DATA 624

Presentation on Linear Regression and its Cousins

Frederick Jones, Melissa Bowman, Gabriel Campos, Shoshana Farber

AN INTRODUCTION TO REGRESSION

- Regression refers to the technique which is used to find the relation between **dependent or response variable** with one or more **independent or predictor variables**. The relation between dependent and independent variables might be linear or non-linear.
- If we have data in the form $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$ then expected value $E(Y|X = x) = \beta_0 + \beta_1 x$
- There might be random error which cannot be predicted thus $Y_i = \beta_0 + \beta_1 x + e_i$
- But the linear regression model has the form $\hat{y} = b_0 + bx$, where b_0 *and* b_1 are unbiased estimates of β_0 *and* β_1 respectively. Slope refers to the rise in y for each unit increment in x, and $\hat{y}$ is known as an estimate of y or expected value of y.
- For each value of x, we get an estimate for y or we can write $model(x_i) \Rightarrow \hat{y}_i$ or we get a pair $(x_i, \hat{y}_i)$ from the model for each input x and joining all the points gives us a straight line called as the **Best fit line** that is why the regression is known as the **Linear Regression**.
- The difference $(y - \hat{y})$ is termed as an error or residual e_i. Sum of all the residuals is always zero. Thus, the Residual Sum of Squares (RSS) is calculated and minimized to get the **Best fit line**.

MATHEMATICAL TREATMENT OF RESIDUALS

- Residual Sum of Squares, $RSS = \sum e_i^2 = \sum(y_i - \hat{y}_i)^2$

$$RSS = \sum_{i=1}^{n}(y_i - b_0 - b_1 x)^2$$

- For RSS to be a minimum, It must be partially differentiated with respect to b_0 and b_1

$$\frac{\partial(RSS)}{\partial b_0} = -2\sum_{i=1}^{n}(y_i - b_0 - b_1 x) = 0$$

$$\frac{\partial(RSS)}{\partial b_1} = -2\sum_{i=1}^{n}x_i(y_i - b_0 - b_1 x) = 0$$

MATHEMATICAL TREATMENT OF RESIDUALS

- Rearranging and solving these two gives us

$$b_1 = \hat{\beta}_1 = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} = \frac{S_{xy}}{S_{xx}}$$

$$b_0 = \bar{y} - b_1\,\bar{x}$$

- **Assumptions** for the single predictor:

1. y is related to x and shows a linear trend on the scatter plot of (x_i, y_i)
2. The errors $e_1, e_2, e_3, \dots e_n$ are independent of each other
3. The errors have a common variance which is constant.
4. The errors are normally distributed with mean 0 and variance σ^2

- Scatter plot and the line of best fit
- The diagram shows that the dots referring to data points are scattered around the line of the best fit.

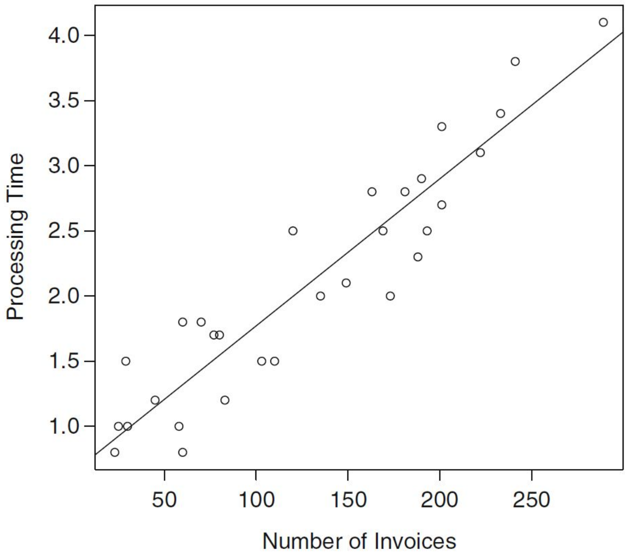


Figure 1: A typical scatter plot with line of best fit

(Source: A modern approach to regression with R)

VALIDITY OF A REGRESSION MODEL

- A regression model is considered to be valid if all the four assumptions are satisfied else the regression model is not valid for any inference.
- **Assumption 1**. There should be a valid linear trend in the data which can be viewed in scatter plot.

In the figure on right, it can be seen that the Data set 1 gives valid linear regression while other data set do not give a valid linear regression. Data set 2 is not linear, Data set 3 has an outlier and Data set 4 also does not give a valid linear regression.

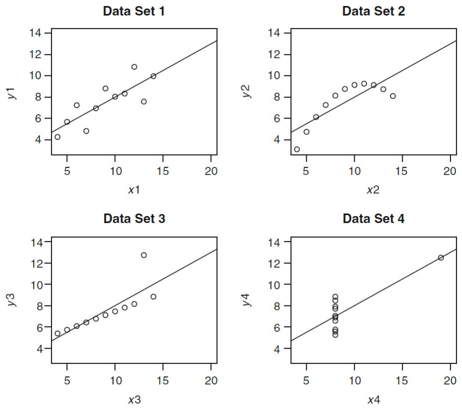


Figure 2: Plot showing trends

VALIDITY OF A REGRESSION MODEL

- **Assumption 2**. Homoscedasticity of Residuals or equal variances

First plot shows uniform variance Second plot shows non-uniform Variance. Thus, first plot will give valid linear regression model and Second will not give valid linear regression model.

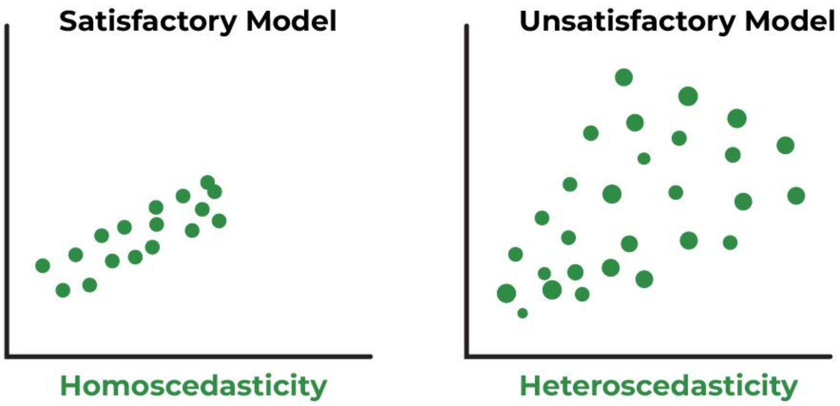


Figure 3: Plot showing homoscedasticity and heteroscedasticity

VALIDITY OF A REGRESSION MODEL

- **Assumption 3**. Number of samples must be greater than the number of predictor variables.

This assumption is critical for the inference of the result obtained from the linear regression. If the number of predictor variables are more than the number of samples (number of rows less than the number of columns) then the result obtained from the linear regression becomes invalid. This assumption needs to be checked even before the split of the dataset. Suppose, we have a data frame such that X has 100 rows and 80 columns so that 100 samples and 80 predictor variable for Y and if we split the data set in the ratio of 70:30 such that the training set has 70 rows and test set has 30 rows. Since 70 is less than 80 hence the number of samples becomes less than the number of predictor variables. Thus, the regression model becomes invalid for any interpretation.

- **Assumption 4.** The residuals must be a white noise. If there is any trend in the residuals then the regression model becomes invalid. The coefficients will not be able to capture the trend and make predictions.

Hence, there is need to check the assumptions before drawing any conclusions.

MULTIPLE LINEAR REGRESSION MODEL

- When the number of predictor variables are more than one then the linear regression is termed as the multiple linear regression. It has mathematical form as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + error$$

- But mathematically by minimizing the residuals we get estimates for coefficients $\beta_0, \beta_1, \ldots \beta_n$ written as $b_0, b_1, b_2 \ldots b_n$.
- Multiple linear regression is valid when there is no multicollinearity among the predictors. If multicollinearity or autocorrelation is found among the variables then there is need to transform the data before running any multiple regression model.
- The transformations such as log transformation, Box-Cox transformation have wide range of applications to make data suitable for further regression.
- The suitable transformation is found by looking the nature of the data and relation between response and predictor variables.

COUSINS OF REGRESSION

- The “cousins” of linear regression share the same purpose, methodology, and principles as linear regression.
- Some “cousins” may include logistic regression and polynomial regression.
- Logistic regression shares some of the same assumptions as linear regression and also involves estimating coefficients to explain the relationship between predictor variables and response variables. Logistic regression is a form of a **Generalized Linear Model** which basically allows for a different type of response variable (binary or categorical instead of continuous).
- Polynomial regression allows for a curved relationship between predictors and response rather than a straight line.
- Other “cousins” include **Penalized Regression Models** such as **lasso** and **ridge** regression which can be used to address collinearity and overfitting for linear models.

CAUSE OF COUSINS OF REGRESSION

- If any of the assumptions of the linear regression is violated, it causes a problem to the inferences of the result and leading to the formulation of the cousins of the regression. To address the issue resulted due to the violation of any of the assumptions there is need to modify the technique of the linear regression leading to a new type of model which is known as cousin of linear regression.
- For example if multicollinearity is found among the predictors then there are two options to address this problem: (a) remove the predictors which have the correlation among them leading to Partial regression (b) to add bias so that the model has reduced the root mean squared error (RMSE) leading to the Ridge regression.
- The problem of multicollinearity can be addressed by using the dimensional reduction technique such as **Principal Component Analysis (PCA)** or **Partial Least Squares (PLS)**.
- PLS is based on the Herman Wold's Non Linear Iterative Partial Least Squares (NIPALS) algorithm (Wold 1982). This algorithm handles correlated predictors efficiently when sample size is less than 2500.
- If response variable is not continuous numeric then there is need to use Bayesian Probabilistic model which can handle binary or categorical response variable leading to logistic regression etc.

R IMPLEMENTATION OF REGRESSION

- The algorithm Simple PLS (SIMPLS) developed by de Jong (1993) through statistics, deflates the covariance matrix at each iteration making it more efficient than NIPALS.
- **R Implementations:**

Let us assume that the data frame containing predictor variables are X_train, and response variable is y_train, then the R function lm() can be used to get the linear regression model as follows:

```
model <- lm(y~ X_train)
```

If the we have predictors as well as response in a data frame df_train then we can get the model as:

```
model <- lm(y~., data = df_train)
```

Here the '.' refers to all the variables except y in the data frame df_train. The summary of the model can be found using the function summary()

```
summary(model)
```

R IMPLEMENTATION OF REGRESSION

Prediction based on the model:

```
pred <- predict(model, X_test)
head(pred)
```

The observed and predicted test values can be collected into a data frame then defaultSumary() from caret can be used to estimate the test set perfomrances.

```
df_Ob_Pred <- data.frame(obs=y_test,pred= pred)
defaultSummary(df_Ob_Pred)
```

Robust linear regression model can be generated using rlm() from MASS package. The argument of rlm are similar to lm(). It will produce result with crossvalidation equals to 10

R IMPLEMENTATION OF REGRESSION

Regression model without highly correlated predictors:

```
threshold <- 0.8
high_cor <- findCorrelation(cor(X_train), threshold)
var_to_rm <-names(X_train)[high_cor]
X_train <- X_train[, -var_to_rm]
X_test <- X_test[, -var_to_rm]
```

Now X_train can be used to build the regression model. It will be just like PLS.

```
set.seed(100)
filtered_model <- train(X_train, y_train, method=“lm”,
+ trControl=ctrl)
summary(filtered_model)
```

This model will drop the any possible highly correlated variable whose correlation between then is more than 0.80

R IMPLEMENTATION OF REGRESSION

Robust Regression model without highly correlated predictors using train():

```
set.seed(100)
robust_model <- train(X_train, y_train, method="rlm",
    + preProcess = 'pca'
    + trControl=ctrl)
summary(robust_model)
```

This will produce summary of the robust model with 10 fold cross-validations.

PARTIAL LEAST SQUARES (PLS)

The package pls has functions for PLS and PCR. The main function in this package is plsr(). This function has arguments for keyword method as ‘oscorespls’, ‘simpls’, ‘widekernelpls’

```
plsr(y~., data = df_train)
```

Or

```
pls.fit <- plsr(y_train~ X_train)
pred <- predict(pls.fit, X_test, ncomp =1:2)
```

The function plsr has options for K-fold, PLS algorithm to be used by method argument. Fine tune the model:

```
set.seed(100)
plsTune <- train(X_train, y_train, method = “pls”, tuneLength=20,
        trControl =ctrl, preProc = c(“center”, “scale”))
```

PENALIZED REGRESSION MODELS

Ridge regression model can be created using lm.ridge() from MASS package or enet() from elasticnet package. Lambda argument in enet() specify the penalty for the ridge regression.

```
ridgeModel <- enet(x=as.matrix(X_train), y=y_train, lambda = 0.001)
```

- The function enet() has both ridge penalty as well as lasso penalty. The predict function for enet generates predictions for one or more value of lasso penalty using s and mode arguments. For the ridge regression, we have s=1 and mode="fraction"

```
ridgePred <- predict(ridgeModel, as.matrix(X_test), s=1,
         mode="fraction", type ="fit")
```

- The lasso model can be generated using lars() from package lars, enet() from elasticnet and glmnet() from glmnet package.

```
lassoModel <- enet(x=as.matrix(X_train), y=y_train, lambda = 0.001,
             normalize = TRUE)
```

USE CASE: CAR PRICE PREDICTION MULTIPLE LINEAR REGRESSION

- Each entry corresponds to a car
- 205 entries
- 25 variables
  - 24 predictor variables
  - 1 response variable - **price**

- **GOAL:** Predict the price of the car based on the numerous features of the car (the engine size, horsepower, make, model, miles per gallon, etc.)

DATA EXPLORATION

- Preview data using glimpse()
- Data made up of numeric and non-numeric variables

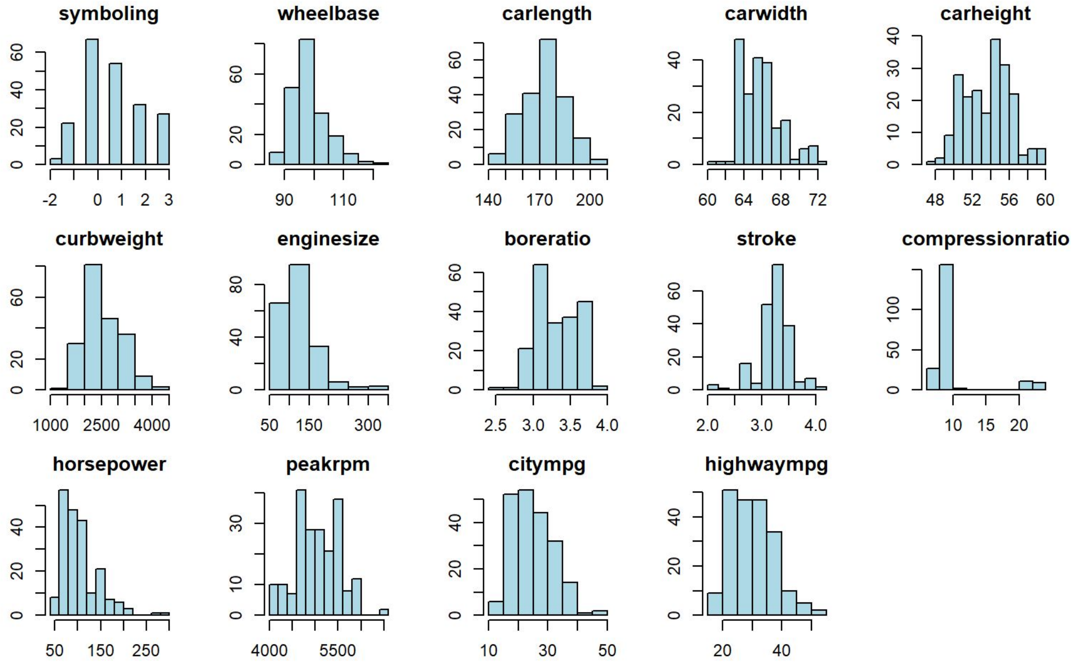
- No missing values

```
     symboling        CarName       fueltype       aspiration
             0              0              0                0
    doornumber        carbody     drivewheel   enginelocation
             0              0              0                0
     wheelbase      carlength       carwidth        carheight
             0              0              0                0
    curbweight     enginetype cylindernumber       enginesize
             0              0              0                0
    fuelsystem      boreratio         stroke compressionratio
             0              0              0                0
    horsepower        peakrpm        citympg       highwaympg
             0              0              0                0
         price
             0
```

```
Rows: 205
Columns: 25
$ symboling        <int> 3, 3, 1, 2, 2, 2, 1, 1, 1, 0, 2, 0, 0, 0, 1, 0, 0, 0, …
$ CarName          <chr> "alfa-romero giulia", "alfa-romero stelvio", "alfa-rom…
$ fueltype         <chr> "gas", "gas", "gas", "gas", "gas", "gas", "gas", "gas"…
$ aspiration       <chr> "std", "std", "std", "std", "std", "std", "std", "std"…
$ doornumber       <chr> "two", "two", "two", "four", "four", "two", "four", "f…
$ carbody          <chr> "convertible", "convertible", "hatchback", "sedan", "s…
$ drivewheel       <chr> "rwd", "rwd", "rwd", "fwd", "4wd", "fwd", "fwd", "fwd"…
$ enginelocation   <chr> "front", "front", "front", "front", "front", "front", …
$ wheelbase        <dbl> 88.6, 88.6, 94.5, 99.8, 99.4, 99.8, 105.8, 105.8, 105.…
$ carlength        <dbl> 168.8, 168.8, 171.2, 176.6, 176.6, 177.3, 192.7, 192.7…
$ carwidth         <dbl> 64.1, 64.1, 65.5, 66.2, 66.4, 66.3, 71.4, 71.4, 71.4, …
$ carheight        <dbl> 48.8, 48.8, 52.4, 54.3, 54.3, 53.1, 55.7, 55.7, 55.9, …
$ curbweight       <int> 2548, 2548, 2823, 2337, 2824, 2507, 2844, 2954, 3086, …
$ enginetype       <chr> "dohc", "dohc", "ohcv", "ohc", "ohc", "ohc", "ohc", "o…
$ cylindernumber   <chr> "four", "four", "six", "four", "five", "five", "five",…
$ enginesize       <int> 130, 130, 152, 109, 136, 136, 136, 136, 131, 131, 108,…
$ fuelsystem       <chr> "mpfi", "mpfi", "mpfi", "mpfi", "mpfi", "mpfi", "mpfi"…
$ boreratio        <dbl> 3.47, 3.47, 2.68, 3.19, 3.19, 3.19, 3.19, 3.19, 3.13, …
$ stroke           <dbl> 2.68, 2.68, 3.47, 3.40, 3.40, 3.40, 3.40, 3.40, 3.40, …
$ compressionratio <dbl> 9.00, 9.00, 9.00, 10.00, 8.00, 8.50, 8.50, 8.50, 8.30,…
$ horsepower       <int> 111, 111, 154, 102, 115, 110, 110, 110, 140, 160, 101,…
$ peakrpm          <int> 5000, 5000, 5000, 5500, 5500, 5500, 5500, 5500, 5500, …
$ citympg          <int> 21, 21, 19, 24, 18, 19, 19, 19, 17, 16, 23, 23, 21, 21…
$ highwaympg       <int> 27, 27, 26, 30, 22, 25, 25, 25, 20, 22, 29, 29, 28, 28…
$ price            <dbl> 13495.00, 16500.00, 16500.00, 13950.00, 17450.00, 1525…
```

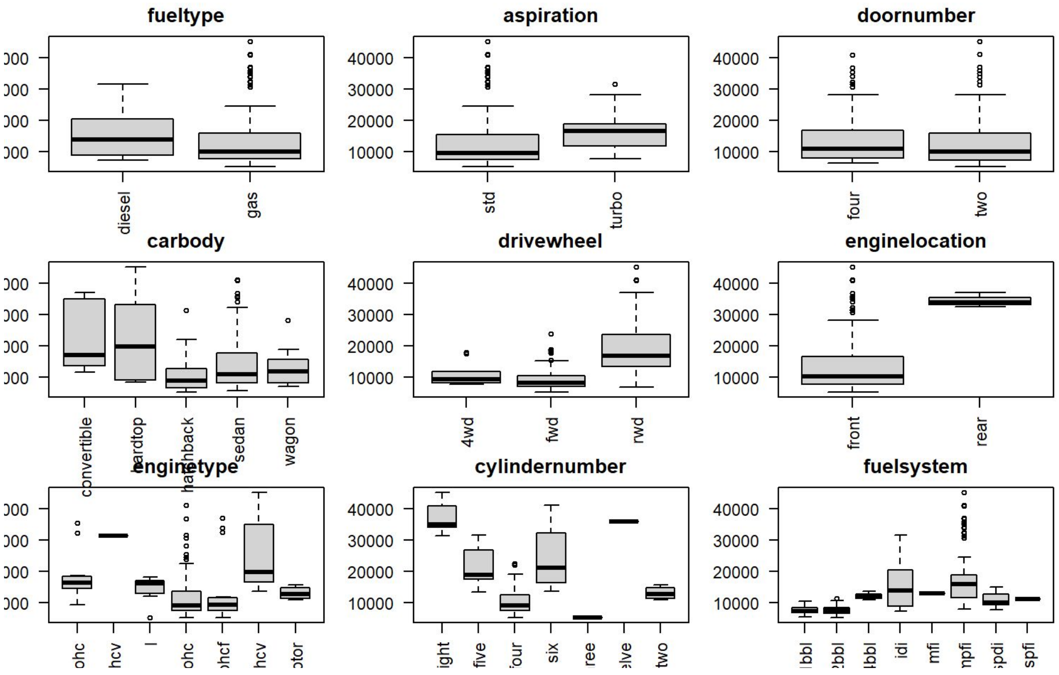
DATA EXPLORATION

- Distributions of numerical predictors
- enginesize and horsepower have a left skew



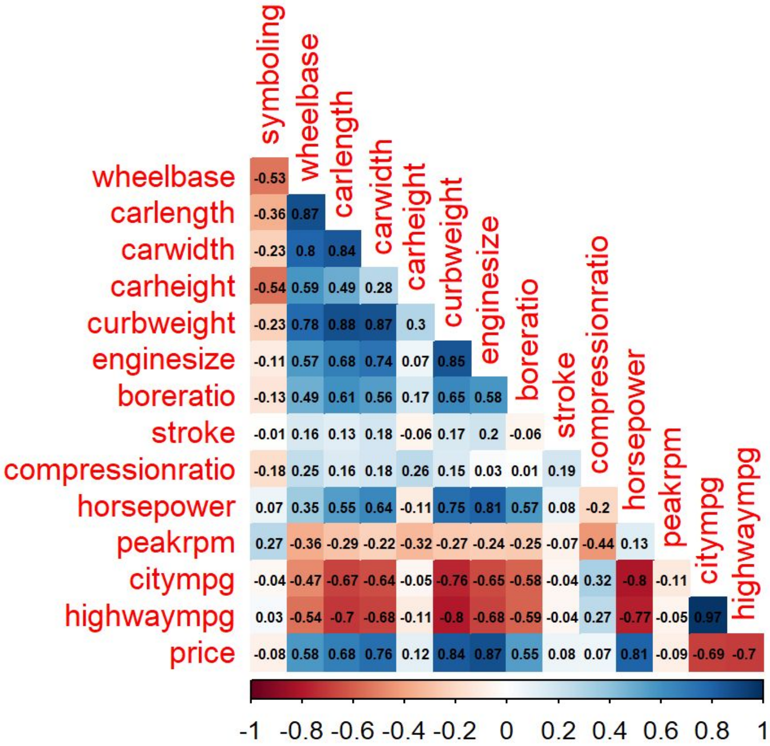
DATA EXPLORATION

- Distributions of categorical variables
- Diesel cars priced higher than gas
- Turbo cars price higher than standard
- Convertible and hardtops priced higher but also more varied
- Rear wheel drive priced higher
- Rear engines priced higher
- Eight cylinder engines priced high



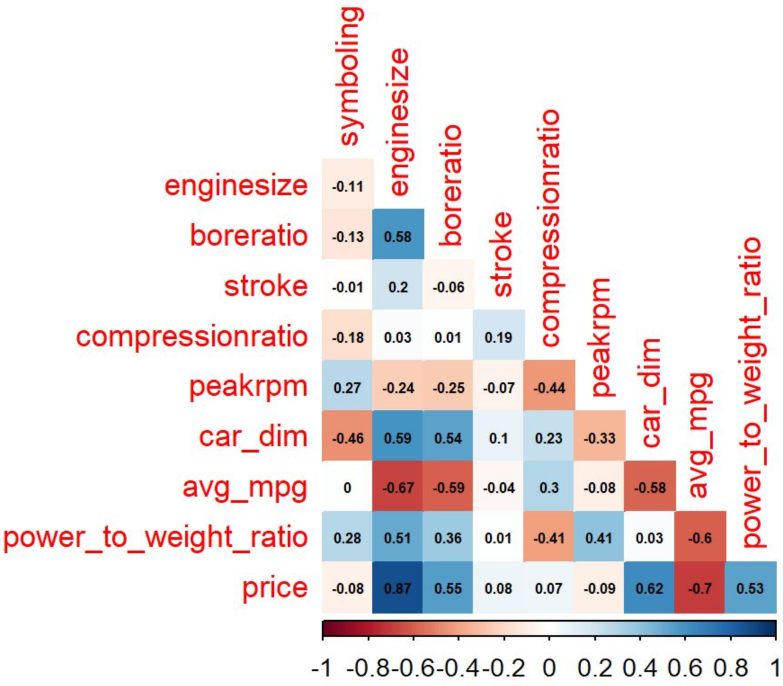
DATA EXPLORATION

- Correlations of numeric variables
- Price positively correlated with wheelbase, carlength, carwidth, curbweight, enginesize, horsepower
- Price negatively correlated with citympg, highwaympg
- Many highly correlated predictor variables (ex: carwidth, wheelbase, carlength, carheight; citympg, highwaympg)



DATA PREPARATION

- New variables to deal with collinearity:
  - car_dim = carlength*carwidth*carheight
  - avg_mpg = (citympg + highwaympg) / 2
  - power_to_weight_ratio = horsepower / curbweight
- Remove cylindernumber - singularity errors



MODEL BUILDING

- “Stepwise” method:

  model <- lm(price~., car_price)

  model_updated <- step(model)

```
Residuals:
    Min      1Q  Median      3Q     Max
-6618.7 -1313.3  -167.5  1107.3 10359.1

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        -1.868e+03  8.138e+03  -0.230 0.818737
aspirationturbo     2.385e+03  6.889e+02   3.463 0.000671 ***
carbodyhardtop     -4.648e+03  1.413e+03  -3.289 0.001214 **
carbodyhatchback   -4.324e+03  1.184e+03  -3.653 0.000341 ***
carbodysedan       -4.105e+03  1.216e+03  -3.375 0.000909 ***
carbodywagon       -5.682e+03  1.374e+03  -4.134 5.50e-05 ***
drivewheelfwd      -6.180e+02  1.017e+03  -0.608 0.544217
drivewheelrwd       1.498e+03  1.155e+03   1.298 0.196055
enginelocationrear  8.112e+03  2.135e+03   3.800 0.000199 ***
enginetypedohcv     8.023e+03  2.715e+03   2.955 0.003556 **
enginetypel        -4.899e+02  1.281e+03  -0.382 0.702607
enginetypeohc       2.756e+03  8.703e+02   3.167 0.001816 **
enginetypeohcf      2.728e+03  1.581e+03   1.725 0.086325 .
enginetypeohcv     -3.000e+03  1.144e+03  -2.622 0.009518 **
enginetyperotor     1.360e+04  2.739e+03   4.966 1.61e-06 ***
enginesize          1.829e+02  1.122e+01  16.301  < 2e-16 ***
fuelsystem2bbl      3.880e+02  9.239e+02   0.420 0.675060
fuelsystem4bbl     -1.010e+03  2.987e+03  -0.338 0.735752
fuelsystemidi       1.904e+04  5.983e+03   3.182 0.001727 **
fuelsystemmfi      -2.810e+03  2.766e+03  -1.016 0.311075
fuelsystemmpfi      7.354e+02  9.752e+02   0.754 0.451802
fuelsystemspdi     -2.529e+03  1.375e+03  -1.838 0.067685 .
fuelsystemspfi      9.792e+02  2.663e+03   0.368 0.713494
boreratio          -5.092e+03  1.188e+03  -4.285 3.00e-05 ***
stroke             -4.016e+03  9.033e+02  -4.446 1.55e-05 ***
compressionratio   -1.372e+03  4.310e+02  -3.184 0.001718 **
peakrpm             2.774e+00  5.897e-01   4.704 5.13e-06 ***
car_dim             2.680e-02  5.215e-03   5.139 7.28e-07 ***
avg_mpg             1.483e+02  7.021e+01   2.112 0.036070 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2441 on 176 degrees of freedom
Multiple R-squared:  0.9195,    Adjusted R-squared:  0.9067
F-statistic: 71.78 on 28 and 176 DF,  p-value: < 2.2e-16
```

Significance of variable

Model’s ability to capture variability

VARIANCE INFLATION FACTOR

- car package

  car::vif(model_updated)

- VIF > 5 shows a high multicollinearity issue

	GVIF	Df	GVIF^(1/(2*Df))
aspiration	2.415550	1	1.554204
carbody	3.496517	4	1.169377
drivewheel	4.759044	2	1.476998
enginelocation	2.261583	1	1.503856
enginetype	91.460718	6	1.456922
enginesize	7.478188	1	2.734628
fuelsystem	2465.098287	7	1.746923
boreratio	3.547939	1	1.883597
stroke	2.748372	1	1.657821
compressionratio	100.368942	1	10.018430
peakrpm	2.709323	1	1.646002
car_dim	5.880626	1	2.425000
avg_mpg	7.501759	1	2.738934

DIAGNOSTICS

```
Residuals:
    Min      1Q  Median      3Q     Max 
-6618.7 -1313.3  -167.5  1107.3 10359.1 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)         -1.868e+03  8.138e+03  -0.230 0.818737    
aspirationturbo      2.385e+03  6.889e+02   3.463 0.000671 ***
carbodyhardtop      -4.648e+03  1.413e+03  -3.289 0.001214 ** 
carbodyhatchback    -4.324e+03  1.184e+03  -3.653 0.000341 ***
carbodysedan        -4.105e+03  1.216e+03  -3.375 0.000909 ***
carbodywagon        -5.682e+03  1.374e+03  -4.134 5.50e-05 ***
drivewheelfwd       -6.180e+02  1.017e+03  -0.608 0.544217    
drivewheelrwd        1.498e+03  1.155e+03   1.298 0.196055    
enginelocationrear   8.112e+03  2.135e+03   3.800 0.000199 ***
enginetypedohcv      8.023e+03  2.715e+03   2.955 0.003556 ** 
enginetypel         -4.899e+02  1.281e+03  -0.382 0.702607    
enginetypeohc        2.756e+03  8.703e+02   3.167 0.001816 ** 
enginetypeohcf       2.728e+03  1.581e+03   1.725 0.086325 .  
enginetypeohcv      -3.000e+03  1.144e+03  -2.622 0.009518 ** 
enginetyperotor      1.360e+04  2.739e+03   4.966 1.61e-06 ***
enginesize           1.829e+02  1.122e+01  16.301  < 2e-16 ***
fuelsystem2bbl       3.880e+02  9.239e+02   0.420 0.675060    
fuelsystem4bbl      -1.010e+03  2.987e+03  -0.338 0.735752    
fuelsystemidi        1.904e+04  5.983e+03   3.182 0.001727 ** 
fuelsystemmfi       -2.810e+03  2.766e+03  -1.016 0.311075    
fuelsystemmpfi       7.354e+02  9.752e+02   0.754 0.451802    
fuelsystemspdi      -2.529e+03  1.375e+03  -1.838 0.067685 .  
fuelsystemspfi       9.792e+02  2.663e+03   0.368 0.713494    
boreratio           -5.092e+03  1.188e+03  -4.285 3.00e-05 ***
stroke              -4.016e+03  9.033e+02  -4.446 1.55e-05 ***
compressionratio    -1.372e+03  4.310e+02  -3.184 0.001718 ** 
peakrpm              2.774e+00  5.897e-01   4.704 5.13e-06 ***
car_dim              2.680e-02  5.215e-03   5.139 7.28e-07 ***
avg_mpg              1.483e+02  7.021e+01   2.112 0.036070 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2441 on 176 degrees of freedom
Multiple R-squared:  0.9195,	Adjusted R-squared:  0.9067 
F-statistic: 71.78 on 28 and 176 DF,  p-value: < 2.2e-16
```

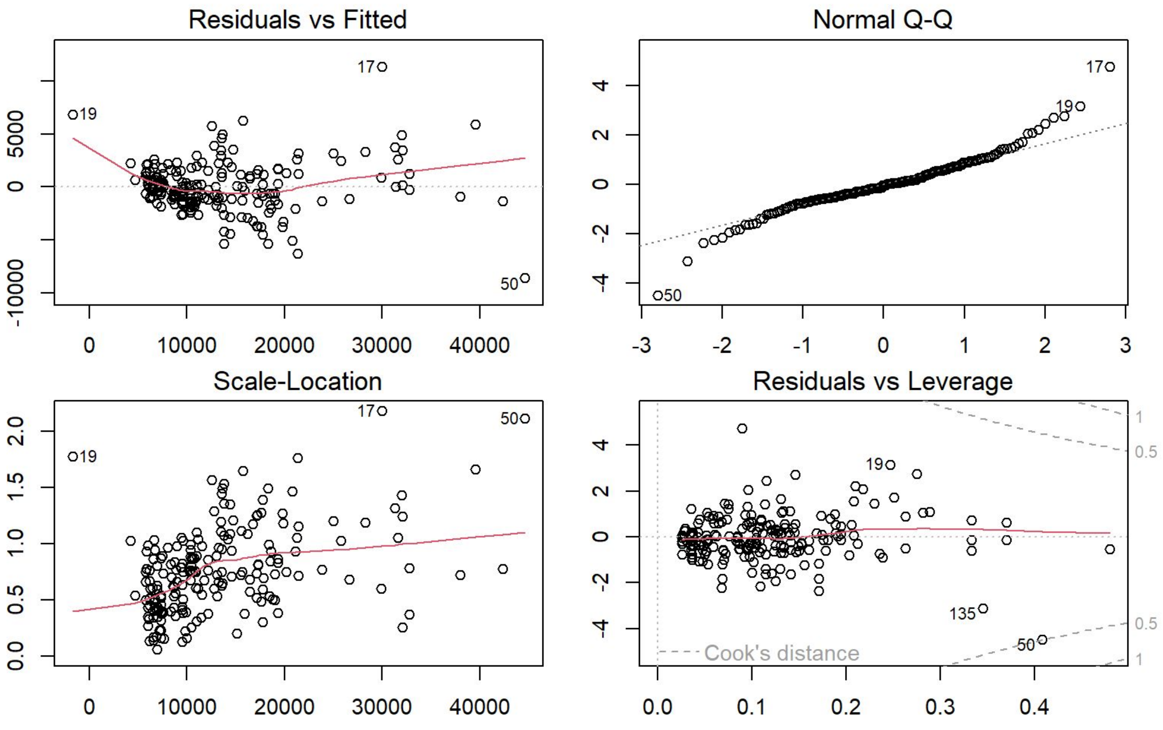
remove compressionratio

```
Residuals:
    Min      1Q  Median      3Q     Max 
-8626.9 -1246.9   -47.2  1226.1 11357.7 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)         -1.330e+04  7.488e+03  -1.776 0.077410 .  
aspirationturbo      3.110e+03  6.667e+02   4.665 6.07e-06 ***
carbodyhardtop      -5.162e+03  1.440e+03  -3.585 0.000435 ***
carbodyhatchback    -4.586e+03  1.211e+03  -3.787 0.000208 ***
carbodysedan        -4.452e+03  1.242e+03  -3.584 0.000437 ***
carbodywagon        -6.112e+03  1.402e+03  -4.358 2.22e-05 ***
drivewheelfwd       -1.086e+03  1.032e+03  -1.052 0.294080    
drivewheelrwd        9.586e+02  1.171e+03   0.818 0.414189    
enginelocationrear   7.429e+03  2.178e+03   3.411 0.000802 ***
enginetypedohcv      7.131e+03  2.769e+03   2.575 0.010840 *  
enginetypel          8.779e+02  1.238e+03   0.709 0.479083    
enginetypeohc        3.132e+03  8.843e+02   3.542 0.000509 ***
enginetypeohcf       3.414e+03  1.607e+03   2.125 0.034975 *  
enginetypeohcv      -2.778e+03  1.171e+03  -2.372 0.018782 *  
enginetyperotor      1.304e+04  2.803e+03   4.651 6.45e-06 ***
enginesize           1.806e+02  1.148e+01  15.730  < 2e-16 ***
fuelsystem2bbl       5.373e+02  9.463e+02   0.568 0.570895    
fuelsystem4bbl      -1.004e+03  3.063e+03  -0.328 0.743516    
fuelsystemidi        4.562e+02  1.349e+03   0.338 0.735637    
fuelsystemmfi       -1.797e+03  2.817e+03  -0.638 0.524383    
fuelsystemmpfi       7.314e+02  1.000e+03   0.731 0.465507    
fuelsystemspdi      -1.372e+03  1.360e+03  -1.009 0.314439    
fuelsystemspfi       5.905e+02  2.728e+03   0.216 0.828870    
boreratio           -5.079e+03  1.219e+03  -4.167 4.81e-05 ***
stroke              -3.154e+03  8.837e+02  -3.568 0.000462 ***
peakrpm              2.500e+00  5.982e-01   4.179 4.60e-05 ***
car_dim              2.707e-02  5.347e-03   5.063 1.03e-06 ***
avg_mpg              8.322e+01  6.888e+01   1.208 0.228607    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2503 on 177 degrees of freedom
Multiple R-squared:  0.9148,	Adjusted R-squared:  0.9018 
F-statistic: 70.42 on 27 and 177 DF,  p-value: < 2.2e-16
```

DIAGNOSTICS



REFERENCES

1. Simon J. Sheather, *A Modern Approach to Regression with R,* Chapter 2, Springer
2. Kuhn Max, Jonson Kjell, *Applied Predictive Modelling,* Chapter 6, Springer
3. Hyndman, Rob J., and George Athanasopoulos. *Forecasting: Principles and Practice ;* Otexts, 2018.
4. Car Price Data: https://www.kaggle.com/datasets/imgowthamg/car-price