

Python 與監督式學習

David Chiu
2016/03/19



機器學習 (Machine Learning)

電腦產生預測



預測分析

■ 預測分析 (Predictive Analysis)

- 從經驗中學習，預測個人未來行為已便做出更佳決定

■ 要預測什麼？

- 個人或特定行為（例如：行動、事件或發生狀況）
- e.g. 每位顧客最可能點擊哪則廣告？

■ 要做什麼用？

- 根據預測所做的決定，組織回應各項預測，或從預測得知應採取行動
- e.g. 依據點擊可能性及廣告商每次點擊的付費金額，秀出效益最佳的廣告

預測可能得到出乎意料的結論

- 買尿布的顧客也有可能買啤酒
 - 因為大部分是男性去買的？
- 上網查費率方案的用戶容易更換手機業者
 - 因為客戶提前解約有違約金要付，所以會先觀看費率，提醒自己何時該更改費率或業者？
- 同個朋友圈的人都會用同一家手機電信公司(社交效應)
 - 若意見領袖更換手機電信業者，他的朋友們更換業者的機率高達七倍
 - 受到網內互打的省錢方案影響？

預測模型

- 根據個人行為的特定機制 (是否點擊廣告)，預測模型把個人特性(身高，性別，職業)當輸入資料，提供預測分數當產出資料，分數越高，產生該行為的機會越高。



簡單的預測模型

■ 線性模型

- e.g. 針對一個化妝品廣告，對女性的吸引力可以給予權重90%，男性權重只有10%，以權重搭配個人點擊機率 (15%) 可以算出對該用戶推薦的分數(或機率)
⇒ 女性13.5%，男性1.5%

■ 規則模型

- e.g.
 - 假使使用者為女性
 - 而且月收入高達3萬以上
 - 而且還沒看過這廣告
 - 點擊機率為11%

機器學習

■ 機器學習的目的是：歸納 (Induction)

□ 從詳細事實到一般通論

A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E

-- Tom Mitchell (1998)

■ 找出有效的預測模型

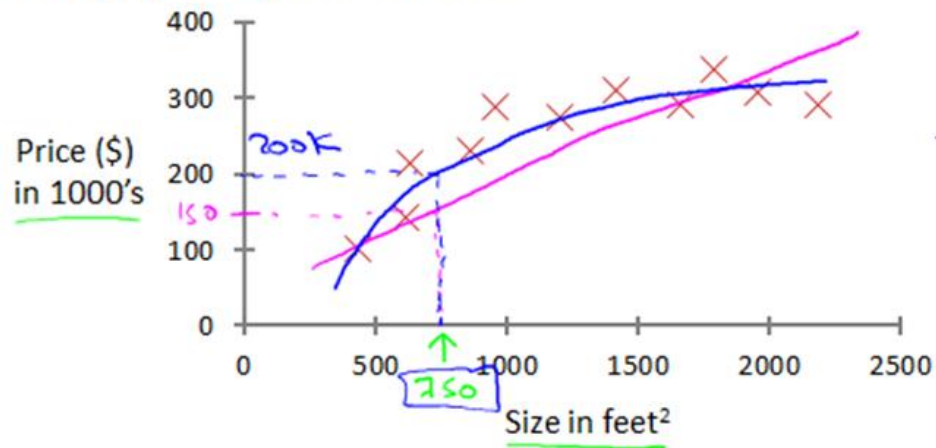
- 一開始都從一個簡單的模型開始
- 藉由不斷餵入訓練資料，修改模型
- 不斷提升預測績效

機器學習問題分類

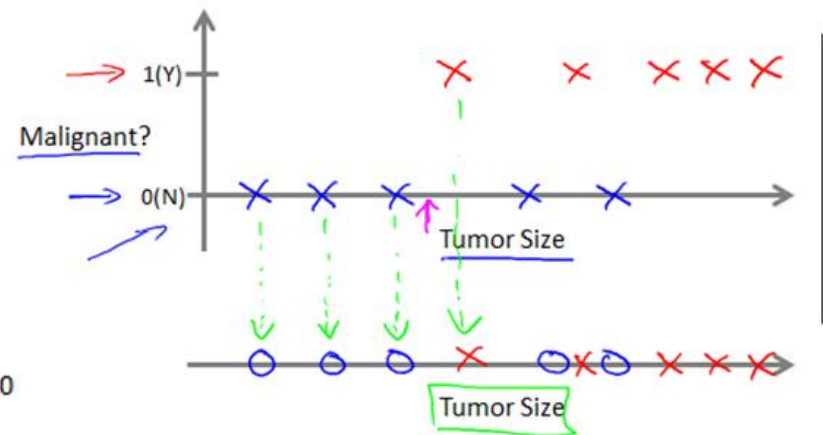
- 監督式學習 (Supervised Learning)
 - 迴歸分析 (Regression)
 - 分類問題 (Classification)
- 非監督式學習 (Unsupervised Learning)
 - 降低維度 (Dimension Reduction)
 - 分群問題 (Clustering)

監督式學習

Housing price prediction.



Breast cancer (malignant, benign)



監督式學習

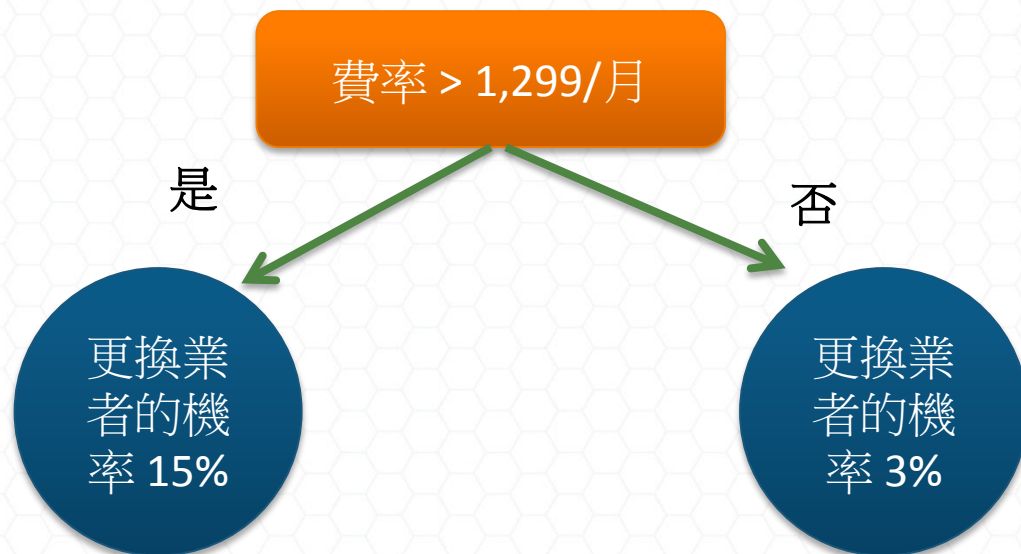
■ 分類問題

- 根據已知標籤的訓練資料集(Training Set)，產生一個新模型，用以預測測試資料集(Testing Set)的標籤。
- e.g. 股市漲跌預測

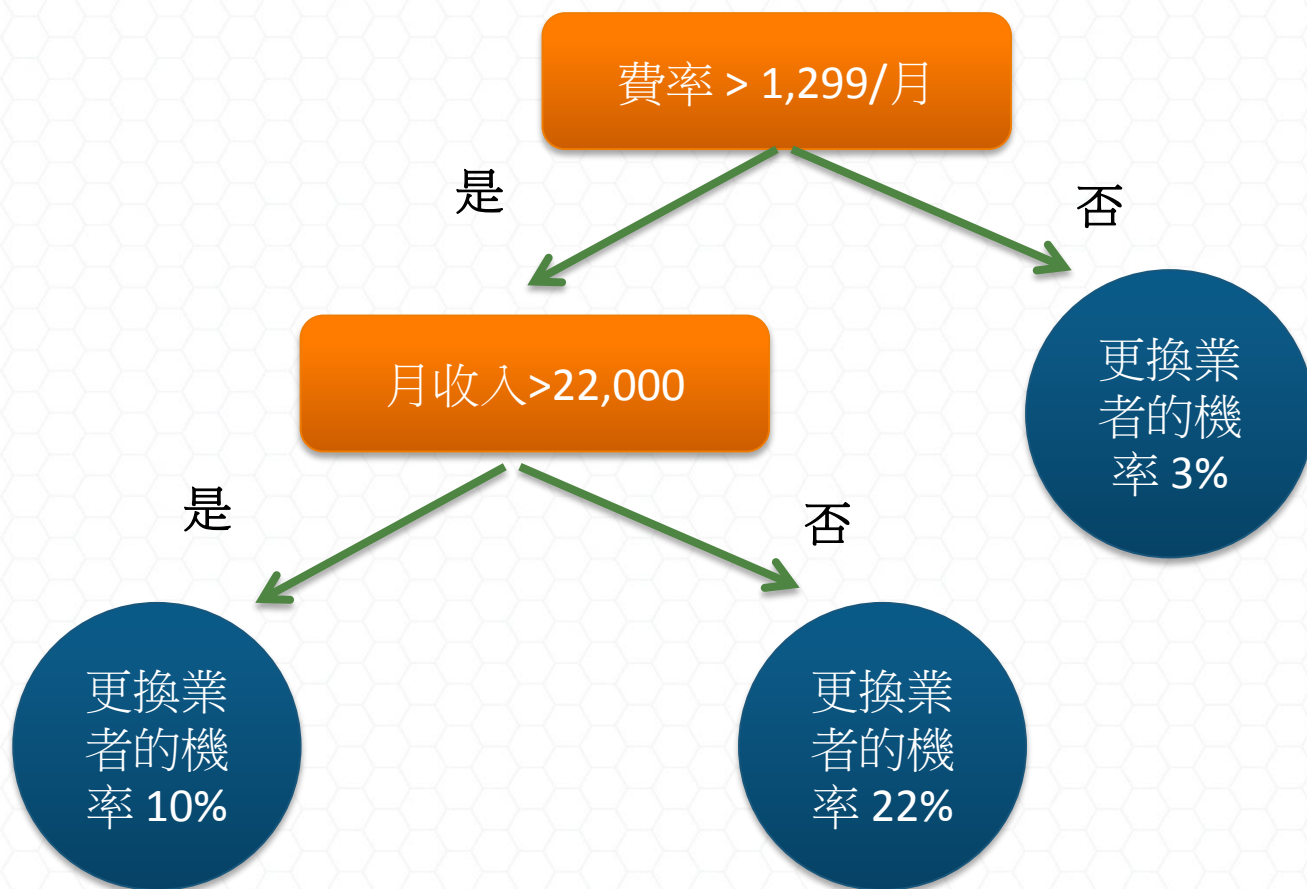
■ 迴歸分析

- 使用一組已知對應值的數據產生的模型，預測新數據的對應值
- e.g. 股價預測

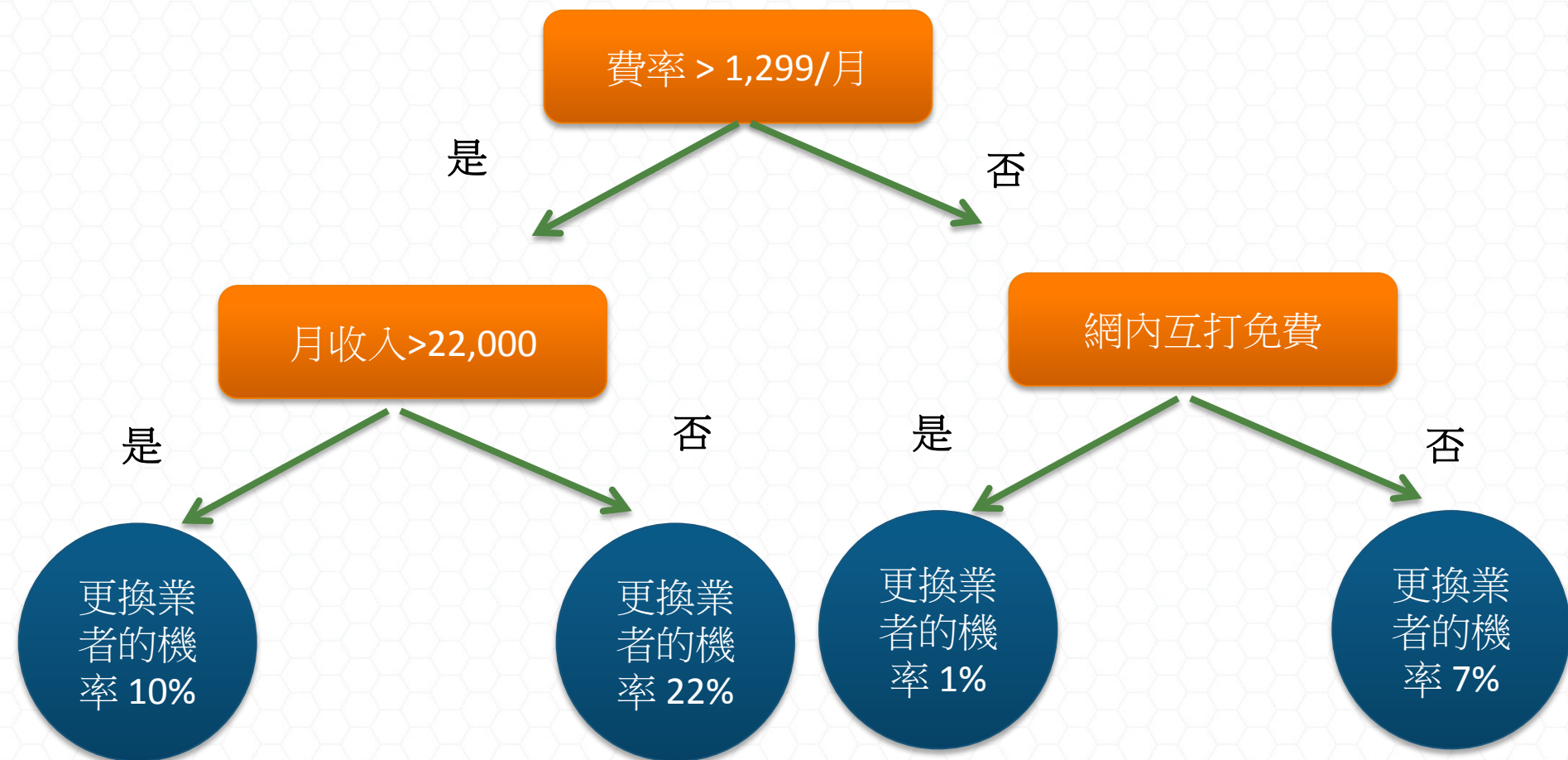
簡單的分類問題(決策樹)



簡單的分類問題(決策樹)



簡單的分類問題(決策樹)



那是不是樹越長越大，預測效果越好？

預測績效

■ 增益值 (Lift)

- 當預測績效越好，所要的成本越高，報酬遞減

■ 決策越多種類，要設計留住客戶的行銷策略是否越多？

- 為不同客戶種類 (Persona) 設計行銷策略

- 4種通路的行銷成本，效益多高？

- 20種通路的行銷成本，效益多高？

過度學習

- 奶油生產速度可以預測標普500的收盤價？



過度學習

■ 誤把雜訊當資訊

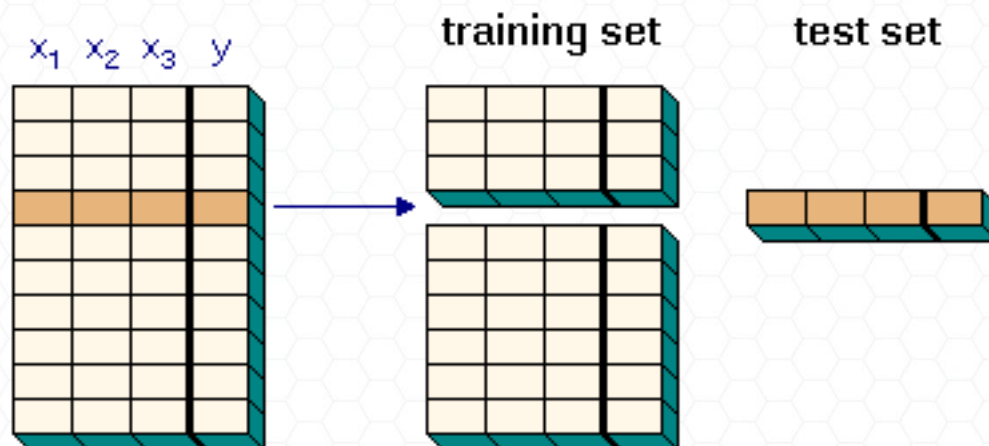
- 過度假設
- 過度解讀
- 無法找出資料背後的事實

■ 過度學習所得到的規則並非通則，只能應用於個案

- 死記答案 v.s. 掌握原則

測試模型

使用外部資料或是一部分的內部資料來測試資料





NUMPY

NUMPY

- 提供了快速的多維數組處理的能力
- 將常用的數學函數都進行數組化，使得這些數學函數能夠直接對數組進行操作
- 讓Python 可以像Matlab 一樣操作資料



引用 NUMPY

```
import numpy  
numpy.version.full_version
```

使用array

```
import numpy as np  
a = np.array([0,1,2,3,4,5])  
print a, a.ndim, a.shape
```

轉為二維陣列

```
b = a.reshape((3,2))  
print b, b.ndim, b.shape
```


會指到同樣的位址

```
b[1][0] = 77
```

```
print b
```

```
print a
```

使用copy

```
c = a.reshape((3,2)).copy()
```

```
print c
```

```
c[0][0] = -99
```

```
print a
```

```
print c
```

對數組進行操作

```
print a *2
```

```
print a **2
```

```
b = [1,2,3,4,5]
```

```
print b * 2
```

```
print b ** 2
```


資料篩選

```
a[np.array([2,3,4])]
```

```
print a > 4
```

```
print a[a>4]
```

```
a[a>4] = 4
```

```
print a
```

```
a.clip(0,4)
```

去除NAN

```
c = np.array([1,2,np.NAN, 3, 4])  
print c  
print np.isnan(c)  
print c[-np.isnan(c)]  
print np.mean(c[-np.isnan(c)])
```

可放入不同物件

```
np.array([1,"string"])
```

```
np.array([1,"string", set([1,2,3])])
```




SCIPY

SCIPY

- SciPy在NumPy基礎上添加了眾多的科學計算所需的各種套件
- 可以讓Python 像 Matlab 一樣進行基本的機器學習



引用 **SCIPY**

```
import scipy  
scipy.version.version
```


線性迴歸

- 線性迴歸是研究**單一依變項**(dependent variable)與一個或以上**自變項**(independent variable)之間的關係
- 線性迴歸有兩個主要用處：
 - **預測**指的是用已觀察的變數來預測依變項
 - **因果分析**則是將自變項當作是依變項發生的原因

簡單線性迴歸

■ 數學模型

□ $y = \beta_1 x + \beta_0 + \epsilon$

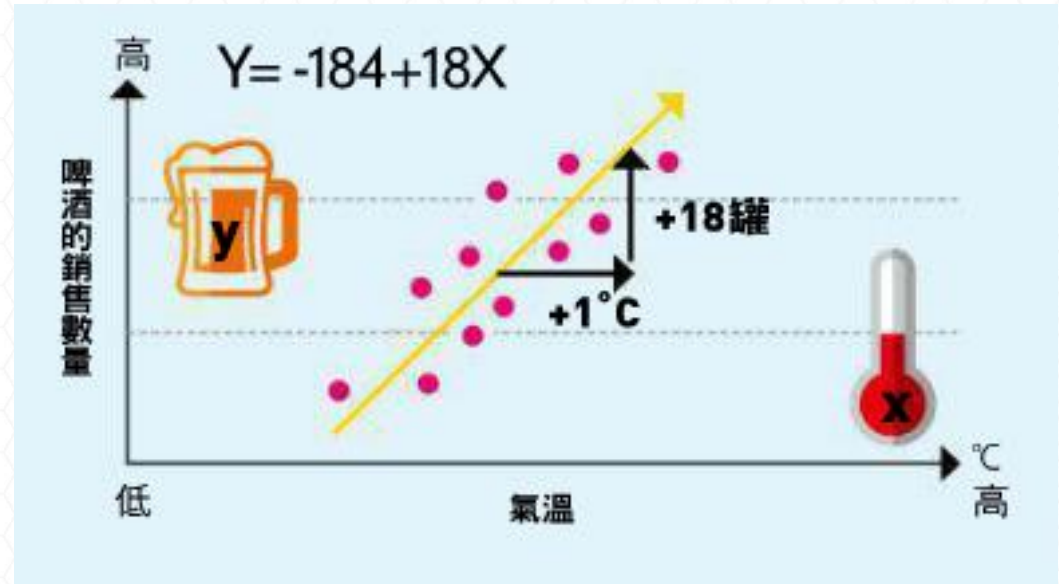
□ y 是依變數

□ x 是自變數

□ b_i 是迴歸係數

□ b_0 是截距

□ ϵ 是誤差

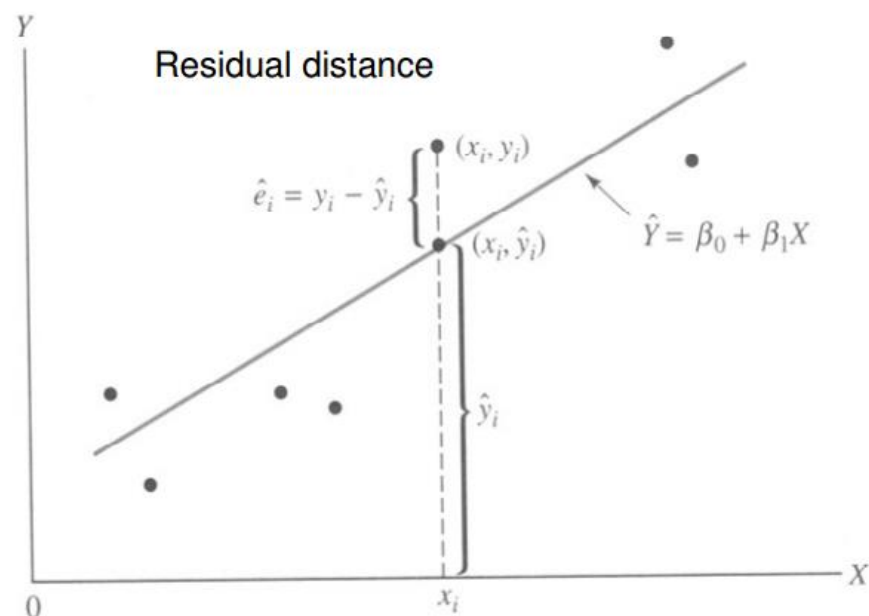


最小平方估計法 - OLS

■ 找出殘差平方和最小的一條線

▣ 殘差 $e_i = (y_i - \hat{y}_i)$

▣ 殘差平方和 $SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2$



計算迴歸係數

X (kilos)	Y (cost, \$)
17	132
21	150
35	160
39	162
50	149
65	170

■ 計算兩變數之間的迴歸係數

```
> X = c(17, 21, 35, 39, 50, 65)
```

```
> Y = c(132, 150, 160, 162, 149,  
170)
```

```
> X_avg = mean(X)
```

```
> Y_avg = mean(Y)
```

```
> B1 = (sum(X*Y) - 6 * X_avg *  
Y_avg)/(sum(X^2) - 6 * X_avg ^2)
```

$$\hat{\beta}_1 = \frac{\sum XY - n\bar{X}\bar{Y}}{\sum X^2 - n\bar{X}^2}$$

計算截距

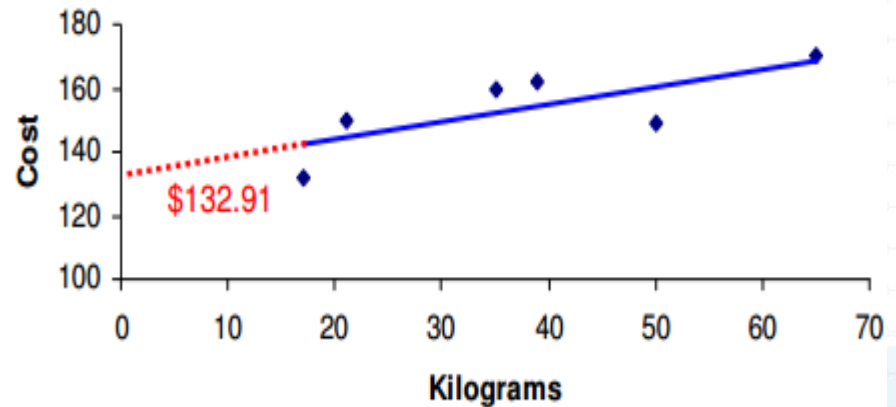
■ 計算截距

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

$$> B0 = Y_avg - B1 * X_avg$$

■ 得到迴歸公式

$$y = 0.553x + 132.91$$



讀入網頁流量數據

```
import os  
import scipy as sp  
import matplotlib.pyplot as plt  
  
data = sp.genfromtxt("web_traffic.tsv",  
delimeter="\t")  
print(data[:10])
```

檢視資料

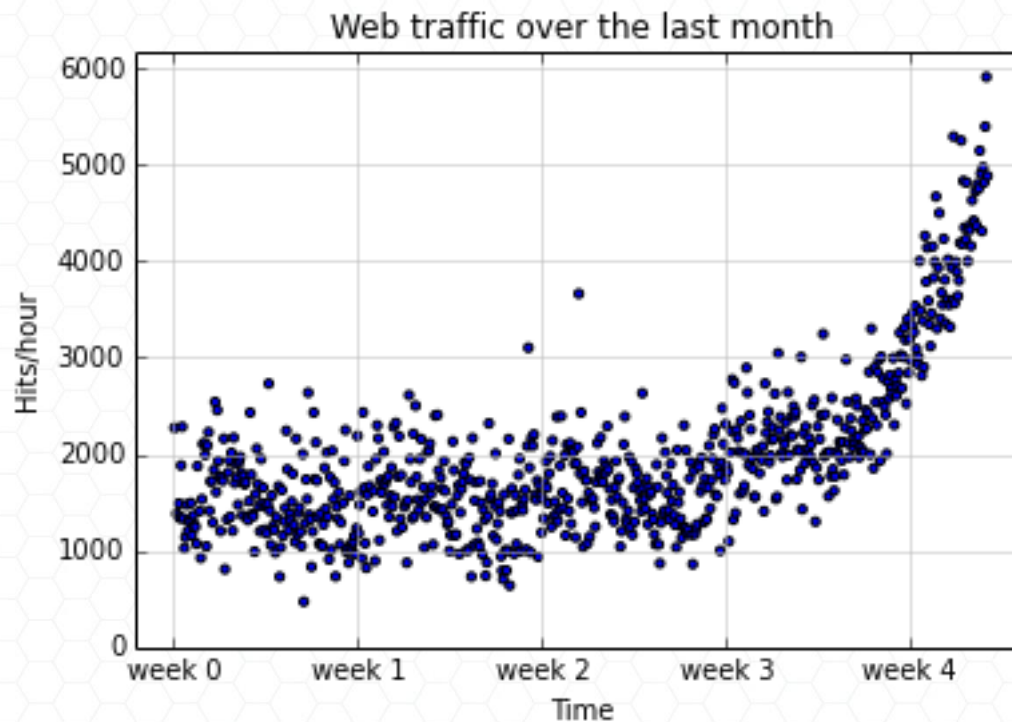
```
# fix invalid entries  
x = data[:, 0]  
y = data[:, 1]  
print("Number of invalid entries:",  
      sp.sum(sp.isnan(y)))  
x = x[~sp.isnan(y)]  
y = y[~sp.isnan(y)]
```

指定繪圖函式

```
def plot_models(x, y, models, mx=None, ymax=None, xmin=None):
    plt.clf()
    plt.scatter(x, y, s=10)
    plt.title("Web traffic over the last month")
    plt.xlabel("Time")
    plt.ylabel("Hits/hour")
    plt.xticks(
        [w * 7 * 24 for w in range(10)], ['week %i' % w for w in range(10)])
    if models:
        if mx is None:
            mx = sp.linspace(0, x[-1], 1000)
        for model, style, color in zip(models, linestyle, colors):
            # print "Model:", model
            # print "Coeffs:", model.coefs
            plt.plot(mx, model(mx), linestyle=style, linewidth=2, c=color)
        plt.legend(["d=%i" % m.order for m in models], loc="upper left")
    plt.autoscale(tight=True)
    plt.ylim(ymin=0)
    if ymax:
        plt.ylim(ymax=ymax)
    if xmin:
        plt.xlim(xmin=xmin)
    plt.grid(True, linestyle='-', color='0.75')
```


繪圖

```
# first look at the data  
plot_models(x, y, None)
```



R Square

- 為相關係數(correlation)的平方值，其值介於0與1之間
 - ▣ R Square越接近1，表示自變數與依變數相關性越高
 - ▣ 越接近0，表示自變數與依變數相關性越低
 - ▣ 迴歸可以解釋的變異比例，可作為自變數預測依變數準確度的指標

模型評估

■ Relative Mean Square Error

□ 評估使用同單位的模型

□
$$RMSE = \sqrt{\frac{\sum_{i=1}^n (p_i - a_i)^2}{n}}$$

■ Relative Square Error

□ 評估使用不同單位的模型

□
$$RSE = \frac{\sum_{i=1}^n (p_i - a_i)^2}{\sum_{i=1}^n (\bar{a} - a_i)^2}$$

R Square

- 迴歸可以解釋的變異比例，可作為自變數預測依變數準確度的指標

$$\text{Coefficient of Determination} \rightarrow R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

$$\text{Sum of Squares Total} \rightarrow SST = \sum (y - \bar{y})^2$$

$$\text{Sum of Squares Regression} \rightarrow SSR = \sum (y' - \bar{y}')^2$$

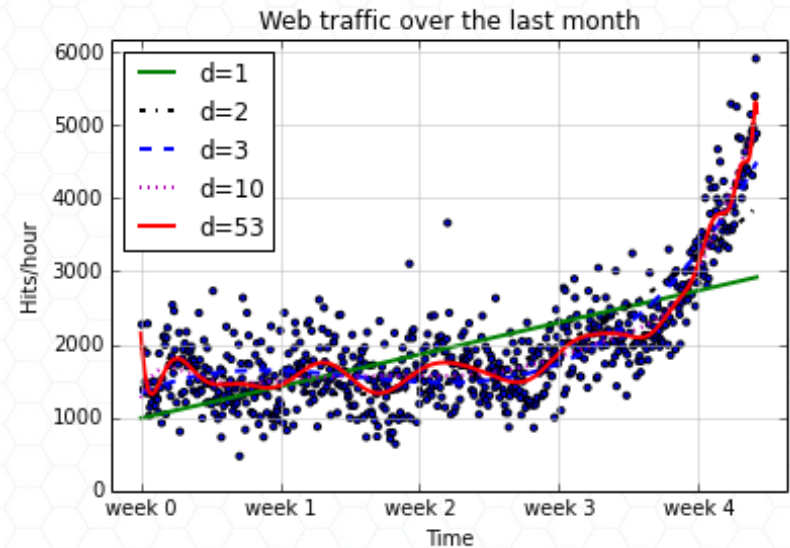
$$\text{Sum of Squares Error} \rightarrow SSE = \sum (y - y')^2$$

計算error rate 及模型參數

```
# obtain model parameter and error rate  
fp1, res, rank, sv, rcond = sp.polyfit(x, y, 1,  
full=True)  
print("Model parameters: %s" % fp1)  
print("Error of the model:", res)
```

繪圖

```
# plot data with different polyfit  
f1 = sp.poly1d(fp1)  
f2 = sp.poly1d(sp.polyfit(x, y, 2))  
f3 = sp.poly1d(sp.polyfit(x, y, 3))  
f10 = sp.poly1d(sp.polyfit(x, y, 10))  
f100 = sp.poly1d(sp.polyfit(x, y, 100))  
  
plot_models(x, y, [f1])  
plot_models(x, y, [f1, f2])  
plot_models(x, y, [f1, f2, f3, f10, f100])
```



考量轉折點

```
# using inflection point for plotting
```

```
inflection = 3.5 * 7 * 24
```

```
xa = x[:inflection]
```

```
ya = y[:inflection]
```

```
xb = x[inflection:]
```

```
yb = y[inflection:]
```

```
fa = sp.poly1d(sp.polyfit(xa, ya, 1))
```

```
fb = sp.poly1d(sp.polyfit(xb, yb, 1))
```

```
plot_models(x, y, [fa, fb])
```




SCIKIT LEARN

Scikit learn

The image shows the top section of the Scikit-learn website. On the left is the Scikit-learn logo, which consists of a blue circle and the text "scikit learn". To the right of the logo is a navigation bar with orange buttons for "Home", "Installation", "Documentation", and "Examples". Further right is a search bar with the text "Google™ Custom Search" and a "Search" button. Below the navigation bar is a large blue banner. On the left side of the banner is a grid of 15 small plots showing various data distributions and model results. On the right side of the banner, the text "scikit-learn" is written in a large, white, sans-serif font, followed by "Machine Learning in Python" in a smaller, white, sans-serif font. Below this text is a list of four bullet points in white text.

scikit-learn
Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which set of categories a new observation belong to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous value for a new example.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to

Model selection

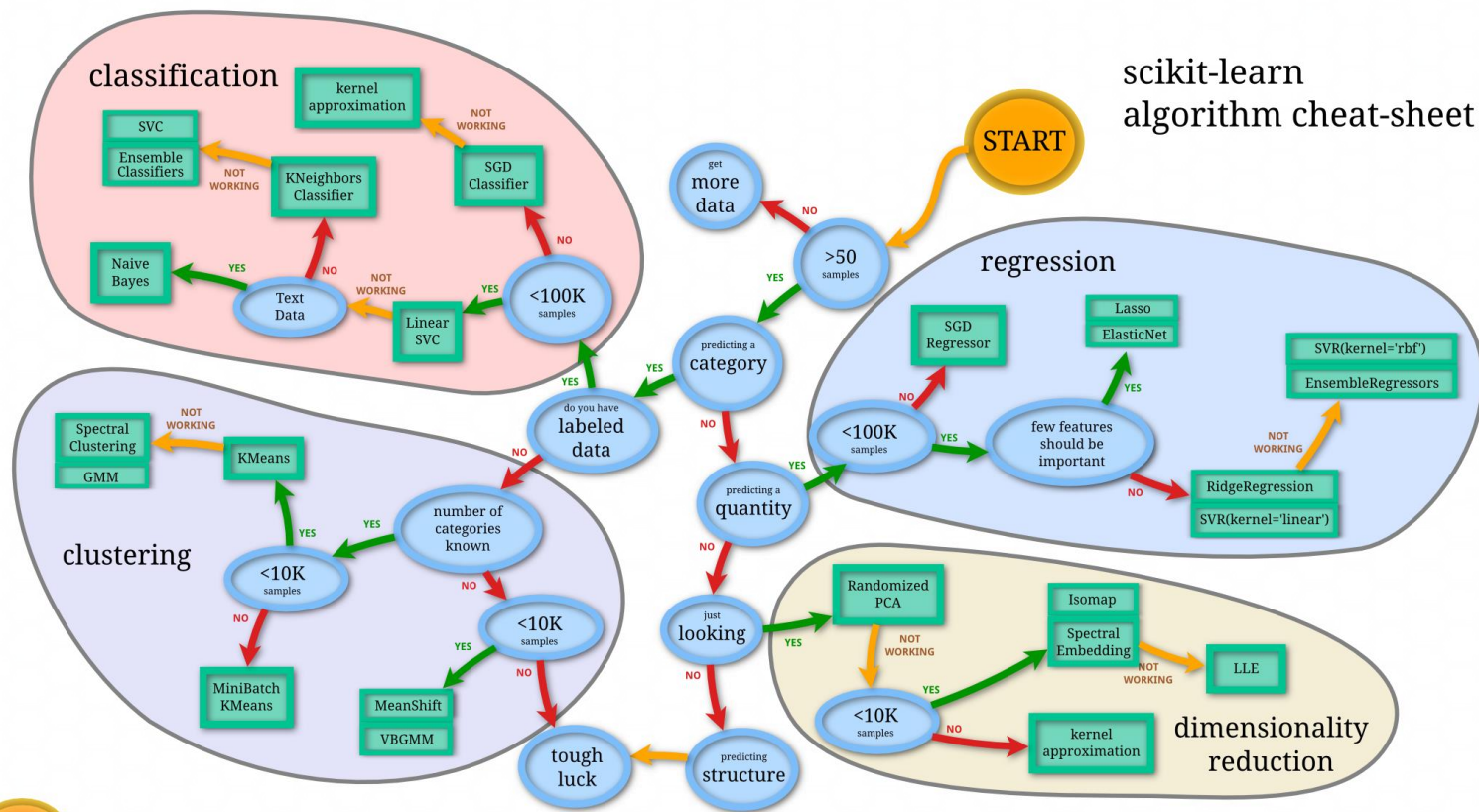
Comparing, validating and choosing

Preprocessing

Feature extraction and normalization.

TableauDesktop-6....exe

學習地圖



Estimators

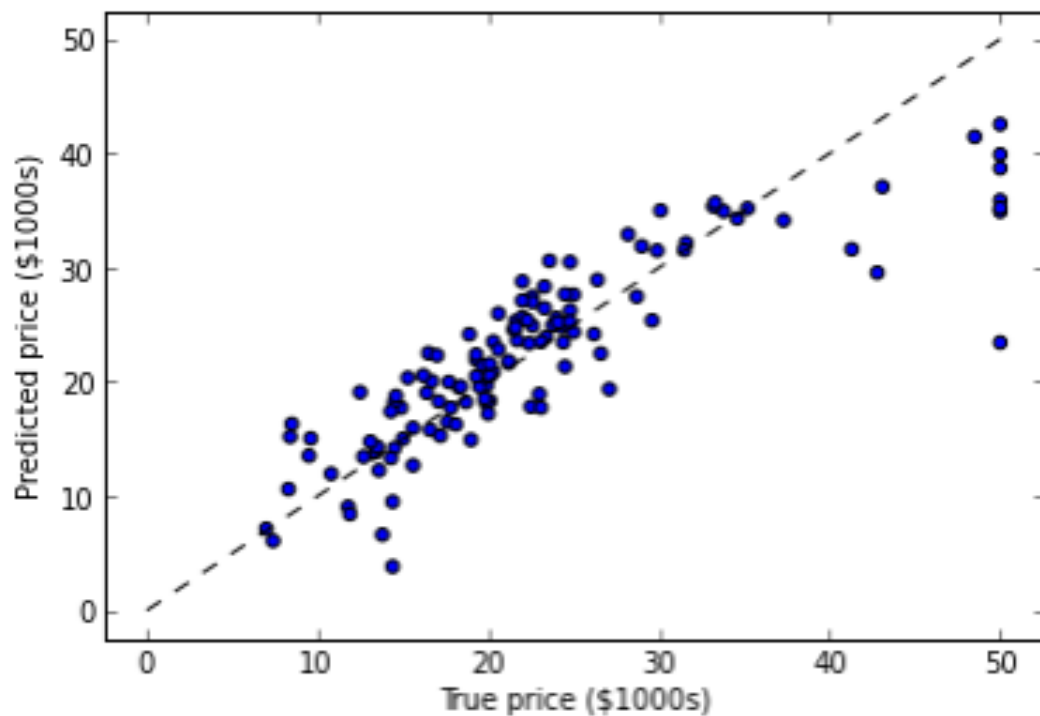
- 可以藉由學習訓練資料集的屬性產生預測能力
- 使用於分類演算法或是回歸分析
- 所有estimators 都有實現fit 函式:
 - `estimator.fit(X, y)`

引用scikit learn

```
# Quick example of building an estimator
from sklearn.linear_model import LinearRegression
model = LinearRegression(normalize=True) # Can set estimator's
parameter while instantiated
print model.normalize
print model
```


回歸分析 - 預測房價

使用回歸分析預測波士頓房價



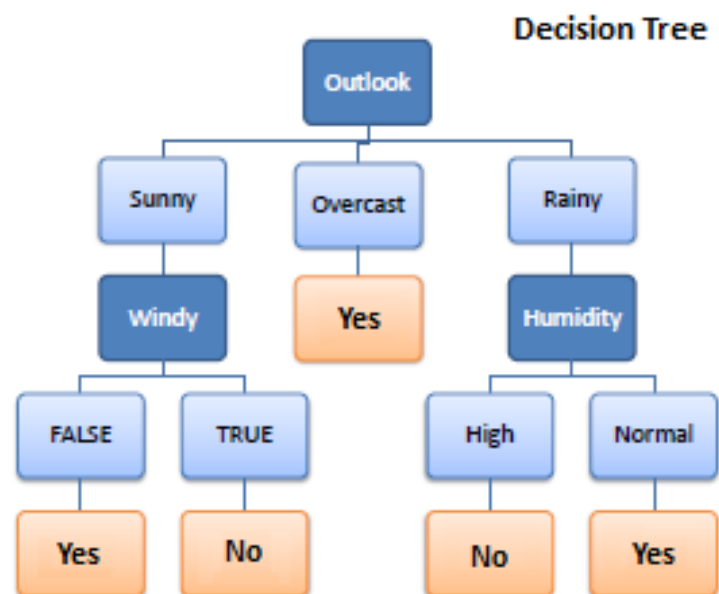
讀取內部資料集

```
# A sample code to get dataset from scikit-learn  
from sklearn import datasets  
# Load Dataset iris  
iris = datasets.load_iris()  
iris.data[1:5], iris.target[1:5]
```

決策樹分類

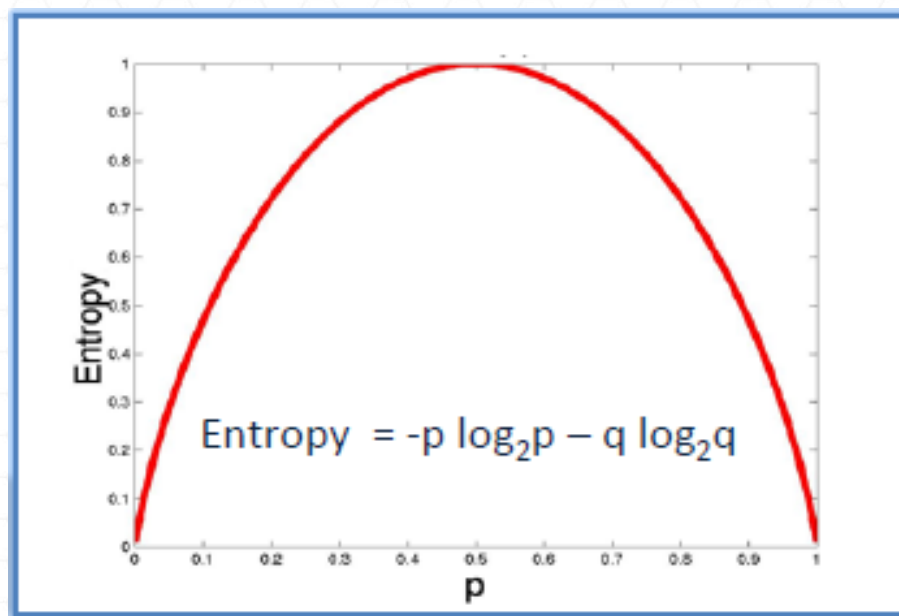
決策樹

Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



Entropy

- 用於計算一個系統中的失序現象，也就是計算該系統混亂的程度



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

單一變數的計算

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5



Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 \log_2 0.36) - (0.64 \log_2 0.64)
= 0.94

多變數的計算

$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14



$$\begin{aligned} E(\text{PlayGolf}, \text{Outlook}) &= P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3) \\ &= (5/14) * 0.971 + (4/14) * 0.0 + (5/14) * 0.971 \\ &= 0.693 \end{aligned}$$

Information Gain

- 根據分割(Split)後，所減少的Entropy
- 因此做分割時，會尋找最大的Information Gain

1. 計算Entropy

$$\begin{aligned}\text{Entropy}(\text{PlayGolf}) &= \text{Entropy}(5,9) \\ &= \text{Entropy}(0.36, 0.64) \\ &= - (0.36 \log_2 0.36) - (0.64 \log_2 0.64) \\ &= 0.94\end{aligned}$$

計算Information Gain

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			


		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

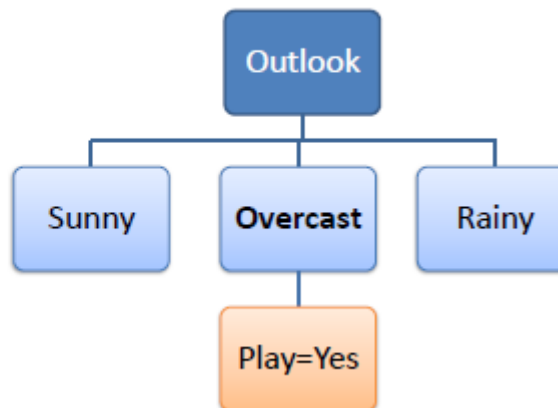
$$\begin{aligned}
 G(\text{PlayGolf}, \text{Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook}) \\
 &= 0.940 - 0.693 = 0.247
 \end{aligned}$$

選擇有最大Information Gain的屬性

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

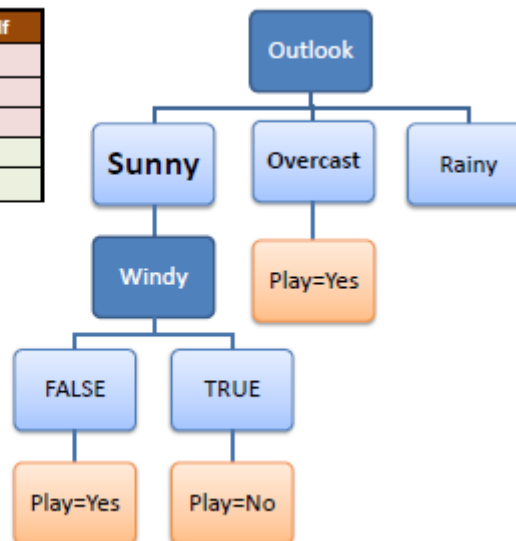
選擇子節點與分割節點

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes
Hot	High	FALSE	Yes



Entropy 為 0
則為子節點

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Entropy 非 0
則為分割節點

決策樹如同IF...ELSE

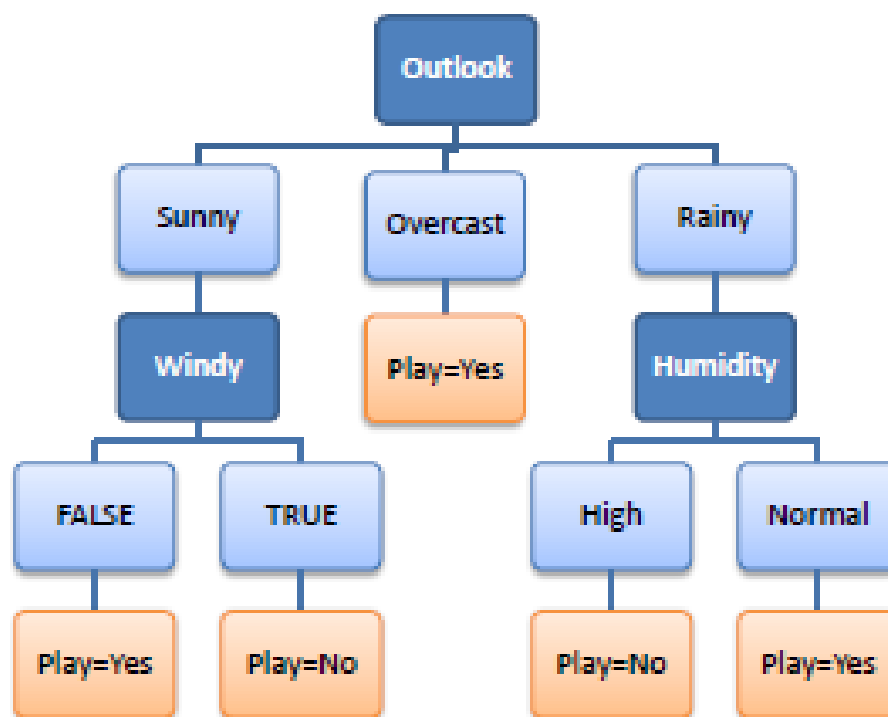
R_1 : IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2 : IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3 : IF (Outlook=Overcast) THEN Play=Yes

R_4 : IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5 : IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes



分類問題 - 分類花種

- 以Sepal.Length, Sepal.Width, Petal.Length, Petal.Width 預測花種

- Iris 資料集

- http://en.wikipedia.org/wiki/Iris_flower_data_set



Iris setosa



Iris versicolor



Iris virginica

使用決策樹分類

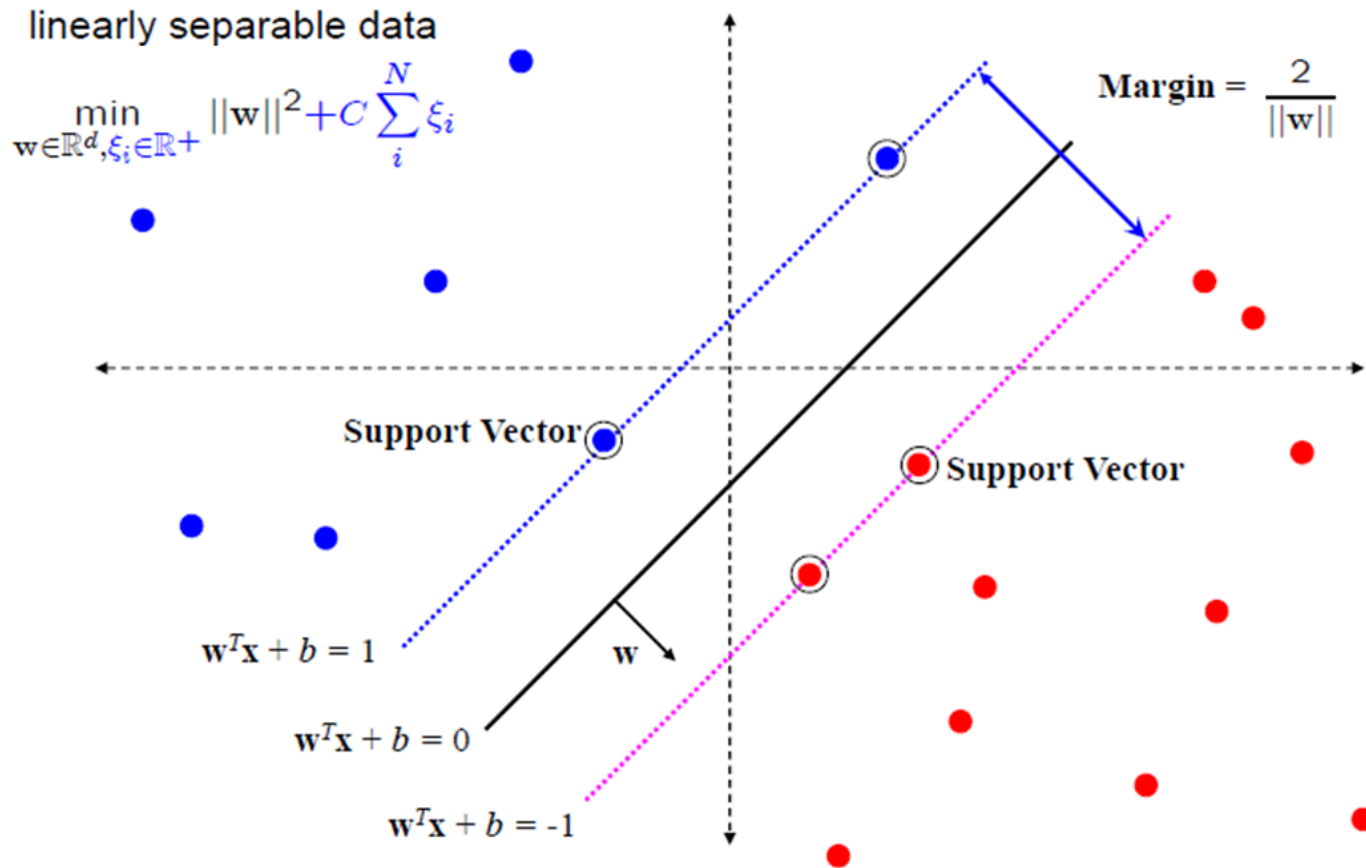
```
from sklearn.datasets import load_iris
from sklearn import tree
iris = load_iris()
clf = tree.DecisionTreeClassifier()
clf = clf.fit(iris.data, iris.target)
```

繪製Decision Tree

```
from IPython.display import Image
dot_data = StringIO()
tree.export_graphviz(clf, out_file=dot_data,
                     feature_names=iris.feature_names,
                     class_names=iris.target_names,
                     filled=True, rounded=True,
                     special_characters=True)
graph = pydot.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```


SVM 分類器

SVM 分類器

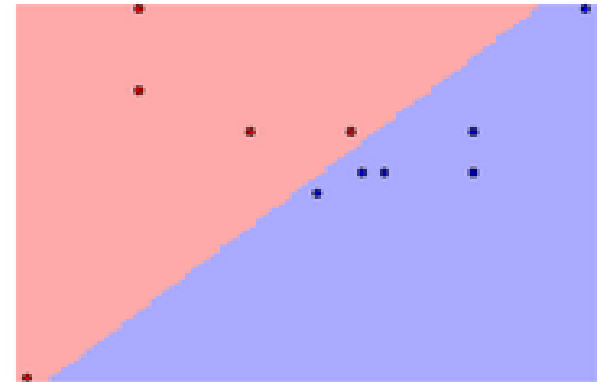
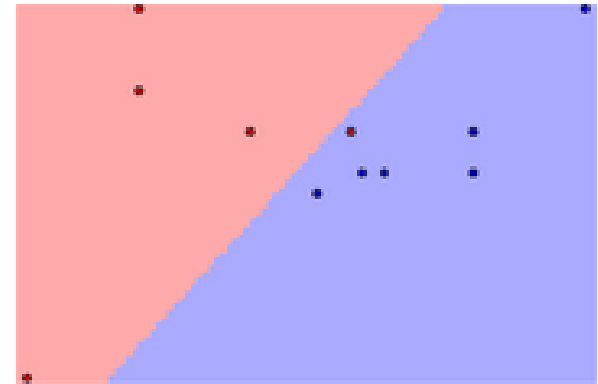


選擇SVM 的 Cost

■ Cost 大小會決定該分類的容忍度

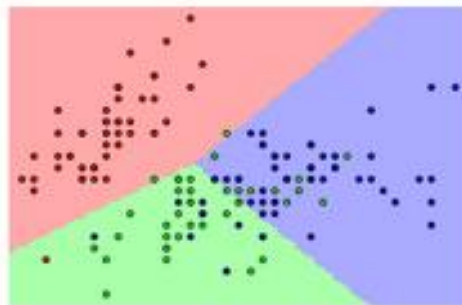
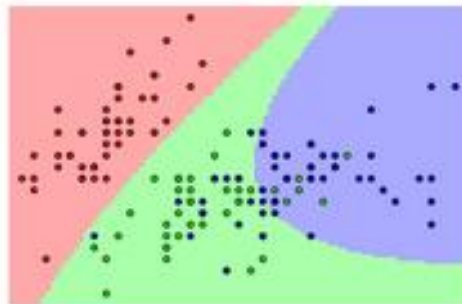
□ + C 越小代表可以忽略限制
→ 寬邊際

□ + C 越大 代表無法忽略邊界
→ 窄邊際



SVM Kernel

- 選擇不同Kernel 時，執行速度不同，分類結果也不同



評估分類結果

進行交叉驗證

■ Holdout 驗證

隨機從最初的樣本中選出部分，形成交叉驗證數據，而剩餘的就當做訓練數據。一般來說，少於原本樣本三分之一的數據被選做驗證數據

■ *K*-fold cross-validation

*K*次交叉驗證，初始採樣分割成*K*個子樣本，一個單獨的子樣本被保留作為驗證模型的數據，其他*K*-1個樣本用來訓練。交叉驗證重複*K*次

■ 留一驗證

正如名稱所建議，留一驗證（**LOOCV**）意指只使用原本樣本中的一項來當做驗證資料，而剩餘的則留下來當做訓練資料

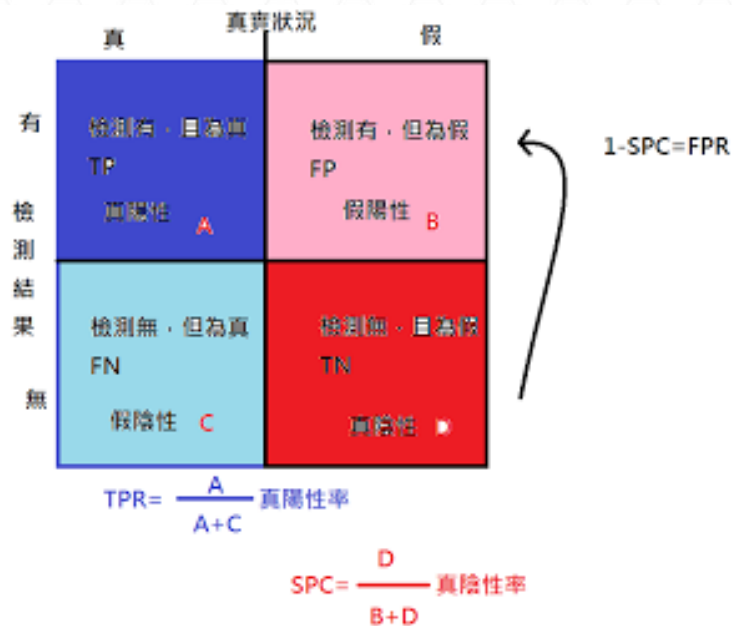
評估結果

- True positive：代表檢測出有，且實際上有的狀況
- False positive：代表檢測出有，而實際上沒有的狀況
- True negative：代表檢測出無，且實際上無的狀況
- False negative：代表檢測出無，而實際上有的狀況

		真實狀況	
		真	假
檢測結果	有	檢測有，且為真 TP 真陽性	檢測有，但為假 FP 假陽性
	無	檢測無，但為真 FN 假陰性	檢測無，且為假 TN 真陰性

評估結果(續)

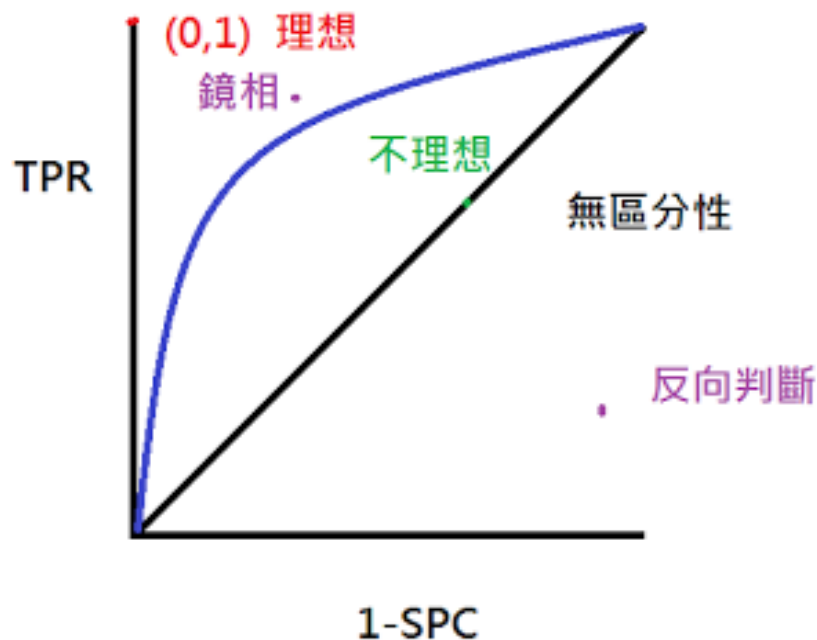
- True positive rate：代表所有陽性樣本中，得以正確檢測出陽性結果的機率，以 $TP/(TP+FN)$ 計算，又稱為靈敏度(sensitivity)。
- True negative rate，代表所有陰性樣本中，得以正確檢測出陰性結果的機率，以 $TN/(FP+TN)$ 計算，又稱為特異性(specificity)。
- False positive rate：代表所有陰性樣本中，檢測出假陽性的機率，以 $FP/(TN+FP)$ 計算，常以 $(1-SPC)$ 的方式呈現。



ROC 曲線

■ 接收者操作特徵(receiver operating characteristic, ROC curve)

- 1.以假陽性率(False Positive Rate, FPR)為X軸，代表在所有陰性相本中，被判斷為陽性(假陽性)的機率，又寫為(1-特異性)。
- 2.以真陽性率(True Positive Rate, TPR)為Y軸，代表在所有陽性樣本中，被判斷為陽性(真陽性)的機率，又稱為敏感性



AUC

曲線下面積(Area Under Curve, AUC)為此篩檢方式性能優劣之指標，AUC越接近1，代表此篩檢方式效能越佳。指標可參考以下條件。

AUC數值	解釋
1	完美分類器，無論cut-off point如何設定都可正確預測。通常不存在
$0.5 < \text{AUC} < 1$	優於隨機，妥善設定可有預測價值
0.5	同隨機，預測訊息沒有價值

新聞分類 (使用NAÏVE BAYES)

新聞分類問題

1. 定義類別 (垃圾郵件、非垃圾郵件)
2. 使用訓練資料的特徵(Feature)與類別建立模型
3. 根據輸入資料的特徵給予對應類別的機率值
4. 給予輸入資料類別

使用Naïve Bayes 分類器

■ Bayes 分類器源自Bayes 理論



Drew Barrymore



Drew Carey

What is the probability of being called
“*drew*” given that you are a **male**?

What is the probability
of being a **male**?

$$p(\text{male} | \text{drew}) = \frac{p(\text{drew} | \text{male}) p(\text{male})}{p(\text{drew})}$$

What is the probability of
being named “*drew*”?

(actually irrelevant, since it is

Naïve Bayes 分類器

- 假設每個特徵(Feature)都為獨立

$$p(d|c_j) = p(d_1|c_j) * p(d_2|c_j) * \dots * p(d_n|c_j)$$

根據Feature d 所產生類別c 的機率

Naïve Bayes 分類器 (續)

■ 假設用Naïve Bayes 預測性別

$$p(\text{officer drew}|c_j) = p(\text{over_170}_{\text{cm}} = \text{yes}|c_j) * p(\text{eye} = \text{blue}|c_j) * \dots$$



Officer Drew
is blue-eyed,
over 170_{cm}
tall, and has
long hair

$$p(\text{officer drew} | \text{Female}) = 2/5 * 3/5 * \dots$$

$$p(\text{officer drew} | \text{Male}) = 2/3 * 2/3 * \dots$$

取得新聞資料

```
import sqlite3
db = sqlite3.connect('news2.sqlite')
cur = db.cursor()
cur.execute('select title, summary, category from news_entry')
allNews = cur.fetchall()

corpus = []
tags = []
for rec in allNews:
    if (rec[2] == '娛樂'.decode('utf-8')) or (rec[2] == '社會'.decode('utf-8')):
        corpus.append(' '.join(jieba.cut(rec[1])))
        tags.append(rec[2])

cur.close()
db.close()
```

建立詞袋

```
import jieba
from sklearn.feature_extraction.text import
CountVectorizer
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(corpus)
word = vectorizer.get_feature_names()
```


將資料分為訓練與測試資料集

```
from sklearn.cross_validation import  
train_test_split  
train_data, test_data, train_tag, test_tag =  
train_test_split(X, tags, test_size=0.50,  
random_state=42)
```

進行文章分類 (Multinomially Distributed)

```
from sklearn.naive_bayes import MultinomialNB
```

```
clf = MultinomialNB(alpha=0.01)
```

```
clf.fit(train_data, train_tag)
```

- 用在文章分類中

- 可根據詞袋

- TF-IDF

- Alpha 是避免沒有若不存在的特徵 (Feature) 導致計算停止

評量預測結果

```
pred = clf.predict(test_data)
```

```
from sklearn.metrics import confusion_matrix  
confusion_matrix(test_tag, pred)
```

```
from sklearn.metrics import accuracy_score  
print accuracy_score(test_tag, pred)
```

The background features a light blue hexagonal grid pattern. Overlaid on this is a large, faint, light blue circular graphic composed of concentric rings and radial segments, resembling a stylized target or a complex orbital path. A solid dark blue horizontal bar runs across the top of the image, and a darker, textured blue horizontal bar runs across the bottom.

THANK YOU