✧ Part 9: Dimensionality Reduction

- 透過降維(Dimensionality Reduction)的方法進行特徵萃取(Feature extraction)
- Principal Component Analysis(PCA)、Linear Discriminant Analysis(LDA)、kernel PCA
- 分別使用 PCA、LDA、kernel PCA 進行特徵萃取後，搭配一個線性模型(這邊是用羅吉斯回歸)進行預測

✧ 為什麼要降維度?

- 資料的特徵(Feature)太多，會使模型過於複雜，影響訓練速度及效能，且很難透過視覺化圖形呈現，所以需要進行降維的處理

✧ Feature Selection 和 Feature Extraction 差異

Feature Selection	Feature Extraction
Backward Elimination	PCA
Forward Selection	LDA
Bidirectional Elimination	Kernel PCA
Score Comparison	

- **Feature Selection**：從特徵集合中挑選一組解釋變量最大的特徵子集，達到降維的效果
- **Feature Extraction**：將現有特徵進行轉換，建構出維度較低的新特徵

✧ 方法介紹

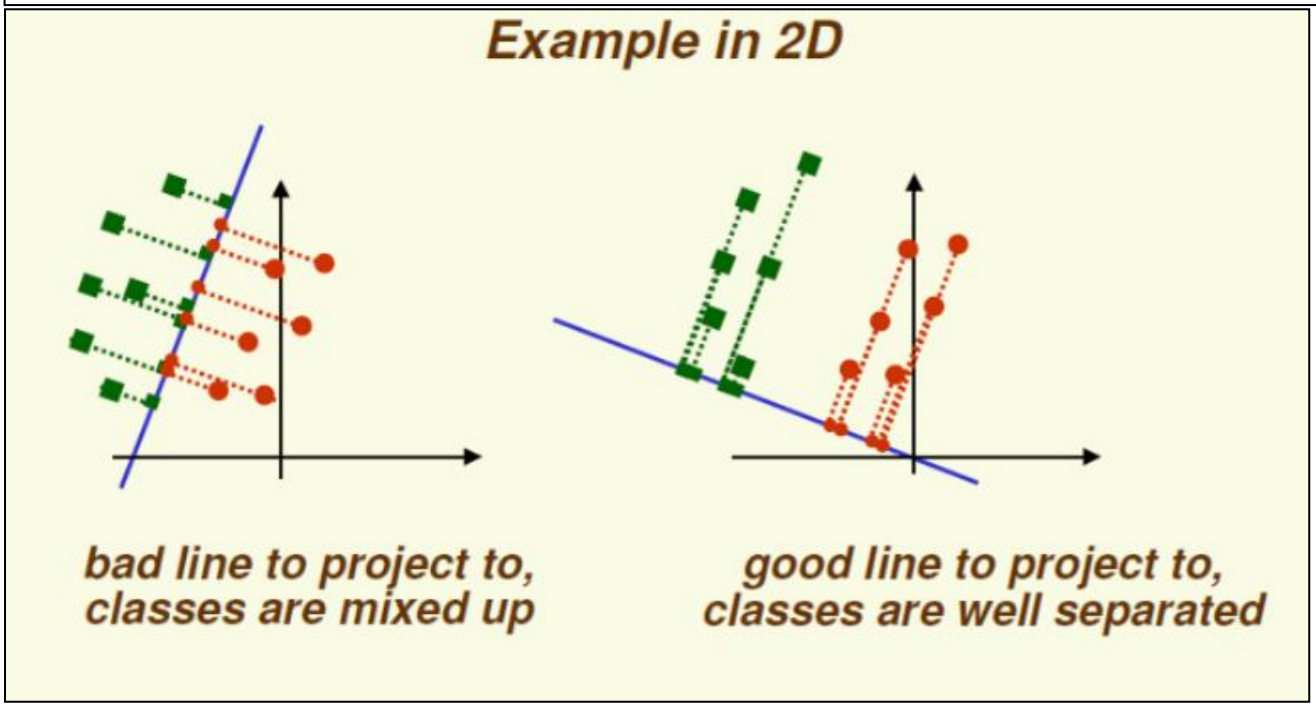
- **Principal Component Analysis(PCA)**
  - 透過降維的方法來進行特徵萃取，適用於處理線性的資料，此方法為 unsupervised，主要是透過某種線性投影，**將高維的數據映射到低維的空間中**，**映射時不考慮數據內的分類資訊**，將數據轉換成各不相關並相互獨立的主成分，藉此能使用較少的數據維度，同時保留住較多的原數據點的特性。

    From the m independent variables of your dataset, PCA extracts p ≤ m new independent variables that explain the most the variance of the dataset, regardless of the dependent variable.

- **Linear Discriminant Analysis(LDA)**

  — 與 PCA 類似，但差別在於此方法為 supervised，在映射的過程中**同時考慮到數據內的分類變數，使映射過後的數據按類別區分開**

  From the n independent variables of your dataset, LDA extracts p ≤ n new independent variables that separate the most the classes of the dependent variable.



✧ 資料集說明

dataset - DataFrame

Index	Proanthocyanins	Color_Intensity	Hue	OD280	Proline	Customer_Segment
0	2.290	5.640	1.040	3.920	1065	1
1	1.280	4.380	1.050	3.400	1050	1
2	2.810	5.680	1.030	3.170	1185	1
3	2.180	7.800	0.860	3.450	1480	1
4	1.820	4.320	1.040	2.930	735	1

情境：使用 Wine 資料預測新品種的酒要推薦給哪個 CUSTOMER_SEGMENT，從現有數據中用降維的方法找出適合的特徵進行預測

✧ 程式碼

```
# Applying PCA
from sklearn.decomposition import PCA
pca = PCA(n_components = None)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
```

```
# Applying LDA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda = LDA(n_components = None)
X_train = lda.fit_transform(X_train, y_train)
X_test = lda.transform(X_test)
```

- 透過 PCA/LDA 方法找出 Components，Components 的組成是由 feature 萃取而成
- 一開始先將參數設定為 None
- 執行 explained_variance_ratio_後，萃取出來的 Components 會依照解釋變異量大小進行排序，由操作者依照不同情況，設定百分比變異量閥值(如下圖)

	0											
0	0.369	3	0.074	7	0.025	11	0.013
1	0.193	4	0.062	8	0.023	12	0.007
2	0.108	5	0.049	9	0.019		
		6	0.041	10	0.017		

✧ 訓練模型

```
# Fitting Logistic Regression to the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
```

- 用羅吉斯跑訓練資料集(沒有限制使用哪個模型來跑，只要是線性模型都可以)

✧ 混淆矩陣分析結果

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

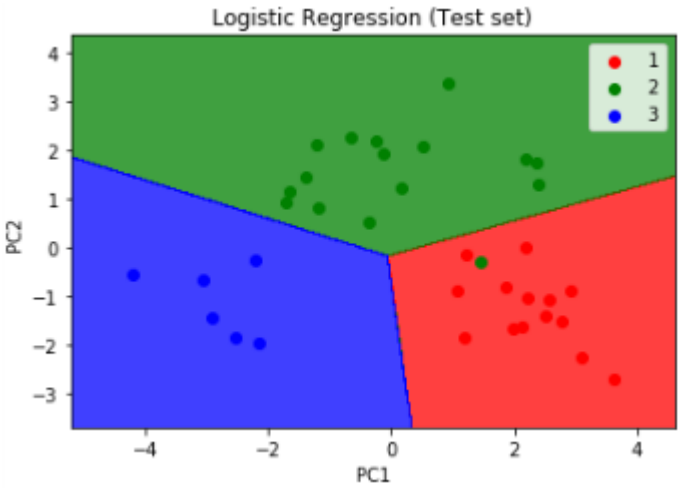
— Confusion Matrix 主要目的為分析預測結果，並觀測筆數分布狀況，x 軸為預測，y 軸為實際發生狀況，而斜對角代表預測正確筆數，其他區塊皆為異常狀態，從下圖中可觀察出 PCA 有 1 筆為異常，而 LDA 全數正確

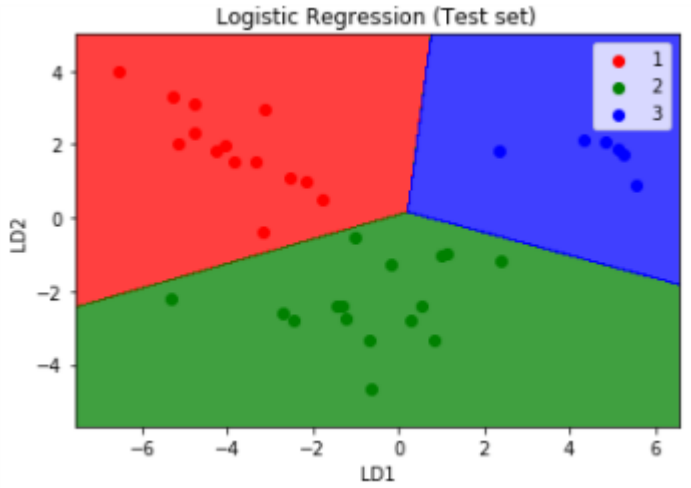
<Confusion Matrix of PCA>

	0	1	2
0	14	0	0
1	1	15	0
2	0	0	6

<Confusion Matrix of LDA>

	0	1	2
0	14	0	0
1	0	16	0
2	0	0	6

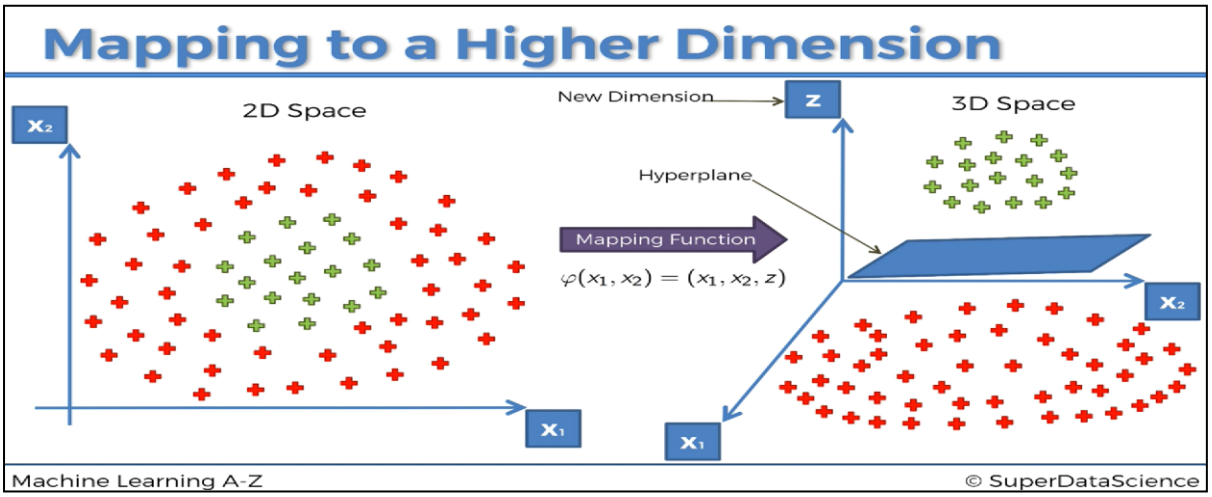




✧ 方法介紹

- **Kernel PCA**

  — 因為資料難以在線性空間進行分類，利用 Kernel 轉換之後在更高的維度上找到合適的分類平面，簡單來說，就是在更高維的空間中做 PCA，在更高維的空間裡，把原始數據向不同的方向投影，再進行分類，**適用於處理非線性的資料**。



✧ 資料集說明

Index	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19.000	19000.000	0
1	15810944	Male	35.000	20000.000	0
2	15668575	Female	26.000	43000.000	0
3	15603246	Female	27.000	57000.000	0

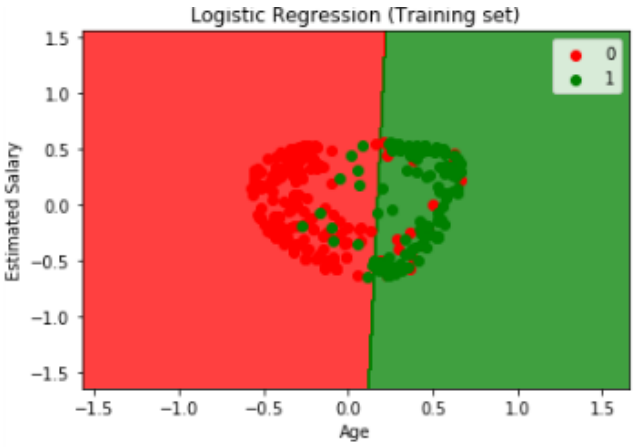
情境：預測使用者是否點擊廣告

✧ 程式碼

```
# Applying Kernel PCA
from sklearn.decomposition import KernelPCA
kpca = KernelPCA(n_components = 2, kernel = 'rbf')
X_train = kpca.fit_transform(X_train)
X_test = kpca.transform(X_test)
```

— 在機器學習中高斯徑向基函數核(rbf)，是一種表現好且常被使用的核函數。

✧ 混淆矩陣分析結果



	0	1
0	64	4
1	6	26

	0	1
0	65	3
1	8	24