Untitled

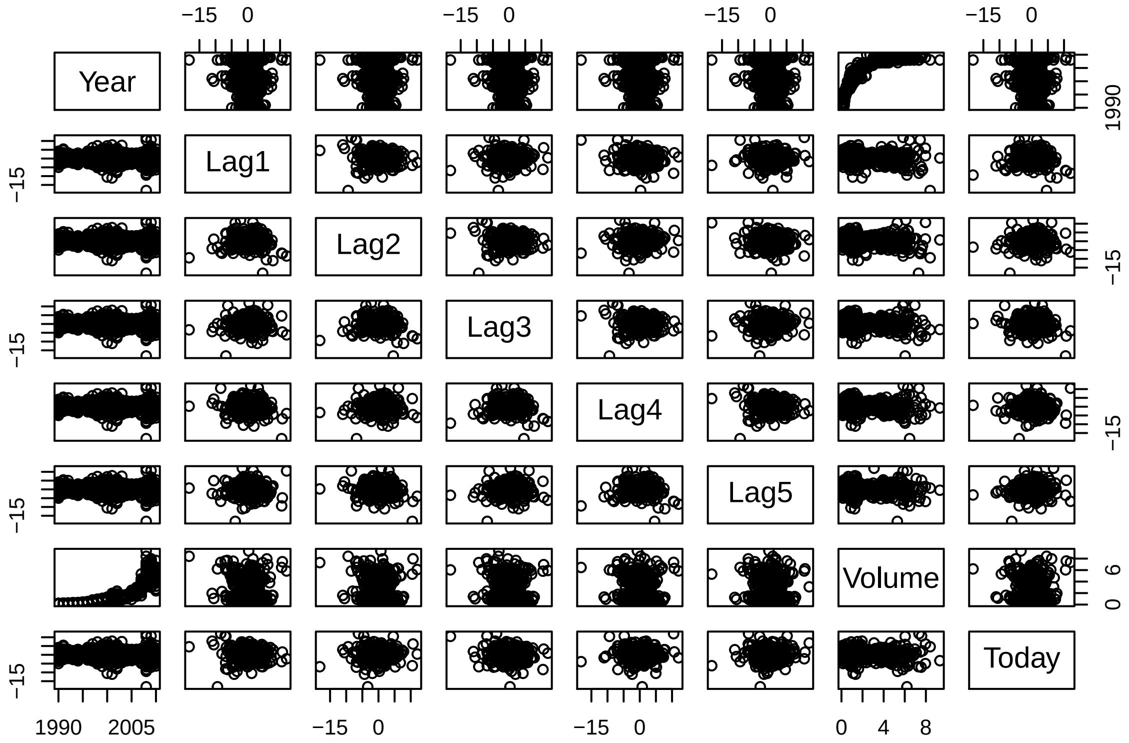
Rohitashwa Chakraborty

30/07/2021

EXERCISE 4.10:

4.10.a

```
##       Year           Lag1               Lag2               Lag3         
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4               Lag5              Volume            Today         
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260  
##  Direction 
##  Down:484  
##  Up  :605  
##            
##            
##            
## 
```



Positive Correlation between `Year` and `Volume` observed.

4.10.b

```
##
## Call:
## glm(formula = Direction ~ ., family = binomial, data = Weekly[,
##     c(2:7, 9)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

To check if a parameter is significant or not, we must check for its **P-Vaue**.

From the Summary, only Lag2 has a **P-Value** < 0.05. Thus, only Lag2 is statistically significant.

Part c)

```
##           Reference
## Prediction Down  Up
##       Down   54  48
##       Up    430 557
```

```
## Accuracy            :  56.11 %
```

```
## Recall/Sensitivity  :  92.07 %
```

```
## Precision           :  56.43 %
```

```
## Specificity         :  11.16 %
```

```
## Up Prediction Rate  :  56.43 %
```

```
## Down Prediction Rate:  52.94 %
```

48 *"Up"* were mistaken for *"Down"*. 430 *"Down"* were mistaken for *"Up"*. 54 *"Down"*+ 557 *"Up"* were predicted accurately . Model is has higher accuracy when the prediction is *"Up"*

These results were obtained from the same set of observations the model was trained upon. Therefore, it is highly likely that the results would prove to be *overly optimistic* when tested on a new set of data.

Part d)

```
##           Reference
## Prediction Down Up
##       Down    9  5
##       Up     34 56
```

```
## [Logistic Regression] Overall Fraction of Correct Predictions (Accuracy) :  0.62
```

Part g)

```
##           Reference
## Prediction Down Up
##       Down   21 30
##       Up     22 31
```

```
## [KNN (k = 1)] Overall Fraction of Correct Predictions (Accuracy) :  0.5
```

Part h)

Considering **only Accuracy** as our metric, we can conclude that *Logistic Regression* outperforms *KNN (with k = 1)*

Part i)

Experimenting with different KNN models:

```
## Predictors: Lag2

## [KNN (k =  30 )] Accuracy :  0.53
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2, Lag1

## [KNN (k =  30 )] Accuracy :  0.54
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2^2

## [KNN (k =  30 )] Accuracy :  0.62
## [KNN (k =  130 )] Accuracy :  0.62
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2*Lag1

## [KNN (k =  30 )] Accuracy :  0.55
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.57
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: All

## [KNN (k =  30 )] Accuracy :  0.89
## [KNN (k =  130 )] Accuracy :  0.86
## [KNN (k =  230 )] Accuracy :  0.79
## [KNN (k =  330 )] Accuracy :  0.75
```

Considering only **Accuracy**, we can conclude that the following models perform the best:

K= 30, Predictors: All Predictors

```
preds <- knn(as.matrix(train[,-9]),
               as.matrix(test[,-9]),
               train$Direction, k=30)
cm <- confusionMatrix(preds, Direction[!filtered_years], positive = "Up")
cat("Confusion Matrix for K= 130, Predictors: All\n")
cm$table
```

Experimenting with different Logistic Regression Models:

```
## Logistic Regression
```

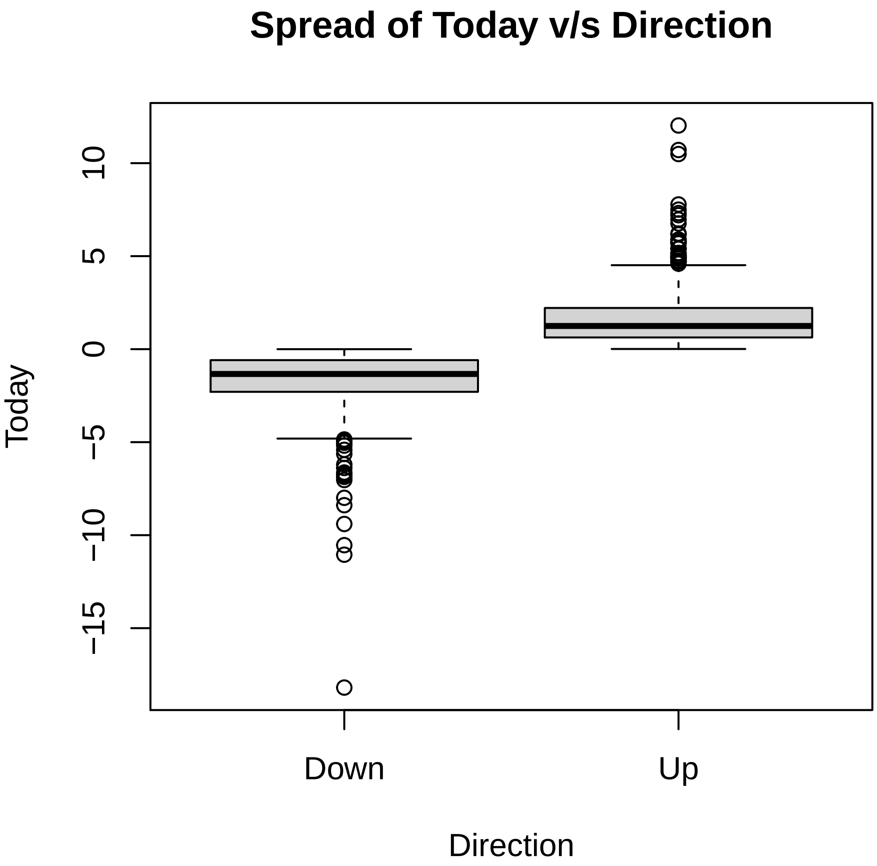
```
##
## [Predictors: Lag2 ] Accuracy :  0.62
## [Predictors: Lag2+Lag1 ] Accuracy :  0.58
## [Predictors: Lag2*Lag1 ] Accuracy :  0.58
## [Predictors: I(Lag2^2) ] Accuracy :  0.59
## [Predictors: All] Accuracy :  1
```

Considering Accuracy, It seems Using **All the Parameters** gives by far the most accurate model with a **100% Accuracy**.

```
## Confusion Matrix for Linear Regression Model with All Predictors:
```

```
##             Reference
## Prediction Down Up
##       Down   43  0
##       Up      0 61
```

NOTE: This is not surprising because one of the predictors the model trains upon is ***Today****. This predictor seems to have a distinct linear boundary when plotted against* ***Direction***



EXERCISE 6.9:

6.9.a

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

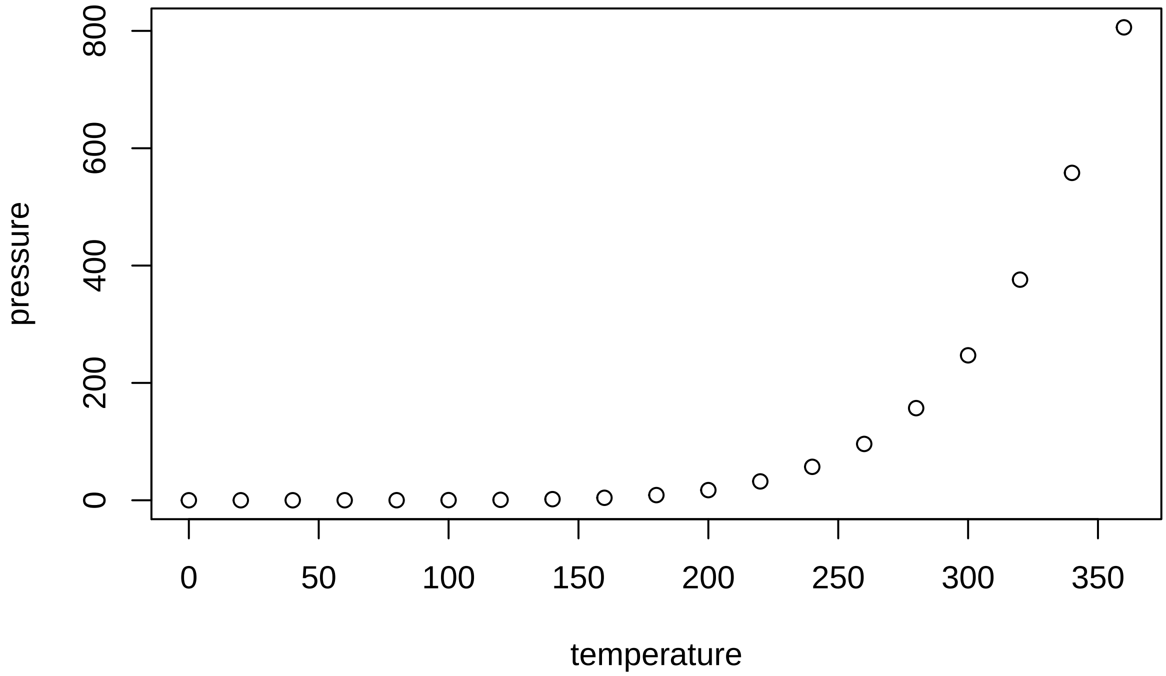
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.