團隊測驗報告

報名序號：109601

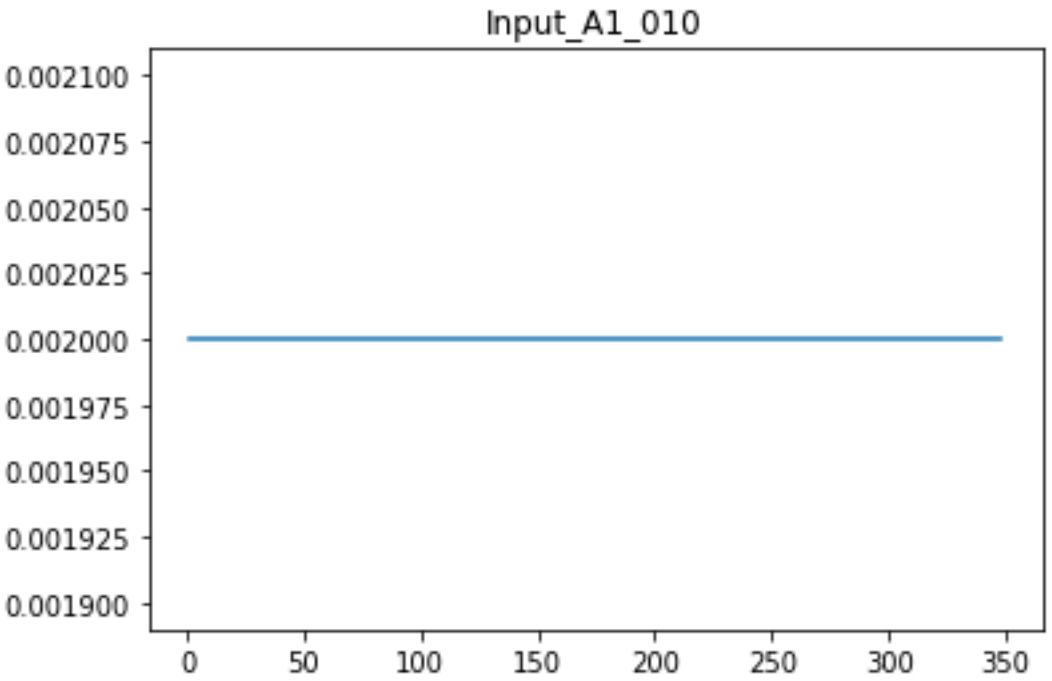
團隊名稱：老闆的免費勞工

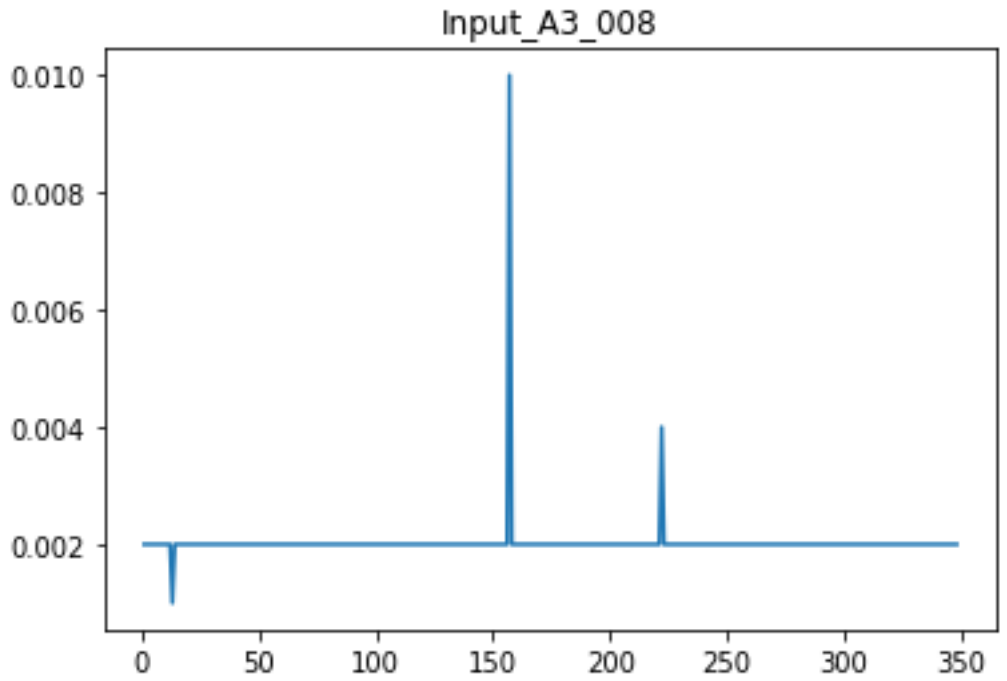
註1：請用本PowerPoint 文件撰寫團隊程式說明，**請轉成PDF檔案繳交**。
註2：依據競賽須知第七條，第4項規定：
測試報告之簡報資料不得出現企業、學校系所標誌、提及企業名稱、學校系所、教授姓名及任何可供辨識參賽團隊組織或個人身分的資料或資訊，違者取消參賽資格或由評審會議決議處理方式。

一、資料前處理(1/4)

去除單一數據

數據中有許多過於單一的數據，對模型在預測上會產生不好的影響，且使訓練速度下降，



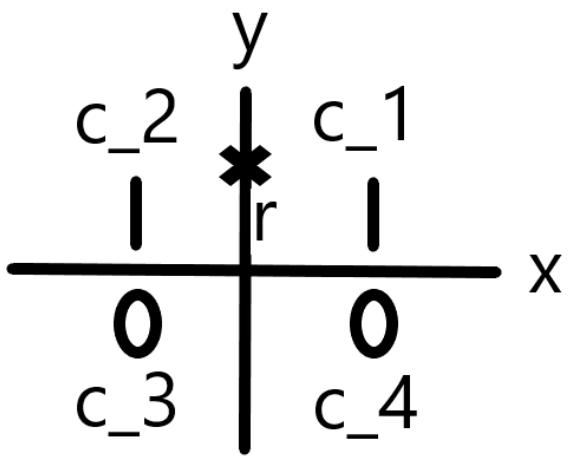


一、資料前處理(2/4)

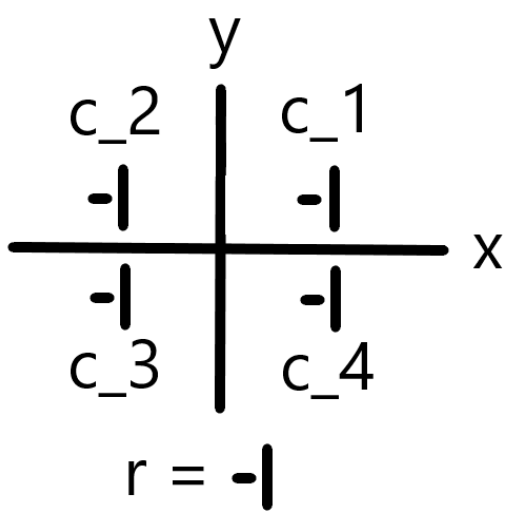
座標參數數據化

給予偏移量參數定義，設定象限一、二、三、四 為 C_1、C_2、C_3、C_4 ，若其存在於象限內則為1，不存在象限內則為0，偏移量 $r = \sqrt{(x^2+y^2)}$ ，對於nan值則全部給予 -1 當作default 值。

U;r;N;0; ⇒



nan ⇒



一、資料前處理(3/4)

缺失值處理

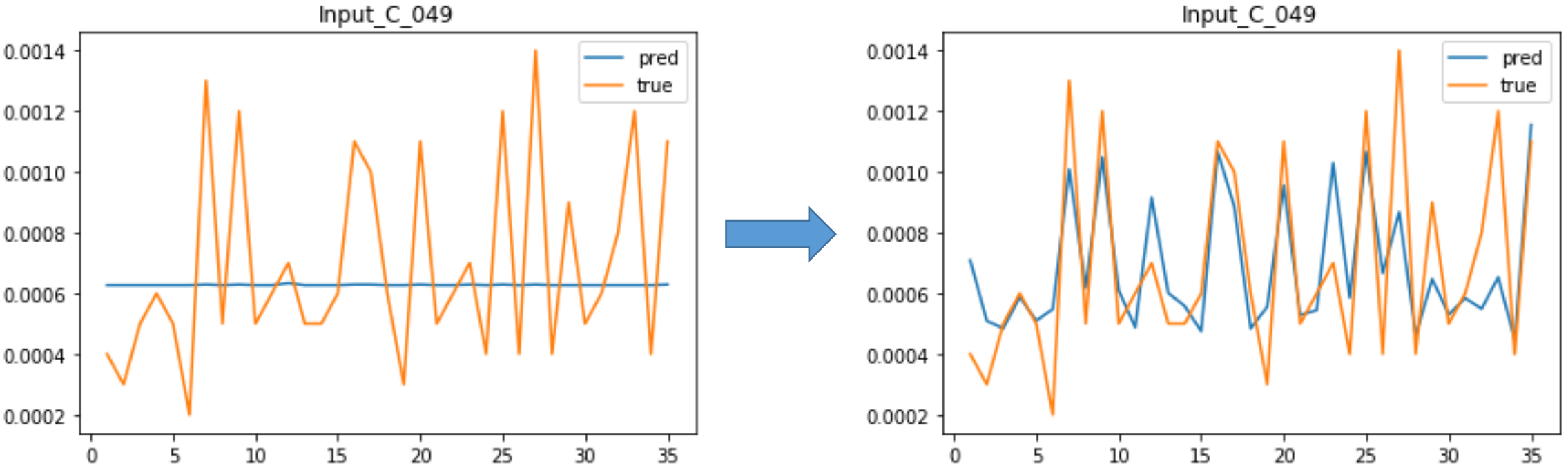
經由測試比較將缺失值補上眾數、平均、定值或者去除，最後發現補上眾數結果的誤差比較小。

```
df = df.fillna(df.mode().iloc[0])
```

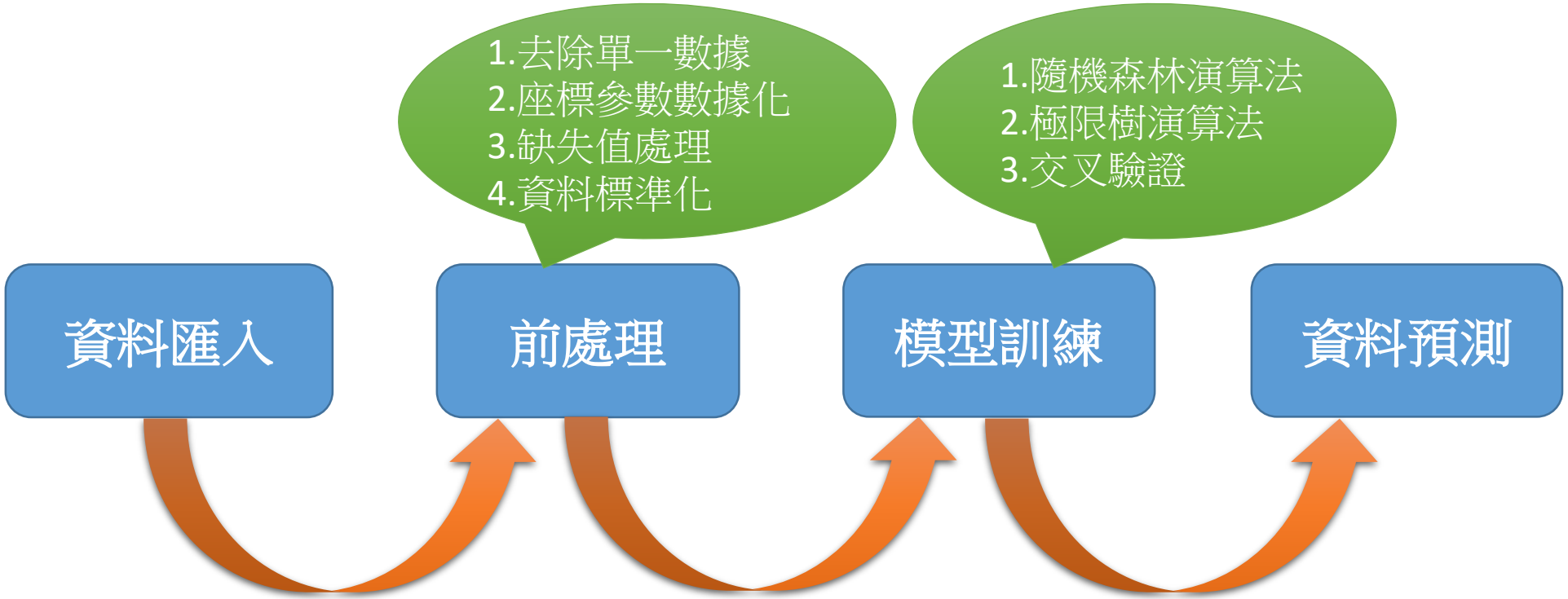
一、資料前處理(4/4)

資料標準化

有些預測項目因數值過小使模型欠擬合，先將項目標準化於0~1之間，再訓練後將預測結果反標準化。



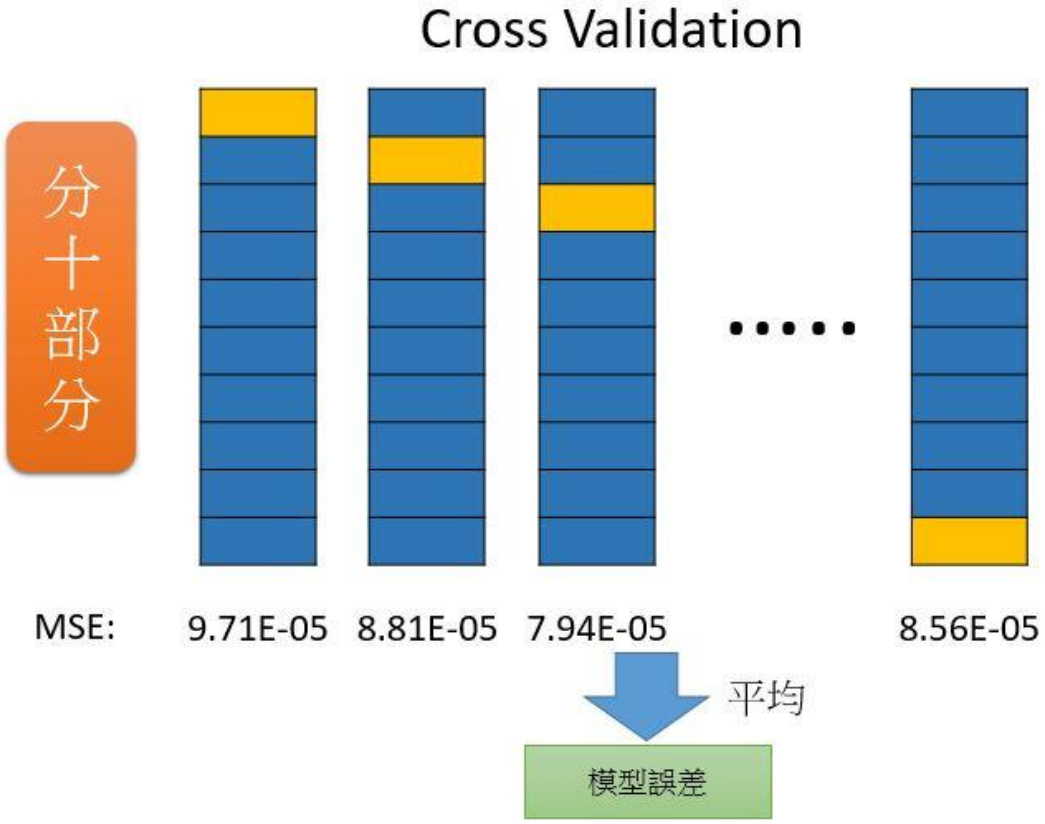
二、演算法和模型介紹(1/5)



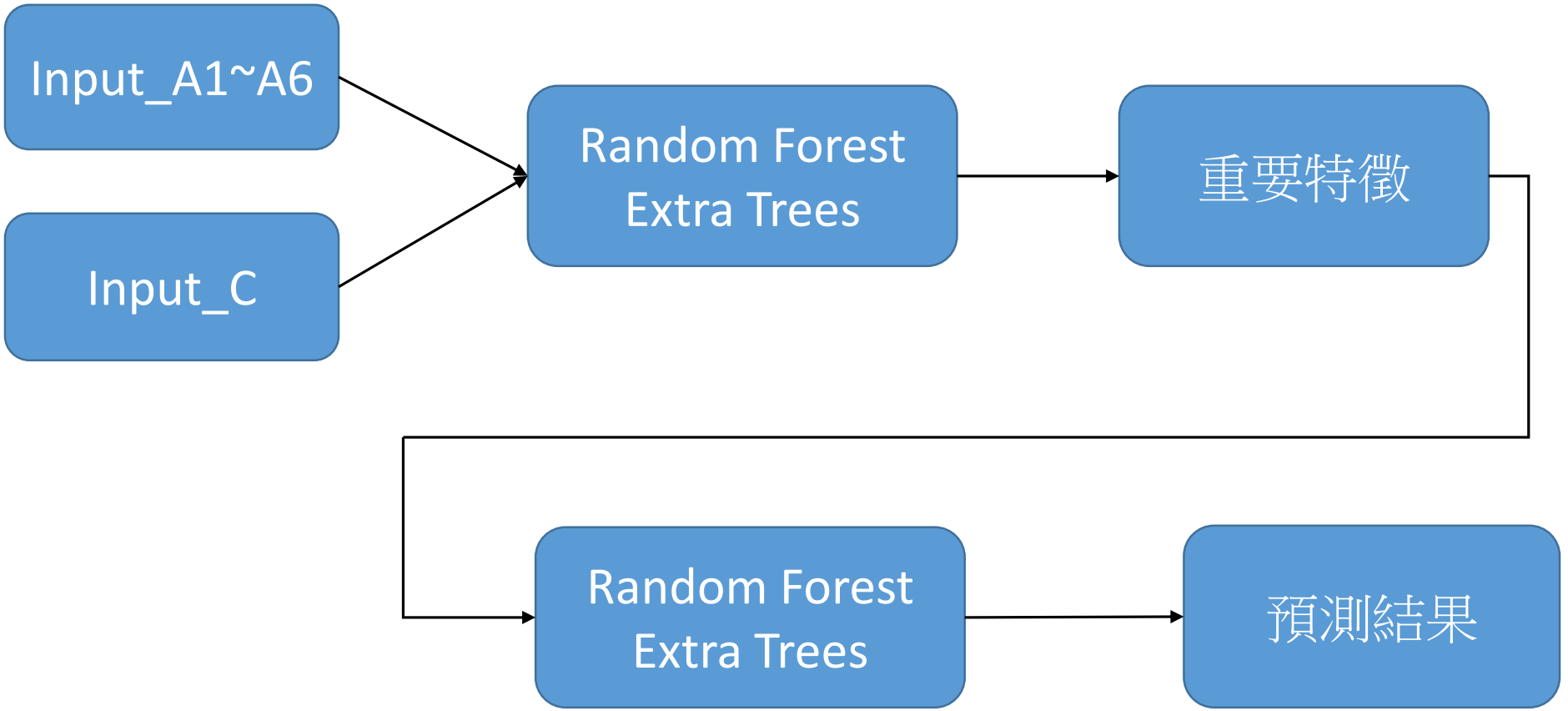
二、演算法和模型介紹(2/5)

交叉驗證：

- 將數據分為訓練、測試兩部分
- 將樣本資料切割成多個子集分別訓練
- 計算準確度平均值，驗證模型特性



二、演算法和模型介紹(3/5)



二、演算法和模型介紹(4/5)

訓練模型演算法程式說明(1/2)

資料引入

```
df = pd.read_csv('0714train.csv', header = 0)
test = pd.read_csv('0728test.csv' , header = 0)
df = df.drop(['Number'],axis=1)
test = test.drop(['Number'],axis=1)

#單一值去掉
df = df.drop(['Input_A1_010','Input_A2_008','Input_A2_010','Input_A4_010',
              'Input_A5_010','Input_A6_010','Input_C_002' ,'Input_C_003' ,
              'Input_C_006' ,'Input_C_132' ],axis = 1)

test = test.drop(['Input_A1_010','Input_A2_008','Input_A2_010','Input_A4_010',
              'Input_A5_010','Input_A6_010','Input_C_002' ,'Input_C_003' ,
              'Input_C_006' ,'Input_C_132' ],axis = 1)

#差異太小值去掉
df = df.drop(['Input_A1_008','Input_A3_008','Input_A3_010','Input_A4_008',
              'Input_A5_008','Input_A6_008','Input_C_004' ,'Input_C_009' ],axis = 1)

test = test.drop(['Input_A1_008','Input_A3_008','Input_A3_010','Input_A4_008',
              'Input_A5_008','Input_A6_008','Input_C_004' ,'Input_C_009' ],axis = 1)
```

去除單一數據

座標參數數據化

```
#將偏移量的文字參數改為極座標
object_cols = list((df.dtypes == 'object')[(df.dtypes == 'object')].index)
df[object_cols] = df[object_cols].fillna('F;-1;F;-1')
test[object_cols] = test[object_cols].fillna('F;-1;F;-1')
def polor(df):
    for i in range(len(object_cols)):
        locals()['C%s_r' % (i)] = []
    for i in range(len(object_cols)):
        locals()['C%s_1' % (i)] = []
    for i in range(len(object_cols)):
        locals()['C%s_2' % (i)] = []
    for i in range(len(object_cols)):
        locals()['C%s_3' % (i)] = []
    for i in range(len(object_cols)):
        locals()['C%s_4' % (i)] = []
    j= 0
    for name in object_cols:
        for i in df.index:
            a = df[name][i].split(";",3)
            if( a[0]=='N' and a[2]=='N') or (a[1]=='0'and a[3]=='0'):
                locals()['C%s_1' % (j)].append(0)
                locals()['C%s_2' % (j)].append(0)
                locals()['C%s_3' % (j)].append(0)
                locals()['C%s_4' % (j)].append(0)
                x = 0
                y = 0
            elif (a[1]=='0' or a[0]=='N') and a[3]!='0':
                if a[2]=='R':
                    locals()['C%s_1' % (j)].append(1)
                    locals()['C%s_2' % (j)].append(0)
                    locals()['C%s_3' % (j)].append(0)
                    locals()['C%s_4' % (j)].append(1)
                    x = float(a[3])
                    y = 0
```

⋮

```
            if a[0]!='F' and a[2]!='F':
                z = cmath.polar(complex(x , y))
                r = z[0]
                locals()['C%s_r' % (j)].append(r)
        j = j+1
    for j in range(len(object_cols)):
        df['C%s_r' % (j)] = locals()['C%s_r' % (j)]
        df['C%s_1' % (j)] = locals()['C%s_1' % (j)]
        df['C%s_2' % (j)] = locals()['C%s_2' % (j)]
        df['C%s_3' % (j)] = locals()['C%s_3' % (j)]
        df['C%s_4' % (j)] = locals()['C%s_4' % (j)]
    df = df.drop(object_cols,axis=1)
    return df

df = polor(df)
test = polor(test)
```

二、演算法和模型介紹(5/5)

訓練模型演算法程式說明(2/2)

缺失值處理

```
#將剩餘少量缺失值補眾數
df = df.fillna(df.mode().iloc[0])
test = test.fillna(test.mode().iloc[0])

#label
y = df[['Input_A1_020','Input_A2_016','Input_A2_017','Input_A2_024','Input_A3_013',
        'Input_A3_015','Input_A3_016','Input_A3_017','Input_A3_018','Input_A6_001',
        'Input_A6_011','Input_A6_019','Input_A6_024','Input_C_013' ,'Input_C_046' ,
        'Input_C_049' ,'Input_C_050' ,'Input_C_057' ,'Input_C_058' ,'Input_C_096'
        ]]

# 去除20項預測目標
df = df.drop(['Input_A1_020','Input_A2_016','Input_A2_017','Input_A2_024','Input_A3_013',
              'Input_A3_015','Input_A3_016','Input_A3_017','Input_A3_018','Input_A6_001',
              'Input_A6_011','Input_A6_019','Input_A6_024','Input_C_013' ,'Input_C_046' ,
              'Input_C_049','Input_C_050' ,'Input_C_057' ,'Input_C_058' ,'Input_C_096'], axis=1)
```

建立預測目標

訓練模型(1/20)

```
# 調整參數 A6_024

rnd_clf = ExtraTreesRegressor(n_estimators=10,random_state=42)
rnd_clf.fit(df , y[y.columns[12]])
feature_imp = pd.Series(rnd_clf.feature_importances_, index = df.columns).sort_values(ascending = False)
X_train_fe = df[feature_imp.index[:21]]
X_test_fe = test[feature_imp.index[:21]]
rnd_clf = ExtraTreesRegressor(max_features = 21,n_estimators=13,min_samples_leaf=1,max_depth=12)
rnd_clf.fit( X_train_fe,y[y.columns[12]] )
predictions = rnd_clf.predict(X_test_fe)
final["Input_A6_024"] = predictions
```

三、預測結果

將數據分成90%train、10%test 得到之方均根誤差。

測試數據	誤差	測試數據	誤差	測試數據	誤差	測試數據	誤差
Input_A1_020	0.456791	Input_A3_015	0.015646	Input_A6_011	0.000995	Input_C_049	0.000211
Input_A2_016	0.010353	Input_A3_016	0.011338	Input_A6_019	9.71E-05	Input_C_050	0.00133
Input_A2_017	0.008691	Input_A3_017	0.009748	Input_A6_024	0.001997	Input_C_057	0.004789
Input_A2_024	0.002442	Input_A3_018	0.010273	Input_C_013	0.000527	Input_C_058	0.00234
Input_A3_013	0.001042	Input_A6_001	0.000433	Input_C_046	0.000228	Input_C_096	0.005075