TAIPEI TECH 國立臺北科技大學

深度學習與類神經網路

作業：House Sale Price Prediction Challenge - Regression

研究生：余俊賢

學號：108368505

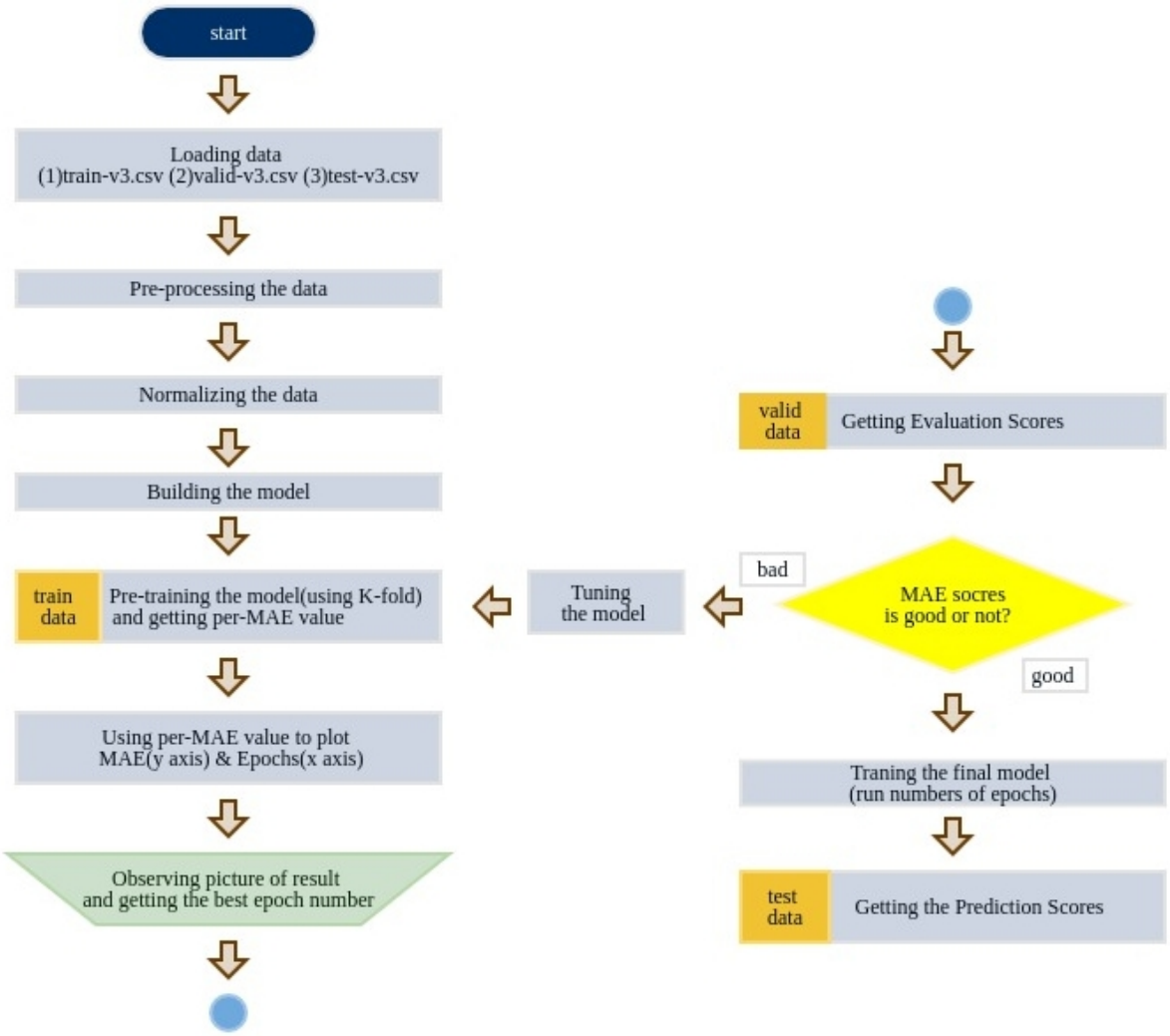
班級：電子工程系碩士 在職專班

指導教授：廖元甫 教授

1.做法說明

輸入 train data 訓練模型
使用 valid data 評估分數,利用此分數來判斷是否需要調整模型
使用 test data 帶入模型做出預測房價,上傳 kaggle 驗證

2.程式流程圖



3.程式寫法

new 一個 class pre_processing_data 類 .物件名稱為 all_data
在建構子時丟入檔案名稱
all_data.do_pre_processing_data() 做資料處理

```
all_data = pre_processing_data("train-v3.csv", "valid-v3.csv", "test-v3.csv")
all_data.do_pre_processing_data()
```

```
def __init__(self, train_data_path, valid_data_path, test_data_path):
    self.train_data_path = train_data_path
    self.valid_data_path = valid_data_path
    self.test_data_path = test_data_path
```

```
def do_pre_proccessing_data(self):
    self.load_data()
    self.train_targets, self.valid_targets = self.get_price_target()
    self.train_data = self.train_data_org
    self.valid_data = self.valid_data_org
    self.test_data = self.test_data_org
    #self.zipecode_type,self.zipecode_len  = self.get_zipcode_type(self.train_data_org)
    #self.train_data = self.process_zipcode(self.train_data)
    #self.valid_data = self.process_zipcode(self.valid_data)
    #self.test_data = self.process_zipcode(self.test_data)
    #print(self.train_data.shape)
    #print(self.valid_data.shape)
    #print(self.test_data.shape)

    self.train_data, self.valid_data, self.test_data = self.drop_id_price(self.train_data, self.valid_data, self.test_data)
    #self.train_data, self.valid_data, self.test_data = self.drop_id_price_zipcode(self.train_data, self.valid_data, self.test
    print(self.train_data.shape)
    print(self.valid_data.shape)
    print(self.test_data.shape)
    self.mean_std()
```

**綠色框框為改進方式但效果不佳,所以註解掉

讀取檔案

```
def load_data(self):
    self.train_data_org = pd.read_csv(self.train_data_path)
    self.valid_data_org = pd.read_csv(self.valid_data_path)
    self.test_data_org = pd.read_csv(self.test_data_path)
```

取得 train 與 valid house price

```
def get_price_target(self):
    train_price_temp = self.train_data_org.price.to_numpy()
    valid_price_temp = self.valid_data_org.price.to_numpy()
    return train_price_temp,valid_price_temp
```

移除 house_price

```
def drop_id_price(self,input_train, input_valid, input_test):
    #train_data_temp = self.train_data_org
    train_data_temp = input_train
    train_data_temp.drop(['id','price'],axis=1,inplace=True)

    #valid_data_temp = self.valid_data_org
    valid_data_temp = input_valid
    valid_data_temp.drop(['id','price'],axis=1,inplace=True)

    #test_data_temp = self.test_data_org
    test_data_temp = input_test
    test_data_temp.drop('id',axis=1,inplace=True)
    return train_data_temp.to_numpy(),valid_data_temp.to_numpy(),test_data_temp.to_numpy()
```

正規化

```
def mean_std(self):
    mean = self.train_data.mean(axis=0)
    self.train_data -= mean
    std = self.train_data.std(axis=0)
    self.train_data /= std

    self.valid_data-=mean
    self.valid_data/=std

    self.test_data-=mean
    self.test_data/=std
```

到此資料前處理完成

建構模型

```
from tensorflow.keras import models
from tensorflow.keras import layers
import tensorflow as tf
tf.config.experimental.list_physical_devices('GPU')
# 設定 Keras 使用的 Session

def build_model(input_shape1):
    model = models.Sequential()
    model.add(layers.Dense(2048, activation='relu',input_shape=(input_shape1,)))
    model.add(layers.Dense(1024, activation='relu'))
    model.add(layers.Dense(512, activation='relu'))
    model.add(layers.Dense(1))
    model.compile(optimizer='adam', loss='mse', metrics=['mae'])
    return model
```

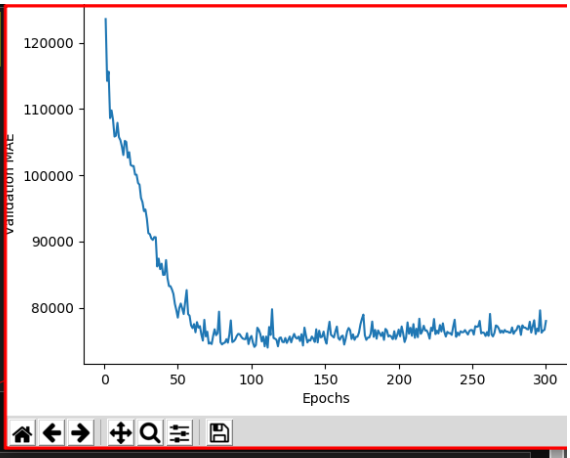
K-fold pre-training model
帶入 train data

```
batch_size_set = 30
import numpy as np
k = 4
num_val_samples = len(all_data.train_data) // k
num_epochs =300
all_mae_histories = []
for i in range(k):
    print('processing fold #', i)
    val_data = all_data.train_data[i * num_val_samples: (i+1) * num_val_samples]
    val_targets = all_data.train_targets[i * num_val_samples: (i+1) * num_val_samples]
    #print(val_data)
    #print(val_data.shape)

    trda1 = all_data.train_data[:i * num_val_samples]
    trda2 = all_data.train_data[(i + 1) * num_val_samples:]
    partial_train_data = np.concatenate([trda1,trda2],axis=0)
    #print(partial_train_data)

    trta1 = all_data.train_targets[:i * num_val_samples]
    trta2 = all_data.train_targets[(i + 1) * num_val_samples:]
    partial_train_targets = np.concatenate([trta1,trta2],axis=0)
    #print(partial_train_targets)

    length = all_data.get_data_shape1(all_data.train_data)
    print('processing fold  model.fit #', i)
    model = build_model(length)
    history = model.fit(partial_train_data, partial_train_targets,
                        validation_data=(val_data, val_targets),
                        epochs=num_epochs, batch_size=batch_size_set, verbose=0)
    print('processing fold model.fit ok #', i)

    #mae_history = history.history['val_mean_absolute_error']
    mae_history = history.history['val_mae']
    all_mae_histories.append(mae_history)
```

利用 pre-training model 得到 平均 MAE 並做出圖形協助判斷

```
average_mae_history = [
    np.mean([x[i] for x in all_mae_histories]) for i in range(num_epochs)]

import copy
#below best_epoch_num just for referencing the best epoch
best_epoch_num = 0
get_min = copy.copy(average_mae_history)
get_min.sort()
min_val = get_min[0]
print(min_val)
counter = 0
for i in average_mae_history:
    counter = counter + 1
    if i == min_val:
        best_epoch_num = counter
        print(f"best_epoch_num:{best_epoch_num}, min:{min_val}")
        break
```

```
73982.664
best_epoch_num:111, min:73982.6640625
```

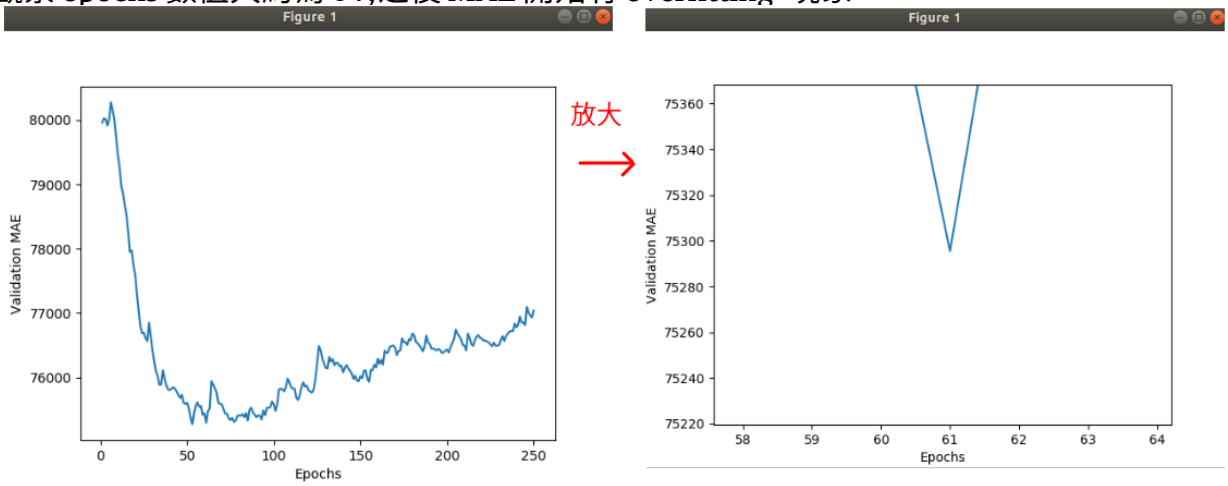



```
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
plt.plot(range(1, len(average_mae_history) + 1), average_mae_history)
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()
```

拿掉前 50 筆數值並平滑化方便觀察,找出下次正式訓練要帶入的 epochs 數值

```
ommit_observed_mae = 50
def smooth_curve(points, factor=0.9):
  smoothed_points = []
  for point in points:
    if smoothed_points:
      previous = smoothed_points[-1]
      smoothed_points.append(previous * factor + point * (1 - factor))
    else:
      smoothed_points.append(point)
  return smoothed_points

smooth_mae_history = smooth_curve(average_mae_history[ommit_observed_mae:])

plt.plot(range(1, len(smooth_mae_history) + 1), smooth_mae_history)
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()
```

觀察 epochs 數值大約為 61,之後 MAE 開始有 overfitting 現象



帶入從上得出的 epochs=111 正式訓練 model(model.fit)
使用 valid_data 評估結果(model.evaluate)

```
get_observed_mae = 61
best_epochs_num = ommit_observed_mae + get_observed_mae
print(best_epochs_num)
length = all_data.get_data_shape1(all_data.train_data)
model = build_model(length)
model.fit(all_data.train_data, all_data.train_targets,
                        epochs=best_epochs_num, batch_size=batch_size_set, verbose=0)
valid_mse_socre, valid_mae_score = model.evaluate(all_data.valid_data, all_data.valid_targets, verbose=0)
```

若不滿意在重新訓練 model
(調整 batch_size 或隱藏層)

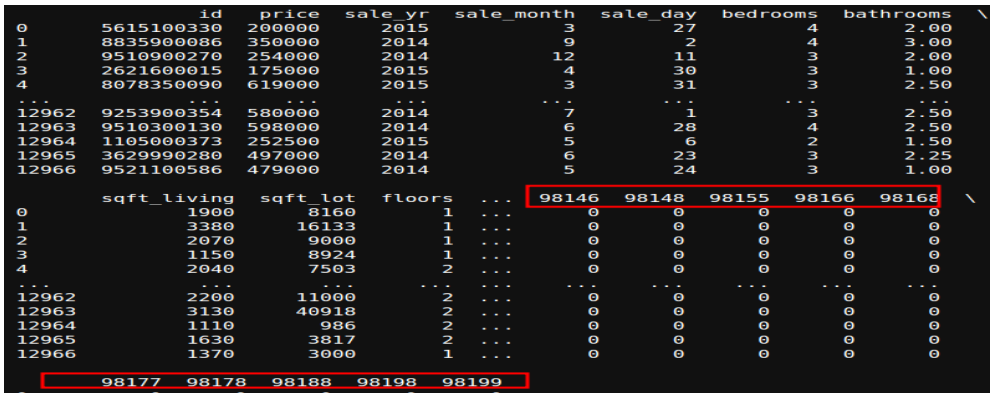
算出預測值

```
y = model.predict(all_data.test_data)
y
```

```
array([[682340.75],
       [731069.  ],
       [484481.97],
       ...,
       [308912.12],
       [415467.38],
       [204546.17]], dtype=float32)
```

4.如何改進?

改進失敗案例
將 zipcode 分割成 70 個欄位 ,屬於該欄位的給 1,否則為 0,
但還是保留原本的 zipcode



	id	price	sale_yr	sale_month	sale_day	bedrooms	bathrooms	\
0	5615100330	200000	2015	3	27	4	2.00	
1	8835900086	350000	2014	9	2	4	3.00	
2	9510900270	254000	2014	12	11	3	2.00	
3	2621600015	175000	2015	4	30	3	1.00	
4	8078350090	619000	2015	3	31	3	2.50	
...	...	...	...	...	...	...	...	
12962	9253900354	580000	2014	7	1	3	2.50	
12963	9510300130	598000	2014	6	28	4	2.50	
12964	1105000373	252500	2015	5	6	2	1.50	
12965	3629990280	497000	2014	6	23	3	2.25	
12966	9521100586	479000	2014	5	24	3	1.00	

	sqft_living	sqft_lot	floors	...	98146	98148	98155	98166	98168	\
0	1900	8160	1	...	0	0	0	0	0	
1	3380	16133	1	...	0	0	0	0	0	
2	2070	9000	1	...	0	0	0	0	0	
3	1150	8924	1	...	0	0	0	0	0	
4	2040	7503	2	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
12962	2200	11000	2	...	0	0	0	0	0	
12963	3130	40918	2	...	0	0	0	0	0	
12964	1110	986	2	...	0	0	0	0	0	
12965	1630	3817	2	...	0	0	0	0	0	
12966	1370	3000	1	...	0	0	0	0	0	

	98177	98178	98188	98198	98199

但實際卻沒有比為做 zipcode 分割前效果好

test.csv 70609.78956

7 hours ago by t108368505_余俊賢

(1)data add others zipcode column,which are number of zip code(other zipcode totally are 70), for example if zipcode column name is 68001,compares every rows original zipcode value,if they equals,give 1 (2)change layers below 2048, 1024, 512 (3)batch size = 250

沒做 zipcode 分割

test.csv 69816.38586

4 days ago by t108368505_余俊賢

change layers struct to test socre, 2048 1024 512

推判也許在資料前處理需要找到更大的關聯性,像 zipcode 可能就跟 price 關聯性不大 ,未來功課會找到相關工具去判斷資料關聯性