Large-Scale Data Mining: Models and Algorithms
Prof. Vwani Roychowdhury
UCLA, Department of ECE

ECE 219 Spring 2018
TA: Tianyi Wang
Behnam Shahbazi

Project 5: Application - Twitter data

Due on Monday, June 11 2018 by 11:59 pm

Introduction: A useful practice in social network analysis is to predict future popularity of a subject or event. Twitter, with its public discussion model, is a good platform to perform such analysis. With Twitter's topic structure in mind, the problem can be stated as: knowing current (and previous) tweet activity for a hashtag, can we predict its tweet activity in the future? More specifically, can we predict if it will become more popular and if so by how much? In this project, we will try to formulate and solve an instance of such problems.

The available Twitter data is collected by querying popular hashtags related to the 2015 Super Bowl spanning a period starting from 2 weeks before the game to a week after the game. We will use data from some of the related hashtags to train a regression model and then use the model to make predictions for other hashtags. To train the model, you need to prepare training sets out of the data, extract features for them, and then fit a regression model on it. The regression model will try to fit a curve through observed values of features and outcomes to create a predictor for new samples. Designing and choosing good features is one of the most important steps in this process and is essential to getting a more accurate system. There are examples of such analysis and useful features in literature [1] (You should look into the literature for this). You will be given training data to create the model, and test data to make predictions. The test data consists of tweets containing a hashtag in a specified time window, and you will use your model to predict number of tweets containing the hashtag posted within one hour immediately following the given time window.

Part 1: Popularity Prediction

1. A first look at the data

Download the training tweet data[2]. The data consists of 6 text files, each one filled with tweet data from one hashtag as indicated in the filenames.

QUESTION 1: Report the following statistics for each hashtag:

- Average number of tweets per hour
- Average number of followers of users posting the tweets per tweet (to make it simple, we average over the number of tweets; if a users posted twice, we count the user and the user's followers twice as well)
- Average number of retweets per tweet

[1] http://arxiv.org/abs/1401.2018
[2] https://ucla.box.com/s/24oxnhsoj6kpxhl6gyvuck25i3s4426d

QUESTION 2: Plot "number of tweets in hour" over time for `#SuperBowl` and `#NFL` (a histogram with 1-hour bins). The tweets are stored in separate files for different hashtags and files are named as `tweet_[#hashtag].txt`.

Note: The tweet file contains one tweet in each line and tweets are sorted with respect to their posting time. Each tweet is a JSON string that you can load in Python as a dictionary. For example, if you parse it to object `json_object = json.loads(json_string)`, you can look up the time a tweet is posted by:
`json_object['citation_date']`

You may also assess the number of retweets of a tweet through the following command:
`json_object['metrics']['citations']['total']`

Besides, the number of followers of the person tweeting can be retrieved via:
`json_object['author']['followers']`

The time information in the data file is in the form of UNIX time, which "encodes a point in time as a scalar real number which represents the number of seconds that have passed since the beginning of 00:00:00 UTC Thursday, 1 January 1970" (see Wikipedia for details). In Python, you can convert it to human-readable date by

```
import datetime
datetime_object = datetime.datetime.fromtimestamp(unix_time)
```

The conversion above gives out a `datetime` object storing the date and time in your local time zone corresponding to that UNIX time.

In later parts of the project, you may need to use the PST time zone to interpret the UNIX timestamps. To specify the time zone you would like to use, refer to the example below:

```
import pytz
pst_tz = pytz.timezone('America/Los_Angeles')
datetime_object_in_pst_timezone = datetime.datetime.fromtimestamp(unix_time,
↪  pst_tz)
```

For more details about `datetime` operation and time zones, see

https://medium.com/@eleroy/10-things-you-need-to-know-about-date-and-time-in-python-with-datetime-pytz-dateutil-timedelta-309bfbafb3f7

2. Linear regression

Create time windows from the data to extract features. Here, use 1-hour time window and calculate the features in each time window, resulting in `<# of hours>` data points.

For each hashtag data file, fit a linear regression model using the following 5 features to predict number of tweets in the next hour, with features extracted from tweet data in the previous hour.
The features you should use are:

- Number of tweets
- Total number of retweets
- Sum of the number of followers of the users posting the hashtag
- Maximum number of followers of the users posting the hashtag

- Time of the day (which could take 24 values that represent hours of the day with respect to a given time zone)

For each hashtag, you should train a separate model.

QUESTION 3: For each of your models, report your model's Mean Squared Error (MSE) and R-squared measure. Also, analyse the significance of each feature using the t-test and p-value. You may use the `OLS` in the libarary `statsmodels` in Python.

3. Feature analysis

QUESTION 4: Design a regression model using any features from the papers you find or other new features you may find useful for this problem. Fit your model on the data of each hashtag and report fitting MSE and significance of features.

QUESTION 5: For each of the top 3 features (*i.e.* with the smallest p-values) in your measurements, draw a scatter plot of predictant (number of tweets for next hour) versus value of that feature, using all the samples you have extracted, and analyze it.

Do the regression coefficients agree with the trends in the plots? If not, why?

4. Piece-wise linear regression

Since we know the Super Bowl's date and time, we can create different regression models for different periods of time. First, when the hashtags haven't become very active; second, their active period; and third, after they pass their high-activity time.

QUESTION 6: We define three time periods and their corresponding window length as follows:

1. Before Feb. 1, 8:00 a.m.: **1-hour** window
2. Between Feb. 1, 8:00 a.m. and 8:00 p.m.: **5-minute** window
3. After Feb. 1, 8:00 p.m.: **1-hour** window

For each hashtag, train 3 regression models, one for each of these time periods (the times are all in PST). Report the MSE and R-squared score for each case.

QUESTION 7: Also, aggregate the data of all hashtags, and train 3 models (for the intervals mentioned above) to predict the number of tweets in the next hour on the aggregated data.

Perform the same evaluations on your combined model and compare with models you trained for individual hashtags.

5. Nonlinear regressions

Ensemble methods

In this part, we use `RandomForestRegressor` and `GradientBoostingRegressor` from `sklearn` as two examples of ensemble regressors. Still use the aggregated data in this part.

QUESTION 8: Use grid search to find the best parameter set for RandomForestRegressor and GradientBoostingRegressor respectively. Use the following param_grid

```
{
  'max_depth': [10, 20, 40, 60, 80, 100, 200, None],
  'max_features': ['auto', 'sqrt'],
  'min_samples_leaf': [1, 2, 4],
  'min_samples_split': [2, 5, 10],
  'n_estimators': [200, 400, 600, 800, 1000,
                   1200, 1400, 1600, 1800, 2000]
}
```

Set `cv = KFold(5, shuffle=True)`, `scoring='neg_mean_squared_error'` for the grid search.

Analyze the result of the grid search. Do the test errors from cross-validation look good? If not, please explain the reason.

QUESTION 9: Compare the best estimator you found in the grid search with OLS on the entire dataset.

QUESTION 10: For each time period described in Question 6, perform the same grid search above for GradientBoostingRegressor (with corresponding time window length). Does the cross-validation test error change? Are the best parameter set you find in each period agree with those you found above?

Neural network

QUESTION 11: Now try to regress the aggregated data with MLPRegressor. Try different architectures (i.e. the structure of the network) by adjusting hidden_layer_sizes. You should try at least 5 architectures with various numbers of layers and layer sizes. Report the architectures you tried, as well as its MSE of fitting the entire aggregated data.

QUESTION 12: Use StandardScaler to scale the data before feeding it to MLPRegressor (with the best architecture you got above). Does its performance increase?

QUESTION 13: Using grid search, find the best architecture **for each period (with corresponding window length) described in** Question 6.

6. Using 6x window to predict

Download the test data[3]. Each file in the test data contains a hashtag's tweets from a 6x-window-length time range. Fit a model on the aggregate of the training data for all hashtags, and predict the number of tweets in the next hour for each test file. The file names consist of sample number followed by the period number the data is from.

For example, a file named `sample0_period1.txt` contains tweets in a sample 6-hour[4] window that lies in the 1st time period described in Question 6, while a file named `sample0_period2.txt` contains tweets in a sample 30-min[5] window that lies in the 2nd time period. One can be creative here, and use the data from all previous 6 hours for making more accurate predictions (as opposed to using features from the previous hour only).

[3] https://ucla.box.com/s/nkgqr39embdt67lod4pc289xe07eg6wr
[4] $1\,\mathrm{h} \times 6 = 6\,\mathrm{h}$
[5] $5\,\mathrm{min} \times 6 = 30\,\mathrm{min}$

QUESTION 14: Report the model you use. For each test file, provide your predictions on the number of tweets in the next time window.

Note: Test data should not be used as a source for training. You are not bounded to only linear models. You can find your best model through cross validation of your training data.

Part 2: Fan Base Prediction

The textual content of a tweet can reveal some information about the author. For instance, users tweeting on a topic may have opposing views about it. In particular, tweets posted by fans of different teams during a sport game describe similar events in different terms and sentiments. Recognizing that supporting a sport team has a lot to do with the user location, we try to use the textual content of the tweet posted by a user to predict their location. In order to make the problem more specific, let us consider all the tweets including `#superbowl`, posted by the users whose specified location is either in the state of Washington (not D.C.!) or Massachusetts. For example, in order to include all the tweets with the author located in the state of Washington, we consider the tweets that include the following substrings in the location field (`json_object['tweet']['user']['location']`):

- Seattle, Washington
- Washington
- WA
- Seattle, WA
- Kirkland, Washington
- etc.

QUESTION 15:

1. Explain the method you use to determine whether the location is in Washington, Massachusetts or neither. Only use the tweets whose authors belong to either Washington or Massachusetts for the next part.
2. Train a binary classifier to predict the location of the author of a tweet (Washington or Massachusetts), given only the textual content of the tweet (using the techniques you learnt in project 1). Try different classification algorithms (at least 3). For each, plot ROC curve, report confusion matrix, and calculate accuracy, recall and precision.

Part 3: Define Your Own Project

QUESTION 16: The dataset in hands is rich as there is a lot of metadata to each tweet. Be creative and propose a **new** problem (something interesting that can be inferred from this dataset) other than the previous parts. You can look into the literature of Twitter data analysis to get some ideas. Implement your idea and show that it works. As a suggestion, you might provide some analysis based on changes of tweet sentiments for fans of the opponent teams participating in the match. You get full credit for briniging in novelty and full or partial implementation of your new ideas.

Submission:

Please submit a zip file containing your report, and your codes with a readme file on how to run your code to CCLE. The zip file should be named as "Project5_UID1_UID2_..._UIDn.zip" where UIDx are student ID numbers of the team members. You should **not** include data in your submission.