HW4

賴冠維

2020/11/16

#### Question 10

## Loading required package: ISLR

## Loading required package: tidyverse

## -- Attaching packages ------------------------------------------------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.2 √ purrr 0.3.4  
## √ tibble 3.0.3 √ dplyr 1.0.2  
## √ tidyr 1.1.2 √ stringr 1.4.0  
## √ readr 1.3.1 √ forcats 0.5.0

## -- Conflicts ---------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

## Loading required package: ggthemes

## Warning: package 'ggthemes' was built under R version 4.0.3

## Loading required package: GGally

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

#### (a)

Weekly為S&P500指數從1990到2010的周報酬率資料， 資料組成有:  
1. (年份)  
2. (滯後1-5期的報酬率資料)

3. (成交量)

4. (當日報酬率)

5. (當天是漲/跌)

## 'data.frame': 1089 obs. of 9 variables:  
## $ Year : num 1990 1990 1990 1990 1990 1990 1990 1990 1990 1990 ...  
## $ Lag1 : num 0.816 -0.27 -2.576 3.514 0.712 ...  
## $ Lag2 : num 1.572 0.816 -0.27 -2.576 3.514 ...  
## $ Lag3 : num -3.936 1.572 0.816 -0.27 -2.576 ...  
## $ Lag4 : num -0.229 -3.936 1.572 0.816 -0.27 ...  
## $ Lag5 : num -3.484 -0.229 -3.936 1.572 0.816 ...  
## $ Volume : num 0.155 0.149 0.16 0.162 0.154 ...  
## $ Today : num -0.27 -2.576 3.514 0.712 1.178 ...  
## $ Direction: Factor w/ 2 levels "Down","Up": 1 1 2 2 2 1 2 2 2 1 ...

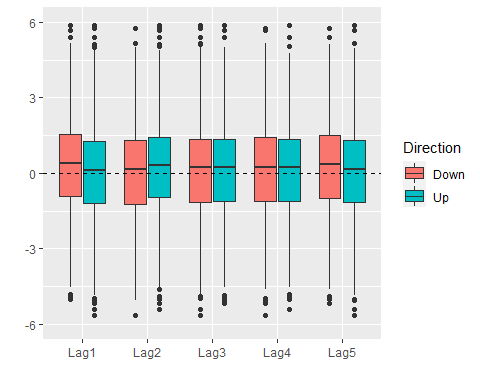
列出不同Lag期之下對應本日漲跌的幅度，單純從數字上看不太出有什麼關係

## `summarise()` regrouping output by 'Variable' (override with `.groups` argument)

## # A tibble: 12 x 6  
## # Groups: Variable [6]  
## Variable Direction Q25 median mean Q75  
## <chr> <fct> <dbl> <dbl> <dbl> <dbl>  
## 1 Lag1 Down -0.937 0.382 0.282 1.59   
## 2 Lag1 Up -1.24 0.099 0.0452 1.31   
## 3 Lag2 Down -1.31 0.154 -0.0404 1.30   
## 4 Lag2 Up -1.00 0.299 0.304 1.46   
## 5 Lag3 Down -1.15 0.250 0.208 1.41   
## 6 Lag3 Up -1.17 0.224 0.0989 1.42   
## 7 Lag4 Down -1.15 0.224 0.200 1.44   
## 8 Lag4 Up -1.16 0.241 0.102 1.35   
## 9 Lag5 Down -1.09 0.328 0.188 1.50   
## 10 Lag5 Up -1.20 0.128 0.102 1.34   
## 11 Today Down -2.29 -1.33 -1.75 -0.592  
## 12 Today Up 0.63 1.25 1.67 2.22

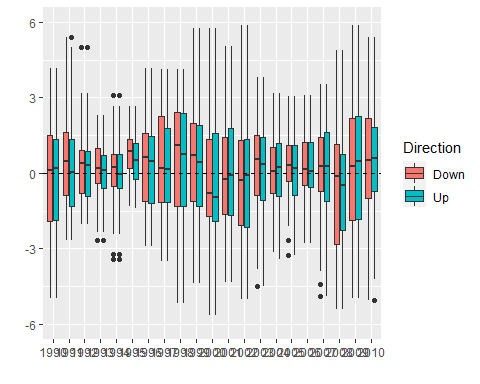
畫出Box Plot之後可以觀察到Lag1、Lag2、Lag5之下，Down跟Up之間盒狀圖有顯著的差異

## Warning: Removed 125 rows containing non-finite values (stat\_boxplot).



若是對Year畫出盒狀圖，可以看到S&P500報酬的波動有群聚的現象， 1992-1995為波動較小的時期，而1996到2002波動較大，對應到當時正面臨網際網路泡沫的衝擊。

## Warning: Removed 125 rows containing non-finite values (stat\_boxplot).



分別對Lag1、Lag2進行Two Sample t-test， 在90%信心水準下，拒絕虛無假設，代表不同Direction之下的Lag1、Lag2間存在差異。

##   
## Welch Two Sample t-test  
##   
## data: Lag1 by Direction  
## t = 1.6563, df = 1047.9, p-value = 0.09795  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -0.04378476 0.51794261  
## sample estimates:  
## mean in group Down mean in group Up   
## 0.28229545 0.04521653

##   
## Welch Two Sample t-test  
##   
## data: Lag2 by Direction  
## t = -2.4154, df = 1053.6, p-value = 0.01589  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -0.62473558 -0.06467351  
## sample estimates:  
## mean in group Down mean in group Up   
## -0.04042355 0.30428099

#### (b)

去掉Year,Today變數後，因Outcome有兩個結果，family使用binomial，為Logistic Regression。 由配飾結果可見，僅Lag2與截距項顯著拒絕虛無假設，通過個別t檢定，故在此認為僅Lag2為較有解釋力之變數。

##   
## Call:  
## glm(formula = Direction ~ . - Year - Today, family = "binomial",   
## data = Weekly)  
##   
## 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

#### (c)

由下表可見準確率(Accuracy)僅56.11%，下表視為的情況下，Sensitivity 雖高達92%， 但Specificity僅11.16%，代表配飾的模型將絕大部分的資料都判斷為，並不能有效區別及。

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 557 430  
## Down 48 54  
##   
## Accuracy : 0.5611   
## 95% CI : (0.531, 0.5908)  
## No Information Rate : 0.5556   
## P-Value [Acc > NIR] : 0.369   
##   
## Kappa : 0.035   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.9207   
## Specificity : 0.1116   
## Pos Pred Value : 0.5643   
## Neg Pred Value : 0.5294   
## Prevalence : 0.5556   
## Detection Rate : 0.5115   
## Detection Prevalence : 0.9063   
## Balanced Accuracy : 0.5161   
##   
## 'Positive' Class : Up   
##

此處試驗全部都猜準確率也有55.56%，代表上述模型配飾結果很差，跟全部猜差不多。

## [1] 0.5555556

#### (d)

因為此資料為時間序列的資料，因此在拆分Train、Test時不能像一般Cross-Section的資料隨機抽樣， 因此按照資料在2008年之前/後分為Train、Test，並且只放入通過個別t檢定的變數:。  
以Train 資料配飾的Logistic Regression在配飾Test資料所得到的Confusion Matrix來看， 看似準確率有提升至62.5%，但若是全部猜之下也有58%的準確度，該模型依舊無顯著的預測能力。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 56 34  
## Down 5 9  
##   
## Accuracy : 0.625   
## 95% CI : (0.5247, 0.718)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.2439   
##   
## Kappa : 0.1414   
##   
## Mcnemar's Test P-Value : 7.34e-06   
##   
## Sensitivity : 0.9180   
## Specificity : 0.2093   
## Pos Pred Value : 0.6222   
## Neg Pred Value : 0.6429   
## Prevalence : 0.5865   
## Detection Rate : 0.5385   
## Detection Prevalence : 0.8654   
## Balanced Accuracy : 0.5637   
##   
## 'Positive' Class : Up   
##

## [1] 0.5865385

#### (e)

使用LDA方法配飾預測模型，同樣僅放入，發現準確率與Logisitc相同,並無提升。

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 56 34  
## Down 5 9  
##   
## Accuracy : 0.625   
## 95% CI : (0.5247, 0.718)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.2439   
##   
## Kappa : 0.1414   
##   
## Mcnemar's Test P-Value : 7.34e-06   
##   
## Sensitivity : 0.9180   
## Specificity : 0.2093   
## Pos Pred Value : 0.6222   
## Neg Pred Value : 0.6429   
## Prevalence : 0.5865   
## Detection Rate : 0.5385   
## Detection Prevalence : 0.8654   
## Balanced Accuracy : 0.5637   
##   
## 'Positive' Class : Up   
##

#### (f)

使用QDA來預測之Confusion Matrix，可得模型判所有的Test資料皆為，無預測能力。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 61 43  
## Down 0 0  
##   
## Accuracy : 0.5865   
## 95% CI : (0.4858, 0.6823)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.5419   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 1.504e-10   
##   
## Sensitivity : 1.0000   
## Specificity : 0.0000   
## Pos Pred Value : 0.5865   
## Neg Pred Value : NaN   
## Prevalence : 0.5865   
## Detection Rate : 0.5865   
## Detection Prevalence : 1.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : Up   
##

#### (g)

使用KNN演算法預測Test資料，我們需要先給定center有幾個，若我們設定center=1，以Train進行配飾，並對Test資料預測所建立的Confusion Matrix，準確率僅50.82%，比全部猜還要更低。

## Loading required package: class

## Confusion Matrix and Statistics  
##   
##   
## knn\_pred Up Down  
## Up 31 22  
## Down 30 21  
##   
## Accuracy : 0.5   
## 95% CI : (0.4003, 0.5997)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.9700   
##   
## Kappa : -0.0033   
##   
## Mcnemar's Test P-Value : 0.3317   
##   
## Sensitivity : 0.5082   
## Specificity : 0.4884   
## Pos Pred Value : 0.5849   
## Neg Pred Value : 0.4118   
## Prevalence : 0.5865   
## Detection Rate : 0.2981   
## Detection Prevalence : 0.5096   
## Balanced Accuracy : 0.4983   
##   
## 'Positive' Class : Up   
##

使用Naive Bayes來預測之Confusion Matrix，可得模型判所有的Test資料皆為，無預測能力。

## Loading required package: e1071

## Confusion Matrix and Statistics  
##   
##   
## pred Up Down  
## Up 61 43  
## Down 0 0  
##   
## Accuracy : 0.5865   
## 95% CI : (0.4858, 0.6823)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.5419   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 1.504e-10   
##   
## Sensitivity : 1.0000   
## Specificity : 0.0000   
## Pos Pred Value : 0.5865   
## Neg Pred Value : NaN   
## Prevalence : 0.5865   
## Detection Rate : 0.5865   
## Detection Prevalence : 1.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : Up   
##

#### (h)

若是僅看Accuracy之下，可能會選擇Logistic Regression或是LDA，但是我們可觀察到KNN在Center=1時，模型預測Test資料為大量出現，也讓 Specificity 明顯提升，故我們可試試看藉由調整Center，優化KNN的結果。

#### (i)

將Lag1、I(Volume^2)加進自變數進行Logistic Regression，可發現Accuracy略下降，但是Specificity 大幅提升，因此認為是較佳的模型。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 30 16  
## Down 31 27  
##   
## Accuracy : 0.5481   
## 95% CI : (0.4474, 0.6459)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.81516   
##   
## Kappa : 0.1139   
##   
## Mcnemar's Test P-Value : 0.04114   
##   
## Sensitivity : 0.4918   
## Specificity : 0.6279   
## Pos Pred Value : 0.6522   
## Neg Pred Value : 0.4655   
## Prevalence : 0.5865   
## Detection Rate : 0.2885   
## Detection Prevalence : 0.4423   
## Balanced Accuracy : 0.5599   
##   
## 'Positive' Class : Up   
##

接著以相同的自變數帶入LDA模型，兩者結果相近。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values Up Down  
## Up 32 17  
## Down 29 26  
##   
## Accuracy : 0.5577   
## 95% CI : (0.457, 0.655)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.7579   
##   
## Kappa : 0.1241   
##   
## Mcnemar's Test P-Value : 0.1048   
##   
## Sensitivity : 0.5246   
## Specificity : 0.6047   
## Pos Pred Value : 0.6531   
## Neg Pred Value : 0.4727   
## Prevalence : 0.5865   
## Detection Rate : 0.3077   
## Detection Prevalence : 0.4712   
## Balanced Accuracy : 0.5646   
##   
## 'Positive' Class : Up   
##

藉由測試Center:1-14之下的模型表現，選出Accuracy最高者，可發現在k=13之下有最高的Accuracy。

## [1] 0.5096154 0.5576923 0.5480769 0.5576923 0.5384615 0.5288462 0.5384615  
## [8] 0.5288462 0.5480769 0.5480769 0.5673077 0.5961538 0.5961538 0.5673077  
## [15] 0.5865385 0.5384615

發現KNN在k=13之下，Accuracy比上面兩模型表現更佳。

## Confusion Matrix and Statistics  
##   
##   
## knn\_pred Up Down  
## Up 40 23  
## Down 21 20  
##   
## Accuracy : 0.5769   
## 95% CI : (0.4761, 0.6732)  
## No Information Rate : 0.5865   
## P-Value [Acc > NIR] : 0.6193   
##   
## Kappa : 0.1217   
##   
## Mcnemar's Test P-Value : 0.8802   
##   
## Sensitivity : 0.6557   
## Specificity : 0.4651   
## Pos Pred Value : 0.6349   
## Neg Pred Value : 0.4878   
## Prevalence : 0.5865   
## Detection Rate : 0.3846   
## Detection Prevalence : 0.6058   
## Balanced Accuracy : 0.5604   
##   
## 'Positive' Class : Up   
##

### Question 11

mpg資料變數介紹：

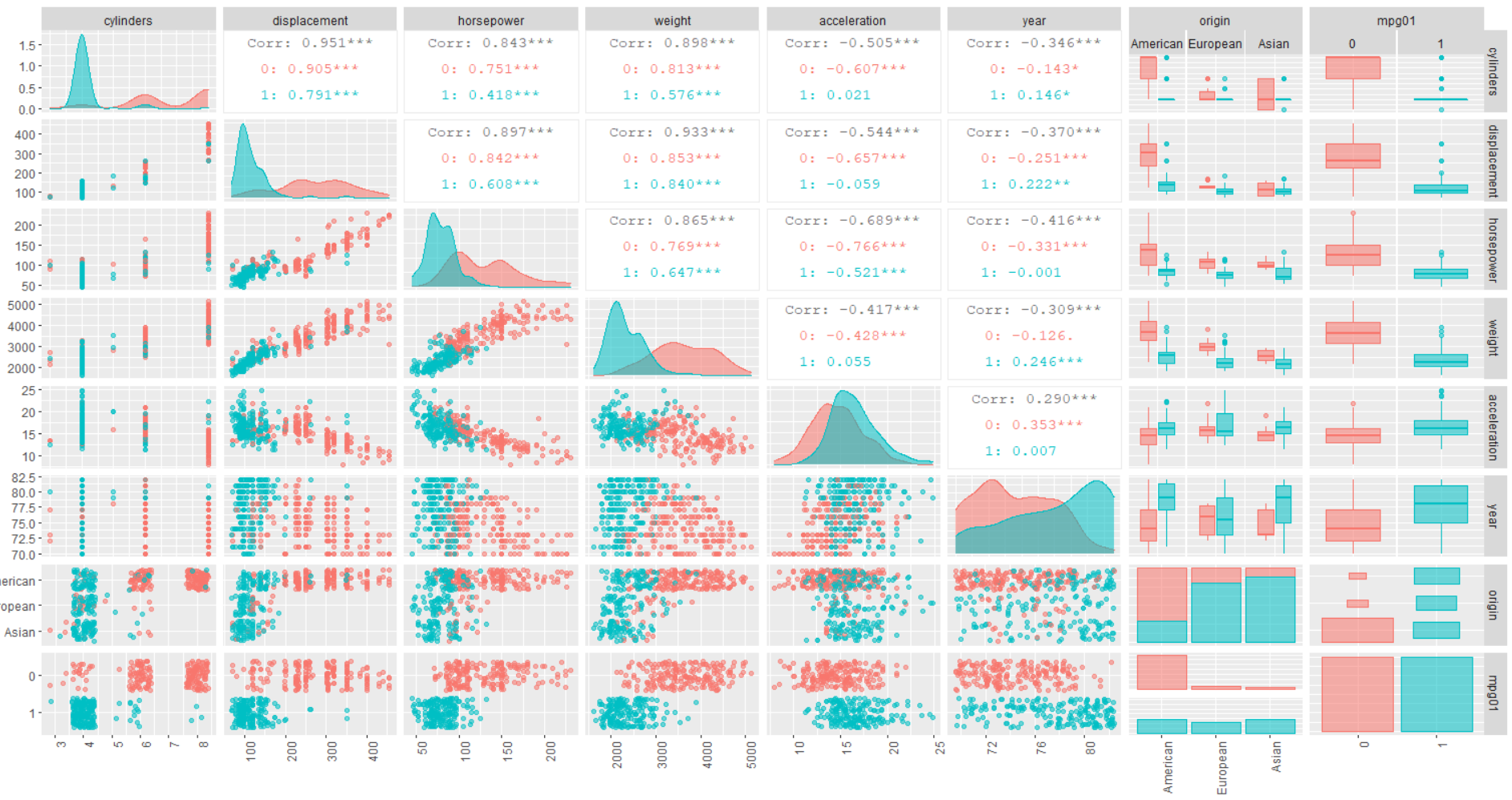
* mpg:miles per gallon
* cylinders:Number of cylinders between 4 and 8
* displacement:Engine displacement (cu. inches)
* horsepower:Engine horsepower
* weight:Vehicle weight (lbs.)
* acceleration:Time to accelerate from 0 to 60 mph (sec.)
* year:Model year (modulo 100)
* origin:Origin of car (1. American, 2. European, 3. Japanese)
* name:Vehicle name

#### (a)

建立mpg01，將mpg大於中位數令為1，否則為0，並且將origin的Outcome改為

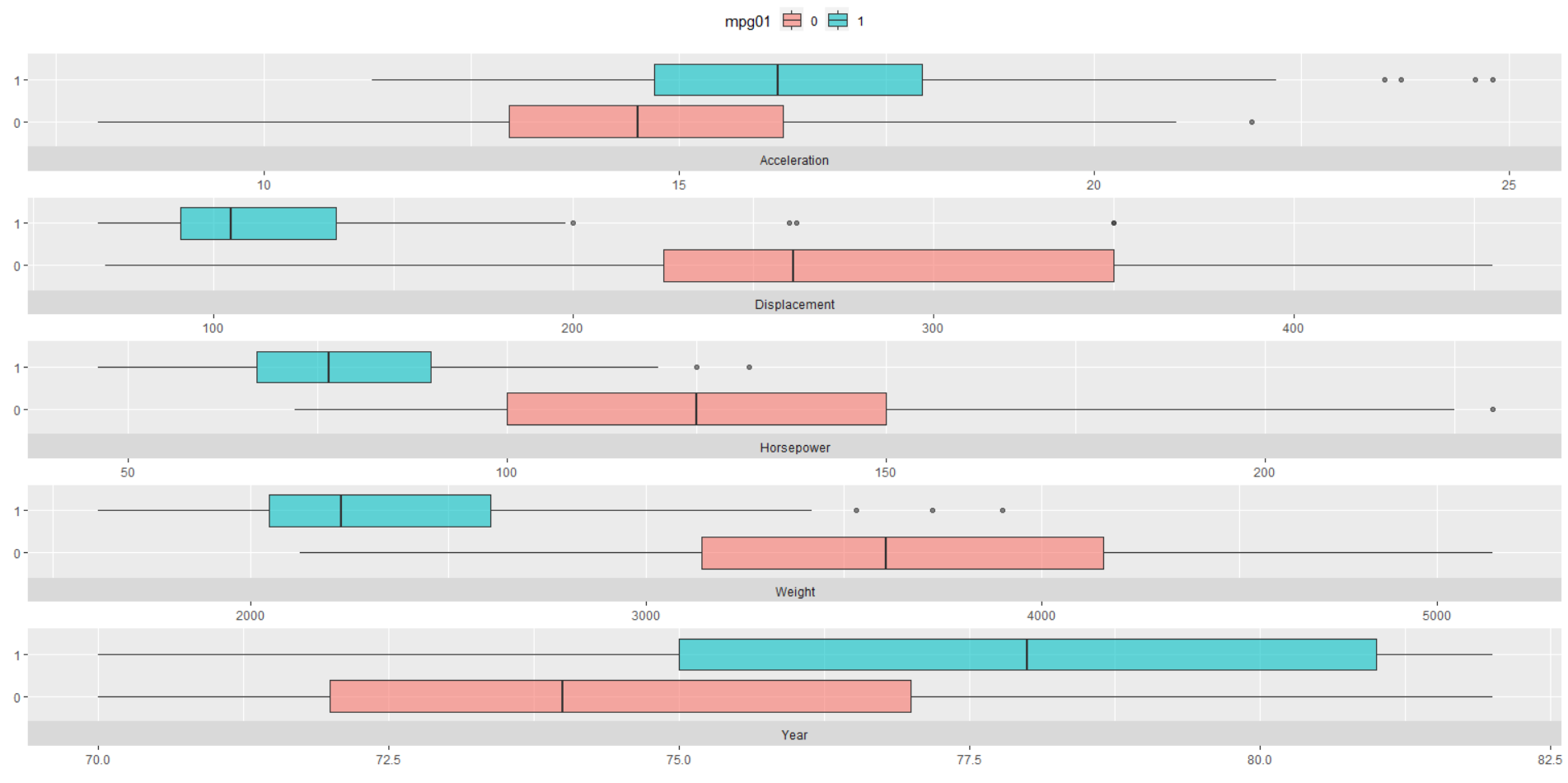
#### (b)

從下圖觀察，發現有以下這些變數對mpg01有較顯著的變化，可能代表著較有解釋力，變數如下：  
cylinders 、displacement 、 horsepower 、 weight 、 year



個別將這些變數對mpg1做Box Plot，更可以觀察到這些變數對mpg01有顯著的不一樣，可能代表著具有較佳的解釋力。

## Warning: 'switch' is deprecated.  
## Use 'strip.position' instead.  
## See help("Deprecated")



#### (c)

以80:20，將資料分成Train、Test

set.seed(1234)

num\_train <- nrow(Auto) \* 0.8

inTrain <- sample(nrow(Auto), size = num\_train)

train <- Auto[inTrain,]

test <- Auto[-inTrain,]

#### (d)

使用LDA模型，以Train資料配飾，預測Test資料，觀察所得之Confusion Matrix，準確率為88.61%，代表模型表現不錯。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values 1 0  
## 1 38 7  
## 0 2 32  
##   
## Accuracy : 0.8861   
## 95% CI : (0.7947, 0.9466)  
## No Information Rate : 0.5063   
## P-Value [Acc > NIR] : 8.396e-13   
##   
## Kappa : 0.7717   
##   
## Mcnemar's Test P-Value : 0.1824   
##   
## Sensitivity : 0.9500   
## Specificity : 0.8205   
## Pos Pred Value : 0.8444   
## Neg Pred Value : 0.9412   
## Prevalence : 0.5063   
## Detection Rate : 0.4810   
## Detection Prevalence : 0.5696   
## Balanced Accuracy : 0.8853   
##   
## 'Positive' Class : 1   
##

我們可以發現在4缸的歐洲車以及六缸的美國車佔錯誤的大宗，若以廠牌來看，Ford判斷錯誤出現次數最多，並且大部分都是mpg01為0代表實際是油耗較差的那群，可能顯示出Ford的造車可能存在與其他車廠之間的落差。

## mpg cylinders displacement horsepower weight acceleration year origin  
## 21 25.0 4 110 87 2672 17.5 70 European  
## 119 20.0 4 114 91 2582 14.0 73 European  
## 120 19.0 4 121 112 2868 15.5 73 European  
## 192 24.0 6 200 81 3012 17.6 76 American  
## 269 21.1 4 134 95 2515 14.8 78 Asian  
## 281 22.3 4 140 88 2890 17.3 79 American  
## 359 22.4 6 231 110 3415 15.8 81 American  
## 361 20.2 6 200 88 3060 17.1 81 American  
## 384 22.0 6 232 112 2835 14.7 82 American  
## name mpg01  
## 21 peugeot 504 1  
## 119 audi 100ls 0  
## 120 volvo 144ea 0  
## 192 ford maverick 1  
## 269 toyota celica gt liftback 0  
## 281 ford fairmont 4 0  
## 359 buick century 0  
## 361 ford granada gl 0  
## 384 ford granada l 0

#### (e)

使用QDA模型，以Train資料配飾，預測Test資料，觀察所得之Confusion Matrix，準確率為87.34%，表現略差於LDA。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values 1 0  
## 1 36 6  
## 0 4 33  
##   
## Accuracy : 0.8734   
## 95% CI : (0.7795, 0.9376)  
## No Information Rate : 0.5063   
## P-Value [Acc > NIR] : 5.838e-12   
##   
## Kappa : 0.7466   
##   
## Mcnemar's Test P-Value : 0.7518   
##   
## Sensitivity : 0.9000   
## Specificity : 0.8462   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.8919   
## Prevalence : 0.5063   
## Detection Rate : 0.4557   
## Detection Prevalence : 0.5316   
## Balanced Accuracy : 0.8731   
##   
## 'Positive' Class : 1   
##

使用QDA也有相似於LDA的結果，判斷錯誤的汽缸數皆是4,6缸，而不同的是此結果亞洲地區的車判斷錯誤比例上升。

## mpg cylinders displacement horsepower weight acceleration year origin  
## 119 20.0 4 114 91 2582 14.0 73 European  
## 192 24.0 6 200 81 3012 17.6 76 American  
## 269 21.1 4 134 95 2515 14.8 78 Asian  
## 281 22.3 4 140 88 2890 17.3 79 American  
## 331 32.7 6 168 132 2910 11.4 80 Asian  
## 357 25.4 6 168 116 2900 12.6 81 Asian  
## 358 24.2 6 146 120 2930 13.8 81 Asian  
## 359 22.4 6 231 110 3415 15.8 81 American  
## 361 20.2 6 200 88 3060 17.1 81 American  
## 384 22.0 6 232 112 2835 14.7 82 American  
## name mpg01  
## 119 audi 100ls 0  
## 192 ford maverick 1  
## 269 toyota celica gt liftback 0  
## 281 ford fairmont 4 0  
## 331 datsun 280-zx 1  
## 357 toyota cressida 1  
## 358 datsun 810 maxima 1  
## 359 buick century 0  
## 361 ford granada gl 0  
## 384 ford granada l 0

#### (f)

使用Logistic Regression模型，以Train資料配飾，預測Test資料，觀察所得之Confusion Matrix，準確率為87.34%，模型表現略差於LDA。

## Confusion Matrix and Statistics  
##   
##   
## pred\_values 1 0  
## 1 36 6  
## 0 4 33  
##   
## Accuracy : 0.8734   
## 95% CI : (0.7795, 0.9376)  
## No Information Rate : 0.5063   
## P-Value [Acc > NIR] : 5.838e-12   
##   
## Kappa : 0.7466   
##   
## Mcnemar's Test P-Value : 0.7518   
##   
## Sensitivity : 0.9000   
## Specificity : 0.8462   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.8919   
## Prevalence : 0.5063   
## Detection Rate : 0.4557   
## Detection Prevalence : 0.5316   
## Balanced Accuracy : 0.8731   
##   
## 'Positive' Class : 1   
##

使用Logistic Regression也有相似於QDA的結果，判斷錯誤的汽缸數皆是4,6缸，亞洲地區的車判斷錯誤比例上升。

## mpg cylinders displacement horsepower weight acceleration year origin  
## 119 20.0 4 114 91 2582 14.0 73 European  
## 192 24.0 6 200 81 3012 17.6 76 American  
## 269 21.1 4 134 95 2515 14.8 78 Asian  
## 281 22.3 4 140 88 2890 17.3 79 American  
## 331 32.7 6 168 132 2910 11.4 80 Asian  
## 357 25.4 6 168 116 2900 12.6 81 Asian  
## 358 24.2 6 146 120 2930 13.8 81 Asian  
## 359 22.4 6 231 110 3415 15.8 81 American  
## 361 20.2 6 200 88 3060 17.1 81 American  
## 384 22.0 6 232 112 2835 14.7 82 American  
## name mpg01  
## 119 audi 100ls 0  
## 192 ford maverick 1  
## 269 toyota celica gt liftback 0  
## 281 ford fairmont 4 0  
## 331 datsun 280-zx 1  
## 357 toyota cressida 1  
## 358 datsun 810 maxima 1  
## 359 buick century 0  
## 361 ford granada gl 0  
## 384 ford granada l 0

#### (g)

將(b)裡所提出較可能較有解釋力的變數帶進KNN，並測試KNN的Center從1-15，可得到在c1=5,7的地方有最佳的Accuracy。

## [1] 0.8101266 0.8101266 0.8860759 0.8734177 0.9113924 0.8987342 0.9113924  
## [8] 0.8860759 0.8734177 0.8734177 0.8734177 0.8860759 0.8860759 0.8860759  
## [15] 0.8734177

最後使用k=5，為所有模型裡面表現最佳

## Confusion Matrix and Statistics  
##   
##   
## knn\_pred 1 0  
## 1 37 4  
## 0 3 35  
##   
## Accuracy : 0.9114   
## 95% CI : (0.8259, 0.9636)  
## No Information Rate : 0.5063   
## P-Value [Acc > NIR] : 1.201e-14   
##   
## Kappa : 0.8227   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9250   
## Specificity : 0.8974   
## Pos Pred Value : 0.9024   
## Neg Pred Value : 0.9211   
## Prevalence : 0.5063   
## Detection Rate : 0.4684   
## Detection Prevalence : 0.5190   
## Balanced Accuracy : 0.9112   
##   
## 'Positive' Class : 1   
##

在判斷錯誤的車裡面，大多屬於美國車，並且同樣為4,6缸。

## mpg cylinders displacement horsepower weight acceleration year origin  
## 18 21.0 6 200 85 2587 16.0 70 American  
## 113 21.0 6 155 107 2472 14.0 73 American  
## 119 20.0 4 114 91 2582 14.0 73 European  
## 192 24.0 6 200 81 3012 17.6 76 American  
## 269 21.1 4 134 95 2515 14.8 78 Asian  
## 271 23.8 4 151 85 2855 17.6 78 American  
## 358 24.2 6 146 120 2930 13.8 81 Asian  
## name mpg01  
## 18 ford maverick 0  
## 113 mercury capri v6 0  
## 119 audi 100ls 0  
## 192 ford maverick 1  
## 269 toyota celica gt liftback 0  
## 271 oldsmobile starfire sx 1  
## 358 datsun 810 maxima 1

附錄(程式碼)：

#### Question 10

require(ISLR); require(tidyverse); require(ggthemes);

require(GGally);

#### (a)

set.seed(1)

data('Weekly')

str(Weekly)

Weekly %>%

gather(Variable, value, starts\_with('Lag'), Today) %>%

group\_by(Variable, Direction) %>%

summarise(Q25 = quantile(value, 0.25),

median = median(value),

mean = mean(value),

Q75 = quantile(value, 0.75))

Weekly %>%

gather(value\_type, value, starts\_with('Lag')) %>%

ggplot(aes(value\_type, value, fill = Direction)) +

geom\_boxplot(notch = F) +

labs(x = '', y = '') +

ylim(c(-6, 6)) +

geom\_hline(yintercept = 0, linetype = 2)

Weekly %>%

gather(value\_type, value, starts\_with('Lag')) %>%

ggplot(aes(as.factor(Year), value, fill = Direction)) +

geom\_boxplot(notch = F) +

labs(x = '', y = '') +

ylim(c(-6,6)) +

geom\_hline(yintercept = 0, linetype = 2)

t.test(Lag1 ~ Direction, data = Weekly)

t.test(Lag2 ~ Direction, data = Weekly)

#### (b)

Log\_ful <- glm(Direction ~ . - Year - Today, data = Weekly, family = 'binomial')

summary(Log\_ful)

#### (c)

pred <- predict(Log\_ful, type = 'response')

pred\_values <- ifelse(pred >= 0.5, 'Up', 'Down')

library(caret)

xtab <- table(pred\_values,Weekly$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

mean(Weekly$Direction == 'Up')

#### (d)

train <- Weekly[Weekly$Year <= 2008,]

test <- Weekly[Weekly$Year > 2008,]

lag2\_logreg <- glm(Direction ~ Lag2, data = train, family = 'binomial')

pred <- predict(lag2\_logreg, newdata = test, type = 'response')

pred\_values <- ifelse(pred >= 0.5, 'Up', 'Down')

xtab <- table(pred\_values,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

mean(test$Direction == 'Up')

#### (e)

require(MASS)

lda\_model <- lda(Direction ~ Lag2, data = train)

pred <- predict(lda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

#### (f)

qda\_model <- qda(Direction ~ Lag2, data = train)

pred <- predict(qda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

#### (g)

require(class)

knn\_pred <- knn(train = data.frame(train$Lag2),

test = data.frame(test$Lag2),

cl = train$Direction, k = 1)

xtab <- table(knn\_pred,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

require(e1071)

NB = naiveBayes(Direction ~Lag2, data = train)

pred <- predict(NB, newdata = test)

xtab <- table(pred,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

#### (h)

#### (i)

lag2\_logreg <- glm(Direction~Lag1+Lag2+I(Volume^2), data = train,family = 'binomial')

pred <- predict(lag2\_logreg, newdata = test, type = 'response')

pred\_values <- ifelse(pred >= 0.5, 'Up', 'Down')

xtab <- table(pred\_values,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

require(MASS)

lda\_model <- lda(Direction ~Lag1+Lag2+I(Volume^2), data = train)

pred <- predict(lda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

acc <- list()

set.seed(12345)

acc = sapply(1:16, function(x){

knn\_pred <- knn(train = data.frame(train$Lag2),

test = data.frame(test$Lag2),

cl = train$Direction, k = x)

acc[as.character(x)] = mean(knn\_pred == test$Direction)

})

unlist(acc)

knn\_pred <- knn(train = data.frame(train$Lag2),

test = data.frame(test$Lag2),

cl = train$Direction, k = 13)

xtab <- table(knn\_pred,test$Direction)

print(confusionMatrix(xtab[2:1,2:1]))

### Question 11

#### (a)

data(Auto)

Auto <- Auto %>%

mutate(mpg01 = factor(ifelse(mpg > median(mpg), 1, 0)),

origin = factor(origin,

levels = c(1,2,3),

labels = c('American', 'European', 'Asian')))

#### (b)

Auto %>%

dplyr::select(-name, -mpg) %>%

ggpairs(aes(col = mpg01, fill = mpg01, alpha = 0.6),

upper = list(combo = 'box'),

diag = list(discrete = wrap('barDiag', position = 'fill')),

lower = list(combo = 'dot\_no\_facet')) +

theme(axis.text.x = element\_text(angle = 90, hjust = 1))

Auto %>%

dplyr::select(-name, -mpg, - origin, -cylinders) %>%

gather(Variable, value, -mpg01) %>%

mutate(Variable = str\_to\_title(Variable)) %>%

ggplot(aes(mpg01, value, fill = mpg01)) +

geom\_boxplot(alpha = 0.6) +

facet\_wrap(~ Variable, scales = 'free', ncol = 1, switch = 'x') +

coord\_flip() +

theme(legend.position = 'top') +

labs(x = '', y = '', title = 'Variable Boxplots by mpg01')

#### (c)

set.seed(1234)

num\_train <- nrow(Auto) \* 0.8

inTrain <- sample(nrow(Auto), size = num\_train)

train <- Auto[inTrain,]

test <- Auto[-inTrain,]

#### (d)

require(MASS)

fmla <- as.formula('mpg01 ~ displacement + horsepower + weight + year + cylinders')

lda\_model <- lda(fmla, data = train)

pred <- predict(lda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$mpg01)

print(confusionMatrix(xtab[2:1,2:1]))

err = test[which(pred\_values!=test$mpg01),]

print(err)

#### (e)

qda\_model <- qda(fmla, data = train)

pred <- predict(qda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$mpg01)

print(confusionMatrix(xtab[2:1,2:1]))

err = test[which(pred\_values!=test$mpg01),]

print(err)

#### (f)

log\_reg <- glm(fmla, data = train, family = binomial)

pred <- predict(qda\_model, newdata = test)

pred\_values <- pred$class

xtab <- table(pred\_values,test$mpg01)

print(confusionMatrix(xtab[2:1,2:1]))

err = test[which(pred\_values!=test$mpg01),]

print(err)

#### (g)

set.seed(1234)

acc <- list()

x\_train <- train[,c('cylinders', 'displacement', 'horsepower', 'weight', 'year')]

y\_train <- train$mpg0

x\_test <- test[,c('cylinders', 'displacement', 'horsepower', 'weight', 'year')]

acc = sapply(1:15, function(x){

knn\_pred <- knn(train = x\_train, test = x\_test, cl = y\_train, k = x)

acc[as.character(x)] = mean(knn\_pred == test$mpg01)

})

unlist(acc)

knn\_pred <- knn(train = x\_train,

test = x\_test,

cl = y\_train, k = 5)

xtab <- table(knn\_pred,test$mpg01)

print(confusionMatrix(xtab[2:1,2:1]))

err = test[which(knn\_pred!=test$mpg01),]

print(err)