Practical session week 6

Data Science course

This is **not** a hand-in assignment. Bring your answers to the next lecture. You need this exercise for the follow-up exercises. Make sure you finish it.

Goals of this exercise:

- Getting to know the clickbait data set
- Get somewhat more acquainted with dataframes
- Learn how to pre-process semi-structured data to be used in a supervised learning task
- Become acquainted with supervised learning in Python with scikit-learn

Clickbait is Internet content whose main purpose is to attract attention and encourage visitors to click on a link to a particular web page.

We are going to conduct a series of classification experiments with the Facebook data (news and clickbait). The research questions that will address in the following weeks are:

- To what extent is it possible automatically distinguish a clickbait post from a mainstream news post on Facebook?
- What features are the most important in making this distinction?

Preliminaries

The practical session of week 1 (data frames in R and Python)

Make sure you have installed Python 3 and the following packages (probably, it will also work with Python 2.7 but I cannot guarantee the compatibility of the packages):

- pandas
- numpy
- scikit-learn

On linux, the command for installing Python packages is:

```
pip install -U pandas numpy scikit-learn
```
(the U means 'update')

It might be useful to work in ipython (or a jupyter notebook) so that you don't have to run the complete script everytime (reading the data takes time).

Tasks

1. Preparation

1. Download the following data files `n_fb_post.csv` and `c_fb_post.csv` (together as zip-file on Blackboard)

`c_fb_post` is a collection of posts published on the Facebook pages of clickbait sites. `n_fb_post` is a collection of posts published on the Facebook pages of mainstream news sites.

2. Import the csv files in Python as Pandas dataframes. Make sure you store the first row of the csv files as column header (so that you can refer to a column by using the column name).

2. Data exploration

3. Answer the following questions about the data:
   a. How many data points and how many columns do the datasets have?
   b. What are the column names?

4. Print the unique values in the column '`fb_page`' to see which pages are included in the two datasets.

3. Pre-processing the data into a feature matrix

X and y are the variable names commonly used for the feature matrix (X) and the array (column) with class labels (y):

5. Initialize the feature matrix:

```
X = []
```

And initialize the label array:

```
y = []
```

6. We are going to create a feature vector for every item in the clickbait and news dataframes.

   Create a loop that walks through both dataframes. For each row in the data, create a 4-dimensional vector with the following features:

```
vector = [row['post_type'],row['num_reaction_cleaned'],row['num_comment_cleaned'],
                                                     row['num_share_cleaned']]
```

Append the vector to the feature matrix:

```
X.append(vector)
```

(Hint: you will need to add a function to replace categorical values before adding them to X: https://scikit-learn.org/stable/modules/preprocessing.html#encoding-categorical-features)

After you filled X, store it as a numpy array:

```
X = numpy.asarray(X)
```

7. For the clickbait items, add the value 'c' to the label array y. For the news items, add the value 'n'. Make sure that the length of y is equal to the length of X: there is a label for each item in the feature matrix.

4. Classification

8. X and y now contain the complete data. For training and validating classifiers, we need a train-test split. Use the `train_test_split` function in `scikit-learn.cross_validation` to create four files: `X_train, X_test, y_train, y_test`. Use a random 20% of the data as test set.

9. Now, let's try our first classifier: LinearSVC (default settings). You can find the documentation on http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
   a. initiate the classifier: `clf = LinearSVC()`
   b. fit the training data
   c. predict values for the test data and assign them to a variable y_pred

10. Use the function `classification_report` from `sklearn_metrics` to evaluate your classifier. What is the precision and recall that you obtain with your classifier?