MODULE 6: PREDICTIVE MODELING FOR TEMPORAL DATA

CASE STUDY ACTIVITY TUTORIAL

6.2 Prediction Engineering Using UK Retail Dataset

uk_retail_case_study

November 26, 2017

1 Prediction engineering case study using UK Retail Dataset

In this case study, we will study prediction engineering. Prediction engineering is a step in predictive modeling where we: * Define an outome we are interested in predicting * Scan the data to find the past occurences of the outcome * Make these past occurences training examples for machine learning/modeling

We will then use featuretools to extract features and learn a predictive model.

In this particular casestudy, we are focusing a retail dataset openly available at http://archive.ics.uci.edu/ml/datasets/online+retail

We will define the prediction problem as the one where the customer has more than `k` purchases

```
In [1]: import featuretools as ft
        from utils import (find_training_examples, load_uk_retail_data,
                           engineer_features_uk_retail, preview, feature_importances)
        import pandas as pd
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import Imputer
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.metrics import confusion_matrix
        ft.__version__
```

```
Out[1]: '0.1.14'
```

2 Step 1: Load and prepare data

If you have not yet downloaded the data it can be downloaded from S3. Once you have downloaded the archive, unzip it and place the uk-retail-data folder in the same directory as this script.

```
In [2]: item_purchases, invoices, items, customers = load_uk_retail_data()
```

The dataset has the following tables: * `item_purchases` * `invoices` * `items` * `customers`

The following relations exist * A customer may have multiple invoices * An item may have been purchased multiple times * An invoice may have multiple item purchases

```
In [3]: entities = {
                "item_purchases": (item_purchases, "item_purchase_id", "InvoiceDate" ),
```

```
            "items": (items, "StockCode"),
            "customers": (customers,"CustomerID"),
            "invoices":(invoices,"InvoiceNo", "first_item_purchases_time")
            }

relationships = [("customers", "CustomerID","invoices", "CustomerID"),
                 ("invoices", "InvoiceNo","item_purchases", "InvoiceNo"),
                 ("items", "StockCode","item_purchases", "StockCode")]
```

3 Step 2 : Find training examples

In the code snippet below, we are trying to find training examples from the data. We set the following parameters: * `prediction_window`=14 days * `training_window`=21 days * `lead` = 7 days * `threshold`=2 --> specifies the number of purchases that the customer need to have in the future to be considered engaged

```
In [4]: label_times = find_training_examples(item_purchases, invoices,
                                              prediction_window=pd.Timedelta("14d"),
                                              training_window=pd.Timedelta("21d"),
                                              lead=pd.Timedelta("7d"),
                                              threshold=5)
```

```
In [5]: preview(label_times, 5)
```

```
Out[5]:    CustomerID    t_start  cutoff_time  purchases>threshold
        0     17505.0 2011-05-18   2011-06-08                False
        2     13592.0 2011-05-18   2011-06-08                False
        3     13650.0 2011-05-18   2011-06-08                False
        4     13756.0 2011-05-18   2011-06-08                False
        5     16116.0 2011-05-18   2011-06-08                False
```

In the output above, we are showing the first 5 training examples. The first column is the CustomerID, the second column is the timestamp after which we can use the data for generating features. The third column is the last timestamp we can use the data from the customer. The fourth column is the label. It is `True` if the customer had more than 5 purchases in the period between (`cutoff_time`, `cutoff_time + prediction_window`)

4 Step 3: Now lets generate features.

Next we generate features for each of the training examples. We use featuretools to generate the features. Featuretools is an automated feature engineering software. We go into detail about this software package in the NYC-Taxi case study. Here we simply use the tool to generate features.

```
In [6]: feature_matrix = engineer_features_uk_retail(entities, relationships,
                                                     label_times, training_window='21d')
```

```
In [7]: preview(feature_matrix, 10)
```

Out[7]:

```
            WEEK(first_invoices_time)  HOUR(first_invoices_time)  \
CustomerID
12353.0                            20                         17
16182.0                             7                         10
16186.0                            48                         14
16187.0                             9                         13
16209.0                            14                         12
16218.0                            48                         11
16226.0                            16                         12
16230.0                            18                         15
16235.0                             2                         15
16239.0                            22                          9

            MAX(item_purchases.Quantity)  STD(item_purchases.UnitPrice)  \
CustomerID
12353.0                                8                       3.911122
16182.0                              200                       0.065997
16186.0                               24                       1.349716
16187.0                               30                       3.168892
16209.0                               48                       1.263808
16218.0                               -1                       2.456296
16226.0                               12                       0.000000
16230.0                               24                       0.935393
16235.0                               24                       3.312724
16239.0                               24                       2.894689

            DAY(first_invoices_time)  IS_WEEKEND(first_invoices_time)  \
CustomerID
12353.0                           19                            False
16182.0                           18                            False
16186.0                            2                            False
16187.0                           28                            False
16209.0                            7                            False
16218.0                            1                            False
16226.0                           18                            False
16230.0                            3                            False
16235.0                           11                            False
16239.0                            3                            False

            MINUTE(first_invoices_time)  MONTH(first_invoices_time)  \
CustomerID
12353.0                              47                           5
16182.0                              11                           2
16186.0                              40                          12
16187.0                              34                           2
16209.0                              23                           4
16218.0                              29                          12
16226.0                              58                           4
```

```
16230.0                                  12                             5
16235.0                                  42                             1
16239.0                                  30                             6

            MAX(item_purchases.UnitPrice)  MEAN(item_purchases.Quantity)  \
CustomerID
12353.0                              9.95                       5.000000
16182.0                              1.79                     166.666667
16186.0                              6.25                      13.625000
16187.0                             16.95                      15.166667
16209.0                              5.95                      16.058824
16218.0                              8.15                      -1.833333
16226.0                              1.25                      12.000000
16230.0                              3.75                      10.625000
16235.0                             10.75                       8.529412
16239.0                             12.75                      12.133333

                               ...                        \
CustomerID                     ...
12353.0                        ...
16182.0                        ...
16186.0                        ...
16187.0                        ...
16209.0                        ...
16218.0                        ...
16226.0                        ...
16230.0                        ...
16235.0                        ...
16239.0                        ...

            MAX(invoices.STD(item_purchases.UnitPrice))  \
CustomerID
12353.0                                        3.911122
16182.0                                        0.065997
16186.0                                        1.349716
16187.0                                        6.798299
16209.0                                        1.263808
16218.0                                        2.456296
16226.0                                        0.000000
16230.0                                        0.935393
16235.0                                        3.312724
16239.0                                        2.894689

            STD(invoices.MAX(item_purchases.Quantity))  \
CustomerID
12353.0                                            0.0
16182.0                                            0.0
16186.0                                            0.0
```

```
16187.0                                           11.0
16209.0                                            0.0
16218.0                                            0.0
16226.0                                            0.0
16230.0                                            0.0
16235.0                                            0.0
16239.0                                            0.0

            MEAN(invoices.STD(item_purchases.UnitPrice))  \
CustomerID
12353.0                                         3.911122
16182.0                                         0.065997
16186.0                                         1.349716
16187.0                                         3.736879
16209.0                                         1.263808
16218.0                                         2.456296
16226.0                                         0.000000
16230.0                                         0.935393
16235.0                                         3.312724
16239.0                                         2.894689

            MAX(invoices.MEAN(item_purchases.Quantity))  \
CustomerID
12353.0                                        5.000000
16182.0                                      166.666667
16186.0                                       13.625000
16187.0                                       16.900000
16209.0                                       16.058824
16218.0                                       -1.833333
16226.0                                       12.000000
16230.0                                       10.625000
16235.0                                        8.529412
16239.0                                       12.133333

            MAX(invoices.STD(item_purchases.Quantity))  \
CustomerID
12353.0                                       2.236068
16182.0                                      47.140452
16186.0                                       5.109733
16187.0                                       6.048967
16209.0                                      10.026263
16218.0                                       1.067187
16226.0                                       0.000000
16230.0                                       3.789377
16235.0                                       5.510606
16239.0                                       5.678811

            MEAN(invoices.MAX(item_purchases.UnitPrice))  \
```

```
CustomerID
12353.0                                                     9.95
16182.0                                                     1.79
16186.0                                                     6.25
16187.0                                                     9.95
16209.0                                                     5.95
16218.0                                                     8.15
16226.0                                                     1.25
16230.0                                                     3.75
16235.0                                                    10.75
16239.0                                                    12.75

            MEAN(invoices.MAX(item_purchases.Quantity))  \
CustomerID
12353.0                                                    8.0
16182.0                                                  200.0
16186.0                                                   24.0
16187.0                                                   19.0
16209.0                                                   48.0
16218.0                                                   -1.0
16226.0                                                   12.0
16230.0                                                   24.0
16235.0                                                   24.0
16239.0                                                   24.0

            MEAN(invoices.MEAN(item_purchases.Quantity))  \
CustomerID
12353.0                                                 5.000000
16182.0                                               166.666667
16186.0                                                13.625000
16187.0                                                11.700000
16209.0                                                16.058824
16218.0                                                -1.833333
16226.0                                                12.000000
16230.0                                                10.625000
16235.0                                                 8.529412
16239.0                                                12.133333

            STD(invoices.MAX(item_purchases.UnitPrice))  \
CustomerID
12353.0                                                    0.0
16182.0                                                    0.0
16186.0                                                    0.0
16187.0                                                    7.0
16209.0                                                    0.0
16218.0                                                    0.0
16226.0                                                    0.0
16230.0                                                    0.0
```

```
16235.0                                                     0.0
16239.0                                                     0.0

              STD(invoices.MEAN(item_purchases.UnitPrice))
CustomerID
12353.0                                                 0.00000
16182.0                                                 0.00000
16186.0                                                 0.00000
16187.0                                                 1.87775
16209.0                                                 0.00000
16218.0                                                 0.00000
16226.0                                                 0.00000
16230.0                                                 0.00000
16235.0                                                 0.00000
16239.0                                                 0.00000

[10 rows x 27 columns]
```

5 Step 4: Let's train a model using Random Forests

Now we are ready to train a model and evaluate it. To do this, we: * First split our training examples in train_test_split * Impute missing values * Train a model using training data * Test on the data set aside for testing

We can split the data using the function `train_test_split` and specifying the proportion we want for testing. In this case we specified that as 35%

```
In [8]: label_times[["CustomerID"]]
        X_y = feature_matrix.merge(label_times[["CustomerID", 'purchases>threshold']],
                                   right_on="CustomerID", left_index=True)
        y = X_y.pop('purchases>threshold')
        X_train, X_test, y_train, y_test = train_test_split(feature_matrix,
                                                            y, test_size=0.35)
```

We can impute the missing values or `NaN` values in the feature_matrix using the `Imputer` in scikit-learn. It replaces the `NaN` values in a feature column with the `mean` of the rest of the entries in that column. This is a simple imputation startegy

```
In [9]: imp = Imputer(missing_values='NaN', strategy='mean', axis=0)
        imp = imp.fit(X_train)
        X_train_imp = imp.transform(X_train)
```

We can train a RandomForest classifier (a type of ensemble classifier). We make use of scikit-learn package for this as well.

```
In [10]: clf = RandomForestClassifier(random_state=0, n_estimators=100,
                                      class_weight="balanced", verbose=True)
         clf.fit(X_train_imp, y_train)
```

```
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed:    0.2s finished
```

```
Out[10]: RandomForestClassifier(bootstrap=True, class_weight='balanced',
                     criterion='gini', max_depth=None, max_features='auto',
                     max_leaf_nodes=None, min_impurity_split=1e-07,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
                     oob_score=False, random_state=0, verbose=True,
                     warm_start=False)
```

6 Step 5: Test the model

To test a model, we: 1. First impute the missing values * Use the trained classifier to predict the labels

```
In [11]: X_test_imp = imp.transform(X_test)
         predicted_labels = clf.predict(X_test_imp)
```

```
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed:    0.1s finished
```

We evaluate by calculating

```
In [12]: tn, fp, fn, tp = confusion_matrix(y_test, predicted_labels).ravel()
```

```
In [13]: tn, fp, fn, tp
```

```
Out[13]: (217, 4, 50, 4)
```

```
In [14]: feature_importances(clf, feature_matrix.columns, n=15)
```

```
1: Feature: STD(invoices.MAX(item_purchases.UnitPrice)), 0.062
2: Feature: MEAN(item_purchases.UnitPrice), 0.055
3: Feature: WEEK(first_invoices_time), 0.050
4: Feature: MEAN(invoices.MEAN(item_purchases.UnitPrice)), 0.049
5: Feature: DAY(first_invoices_time), 0.049
6: Feature: STD(invoices.MEAN(item_purchases.Quantity)), 0.046
7: Feature: MEAN(item_purchases.Quantity), 0.045
8: Feature: STD(invoices.MEAN(item_purchases.UnitPrice)), 0.044
9: Feature: MEAN(invoices.MEAN(item_purchases.Quantity)), 0.043
10: Feature: MAX(invoices.MEAN(item_purchases.UnitPrice)), 0.042
11: Feature: MAX(invoices.MEAN(item_purchases.Quantity)), 0.041
12: Feature: MINUTE(first_invoices_time), 0.040
13: Feature: STD(invoices.MAX(item_purchases.Quantity)), 0.038
14: Feature: STD(item_purchases.Quantity), 0.035
15: Feature: HOUR(first_invoices_time), 0.035
```