January 2024

Academic Drop Out Analysis - Classification Model

Diagnostic and Predictive Analysis of Student Dropouts and Academic Success

Python, Microsoft Excel, Tableau

Classification Model:

- 2 classification models created
  - 2 datasets used for models:
    1. Model 1 - All variables included
    2. Model 2 -All variables excluding 1st semester and 2nd semester unit variables so it can be used on incoming students

- One Hot Encoding for non numerical variables
  Explanation:
  https://www.geeksforgeeks.org/ml-one-hot-encoding-of-datasets-in-python/

from Jupyter notebook

In [ ]:

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn import metrics

# Read Dataset
acaddropout = pd.read_csv('acaddropout.csv')

# Split dataset into features and labels
acaddropout1 = acaddropout.drop('Target', axis =1)
labels = acaddropout.Target

# Create dataset without units
acaddropout2 = acaddropout.drop(['Target','Curricular units 1st sem credited','Curricular units 1st sem enrolled'
                                 'Curricular units 1st sem evaluations','Curricular units 1st sem approved',
                                 'Curricular units 1st sem grade','Curricular units 1st sem without evaluations',
                                 'Curricular units 2nd sem credited','Curricular units 2nd sem enrolled',
                                 'Curricular units 2nd sem evaluations','Curricular units 2nd sem approved',
                                 'Curricular units 2nd sem grade','Curricular units 2nd sem without evaluations']


# One Hot Encoding for string variables

features = pd.get_dummies(acaddropout1, columns=["Marital status", "Application Mode", "Application order", "Cour
                                                 "attendance_time", "Previous Qualification", "Nationality", "
                                                 "Mother Occupation","Father Occupation", "Gender"])

features2 = pd.get_dummies(acaddropout2, columns=["Marital status", "Application Mode", "Application order", "Cou
                                                  "attendance_time", "Previous Qualification", "Nationality", "
                                                  "Mother Occupation","Father Occupation", "Gender"])

print(features.head())
print(features2.head())
```

Output:

Classification Model 1, Classification Model 2

```
   Previous qualification grade  Admission grade  ...  Gender_Female  Gender_Male
0                           122.0            127.3  ...              0            1
1                           160.0            142.5  ...              0            1
2                           122.0            124.8  ...              0            1
3                           122.0            119.6  ...              1            0
4                           100.0            141.5  ...              1            0

[5 rows x 153 columns]
   Previous qualification grade  Admission grade  ...  Gender_Female  Gender_Male
0                           122.0            127.3  ...              0            1
1                           160.0            142.5  ...              0            1
2                           122.0            124.8  ...              0            1
3                           122.0            119.6  ...              1            0
4                           100.0            141.5  ...              1            0

[5 rows x 141 columns]
```

Input:

5 fold cross validation for each model

In [ ]:

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.30, random_state = 42)
X_train2, X_test2, y_train2, y_test2 = train_test_split(features2, labels, test_size=0.30, random_state = 42)


# Using Grid Search to find the best parameters
param_grid1 = {
    'n_estimators': [200],
    'max_features': [153],
    'max_depth' : [9,11],
    'criterion' :['gini'],
    'min_samples_split':[2]
    }

param_grid2 = {
    'n_estimators': [100,150,200],
    'max_features': [141],
    'max_depth' : [9,11,13,15],
    'criterion' :['gini'],
    'min_samples_split':[2,3]
    }

# Training RF Models with K-Fold of 5
rf_models = GridSearchCV(RandomForestClassifier(random_state = 42), param_grid=param_grid1, cv=5, verbose=1)
rf_models.fit(X_train, y_train)
print(rf_models.best_params_)

# Training RF Models with K-Fold of 5
rf_models2 = GridSearchCV(RandomForestClassifier(random_state = 42), param_grid=param_grid2, cv=5, verbose=1)
rf_models2.fit(X_train2, y_train2)
print(rf_models2.best_params_)
```

Output:

```
Fitting 5 folds for each of 2 candidates, totalling 10 fits
{'criterion': 'gini', 'max_depth': 9, 'max_features': 153, 'min_samples_split': 2, 'n_estimators': 200}
Fitting 5 folds for each of 24 candidates, totalling 120 fits
{'criterion': 'gini', 'max_depth': 13, 'max_features': 141, 'min_samples_split': 2, 'n_estimators': 100}
```

Input:

Print Accuracy of Model and Variable Importance (top 10)

In [ ]:

```
predictions = rf_models.predict(X_test)
predictions2 = rf_models2.predict(X_test2)

# Print the Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(predictions, y_test))

# Print Feature Importance
feature_importance = pd.DataFrame(data={"features":X_test.columns, "importance":rf_models.best_estimator_.feature
print("Feature Importance")
print(feature_importance.sort_values('importance', ascending=False).head(10))

# Print the 2nd Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(predictions2, y_test2))

# Print 2nd Model Feature Importance
feature_importance2 = pd.DataFrame(data={"features":X_test2.columns, "importance":rf_models2.best_estimator_.feat
print("Feature Importance")
print(feature_importance2.sort_values('importance', ascending=False).head(10))
```

Output:

```
Accuracy: 0.7643072289156626
Feature Importance
                                features  importance
18     Curricular units 2nd sem approved   47.021838
5                Tuition fees up to date    5.381794
19         Curricular units 2nd sem grade    3.576575
11  Curricular units 1st sem evaluations    3.161188
1                        Admission grade    2.810703
0            Previous qualification grade    2.766756
12     Curricular units 1st sem approved    2.445071
7                      Age at enrollment    2.434991
16     Curricular units 2nd sem enrolled    2.350735
17  Curricular units 2nd sem evaluations    2.331269
Accuracy: 0.641566265060241
Feature Importance
                         features  importance
5         Tuition fees up to date   15.364829
1                 Admission grade    9.007001
7               Age at enrollment    7.216262
0    Previous qualification grade    6.636007
6              Scholarship holder    6.560176
11                            GDP    4.448183
9               Unemployment rate    3.366316
46                    Course_9500    2.585017
10                Inflation rate    2.410330
41                    Course_9119    2.042362
```

Model 1:

76.5% accuracy

Features	**Importance**
Curricular units 2nd sem approved	47.021838
Tuition fees up to date	5.381794
Curricular units 2nd sem grade	3.576575
Curricular units 1st sem evaluations	3.161188
Admission grade	2.810703
Previous qualification grade	2.766756
Curricular units 1st sem approved	2.445071
Age at enrollment	2.434991
Curricular units 2nd sem enrolled	2.350735
Curricular units 2nd sem evaluations	2.331269

Model 2:

64.2% accuracy

Features	**Importance**
Tuition fees up to date	15.364829
Admission grade	9.007001
Age at enrollment	7.216262
Previous qualification grade	6.636007
Scholarship holder	6.560176
GDP	4.448183
Unemployment rate	3.366316
Course_9500	2.585017
Inflation rate	2.410330
Course_9119	2.042362

Model 2 saved to predict dropout rate for incoming students