中原大學人工智慧應用學士學位學程

深度學習專題實驗報告

消費預測(航空公司每月客流量)

10612201 葉羽修

授課老師: 朱守禮 教授

目錄

一、選題動機及目的

之前因為我們都在影像(cnn 做辛普森圖像辨識)、語言(chatbot)等基礎應用上打轉。

這次我們選擇要探討一個可應用在企業運作上的實例。

因此我們選擇做銷售預測，希望由過去的銷售記錄預測下一個週期的銷售量。

二、資料介紹：

1. 測試資料集: 美國航空公司每月乘客人數


airline-passengers - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
"Month","Passengers"
"1949-01",112
"1949-02",118
"1949-03",132
"1949-04",129
"1949-05",121
"1949-06",135
"1949-07",148
"1949-08",148
"1949-09",136
"1949-10",119
"1949-11",104
"1949-12",118
"1950-01",115
"1950-02",126
"1950-03",141
"1950-04",135
"1950-05",125
"1950-06",149
"1950-07",170
"1950-08",170
"1950-09",158
```

資料訊息:日期 、乘客數— 格式為：yy-mm

2. 訓練資料集: 桃園國際機場每月出入境客運量

(1979/1~2020/5)

含新舊統計方式

492 個月，共 492 筆資料

三:模型介紹

原本我們打算先使用簡單迴歸(Regression) 公式 y=ax+b，去做基本分析，但在銷售量的表現上，這個假設並不合理的，因為基於 y(i) 與 y(j)(丟入新參數)是相互獨立，沒有任何關聯

因為專題在做聊天機器人時，有遇到自然語言處理的問題所以有涉略 LSTM 模型在『自然語言處理』時，我們會使用 LSTM 考慮上下文的關係，這個模型恰好與前面講的消費量預測模式，似乎不謀而合，所以我們打算以 LSTM 這個模型來試試看預測消費趨勢。

淺談時間序列(Time Series Analysis)公式

簡單迴歸(Regression) 公式 y=ax+b，是基於 y(i) 與 y(j) 是相互獨立，沒有任何關聯，但在銷售量的表現上，這個假設並不合理，公司銷售業績通常不會暴漲暴跌，而是『逐步』上升或下跌，也就是與前期的表現有緊密的關聯。
另外，大部分的公司也會有淡、旺季，即所謂的『季節效應』(Seasonal Effect)，因此，使用更複雜的『時間序列分析』(Time Series Analysis)預測會更貼近事實。
時間序列分析的模型因應問題的型態不同也有很多種

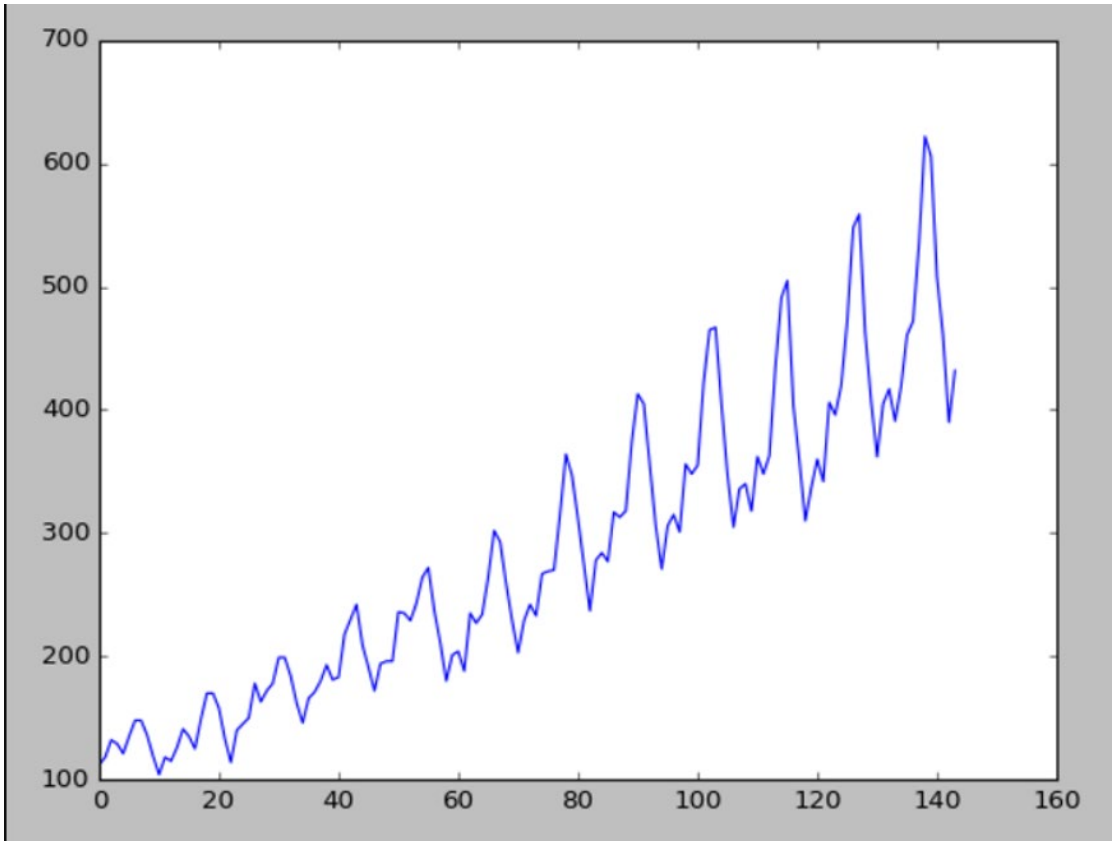
我們以 ARIMA(Autoregressive Integrated Moving Average)作為我們的計算方法
ARIMA 是 ARMA 的擴充模型，而 ARMA 就等於 AR + MA
AR 談的就是前期與後期的關係，有一重要的參數 p(p 代表期數)

四、程式碼:

1. 原始資料的繪圖

```
import pandas
import matplotlib.pyplot as plt
dataset = pandas.read_csv('international-airline-passengers.csv', usecols=[1],
engine='python', skipfooter=3)
plt.plot(dataset)
plt.show()
```

每月乘客人數折線圖



我們採用網路上的建議 進行 ACF PACF 的檢查

```
import numpy as np
from scipy import stats
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.graphics.api import qqplot
# 畫出 ACF 12 期的效應
sm.graphics.tsa.plot_acf(dataset, lags=12)
plt.show()
# 畫出 PACF 12 期的效應
sm.graphics.tsa.plot_pacf(dataset, lags=12)
plt.show()
```

2. 載入 LSTM

```
# LSTM 的載入
import numpy
import matplotlib.pyplot as plt
from pandas import read_csv
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
```

3. 初次產生資料集

```
# 先產生 (X, Y) 資料集,(Y 是下一期的乘客數)
def create_dataset(dataset, look_back=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i + look_back, 0])
    return numpy.array(dataX), numpy.array(dataY)
```

4. 載入資料並正規化

```
# 載入訓練資料
dataframe = read_csv('international-airline-passengers.csv', usecols=[1],
engine='python', skipfooter=3)
dataset = dataframe.values
dataset = dataset.astype('float32')
# 正規化(normalize) 資料，使資料值介於[0, 1]-->當初忘記正規劃，資料顯示變得很奇怪
scaler = MinMaxScaler(feature_range=(0, 1))
dataset = scaler.fit_transform(dataset)
```

5. 區分資料集

```
# 2/3 資料為訓練資料， 1/3 資料為測試資料--> 我們原本想載入的測試資料無法順利正規畫
# 所以我們先拿原本的資料集做實測
train_size = int(len(dataset) * 0.67)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]
```

6. 重新產生資料集

```
# 產生 (X, Y) 資料集, Y 為下一期的乘客數(X=t and Y=t+1)
look_back = 1
trainX, trainY = create_dataset(train, look_back)
testX, testY = create_dataset(test, look_back)
# reshape input to be [samples, time steps, features]
trainX = numpy.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
testX = numpy.reshape(testX, (testX.shape[0], 1, testX.shape[1]))
```

7. LSTM 的建置

```
# 開始建立及訓練 LSTM 模型
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(trainX, trainY, epochs=100, batch_size=1, verbose=2)
```

8. 調整數據集大小

```
# 將預測資料值為原始數據的規模大小
trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])
```

9. 計算誤差值

```
# 計算 均方根誤差
trainScore = math.sqrt(mean_squared_error(trainY[0], trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(testY[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))
```

10. 畫出比較圖表

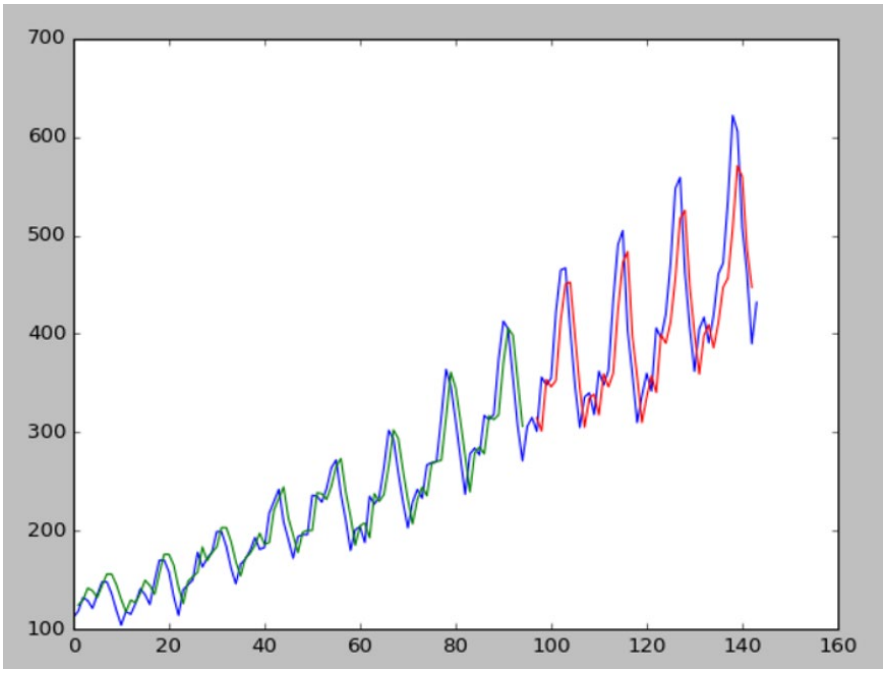
```
# 畫訓練資料趨勢圖

trainPredictPlot = numpy.empty_like(dataset)
trainPredictPlot[:, :] = numpy.nan
trainPredictPlot[look_back:len(trainPredict)+look_back, :] = trainPredict

# 畫測試資料趨勢圖

testPredictPlot = numpy.empty_like(dataset)
testPredictPlot[:, :] = numpy.nan
testPredictPlot[len(trainPredict)+(look_back*2)+1:len(dataset)-1, :] = testPredict

# 畫原始資料趨勢圖
# plot baseline and predictions
plt.plot(scaler.inverse_transform(dataset))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```

程式執行:

將下載的資料檔 international-airline-passengers.csv 與程式 SimpleLSTM.py 放在同一目錄，在 DOS 內執行以下指令：

python SimpleLSTM.py

結果:



綠色:訓練

紅色:測試

藍色:原資料集

主要步驟:

1.將日期及乘客數，正規化(normalize) 資料，使資料值介於[0, 1]。

2.我們要用前期預測當期，故將資料轉為

（前期乘客數 當期乘客數)，當作(X, Y)。

3.建立及訓練 LSTM 模型。

4.訓練 LSTM 模型並進行預測。

5.針對實際值、預測值進行繪圖。

五、遇到的問題

1. 載入套件的問題(StatsModels 套件):

之前一直無法順利下載並使用 StatsModels 套件進行繪圖、分析，後來才發現 pip install 會有錯 。

必須使用

conda install -c conda-forge statsmodels

2. 資料正規化問題:

遲遲無法將桃園機場，客流量資料正規化。不確定是不是數字的位數問題沒整理好，所以只好先拿已經處理好的訓練資料集做切割。

3. 繪圖問題:

載入目錄沒有整理好一開始無法順利整理在同一張圖之中

六、未來期望

這一次選擇的主題較容易，希望在了解這些操作方式以及原理後 能夠證準確地運用在產品消費預測。並且藉由這個例子，熟練LSTM，能運用在上下文的分析。