HR Analytics: Job Change of Data Scientists

Note:

多數特稱為類別，且部分為特徵有高比例為唯一值

特徵：

enrollee_id : 唯一ID city: 所在城市 city_ development _index : 城市發展指數 (連續) gender: 性別 relevent_experience: 相關經驗 enrolled_university: 大學課程類型 education_level: 教育程度 major_discipline :畢業學科 experience: 工作經驗年數 company_size: 當前公司員工數 company_type : 當前雇主類型 last_new_job: 上份工作與當前工作的年數差異 training_hours: 培訓時數 (連續) target: 0 不尋找新工作, 1 尋找工作

定義問題：了解特徵與 '學員願意到新公司上班而進行培訓' 之間的關係，意即具有何者特徵的學員，較有可能是為了找到新工作

In [16]:

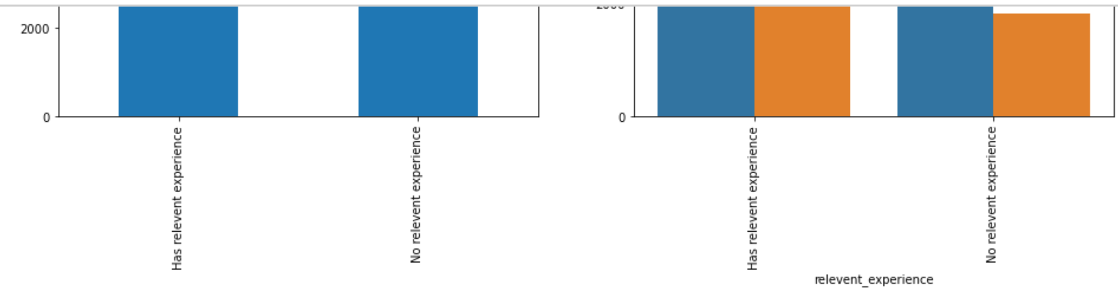
```
from sklearn.feature_selection import RFECV
from sklearn.pipeline import Pipeline
from imblearn.under_sampling import RandomUnderSampler
from sklearn.pipeline import Pipeline
import numpy as np
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split, RandomizedSearchCV, cross_val_score
from sklearn import metrics
from sklearn.metrics import mean_squared_error, f1_score, mean_absolute_error
from sklearn import model_selection
from sklearn import feature_selection
from xgboost import XGBClassifier
from sklearn import svm, tree, linear_model, neighbors, naive_bayes, ensemble, discrimina
from sklearn.preprocessing import LabelEncoder, OneHotEncoder, QuantileTransformer, Stanc
from sklearn.compose import ColumnTransformer
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

train = pd.read_csv('aug_train.csv')
train.info()
```

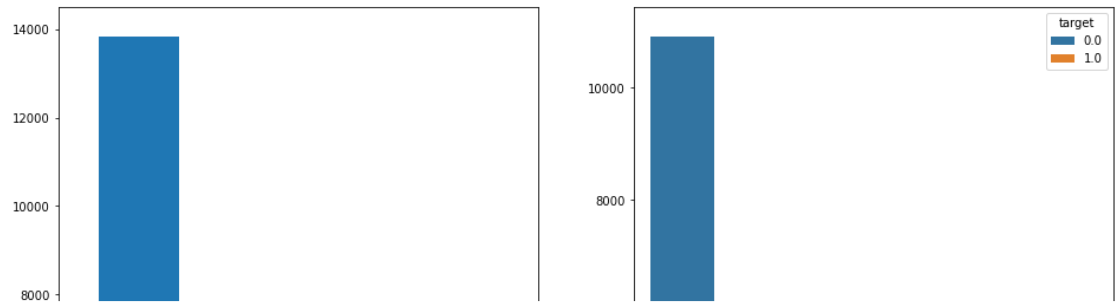
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19158 entries, 0 to 19157
Data columns (total 14 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   enrollee_id             19158 non-null  int64  
 1   city                    19158 non-null  object 
 2   city_development_index  19158 non-null  float64
 3   gender                  14650 non-null  object 
 4   relevent_experience     19158 non-null  object 
 5   enrolled_university     18772 non-null  object 
 6   education_level         18698 non-null  object 
 7   major_discipline        16345 non-null  object 
 8   experience              19093 non-null  object 
 9   company_size            13220 non-null  object 
 10  company_type            13018 non-null  object 
 11  last_new_job            18735 non-null  object 
 12  training_hours          19158 non-null  int64  
 13  target                  19158 non-null  float64
dtypes: float64(2), int64(2), object(10)
memory usage: 2.0+ MB
```

In [2]:

```
for col in train.columns:
    if train[col].dtypes == 'object':
        print(list(set(train[col])))
        fig, ax = plt.subplots(1, 2, figsize=(16, 10))
        train[col].value_counts().plot(kind='bar', ax=ax[0])
        plt.xticks(rotation=90)
        sns.countplot(x=col, hue='target', ax=ax[1], data=train)
        plt.show()
    else:
        continue
```




```
[nan, 'Part time course', 'Full time course', 'no_enrollment']
```



city: 過度集中，'city_21'較容易找新工作

experience: 大於20較多 (間隔5?) ['10', '17', nan, '14', '11', '18', '16', '4', '1', '<1', '9', '15', '12', '5', '2', '>20', '8', '7', '3', '6', '20', '13', '19']

company_size: 較分散 (小公司跳槽大公司?) ['100-500', '10/49', '1000-4999', nan, '<10', '5000-9999', '50-99', '10000+', '500-999']

education_level: Graduate較多，且較容易找新工作 ['High School', 'Masters', 'Phd', nan, 'Graduate', 'Primary School']

major_discipline: 過度集中STEM，且較容易找新工作 ['Humanities', nan, 'Other', 'Arts', 'Business Degree', 'No Major', 'STEM']

company_type: 過度集中Pvt Ltd ['Pvt Ltd', 'NGO', 'Early Stage Startup', nan, 'Public Sector', 'Funded Startup', 'Other']

last_new_job: 1年較多，never較容易找新工作 ['1', '2', nan, '3', '4', '>4', 'never']

gender: 多為男性，男女在target部分較無差異 ['Other', 'Male', nan, 'Female']

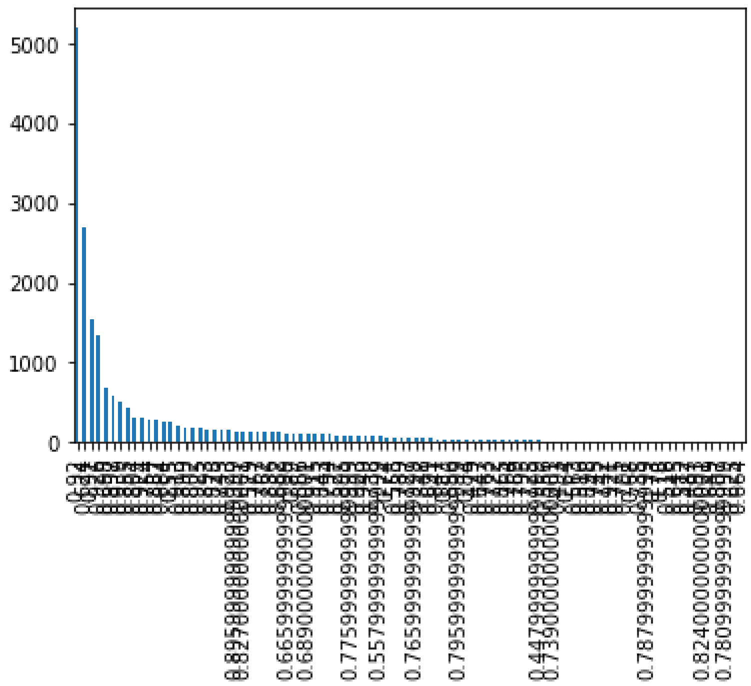
relevent_experience: 多具有相關經驗，且較不容易找新工作 ['Has relevent experience', 'No relevent experience']

In [3]:

```
train['city_development_index'].value_counts().plot(kind='bar')
# city_development_index: 和 city 分布相似
```

Out[3]:

<AxesSubplot:>

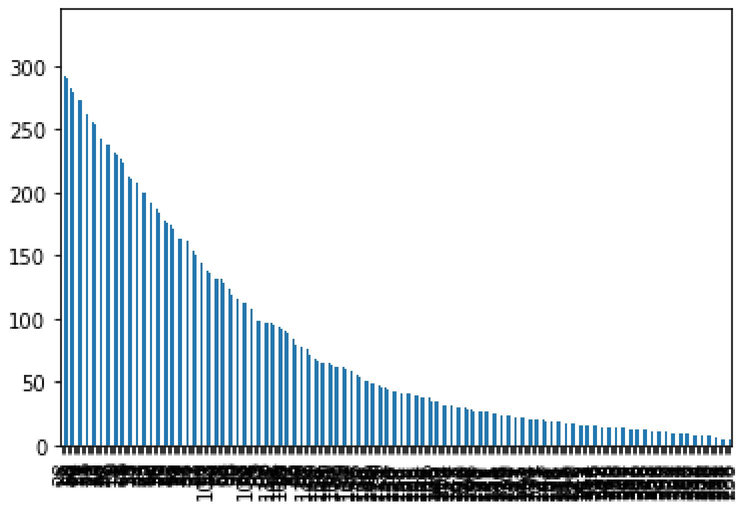


In [4]:

```
train['training_hours'].value_counts().plot(kind='bar')
# training_hours: 較分散(跟城市有關?)
```

Out[4]:

<AxesSubplot:>

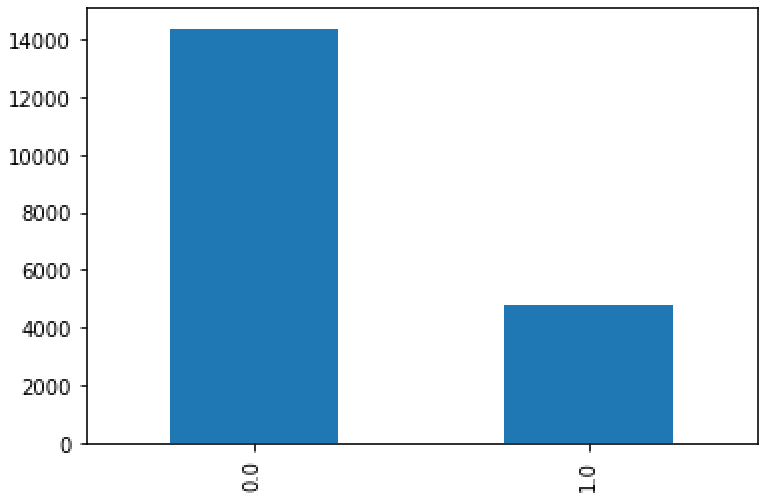


In [5]:

```
train['target'].value_counts().plot(kind='bar')
# target: 0約為1的3倍(不尋找工作? 找不到工作?)
```

Out[5]:

```
<AxesSubplot:>
```



In [6]:

```
train.columns[train.isnull().any().values == True]
```

Out[6]:

```
Index(['gender', 'enrolled_university', 'education_level', 'major_discipli
ne',
       'experience', 'company_size', 'company_type', 'last_new_job'],
      dtype='object')
```

In [7]:

```
pd.crosstab(train[col].fillna('missing'),train['target'].fillna('missing'), normalize=Tru
```

Out[7]:

target	0.0	1.0	All
target			
0.0	0.750652	0.000000	0.750652
1.0	0.000000	0.249348	0.249348
All	0.750652	0.249348	1.000000

In [8]:

```
for col in train.columns:
    if (len(set(train[col])) < 20) & (train[col].dtype == 'object'):
        print('correlation by :', col)
        print(pd.crosstab(train[col], train['target'],
              normalize=True, dropna=True, margins=True))
        print('-'*20, '\n')
```

```
correlation by : gender
target       0.0       1.0       All
gender
Female  0.062253  0.022253  0.084505
Male    0.696860  0.205597  0.902457
Other   0.009625  0.003413  0.013038
All     0.768737  0.231263  1.000000
--------------------

correlation by : relevent_experience
target                        0.0       1.0       All
relevent_experience
Has relevent experience  0.565351  0.154557  0.719908
No relevent experience   0.185301  0.094791  0.280092
All                      0.750652  0.249348  1.000000
--------------------

correlation by : enrolled_university
target                    0.0       1.0       All
enrolled_university
Full time course     0.123908  0.076231  0.200139
Part time course     0.047731  0.016088  0.063818
no_enrollment        0.580439  0.155604  0.736043
All                  0.752078  0.247922  1.000000
--------------------

correlation by : education_level
target                0.0       1.0       All
education_level
Graduate         0.446732  0.173548  0.620280
High School      0.086801  0.021072  0.107872
Masters          0.183228  0.050005  0.233234
Phd              0.019039  0.003102  0.022141
Primary School   0.014280  0.002193  0.016472
All              0.750080  0.249920  1.000000
--------------------

correlation by : major_discipline
target                 0.0       1.0       All
major_discipline
Arts              0.012236  0.003243  0.015479
Business Degree   0.014745  0.005262  0.020006
Humanities        0.032303  0.008626  0.040930
No Major          0.010278  0.003365  0.013643
Other             0.017069  0.006240  0.023310
STEM              0.654696  0.231936  0.886632
All               0.741328  0.258672  1.000000
--------------------

correlation by : company_size
target             0.0       1.0       All
company_size
10/49         0.085250  0.026021  0.111271
100-500       0.163086  0.031392  0.194478
1000-4999     0.085325  0.015129  0.100454
10000+        0.123601  0.029123  0.152723
50-99         0.191982  0.041225  0.233207
500-999       0.054841  0.011498  0.066339
5000-9999     0.034871  0.007716  0.042587
<10           0.081997  0.016944  0.098941
All           0.820953  0.179047  1.000000
```

```
--------------------

correlation by : company_type
target                    0.0       1.0       All
company_type
Early Stage Startup  0.035413  0.010908  0.046320
Funded Startup       0.066139  0.010754  0.076894
NGO                  0.032570  0.007451  0.040022
Other                0.007067  0.002228  0.009295
Public Sector        0.057228  0.016132  0.073360
Pvt Ltd              0.617760  0.136350  0.754110
All                  0.816178  0.183822  1.000000
--------------------

correlation by : last_new_job
target             0.0       1.0       All
last_new_job
1             0.315719  0.113424  0.429143
2             0.117427  0.037363  0.154790
3             0.042327  0.012330  0.054657
4             0.042754  0.012170  0.054924
>4            0.143582  0.032026  0.175607
never         0.091433  0.039445  0.130878
All           0.753243  0.246757  1.000000
--------------------
```

In [9]:

```
# 資料清理
# 具有順序的編號
clean_nums = {
    'enrolled_university': {'Full time course': 3, 'Part time course': 2, 'no_enrollment'
    'relevent_experience': {'Has relevent experience': 1, 'No relevent experience': 0},
    'education_level': {'Masters': 4, 'Phd': 5, 'High School': 2, 'Primary School': 1, 'G
    'company_size': {'<10': 0, '10/49': 1, '50-99': 2, '100-500': 3, '500-999': 4, '1000-
    'last_new_job': {'>4': 5, 'never': 0},
    'experience': {'>20': '25', '<1': '0'}
}

train = train.replace(clean_nums)

# 將city 編碼
label = LabelEncoder()
train['city'] = label.fit_transform(train['city'])

# major_discipline 與 company_type 刪除
train = train.drop(['major_discipline', 'company_type'], axis=1)

# NA 處理
# print(train['gender'].unique())
train['gender'] = train['gender'].fillna('Other')

# 假設NA代表在職中，且無經驗，教育程度為未受過教育，因此為0
train[['last_new_job', 'experience', 'education_level']] = train[[
    'last_new_job', 'experience', 'education_level']].fillna(0)

# 假設NA代表輟學，因此同屬於'no_enrollment'
train['enrolled_university'] = train['enrolled_university'].fillna(1)

# company_size 則依據 'education_level' 進行插補
train['company_size'] = train['company_size'].fillna(train.groupby(
    'education_level')['company_size'].transform('mean').round())

# gender 用 one-hot，照字母順序處理列名稱
onehot = OneHotEncoder()
train[['Female', 'Male', 'Other']] = onehot.fit_transform(
    train[['gender']]).toarray()

# training_hours 和 experience 分為5個級距
train['training_hours_cut'] = pd.qcut(
    train['training_hours'], 5, labels=[1, 2, 3, 4, 5])
train['experience_cut'] = pd.qcut(
    train['experience'].astype(int), 5, labels=[1, 2, 3, 4, 5])

for col in train.columns:
    if col not in ['gender', 'city_development_index']:
        train[col] = train[col].astype('int64')
train.head(5)
```

Out[9]:

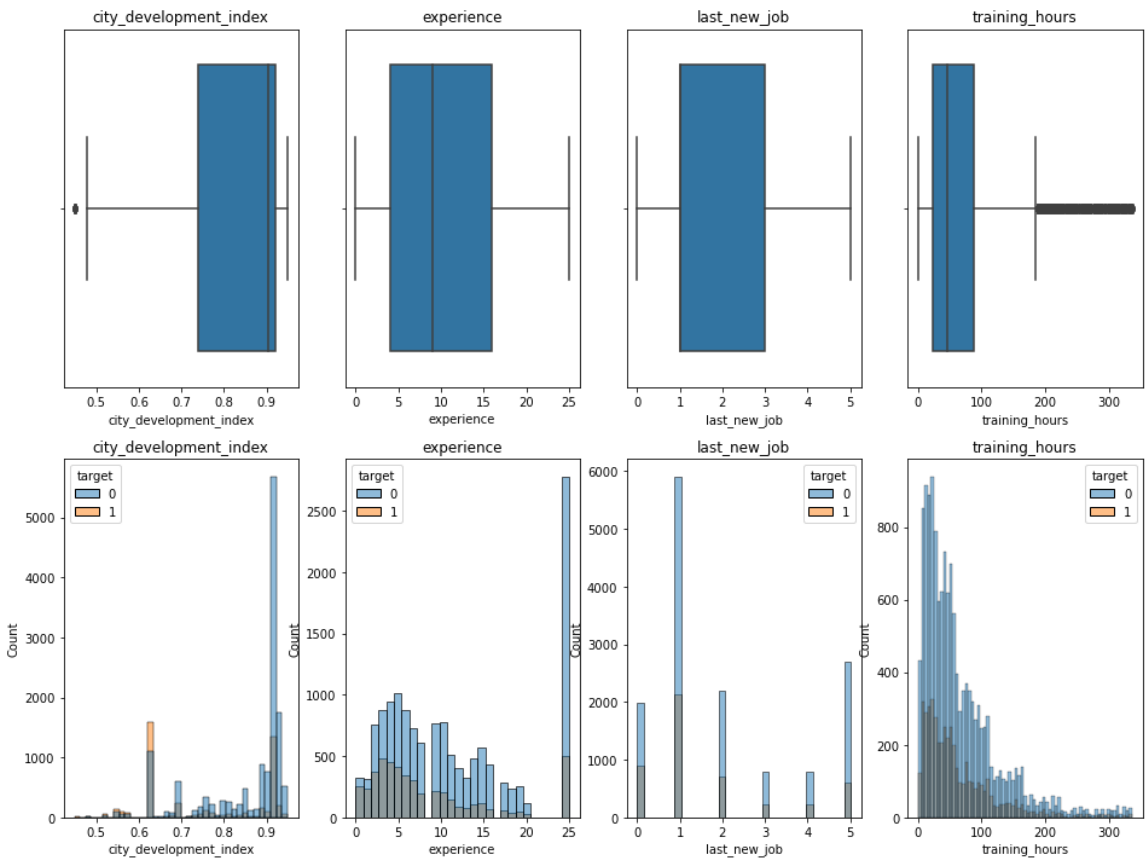
	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university
0	8949	5	0.920	Male	1	1
1	29725	77	0.776	Male	0	1
2	11561	64	0.624	Other	0	3
3	33241	14	0.789	Other	0	1
4	666	50	0.767	Male	1	1

In [10]:

```
fig, ax = plt.subplots(2, 4, figsize=(16, 12))
sns.boxplot(x='city_development_index', hue='target', data=train,
              ax=ax[0, 0]).set(title='city_development_index')
sns.boxplot(x='experience', hue='target', data=train,
              ax=ax[0, 1]).set(title='experience')
sns.boxplot(x='last_new_job', hue='target', data=train,
              ax=ax[0, 2]).set(title='last_new_job')
sns.boxplot(x='training_hours', hue='target', data=train,
              ax=ax[0, 3]).set(title='training_hours')
sns.histplot(x='city_development_index', hue='target', data=train,
             ax=ax[1, 0]).set(title='city_development_index')
sns.histplot(x='experience', hue='target', data=train,
             ax=ax[1, 1]).set(title='experience')
sns.histplot(x='last_new_job', hue='target', data=train,
             ax=ax[1, 2]).set(title='last_new_job')
sns.histplot(x='training_hours', hue='target', data=train,
             ax=ax[1, 3]).set(title='training_hours')
```

Out[10]:

```
[Text(0.5, 1.0, 'training_hours')]
```



In [11]:

```
y_re = train[['target']]
x_test = train.drop('target', axis=1)

# 將資料標準化
num_col = ['experience', 'company_size', 'last_new_job', 'training_hours_cut',
           'experience_cut', 'education_level', 'enrolled_university']
skew_col = ['city_development_index', 'training_hours']


norm_num = Pipeline(
    steps=[
        ('scaler', StandardScaler())
    ]
)
skew_num = Pipeline(
    steps=[
        ('Quantile', QuantileTransformer(output_distribution='normal')),
        ('scaler', StandardScaler())
    ]
)
preprocess = ColumnTransformer(
    [
        ('norm', norm_num, num_col),
        ('skew', skew_num, skew_col)
    ]
)

X = preprocess.fit_transform(x_test)
xtest = pd.DataFrame(X, columns=num_col +
                     skew_col).join(x_test.drop(num_col+skew_col, axis=1))


x_re = xtest[['city_development_index', 'relevent_experience', 'enrolled_university', 'ed
              'experience', 'company_size', 'last_new_job', 'training_hours',
              'Female', 'Male', 'Other']]
```

In [12]:

```
# Undersampling
undersampler = RandomUnderSampler(random_state=42)
x, y = RandomUnderSampler(random_state=42).fit_resample(x_re, y_re)


x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size=0.2, random_state=432)
```

In [17]:

```
model = [
    # Ensemble Methods
    ensemble.AdaBoostClassifier(),
    ensemble.BaggingClassifier(),
    ensemble.ExtraTreesClassifier(),
    ensemble.GradientBoostingClassifier(),
    ensemble.RandomForestClassifier(),

    # GLM
    linear_model.SGDClassifier(),

    # Navies Bayes
    naive_bayes.BernoulliNB(),
    naive_bayes.GaussianNB(),

    # Nearest Neighbor
    neighbors.KNeighborsClassifier(),

    # SVM
    svm.SVC(probability=True),
    svm.LinearSVC(),

    # Trees
    tree.DecisionTreeClassifier(),
    tree.ExtraTreeClassifier(),

    # Discriminant Analysis
    discriminant_analysis.LinearDiscriminantAnalysis(),
    discriminant_analysis.QuadraticDiscriminantAnalysis(),

    # xgboost
    XGBClassifier()
]

model_result_org = pd.DataFrame(
    columns=['model', 'parameters', 'cv_accuracy', 'mae', 'mse', 'f1'])
row_index = 0

for ml in model:
    print(ml.__class__.__name__)
    ml.fit(x_train, y_train)
    y_pred = ml.predict(x_test)
    model_result_org.loc[row_index, 'model'] = ml.__class__.__name__
    model_result_org.loc[row_index, 'parameters'] = str(ml.get_params())
    model_result_org.loc[row_index, 'mae'] = mean_absolute_error(y_test, y_pred)
    model_result_org.loc[row_index, 'mse'] = mean_squared_error(y_test, y_pred)
    model_result_org.loc[row_index, 'f1'] = f1_score(y_test, y_pred)
    model_result_org.loc[row_index, 'cv_accuracy'] = cross_val_score(
        ml, x_train, y_train, cv=10, scoring='accuracy').mean()
    row_index += 1

model_result_org
# 第一次試做 : f1_score最好為 GaussianNB  之 0.513875
# 第二次試做 : f1_score最好為 GradientBoosting 為 0.751387  (部分資料標準化 + 欠採樣 + Quant
# 第三次試做 : f1_score最好為 GradientBoosting  (加入cross_val做確認)
```

2	ExtraTreesClassifier	{'bootstrap': False, 'ccp_alpha': 0.0, 'class_...	0.686378	0.308216	0.308216	0.700559
3	GradientBoostingClassifier	{'ccp_alpha': 0.0, 'criterion': 'friedman_mse'...	0.746044	0.25798	0.25798	0.751387
4	RandomForestClassifier	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...	0.718697	0.271586	0.271586	0.73371
5	SGDClassifier	{'alpha': 0.0001, 'average': False, 'class_wei...	0.65537	0.374673	0.374673	0.676018
6	BernoulliNB	{'alpha': 1.0, 'binarize': 0.0, 'class_prior':...	0.619126	0.379906	0.379906	0.630346
7	GaussianNB	{'priors': None, 'var_smoothing': 1e-09}	0.652624	0.349555	0.349555	0.65567

In [18]:

```python
# Hyper-Parameters

model = ensemble.GradientBoostingClassifier()
model.fit(x_train, y_train)

result_cv = cross_val_score(model, x_train, y_train,
                            cv=10, scoring='accuracy').mean()

param = {'n_estimators': [40, 50, 60], 'learning_rate':[0.5, 0.1, 0.05], 'max_features':
new_model = GridSearchCV(model, param, cv=10, scoring='accuracy')
new_model.fit(x_train, y_train)

print(result_cv)
print(new_model.score(x_train, y_train))
print(new_model.best_params_)
```

```
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
```

In [19]:

```
# feature selection

model = ensemble.GradientBoostingClassifier()
model_rfecv = RFECV(estimator=model, step=1, cv=10)
model_rfecv.fit(x_train, y_train)
column_rfe = x_train.columns.values[model_rfecv.get_support()]

result_cv = cross_val_score(model, x_train[column_rfe], y_train,
                            cv=10, scoring='accuracy').mean()

param = {'n_estimators': [40, 50, 60], 'learning_rate': [
    0.5, 0.1, 0.05], 'max_features': ['sqrt', 'log2', 8, None], 'max_depth': [3, 6, 9]}
new_model = GridSearchCV(model, param, cv=10, scoring='accuracy')
new_model.fit(x_train[column_rfe], y_train)

print(result_cv)
print(column_rfe)
print(new_model.score(x_train[column_rfe], y_train))
print(new_model.best_params_)
```

```
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494:
```

['city_development_index' 'relevent_experience' 'enrolled_university' 'education_level' 'experience' 'company_size' 'last_new_job' 'training_hours' 'Male'] {'learning_rate': 0.1, 'max_depth': 3, 'max_features': None, 'n_estimators': 50}

In [22]:

```
test = pd.read_csv('aug_test.csv')

test = test.replace(clean_nums)
test['city'] = label.fit_transform(test['city'])
test = test.drop(['major_discipline', 'company_type'], axis=1)
test['gender'] = test['gender'].fillna('Other')
test[['last_new_job', 'experience', 'education_level']] = test[[
    'last_new_job', 'experience', 'education_level']].fillna(0)
test['enrolled_university'] = test['enrolled_university'].fillna(1)
test['company_size'] = test['company_size'].fillna(test.groupby(
    'education_level')['company_size'].transform('mean').round())
test[['Female', 'Male', 'Other']] = onehot.fit_transform(
    test[['gender']]).toarray()
test['training_hours_cut'] = pd.qcut(
    test['training_hours'], 5, labels=[1, 2, 3, 4, 5])
test['experience_cut'] = pd.qcut(
    test['experience'].astype(int), 5, labels=[1, 2, 3, 4, 5])

for col in test.columns:
    if col not in ['gender', 'city_development_index']:
        test[col] = test[col].astype('int64')

X = preprocess.fit_transform(test)

xtest = pd.DataFrame(X, columns=num_col +
                     skew_col).join(test.drop(num_col+skew_col, axis=1))
column_new = ['city_development_index', 'relevent_experience', 'enrolled_university',
              'education_level', 'experience', 'company_size', 'last_new_job', 'training_
x_re = xtest[column_new]

model = ensemble.GradientBoostingClassifier(
    n_estimators=50, learning_rate=0.1, max_depth=3)
model.fit(x_train[column_new], y_train)
test["target"] = model.predict(x_re)
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2129 entries, 0 to 2128
Data columns (total 17 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   enrollee_id             2129 non-null   int64
 1   city                    2129 non-null   int64
 2   city_development_index  2129 non-null   float64
 3   gender                  2129 non-null   object
 4   relevent_experience     2129 non-null   int64
 5   enrolled_university     2129 non-null   int64
 6   education_level         2129 non-null   int64
 7   experience              2129 non-null   int64
 8   company_size            2129 non-null   int64
 9   last_new_job            2129 non-null   int64
 10  training_hours          2129 non-null   int64
 11  Female                  2129 non-null   int64
 12  Male                    2129 non-null   int64
 13  Other                   2129 non-null   int64
 14  training_hours_cut      2129 non-null   int64
 15  experience_cut          2129 non-null   int64
 16  target                  2129 non-null   int64
dtypes: float64(1), int64(15), object(1)
memory usage: 282.9+ KB

C:\Users\jerry\anaconda3\lib\site-packages\sklearn\ensemble\_gb.py:494: Da
taConversionWarning: A column-vector y was passed when a 1d array was expe
cted. Please change the shape of y to (n_samples, ), for example using rav
el().
  y = column_or_1d(y, warn=True)
```