31250 Introduction to Data Analytics

Assignment #3

Data Mining in Action

Florian Lubitz
FEIT
13688799
florian.lubitz@student.uts.edu.au

Contents

1 The data mining problem

In the given scenario we have received a data set. This contains data points for various customers of an insurance company. The aim of data mining is to predict whether a customer will purchase an insurance.

The dataset contains multiple features, that have been explored in Assesment 2 of this subject. The list of features contains different type of features. Some of them are dichotomous, some are categorical and some are rational or ordinal features. This wide variety will require some preprocessing to build a good model. Most of the features contain characters as values, one of them contains a date in australian format.

The goal for this problem will be to categorize each datapoint. There are only two categories available: 1 for "will purchase insurance" and 0 for "will not purchase insurance".

2 Preprocessing and transformation

To provide a good starting point for the classifiers I preprocess the raw data. To solve the problem I use python with jupyter notebooks. The following explanations will be supported by short listings. The main sourcecode can be found in the appendix of this report.

After importing the data from csv, I start by converting the date into a unix timestamp and all ordinal features into numbers. I also convert dichotomous features that are coded with "Y" and "N" into machine readable "0" and "1", respectively.

```
1  def parse_data(df):
2      # Convert Date
3      df['Original_Quote_Date'] = df['Original_Quote_Date'].apply(
           str_to_timestamp)
4      # Convert bool-values to int of 1 and 0
5      df['Field_info4'] = df['Field_info4'].apply(string_to_bool)
6      df['Personal_info1'] = df['Personal_info1'].apply(string_to_bool)
7      df['Property_info1'] = df['Property_info1'].apply(string_to_bool)
8      df['Sales_info4'] = df['Sales_info4'].apply(string_to_value)
9      df['Personal_info3'] = df['Personal_info3'].apply(string_to_value)
10     df['Property_info3'] = df['Property_info3'].apply(string_to_value)
11     # Convert special amount to int
```

```
12      df['Field_info3'] = df['Field_info3'].apply(format_amount)
```

Listing 1: An excerpt of the parse function

After parsing all data I convert categorical features into many flags. This makes it easier for following classifiers to work with these features. To do this I use a function of `pandas` called `get_dummies`. After adding those flags I delete the original feature as it would contain redundant information. The shown function can handle multiple feature conversions at once.

```
1  def categorical_to_many(df, columns, keep_columns=None):
2      # Change Categorical
3      if keep_columns is None:
4          keep_columns = []
5      dummies = dict()
6      for col in columns:
7          dummies[col] = pd.get_dummies(df[col]).add_prefix(col + '_')
8      for dum in dummies:
9          # Keep generated columns as they might include lots of empty(same)
               values
10         keep_columns = keep_columns + list(dummies[dum].keys())
11         df.drop(columns=[dum], inplace=True)
12         df = pd.concat([df, dummies[dum]], axis=1)
13     return df, keep_columns
```

Listing 2: The function to convert one categorical feature into many dichotomous

After converting the features, the training set and and test set could contain different amount of features (flags). To solve this, I populate the sets with all missing flags.

```
1      # Fill up train and test frame to have the same column length
2      for key in list(set(train_df.keys()) - set(test_df.keys())):
3          test_df.loc[:, key] = pd.Series(np.zeros(len(test_df['
               Original_Quote_Date'])), index=test_df.index)
4      for key in list(set(test_df.keys()) - set(train_df.keys())):
5          train_df.loc[:, key] = pd.Series(np.zeros(len(train_df['
               Original_Quote_Date'])), index=train_df.index)
```

Listing 3: Add missing flag features to both sets

3 Attacking the problem

4 Classification techniques

5 The best classifier

Quote_ID, Original_Quote_Date, QuoteConversion_Flag, Field_info1, Field_info2, Field_info3, Field_info4, Coverage_info1, Coverage_info2, Coverage_info3, Sales_info1, Sales_info2, Sales_info3, Sales_info4, Sales_info5, Personal_info1, Personal_info2, Personal_info3, Personal_info4, Personal_info5, Property_info1, Property_info2, Property_info3, Property_info4, Property_info5, Geographic_info1, Geographic_info2, Geographic_info3, Geographic_info4, Geographic_info5.

List of Figures

Listings