گزارش تمرین

عرفان رفیعی اسکویی - 98243027

1) داده های از دست رفته

در این پروژه برای اصلاح داده‌های از دست رفته ابتدا تمامی سطرهایی که همه ویژگی‌هایش از دست رفته اند پاک میکنیم. سپس ریفت‌هایی که داده های از دست رفته دارند را با روش bfill پر میکنیم. با این کار تمام نمونه‌ها دیگر مشکل ندارند.

```
################ YOUR CODE STARTS HERE ################
train_df = train_df.dropna(how = 'all')
train_df = train_df.fillna(method ='bfill')

test_df = test_df.dropna(how = 'all')
test_df = test_df.fillna(method ='bfill')

print("Training set:")
print(train_df.isna().sum())
print("============================")
print("Test set:")
print(test_df.isna().sum())
################ YOUR CODE ENDS HERE ##################
```

```
Training set:
CustomerID                0
Age                       0
Gender                    0
City                      0
State                     0
No_of_orders_placed       0
Sign_up_date              0
Last_order_placed_date    0
is_premium_member         0
Women's_Clothing          0
Men's_Clothing            0
Kid's_Clothing            0
Home_&_Living             0
Beauty                    0
Electronics               0
Preferred_Theme           0
dtype: int64
==========================
Test set:
CustomerID                0
Age                       0
Gender                    0
City                      0
```

2) استانداردسازی:

در این مرحله داده‌های ستون‌های عددی با فرایند StandardScaler نرمال می‌شوند. کد این مرحله:

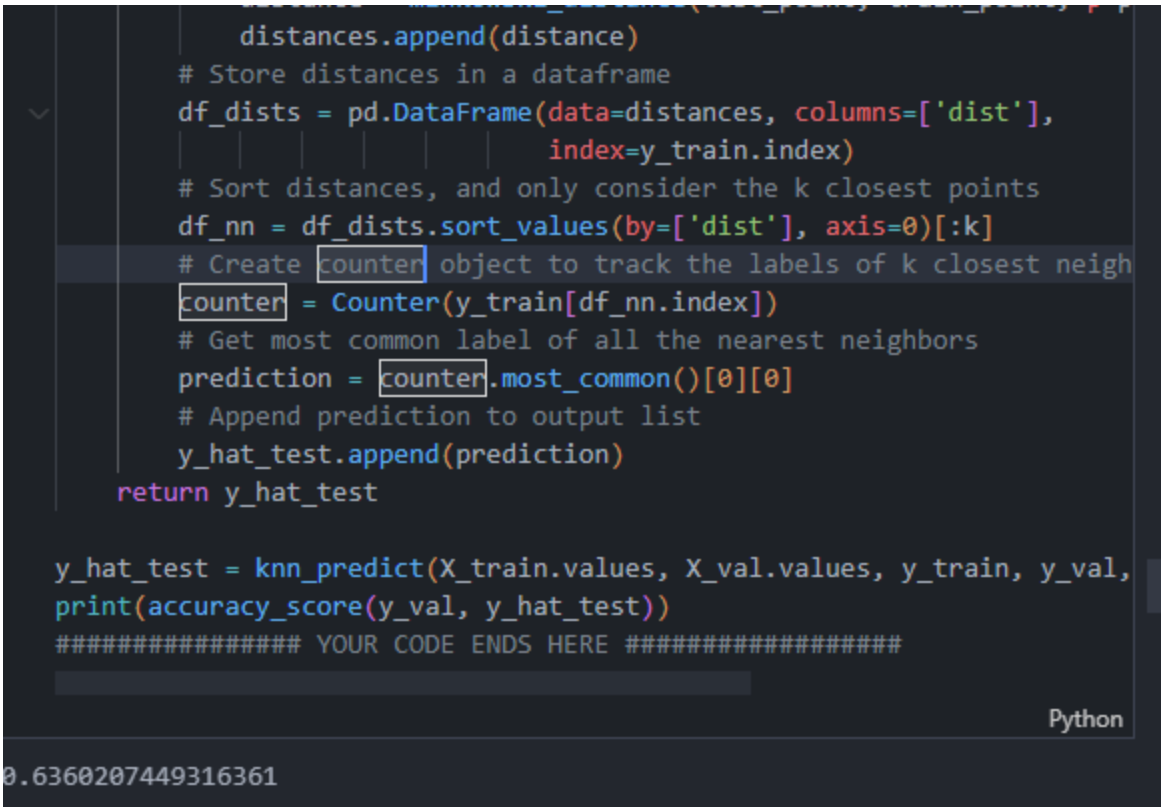
```python
# Standardize your train/test data.
# Important note: You must use the mean and standard deviation of trai
from sklearn.preprocessing import StandardScaler
################ YOUR CODE STARTS HERE ################
numeric_cols = ['No_of_orders_placed','Women's_Clothing','Men's_Clothi
scaler = StandardScaler()
scaler.fit(train_df[numeric_cols].values)
train_df_scaled_numeric = scaler.transform(train_df[numeric_cols])
test_df_scaled_numeric = scaler.transform(test_df[numeric_cols])
train_df_scaled_numeric = pd.DataFrame(train_df_scaled_numeric,columns
test_df_scaled_numeric = pd.DataFrame(test_df_scaled_numeric,columns=n
################ YOUR CODE ENDS HERE ##################
```

حال نوبت به ستون‌های غیرعددی و Categorical است که در این روش داده‌ها بصورت onehotencoder نرمال می‌شوند.

```
# Now you have to convert the categorical features in the
# dataset to one-hot encoded representation.
# Hint: Use pd.DataFrame.get_dummies()

# before that, we drop 'City' column from both training and test set,
train_df.drop('City', axis=1, inplace=True)
test_df.drop('City', axis=1, inplace=True)

cols_to_be_encoded = ['Gender', 'State']

################ YOUR CODE STARTS HERE ################

train_df_encoded = pd.get_dummies(train_df[cols_to_be_encoded], prefix
test_df_encoded = pd.get_dummies(test_df[cols_to_be_encoded], prefix=[

################ YOUR CODE ENDS HERE ##################
```

3) الگوریتم KNN

الگوریتم KNN با روش فاصله Minkowski نوشته شده است که برای هر داده نزدیک ترین همسایه آن معرفی می‌شود. با این الگوریتم مقددار دقت برای داده validation برابر با 63 درصد است.



4) حال با روش‌های دیگر مقدار دقت را بر روی داده‌های مشخص شده بدست می‌آوریم.

روش اول Decision Tree، روش دوم SVC و روش سوم RandomForest است.

```python
################ YOUR CODE STARTS HERE ################
from sklearn import tree
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier

#DecisionTreeClassifier
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train.values, y_train.values)
y_hat_test_DT = clf.predict(X_test.values)
print('DecisionTreeClassifier: ',accuracy_score(y_test, y_hat_test_DT))

#SVC
clf = svm.SVC()
clf = clf.fit(X_train.values, y_train.values)
y_hat_test_SVC = clf.predict(X_test.values)
print('SVC: ',accuracy_score(y_test, y_hat_test_SVC))

#RandomForestClassifier
clf = RandomForestClassifier(n_estimators=10)
clf = clf.fit(X_train.values, y_train.values)
y_hat_test_RF = clf.predict(X_test.values)
print('RandomForestClassifier: ',accuracy_score(y_test, y_hat_test_RF))
################ YOUR CODE ENDS HERE ##################
```

Python

```
DecisionTreeClassifier:  0.5991199119911991
SVC:  0.6941694169416942
RandomForestClassifier:  0.6664466446644665
```

5) ارزیابی

در این مرحله برای داده‌های تست مقدار Accuracy، Precision، Recall و F1-Score را بدست می‌آوریم.

```
################ YOUR CODE STARTS HERE ################
from sklearn.metrics import precision_recall_fscore_support

#DecisionTreeClassifier
precision,recall,fscore,support = precision_recall_fscore_support(y_te
print('precision for each label (DecisionTreeClassifier): ',precision)
print('recall for each label (DecisionTreeClassifier): ',recall)
print('fscore for each label (DecisionTreeClassifier): ',fscore)
print('support for each label (DecisionTreeClassifier): ',support)
print('---------------------------------')

#SVC
precision,recall,fscore,support = precision_recall_fscore_support(y_te
print('precision for each label (SVC): ',precision)
print('recall for each label (SVC): ',recall)
print('fscore for each label (SVC): ',fscore)
print('support for each label (SVC): ',support)
print('---------------------------------')

#RandomForestClassifier
precision,recall,fscore,support = precision_recall_fscore_support(y_te
print('precision for each label (RandomForestClassifier): ',precision)
print('recall for each label (RandomForestClassifier): ',recall)
print('fscore for each label (RandomForestClassifier): ',fscore)
print('support for each label (RandomForestClassifier): ',support)
################ YOUR CODE ENDS HERE ##################
```

```
precision for each label (DecisionTreeClassifier):  [0.92450331
0.91622807]
recall for each label (DecisionTreeClassifier):  [0.91641138 0.92433628]
fscore for each label (DecisionTreeClassifier):  [0.92043956 0.92026432]
support for each label (DecisionTreeClassifier):  [2285 2260]
--------------------------------
precision for each label (SVC):  [0.72904762 0.69161554]
recall for each label (SVC):  [0.67002188 0.74823009]
fscore for each label (SVC):  [0.69828962 0.71880978]
support for each label (SVC):  [2285 2260]
--------------------------------
precision for each label (RandomForestClassifier):  [0.90752417
0.94182825]
recall for each label (RandomForestClassifier):  [0.94485777 0.90265487]
fscore for each label (RandomForestClassifier):  [0.92581475 0.92182558]
support for each label (RandomForestClassifier):  [2285 2260]
```