首先拿到資料集的第一件事，就是要先對資料有一定的了解及處理才可以開始做模型訓練，所以我先對資料集的每一個 feature 去檢查是否有遺失值的存在，得到的結果如下圖。

```
Checking null value
Train dataset:
title              False
created_at         False
like_count_1h      False
like_count_2h      False
like_count_3h      False
like_count_4h      False
like_count_5h      False
like_count_6h      False
comment_count_1h   False
comment_count_2h   False
comment_count_3h   False
comment_count_4h   False
comment_count_5h   False
comment_count_6h   False
forum_id           False
author_id          False
forum_stats        False
like_count_24h     False
```

確認資料集皆沒有遺失值後，再來了解每一個 feature 的 data type 為何，以幫助我在選擇模型及資料預處理上有大致方向，結果如下。

```
Checking data type
title              object
created_at         object
like_count_1h       int64
like_count_2h       int64
like_count_3h       int64
like_count_4h       int64
like_count_5h       int64
like_count_6h       int64
comment_count_1h    int64
comment_count_2h    int64
comment_count_3h    int64
comment_count_4h    int64
comment_count_5h    int64
comment_count_6h    int64
forum_id            int64
author_id           int64
forum_stats       float64
like_count_24h      int64
```

對資料集有一定了解後，我選擇將“created_at”、”forum_id”、”author_id”和”forum_stats”這四項與預測結果相關度不高的 features 移除，並且因為 title 的型別為 string，要可以與其他 features 併入一起做訓練的話，要將 string 轉換成數字的型別，如 int 或 float。此時，我想到我學過的 transformer 的觀念，我決定將 title 去做 word embedding 將其轉換成數字 vector 再與其他 features 串接做成訓練集及驗證集。

根據網上查找到的資料，我選擇利用 pretrained 好的 Bert model 去做 tokenization，並利用 model 的 output 取得 embedding 的結果，每一個 title 固定皆為 768 維度。

```
torch.Size([50000, 768])
```

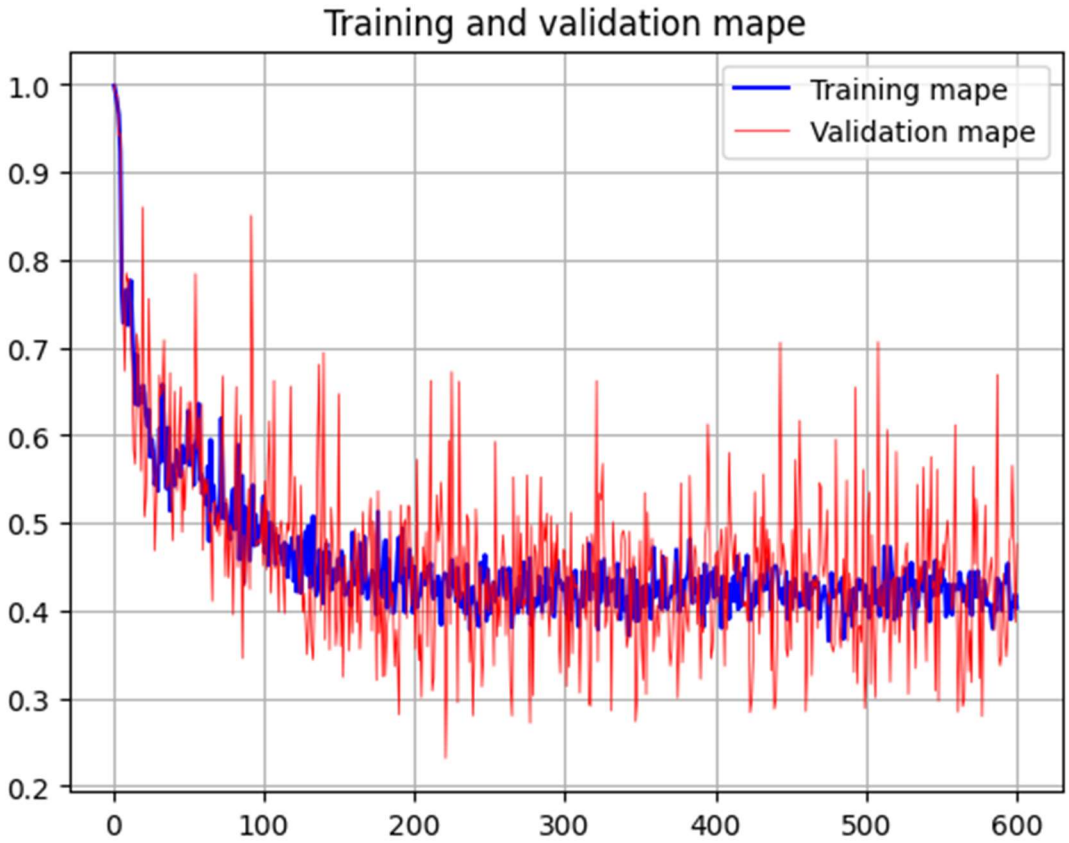
但若是直接將 768 維的 word embedding title 和剩下 12 維的 feature 結合在一起放入 neural network 去做訓練，得到的結果貌似模型會逐漸偏重那 768 維的 feature，使得原本給定的累積愛心數和累積評論數派不上用場，在訓練後期的 mape 不降反升，調整後的結果約在 70%~80%。

在調整 model 的期間我有嘗試使用 Random forest 和 Polynomial regression 去做結果評估，所得出的結果比先前的 NN 再上升一些，約在 85%~90%。

考慮到 word embedding features 影響可能有一點大，於是我利用一個自定義的 BertNetwork 先將 768 維經過兩層運算降維成 64 維，並利用另一個 FeatureNetwork 將累計愛心數和累積評論數從 12 維經過一層運算升維到 64，再將兩者串接起來的 128 維放入最終訓練的四層 Network，所得到的結果有明顯的下降一些，約在 55%~65%。

但在經過 Hyperparameters 的調整及運用 Learning rate scheduling 逐步降低 lr，所得到的結果也只有使 mape 降低幾個百分點。此時我注意到訓練過程的 loss 率震蕩不已，從幾百到幾萬都有，我便嘗試使用 SmothL1Loss 去取代 MSE loss 做模型訓練，得到的結果讓訓練過程沒有那麼震蕩，mape 更是下降了不少，約

在 40%左右。



最後再對模型架構和其他 Hyperparameters 做些許調整，便形成了最終的模型，

後記：

在繳交前一個晚上突然有想到利用 LSTM 之類的具有時間序列特性的模型對累積愛心數和累積評論數做處理應該會是好選擇，但礙於近期為期中考周，時間所剩無幾所以只能先選擇以目前模型繳交。

期中考周結束後會嘗試將以上的想法付諸實現，實驗看看結果是否會變好，後續會再推上 Github 上。