lab6-wp871q

May 20, 2024

#

Kiss Dániel Márk

##

WP871Q

1 Library import

[1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

2 Loading train set

[2]:
```
df_verseny_public_train = pd.read_csv('data/verseny_public_train.csv', sep=',',␣
↪low_memory=False)
```

3 Remove missing values

[3]:
```
df_verseny_public_train = df_verseny_public_train.dropna()
```

4 Selecting columns with the highest variance in the training set

[4]:
```
df_verseny_public_train.var().sort_values(ascending=False)
```

[4]:
```
cookie_id      8.333417e+08
Topic63_ec     6.155122e+03
Topic52_ec     4.236199e+03
Topic42_ec     3.855324e+03
Topic33_ec     3.570435e+03
                   …
Topic173_ic    0.000000e+00
```

```
Topic171_ec    0.000000e+00
Topic171_ic    0.000000e+00
Topic170_ec    0.000000e+00
Topic170_ic    0.000000e+00
Length: 258, dtype: float64
```

[5]:
```
y = df_verseny_public_train['target']

var = df_verseny_public_train[df_verseny_public_train.var().
 ↪sort_values(ascending=False).index[:100]]
```

[6]:
```
X = var.drop(['cookie_id'], axis=1)
```

[7]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X = scaler.fit_transform(X)
```

[8]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

5 Decision tree

[9]:
```
dtcl = DecisionTreeClassifier(random_state=42, max_depth=15,␣
 ↪min_samples_split=15, min_samples_leaf=5, max_features=0.6,␣
 ↪class_weight='balanced', criterion='gini')

dtcl.fit(X_train, y_train)

y_pred = dtcl.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.87815
```

6 Random forest

[10]:
```
from sklearn.ensemble import RandomForestClassifier

rfcl = RandomForestClassifier(random_state=42, n_estimators=100, max_depth=15,␣
 ↪min_samples_split=15, min_samples_leaf=5, max_features=0.6,␣
 ↪class_weight='balanced', criterion='gini')

rfcl.fit(X_train, y_train)
```

```
y_pred = rfcl.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.9634
```

7 Gradient boosting classifier

[11]:
```
from sklearn.ensemble import GradientBoostingClassifier

gbcl = GradientBoostingClassifier(random_state=42, n_estimators=100,␣
 ↪max_depth=15, min_samples_split=15, min_samples_leaf=5, max_features=0.6,␣
 ↪criterion='friedman_mse')

gbcl.fit(X_train, y_train)

y_pred = gbcl.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.9848
```

8 XGBoost

[14]:
```
from xgboost import XGBClassifier

xgbcl = XGBClassifier(random_state=42, n_estimators=100, max_depth=15,␣
 ↪min_samples_split=15, min_samples_leaf=5, max_features=0.6,␣
 ↪class_weight='balanced', criterion='gini')

xgbcl.fit(X_train, y_train)

y_pred = xgbcl.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
/Users/kissdanielmark/Documents/01.Iskola/MSc/3/Customer
Analytics/Competition/CustomerAnalytics_Competition/.venv/lib/python3.9/site-
packages/xgboost/core.py:160: UserWarning: [14:16:44] WARNING:
/Users/runner/work/xgboost/xgboost/src/learner.cc:742:
Parameters: { "class_weight", "criterion", "max_features", "min_samples_leaf",
"min_samples_split" } are not used.

  warnings.warn(smsg, UserWarning)
```

```
Accuracy:  0.985
```

9 Ensembling Gradient Boosting and Decision Tree

[15]:
```
from sklearn.ensemble import VotingClassifier

vc = VotingClassifier(estimators=[('gb', gbcl), ('dtcl', dtcl)], voting='soft')

vc.fit(X_train, y_train)

y_pred = vc.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.98365
```

10 Ensembling Random forest and XGBoost

[16]:
```
vc = VotingClassifier(estimators=[('xgbcl', xgbcl), ('rfcl', rfcl)],␣
↪voting='soft')

vc.fit(X_train, y_train)

y_pred = vc.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
/Users/kissdanielmark/Documents/01.Iskola/MSc/3/Customer
Analytics/Competition/CustomerAnalytics_Competition/.venv/lib/python3.9/site-
packages/xgboost/core.py:160: UserWarning: [14:19:34] WARNING:
/Users/runner/work/xgboost/xgboost/src/learner.cc:742:
Parameters: { "class_weight", "criterion", "max_features", "min_samples_leaf",
"min_samples_split" } are not used.

  warnings.warn(smsg, UserWarning)
Accuracy:  0.9847
```

11 Ensembling all of them

[17]:
```
vc = VotingClassifier(estimators=[('xgbcl', xgbcl), ('rfcl', rfcl), ('gb',␣
↪gbcl), ('dtcl', dtcl)], voting='soft')

vc.fit(X_train, y_train)

y_pred = vc.predict(X_test)

print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
/Users/kissdanielmark/Documents/01.Iskola/MSc/3/Customer
Analytics/Competition/CustomerAnalytics_Competition/.venv/lib/python3.9/site-
packages/xgboost/core.py:160: UserWarning: [14:20:39] WARNING:
/Users/runner/work/xgboost/xgboost/src/learner.cc:742:
Parameters: { "class_weight", "criterion", "max_features", "min_samples_leaf",
"min_samples_split" } are not used.

  warnings.warn(smsg, UserWarning)
```

```
Accuracy:  0.9845
```

12 Loading test set

[20]:
```
df_verseny_public_test = pd.read_csv('data/verseny_public_test.csv', sep=',',␣
↪low_memory=False)
```

[21]:
```
X_test = df_verseny_public_test[var.columns].drop(['cookie_id'], axis=1)

X_test = scaler.transform(X_test)

y_pred = vc.predict_proba(X_test)[:, 1]

df_verseny_public_test['target'] = y_pred

df_verseny_public_test[['cookie_id', 'target']].to_csv('data/lab6.csv',␣
↪index=False)
```

13 Public score: 0,7343