Capstone Project

Akansha Kumar

6/18/2016

Contents

Definition/Introduction/Motivation

User clickstream data and information about a group of hotels is given. The users are classified into a set of classes/segments. The segments are,

- Backpackers,
- Family,
- Couple.

The objective is to predict the segment for a new customer using a classifier. The task is to explore different classifiers and then select the best one for predictions. Cross-validation is used to test the models by determining the F1 score. GridCV and RandomCV are used to determine optimum hyper parameters for the classifiers.

Data

Data is given in the form of two tables,

- train_search.csv:- This table contains details about the hotel bookings. The details include the check-in and check-out dates, booking date, hotel etc …
- Hotel.csv:- This table contains details about the hotel locations including a physical location (latitude, longitude, city) and previous ratings .

Objectives

1. Build a model than can predict the customer segment.
2. Analyze different classifiers and understand their accuracy and speed.
3. Implement CV based hyper-parameter optimization to tune the classifiers

Metrics

Cross-validation is used to validate the model for the test data against a model developed using the training data. The test and training data are randomly obtained from the "train_search.csv". An F1 score is obtained based on the predicted target and the actual values for the target. This F1 score is the metrics for this project.

Tasks

The exercise is done based on the following steps,

Data mining and extraction (Step-1):-

This step involves reading the train_search.csv and hotel.csv and converting the data into a single table, globalTrain.csv. This process is executed using "join" using python pandas. However, here due to the size of data, I used mysql to generate the table.

Identifying features and target in the test and training data (Step-2).

The features are extracted from the globalTrain and the target is the "Segment". This step involves the python pandas library and the following dataframes are created: X_train (features of trainigng data), y_train (target of training data), X_test (features of test dat i.e. evaluation.csv) and y_test (target of test data i.e. our final objective, the y_test is generated at the end). Features identification is done by analyzing the covariance matrix, principal components an dthe independent components.

Classifier Identification and hyper parameter optimization (Step-3)

Several classifiers are used to fit the data, and then cross-validation is used to determine the accuracy of the classifier. This task involves two sub-tasks,

1> Determine a set of three best classifiers,

2> Cross-validation is used to determine optimum values for the hyper parameters to select the final model.

Step-1

Data Extraction and Mining

This step involved data extraction, mining and manipulations. Data is extracted and mined from the two inputted csv files. This step involves the extraction of data and applying necessary joins to converts multiple dataframes into a single dataframe. The primary tables are,

- train_search.csv:-
  - Number of rows:- 162848
  - Number of columns:- 12
  - Columns:-
    - Search ID:- An unique number ...
    - Booking Date :- Hotel booking date ...
    - HotelCode:- A code for hotel identification. This is a foreign key mapped to the primary key in Hotel.csv ..
    - Age :- Age of the customer ..
    - Gender:- Gender of the customer ..
    - Number of Rooms :- Number of rooms booked in the hotel ..
    - Check in date :- Check in date in the hotel ..
    - Check out date :- Check out date for the booking ..
    - Seen Price :- Price for the booking ..
    - isClicked:- click identifier on the website ..
    - isBooked :- a boolean value to see if it was booked online or not ..
    - Segment:- This is the classifier target.
- Hotel.csv:-
  - Number of rows:- 1000
  - Number of columns:- 12
  - Columns:-
    - HotelCode:- An unique hotel code identifier. This is the primary key in this table and used as a foreign key in train_search.csv...
    - City :- City in which the hotel is located
    - Latitude :- Latitude of the location of the hotel
    - Longitude:- Longitude of the location of hotel
    - Star Rating:- Start rating of the hotel
    - TripAdvisor Ration:- Trip advisor rating of the hotel

Data Manipulation

This step involves the following tasks,

1> Removal of "NA" and missing data
2> Convert strings to date time objects in the dataframes
3> Addition of new columns by performing arithmetic operations on the existing columns
4> Convert discrete segments to discrete numbers understandable by the classifier. e.g. convert True/False to 1/0

In the current data set following actions are performed,

1> Replace True/False in "isClicked" column to 1/0
2> Replace True/False in "isBooked" column to 1/0
3> Replace Male/Female in “Gender” column to 1/0
4> Replace 'backpacker'/'couple'/'family' in "Segment" to 1/3/2
5> Convert "Booking Date" from string to datetime object
6> Convert "Check in date" from string to datetime object
7> Convert "Check Out Date" from string to datetime object
8> Created a new column "Stay Period" - Difference in days between Check out date and Check in date
9> Created a new column "Travel Gap" - Difference in days between booking date and check in date

Global Data

Now the X_train consists of the following 10 features,

1> Age (3),
2> Gender (4),
3> Number of rooms (5),
4> Seen price (8),
5> isCLicked (9),
6> isBooked (10),
7> Star Rating (16),
8> Trip Adviser Rating (17),
9> Stay period (Difference in days between Check out date and Check in date) (18),
10> Travel Gap (Difference in days between booking date and check in date) (19).

Therefore, there are 10 features as mentioned above. Some of the features have a string/object value, convert those to discrete ones and zeros. The number in braces is the column number in the "globalData" dataframe

Step - 2

This step is very significant in the development of the classifier model. In this step a detailed analysis is done to determine the features those have a strong influence of the target. Intuitively it is logical that we need to build the classifier based on influential features to prevent over fitting. Through this analysis an exploration of different feature selection methods, such as KBest, Principal Component ANalysis (PCA) and Independent Component Analysis (ICA) are implemented.

VarianceThreshold:- This method selects all the features those have a variance greater than the threshold. In the current implementation we are removing the features those have a variance of less than 25% in data. The results are,

```
[  1.80000000e+01   7.10000000e+03   3.00000000e+00   1.00000000e+00
   0.00000000e+00]
   Age  Gender  Number of Rooms  Seen Price  isClicked  isBooked  Star Rating  \
0   18       0               1        7100          1         0            3

   TripAdvisor Rating  Stay Period  Travel Gap
0                 3.5          1.0         0.0
```

In the output, only the features those pass the threshold test are printed in the function call "print sel.fit_transform(X_train)[0,:]". From the variance threshold feature selection we find out FIVE features have a strong variance. The following features are selected,

1> Age,
2> Seen Price
3> Star rating
4> Stay Period,
5> Travel Gap.

Now let us build the features training data only with the five features in them. Now a correlation matrix is calculated that determines the covariance between the parameters. It calculates the Pearson's coefficient. The values range from -1 to 1,

	Segment
Age	0.075425
Seen Price	-0.002862
Star Rating	0.002235
Stay Period	-0.025639
Travel Gap	0.018113
Segment	1.000000

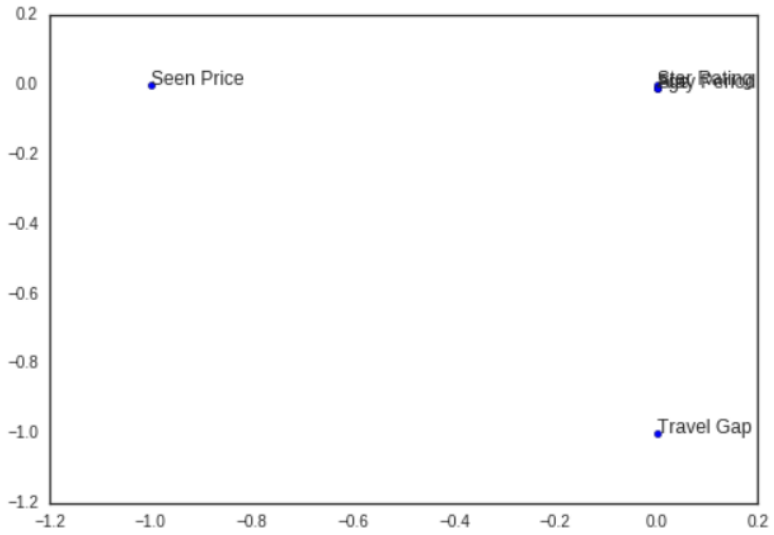
From the correlation matrix we observe the following,

1) Age is the most influential on the Segment as compared to other features.
2) Age is followed by "Stay Period" and "Travel Gap" in terms of covariances.
3) Star Rating is the least influential.

Next we look at the principal components by performing a principal component analysis,

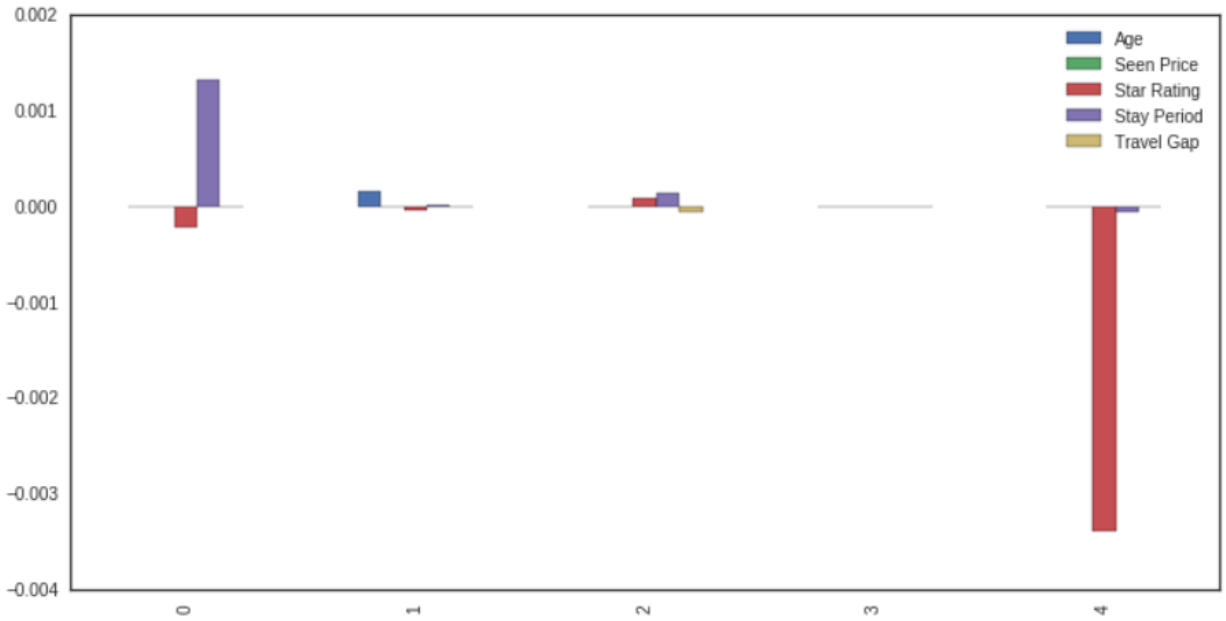
```
[  9.99999999e-01   1.05329430e-09   7.57287359e-11   1.22388248e-12
   1.87696352e-13]

<matplotlib.text.Annotation at 0x7f8af9ea2510>
```



From the principal components we observe that only one component (1st component) accounts for 99% of variance in data. From the plot it is visible that “Seen Price”, “Travel Gap” and (Age, Start Rating, Stay Period) are far apart from each other. Therefore each of these components can be individually used for building the predictive model.
Now let us look at the independent component by performing and independent component analysis (ICA).

ICA converts a multivariate dataset to additive individual components. The resulting matrix above is the ICA components. Each vector corresponds to a component, and it defines the independence with respect to other features. These components help in projecting the data from ICA axis to the original axis. The row feature can be reconstructed by a linear combination of the columns in that specific row. The ICA can be shown as,



At this point based on PCA and ICA, the features selected are, "Age", "Star rating" and "Travel Gap". Let us update our training data to contain only these three features.

Step -3

Build testing codes

The best way to test a classifier model is cross-validation. In cross-validation an input set is broken into training and testing set. The testing set is used to test the model built using the training set. And then the F1 score is computed. “train_test_split” method is used to split the input data randomly into training set and testing set. After the block is executed a set of training and test features and target are available for cross validation. The variables generated are X_train_cv, y_train_cv, X_test_cv and y_test_cv.

train_classifier method:-

This method fits a training data into an inputted classifier and also returns and prints the time it takes to train the model.

predict_labels method:-

This method takes a fitted classifier and test data to predict the labels. This method prints the prediction time.

Now try different classifiers and track the time and the accuracy.

<Table length=10>

Type	Training time	Prediction time-Training	F1 score-Training	Prediction time-Testing	F1 score-Testing
str25	float32	float32	float32	float32	float32
PassiveAggressiveClassifi	0.00794792	0.00794792	0.26596	0.00275993	0.26858
GaussianNB	0.0181749	0.0181749	0.587897	0.00532699	0.589002
DecisionTreeClassifier	0.0221438	0.0221438	0.986583	0.007617	0.977333
RandomForestClassifier	0.290855	0.290855	0.983003	0.095232	0.952203
BernoulliNB	0.015748	0.015748	0.323137	0.00455809	0.321842
LogisticRegression	0.00847602	0.00847602	0.479718	0.00287604	0.477646
KNeighborsClassifier	0.880349	0.880349	0.940098	0.286511	0.909916
AdaBoostClassifier	1.43477	1.43477	0.601057	0.459271	0.598315
LinearDiscriminantAnalysi	0.00837684	0.00837684	0.47976	0.00288296	0.477299
QuadraticDiscriminantAnal	0.0278611	0.0278611	0.587528	0.00824785	0.588294

The above table presents the performance analysis of a set of 10 different classifiers. The table presents the training time, prediction time of training data, f1 score of training data, prediction time of test data and f1 score of test data. Let us select the BEST THREE classifiers for further analysis. The next step is to use grid and random CV to determine optimized hyper parameters. The classifiers chosen at this point are,

1> Decision Tree Classifier
2> Random Forest Classifier
3> KNeighbors Classifiers

Hyper-parameter Optimization

All the classifiers have tuning parameters those affect the way the predictions are performed. These tuning parameters are called hyper-parameters. For the search of hyper parameters, the idea is to perform grid based search and random search for all the three classifier types and the analysis is plotted. First, few methods are implemented,

run_gridsearch method :-

This function takes the data, parameters, number of folds and the classifier as input. It runs different several parameter options based on a grid and returns the best parameter set.

run_randomsearch method :-

This functions takes the data, parameters, number of folds and the classifier as input. It runs different several parameter options randomly and returns the best parameter set.

report method :-

Reports the best scores.

Based on the hyper-parameter optimization following are the results for the best parameters for Grid and random based search for all the three classifiers,

DecisionTreeClassifier - RandomCV :- Model with rank: 1 Mean validation score: 0.589 (std: 0.005) Parameters: {'min_samples_split': 95, 'max_leaf_nodes': 54, 'criterion': 'entropy', 'max_depth': 11, 'min_samples_leaf': 42} DecisionTreeClassifier - GridCV :- Model with rank: 1 Mean validation score: 0.609003 (std: 0.005) Parameters: {'min_samples_split': 10, 'max_leaf_nodes': 10, 'criterion': 'gini', 'max_depth': 'None', 'min_samples_leaf': 5}

RandomForest - RandomCV :- Model with rank: 1 Mean validation score: 0.589 (std: 0.003) Parameters: {'bootstrap': True, 'min_samples_leaf': 9, 'min_samples_split': 8, 'criterion': 'entropy', 'max_features': 3, 'max_depth': 3} RandomForest - GridCV :- Model with rank: 1 Mean validation score: 0.590211 (std: 0.004) Parameters: {'bootstrap': True, 'min_samples_leaf': 2, 'min_samples_split': 5, 'criterion': 'gini', 'max_features': 1, 'max_depth': 3}

KNearestNeighbors - RandomCV :- Model with rank: 1 Mean validation score: 0.549 (std: 0.004) Parameters: {'n_neighbors': 86, 'weights': 'uniform', 'leaf_size': 93, 'algorithm': 'kd_tree', 'p': 5} KNearestNeighbors - GridCV :- Model with rank: 1 Mean validation score: 0.542189 (std: 0.004) Parameters: {'n_neighbors': 1, 'weights': 'uniform', 'leaf_size': 5, 'algorithm': 'kd_tree', 'p': 10}

The best one chosen is,

DecisionTreeClassifier with the following parameters,

('min_samples_split': 10, 'max_leaf_nodes': 10, 'criterion': 'gini', 'max_depth': 'None', 'min_samples_leaf': 5)