3. Models

Outline

這一節會教大家建立模型，第一個是用 Logistic Regression 的方法來做分類器，第二個是 Linear Model 來做數值的預測，也會帶大家使用 scikit-learn 這個套件。

Preprocessing

先將資料做前處理，將新聞的內容斷詞計算詞頻。

In [ ]:
```
import pandas as pd
from pathlib import Path
data_folder = Path("../data/")

news = pd.read_csv(data_folder / "news.csv")
news.head()
```

In [ ]:
```
news['length'] = news['content'].apply(len)
```

Segmentation

使用 jieba 來斷詞

In [ ]:
```
import jieba
```

In [ ]:
```
text = news.content[0]
print(text)
```

In [ ]:
```
print(" ".join(jieba.cut(text)))
```

In [ ]:
```
news['segmentation'] = news.content.apply(lambda text: " ".join(jie
ba.cut(text)))
```

tf-idf

tf: term frequency 詞頻，詞語在單一文本中出現的頻率， idf: inverse document frequency 逆向檔案頻率，全部文本的數量除以包含詞語的文本的數量

$$\text{tf-idf} = tf * idf$$

例如「的」可能在文本中詞頻高，但是每個文本都有「的」，因此 idf 很小，tf-idf 相乘起來就很小，代表不是重要的訊息

In [ ]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
v = TfidfVectorizer()
news_tfidf = v.fit_transform(news.segmentation)
```

In [ ]:
```
news_tfidf.shape
```

Binary Logistic Regression Models

使用二元分類的模型來預測資料的類別

In [ ]:
```
selected_news = news.loc[news.provider.isin(['中央社','聯合新聞網']),
['content','provider']]
selected_news.head()
```

In [ ]:
```
selected_news_tfidf = news_tfidf[selected_news.index]
```

In [ ]:
```
import sklearn
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    selected_news_tfidf,
    selected_news[['provider']],
    test_size=0.3,
    random_state=0)
```

In [ ]:
```
X_train
```

In [ ]:
```
X_test
```

In [ ]:
```
y_train
```

In [ ]:
```
y_test
```

In [ ]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train,y_train.provider.values)
```

In [ ]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test.provider.values, lr.predict(X_test))
```

In [ ]:
```
y_test.provider.values
```

In [ ]:
```
lr.predict(X_test)
```

Cross Validation

我們可以使用 Cross Validation 來評估 Classifier 的效果，常用的方法是 k-fold ，也就是將資料分成 k 等份，每次使用其 k-1 份來 training，剩下一份來 testing，總共執行 k 次，這樣做可以充分利用手上已經有的資料來學習。

In [ ]:
```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(lr, selected_news_tfidf, selected_news.pro
vider.values, cv=5)
print(scores)
```

In [ ]:
```
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
* 2))
```

Linear Models

使用線性的模型來模擬預測未知數值

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(
    news_tfidf,
    news[['length']],
    test_size=0.3,
    random_state=7)
```

In [ ]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

regr = LinearRegression()
regr.fit(X_train, y_train)
y_pred = regr.predict(X_test)
```

In [ ]:
```
print('Coefficients: \n', regr.coef_)

print("Mean squared error: %.2f"
      % mean_squared_error(y_test, y_pred))

# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(y_test, y_pred))
```

Exercises and Solutions

▶ 1. 改用 F1 score 來評定 Classifer 的成效
▶ 2. 使用 Multinomial Naive Bayes 來做一個新的 Classifier

More about:

1. An introduction to machine learning with scikit-learn (http://scikit-learn.org/stable/tutorial/basic/tutorial.html)
2. Working With Text Data (http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html)
3. Scikit Learn User Guide (http://scikit-learn.org/stable/user_guide.html)