Statistical Learning Homework 4

110024516 邱繼賢

```
library(leaps)
library(splines)
library(tree)
library(latex2exp)
library(randomForest)
library(caret)
library(gbm)
library(dplyr)
library(ggplot2)
library(ggpubr)
library(corrplot)
library(ROSE)
```

Problem 1.

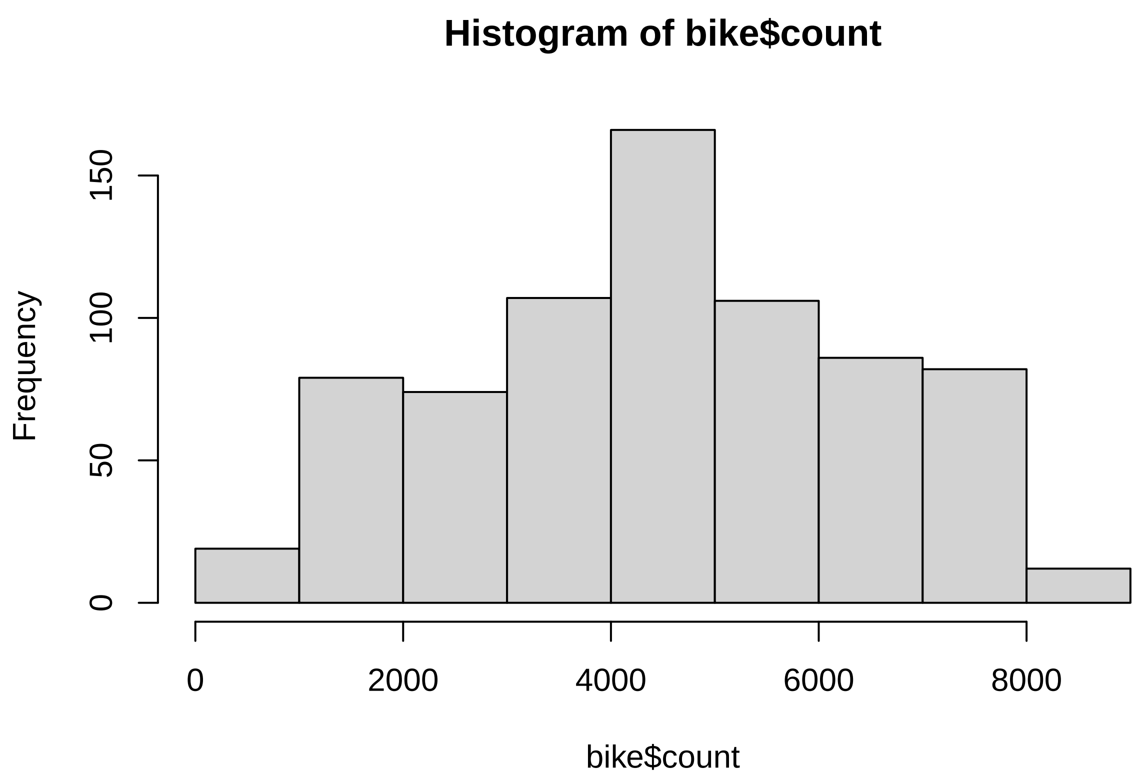
EDA

```
bike <- read.csv(file="bike.csv") #read data
for (i in c(2,4,5,6,7,8)) {
    bike[,i] = as.factor(bike[,i])
}
dim(bike)
```

```
## [1] 731  12
```

- 此資料共 731 比觀測值，12 個變數，其中沒有任何 NA 值

```r
hist(bike$count)
```

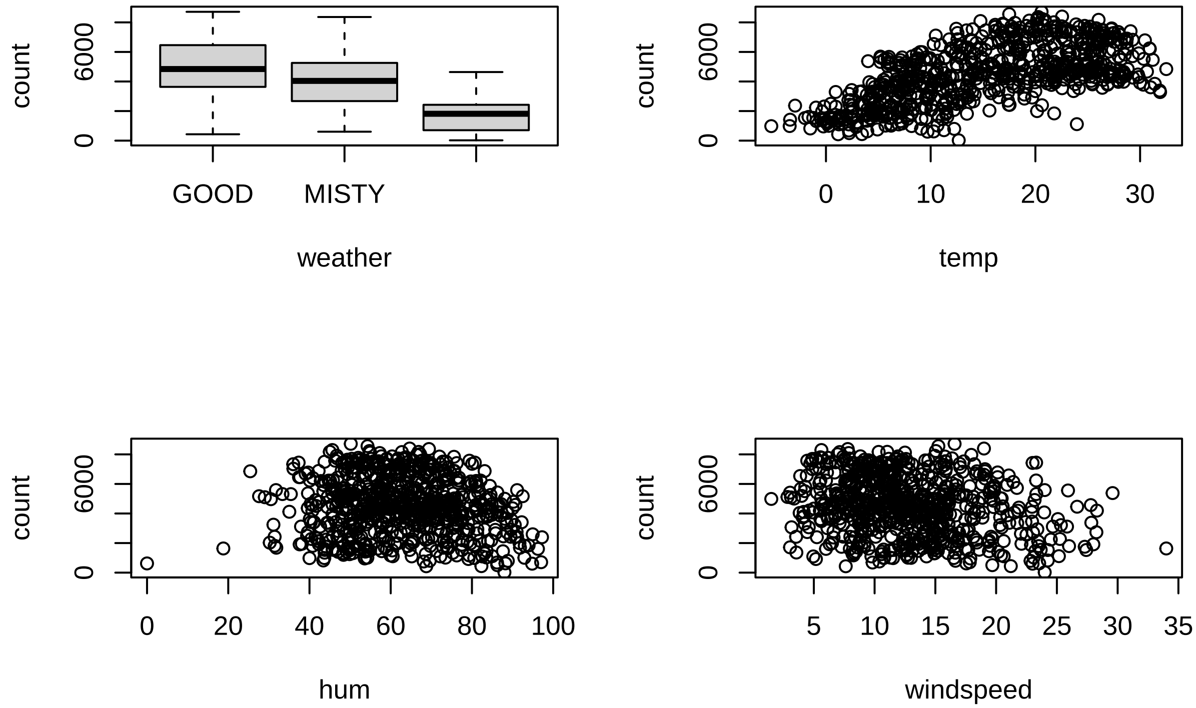


- Response variable：count 為 discrete variable 但是分布數值很廣且左右對稱，可視為 normal continuous variable 處理

- Predictor variables： 其中 7 個為跟時間相關的變數，剩餘 4 個為跟天氣相關的變數，也是我們此次分析主要著重的變數

將跟天氣相關的 4 個變數對 count 作圖：

```r
par(mfrow = c(2,2))
boxplot(count ~ weather, bike)
plot(count ~ temp, bike)
plot(count ~ hum, bike)
plot(count ~ windspeed, bike)
```



- weather 為 nomial variable 共 3 個 levels，可明顯看出隨著天氣狀況變差，count 數量有明顯的下降
- temp, hum, windspeed 皆為 continuous variables，其中只有 temp 對 count 有明顯的 non-linear 趨勢變化，另兩個變數對 count 在圖形上關聯性不明顯

將全部資料以 600:131 的比例隨機分割成 training data 和 testing data：

```
set.seed(1209)
idx = sample(1:731, 131)
train_bike = bike[-idx,]
test_bike = bike[idx,]
```

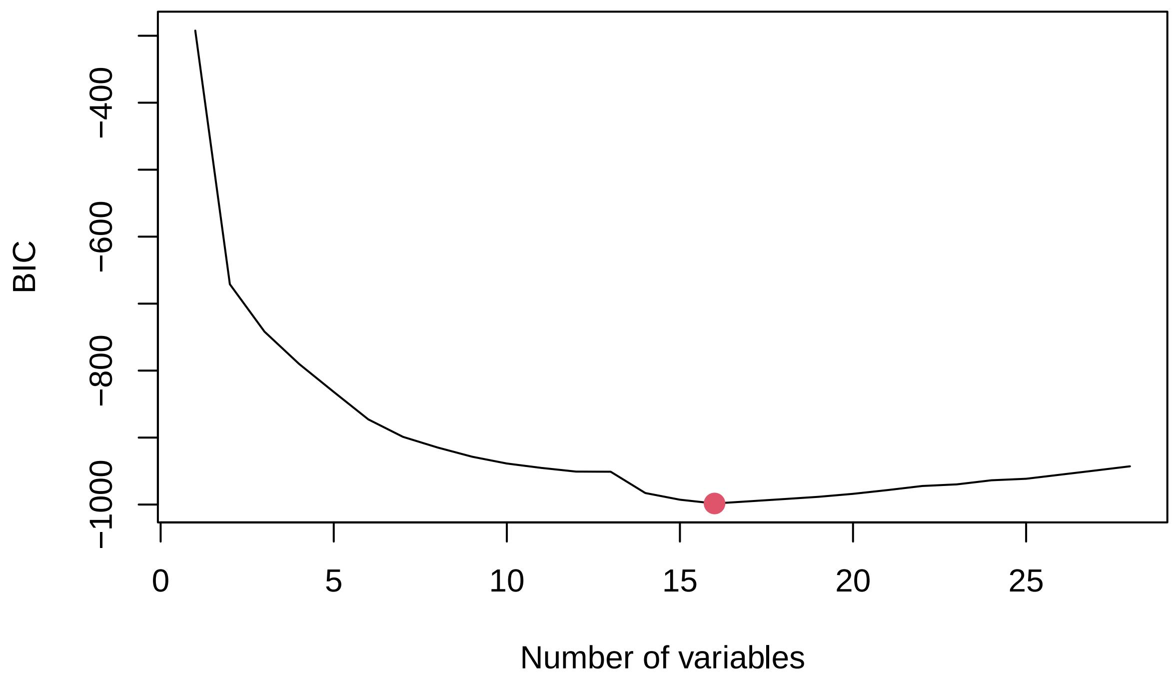
以下建模皆是對 training data 進行，並比較其在 testing data 上的表現

Linear Regression

以 count 為 response 建構 linear model，並且以 BIC criterion 對模型做 forward seletion

```r
fit1.1_reg = regsubsets(count~., data = train_bike[,-7],
                        nvmax = 28, method = "forward")
fit1.1_reg_sum = summary(fit1.1_reg)
```

```r
plot(fit1.1_reg_sum$bic, type = "l", xlab = "Number of variables",
     ylab = "BIC")
points(16,fit1.1_reg_sum$bic[16], cex=1.5,pch=16,col=2)
```




```r
which.min(fit1.1_reg_sum$bic)
```

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

選取 16 個變數時，BIC 達到最小，此 16 個變數的估計係數如下：

```r
coef(fit1.1_reg, 16) %>% round(2)
```

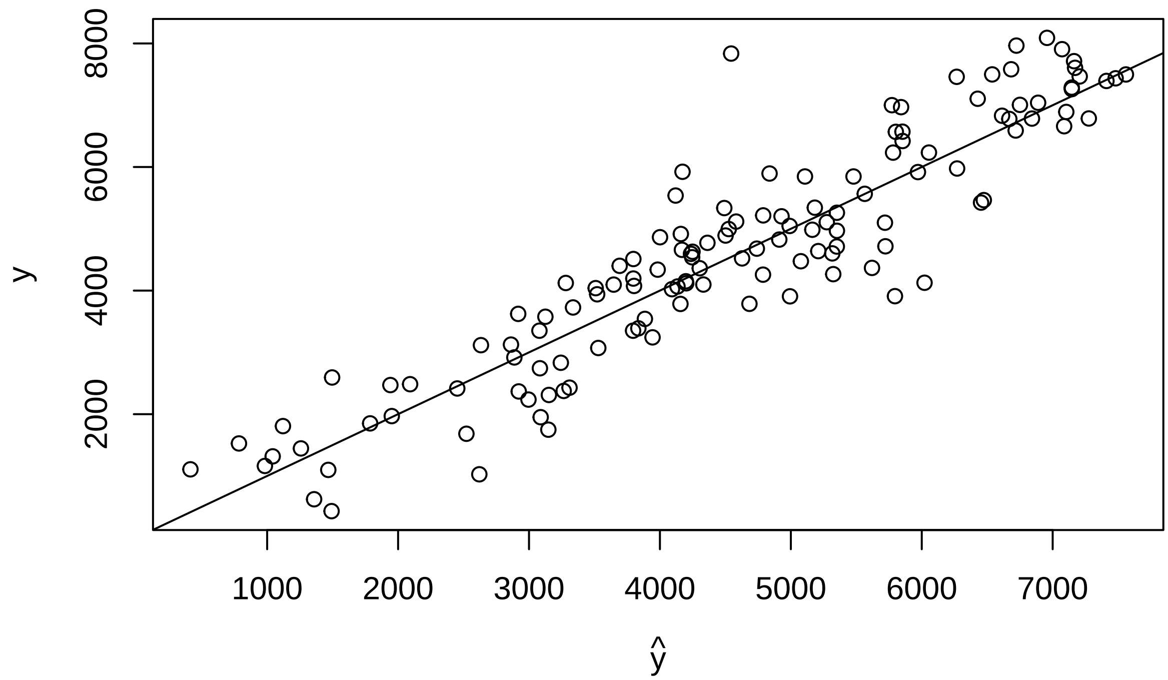
```
##          (Intercept)     days_since_2011         seasonSPRING
##          -4703601.13               -0.86              -346.96
```

```
##          seasonSUMMER          seasonWINTER                  year
##               -585.43              -1554.36               2340.79
##              monthDEC              monthJUL              monthOCT
##                 -7.11               -524.98                464.60
##              monthSEP      holidayNO HOLIDAY            weekdaySUN
##                750.37                738.58               -292.53
##          weatherMISTY weatherRAIN/SNOW/STORM                 temp
##               -436.86              -1825.83                107.50
##                   hum             windspeed
##                -17.21                -43.89
```

```
predict.regsubsets <- function(object, newdata, id, ...) {
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id = id)
  xvars <- names(coefi)
  mat[, xvars] %*% coefi
 }
```

以此模型對 testing data 進行預測，並計算 MSE

```
pred = predict.regsubsets(fit1.1_reg, test_bike, id=16)
plot(pred, test_bike$count, xlab=TeX("$\\hat{y}$"), ylab = "y")
abline(0,1)
```




```
mean((pred-test_bike$count)^2)
```

```
## [1] 551245.4
```

Non-linear Regression

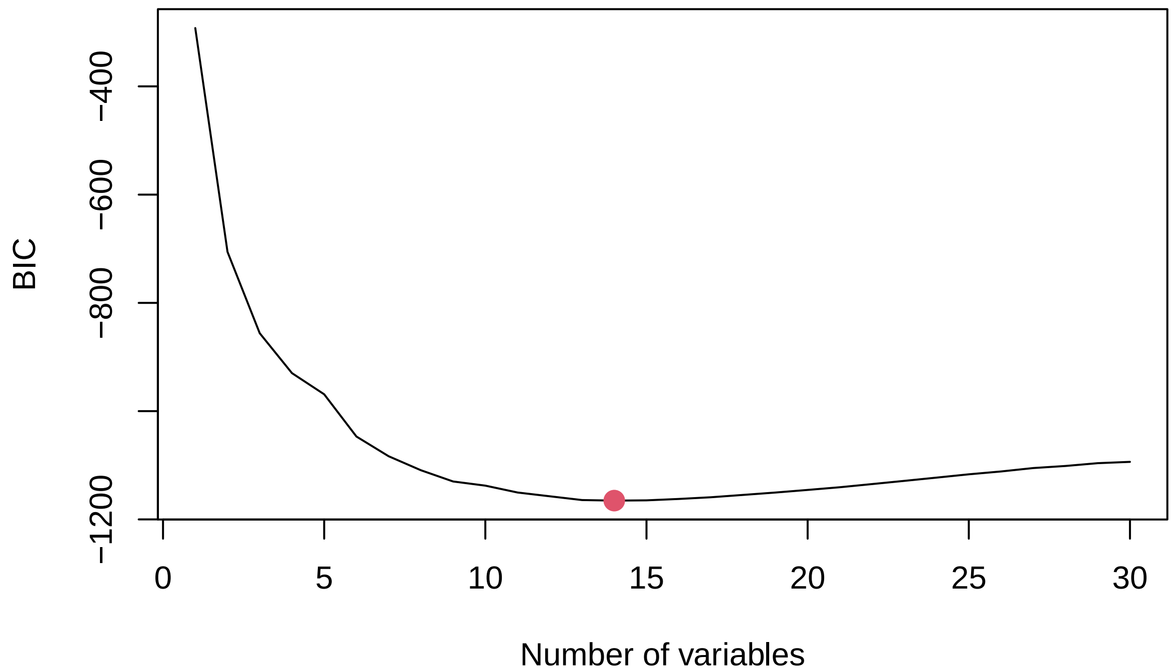
藉由 EDA 對變數 temp 的觀察，可以發現其對 response count 有非線性影響，以下建構 non-linear model：對 temp 做 natural spline 取 knots = 10, 20，然後一樣使用 BIC criterion 做 forward selection

```
fit1.2_reg = regsubsets(count ~ . - temp + ns(temp, knots = c(10,20)),
                        train_bike[,-7], nvmax = 30, method = "forward")
fit1.2_reg_sum = summary(fit1.2_reg)
```

```
plot(fit1.2_reg_sum$bic, type="l", xlab = "Number of variables",
     ylab = "BIC")
which.min(fit1.2_reg_sum$bic)
```

```
## [1] 14
```

```r
points(14,fit1.2_reg_sum$bic[14], pch=16, cex=1.5, col=2)
```



選取 14 個變數時有最小的 BIC，估計係數如下：

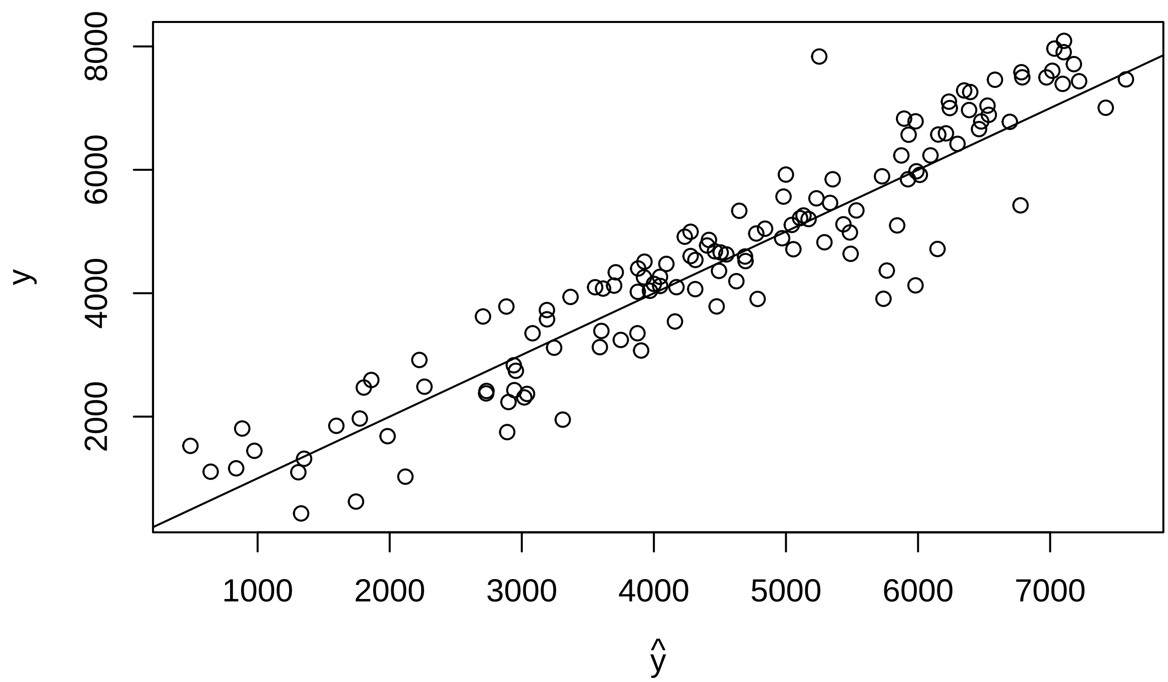
```r
coef(fit1.2_reg, 14) %>% round(2)
```

```
##             (Intercept)          days_since_2011
##             -4573003.29                    -0.84
##            seasonSPRING             seasonSUMMER
##                 -709.10                  -351.64
##            seasonWINTER                     year
##                -1550.59                  2276.26
##              weekdayMON               weekdaySUN
##                 -304.71                  -345.02
##            weatherMISTY   weatherRAIN/SNOW/STORM
##                 -401.64                 -1780.55
##                     hum                windspeed
##                  -22.26                   -49.24
```

```
## ns(temp, knots = c(10, 20))1 ns(temp, knots = c(10, 20))2
##                      4103.45                      2998.85
## ns(temp, knots = c(10, 20))3
##                      1100.30
```

以此 non-linear model 對 testing data 進行預測，並計算 MSE

```
pred = predict.regsubsets(fit1.2_reg, test_bike, 14)
plot(pred, test_bike$count, xlab = TeX("$\\hat{y}$"), ylab = "y")
abline(0,1)
```




```
mean((pred-test_bike$count)^2)
```

```
## [1] 438341.7
```

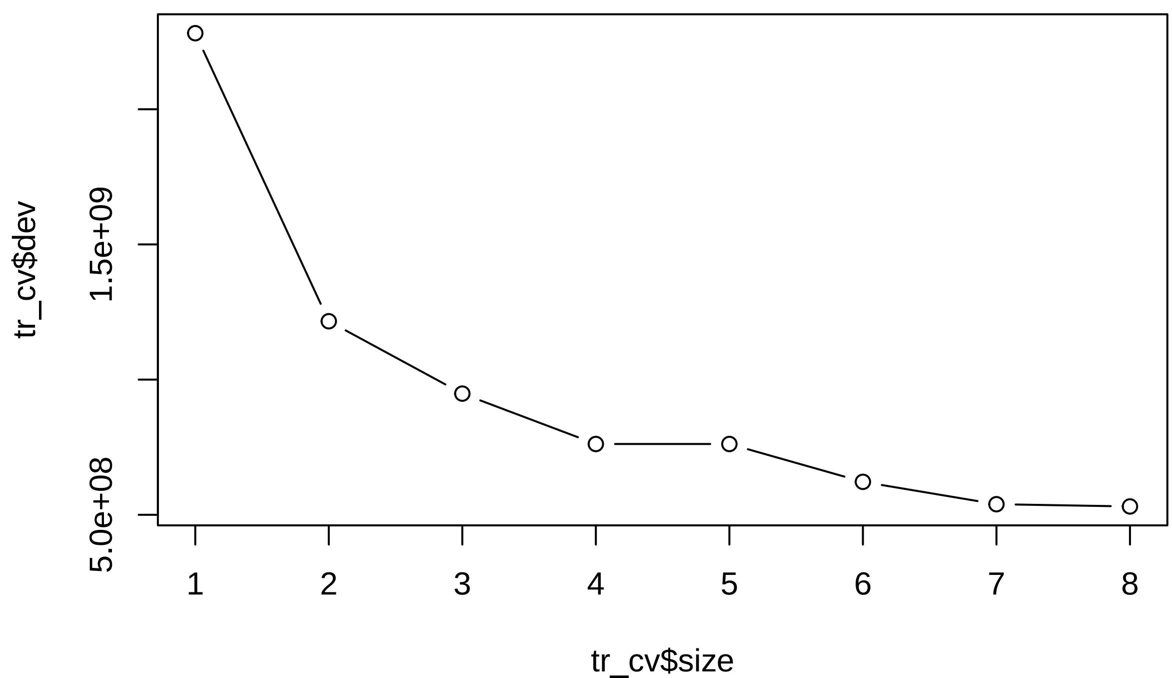
$\Rightarrow$ 相對於 linear model MSE 有所減少

Tree-Based Models

Tree

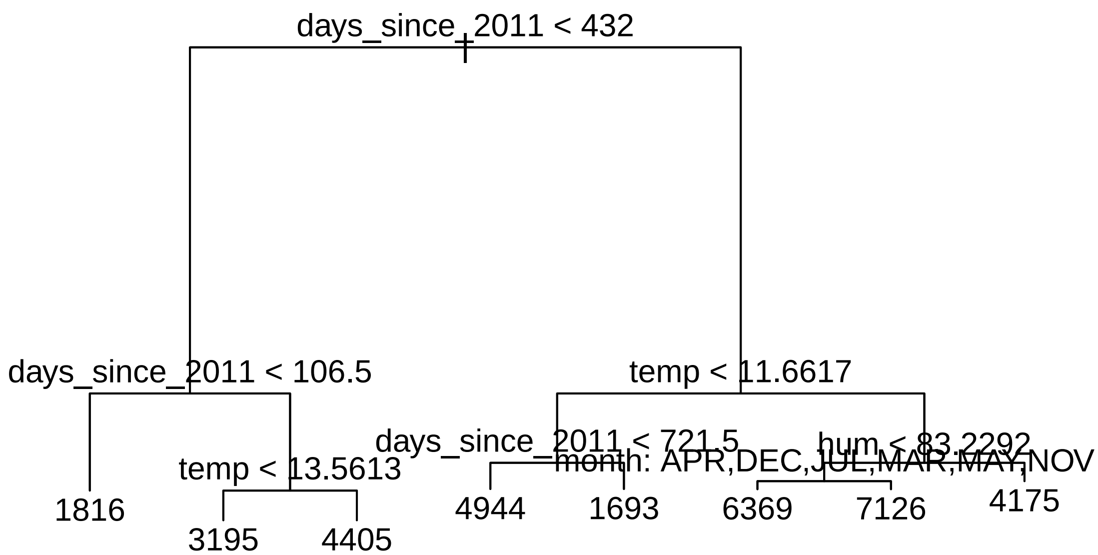
建構 tree model，並且利用 5-fold CV 決定 terminal nodes 的數量

```
tr = tree(count ~ ., data = train_bike)
set.seed(12091)
tr_cv = cv.tree(tr, FUN = prune.tree, K=5)
plot(tr_cv$size, tr_cv$dev, type = "b")
```



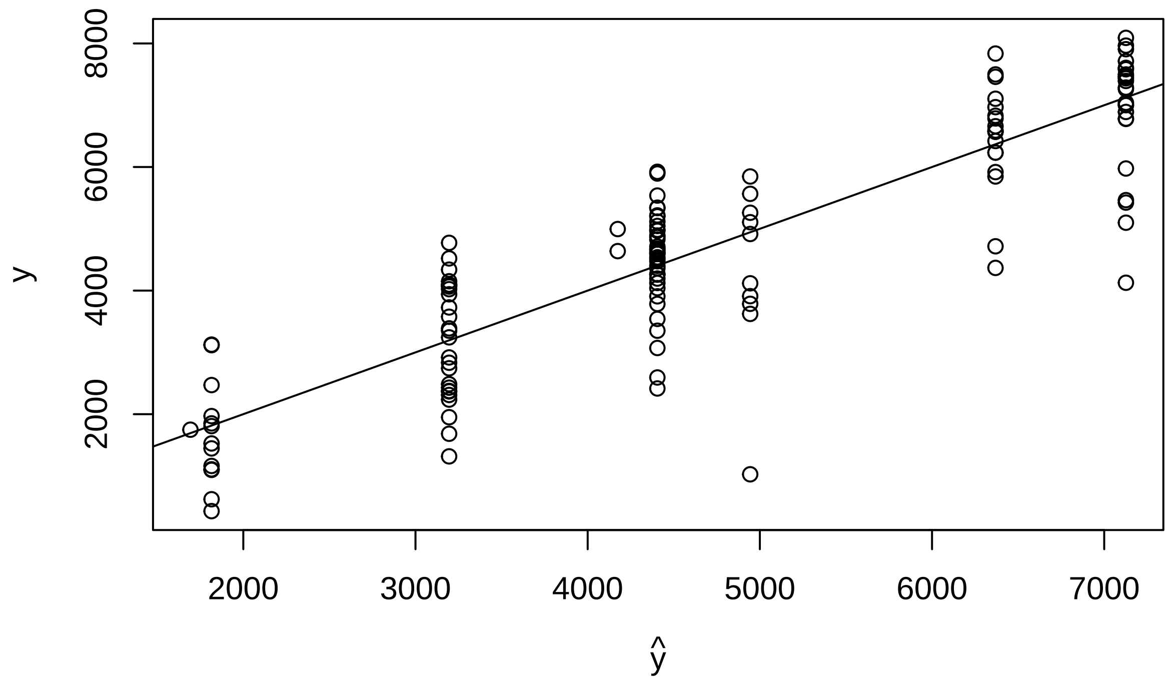
決定 terminal nodes = 8

```
fit1.3 = prune.tree(tr, best = 8)
plot(fit1.3)
text(fit1.3, pretty = 0)
```



使用此 tree model 對 testing data 進行預測，並計算 MSE

```r
pred = predict(fit1.3, test_bike)
plot(pred, test_bike$count, xlab = TeX("$\\hat{y}$"), ylab = "y")
abline(0,1)
```



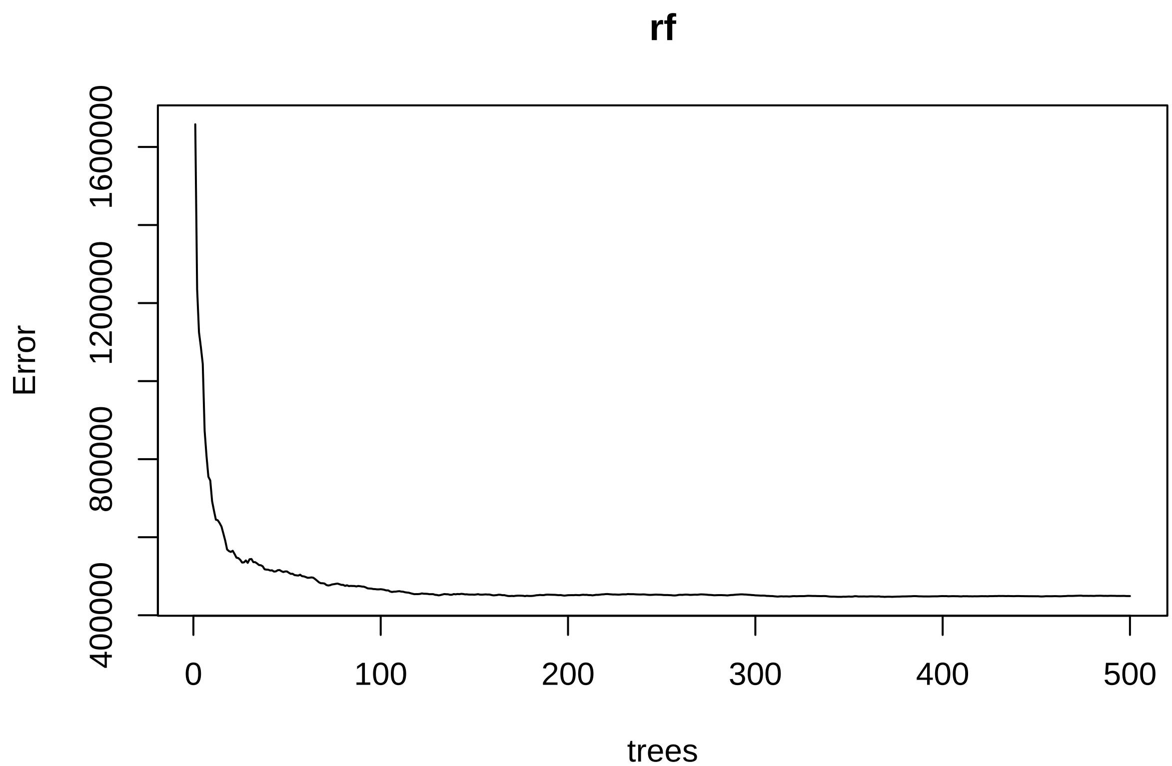

```
mean((pred-test_bike$count)^2)
```

```
## [1] 878507.9
```

Random Forests

Bagging of trees 只是 random forest 在 m=p 時的特例，故在此只建構 random forest model

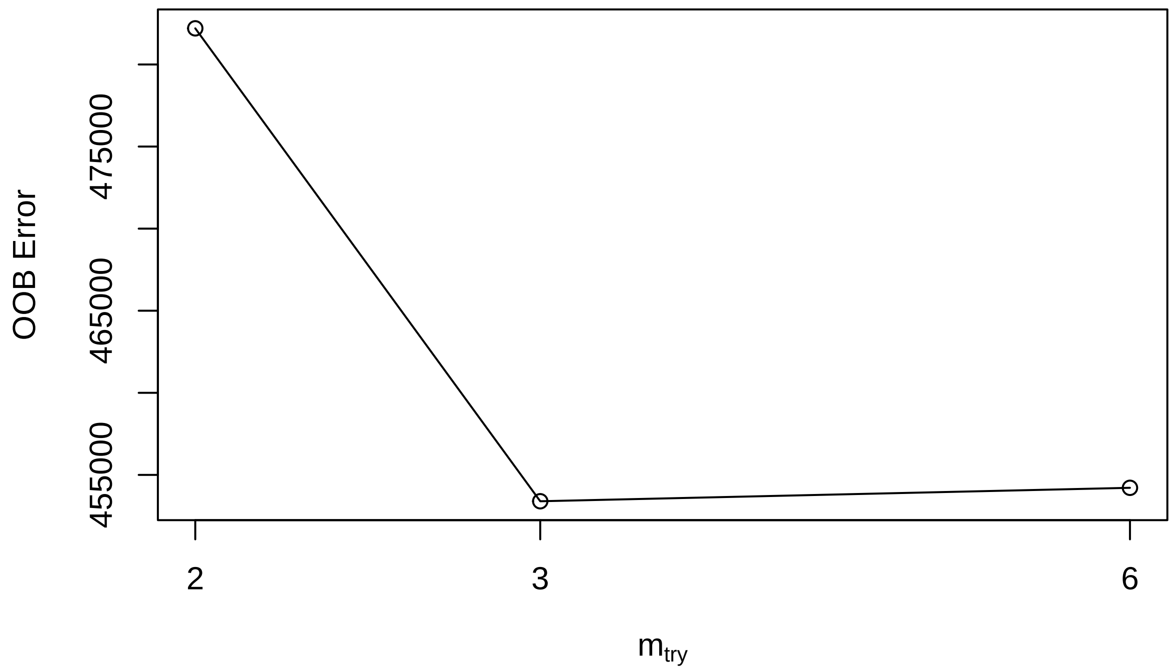
```
rf = randomForest(count ~ ., train_bike)
plot(rf)
```



大概 ntrees > 200 後 error 趨於穩定，再來利用 OOB error 決定參數 m

```
tuneRF(train_bike[,-12],train_bike[,12], ntreeTry = 200)
```

```
## mtry = 3  OOB error = 453395.4
## Searching left ...
## mtry = 2     OOB error = 482199.6
## -0.06353008 0.05
## Searching right ...
## mtry = 6     OOB error = 454212.8
## -0.001802786 0.05
```

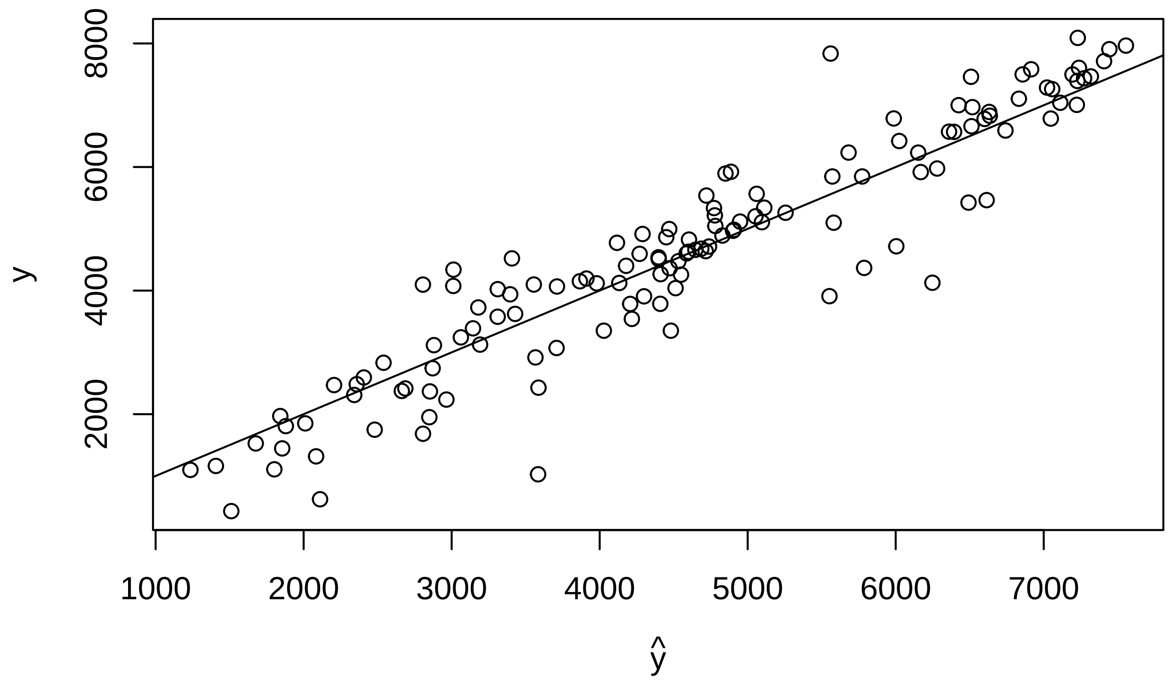



```
##   mtry OOBError
## 2    2 482199.6
## 3    3 453395.4
## 6    6 454212.8
```

⇒ 決定 m=3

利用 (ntree $= 200$，$m = 3$) 的 random forests model 對 testing data 進行預測，並計算 MSE

```
set.seed(12090)
fit1.4 = randomForest(count ~., train_bike, ntree=200, mtry=3,
                      importance = T)
pred = predict(fit1.4, test_bike)
plot(pred, test_bike$count, xlab = TeX("$\\hat{y}$"), ylab="y")
abline(0,1)
```




```
mean((pred-test_bike$count)^2)
```

```
## [1] 442875.1
```

Boosting

建構 boosting model，並利用 5-fold CV 選取 tuning parameter：n.trees, interaction.depth, shrinkage

```
set.seed(12094)
contrl = trainControl(method = "repeatedcv", number=5, repeats = 1)
boots = train(count ~ ., train_bike, method = "gbm",
              trControl = contrl, verbose=F)
boots
```

```
## Stochastic Gradient Boosting
##
## 600 samples
##  11 predictor
```

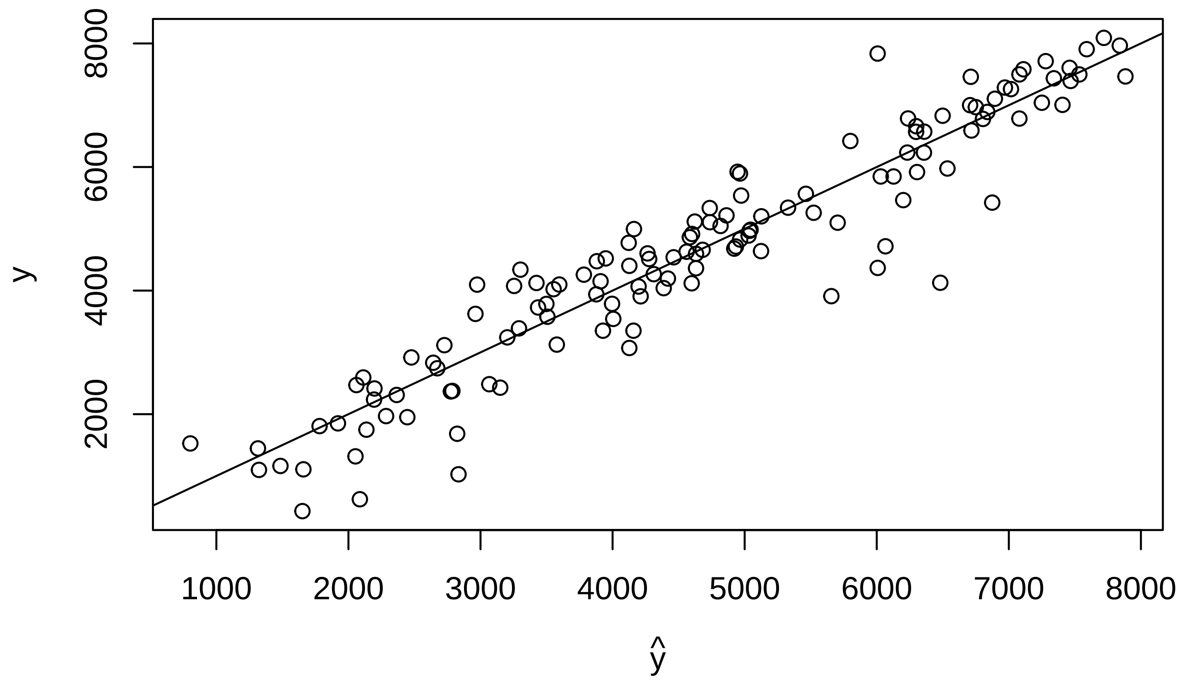
```
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 1 times)
## Summary of sample sizes: 480, 480, 480, 480, 480
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  RMSE      Rsquared   MAE
##   1                   50      871.0494  0.8170994  647.7889
##   1                  100      758.4417  0.8507546  550.6932
##   1                  150      727.6231  0.8609595  521.1160
##   2                   50      750.4880  0.8546718  541.4160
##   2                  100      688.8816  0.8743615  489.5386
##   2                  150      673.9056  0.8804652  479.3101
##   3                   50      714.5174  0.8656928  502.3411
##   3                  100      673.3243  0.8795628  463.2402
##   3                  150      655.2624  0.8861280  451.1658
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were n.trees = 150, interaction.depth =
##  3, shrinkage = 0.1 and n.minobsinnode = 10.
```

$\Rightarrow$ (n.trees, interaction.depth, shrinkage) = (150, 3, 0.1)

以選取後的參數建構 boosting model 對 testing data 進行預測，並計算 MSE

```r
fit1.5 = gbm(count ~., data=train_bike, n.trees = 150, distribution = "gaussian",
             interaction.depth = 3, shrinkage = 0.1)
```

```r
pred = predict(fit1.5, test_bike)
plot(pred, test_bike$count, xlab = TeX("$\\hat{y}$"), ylab = "y")
abline(0,1)
```




```r
mean((pred-test_bike$count)^2)
```

```
## [1] 377254.1
```

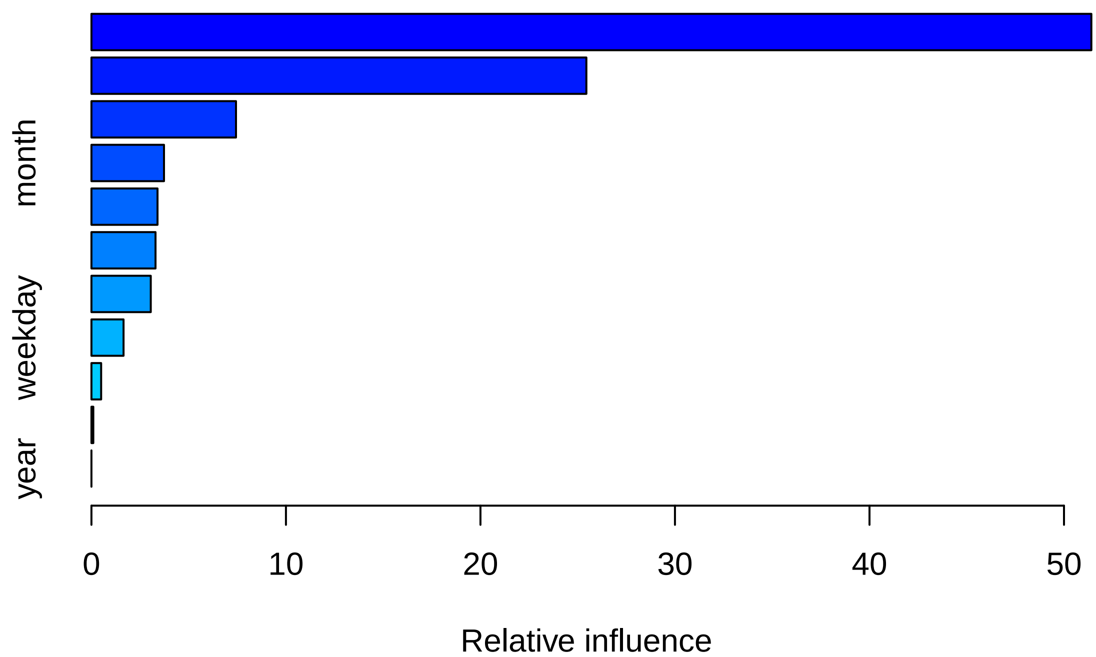
Performance comparison

	Linear model	Non-Linear model	Tree	Random Forests	Boosting
MSE	551245.4	438341.7	878507.9	442875.1	344140.7

⇒ Boosting model 在 MSE 上的表現最好，以下對該模型重要變數解釋

Important input variable and Summary

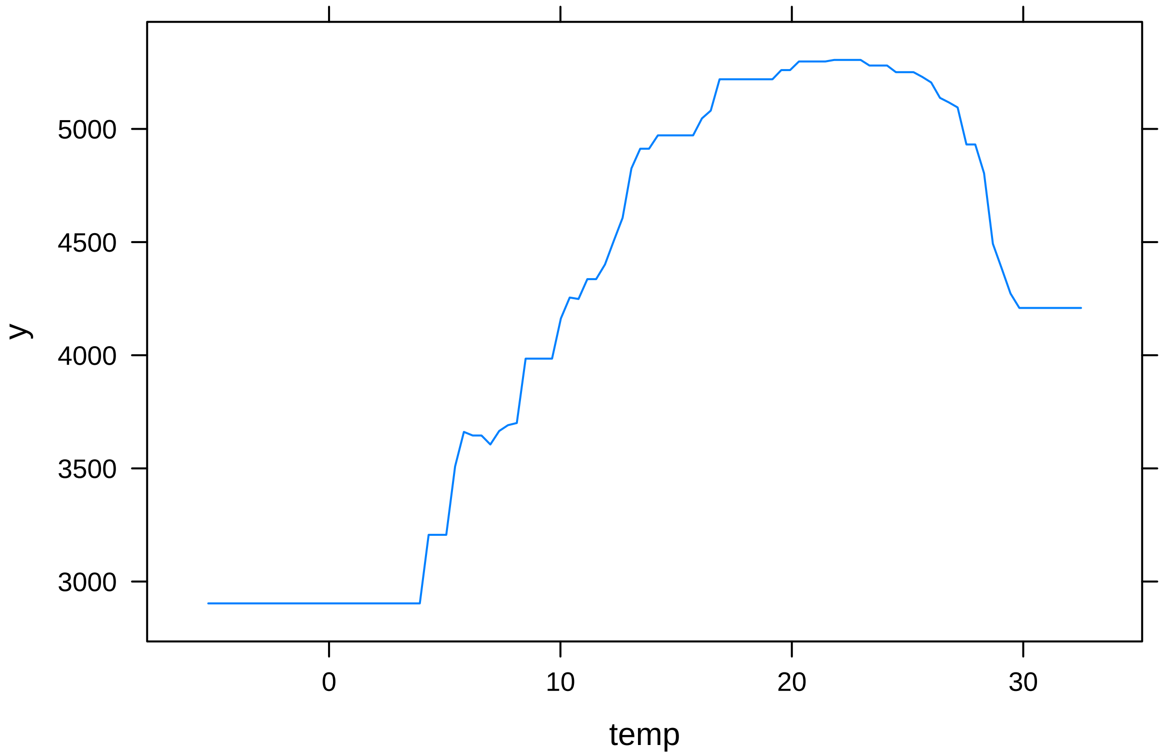
```r
summary(fit1.5)
```



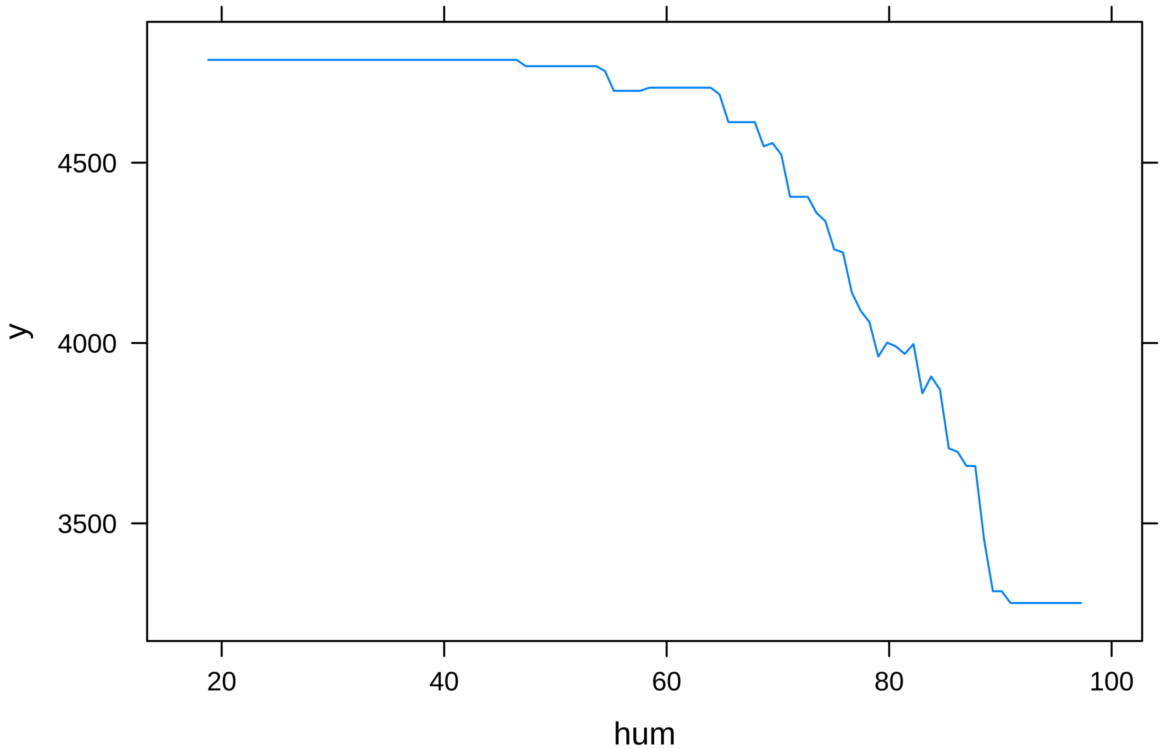

```
##                           var     rel.inf
## days_since_2011 days_since_2011 51.40688272
## temp                       temp 25.44431404
## hum                         hum  7.43147291
## month                     month  3.72995785
## windspeed             windspeed  3.39723909
## weather                 weather  3.29454586
## season                   season  3.05021915
## weekday                 weekday  1.65022347
## workingday           workingday  0.49870047
## holiday                 holiday  0.09644444
## year                       year  0.00000000
```

最重要的變數是 day_since_2011 和 temp，以下僅對跟天氣有關的四個變數繪製 partial dependence plot

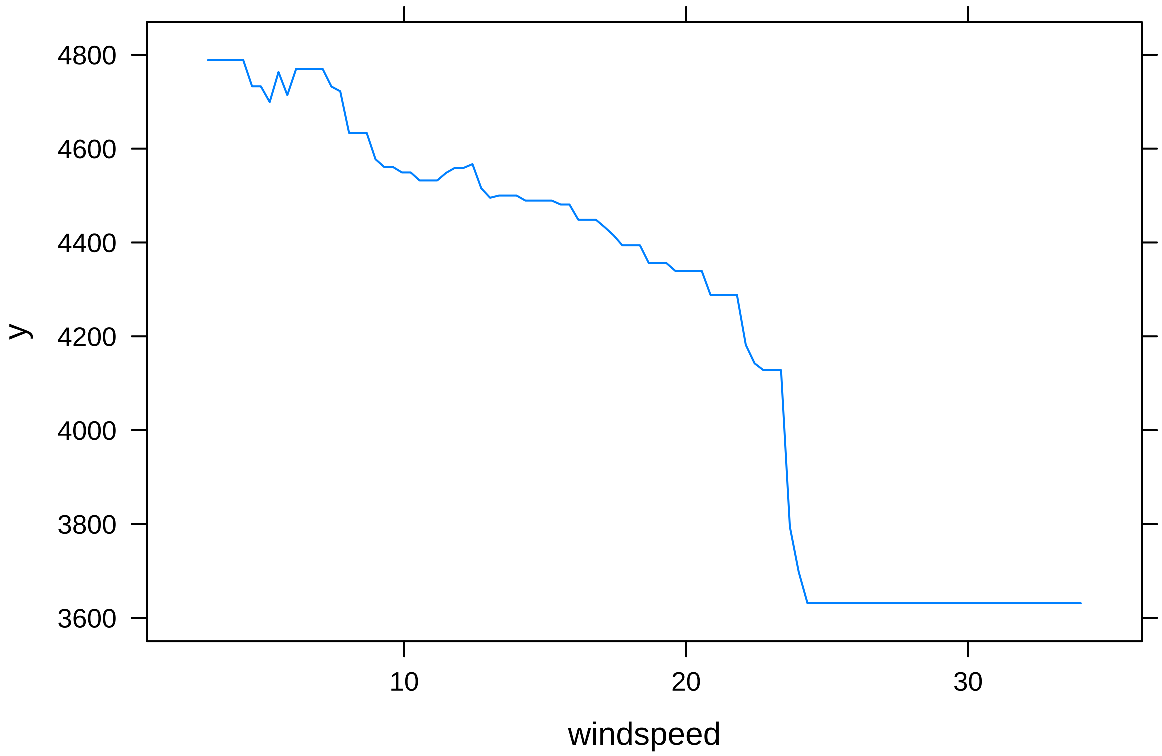
```
par(mfrow = c(2,2))
plot(fit1.5, i = "temp")
```



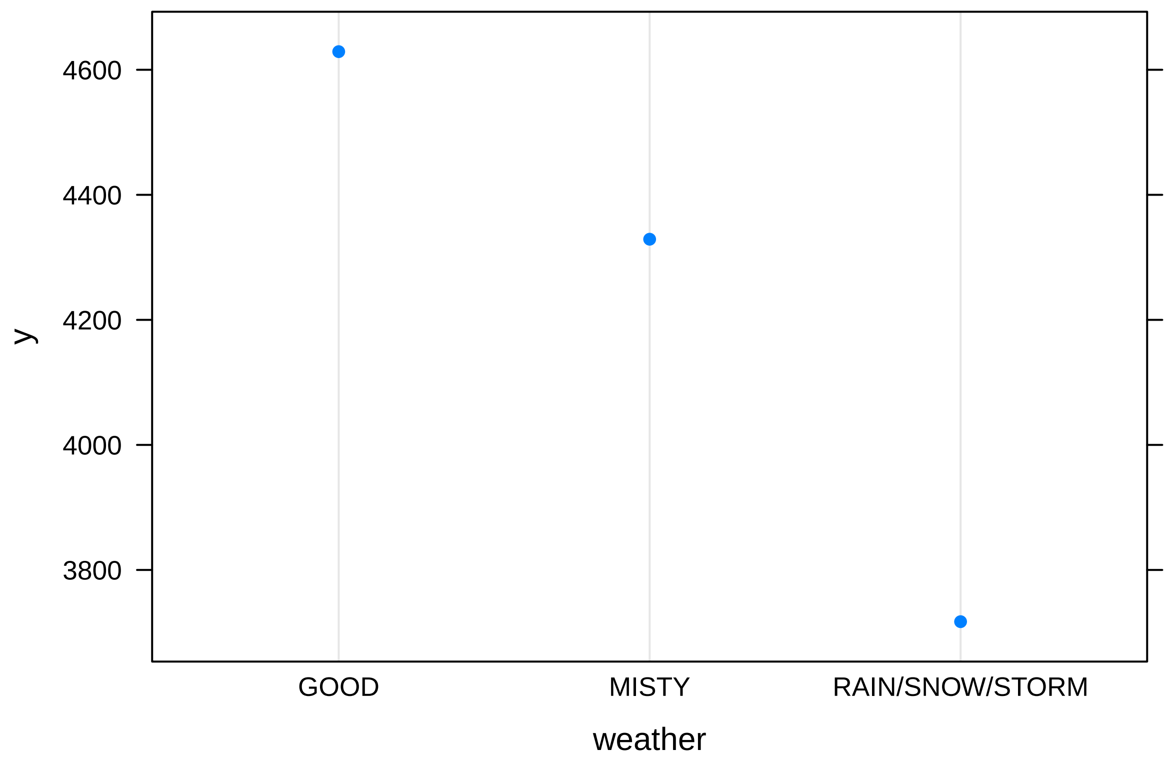

```
plot(fit1.5, i = "hum")
```




```
plot(fit1.5, i = "windspeed")
```




```
plot(fit1.5, i = "weather")
```



- 隨著 temp 上升，response 呈現先升後降的曲線趨勢，與我們在 EDA 時的觀察一致
- hum 上升到 50 之後，response 數量開始下降
- windspeed 上升 response 隨著下降，約升至 25 時，response 降到最低
- weather 越差，response 數量越低，與 EDA 時的觀察一致

藉由以上天氣變數可以推斷，天氣會明顯的影響使用者對腳踏車的租借數量，舒適的天氣 (溫度適中、濕度低、風速小、天氣晴朗) 則租借腳踏車的數量越多，而不適的天氣 (溫度過高或過低、濕度高、風速大、天氣陰雨) 則租借腳踏車的數量越少。

Problem 2.

EDA

導入資料並移除 NA 值：

```r
survey <- read.csv(file="airline.csv") #read data
survey1 <- na.omit(survey) #remove missing data
for (i in c(2,3,5,6,24)) {
    survey1[,i] = as.factor(survey1[,i])
}
survey1 = survey1[,-1]
```

```r
glimpse(survey1)
```

```
## Rows: 103,594
## Columns: 23
## $ Gender                            <fct> Male, Male, Female, Female, Male, Fe~
## $ Customer.Type                     <fct> Loyal Customer, disloyal Customer, L~
## $ Age                               <int> 13, 25, 26, 25, 61, 26, 47, 52, 41, ~
## $ Type.of.Travel                    <fct> Personal Travel, Business travel, Bu~
## $ Class                             <fct> Eco Plus, Business, Business, Busine~
## $ Flight.Distance                   <int> 460, 235, 1142, 562, 214, 1180, 1276~
## $ Inflight.wifi.service             <int> 3, 3, 2, 2, 3, 3, 2, 4, 1, 3, 4, 2, ~
## $ Departure.Arrival.time.convenient <int> 4, 2, 2, 5, 3, 4, 4, 3, 2, 3, 5, 4, ~
## $ Ease.of.Online.booking            <int> 3, 3, 2, 5, 3, 2, 2, 4, 2, 3, 5, 2, ~
## $ Gate.location                     <int> 1, 3, 2, 5, 3, 1, 3, 4, 2, 4, 4, 2, ~
## $ Food.and.drink                    <int> 5, 1, 5, 2, 4, 1, 2, 5, 4, 2, 2, 1, ~
## $ Online.boarding                   <int> 3, 3, 5, 2, 5, 2, 2, 5, 3, 3, 5, 2, ~
## $ Seat.comfort                      <int> 5, 1, 5, 2, 5, 1, 2, 5, 3, 3, 2, 1, ~
## $ Inflight.entertainment            <int> 5, 1, 5, 2, 3, 1, 2, 5, 1, 2, 2, 1, ~
## $ On.board.service                  <int> 4, 1, 4, 2, 3, 3, 3, 5, 1, 2, 3, 1, ~
## $ Leg.room.service                  <int> 3, 5, 3, 5, 4, 4, 3, 5, 2, 3, 3, 2, ~
## $ Baggage.handling                  <int> 4, 3, 4, 3, 4, 4, 4, 5, 1, 4, 5, 5, ~
## $ Checkin.service                   <int> 4, 1, 4, 1, 3, 4, 3, 4, 4, 4, 3, 5, ~
## $ Inflight.service                  <int> 5, 4, 4, 4, 3, 4, 5, 5, 1, 3, 5, 5, ~
## $ Cleanliness                       <int> 5, 1, 5, 2, 3, 1, 2, 4, 2, 2, 2, 1, ~
## $ Departure.Delay.in.Minutes        <int> 25, 1, 0, 11, 0, 0, 9, 4, 0, 0, 0, 0~
## $ Arrival.Delay.in.Minutes          <int> 18, 6, 0, 9, 0, 0, 23, 0, 0, 0, 0, 0~
## $ satisfaction                      <fct> neutral or dissatisfied, neutral or ~
```

這是一筆來自航空公司的問卷調查資料，一共 103594 比觀測值，23 個變數：

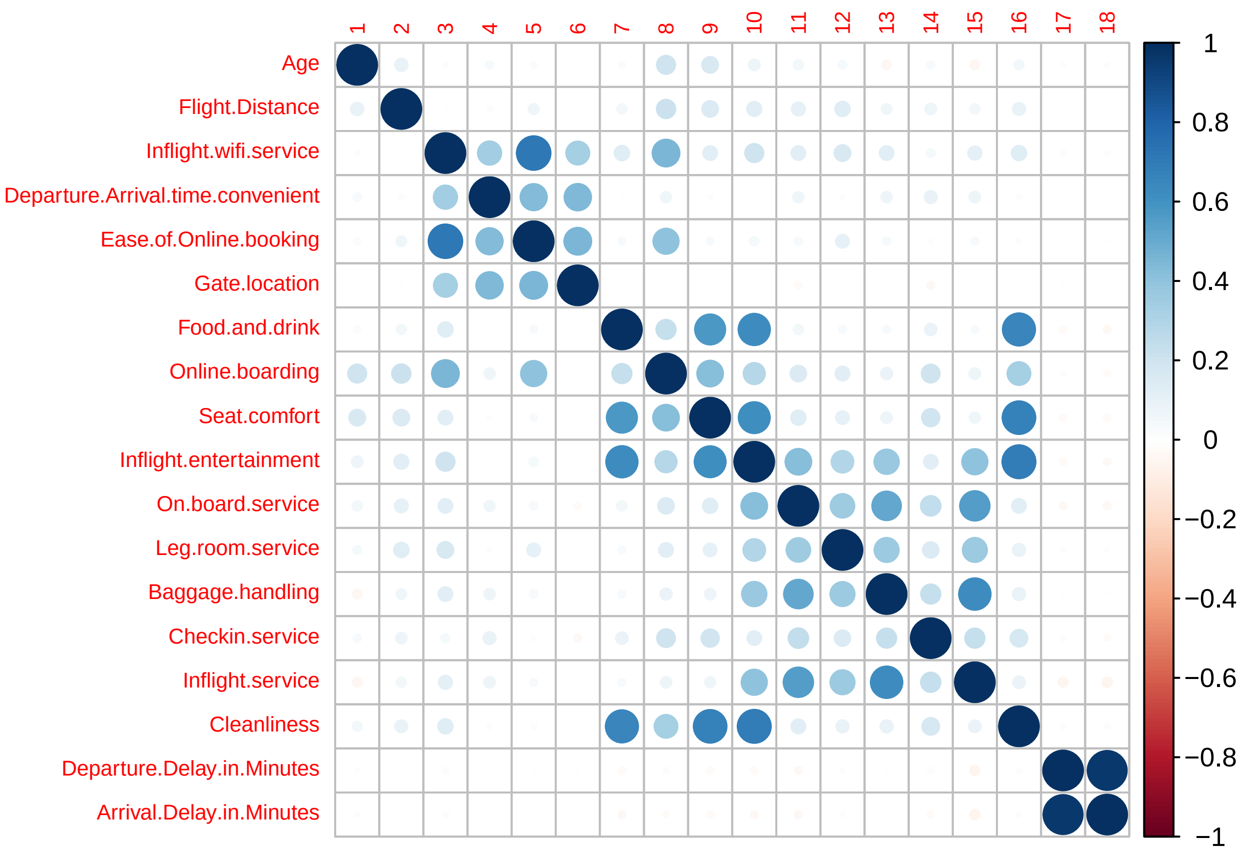
- response variable： *satisfaction* 為一 2-level 類別型變數

- *Gender, Customer.Type, Type.of.Travel, Class* 皆為類別型變數

- 有一大部分變數是問卷評分，為評分 0~5 的 ordinal 變數，在以下分析將其皆視為 continuous 變數

觀察所有連續型變數各自間的 corrplot

```
cor_ = cor(survey1[,-c(1,2,4,5,23)])
colnames(cor_) = NULL
corrplot(cor_, tl.cex=0.7)
```



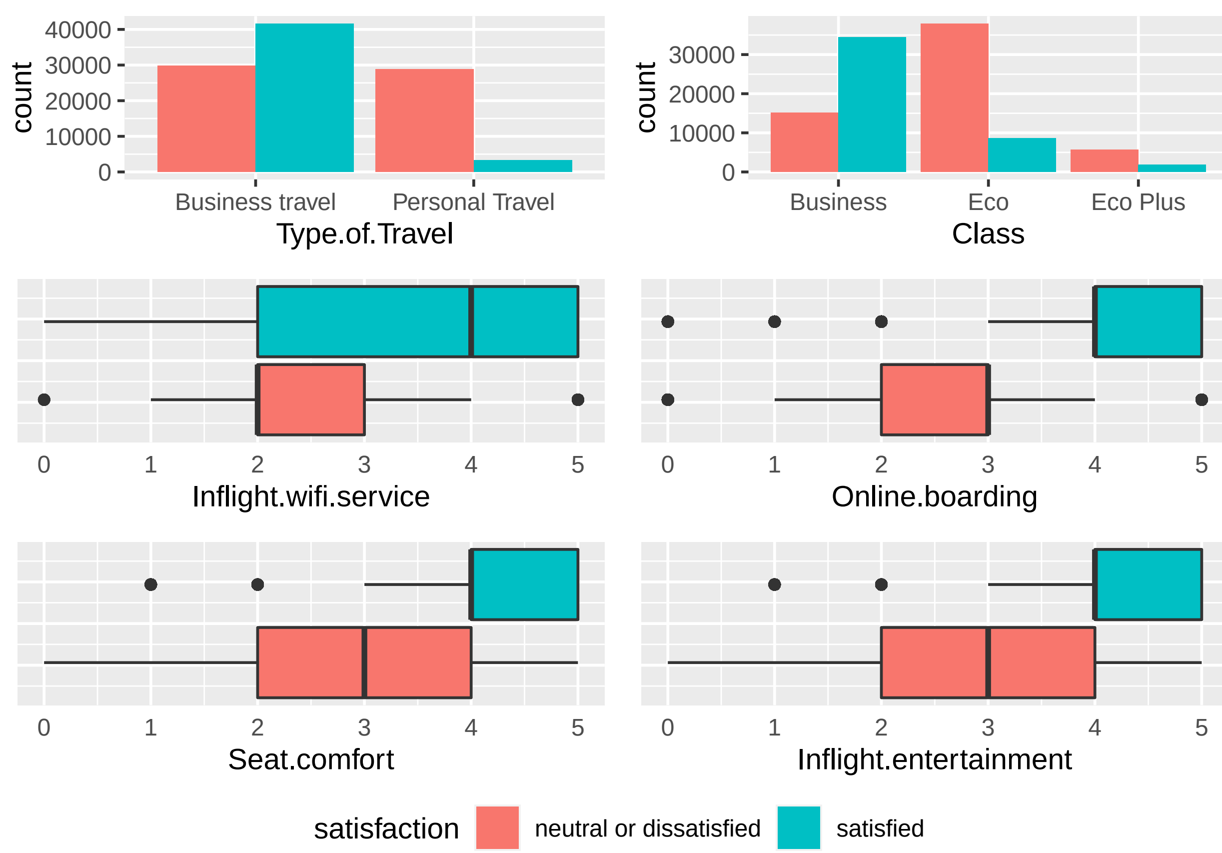
Departure.Delay.in.Minutes 和 *Arrival.Delay.in.Minutes* 有非常強的正相關，建構模型時此二變數可能會有很強的共線性

繪製各變數對 response 影響 (以下僅畫出幾個較明顯的變數)：

```
p1 = ggplot(survey1, aes(x=Type.of.Travel, fill=satisfaction)) +
    geom_bar(position = "dodge")
```

```
p2 = ggplot(survey1, aes(x = Class, fill = satisfaction)) +
    geom_bar(position = "dodge")
p3 = ggplot(survey1, aes(x=Inflight.wifi.service, fill=satisfaction)) +
    geom_boxplot() +
    theme(axis.text.y = element_blank(), axis.ticks = element_blank())
p4 = ggplot(survey1, aes(x=Online.boarding, fill=satisfaction)) +
    geom_boxplot() +
    theme(axis.text.y = element_blank(), axis.ticks = element_blank())
p5 = ggplot(survey1, aes(x=Seat.comfort, fill=satisfaction)) +
    geom_boxplot() +
    theme(axis.text.y = element_blank(), axis.ticks = element_blank())
p6 = ggplot(survey1, aes(x=Inflight.entertainment, fill=satisfaction)) +
    geom_boxplot() +
    theme(axis.text.y = element_blank(), axis.ticks = element_blank())

ggarrange(p1,p2,p3,p4,p5,p6, ncol=2, nrow=3, common.legend = T, legend = "bottom")
```



將資料以 83594:2000 的比例隨機分割成 training data 和 testing data，以下建模皆是利用 training data 建構，

並觀察其在 testing data 上表現

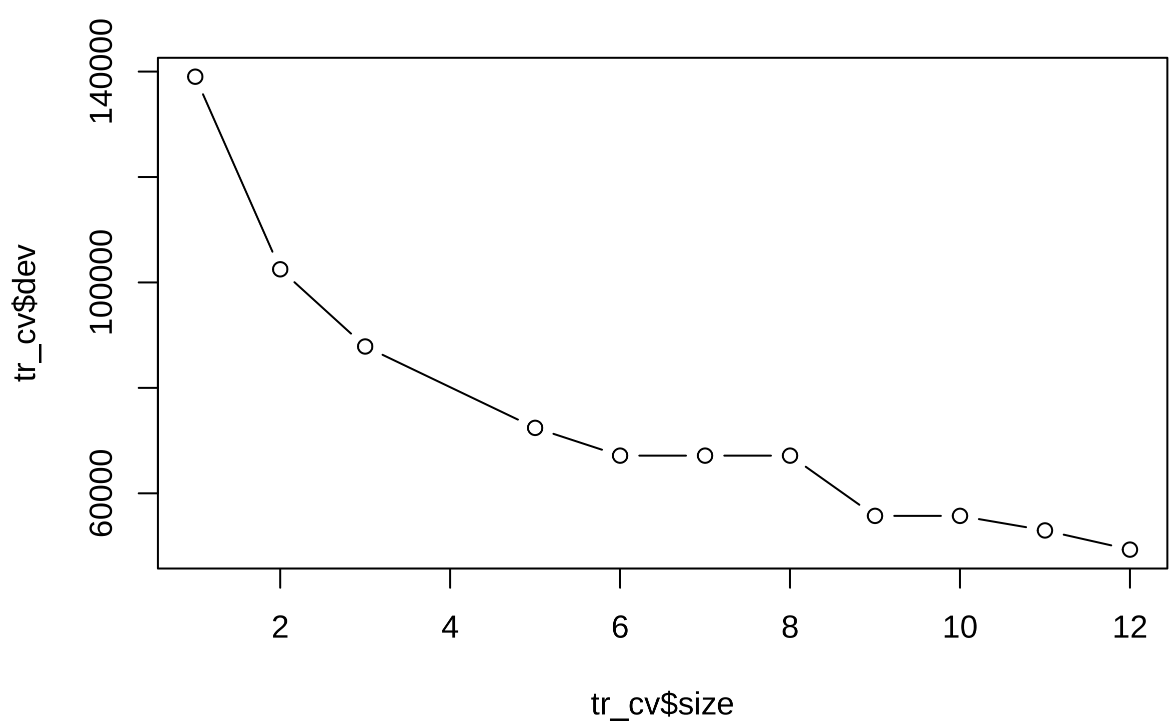
```
set.seed(1210)
idx = sample(1:103594, 2000)
train_survey = survey1[-idx,]
test_survey = survey1[idx,]
```

Tree based model

Tree

建構 tree model，並且利用 5-fold CV 決定 terminal nodes 的數量

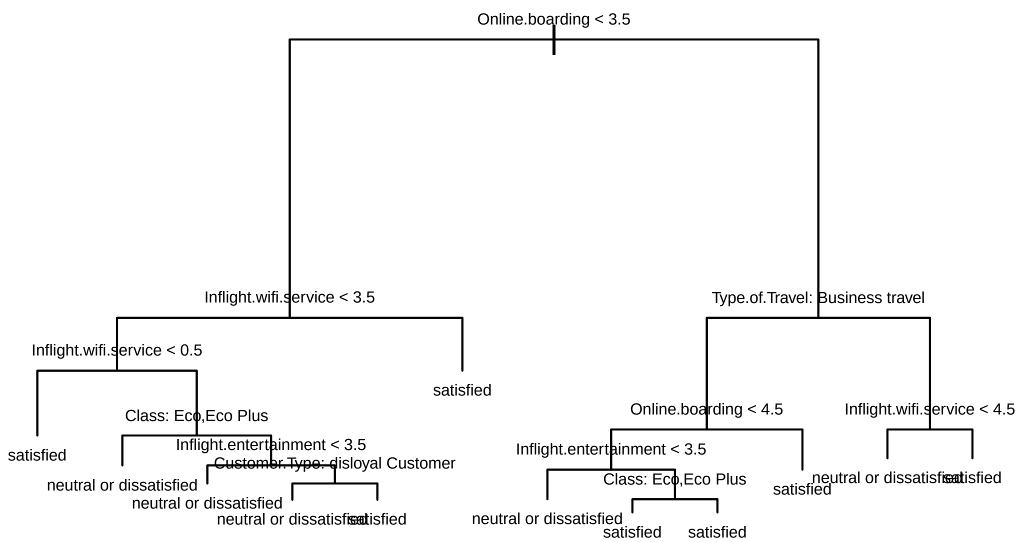
```
tr = tree(satisfaction ~ ., data = train_survey)
set.seed(12121)
tr_cv = cv.tree(tr, FUN = prune.tree, K=5)
plot(tr_cv$size, tr_cv$dev, type = "b")
```



⇒ terminal nodes = 12

建構 classification tree 如下

```
fit2.1 = prune.tree(tr, best = 12)
plot(fit2.1)
text(fit2.1, pretty = 0, cex = 0.5)
```



將此模型對 testing data 進行預測並計算 ACC 和 AUC

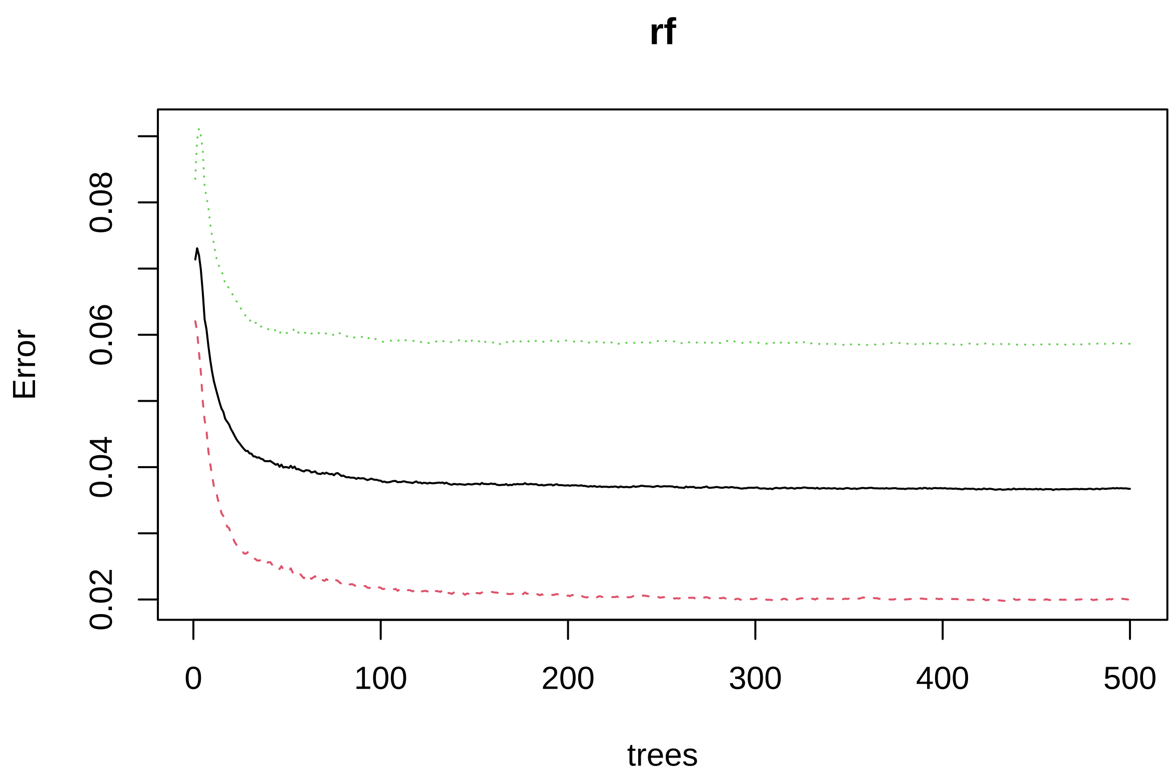
```
pred = predict(fit2.1, test_survey, type = "class")
ACC = mean(test_survey$satisfaction==pred)
AUC = roc.curve(test_survey$satisfaction,pred,plotit = F)$auc
c(ACC,AUC) %>% round(3)
```

```
## [1] 0.898 0.896
```

Random Forests

Bagging of trees 只是 random forest 在 m=p 時的特例，故在此只建構 random forest model

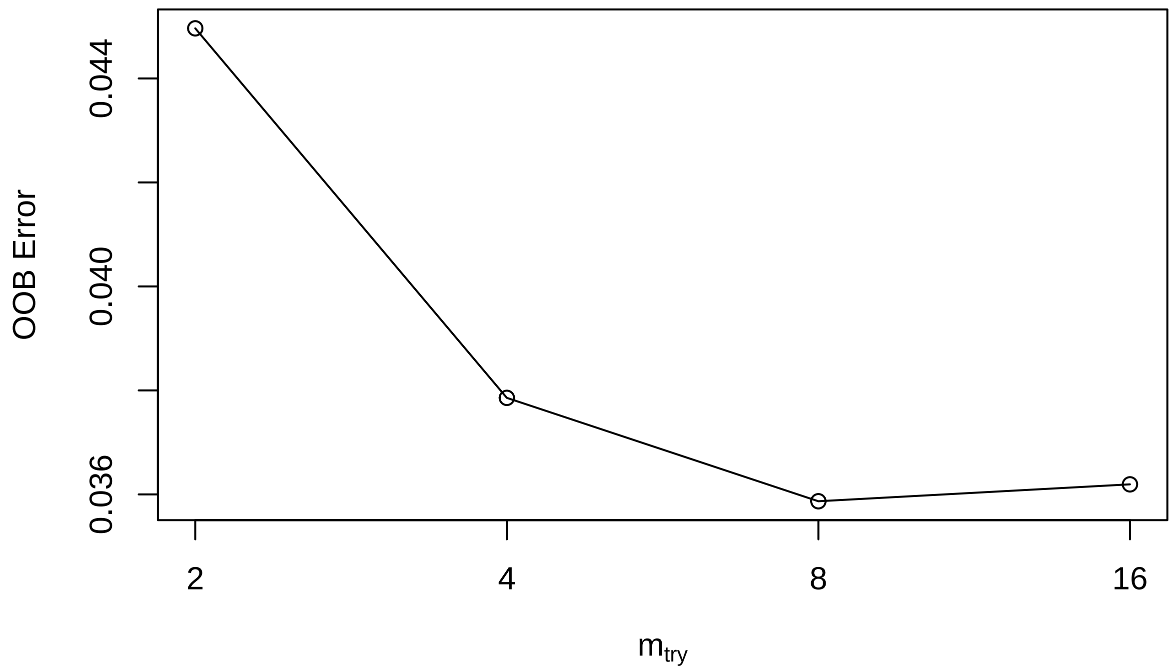
```r
rf = randomForest(satisfaction ~ ., train_survey)
plot(rf)
```



⇒ ntree > 200 後 error 呈現穩定

```r
set.seed(12122)
tuneRF(train_survey[,-23], train_survey[,23], ntreeTry = 200)
```

```
## mtry = 4  OOB error = 3.79%
## Searching left ...
## mtry = 2     OOB error = 4.5%
## -0.1877275 0.05
## Searching right ...
## mtry = 8     OOB error = 3.59%
## 0.0525221 0.05
## mtry = 16    OOB error = 3.62%
## -0.009055982 0.05
```




```
##        mtry   OOBError
## 2.OOB     2 0.04496329
## 4.OOB     4 0.03785657
## 8.OOB     8 0.03586826
## 16.OOB   16 0.03619308
```

⇒ 利用 OOB eror 決定參數 mtry = 8

設定參數 ntree = 200, ntry = 8 建構 RandomForests model，並對 testing model 進行預測且計算 ACC 和 AUC

```
fit2.2 = randomForest(satisfaction ~., train_survey, ntree = 200, mtry=8, importance=T)
pred = predict(fit2.2, test_survey, type = "class")
ACC = mean(test_survey$satisfaction==pred)
AUC = roc.curve(test_survey$satisfaction,pred,plotit = F)$auc
c(ACC,AUC) %>% round(3)
```

```
## [1] 0.969 0.968
```

Boosting

建構 boosting model，並利用 5-fold CV 選取 tuning parameter：n.trees, interaction.depth, shrinkage

```
set.seed(12101)
contrl = trainControl(method = "repeatedcv", number=5, repeats = 1)
boots = train(satisfaction ~ ., train_survey, method = "gbm",
              trControl = contrl, verbose=F)
boots
```

```
## Stochastic Gradient Boosting
##
## 101594 samples
##     22 predictor
##      2 classes: 'neutral or dissatisfied', 'satisfied'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 1 times)
## Summary of sample sizes: 81275, 81275, 81276, 81275, 81275
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa
##   1                   50      0.8819418  0.7587004
##   1                  100      0.9121011  0.8197695
##   1                  150      0.9238537  0.8442407
##   2                   50      0.9189322  0.8343116
##   2                  100      0.9307636  0.8585359
##   2                  150      0.9339725  0.8651519
##   3                   50      0.9274760  0.8519915
##   3                  100      0.9346812  0.8666104
##   3                  150      0.9399768  0.8774254
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were n.trees = 150, interaction.depth =
```

```
##  3, shrinkage = 0.1 and n.minobsinnode = 10.
```

⇒ 選定參數：ntree=150, interaction.depth=3, shrinkage=0.1

以上述選定的參數建構 Boosting model 並對 testing data 進行預測且計算 ACC 和 AUC

```
fit2.3 = gbm(I(satisfaction=="satisfied") ~., data=train_survey, n.trees = 150,
           distribution = "bernoulli", interaction.depth = 3, shrinkage = 0.1)
pred = predict(fit2.3, test_survey, type = "response")
```

```
## Using 150 trees...
```

```
pred_class = ifelse(pred>0.5,"satisfied","neutral or dissatisfied")
ACC = mean(test_survey$satisfaction==pred_class)
AUC = roc.curve(test_survey$satisfaction,pred_class,plotit = F)$auc
c(ACC,AUC) %>% round(3)
```

```
## [1] 0.944 0.942
```

Performance comparison

	Tree	Random Forests	Boosting
ACC	0.898	0.970	0.946
AUC	0.896	0.968	0.944

⇒ Random Forests model 在 ACC 和 AUC 上的表現最佳

觀察 Random Forests 模型的重要解釋變數

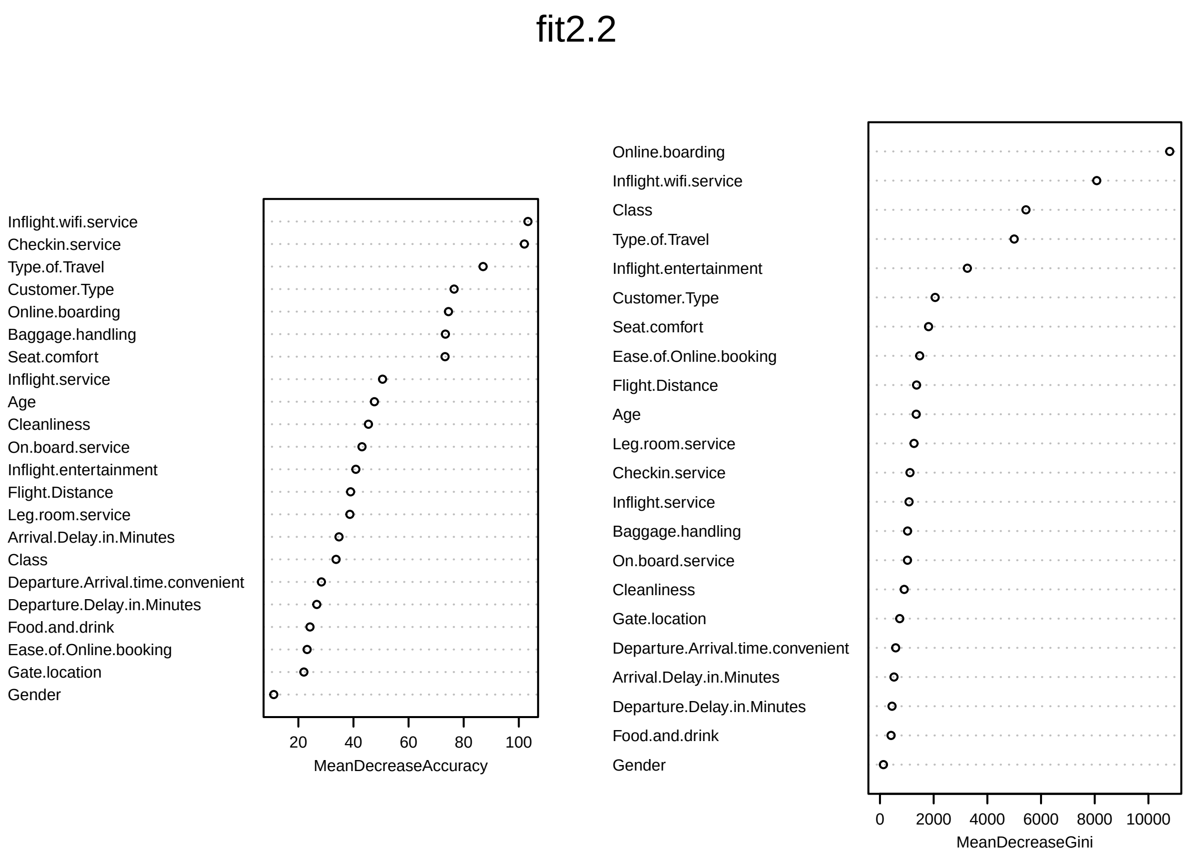
```
importance(fit2.2)
```

```
##                                     neutral or dissatisfied  satisfied
## Gender                                             10.23119   6.523206
## Customer.Type                                      47.40289  92.960158
## Age                                                36.07735  36.854078
## Type.of.Travel                                     53.30188 145.010659
## Class                                              28.22788  30.681970
```

```
## Flight.Distance                             25.38268  31.380118
## Inflight.wifi.service                      167.57065  51.974720
## Departure.Arrival.time.convenient           30.20996  18.848181
## Ease.of.Online.booking                      18.42638  23.908916
## Gate.location                               26.68138  20.246859
## Food.and.drink                              13.12033  25.990484
## Online.boarding                             64.17580  50.868922
## Seat.comfort                                63.21041  40.837640
## Inflight.entertainment                      31.46400  27.260744
## On.board.service                            44.31448  21.250631
## Leg.room.service                            36.77020  27.554057
## Baggage.handling                            65.40479  28.184870
## Checkin.service                             90.35948  42.617513
## Inflight.service                            65.65949  14.173065
## Cleanliness                                 36.04765  31.969213
## Departure.Delay.in.Minutes                  21.66875  11.315160
## Arrival.Delay.in.Minutes                    25.69945  25.952244
##                                   MeanDecreaseAccuracy MeanDecreaseGini
## Gender                                        11.07344         128.9260
## Customer.Type                                 76.51663        2055.4948
## Age                                           47.58947        1350.5424
## Type.of.Travel                                87.03393        5000.0690
## Class                                         33.69211        5438.5422
## Flight.Distance                               38.95668        1363.0022
## Inflight.wifi.service                        103.30963        8075.0826
## Departure.Arrival.time.convenient             28.36737         584.2968
## Ease.of.Online.booking                        23.18315        1478.8259
## Gate.location                                 22.00786         735.2294
## Food.and.drink                                24.22428         413.4067
## Online.boarding                               74.47639       10792.8938
## Seat.comfort                                  73.23281        1809.7969
## Inflight.entertainment                        40.85662        3255.7777
## On.board.service                              43.09728        1025.0434
## Leg.room.service                              38.69369        1268.1415
## Baggage.handling                              73.35833        1029.0840
## Checkin.service                              102.01852        1118.1636
```

```
## Inflight.service                    50.55683        1083.4860
## Cleanliness                         45.43543         903.0098
## Departure.Delay.in.Minutes          26.68014         449.1059
## Arrival.Delay.in.Minutes            34.74999         522.1362
```

```
varImpPlot(fit2.2, cex=0.5)
```



藉由 Decrease in ACC 可以選取出前幾個最重要的解釋變數：Inflight.wifi.service, Checkin.service, Online.boarding, Type.of.Travel, Seat.comfort, Customer.Type, Baggage.handling
藉由 Decrease in Gini 可以選取出錢幾個最重要的解釋變數：Online.boarding, Inflight.wifi.service, Type.of.Travel, Class

但是這種決定重要解釋變數個數的方法很主觀，沒辦法確定後面應該留幾個變數當作不重要變數，以下可以利用在模型中加入 random noise factor 的方式來輔助。

Add noise factor

加入兩個 noise factor：

- $z_1 \sim N(0,1)$

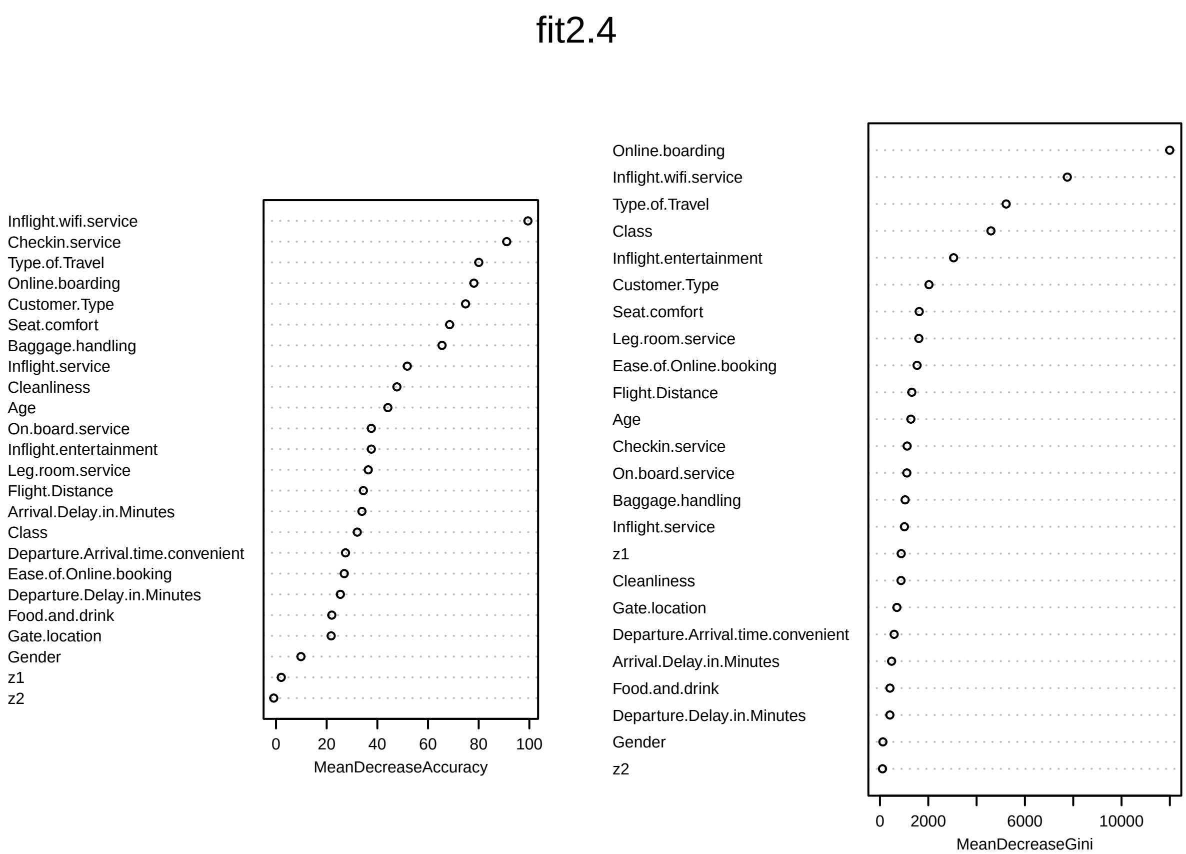
- $z_2 \sim Ber(p = 0.5)$

```
set.seed(12126)
n = dim(survey1)[1]
survey2 = survey1 %>%
    mutate(z1 = rnorm(n), z2 = as.factor(rbinom(n,1,0.5)))
```

對加入 noise factor 的資料建構 Random Forests model，一樣觀察其重要解釋變數

```
fit2.4 = randomForest(satisfaction ~ ., survey2, ntree=200, mtry = 8, importance=T)
```

```
varImpPlot(fit2.4, cex = 0.5)
```



藉由 Decrease in Gini 可以看出有數個變數重要性比 z_1 還要低，那麼我們就可以確定這些變數是不重要變數，而其餘剩餘的就是重要解釋變數。

Suggestions

上述重要解釋變數中，有一部分是顧客的個人訊息，如：*Type.of.Travel, Class* 等，這些變數是航空公司無法控制的，故我們只能針對 *Online.boarding, Inflignt.wifi.service, Inflight.entertainment, Seat.comfort, Leg.room.service, Ease.of.Online.booking, Checkin.service, On.board.service, Baggage.handling, Inflight.service* (重要度高到低) 等變數給建議：

- 你們可以先增進公司的軟體系統及電子設備，像是：線上登機系統 (*Online.boarding*)、飛行途中無線網路服務 (*Inflignt.wifi.service*)、飛行途中娛樂設施 (*Inflight.entertainment*)，這會大幅提高顧客滿意比例。

- 再來改善你們飛機座位的舒適程度 (*Seat.comfort*) 和座位放置腳的空間 (*Leg.room.service*)。
- 最後可以再多注意你們公司在 Checkin (*Checkin.service*), On board (*On.board.service*), Inflight (*Inflight.service*) 各階段的服務品質。