SVM

November 8, 2021

- SVM can handle multiple continuous, and categorical variables
- Will try different kernels: linear, poly, RBF, and Sigmoid

[ ]:
```
import numpy as np
import pandas as pd

from sklearn import preprocessing
from imblearn.combine import SMOTETomek

from sklearn import svm
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,␣
↪recall_score, roc_curve, roc_auc_score
from sklearn import preprocessing

from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RepeatedKFold
from sklearn.model_selection import cross_val_score
from numpy import mean
from numpy import std
```

[ ]:
```
#Import Drive API and authenticate
from google.colab import drive
#Mount Drive to the Colab VM
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:
```
#Load the dataset into pandas DataFrame
df = pd.read_csv("/content/drive/MyDrive/Capstone_project/v2_credit_default.
↪csv")
```

[ ]:
```
#Seperate the independent and dependent variables.
df_independent = df.drop(['Default'], axis=1)
```

```
df_default = df['Default']
```

[ ]:
```
# split the data into 80% training+validation and 20% test
X_train, X_test, y_train, y_test = train_test_split(df_independent, df_default,␣
↪test_size=0.20, random_state=1)
```

[ ]:
```
# Scale input variables for training+validation (X_train)
X_train_scaled = preprocessing.MinMaxScaler().fit_transform(X_train)
```

[ ]:
```
# Balancing using SMOTE Tomek
X_smt, y_smt = SMOTETomek(random_state=1).fit_sample(X_train_scaled, y_train.
↪squeeze())
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87:
FutureWarning: Function safe_indexing is deprecated; safe_indexing is deprecated
in version 0.22 and will be removed in version 0.24.
  warnings.warn(msg, category=FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87:
FutureWarning: Function safe_indexing is deprecated; safe_indexing is deprecated
in version 0.22 and will be removed in version 0.24.
  warnings.warn(msg, category=FutureWarning)
```

[ ]:
```
# #hyperparameter adjustment with GridSearchCV
# # Testing different kernels. Can add additional hyperparameters to the␣
↪param_grid dictionary for optimization too.
# svm_model = svm.SVC(random_state=1, degree=8)

# kernel_options = ['linear', 'poly', 'rbf', 'sigmoid']

# param_grid = dict(kernel = kernel_options)
# grid = GridSearchCV(svm_model, param_grid, cv=10, scoring = 'accuracy')
# grid.fit(X_smt,y_smt)

# print (grid.best_params_)
# print (grid.best_score_)
```

[ ]:
```
#Finally test with the test set (X_test):
# Fit the model with linear kernel
svm_model = svm.SVC(kernel ='linear',random_state=1)
svm_model.fit(X_smt, y_smt)
# Predict using the scaled X_test
X_test_scaled = preprocessing.MinMaxScaler().fit_transform(X_test)
y_pred = svm_model.predict(X_test_scaled)
# performance metrics
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy', accuracy_score(y_test, y_pred))
```

```
print('precision', precision_score(y_test, y_pred))
print('recall', recall_score(y_test, y_pred))
```

```
[[4412  253]
 [ 896  432]]
accuracy 0.8082763223761055
precision 0.6306569343065693
recall 0.3253012048192771
```

[ ]:
```
# Fit the model with poly kernel
svm_model = svm.SVC(kernel ='poly',random_state=1, degree=8)
svm_model.fit(X_smt, y_smt)
# Predict using the scaled X_test
X_test_scaled = preprocessing.MinMaxScaler().fit_transform(X_test)
y_pred = svm_model.predict(X_test_scaled)
# performance metrics
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy', accuracy_score(y_test, y_pred))
print('precision', precision_score(y_test, y_pred))
print('recall', recall_score(y_test, y_pred))
```

```
[[4154  511]
 [1027  301]]
accuracy 0.7433672618054397
precision 0.3706896551724138
recall 0.22665662650602408
```

[ ]:
```
# Fit the model with sigmoid kernel
svm_model = svm.SVC(kernel ='sigmoid',random_state=1)
svm_model.fit(X_smt, y_smt)
# Predict using the scaled X_test
X_test_scaled = preprocessing.MinMaxScaler().fit_transform(X_test)
y_pred = svm_model.predict(X_test_scaled)
# performance metrics
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy', accuracy_score(y_test, y_pred))
print('precision', precision_score(y_test, y_pred))
print('recall', recall_score(y_test, y_pred))
```

```
[[2841 1824]
 [ 736  592]]
accuracy 0.5728349741364925
precision 0.24503311258278146
recall 0.4457831325301205
```

[ ]:
```
# Fit the model with Radial Basis Function kernel
svm_model = svm.SVC(kernel ='rbf',random_state=1)   #tried gamma=0.1 instead of␣
 ↪the default gamma, with similar results
svm_model.fit(X_smt, y_smt)
# Predict using the scaled X_test
X_test_scaled = preprocessing.MinMaxScaler().fit_transform(X_test)
y_pred = svm_model.predict(X_test_scaled)
# performance metrics
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy', accuracy_score(y_test, y_pred))
print('precision', precision_score(y_test, y_pred))
print('recall', recall_score(y_test, y_pred))
```

```
[[4042  623]
 [ 637  691]]
accuracy 0.7897547138328049
precision 0.525875190258752
recall 0.5203313253012049
```