Classification

Student:hyx

III. Construction step

1. Integration of all the cases defined in the elaboration step.
2. Machine Learning
3. Visualization of the dataset
4. Program the application and make the main tests

After data cleaning and chansfer, I would like to use Randon Forest Classifier to classify the customer's marital status from the black data set.The features is all the other columns including "Purchase", and the label is "Martial Staus".

(1) First to check the processed data set by listing 3 rows.

```
train_df_with_dummies_noNaN.head(3)
```

(2) Create X and y array for classification, it's the training phase, create Marital_Status label:

```
y_classf = train_df_with_dummies_noNaN['Marital_Status'].values
y_classf.shape
```

(3) Create features:

```
X_classf = train_df_with_dummies_noNaN.drop(['Marital_Status'], axis=1).values
X_classf.shape
```

(4) Split test set and training set for classification:

```
X_train_classf, X_test_classf, y_train_classf, y_test_classf = train_test_split(X_classf, y_classf, test_size=0.2, random_state=1)
```

(5) Define the random forest classification model:

```
RF_classifier = RandomForestClassifier(n_estimators=100, max_depth= 300, random_state=0)
```

(6) **Use functions to train the model:**

```
RF_classifier.fit(X_train_classf, y_train_classf)
```

(7) examine this classifier on the validation set:

```
RF_classifier.score(X= X_test_classf, y= y_test_classf)
```

```
   ....
   ...: train_df_with_dummies_noNaN.head(3)
Out[3]:
    Unnamed: 0  ...  Stay_In_Current_City_Years_4+
0       161273  ...                              0
6       161279  ...                              0
17      161290  ...                              1

[3 rows x 3595 columns]

In [4]: # create X and y array for classification (training phase)
   ...:
   ...: # create label
   ...:
   ...: y_classf = train_df_with_dummies_noNaN['Marital_Status'].values
   ...: y_classf.shape
Out[4]: (114928,)
```

```
In [5]: # create features
   ...:
   ...: X_classf = train_df_with_dummies_noNaN.drop(['Marital_Status'], axis=1).values
   ...: X_classf.shape
Out[5]: (114928, 3594)

In [6]: # create train-validation split for classification
   ...:
   ...: X_train_classf, X_test_classf, y_train_classf, y_test_classf =
train_test_split(X_classf, y_classf, test_size=0.2, random_state=1)

In [7]: RF_classifier = RandomForestClassifier(n_estimators=100, max_depth= 300,
random_state=0)

In [8]: RF_classifier.fit(X_train_classf, y_train_classf)
Out[8]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=300, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
            oob_score=False, random_state=0, verbose=0, warm_start=False)

In [9]: RF_classifier.score(X= X_test_classf, y= y_test_classf)
Out[9]: 0.812451057165231

In [10]:
```