國立雲林科技大學

資訊管理系所

資料探勘專案作業報告

學生/學號：

廖帝登 D10823001

張　靖 M10823001

劉博茜 M10823015

盧廼安 M10823023

【多那之鮮奶茶】

指導教授：許中川

中華民國 109 年 1 月

摘要

本研究針對 AIdea 網站，進行載客熱點的預測，隨著大量資料的快速累積以及演算法與雲端運算的發展，大數據分析已經成為學術界與各行業關注與學習的焦點。從巨量資料中篩選有用的資訊，其運用的方法為資料探勘。本研究將使用 Python 撰寫隨機森林之程式進行模型訓練；然而本議題所提供之資料為內湖區的計程車載客時間、地點 GPS 資料。該資料涵蓋時間為 2016-02-01 08:00:00~2017-01-31 23:59:59，共計 4,118,812 筆。載客熱點預測方式為：預測內湖區於一段時間（一個月）中，各時段的乘車需求。本研究將 Hire_count 欄位值：內湖區該時段的乘車需求預測數量，該欄位下載時為空值，透過填入該欄位預測值（正整數值，0~n），計算 RMSE，了解訓練模型和預測值的差異。

關鍵字：資料探勘、Python、隨機森林

目錄

第1章 緒論

1.1 研究背景與動機

隨著資訊科技發展下，資料的處理與分析變得更加重要，資料探勘則是現今挖掘資訊最重要的一項技術。Random Forest 的基本原理是，結合多顆 CART 樹（CART 樹為使用 GINI 算法的決策樹），訓練資料則是隨機分配的，以大幅增進最終的運算結果。不過，這個方法必須基於下面的理論：Ensemble Method。Ensemble Method（集成方法）的想法是，如果一個分類器表現可以，那麼將多個分類器組合起來的話，其表現則會優於單個分類器。不過要滿足這理論是有條件的：各個分類器之間須具有差異性。每個分類器的準確度必須大於 0.5。也就是說，可不是一群一樣笨的臭皮匠就可以勝過一個諸葛亮，這些臭皮匠要符合如下的條件：1. 他們不能太笨（分類器的準確度必須大於 0.5），2. 他們彼此之間的經驗有所不同（分類器之間具有差異性）。必須符合這兩個條件，才能讓結合多顆 CART 樹的隨機森林其效力大於單一的決策樹，這種方式稱為 Ensemble Method（集成方法）。

本研究將資料為內湖區的計程車載客時間、地點 GPS 資料。該資料涵蓋時間為 2016-02-01 08:00:00~2017-01-31 23:59:59，共計 4,118,812 筆。作為資料集輸入。載客熱點預測系統，是經由分析計程車乘客的相關歷史乘車時間與地點資料，藉此預測未來在特定的時間、特定的地點，乘客的乘車需求。讓需要的業者運用合宜的預測模型可有效媒合司機與乘客、增加計程車的載客率、並且降低乘客等待的乘車時間，也可以減少道路上計程車空車問題、並降低空汙廢氣排放等議題，最終達到提升交通的服務品質，做為發展與經營智慧城市之重要參考依據。

1.2 研究目的

本研究將使用 Python 撰寫隨機森林程式碼，透過 RMSE 的值，了解本研究的模型績效程度。

第2章 方法

程式架構- 隨機森林

```
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from datetime import datetime

train_hire_stats =
pd.read_csv("D:/Python/DM_report4/taxi_data/train_hire_stats.csv")
test_hire_stats = pd.read_csv("D:/Python/DM_report4/taxi_data/test_hire_stats.csv")
data_feature_name = train_hire_stats.columns[0:-1]

# train_hire_stats 日期轉星期
date =    train_hire_stats['Date']
for i in range(len(date)+1):
      ch = date[i]
      ch = datetime.strptime(ch, "%Y-%m-%d").weekday()
      date[i] = ch
train_hire_stats['Date'] = date

# test_hire_stats 日期轉星期
date =    test_hire_stats['Date']
for i in range(len(date)+1):
      ch = date[i]
      ch = datetime.strptime(ch, "%Y-%m-%d").weekday()
      date[i] = ch
test_hire_stats['Date'] = date



test_hire_stats = test_hire_stats.drop("Hire_count",axis=1)
test_hire_stats = test_hire_stats.drop('Test_ID',axis=1)

train_features = train_hire_stats.drop('Hire_count', axis=1)
train_labels = train_hire_stats['Hire_count']
# 使用隨機森林演算法(1000 顆樹)
rf = RandomForestRegressor(n_estimators = 1000,random_state = 42)
rf.fit(train_features, train_labels)
```

```
# 進行預測
# Use the forest's predict method on the test data
predictions = rf.predict(test_hire_stats)
name = ['predictions']
new_predictions = pd.DataFrame(columns=name)
new_predictions['predictions'] = predictions
new_predictions.to_csv('D:\Python\DM_report4\
new_predictions.csv',encoding='gbk')
```

第3章 實驗

3.1 資料集

train_gps_points.csv：乘客上車 GPS 點位記錄，此資料乃經由篩選計程車錶於內湖區範圍內的啟動記錄而得，並且將計程車錶啟動時間與地點視為乘客上車點，共 4,118,812 筆資料。

Datetime：計程車錶啟動時間（台北時間 GMT+8:00）

Longitude_X：GPS 記錄經度 (WGS84 座標)

Latitude_Y：GPS 記錄緯度（WGS84 座標）

test_hire_stats.csv：乘車需求預測。

Test_ID：乘車需求之流水序號，共 672 筆

Date：預測日期，格式 YYYY-mm-dd，如 2017-02-01

Hour_slot：預測時段，以一小時計，本欄位的值為 0~23。

0 代表 0:00:00~0:59:59 時段，1 代表 1:00:00~1:59:59，依此類推，23 代表 23:00:00~23:59:59

3.2 實驗設計

利用 python 進行實驗，透過原始數據(training data)進行隨機森林的訓練，藉此建立該數據的模型，透過測試數據(testing data)進行結果的預測，透過 RMSE 的來了解該演算法績效的程度。

3.3 實驗結果

8	我現在要出征我現在要出征哈囉哈囉哈囉	10.931094	2019/12/29 13:38:46	36
9	多那之鮮奶茶	10.939775	2019/12/27 23:30:59	4

第4章 結論

本研究的目的是利用 python 建構隨機森林的預測模型，並計算 RMSE 值來了解預測模型的誤差。RMSE 則是一種常用的測量數值之間差異的量度，其數值常為模型預測的值或是被觀察到的估計值。RMSE 代表預測的值和觀察到的值之差的樣本標準差(sample standard deviation)，當這些差值是以資料樣本來估計時，他們通常被稱為殘差；當這些差值不以樣本來計算時，通常被稱為預測誤差（prediction errors)。RMSE 是一個好的準度的量度。本研究最終透過系統，將會參照剩餘測試資料（60%）的真實數值（Ground Truth）來驗證與計算分數，最終得分為 10.9339。

附錄 1

```
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from datetime import datetime

train_hire_stats =
pd.read_csv("D:/Python/DM_report4/taxi_data/train_hire_stats.csv")
test_hire_stats = pd.read_csv("D:/Python/DM_report4/taxi_data/test_hire_stats.csv")
data_feature_name = train_hire_stats.columns[0:-1]

# train_hire_stats 日期轉星期
date =    train_hire_stats['Date']
for i in range(len(date)+1):
      ch = date[i]
      ch = datetime.strptime(ch, "%Y-%m-%d").weekday()
      date[i] = ch
train_hire_stats['Date'] = date

# test_hire_stats 日期轉星期
date =    test_hire_stats['Date']
for i in range(len(date)+1):
      ch = date[i]
      ch = datetime.strptime(ch, "%Y-%m-%d").weekday()
      date[i] = ch
test_hire_stats['Date'] = date



test_hire_stats = test_hire_stats.drop("Hire_count",axis=1)
test_hire_stats = test_hire_stats.drop('Test_ID',axis=1)

train_features = train_hire_stats.drop('Hire_count', axis=1)
train_labels = train_hire_stats['Hire_count']
# 使用隨機森林演算法(1000 顆樹)
rf = RandomForestRegressor(n_estimators = 1000,random_state = 42)
rf.fit(train_features, train_labels)
# 進行預測
```

```
# Use the forest's predict method on the test data
predictions = rf.predict(test_hire_stats)
name = ['predictions']
new_predictions = pd.DataFrame(columns=name)
new_predictions['predictions'] = predictions
new_predictions.to_csv('D:\Python\DM_report4\
new_predictions.csv',encoding='gbk')
```

附錄 2

課程心得與建議:

從期初資料探勘課程開始前，一直都不瞭解資料探勘內容是在講些什麼，在老師的帶領之下，從線性代數、機率這兩門基礎開始著手，才慢慢的知道，原來數據的呈現和運算，幾乎都脫離不這兩門課程，在認識這兩門課程之後，也漸漸熟知資料探勘的內容，從預測、解釋、降維度、各種(監督式、非監督式)演算法，這些內容有作業的練習，也有老師的講解，才慢慢清楚資料探勘的意涵，非常榮幸能夠修這門課程，雖然有很多的數學式，但想必上了一學期以後，功力定會大增。課程建議的部分，則是希望能夠有大家互相討論，互相分享的時間。

透過一學期課程初步了解到處理資料的過程,從資料的問題定義、資料獲取、資料清洗。缺失值處理、特徵選擇、資料集劃分，對於每一項步驟的初步了解以及資料集結果的效度評估都是一門學問，透過四次的專案作業，使用 python 實做演算法運用在處理資料上，並進行程式的編碼，實際的操作也清楚了解到演算法的細節，在未來朝向資料分析領域研究有很大的幫助。

這門課程有從理論到技術，內容則是由淺入深，先由最基礎的概念開始，之後慢慢的教了許多演算法公式以及所代表的意涵，那也慢慢的內容難度隨之增加，再加上一次三堂課下來，吸收的程度有限，但是最重要的還是透過課後複習來補足上課不懂的地方，那也有課後測驗以及作業的部分，更可加深所學知識。

一開始對於資料探勘只是初步的了解，之前也有上過類似的課，但是只有教一些基本的演算法，在這學期的課堂中學到很多有關資料分析的演算法，雖然剛開始一下子就接觸到那麼多的數學很難一下就完全消化,需要花時間去練習慢慢地去理解。透過每一次的作業和考試加深了對資料探勘的認知，每一堂課都教得非常扎實，老師上課也會細心的講解，透過這門課讓我受益良多。