

# 06XgbCV

November 4, 2018

## 1 XGB + CV

```
In [2]: import matplotlib.pyplot as plt
        from planar_utils import plot_decision_boundary, sigmoid, load_planar_dataset, load_ext
        import numpy as np
        import pandas as pd
        import os
        from collections import Counter

        from sklearn.neighbors import KNeighborsClassifier  ## KNN
        from sklearn.linear_model import LogisticRegressionCV  ## logistic regression
        from sklearn.tree import DecisionTreeClassifier  ## decision tree
        from sklearn.svm import SVC  ## SVM

        from sklearn.tree import DecisionTreeClassifier  ## decision tree
        from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
        from xgboost import XGBClassifier

        import math
        import string
        import re

        import xgboost

        from preprocess import preprocess
```

## 2 鐵達尼號資料集

```
In [3]: df = pd.read_csv('train.csv')
        df = preprocess(df)
        df.head()
```

```
Out[3]:
```

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Cabin	\
0	1	0	3	1	22.0	1	0	2	0	
1	2	1	1	0	38.0	1	0	5	3	
2	3	1	3	0	26.0	0	0	7	0	

3	4	1	1	0	35.0	1	0	1	3
4	5	0	3	1	35.0	0	0	1	0

	Embarked	Has_Cabin	Age_Cat	Fare_log2	Fare_Cat	Name_Length	\
0	0	0	1	2.857981	0	23	
1	2	1	2	6.155492	5	51	
2	0	0	1	2.986411	0	22	
3	0	1	2	5.730640	4	44	
4	0	0	2	3.008989	0	24	

	Name_With_Special_Char	Family_Size	Title
0	0	1	1
1	1	1	3
2	0	0	2
3	1	1	3
4	0	0	1

```
In [4]: X = df[['PassengerId', 'Pclass', 'Sex', 'Age', 'SibSp', 'Parch',
               'Ticket', 'Cabin', 'Embarked', 'Has_Cabin', 'Age_Cat', 'Fare_log2',
               'Fare_Cat', 'Name_Length', 'Name_With_Special_Char', 'Family_Size',
               'Title']].values
        Y = df['Survived'].values
```

```
In [5]: from sklearn.model_selection import train_test_split
```

```
X_train, X_valid, Y_train, Y_valid = train_test_split(X, Y, test_size =0.3, random_state=42)
print(X_train.shape) ## (445, 17)
print(X_valid.shape) ## (446, 17)
print(Y_train.shape) ## (445,)
print(Y_valid.shape) ## (446,)
```

```
(623, 17)
(268, 17)
(623,)
(268,)
```

```
In [6]: def get_accuracy(clf):
        #=====your works starts=====#
        clf = SVC()
        clf = DecisionTreeClassifier()
        y_pred = clf.predict(X_train)
        accuracy = sum(y_pred == Y_train) / len(Y_train)
        #=====your works ends=====#
        return accuracy

print('SVM: ', get_accuracy(SVC))
print('DecisionTree: ', get_accuracy(DecisionTreeClassifier))
print('RandomForest: ', get_accuracy(RandomForestClassifier))
```

```
print('AdaBoost: ', get_accuracy(AdaBoostClassifier))  ## Boosting 的演算法
print('XGB: ', get_accuracy(XGBClassifier))
```

```
# SVM: 0.609865470852
# DecisionTree: 0.764573991031
# RandomForest: 0.795964125561
# AdaBoost: 0.784753363229
# XGB: 0.80269058296
```

```
SVM: 0.6455223880597015
DecisionTree: 0.7611940298507462
RandomForest: 0.8544776119402985
```

```
"avoid this warning.", FutureWarning)
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

```
AdaBoost: 0.7910447761194029
XGB: 0.8432835820895522
```

```
In [7]: # Set our parameters for xgboost
```

```
params = {}
# 請填入以下參數:
# 目標函數: 二元分類
# 評價函數: logloss
# 學習速度: 0.04
# 最大深度: 5
#=====your works starts=====#
params['objective'] =
params['eval_metric'] =
params['eta'] =
params['max_depth'] =
#=====your works ends=====#
```

```
d_train = xgboost.DMatrix(X_train, label=Y_train)
d_valid = xgboost.DMatrix(X_valid, label=Y_valid)
```

```
watchlist = [(d_train, 'train'), (d_valid, 'valid')]
```

```
bst = xgboost.train(params, d_train, 100, watchlist, early_stopping_rounds=100, verbose=0)
y_pred = bst.predict(xgboost.DMatrix(X_valid))
print("Accuracy: ", str(sum(Y_valid == (y_pred > 0.5))/Y_valid.shape[0]))
```

```
[19:18:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:18:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:18:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:18:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
```

[illegible]



Accuracy: 0.835820895522388

### 3 房價資料集

```
In [8]: import urllib.request
        if 'df_realestate_processed.csv' not in os.listdir():
            url = 'https://s3.amazonaws.com/datasets-jeremy/df_realestate_processed.csv'
            urllib.request.urlretrieve(url, 'df_realestate_processed.csv')

        # processed
        path = "df_realestate_processed.csv"
        df_realestate_processed = pd.read_csv(path)
        X = df_realestate_processed.drop(["price_per_meter", "total_price"], axis=1)
        Y = df_realestate_processed['total_price']

In [9]: X_train = X.iloc[:-1000]
        Y_train = Y.iloc[:-1000]
        Y_train = np.log(Y_train)

        X_valid = X.iloc[-1000:]
        Y_valid = Y.iloc[-1000:]
        Y_valid = np.log(Y_valid)

In [10]: # Set our parameters for xgboost
        params = {}

        # 請填入以下參數:
        # 目標函數: 線性回歸
        # 評價函數: rmse
        # 學習速度: 0.01
        # 最大深度: 5
        # bst = xgboost.train(params, d_train, 3000, watchlist, early_stopping_rounds=50, verbose=1)
        #=====your works starts=====#
        params['objective'] =
        params['eval_metric'] =
        params['eta'] =
        params['max_depth'] =
        d_train =
        d_valid =
        watchlist =
        bst =
        Y_pred =
        #=====your works ends=====#

[19:18:33] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[0]          train-rmse:15.8112          valid-rmse:15.6215
Multiple eval metrics have been passed: 'valid-rmse' will be used for early stopping.
```

Will train until valid-rmse hasn't improved in 10 rounds.











[illegible]













```

[19:25:20] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[480]      train-rmse:0.238993      valid-rmse:0.331617
[19:25:21] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:22] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:22] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:23] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:24] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:25] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:26] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:26] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:27] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[490]      train-rmse:0.23864      valid-rmse:0.331323
[19:25:28] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:29] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:29] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:30] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:31] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:31] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:32] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:33] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:33] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:34] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[500]      train-rmse:0.238285      valid-rmse:0.331034
[19:25:35] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:36] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:36] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:37] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:38] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:38] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:39] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:40] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:41] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:41] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[510]      train-rmse:0.23793      valid-rmse:0.331124
[19:25:42] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[19:25:43] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
Stopping. Best iteration:
[502]      train-rmse:0.238198      valid-rmse:0.33097

```

```

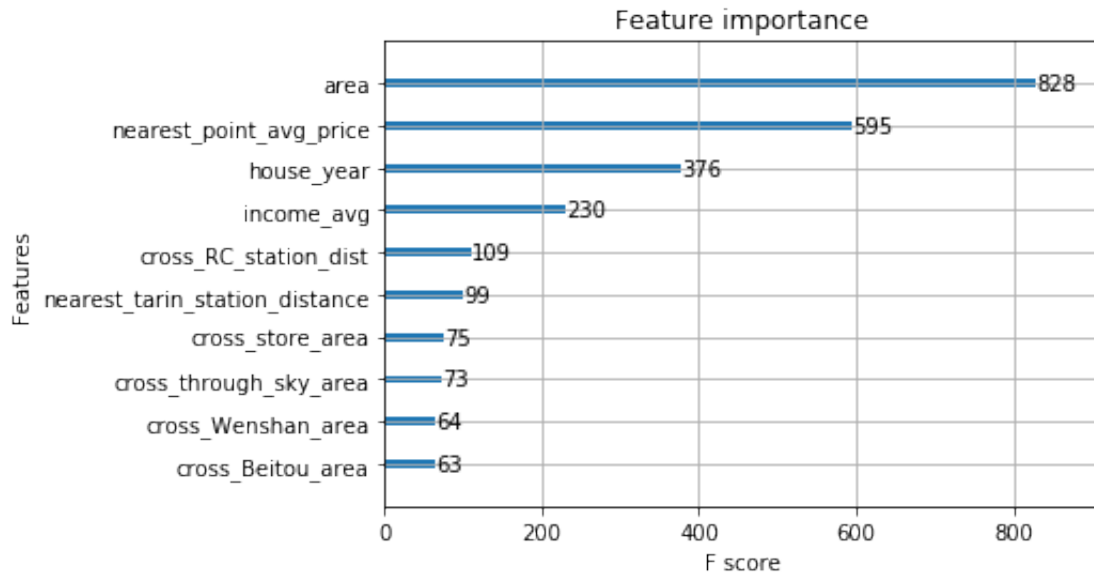
In [ ]: # 模型 save 與 load 的方式自己看
        # bst.save_model("bst_subtotal_log_with_cross.pickle.dat")
        # bst = xgboost.Booster({'nthread':1}) #init model
        # bst.load_model("bst_subtotal_log_with_cross.pickle.dat") # load data

In [11]: # 請使用 xgboost.plot_importance · 並設定 max_num_features=10
         #!=====your works starts=====!#

```

```
#!=====your works ends=====!#
```

```
plt.show()
```



```
In [12]: df_result = pd.DataFrame()
```

```
# 1. 使用 X_valid 去評價此模型
# 2. 使用 ['predict', 'truth', 'error'] 三個欄位的 DataFrame 去使決畫呈現預測結果
# (1). 請注意與測結果 (Y_pred) 與真實值 (Y_valid) 都必須取 exp 方能反映實際情況
# (2). error 請使用計算 (predict-truth)/truth 計算誤差百分比
#!=====your works starts=====#
Y_pred =
df_result['predict'] =
df_result['truth'] =
df_result['error'] =
df_result_sort =
#!=====your works ends=====#
```

```
df_result.head()
```

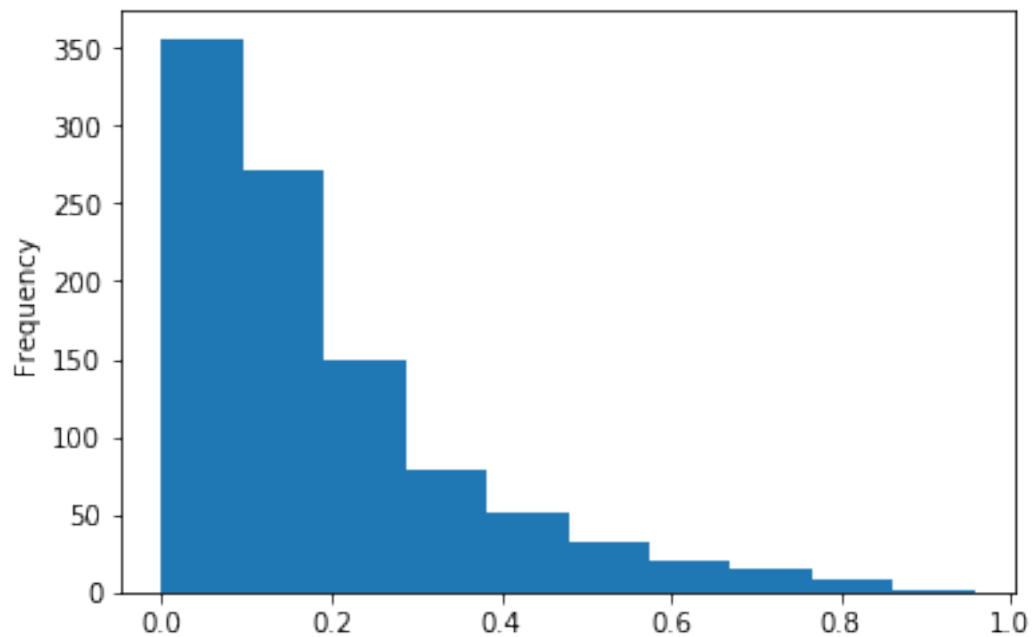
```
Out[12]:
```

	predict	truth	error
0	15413886.0	15880006.78	0.029353
1	12065383.0	10999982.00	0.096855
2	29951496.0	28199982.04	0.062110
3	23521218.0	21920043.69	0.073046
4	5400714.5	3220663.36	0.676895

```
In [13]: # 請使用 df_result_sort 濾掉 error 大於 1 的部分畫出 error 的分布圖
#!=====your works starts=====!#

#!=====your works ends=====!#

plt.show()
```



```
In [14]: # 請使用 plt.scatter 以 0~len(df_result) 作為 x，預測值（黑色）與實際值（紅色）作為 y。
#!=====your works starts=====!#

#!=====your works ends=====!#

plt.show()
```

