Classification

Data import

```
1 import pandas as pd
2 import os
3 import numpy as np
```

```
1 traindata_url = 'https://bitbucket.org/hyuk125/lg_dic/raw/889649d1bc273bf53967
2 testdata_url = 'https://bitbucket.org/hyuk125/lg_dic/raw/889649d1bc273bf53967c
3 train_data = pd.read_csv(traindata_url)
4 test_data = pd.read_csv(testdata_url)
```

```
1 X_train = train_data.values[:, 1:]
2 y_train = train_data.values[:, 0]
3 X_test = test_data.values[:, 1:]
4 y_test = test_data.values[:, 0]
```

```
1 X = np.r_[X_train, X_test]
2 y = np.r_[y_train, y_test]
```

Classification 모델

Data 처리

Training / test 분할

```
1 from sklearn.model_selection import train_test_split
2 X_train, X_test, y_train, y_test = train_test_split(X, y)
```

Randomforest classification 모델 학습

```
1 from sklearn.ensemble import RandomForestClassifier
```

```
1 model = RandomForestClassifier(n_estimators=50, max_leaf_nodes=16, n_jobs=-1)
2 model.fit(X_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
```

```
max_leaf_nodes=16, max_samples=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=50, n_jobs=-1,
oob_score=False, random_state=None, verbose=0,
warm_start=False)
```

Model Test

정확도 판단

Confution matrix

```
1 from sklearn.metrics import confusion_matrix
```

```
1 predict = model.predict(X_test)
```

```
1 matrix = confusion_matrix(y_test, predict)
2 matrix
```

```
array([[1691,    0,    3,    5,    0,    3,   15,    6,   26,    1],
       [   1, 1918,    9,   12,    0,    3,    3,   10,    7,    1],
       [  41,   75, 1401,   36,   18,    4,   65,   55,   45,   17],
       [  42,   52,   44, 1415,   11,   18,    6,   52,   48,   84],
       [  11,    7,   16,    4, 1395,    4,   65,   33,   21,  195],
       [  88,   68,    7,  446,   41,  613,   48,   49,   48,  141],
       [  66,   35,   27,   15,   27,   10, 1470,   13,   18,    1],
       [  14,   42,   24,    3,   29,    0,    1, 1652,   16,   95],
       [  14,  152,   36,  134,   28,    4,   26,   11, 1212,   84],
       [  18,   22,   16,   25,   61,    1,    6,  142,   27, 1380]])
```

Precision, recall

```
1 from sklearn.metrics import precision_score, recall_score
```

```
1 print(precision_score(y_test, predict, average=None))
2 print('average: ', precision_score(y_test, predict, average='weighted'))
```

```
[0.85146022 0.80894137 0.88502843 0.67541766 0.86645963 0.92878788
 0.86217009 0.816609   0.82561308 0.69034517]
average:  0.8197266878585333
```

```
1 print(recall_score(y_test, predict, average=None))
2 print('average: ', recall_score(y_test, predict, average='weighted'))
```

```
[0.96628571 0.97657841 0.7973819  0.79853273 0.79668761 0.39573919
 0.87395957 0.88059701 0.71252205 0.81272085]
average:  0.8084
```

```
1
```