YouBike Prediction Report

● 資料處理

根據報告中的要求處理遺失資料，例如 11 月 25 日 11 點 25 分的資料遺失，就去檢查 26 分有沒有資料，沒有的話就再往下找 27 分，以此類推；如果是 23 點 59 分資料遺失，就要從 11 月 26 日的 0 點 0 分開始檢查。
下面三個模型使用的訓練資料時間範圍均為助教提供的資料中 10 月 2 號一直到 12 月 3 號；由於 10 月 11 號跟 10 月 15 號的晚上資料遺失太多，所以三個模型訓練的時候並未使用這兩天的資料。驗證資料為 12 月 11 號到 12 月 17 號，測試資料為 12 月 18 號到 12 月 24 號。

● 模型介紹

1. 平均值模型 Mean Value Model(MV model)

這個模型的訓練方式就跟其名稱一樣。對於各個站點，我將訓練資料以星期分類，然後對每個時刻的值都做平均值，得到 1440 個代表個時刻的平均值。舉例來說，對於捷運公館站 3 號出口，假如分類好後星期一的資料有 10 天，因此一天中的 1440 分鐘，每分鐘就有 10 筆數據，每分鐘各自對自己的 10 筆數據取平均，得到的值就是我們之後拿來預測這個站點，在星期一的每分鐘的值。但是為了考慮到這次的 evaluation metric 是帶有權重的，所以我是取加權平均，權重就是根據 evaluation metric 中的：

$$\left|\frac{b_{i,t}}{s_i}-\frac{1}{3}\right|+\left|\frac{b_{i,t}}{s_i}-\frac{2}{3}\right|$$

部分作為加權。

2. 梯度類型模型 Gradient Descent Type Model(GD model)

這個模型是以前 20 分鐘(包含當前分鐘)的站點資料，來預測下一週跟當前相同分鐘的腳踏車數量，例如要預測公館站 3 號出口在 11 月 26 號 11 點 26 分的腳踏車數量，輸入資料就會是 11 月 19 號 11 點 07 分到 11 點 26 分的資料。模型更新的方式是使用題目給的 evaluation metric 作為 loss function，使用 scipy.optimize.minimize，method 根據 ChatGPT 的建議以及個人的測試後使用 L-BFGS-B，以用於帶絕對值的 loss function，參數的部分均按照 scipy 內建預設，這裡只有選擇使用 L-BFGS-B 作為 method 而已。預測的時候，我讓 0 點 0 分到 0 點 18 分的預測數值跟 0 點 19 分的預測數值一樣。

3. 1-D CNN 神經網路模型 1-D CNN model(CNN model)

這個模型使用了經常用於時間序列預測的 1 維 CNN 神經網路來進行預測。這裡以前 1440 分鐘(包含當前分鐘)的站點資料，來預測下一週跟當前相同分鐘的腳踏車數量，例如要預測公館站 3 號出口在 11 月 26 號 11 點 26 分的腳踏車數量，輸入資料就會是 11 月 18 號 11 點 27 分到 11 月 19 號 11 點 26 分的資料。

建構模型採用了 torch 的模組，定義自己的卷積神經網路，下面是訓練時有關的參數以及配置：

- 架構使用了兩層一維卷積，一層平均池化層和一層機率為 0.5 的 dropout 層。
- optimizer 使用 Adam，1.126E-4 的 learning rate，和係數 1.126E-6 的 L2 regularizer；同時 optimizer 搭配 learning rate scheduler，採用 StepLR，每經過 4 個 epoch 就將 learning rate 乘以 0.1。
- batch size 為 64，並且訓練 10 個 epoch。
- loss function 一樣採用 error metric。

下面是 forward 時的傳遞路徑：

layer	parameters	input
Conv1d_1(nn.Conv1d)	in_channels=1, out_channels=32, kernel_size=20,stride=1	1x1440
ReLU(nn.ReLU)		
Conv1d_2(nn.Conv1d)	in_channels=32, out_channels=64,kernel_size=20,stride=1	32x1421
ReLU(nn.ReLU)		
AvgP_1(nn.AvgPool1d)	kernel_size=20, stride=20	64x1402
flat(nn.Flatten)	start_dim=1,end_dim=-1	64x70
Drop1(nn.Dropout)	p=0.5	64x70
dense(nn.Linear)	in_features=4480,out_features=1	4480
ReLU(nn.ReLU)		

● 預測成績

預測時均為預測每天的 0 點 0 分到 23 點 59 分，沒有以 20 分鐘為間隔。

	MV model	GD model	CNN model
E_{val}平均值	0.30704	0.35262	0.33152
E_{val}標準差	0.10809	0.13656	0.12976
E_{out}平均值	0.42588	0.51080	0.46772
E_{out}標準差	0.22705	0.28022	0.21245

平均值跟標準差為 112 個站，在七天預測的平均值中的平均值跟標準差。

● 模型比較

從E_{val}跟E_{out}的平均值來比較的話，GD model 皆不及 MV model 跟 CNN model；而 MV model 跟 CNN model 則是 MV model 均低於 CNN model；標準差的部分，GD model 一樣有著較大的值；MV model 跟 CNN model 則是 MV model 測試資料些微高於 CNN model，驗證資料則低於 CNN model。綜合兩點來看，GD model 的表現均遜於另外兩個 model。MV model 跟 CNN model 則是 MV model 表現較好，並且從標準差的部分可以看出兩者預測的變異性都較 GD model 低。

接著從實作層面以及訓練時間的部分來看，本次的平台為 AMD 5900X，32G DDR4 記憶體，NVDIA RTX3080 以及 PCIE gen4 SSD。
首先 CNN model 訓練時間最久，並且只有他有使用 GPU 訓練，需要 49 分鐘 36.1 秒，還有著較高的建構複雜度，以及要求更多的預先知識；接著是 GD model，需要 29 分鐘 17.8 秒，建構上的複雜度不低，也需要有一定的知識；最後是 MV model，訓練時間最短，只需要 1 分鐘 42.6 秒，建構上的複雜度最低，而且知識門檻也非常的低，並且因為其根本上就是在算平均值，具有非常高度的可解釋性。

容量大小方面，CNN model 儲存所需的大小最大，需要大約 20MB 左右的空間，MV model 則是其次，大約 8MB，GD model 最少，只要 2MB。此處都是利用 python 的 pickle 模組做資料儲存。

預測所需資源的部分，根據運算時間為基準的話，MV 最少，只需要 5 秒左右，GD model 其次，大約 15 秒左右，CNN 模型最長，需要近 1 分鐘。

至於日後更新方面，MV model 具有較大的隨時更新彈性，而 CNN model 需要對網路的內部層做修改，需要一點技術，GD model 則是可能要考慮改採用帶有 SGD 單一資料用來更新的性質的 method。

● 推薦選擇

經過上述的評估，我這裡鄭重的推薦 MV model 做為日後預測的模型，訓練快，容易理解跟解釋，預測效果佳且快速，易於更新，甚至在低開發程度的地區，如果沒有電子設備的環境下也可以直接人工實作。以上的種種優點，使得我認為 MV model 是最佳的選擇。
如果一定要說缺點的話，那就是一般人如果知道我們採用了 MV model 可能會覺得他也可以來做資料科學或機器學習…。

● 給公司的建議

在整理資料的過程中，其實不難看出貴公司會在凌晨進行車的調動，然而這成了我們團隊這次的些許困擾：
首先，每次調動的時間點沒有固定，這可能難以使我們的模型學會調動的時機，降低預測的準確性；再來每次車子調動的量也沒有固定的範圍，可能有時站點調整後的車輛很少，有時卻很多，這對於我們這次的 evaluation metric 有著極大的影響，因為我們的模型有可能是因為這些突然與往常不同的調動，導致預測失準，而恰好該時候的量是很少或是很滿的，使得錯誤量偏高。
或者我們團隊可以嘗試使用更多或更深入的資料處理，讓模型學會公司的相關調動行為，近一步可以推測出調動的時間點等等資訊，以應對這個問題；如有這個需求，未來我們會嘗試開發相對應的模型。

● Reference

GD model：
1. scipy.optimize.minimize 的使用方法
https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html
2. GPT 的建議
https://chat.openai.com/share/5869d231-40a1-407f-a464-12fb1e466feb
CNN model
1. 建構時是詳細再詳細的閱讀 torch 官網的 document。
2. 認知到 CNN 可以處理時間序列問題：
https://bc165870081.medium.com/%E6%99%82%E9%96%93%E5%BA%8F%E5%88%97%E7%9A%84ai%E4%BB%8B%E7%B4%B9-ff250cfc2ff9
https://blog.csdn.net/weixin_38346042/article/details/121742025
https://www.twblogs.net/a/5e905b7dbd9eee342090101e
https://machinelearningmastery.com/how-to-develop-convolutional-neural-network-models-for-time-series-forecasting/
雖然以上都是 tensorflow 的，但後來有查找 pytorch 官網和一些網站，了解如何用 pytorch 建構 1D-CNN。

● 團隊分工

此為一人團隊，如有不足之處，敬請見諒。