13 Machine Learning II - Case Study

Telung Pan
telung@mac.com

監督式機器學習

① 蒐集訓練資料

② 訓練分類器 (Classifier)

③ 執行預測

① 蒐集訓練資料

訓練資料

Examples

Leabel

Features

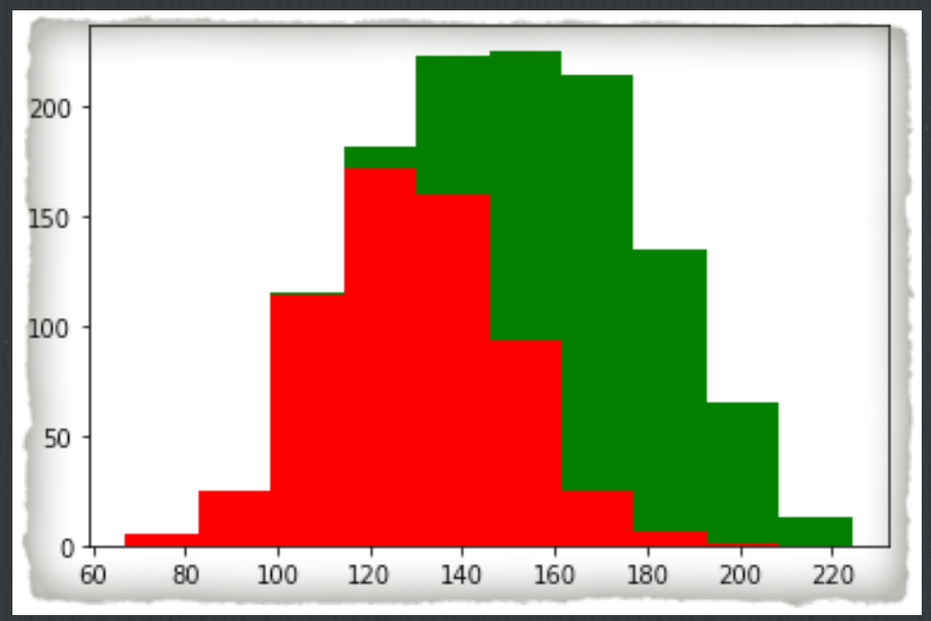
重量	表皮	顏色	答案
150g	粗糙	黃色	柳丁
170g	粗糙	棕色	柳丁
140g	光滑	紅色	蘋果
130g	光滑	紅色	蘋果
...	...	...	...

好的 **Feature**

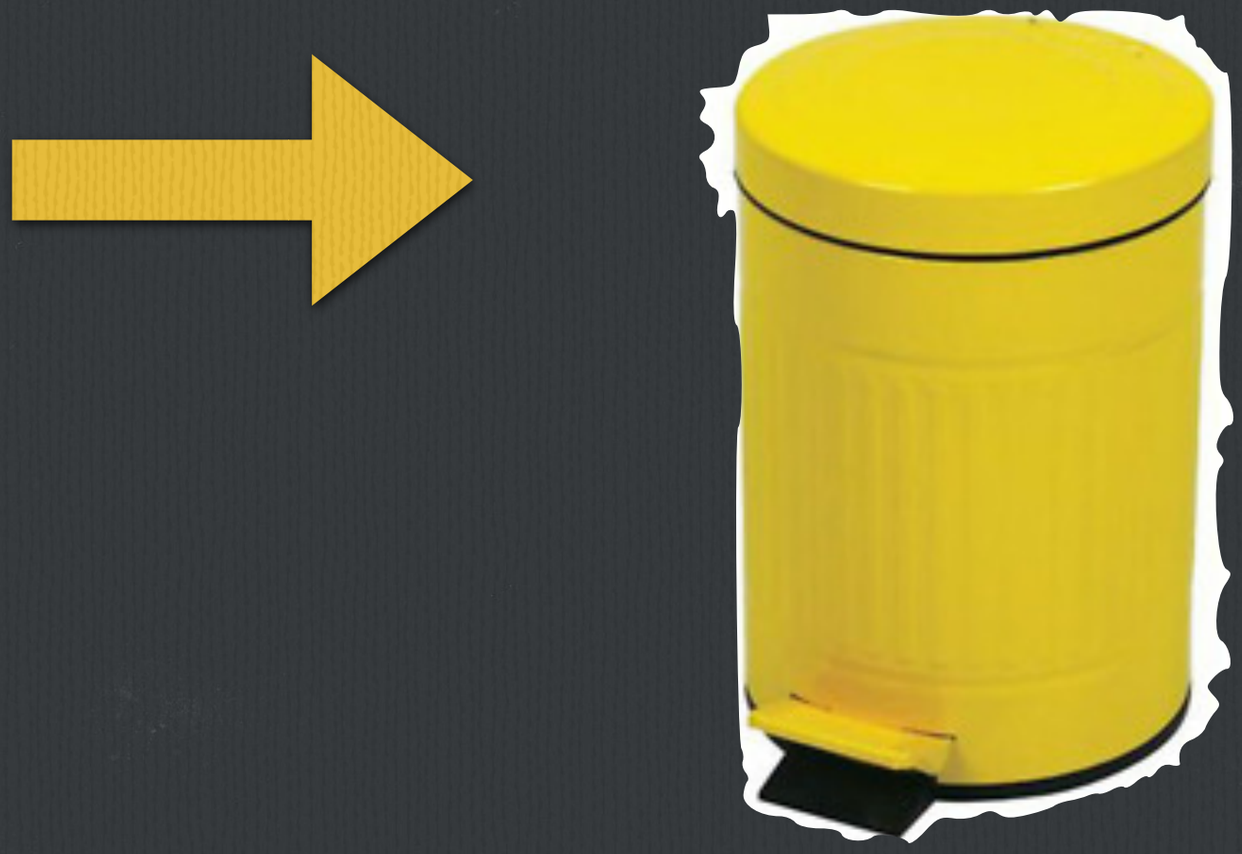
- ☐ **Informative**
- ☐ **Independent**
- ☐ **Simple**

觀察資料找到好的特徵

```python
import numpy as np
import matplotlib.pyplot as plt

apple = 600
orange = 600

apple_weight = 130 + 20 * np.random.randn(apple)
orange_weight = 170 + 20 * np.random.randn(orange)

plt.hist([apple_weight, orange_weight], stacked=True, color=['r','g'])
plt.show()
```



Avoid Useless Features



轉換資料格式

```
import sklearn
features = [[140, "smooth"],[130, 'smooth'],[150,"bumpy"],[170,"bumpy"]]
labes = ["apple", "apple", "orange", "orange"]
```

- 提供給分類器機器學習的資料必須為數字，因此將原始資料轉換成 DICT 格式，然後再使用 DiuctVectorizer 進行 one-hot 類別編碼：

```
dict_features = [
    {'weight':140, 'skin':'smooth'},
    {'weight':130, 'skin':'smooth'},
    {'weight':150, 'skin':'bumpy'},
    {'weight':170, 'skin':'bumpy'},
]

labels = ["apple", "apple", "orange", "orange"]

from sklearn.feature_extraction import DictVectorizer
vec = DictVectorizer()
tranning_features = vec.fit_transform(dict_features).toarray()
```

```
array([[  0.,   1., 140.],
       [  0.,   1., 130.],
       [  1.,   0., 150.],
       [  1.,   0., 170.]])
```

```
[21]: vec.get_feature_names()
```

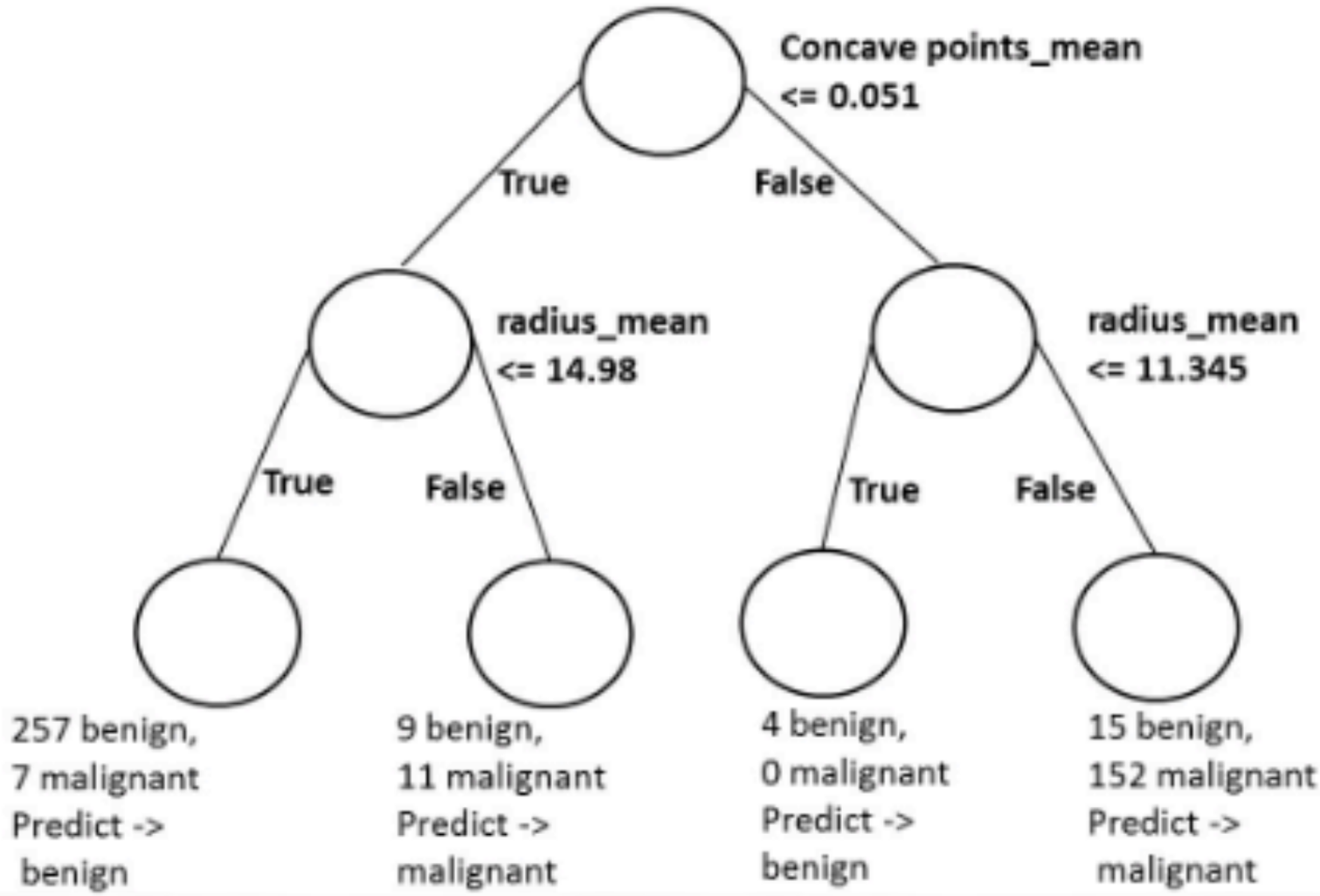
```
[21]: ['skin=bumpy', 'skin=smooth', 'weight']
```

② 訓練分類器機器

決定機器學習所使用的分類器演算法

- ☐ Decision Tree
- ☐ Cluster
- ☐ Classification
- ☐ Regression
- ☐ Bayes

Decision-tree Diagram



用決策樹演算法建立分類器

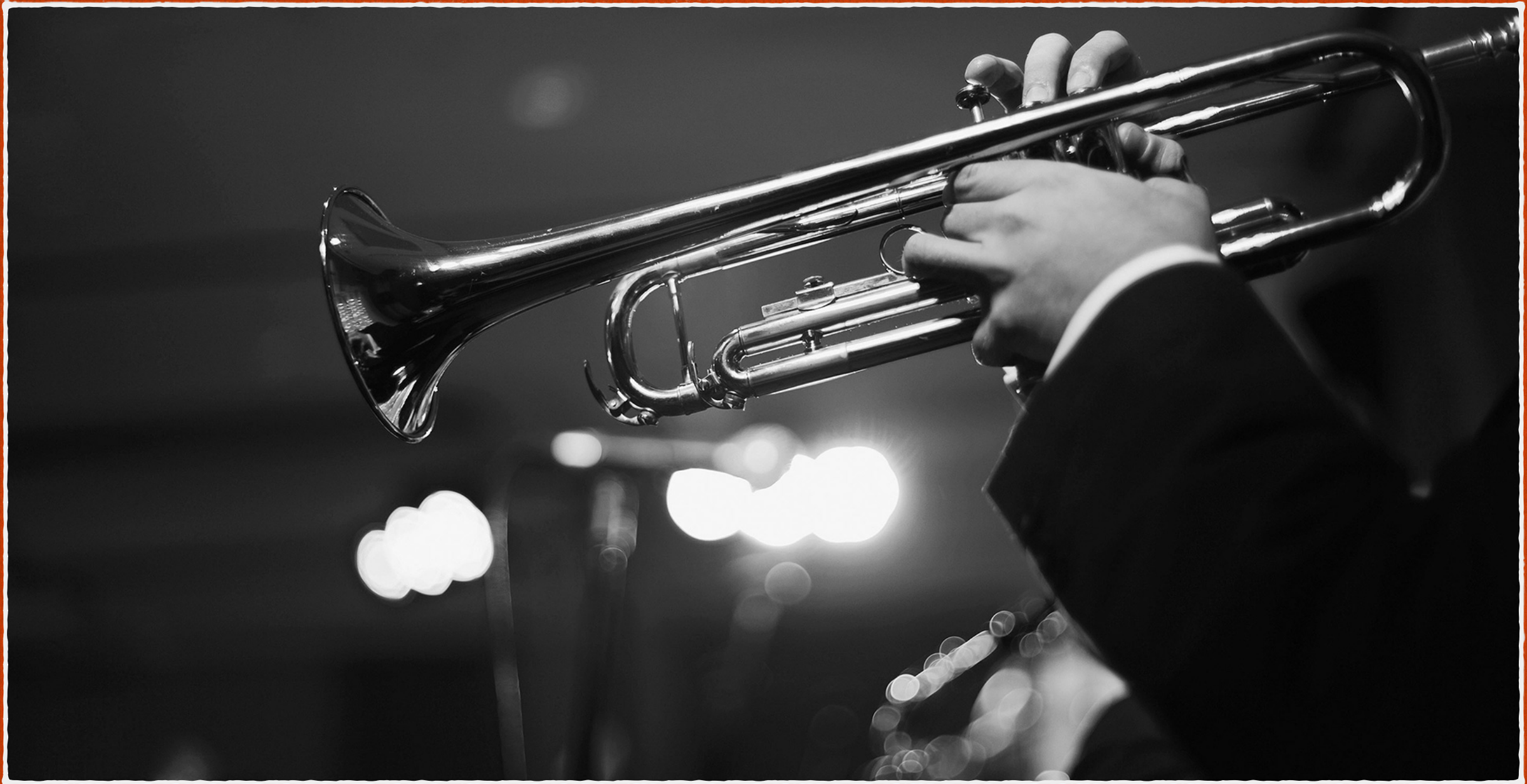
```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(tranning_features, labels)
```

③ 執行預測

執行預測

```
print (clf.predict([[0, 1, 120 ]]))
```

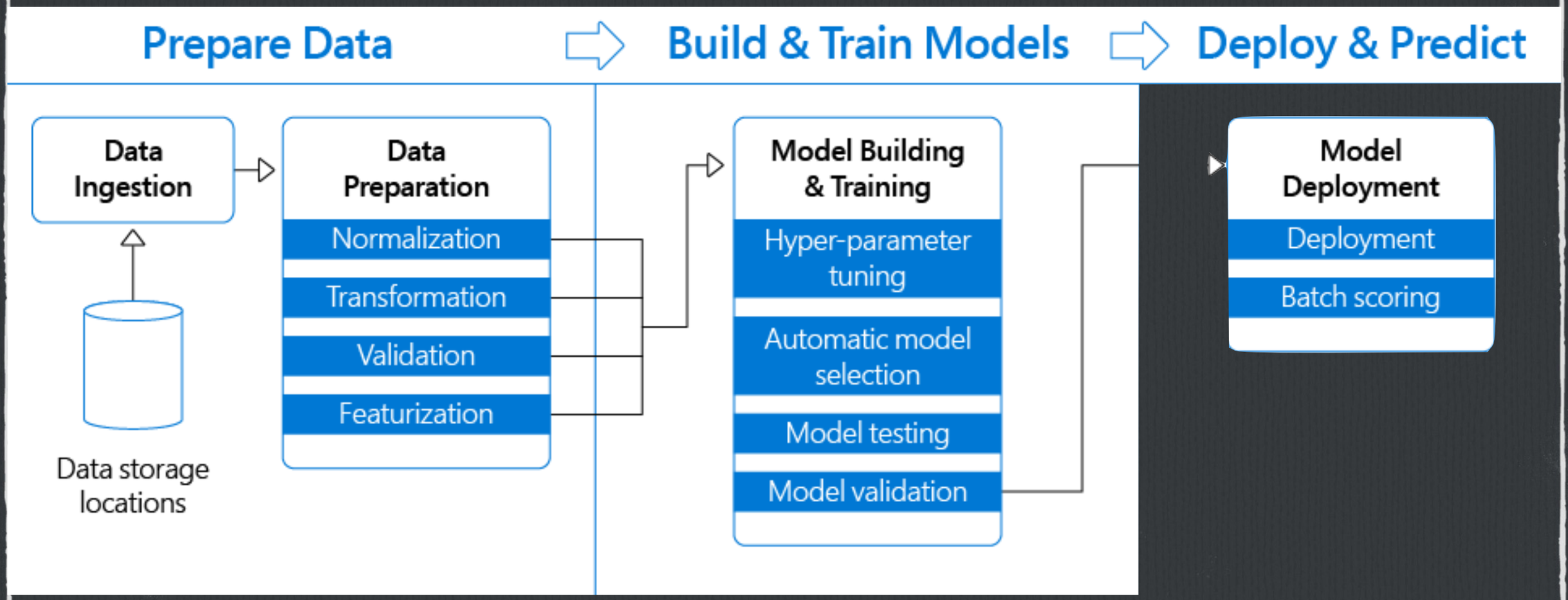
```
['apple']
```



Pipeline

Machine Learning

What is Pipeline?



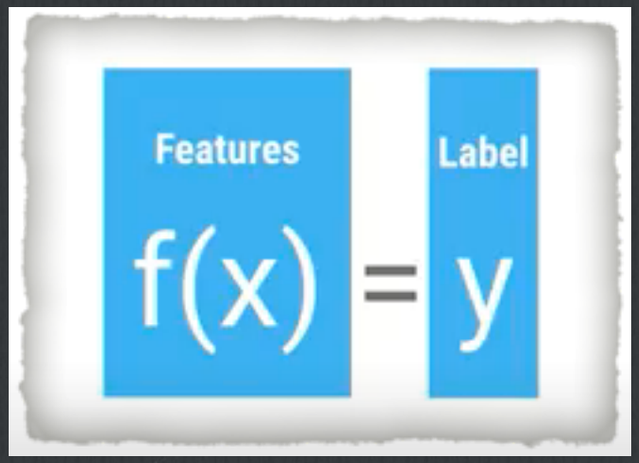
完成一個機器學習架構需要做的事情

[46]:
```python
#匯入 iris 資料集
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data   #Features
y = iris.target #Label
from sklearn.model_selection import train_test_split
#切割訓練與測試資料
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = .5)
from sklearn import tree
my_classifier = tree.DecisionTreeClassifier()
my_classifier.fit(X_train, y_train)
predictions = my_classifier.predict(X_test)
```

print(predictions)

```python
from sklearn.metrics import accuracy_score
print (accuracy_score(y_test, predictions))
```

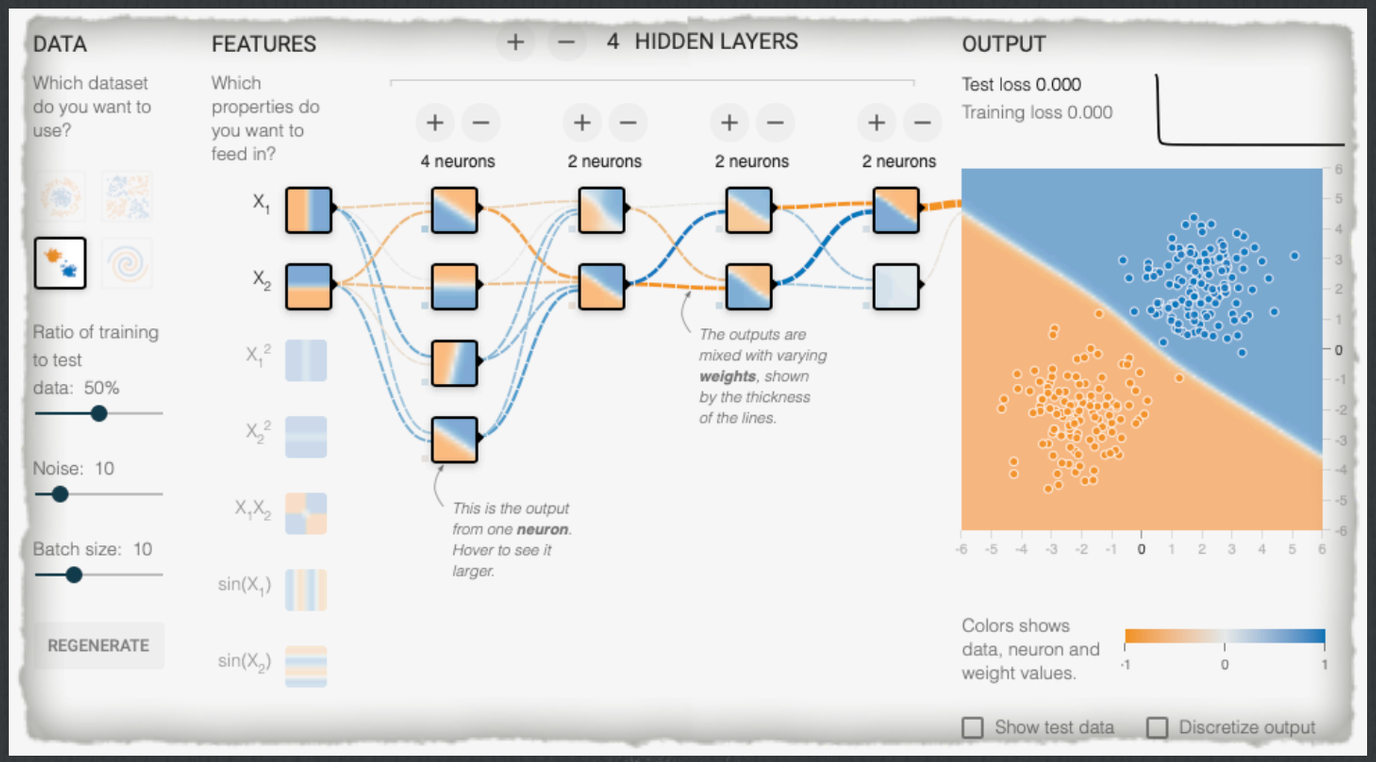
顯示決策樹針對這類資料的預測準確性



測試不同的機器學習演算法

- 修改下列程式碼，改採 Cluster 類別的演算法：

```
from sklearn.neighbors import KNeighborsClassifier
my_classifier = KNeighborsClassifier()
```



使用tensorflow
協助建立機器學習架構

- [ ] https://www.tensorflow.org/overview/?hl=zh_tw

- [ ] 載入 MNIST dataset 然後將資料轉換成浮點數，手寫數位資料：