合金生產品質預測

使用資料集

資料集來源:

Kaggle：Metal-Furnace

描述:

此資料集紀錄了金屬熔爐中冶金製程的28 個匿名因素(編號為f0至f27)與其產品的合金品質。

專題目標與流程

01 找出冶金製程中與產品品質高度相關的關鍵因子

02 選擇不同的ML/DL模型訓練並評估結果

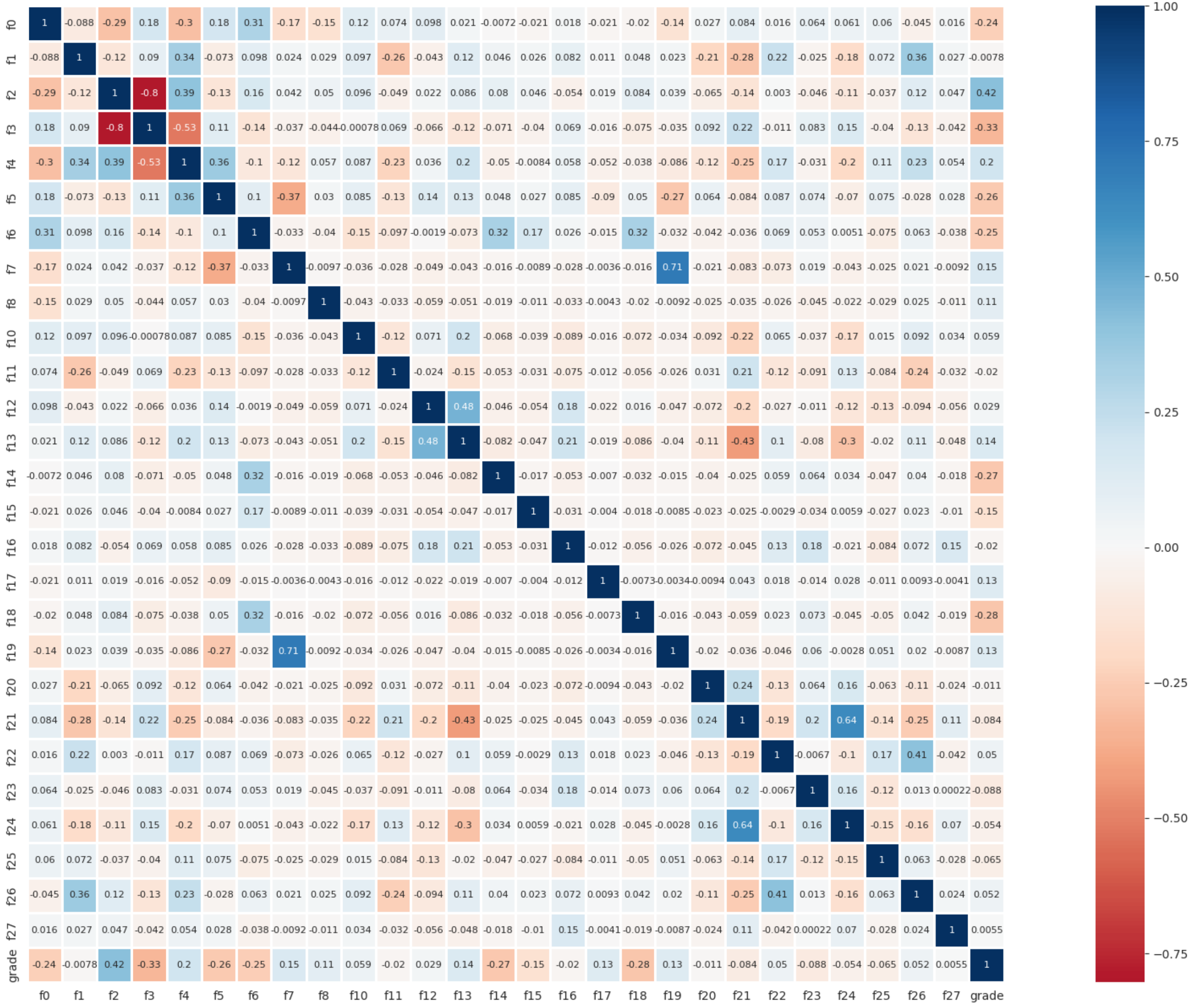
03 使用精準度較高的模型對新資料集進行預測

資料清洗

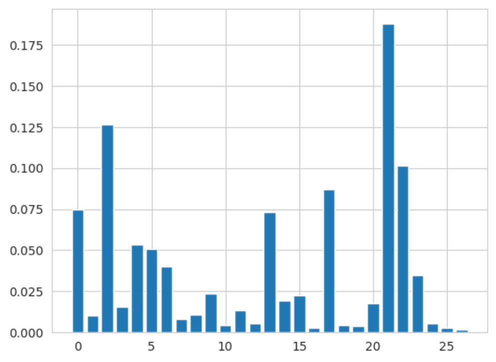
f9的欄位0值太多，選擇drop不放入模型訓練

```
[ ]   1 data = data.drop(['f9'],axis=1)
```

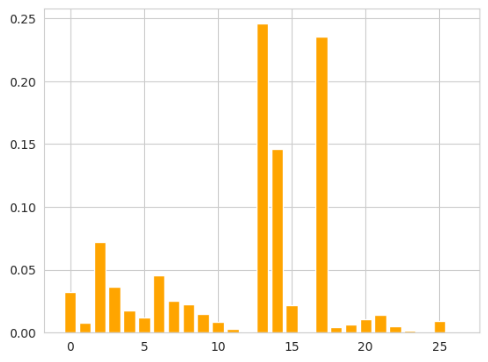
Correlation Matrix



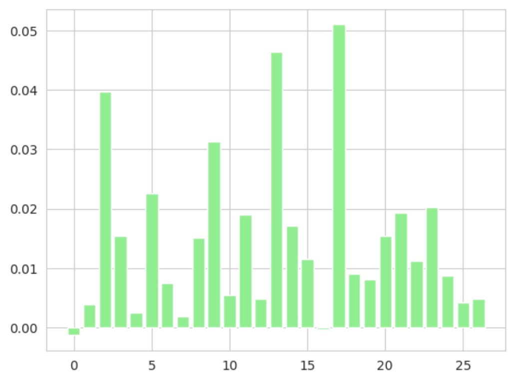
Feature Importance



<u>RandomForestClassifier</u>



<u>XGBClassifier</u>



KNeighborsClassifier

與產品品質高度相關的因子

Correlation Matrix	f2
Feature Importance	f12 f13 f16

切割資料集與載入模型

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
X = data.iloc[:,:-1]
y = data.iloc[:,-1]
train_X,test_X,train_y,test_y=train_test_split(X,y,test_size=0.2,random_state=17,shuffle=True,stratify=y)
```

ML模型選擇

RandomForestClassifier

GradientBoostingClassifier

XGBoostClassifier

DNN
(Deep Neural Network)

資料預處理

```python
import numpy as np
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense,Dropout
```

資料預處理

[40]
```python
print(train_y.shape)
```

```
(496,)
```

[42]
```python
train_y = train_y.values.reshape(-1, 1)
test_y = test_y.values.reshape(-1, 1)
```

[43]
```python
print(train_X.shape,train_y.shape)
```

```
(496, 27) (496, 1)
```

[47]
```python
from keras.utils import to_categorical

# 將標籤進行 one-hot 編碼
train_y = to_categorical(train_y, num_classes=5)
test_y = to_categorical(test_y, num_classes=5)
```

建立神經網路model

```
#建立模型
model  =  Sequential([
        Dense(300,input_dim=27,activation='relu'),
        Dropout(0.3),
        Dense(50,activation='relu'),
        Dropout(0.3),
        Dense(5,activation='softmax')
])
```

[17]

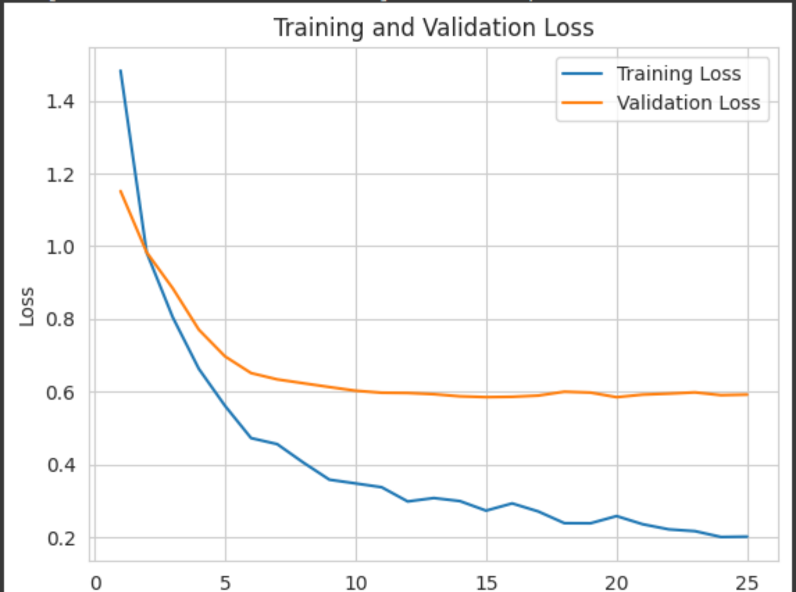
```
#  編譯模型
model.compile(optimizer='adam',
                    loss='categorical_crossentropy',
                    metrics=['accuracy'])

```

[18]

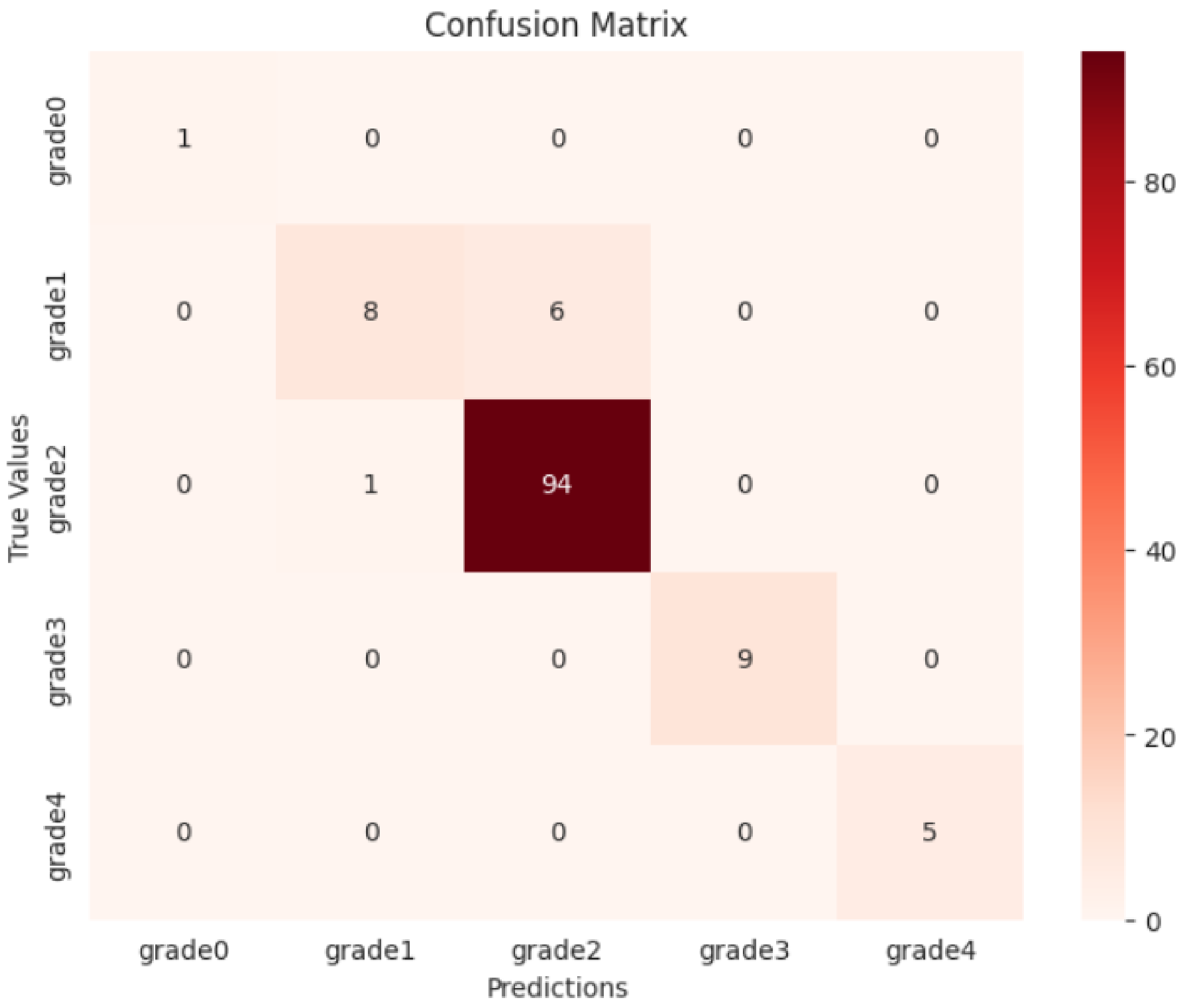
```
from  keras.callbacks  import  EarlyStopping

#  建立早停回調函數
early_stopping  =  EarlyStopping(monitor='val_loss',  patience=5,  restore_best_weights=True)
```

Training and Validation Loss Curves

```
[19] Epoch 16/100
9/9 [==============================] - 0s 7ms/step - loss: 0.2923 - accuracy: 0.8857 - val_loss: 0.5855 - val_accuracy: 0.7600
Epoch 17/100
9/9 [==============================] - 0s 6ms/step - loss: 0.2701 - accuracy: 0.8812 - val_loss: 0.5889 - val_accuracy: 0.7600
Epoch 18/100
9/9 [==============================] - 0s 6ms/step - loss: 0.2379 - accuracy: 0.9058 - val_loss: 0.5996 - val_accuracy: 0.7600
Epoch 19/100
9/9 [==============================] - 0s 8ms/step - loss: 0.2376 - accuracy: 0.9036 - val_loss: 0.5971 - val_accuracy: 0.7600
Epoch 20/100
9/9 [==============================] - 0s 7ms/step - loss: 0.2574 - accuracy: 0.8924 - val_loss: 0.5845 - val_accuracy: 0.7600
Epoch 21/100
9/9 [==============================] - 0s 7ms/step - loss: 0.2349 - accuracy: 0.9081 - val_loss: 0.5916 - val_accuracy: 0.7600
Epoch 22/100
9/9 [==============================] - 0s 6ms/step - loss: 0.2208 - accuracy: 0.9103 - val_loss: 0.5943 - val_accuracy: 0.7600
Epoch 23/100
9/9 [==============================] - 0s 6ms/step - loss: 0.2160 - accuracy: 0.9170 - val_loss: 0.5977 - val_accuracy: 0.7600
Epoch 24/100
9/9 [==============================] - 0s 8ms/step - loss: 0.1999 - accuracy: 0.9260 - val_loss: 0.5901 - val_accuracy: 0.7600
Epoch 25/100
9/9 [==============================] - 0s 9ms/step - loss: 0.2008 - accuracy: 0.9238 - val_loss: 0.5916 - val_accuracy: 0.8000
```



Confusion Matrix



預測新資料集

載入驗證集進行預測

＋ 程式碼　＋ 文字

```python
pre_data = pd.read_csv("/content/gdrive/My Drive/Test1.csv")
pre_data
```

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	...	f18	f19	f20	f21	f22	f23	f24	f25	f26	f27
0	-0.837812	-0.273636	1.276580	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	0.886135	-0.568935	1.100428	-0.244589	0.229718	-0.217109	0.087039
1	2.078087	-0.273636	-0.496119	0.463262	-2.438092	-0.24287	0.349804	0.12356	0.166795	0.06143	...	-5.059644	0.06143	0.27735	0.886135	0.504299	-0.434268	-0.244040	0.229718	-0.217109	0.087039
2	-0.837812	-0.273636	1.276580	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	-1.128496	-0.568935	-0.434268	-0.662763	0.229718	-0.217109	0.087039
3	-0.837812	-0.273636	-0.496119	0.463262	1.267808	-0.24287	-2.858743	0.12356	0.166795	0.06143	...	-5.059644	0.06143	0.27735	-1.128496	-0.449819	-1.918647	-0.662763	0.229718	-0.217109	0.087039
4	-0.837812	-0.273636	-0.496119	0.463262	-0.585142	-0.24287	-2.858743	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	-1.128496	-0.568935	-0.434268	-0.662763	0.229718	-0.217109	0.087039
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
261	0.411859	-0.273636	-0.496119	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	0.886135	-0.567744	-1.025755	-0.244040	0.229718	-0.217109	0.087039
262	-0.837812	-0.273636	-0.496119	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	-1.128496	2.410154	-0.434520	-0.662763	0.229718	-0.217109	0.087039
263	0.411859	-0.273636	-0.496119	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	0.886135	-0.448628	-0.434268	-0.662213	0.229718	-0.217109	0.087039
264	0.828416	4.569609	-0.496119	0.463262	1.267808	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	0.886135	0.504299	1.352018	-0.244040	0.229718	-0.217109	0.087039
265	-0.837812	-0.273636	-0.496119	0.463262	-0.585142	-0.24287	0.349804	0.12356	0.166795	0.06143	...	0.197642	0.06143	0.27735	-1.128496	2.410154	1.352018	-0.662763	0.229718	-0.217109	0.087039

266 rows × 28 columns

```python
pre_data = pre_data.drop(['f9'],axis=1)
```

輸出預測結果

使用預測準確率最高的模型(RandomForestClassifier)進行預測

```
TestPredictions = rfcgrid.predict(pre_data)
PredictResult = {'grade':TestPredictions}
Result = pd.DataFrame(PredictResult)
Result
```

	grade
0	2
1	3
2	2
3	3
4	2
...	...
261	2
262	2
263	1
264	2
265	2

266 rows × 1 columns

```
Result.to_csv("/content/gdrive/My Drive/predict_result.csv")
```

1		grade
2	0	2
3	1	3
4	2	2
5	3	3
6	4	2
7	5	2
8	6	2
9	7	2
10	8	2
11	9	2
12	10	1
[illegible]	11	2
[illegible]	12	2
15	13	2
16	14	3
17	15	2
18	16	1
19	17	2
20	18	2
21	19	2
22	20	2
23	21	2
24	22	2
25	23	2
26	24	2
27	25	2
28	26	2
29	27	2
30	28	2

Thank You For Watching!