# Big Data Analytics Techniques and Applications

# Homework 4

309552063 吳冠潔

使用平台：Google Colab

Target: Build a predictive framework for predicting whether each flight in 2005 will delay or not by using the data from 2003 to 2004 as training data.

Q1: Show the predictive framework you designed.
Hint: What features do you extract? What algorithms do you use in the framework?
Features: Month, DayofMonth, CRSDepTime, CRSArrTime, Distance
Model: LogisticRegressionWithLBFGS

Method:
一開始先將 train, test 資料分別讀出來

```
traindata = prepData(trainpath)
print(traindata.first())
testdata = prepData(testpath)
print(testdata.first())
```

```
['2003', '6', '29', '7', '1756', '1725', '1904', '1838', 'UA', '781', 'N228UA', '128', '133', '103', '26', '31', 'DEN', 'LAX',
['2005', '1', '28', '5', '1603', '1605', '1741', '1759', 'UA', '541', 'N935UA', '158', '174', '131', '-18', '-2', 'BOS', 'ORD',
```

```
'862', '8', '17', '0', '', '0', '0', '26', '0', '0', '0']
, '867', '4', '23', '0', '', '0', '0', '0', '0', '0', '0']
```

其中的 prepData，是先將 csv 檔存成 RDD，接著先刪除資料中的 header，接著將資料中 Cancelled 的班機和 Diverted 的班機從資料中刪除，接著從選出的 features 中(selcol)，刪除有 NA 資料的班機。在 selcol 中還有選 WeatherDelay 這一欄，預計將該欄作為要預測有無 delay 發生，在有無發生 delay 的部分，我只取 WeatherDelay 的部分，是因為 WeatherDelay 較能有規律地發生，像是在台灣，可能會在颱風好發時期，有較多的班機延遲，所以決定使用 WeatherDelay。另外其他在 selcol 中除了 WeatherDelay，其他分別為 Month, DayofMonth, CRSDepTime, CRSArrTime, Distance，用來作為訓練 model 的 feature。

```
selcol = [1, 3, 5, 7, 18, 25]

def handelNa(line):
    for col in selcol:
        if line[col] == 'NA':
```

```
            return False
    return True

def prepData(path):
    rawdata = sc.textFile(path)
    header = rawdata.first()

    data = rawdata.filter(lambda line: line != header) \
                .map(lambda line: line.split(',')) \
                .filter(lambda line: line[21] == '0') \
                .filter(lambda line: line[23] == '0') \
                .filter(lambda line: handelNa(line))
    # print(data.first())

    return data
```

接著將資料取出 feature，並用 WeatherDelay 做 label，當發生 WeatherDelay 時，就 label 為 1，反之為 0，分別得到 label 好的 traindata, testdata

```
from pyspark.mllib.regression import LabeledPoint

def labelData(line):
    features = []
    for col in selcol[:-1]:
        features.append(float(line[col]))

    lbl = 0.0
    if(float(line[25]) > 0.0):
        lbl = 1.0
    return LabeledPoint(lbl, features)

trainlabeldata = traindata.map(lambda line: labelData(line))
print(trainlabeldata.first())
testlabeldata = testdata.map(lambda line: labelData(line))
print(testlabeldata.first())
```

```
(1.0,[6.0,7.0,1725.0,1838.0,862.0])
(0.0,[1.0,5.0,1605.0,1759.0,867.0])
```

接下來用 label 好的 training data 訓練 model，在這邊我用 spark MLlib 中的

LogisticRefressionWithBFGS 來訓練 model。

```
from pyspark.mllib.classification import
LogisticRegressionWithLBFGS

model = LogisticRegressionWithLBFGS.train(trainlabeldata)
```

Q2: Explain the validation method you use.

Hint: Leave-one-out, Holdout, k-fold, or other methods?

我用 Holdout validation 來做 validation

Method:

先將 training data 用 tandomSplit，分成 7 比 3，然後用同樣的 Model 對分出來的 7 成的 training data 進行訓練。

```
valtraindata, validationdata = trainlabeldata.randomSplit([0.7,
0.3])
val_model = LogisticRegressionWithLBFGS.train(valtraindata)
```

Q3: Explain the evaluation metric you use.

Hint: Don't just show the prediction results, you should show the effectiveness of your framework (e.g., using a confusion matrix).

在 evaluation metric 中，使用 spark MLlib 中的 BinaryClassificationMetrics。並看 ROC curve 和 PR curve。

Method:

得到 testing data 的預測值，用 spark MLlib 中的 BinaryClassificationMetrics 得到期 evaluation metric，並印出期 PR 線下面積，和 ROC 線下面積。

```
from pyspark.mllib.evaluation import BinaryClassificationMetrics

predictionAndLabels = testlabeldata.map(lambda lp:
(float(model.predict(lp.features)), lp.label))
metrics = BinaryClassificationMetrics(predictionAndLabels)

print("Area under PR = %s" % metrics.areaUnderPR)
print("Area under ROC = %s" % metrics.areaUnderROC)
```

```
Area under PR = 0.015928868937046734
Area under ROC = 0.5
```

```
val_predictionAndLabels = validationdata.map(lambda lp:
(float(val_model.predict(lp.features)), lp.label))
val_metrics = BinaryClassificationMetrics(val_predictionAndLabels)

print("Area under PR = %s" % val_metrics.areaUnderPR)
print("Area under ROC = %s" % val_metrics.areaUnderROC)
```

```
Area under PR = 0.014896737991571398
Area under ROC = 0.5
```

Q4: Show the validation results and give a summary of results.

用訓練出的 model 去預測 testing data，得到的 area under PR 和 area under ROC 分別為 0.0159 和 0.5；在用 validation 得到的則是分別為 0.0149 和 0.5。

Test:

```
Area under PR = 0.015928868937046734
Area under ROC = 0.5
```

Validation:

```
Area under PR = 0.014896737991571398
Area under ROC = 0.5
```

雖然模型表現得沒有很好，但是比起我一開始直接將 DepDelay 大於 15 分鐘的班機都視為有發生延遲時訓練出來的 model 要好，不過單看 AUC 的表現只有在 0.5，如果能將其提升到 0.7 以上才可以算是一個好的 model，因為我只取有無發生 WeatherDelay，如果能將當天的天氣資訊也加到 feature 中，一定能夠大大提升 model 的效能。

Difficulties Encountered:

在這次的作業中，我是以 Weatherdelay 來判斷有無發生 delay，本來是嘗試用看 DepDelay 這欄來做判斷，但是做完之後判斷的結果很差，所以我後來決定只使用因為 Weatherdelay 而延遲的班機來判斷 2005 年的班機是否會因為 Weatherdelay 而延遲，效果就比一開始只看 DepDelay 好。

另外，在使用 google colab 時，常常會發生 py4jjavaerror，試過改 java 版本，但是都無法成功解決該問題，因為解決不了該 error，所以只能把整個訓練 model 的方式改掉，在這部分花了很多時間。

Source Code : bigdata_hw4.ipynb