Assignment1 COVID-19 Forecast Report

106062314 蔡政諺

Abstract

本次作業我使用自己手刻的 AutoRegression model。

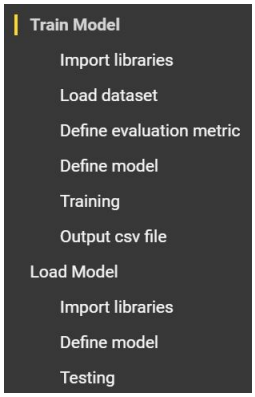
參數：DAY_NUM=50、WINDOW=7。

Link of my code

https://colab.research.google.com/drive/1-Onj3OJIWsvYQfl3PYDeP8IvMICCGobk

Descriptions

程式碼可以分為兩個 sections，分別是 **Train Model** 跟 **Load Model**。這段會講解 **Train Model** 的過程。



(1) 首先，import 需要的 library，如果使用 colab 的話，要 import google drive。

```
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets, linear_model
from sklearn.linear_model import LinearRegression

from google.colab import drive
drive.mount("/content/drive")
```

(2) 讀取 csv 檔中的資料。

```
path = "drive/My Drive/ML/Hw1/"
df = pd.read_csv(path + "input.csv")
print("Shape:", df.shape)
```

將資料中的負值修改為 0。

```
# set negative values to zero
num = df._get_numeric_data()
num[num < 0] = 0
```

讀出來的是包含 210 個國家資料的 DataFrame，我根據不同國家名稱，切成 210 個 DataFrame，並將 row 的順序反轉（這樣 index 愈大才會是愈新的資料），再 append 到一個 list 中。

```
# split data based on country
countries = df['countriesAndTerritories'].unique().tolist()
# countries.remove('Cases_on_an_international_conveyance_Japan')
data_split = []
for country in countries:
        # extract certain country's data
        data = df.loc[df.countriesAndTerritories == country]
        data = data.iloc[::-1] # reverse the order
        data_split.append(data)
```

(3) 定義 MAPE function，並使用作業公告中的數據測試。

```
# define MAPE evaluation function
def MAPE(label, pred):
    error = 0
    for i in range(len(label)):
        if label[i] == 0:
            pass
        else:
            error = error + abs((label[i] - pred[i]) / label[i])
    error = error * 100 / len(label)
    return error
```

```
# test MAPE function
predict = [39356, 40034, 38176, 34418, 32710, 32234, 36450]
ground_truth = [50209, 43567, 24598, 51473, 34841, 33871, 40820]
print("MAPE: ", MAPE(ground_truth, predict), "%")
```

```
MAPE:  19.959059690941643 %
```

(4) 定義 AutoRegression model。

```
# define autoregressive model
class AutoRegression:
    def __init__(self, train=[], window=0):
        self._coefficients = []
        self._intercept = 0
        self.train = train
        self.window = window
```

fit 是 training 時 call 的 function，內容就是解 multivariate linear regression，multivariate 的數量是根據 WINDOW 而定。

```
def fit(self):
    a = []
    for i in range(self.window):
        a.append(self.train[self.window-1-i:len(self.train)-1-i])
    a = np.transpose(np.array(a))
    b = np.array(self.train[self.window:len(self.train)])

    regr = linear_model.LinearRegression()
    regr.fit(a, b)

    self._intercept = regr.intercept_
    self._coefficients = regr.coef_
    return
```

predict 是 model train 完之後，用來預測結果的 function。argument 的 start 跟 end 代表我想要預測第幾天到第幾天的結果。例如 start=60、end=67 就代表我想要預測第 60 天到第 67-1 天的結果（最舊的資料為第 0 天）。

```
def predict(self, start, end):
    pred_list = []
    data = self.train.copy()

    for index in range(start, end):
        pred = self._intercept
        for i in range(self.window):
            pred = pred + (self._coefficients[i] * data[index-1-i])

        if index == len(data):
            data.append(pred)
        pred_list.append(pred)
    return pred_list
```

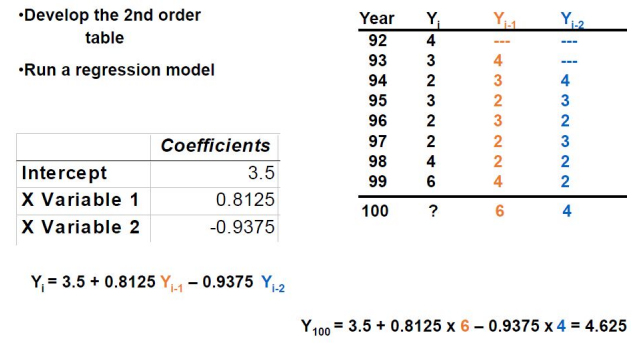
info 會顯示 model 的相關資訊。

```
def info(self):
    print("Coefficients:", self._coefficients)
    print("Intercept:", self._intercept)
    print("Train size:", len(self.train))
    print("Window:", self.window, "\n")
```

我的 model 格式是 csv 檔，因此 save 就是將 model 寫進一個 csv 檔，而 load 就是將 csv 檔中的數值讀回 model。

```
def save(self, filename):
    with open(filename, 'w', newline='') as file:
        writer = csv.writer(file)
        writer.writerow(self._coefficients)
        writer.writerow([self._intercept])
        writer.writerow(self.train)
        writer.writerow([self.window])
    return

def load(self, filename):
    with open(filename, newline='') as csvfile:
        reader = csv.reader(csvfile)
        rows = list(reader)
        self._coefficients = [float(i) for i in rows[0]]
        self._intercept = float(rows[1][0])
        self.train = [int(i) for i in rows[2]]
        self.window = int(rows[3][0])
    return
```

使用上課講義中的數據測試。




```
Coefficients: [ 0.8125 -0.9375]
Intercept: 3.4999999999999987
Train size: 8
Window: 2

[4.625000000000001, 1.6328125000000018, 0.49072265625, 2.3679504394531223, 4.9639072418212855, 5.313221096992493, 3.1633291020989462]
```

(5) 定義 parameters，並跑 for loop 去 train 所有國家的 AutoRegression model。

```
# note: DAY_NUM must be greater than WINDOW
DAY_NUM = 50
WINDOW = 7
```

DAY_NUM 是我 training 共使用幾天的 data，WINDOW 是 AutoRegression 的公式共 reference 幾天的 data，也就是上課講義的公式中的 p 值。

$$Y_i = A_0 + A_1 Y_{i-1} + A_2 Y_{i-2} + \cdots + A_p Y_{i-p} + \delta_i$$

取出每一個國家的最新 DAY_NUM 筆資料。train_X 是資料的 index，是一個[0, 1, ..., DAY_NUM-1]的 list。接著再從 DataFrame 中取出 case 這個 column 的值當 train_Y。然後就可以宣告一個 AutoRegression model 執行 fit。

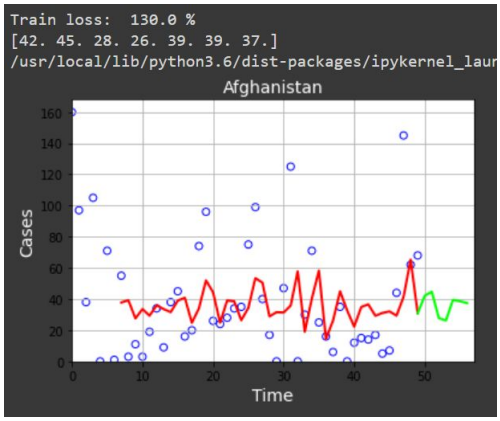
```
data = data[-DAY_NUM:].copy()

# training data
train_X = list(range(data.shape[0]))
train_Y = data["cases"].values.tolist()

# testing data
test_X = list(range(data.shape[0], data.shape[0]+7))

# build the model
model = AutoRegression(train_Y, window=WINDOW)

# train the model
model.fit()
```

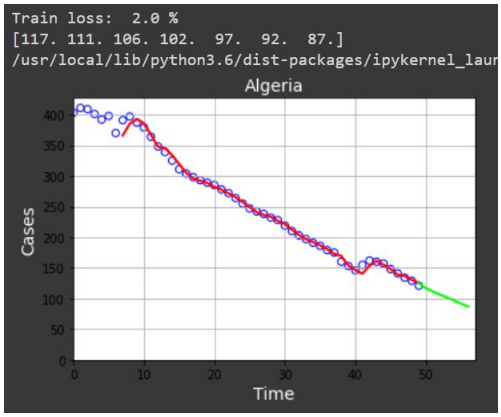
Model train 完之後，就可以用 predict 畫出 regression line，以及預測之後的結果。此外，也可以計算 ground truth 跟 prediction 之間的 MAPE。最後 predict 出來的結果會經過後處理，將負值設為 0，並四捨五入為整數。

```
# apply the model on training data
pred_Y = model.predict(start=WINDOW, end=data.shape[0])

# evaluate the model
loss = MAPE(train_Y[WINDOW:], pred_Y)
print("Train loss: ", round(loss), "%")
total_loss = total_loss + loss

# apply the model on testing data
test_Y = model.predict(start=data.shape[0], end=data.shape[0]+7)
test_Y = [0 if i < 0 else i for i in test_Y] # set negative values to zero
print(np.round(test_Y))
predictions.append(np.round(test_Y))
```

在座標平面點出 training data，並畫出 regression line（紅色部分是對已知日期的預測結果，綠色部分是對未來 7 天的預測結果）。

```
# plot training data and the regression line
plt.scatter(train_X, train_Y, color='', edgecolors='#0000FF') # training data
plt.plot(train_X[WINDOW:], pred_Y, color='#FF0000', linewidth=2) # regression line
plt.plot([train_X[-1]]+test_X, [pred_Y[-1]]+test_Y, color='#00FF00', linewidth=2) # regression line
plt.title(data.iat[0, 6], fontsize=14, color='#FFFFFF')
plt.xlabel('Time', fontsize=14, color='#FFFFFF')
plt.ylabel('Cases', fontsize=14, color='#FFFFFF')
plt.grid(True)
plt.ylim(bottom=0)
plt.xlim(0, data.shape[0]+7)
plt.show()
```





呼叫 save 儲存 model，到這裡就結束一個國家的 training process。

```
# save model
filename = '106062314_' + data.iat[0, 6] + '.csv'
model.save(path + "models/" + filename)
```

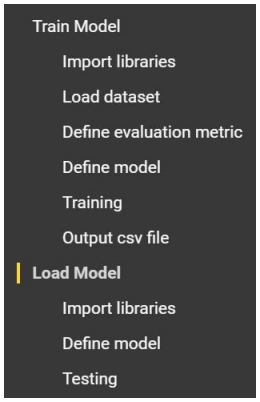
(6) 最後，將 model predict 的結果輸出成 7×210 的 csv 檔。

```
# write result into csv file
predictions = np.asarray(predictions).reshape(-1, 7)
predictions_tp = np.transpose(predictions)
print(predictions_tp.shape)
print(predictions_tp)
df = pd.DataFrame(predictions_tp)
df.columns = countries
df.index = ["2020/10/9", "2020/10/10", "2020/10/11", "2020/10/12", "2020/10/13", "2020/10/14", "2020/10/15"]
df.to_csv(path + "106062314_HW1.csv")
```

```
(7, 210)
[[ 42. 152. 117. ...   3.  91.  19.]
 [ 45. 145. 111. ...   2.  55.  25.]
 [ 28. 143. 106. ...   3.  79.  24.]
 ...
 [ 39. 137.  97. ...   4.  76.  28.]
 [ 39. 135.  92. ...   2.  73.  29.]
 [ 37. 134.  87. ...   3.  73.  28.]]
```

- **How to use the model file**

程式碼從第一頁連結中的 **Load Model** 這個 section 開始。



(1) 首先，import 需要的 library，如果使用 colab 的話，要 import google drive。

```
import csv
import numpy as np
from sklearn import datasets, linear_model
from sklearn.linear_model import LinearRegression

from google.colab import drive
drive.mount("/content/drive")
```

(2) 因為我是自己實作 model，所以 testing 時也要定義 AutoRegression 這個 class。

```
# define autoregressive model
class AutoRegression:
    def __init__(self, train=[], window=0):
        self._coefficients = []
        self._intercept = 0
        self.train = train
        self.window = window

    def fit(self):
        a = []
        for i in range(self.window):
```

(3) 定義 parameters，DAY_NUM 是我 training 共使用幾天的 data，path 是 model 的路徑。接下來就可以跑一個 for loop 將想測試的 model load 出來。最後 predict 出來的結果會經過後處理，將負值設為 0，並四捨五入為整數。

使用時只需修改 path 與 countries 兩者的內容即可。

```
DAY_NUM = 50
path = "drive/My Drive/ML/Hw1/models/"

countries = ["Afghanistan", "Albania", "Algeria", "Andorra", "Angola"]
for country in countries:
    # load model
    model = AutoRegression()
    filename = '106062314_' + country + '.csv'
    model.load(path + filename)

    # predict the future
    test_Y = model.predict(start=DAY_NUM, end=DAY_NUM+7)
    test_Y = [0 if i < 0 else i for i in test_Y] # set negative values to zero
    print(country, ":", np.round(test_Y))
```

```
Afghanistan : [42. 45. 28. 26. 39. 39. 37.]
Albania : [152. 145. 143. 140. 137. 135. 134.]
Algeria : [117. 111. 106. 102.  97.  92.  87.]
Andorra : [  0.  57.  28.   0. 380.   0. 300.]
Angola : [227.   0. 309.   0. 279.   0. 302.]
```