Regression(Randomforestregressor)

Feature engineering

We will also remove Product_Category_2 and _3 since they have a high rate of null values and other categorical features have a few number of unique values so we will encode them using OneHotEncoder.

```
#Regression Model
train = bf.drop(['Product_Category_2', 'Product_Category_3','Product_ID','User_ID'], axis=1)
y = train.pop('Purchase')
#print(train.columns)
#print(train.head(10))
train.loc[:, 'Gender'] = np.where(train['Gender'] == 'M', 1, 0)

#print(train.columns)
#print(train.head(10))
# One hot encoding other features
featruecoder = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years','Product_Category_1' ]
encoder = OneHotEncoder().fit(train[featruecoder])

#print(train.columns)
#print(train.head(2))
train = pd.concat([train, pd.DataFrame(encoder.transform(train[featruecoder]).toarray(), index=train.index, columns=encoder.get_feature_names(featruecoder))], axis=1)

#print(train.columns)
#print(train.head(2))
train.drop(featruecoder, axis=1, inplace=True)
```

Then split out train and test data and standardize the data using StandardScaler

```
# Splitting train and test setsAA
X_train, X_test, y_train, y_test = train_test_split(train, y )

# Standardizing
scaler = StandardScaler().fit(X_train)
X_train, X_test = scaler.transform(X_train), scaler.transform(X_test)
```

What we are trying to do here is to predict purchase amount from other features, namely Gender,Age,occupation,city etc. We'll choose number of trees (n_estimators) in our forest and max_depth for each tree by calculating scores for each combination and choosing the best one. Scoring metric we'll use is MSE.

```
params = {
    'n_estimators': [10, 30, 100, 300],
    'max_depth': [3, 5, 7]
}

grid_search = GridSearchCV(RandomForestRegressor(), param_grid=params, cv=3, scoring='neg_mean_squared_error', n_jobs=-1)
grid_search.fit(X_train, y_train)
preds = grid_search.predict(X_test)
print("Best params found: {}".format(grid_search.best_params_))
print("MSE score: {}".format(np.mean(preds-y_test)**2) )

model = grid_search.best_estimator_
joblib.dump(model, './model/RandomForestRegressor.pkl')
```

And the output are as followed:

```
Best params found: {'max_depth': 7, 'n_estimators': 300}
MSE score: 72.25324740602755
```

It seens that the model does not work well as we expected.

Then I try to test the testset I create myself.

```
def upload(Gender,Age,Occupation,City,CityStay,Marital_Status,Category):
    uploadset = pd.read_csv('testset.csv')#数据导入
    uploadset = uploadset.append([{'Gender':Gender,'Age':Age,'Occupation':Occupation,'City':City,'CityStay':CityStay,'Marital_Status':Marital_Status,'Category':Category}], i
    uploadset.to_csv('testset.csv',index=False)

def testpredict():
    testset = pd.read_csv('testset.csv')#数据导入
    testset.loc[:, 'Gender'] = np.where(testset['Gender'] == 'M', 1, 0)
    categoricals = ['Gender', 'Age', 'Occupation', 'City', 'CityStay','Category' ]
    encoder = OneHotEncoder().fit(testset[categoricals])
    testset = pd.concat([testset, pd.DataFrame(encoder.transform(testset[categoricals]).toarray(), index=testset.index, columns=encoder.get_feature_names(categoricals))], ax
    #print(testset.tail(1))
    testset.drop(categoricals, axis=1, inplace=True)
    model=joblib.load('./model/RandomForestRegressor.pkl')
    print(model.predict(testset.head(38)))
    return model.predict(testset.head(38))
```

```
127.0.0.1 - - [27/May/2019 10:33:47] "POST /testmodel HTTP/1.1" 200 -
[12969.27418771  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555 13298.52347267
 12978.55557205  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555  6519.59364555  6519.59364555
  6519.59364555  6519.59364555]
```

It has a low accuracy and can't predict accurately(different records are predicted into the same labels)

1	Gender	Age	Occupatic	City	CityStay	Marital_S	Category
2	M	0-17	1	A	0	0	1
3	F	18-25	2	B	1	1	2
4	M	26-35	3	C	2	1	3
5	F	36-45	4	A	3	0	4
6	M	46-60	5	B	4+	1	5
7	F	51-55	6	C	1	0	6
8	M	55+	7	A	2	1	7
9	F	0-17	8	B	3	1	8
10	M	18-25	9	C	4+	1	9
11	F	26-35	10	A	1	1	10
12	M	26-35	11	B	2	1	11
13	F	26-35	12	C	3	1	12
14	M	26-35	13	A	4+	1	13
15	F	26-35	14	B	1	1	14
16	M	26-35	15	C	2	1	15
17	F	26-35	16	A	3	1	16
18	M	26-35	17	B	4+	1	17
19	F	26-35	18	C	1	1	18
20	M	26-35	19	A	2	1	2
21	F	26-35	20	B	3	1	1
22	M	26-35	0	A	1	1	1
23	M	26-35	15	A	4+	0	8
24	F	55+	14	B	1	0	14
25	F	55+	14	B	2	1	14
26	F	18-25	7	B	2	1	2
27	M	26-35	8	C	3	0	3
28	M	0-17	9	A	4+	1	4
29	F	18-25	10	B	1	1	5
30	M	26-35	11	A	4+	1	6
31	M	0-17	12	B	2	0	7
32	M	55+	13	C	3	0	8
33	F	51-55	14	A	1	0	9
34	M	55+	15	B	2	1	10
35	F	0-17	16	A	3	1	11