value    Pr(>F)
## EngineSize    1 6956.6  6956.6 735.3548 < 2.2e-16 ***
## CurbWeight    1  355.5   355.5  37.5805 7.884e-09 ***
## Width         1  131.2   131.2  13.8683   0.00028 ***
## MPG           1   12.2    12.2   1.2901   0.25791
## Residuals   145 1371.7     9.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(Price~EngineSize, data = CarDetrailstrain, xlab="EngineSize in Cubic Inches",ylab =
abline(a= -6.385, b=0.152, col = "green", lwd = 2)
lines(smooth.spline(CarDetrailstrain$EngineSize, predict(MultiLinearModel)), col= "blue",
legend(100, 45, legend=c("Observation", "Linear", "MultipleLinear"),
       col=c("red", "green", "blue"), c( 2, 3, 4),cex=0.8)
```

Price Vs EngineSize



Hybrid Model

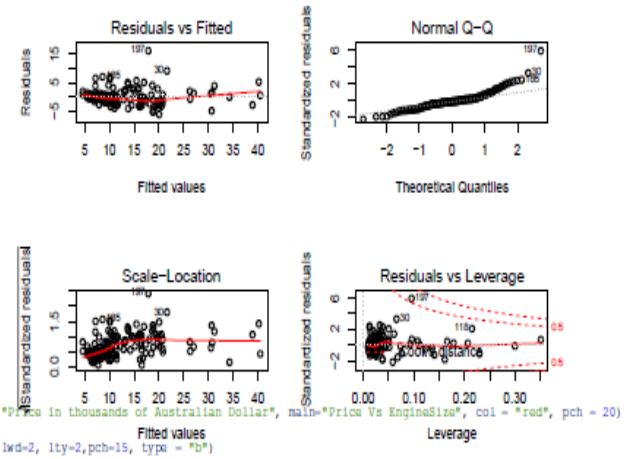
```
selectModel = lm(Price~EngineSize+Width+CurbWeight+Width*CurbWeight+I(CurbWeight*CurbWeight) , data
summary(selectModel)
```

```
##
## Call:
## lm(formula = Price ~ EngineSize + Width + CurbWeight + Width *
##     CurbWeight + I(CurbWeight * CurbWeight), data = carDetails,
##     subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1868 -1.3618 -0.2529  0.6973 16.1788
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   2.364e+02  7.417e+01   3.187 0.001764 **
## EngineSize                    7.590e-02  1.190e-02   6.380 2.27e-09 ***
## Width                        -4.200e+00  1.305e+00  -3.218 0.001594 **
## CurbWeight                   -8.783e-02  2.123e-02  -4.137 5.95e-05 ***
## I(CurbWeight * CurbWeight)   -4.623e-06  1.821e-06  -2.538 0.012200 *
## Width:CurbWeight              1.769e-03  4.513e-04   3.920 0.000136 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.9 on 144 degrees of freedom
## Multiple R-squared:  0.8628, Adjusted R-squared:  0.858
## F-statistic: 181.1 on 5 and 144 DF,  p-value: < 2.2e-16
```

```
anova(selectModel)
```

```
## Analysis of Variance Table
##
## Response: Price
##                               Df Sum Sq Mean Sq  F value    Pr(>F)
## EngineSize                     1 6956.6  6956.6 827.1442 < 2.2e-16 ***
## Width                          1  434.4   434.4  51.6479 3.305e-11 ***
## CurbWeight                     1   52.3    52.3   6.2230 0.0137403 *
## I(CurbWeight * CurbWeight)     1   43.6    43.6   5.1829 0.0242837 *
## Width:CurbWeight               1  129.2   129.2  15.3675 0.0001364 ***
## Residuals                    144 1211.1     8.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
par(mfrow=c(2,2))
plot(selectModel)
```




```
= "Price in thousands of Australian Dollar", main="Price Vs EngineSize", col = "red", pch = 20)
s",lwd=2, lty=2,pch=15, type = "b")
```

Crossvaidation

```
dim(carDetails[-train,])
```

```
## [1] 53 16
```

```
carDetailstest <- carDetails[-train,]
carDetailstest <- rbind(carDetailstest, carDetails[-train,])
carDetailstest <- rbind(carDetailstest, carDetails[-train,])
```

```
dim(carDetailstest)
```

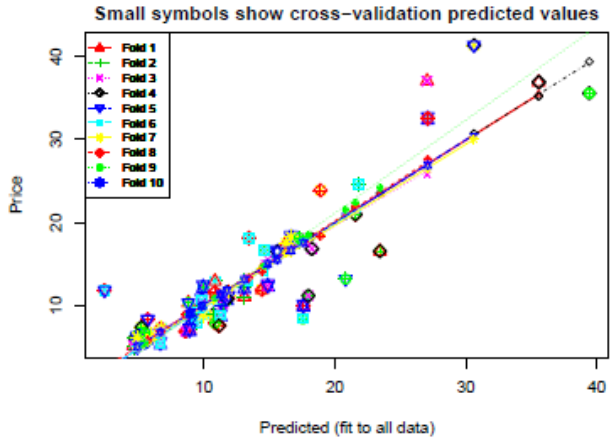
```
## [1] 159  16
```

```
cv.lm(data = carDetailstest, selectModel, m= 10)
```

```
## Analysis of Variance Table
##
## Response: Price
##                              Df Sum Sq Mean Sq F value Pr(>F)
## EngineSize                    1   9257    9257  588.12 <2e-16 ***
## Width                         1     84      84    5.37 0.0218 *
## CurbWeight                    1     61      61    3.86 0.0514 .
## I(CurbWeight * CurbWeight)    1      3       3    0.22 0.6422
## Width:CurbWeight              1    174     174   11.04 0.0011 **
## Residuals                   153   2408      16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Warning in cv.lm(data = carDetailstest, selectModel, m = 10):
##
```

```
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 15
##                 15    26    32    38   109   111   154   168   261  1251
## Predicted   11.895 16.29 35.56  9.97  8.83 4.638 13.11 17.61 16.29  8.93
## cvpred      11.980 16.47 35.43 10.03  8.93 4.676 13.23 17.55 16.47  8.90
## Price       11.248 17.50 36.88  8.85 10.35 5.200 11.05  9.99 17.50  7.10
## CV residual -0.732  1.03  1.45 -1.18  1.42 0.524 -2.18 -7.56  1.03 -1.80
##                1541 1981  2021   402   592
## Predicted     13.11 27.0 23.45 10.86 14.45
## cvpred        13.23 26.8 23.41 10.96 14.68
## Price         11.05 37.0 16.56 12.95 11.90
## CV residual -2.18 10.2 -6.85  1.99 -2.78
##
## Sum of squares = 242    Mean square = 16.1    n = 15
##
## fold 2
## Observations in test set: 16
##                  36   40    60    69    85    96    97   331   971 1111
## Predicted     9.47 10.9 20.83 17.29 10.86 18.91 21.79 6.047 21.79 4.64
## cvpred        9.49 10.9 20.82 17.30 10.89 18.39 21.91 5.995 21.91 4.63
## Price         7.90 12.9 13.20 18.15  7.90 23.88 24.57 6.580 24.57 5.20
## CV residual -1.59  2.0 -7.62  0.85 -2.99  5.49  2.66 0.585  2.66 0.57
##                1601   2041   114   472 1122  1542
## Predicted    13.10 10.852 10.74 11.50 4.74 13.11
## cvpred       13.18 10.872 10.74 11.51 4.74 13.32
## Price        12.17 11.600  8.95 10.25 6.10 11.05
## CV residual -1.01  0.728 -1.79 -1.26 1.36 -2.27
##
## Sum of squares = 132    Mean square = 8.28    n = 16
##
##
## fold 3
## Observations in test set: 16
##                 92  198   121   211   361   471   611   671   691  1161
## Predicted     9.95 27.0  9.826 18.25  9.47 11.50 14.90 14.6 17.292 11.29
## cvpred       10.17 25.8  9.962 18.35  9.44 11.43 15.56 15.1 17.604 11.21
## Price        12.29 37.0 10.698 16.84  7.90 10.25 12.44 16.6 18.150  8.85
## CV residual  2.12 11.3  0.736 -1.51 -1.54 -1.18 -3.12  1.5  0.546 -2.36
##                1171  1241  1691   802  1112 1982
## Predicted    11.288  6.70 17.98  8.63 4.638 27.0
## cvpred       11.209  6.79 17.49  8.70 4.889 25.8
## Price        10.600  5.50 11.20  6.92 5.200 37.0
## CV residual -0.609 -1.29 -6.29 -1.78 0.311 11.3
##
## Sum of squares = 328    Mean square = 20.5    n = 16
##
##
## fold 4
## Observations in test set: 16
##                  11    21    44   311   321   481 1121  1801   212    282
## Predicted    10.74 18.25 39.4 30.6 35.56 11.78 4.74 15.61 18.25 21.582
## cvpred       10.92 18.30 39.3 30.7 35.20 11.94 4.78 15.69 18.30 21.689
## Price         8.95 16.84 35.5 41.3 36.88 10.80 6.10 16.50 16.84 20.970
## CV residual -1.97 -1.45 -3.8 10.6  1.68 -1.14 1.32  0.81 -1.45 -0.719
##                322   482 1152  1652 1692  2022
## Predicted    35.56 11.78 5.25 11.18 18.0 23.45
## cvpred       35.20 11.94 5.29 11.17 18.1 23.52
## Price        36.88 10.80 7.40  7.60 11.2 16.56
## CV residual  1.68 -1.14 2.11 -3.57 -6.9 -6.96
##
## Sum of squares = 260    Mean square = 16.2    n = 16
##
## fold 5
## Observations in test set: 16
##                  33    47    50    117  156   601 1101 1571    152  312   612
## Predicted    6.047 11.5 16.7 11.288 4.93 20.83 5.73 5.53 11.895 30.6 14.90
## cvpred       5.789 11.4 16.6 11.121 4.61 20.82 5.44 5.25 11.762 30.4 14.86
## Price        6.580 10.2 18.3 10.600 6.19 13.20 8.37 6.67 11.248 41.3 12.44
## CV residual 0.791 -1.1  1.7 -0.521 1.58 -7.62 2.93 1.42 -0.514 10.9 -2.42
##                1052  1092 1102  1802  1832
## Predicted     9.094  8.83 5.73 15.61  2.44
## cvpred        8.958  8.63 5.44 15.48  2.27
## Price         8.560 10.35 8.37 16.50 11.85
## CV residual -0.398  1.72 2.93  1.02  9.58
##
## Sum of squares = 305    Mean square = 19.1    n = 16
##
```

```
## fold 6
## Observations in test set: 16
##                  35   98   112   381   401  981  1831 2011   122   362   672
## Predicted    6.728 5.73 4.74  9.97 10.86 5.73  2.44 13.5  9.83  9.47 14.63
## cvpred       6.621 5.77 4.75  9.86 10.50 5.77  1.47 12.9  9.56  9.45 13.88
## Price        7.300 6.30 6.10  8.85 12.95 6.30 11.85 18.1 10.70  7.90 16.63
## CV residual 0.679 0.53 1.35 -1.01  2.45 0.53 10.38  5.3  1.14 -1.55  2.75
##                972  1162  1242  1662 2012
## Predicted    21.79 11.29  6.70 17.56 13.5
## cvpred       21.59 11.38  6.85 17.81 12.9
## Price        24.57  8.85  5.50  8.45 18.1
## CV residual  2.98 -2.53 -1.35 -9.36  5.3
##
## Sum of squares = 290    Mean square = 18.1    n = 16
##
## fold 7
## Observations in test set: 16
##                  31    59  115  157    160   196   113    151   961   262
## Predicted    30.6 14.45 5.25 5.53 13.100 27.10 10.74 11.895 18.91 16.29
## cvpred       30.0 14.46 5.25 5.54 13.068 26.39 10.58 11.767 18.53 16.25
## Price        41.3 11.90 7.40 6.67 12.170 32.53  8.95 11.248 23.88 17.50
## CV residual 11.3 -2.56 2.15 1.13 -0.898  6.14 -1.64 -0.519  5.35  1.25
##                332   352   382   502  982 1562
## Predicted    6.047 6.728  9.97 16.67 5.73 4.93
## cvpred       5.987 6.701  9.90 16.44 5.66 4.93
## Price        6.580 7.300  8.85 18.28 6.30 6.19
## CV residual 0.593 0.599 -1.05  1.84 0.64 1.26
##
## Sum of squares = 220    Mean square = 13.7    n = 16
##
## fold 8
## Observations in test set: 16
##                  28    80   110   165   183   201   204   351   591   801
## Predicted    21.582  8.63 5.73 11.18  2.44 13.45 10.85 6.728 14.45  8.63
## cvpred       21.969  8.95 5.85 11.51  2.15 13.36 11.08 6.859 14.12  8.95
## Price        20.970  6.92 8.37  7.60 11.85 18.15 11.60 7.300 11.90  6.92
## CV residual -0.999 -2.03 2.52 -3.91  9.70  4.79  0.52 0.441 -2.22 -2.03
##                851  1651  1681   962  1962  2042
## Predicted    10.86 11.18 17.61 18.91 27.10 10.85
## cvpred       11.05 11.51 17.75 18.36 27.58 11.08
## Price         7.90  7.60  9.99 23.88 32.53 11.60
## CV residual -3.15 -3.91 -7.76  5.52  4.95  0.52
##
## Sum of squares = 294    Mean square = 18.4    n = 16
##
##
## fold 9
## Observations in test set: 16
##                  67   166  169   202   281   441   921   1051  1091 1151
## Predicted    14.63 17.56 18.0 23.45 21.58 39.43  9.95  9.094  8.83 5.25
## cvpred       14.79 18.00 18.5 24.22 22.44 43.59  9.87  9.001  8.88 5.31
## Price        16.63  8.45 11.2 16.56 20.97 35.55 12.29  8.560 10.35 7.40
## CV residual  1.84 -9.55 -7.3 -7.66 -1.47 -8.04  2.42 -0.441  1.47 2.09
##                1661   442   602   692  852 1572
## Predicted    17.56 39.43 20.83 17.292 10.9 5.53
## cvpred       18.00 43.59 21.54 17.714 10.9 5.55
## Price         8.45 35.55 13.20 18.150  7.9 6.67
## CV residual -9.55 -8.04 -8.34  0.436 -3.0 1.12
##
## Sum of squares = 522    Mean square = 32.6    n = 16
##
## fold 10
## Observations in test set: 16
##                  12    48   61    105   116   124   125    180   501 1561
## Predicted     9.826 11.78 14.9  9.094 11.29  6.70  8.93 15.614 16.67 4.93
## cvpred        9.910 11.83 15.0  9.174 11.34  6.82  9.02 15.579 16.66 5.07
## Price        10.698 10.80 12.4  8.560  8.85  5.50  7.10 16.500 18.28 6.19
## CV residual  0.788 -1.03 -2.6 -0.614 -2.49 -1.32 -1.92  0.921  1.62 1.12
##                1961   922   1172  1252  1602  1682
## Predicted    27.10  9.95 11.288  8.93 13.10 17.61
## cvpred       26.91 10.04 11.343  9.02 13.15 17.57
## Price        32.53 12.29 10.600  7.10 12.17  9.99
## CV residual  5.62  2.25 -0.743 -1.92 -0.98 -7.58
##
## Sum of squares = 124    Mean square = 7.78    n = 16
##
## Overall (Sum over all 16 folds)
##   ms
## 17.1
```

Claasification

```
data2<- sample(1: nrow(carDetails), 200)
carDetailsClass <- carDetails[data2,]
dim(carDetailsClass)
```

```
## [1] 200  16
```

Price to HighPrice

```
HighPrice = ifelse(carDetailsClass$Price <= 13.17, "No", "Yes")

HighPrice <- factor(HighPrice)
carDetailsClass <- add_column(carDetailsClass, HighPrice, .after = 15)

HighPriceCode = carDetailsClass$HighPrice
levels(HighPriceCode)
```

```
## [1] "No"  "Yes"
```

```
table(HighPriceCode)
```

```
## HighPriceCode
```

```
##  No Yes
## 123  77
```

```
table(HighPrice)
```

```
## HighPrice
##  No Yes
## 123  77
```

```
levels(HighPriceCode)[levels(HighPriceCode)=="No"]=0
levels(HighPriceCode)[levels(HighPriceCode)=="Yes"]=1
levels(HighPriceCode)
```

```
## [1] "0" "1"
```

```
HighPriceCode <- as.numeric(levels(HighPriceCode))[HighPriceCode]
table(HighPriceCode)
```

```
## HighPriceCode
##   0   1
## 123  77
```

Reformat Data

```
head(carDetailsClass)
```

```
##     MakeCode FuelTypeCode WheelBase Length Width Height CurbWeight
## 128       13            1      94.5    170  63.8   53.5       2024
## 61        14            1     114.2    199  68.4   58.7       3230
## 34         6            1      96.5    163  64.0   54.5       2010
## 187       12            1      96.3    173  65.4   49.4       2370
## 147        2            1      99.8    177  66.2   54.3       2337
## 68        14            0     107.9    187  68.4   56.7       3252
##     Cylinder EngineSize Bore Stroke CompressionRatio HorsePower PeakRPM
## 128     four         97 3.15   3.29              9.4         69    5200
## 61      four        120 3.46   3.19              8.4         97    5000
## 34      four         92 2.91   3.41              9.2         76    6000
## 187     four        110 3.17   3.46              7.5        116    5500
## 147     four        109 3.19   3.40             10.0        102    5500
## 68      four        152 3.70   3.52             21.0         95    4150
##      MPG HighPrice Price
## 128 49.5        No  7.35
## 61  31.0        No 12.44
## 34  47.0        No  7.30
## 187 38.0        No  9.96
## 147 39.0       Yes 13.95
## 68  44.5       Yes 17.95
```

```
carDetailsClass <- add_column(carDetailsClass, HighPriceCode, .after = 15)
carDetailsClass = carDetailsClass[,-17]
```

```
head(carDetailsClass)
```

```
##     MakeCode FuelTypeCode WheelBase Length Width Height CurbWeight
## 128       13            1      94.5    170  63.8   53.5       2024
## 61        14            1     114.2    199  68.4   58.7       3230
## 34         6            1      96.5    163  64.0   54.5       2010
## 187       12            1      96.3    173  65.4   49.4       2370
## 147        2            1      99.8    177  66.2   54.3       2337
## 68        14            0     107.9    187  68.4   56.7       3252
##     Cylinder EngineSize Bore Stroke CompressionRatio HorsePower PeakRPM
## 128     four         97 3.15   3.29              9.4         69    5200
## 61      four        120 3.46   3.19              8.4         97    5000
## 34      four         92 2.91   3.41              9.2         76    6000
## 187     four        110 3.17   3.46              7.5        116    5500
## 147     four        109 3.19   3.40             10.0        102    5500
## 68      four        152 3.70   3.52             21.0         95    4150
##      MPG HighPriceCode Price
## 128 49.5             0  7.35
## 61  31.0             0 12.44
## 34  47.0             0  7.30
## 187 38.0             0  9.96
## 147 39.0             1 13.95
## 68  44.5             1 17.95
```

```
colnames(carDetailsClass)
```

```
##  [1] "MakeCode"        "FuelTypeCode"    "WheelBase"
##  [4] "Length"          "Width"           "Height"
##  [7] "CurbWeight"      "Cylinder"        "EngineSize"
## [10] "Bore"            "Stroke"          "CompressionRatio"
## [13] "HorsePower"      "PeakRPM"         "MPG"
## [16] "HighPriceCode"   "Price"
```

Get Train and Test

```
set.seed(9003)
trainClass<-sample(1:nrow(carDetailsClass),100)
testClass<-carDetailsClass[-trainClass,]

HighPrice.train=HighPrice[trainClass]
HighPrice.test=HighPrice[-trainClass]

carDetailsClassTrain <- carDetailsClass[trainClass,]
dim(carDetailsClassTrain)
```

```
## [1] 100  17
```

```
head(carDetailsClassTrain)
```

```
##     MakeCode FuelTypeCode WheelBase Length Width Height CurbWeight
## 101        5            1      93.7    157  63.8   50.8       2128
## 141       20            1      95.7    159  63.6   54.5       2040
## 149        2            1      99.8    177  66.3   53.1       2507
## 176       21            1      97.3    172  65.5   55.7       2275
## 175       21            1      97.3    172  65.5   55.7       2212
## 191       12            1      95.9    173  66.3   50.2       2926
##     Cylinder EngineSize Bore Stroke CompressionRatio HorsePower PeakRPM
## 101     four         98 3.03   3.39              7.6        102    5500
## 141     four         92 3.05   3.03              9.0         62    4800
## 149     five        136 3.19   3.40              8.5        110    5500
## 176     four        109 3.19   3.40              9.0         85    5250
## 175     four        109 3.19   3.40              9.0         85    5250
## 191     four        156 3.59   3.86              7.0        145    5000
##      MPG HighPriceCode Price
## 101 39.0             0  7.96
## 141 50.0             0  6.34
## 149 31.5             1 15.25
## 176 44.0             0  8.50
## 175 44.0             0  8.20
## 191 31.0             1 14.49
```

```
carDetailsClassTest <- testClass
dim(carDetailsClassTest)
```

```
## [1] 100  17
```

Linear Logistic Model

```
LinearLogisticMod=glm(HighPriceCode~EngineSize, data = carDetailsClassTrain, family=binomial)
summary.glm(LinearLogisticMod)
```

```
##
## Call:
## glm(formula = HighPriceCode ~ EngineSize, family = binomial,
##     data = carDetailsClassTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.812  -0.558  -0.321   0.292   2.984
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -9.2134     1.7393   -5.30  1.2e-07 ***
## EngineSize    0.0682     0.0137    4.97  6.8e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.684  on 99  degrees of freedom
## Residual deviance:  71.955  on 98  degrees of freedom
## AIC: 75.96
##
## Number of Fisher Scoring iterations: 6
```

Training dataset linear

```
glm_prob<-predict(LinearLogisticMod, type="respon
```

```
glm_pred<- rep("No", 100)
glm_pred[glm_prob>0.5]="Yes"
```

```
table(glm_pred,HighPrice.train)
```

```
##          HighPrice.train
## glm_pred No Yes
##       No  58  11
##       Yes  6  25
```

```
table(glm_pred,HighPrice.test)
```

```
##          HighPrice.test
## glm_pred No Yes
##       No  41  28
##       Yes 18  13
```

polylogstic model

```
PolyLogisticMod=glm(HighPriceCode~poly(EnginaSize,3), data = carDetailsClassTrain, family=binomial)
summary.glm(PolyLogisticMod)
```

```
##
## Call:
## glm(formula = HighPriceCode ~ poly(EnginaSize, 3), family = binomial,
##     data = carDetailsClassTrain)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.865  -0.542  -0.355   0.241   2.636
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -0.526      0.496   -1.06    0.289
##
## poly(EnginaSize, 3)1   24.538     11.115    2.21    0.027 *
## poly(EnginaSize, 3)2   -7.074      8.724   -0.81    0.417
## poly(EnginaSize, 3)3   -4.663      6.128   -0.76    0.447
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.684  on 99  degrees of freedom
## Residual deviance:  71.587  on 96  degrees of freedom
## AIC: 79.59
##
## Number of Fisher Scoring iterations: 8
```

```
glm_prob1<-predict(PolyLogisticMod, type="response")

glm_pred1<- rep("No", 100)
glm_pred1[glm_prob1>0.5]="Yes"

table(glm_pred1,HighPrice.train)
```

```
##          HighPrice.train
## glm_pred1 No Yes
##       No  58  12
##       Yes  6  24
```

```
table(glm_pred1,HighPrice.test)
```

```
##          HighPrice.test
## glm_pred1 No Yes
##       No  41  29
##       Yes 18  12
```

interaction logistic model

```
InteractionLogisticMod=glm(HighPriceCode~Width+CurbWeight+Width*CurbWeight, data = carDe
summary.glm(InteractionLogisticMod)
```

```
##
## Call:
## glm(formula = HighPriceCode ~ Width + CurbWeight + Width * CurbWeight,
##     family = binomial, data = carDetailsClassTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6669  -0.3771  -0.1823   0.0833   2.1891
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        1.19e+02   1.78e+02    0.67     0.50
## Width             -2.07e+00   2.74e+00   -0.75     0.45
## CurbWeight        -5.30e-02   6.78e-02   -0.78     0.43
## Width:CurbWeight   9.03e-04   1.04e-03    0.87     0.39
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.684  on 99  degrees of freedom
## Residual deviance:  49.982  on 96  degrees of freedom
## AIC: 57.98
##
## Number of Fisher Scoring iterations: 8
```

```
glm_prob2<-predict(PolyLogisticMod, type="response")

glm_pred2<- rep("No", 100)
glm_pred2[glm_prob2>0.5]="Yes"

table(glm_pred2,HighPrice.train)
```

```
##          HighPrice.train
## glm_pred2 No Yes
##       No  58  12
##       Yes  6  24
```

```
table(glm_pred2,HighPrice.test)
```

```
##          HighPrice.test
## glm_pred2 No Yes
##       No  41  29
##       Yes 18  12
```

Multilinear logistic model

```
MultiLogisticMod=glm(HighPriceCode~EnginaSize+CurbWeight+Width+MPG, data = carDetailsC
summary.glm(MultiLogisticMod)
```

```
##
## Call:
## glm(formula = HighPriceCode ~ EnginaSize + CurbWeight + Width +
##     MPG, family = binomial, data = carDetailsClassTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1171  -0.2102  -0.0329   0.1407   2.0472
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -24.79574   23.66882   -1.05   0.2948
## EnginaSize   -0.00811    0.01926   -0.42   0.6738
## CurbWeight    0.00432    0.00189    2.29   0.0222 *
## Width         0.37468    0.39103    0.96   0.3380
## MPG          -0.28675    0.09445   -3.04   0.0024 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.684  on 99  degrees of freedom
## Residual deviance:  38.293  on 95  degrees of freedom
## AIC: 48.29
##
## Number of Fisher Scoring iterations: 7
```

```
glm_prob3<-predict(MultiLogisticMod, type="response")

glm_pred3<- rep("No", 100)
glm_pred3[glm_prob3>0.5]="Yes"

table(glm_pred3,HighPrice.train)
```

```
##          HighPrice.train
## glm_pred3 No Yes
##       No  60   4
##       Yes  4  32
```

```
table(glm_pred3,HighPrice.test)
```

```
##          HighPrice.test
## glm_pred3 No Yes
##       No  38  26
##       Yes 21  15
```

Selection model

```
HybridlogicModel = glm(HighPriceCode~EnginaSize+MPG, data = carDetailsClassTrain, family=binomial)
summary.glm(HybridlogicModel)
```

```
##
## Call:
## glm(formula = HighPriceCode ~ EnginaSize + MPG, family = binomial,
##     data = carDetailsClassTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2824  -0.3486  -0.0957   0.2786   1.8640
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   5.9078     3.7952    1.56   0.1196
## EnginaSize    0.0361     0.0158    2.28   0.0224 *
## MPG          -0.2909     0.0783   -3.72   0.0002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.684  on 99  degrees of freedom
## Residual deviance:  51.465  on 97  degrees of freedom
## AIC: 57.46
##
## Number of Fisher Scoring iterations: 7
```

```
glm_prob4<-predict(HybridlogicModel, type="response")

glm_pred4<- rep("No", 100)
glm_pred4[glm_prob4>0.5]="Yes"

table(glm_pred4,HighPrice.train)
```

```
##          HighPrice.train
## glm_pred4 No Yes
##       No  60   4
##       Yes  4  32
```

```
table(glm_pred4,HighPrice.test)
```

```
##          HighPrice.test
## glm_pred4 No Yes
##       No  39  25
##       Yes 20  16
```

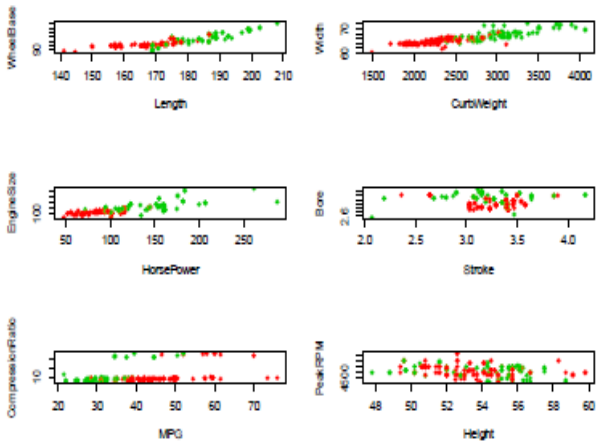
```
anova(HybridlogicModel)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: HighPriceCode
##
## Terms added sequentially (first to last)
##
##
##             Df Deviance Resid. Df Resid. Dev
## NULL                           99      130.7
## EnginaSize   1     58.7        98       72.0
## MPG          1     20.5        97       51.5
```

PCA

```
par(mfrow=c(3,2))
plot(WheelBase~Length, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
plot(Width~CurbWeight, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
plot(EngineSize~HorsePower, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
plot(Bore~Stroke, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
plot(CompressionRatio~MPG, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
```

```
plot(PeakRPM~Height, data=carDetailsClass,
     col=unclass(HighPrice)+1, pch=16)
```



Reformat dataset

```
View(carDetailsClass)
carDetailsClass = carDetailsClass[,-8]

carDetailsClass_new <- carDetailsClass[,3:5]

carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,7])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,8])
head(carDetailsClass_new)
```

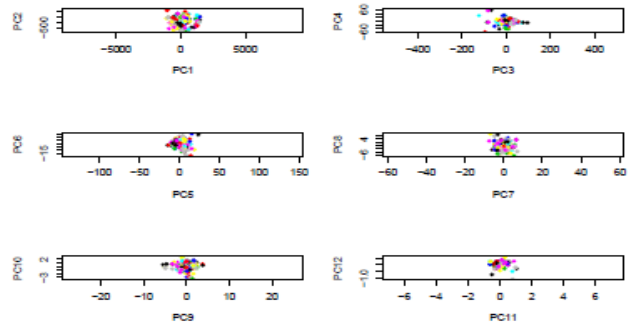
```
##     WheelBase Length Width carDetailsClass[, 7] carDetailsClass[, 8]
## 128      94.5    170  63.8                 2024                   97
## 61      114.2    199  68.4                 3230                  120
## 34       96.5    163  64.0                 2010                   92
## 187      96.3    173  65.4                 2370                  110
## 147      99.8    177  66.2                 2337                  109
## 68      107.9    187  68.4                 3252                  152
```

```
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,12])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,9:10])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,11])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,14])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,13])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,6])
carDetailsClass_new <- cbind(carDetailsClass_new, carDetailsClass[,16])

colnames(carDetailsClass_new) <- c("WheelBase", "Length", "Width",
                                   "CurbWeight", "EngineSize", "HorsePower" ,
                                   "Bore", "Stroke", "CompressionRatio","MPG",
                                   "PeakRPM", "Height", "HighPrice")
```

PCA Perform

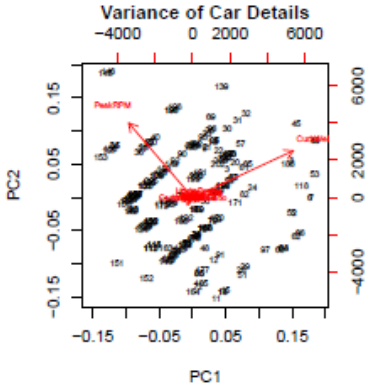
```
carDetailsClass_new <- subset(carDetailsClass_new, carDetailsClass_new$Bore>0)
View(carDetailsClass_new)
obj3 = prcomp(carDetailsClass_new[,1:12]) # perform PCA
par(mfrow=c(3,2))
obj3 = prcomp(carDetailsClass_new[,1:12]) # perform PCA
plot(obj3$x[,1:2], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
plot(obj3$x[,3:4], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
plot(obj3$x[,5:6], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
plot(obj3$x[,7:8], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
plot(obj3$x[,9:10], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
plot(obj3$x[,11:12], col=unclass(carDetailsClass_new$HighPrice)+1,pch=16, asp=1)
```




```
par(mfrow=c(1,1))
biplot(obj3, cex= 0.5, main = "Variance of Car Details")
```

```
## Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length
## = arrow.len): zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length
## = arrow.len): zero-length arrow is of indeterminate angle and so skipped
```




```
summary(obj3)
```

```
## Importance of components:
##                           PC1      PC2      PC3       PC4      PC5   PC6
## Standard deviation     571.023 422.189 28.08530 13.88633 6.33803 5e+00
## Proportion of Variance   0.645   0.353  0.00156  0.00038 0.00008 5e-05
## Cumulative Proportion    0.645   0.998  0.99946  0.99984 0.99992 1e+00
##                            PC7     PC8     PC9  PC10  PC11  PC12
## Standard deviation     2.59636 2.26506 1.62220 0.896 0.296 0.186
## Proportion of Variance 0.00001 0.00001 0.00001 0.000 0.000 0.000
## Cumulative Proportion  0.99998 0.99999 1.00000 1.000 1.000 1.000
```

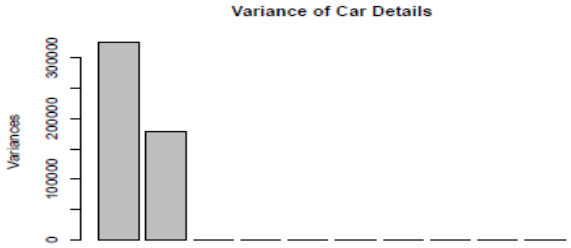
```
obj3$center
```

```
##         WheelBase            Length             Width         CurbWeight
##             98.89             174.20             65.92            2563.50
##        EngineSize        HorsePower              Bore            Stroke
##             128.23             104.50              3.33              3.25
## CompressionRatio               MPG           PeakRPM            Height
##             10.19              40.85            5108.93             53.81
```

```
obj3$sdev
```

```
##  [1] 571.023 422.189  28.085  13.886   6.338   5.001   2.596   2.265
##  [9]   1.622   0.896   0.296   0.186
```

```
screeplot(obj3, main = "Variance of Car Details")
```



Different models
Toyota

```
ToyotaCarDetails <- subset(carDetails, carDetails$MakeCode==20)
dim(ToyotaCarDetails)
```

```
## [1] 32 16
```

```
ToyotaselectModal = lm(Price~EngineSize+Width+CurbWeight+Width*CurbWeight+I(CurbWeight*C
summary(ToyotaselectModal)
```

```
##
## Call:
## lm(formula = Price ~ EngineSize + Width + CurbWeight + Width *
##     CurbWeight + I(CurbWeight * CurbWeight), data = ToyotaCarDetails)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.818 -0.442 -0.182  0.438  4.495
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  2.81e+02   1.21e+02    2.33    0.028 *
## EngineSize                  -1.86e-02   2.45e-02   -0.76    0.454
## Width                       -4.66e+00   2.08e+00   -2.24    0.034 *
## CurbWeight                  -1.24e-01   4.32e-02   -2.87    0.008 **
## I(CurbWeight * CurbWeight) -2.68e-06   3.08e-06   -0.87    0.392
## Width:CurbWeight             2.22e-03   8.23e-04    2.69    0.012 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.25 on 26 degrees of freedom
## Multiple R-squared:  0.872,  Adjusted R-squared:  0.847
## F-statistic: 35.4 on 5 and 26 DF,  p-value: 8.4e-11
```

```
anova(ToyotaselectModal)
```

```
## Analysis of Variance Table
##
## Response: Price
##                              Df Sum Sq Mean Sq F value   Pr(>F)
## EngineSize                    1  225.2   225.2  143.55 4.3e-12 ***
## Width                         1    4.7     4.7    3.01   0.0944 .
## CurbWeight                    1   32.9    32.9   20.98   0.0001 ***
## I(CurbWeight * CurbWeight)    1    3.3     3.3    2.11   0.1585
## Width:CurbWeight              1   11.4    11.4    7.26   0.0122 *
## Residuals                    26   40.8     1.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Nissan

```
NissanCarDetails <- subset(carDetails, carDetails$MakeCode==13)
dim(NissanCarDetails)
```

```
## [1] 18 16
```

```
NissanselectModal = lm(Price~EngineSize+Width+CurbWeight+Width*CurbWeight+I(CurbWeight*
summary(NissanselectModal)
```

```
##
## Call:
## lm(formula = Price ~ EngineSize + Width + CurbWeight + Width *
##     CurbWeight + I(CurbWeight * CurbWeight), data = NissanCarDetails)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.712 -0.285 -0.063  0.266  0.980
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  5.04e+02   1.34e+02    3.77  0.00266 **
## EngineSize                  -1.00e-01   3.25e-02   -3.09  0.00930 **
## Width                       -8.54e+00   2.30e+00   -3.71  0.00297 **
## CurbWeight                  -2.00e-01   3.73e-02   -5.36  0.00017 ***
## I(CurbWeight * CurbWeight) -7.83e-06   2.47e-06   -3.17  0.00805 **
## Width:CurbWeight             3.84e-03   7.45e-04    5.15  0.00024 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.522 on 12 degrees of freedom
## Multiple R-squared:  0.99,   Adjusted R-squared:  0.986
## F-statistic:  248 on 5 and 12 DF,  p-value: 1.13e-11
```

```
anova(NissanselectModal)
```

```
## Analysis of Variance Table
##
## Response: Price
##                              Df Sum Sq Mean Sq F value   Pr(>F)
## EngineSize                    1  298.4   298.4 1095.79 3.7e-13 ***
## Width                         1   26.9    26.9   98.88 3.8e-07 ***
## CurbWeight                    1    2.7     2.7   10.00 0.00819 **
## I(CurbWeight * CurbWeight)    1    2.2     2.2    8.12 0.01463 *
## Width:CurbWeight              1    7.2     7.2   26.55 0.00024 ***
## Residuals                    12    3.3     0.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
MitsubishiCarDetails <- subset(carDetails, carDetails$MakeCode==12)
dim(MitsubishiCarDetails)
```

```
## [1] 13 16
```

```
MitsubishiselectModal = lm(Price~EngineSize+Width+CurbWeight+Width*CurbWeight+I(CurbWeight*CurbWei
summary(MitsubishiselectModal)
```

```
##
## Call:
## lm(formula = Price ~ EngineSize + Width + CurbWeight + Width *
##     CurbWeight + I(CurbWeight * CurbWeight), data = MitsubishiCarDetails)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.946 -0.324 -0.122  0.185  1.078
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  2.31e+02   1.06e+03    0.22    0.833
## EngineSize                  -9.42e-02   4.26e-02   -2.21    0.063 .
## Width                       -3.43e+00   1.73e+01   -0.20    0.849
## CurbWeight                  -1.01e-01   4.57e-01   -0.22    0.831
## I(CurbWeight * CurbWeight)  3.62e-06   1.44e-05    0.25    0.809
## Width:CurbWeight            1.50e-03   8.00e-03    0.19    0.856
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.698 on 7 degrees of freedom
## Multiple R-squared:  0.969,  Adjusted R-squared:  0.947
## F-statistic: 44.2 on 5 and 7 DF,  p-value: 3.8e-05
```

```
anova(MitsubishiselectModal)
```

```
## Analysis of Variance Table
##
## Response: Price
##                              Df Sum Sq Mean Sq F value  Pr(>F)
## EngineSize                    1   90.7    90.7  186.16 2.7e-06 ***
## Width                         1    0.2     0.2    0.45  0.5217
## CurbWeight                    1   12.2    12.2   24.96  0.0016 **
## I(CurbWeight * CurbWeight)    1    4.6     4.6    9.35  0.0184 *
## Width:CurbWeight              1    0.0     0.0    0.04  0.8562
## Residuals                     7    3.4     0.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

comepare

```
par(mfrow=c(1,3))

plot(Price~EngineSize, data = ToyotaCarDetails, xlab="EngineSize in Cubic Inches",ylab = "Price in t
lines(smooth.spline(ToyotaCarDetails$EngineSize,predict(ToyotaselectModal)), col= "blue",lwd=2, lty
par(new = TRUE)
plot(Price~EngineSize, data = NissanCarDetails, xlab="EngineSize in Cubic Inches",ylab = "Price in t
lines(smooth.spline(NissanCarDetails$EngineSize,predict(NissanselectModal)), col= "green",lwd=2, lt
par(new = TRUE)
plot(Price~EngineSize, data = MitsubishiCarDetails, xlab="EngineSize in Cubic Inches",ylab = "Price
lines(smooth.spline(MitsubishiCarDetails$EngineSize,predict(MitsubishiselectModal)), col= "red",lwd
legend(100, 14, legend=c("Totota", "Nissan", "Mitsubishi"),col=c("green", "blue", "red"), c( 3, 4, 2

plot(Price~CurbWeight, data = ToyotaCarDetails, xlab="CurbWeight in grams",ylab = "Price in thousan
lines(smooth.spline(ToyotaCarDetails$CurbWeight,predict(ToyotaselectModal)), col= "blue",lwd=2, lty
par(new = TRUE)
plot(Price~CurbWeight, data = NissanCarDetails, xlab="CurbWeight in gramss",ylab = "Price in thousa
lines(smooth.spline(NissanCarDetails$CurbWeight,predict(NissanselectModal)), col= "green",lwd=2, lt
par(new = TRUE)
plot(Price~CurbWeight, data = MitsubishiCarDetails, xlab="CurbWeight in grams",ylab = "Price in tho
lines(smooth.spline(MitsubishiCarDetails$CurbWeight,predict(MitsubishiselectModal)), col= "red",lwd
legend(2000, 14, legend=c("Totota", "Nissan", "Mitsubishi"),col=c("green", "blue", "red"), c( 3, 4,

plot(Price~Width, data = ToyotaCarDetails, xlab="Width in centimeter",ylab = "Price in thousands of
lines(smooth.spline(ToyotaCarDetails$Width,predict(ToyotaselectModal)), col= "blue",lwd=2, lty=1)
par(new = TRUE)
plot(Price~Width, data = NissanCarDetails, xlab="Width in centimeter",ylab = "Price in thousands of
lines(smooth.spline(NissanCarDetails$Width,predict(NissanselectModal)), col= "green",lwd=2, lty=1)
par(new = TRUE)
plot(Price~Width, data = MitsubishiCarDetails, xlab="Width in centimeter",ylab = "Price in thousand
lines(smooth.spline(MitsubishiCarDetails$Width,predict(MitsubishiselectModal)), col= "red",lwd=2, l
legend(65, 14, legend=c("Totota", "Nissan", "Mitsubishi"),col=c("green", "blue", "red"), c( 3, 4, 2
```