Foodcasting

What is it?

> forecasting restaurant success based on menu items and yelp reviews

Class: CS 410: Text Information Systems Group Project
Authors:

- James Banasiak jamesmb3@illinois.edu
- Mark Hornback markh4@illinois.edu

Date: 10/12/2019

Git'ing the code

This code and all materials (except data sets) will be stored at the following [private] repository location. Request permission to the repo by emailing the above.

```
git clone https://lab.textdata.org/jamesmb3/foodcasting
```

Presentation Slide Deck

Google Slides **public**

Presentation.mp4 requires **@illinos.edu** identification

Data used

data.zip requires **@illinos.edu** identification.

unzip to `data` directory - then you can import it (below) under `Rebuilding local data`

```
cd foodcasting
unzip data.zip
```

Report Tools Used

To reproduce this report in its entirety, the following tools must be installed.

- **Atom** - A hackable text editor for the 21st Century download
  - used Markdown Preview Enhanced package for markdown and prince for pdf conversion.
  - used mermaid for diagrams

Overview

Restaurant menus and yelp reviews have been a target for testing various computational tasks because the data is publicly available. We plan to utilize `menu` data and `yelp` reviews data to attempt to predict the likelihood that a restaurant will have positive reviews based on the menu provided.

We plan to achieve this by creating a language model for the menus and separately performing sentiment analysis for reviews. We will take into account document length and term frequency.

Professor of Computer Science at Stanford University Dan Jurafsky has authored several papers on the subject showing how `word choice` distinguishes `price` and how `low` one-star `ratings` were related to traumatic experiences and `low` sentiment.

ℹ Narrative Framing of Consumer Sentiment in Online Restaurant Reviews

We will attempt to further this research by utilizing data sets and providing a reverse mechanism to estimate sentiment based on the menu contents.

Proposal

What is the function of the tool? Who will benefit from such a tool?

This tool is designed for `restaurants` to help distinguish which `menus` and `menu items` may be more successful than other menus and items. The individual descriptions for menu items and word choice can play a key role in determining price and positive sentiment of reviews, which translates into success of the restaurant. This will also potentially help restaurant suppliers and/or investors focus their energy on restaurants that will be more successful. If time permits, this can also be extended into item ordering and layout as well.

Does this kind of tools already exist?

As of the time of our research, it appears that no such tool does exist.

What existing resources can you use?

- Yelp dataset

- Scrapy
- Tesseract
- AWS Sagemaker

What techniques/algorithms will you use to develop the tool?

- Logistic regression with the SGD optimizer
- Logistic regression with the LBFGS optimizer
- Support Vector Machines
- Decision Trees
- Gradient Boosted Trees
- Random Forests
- Naive Bayes
- NTLK
- Valence Aware Dictionary and sEntiment Reasoner (Vader)

How will you demonstrate the usefulness of your tool.

We will demonstrate a user calling our application and passing it a menu file. The output will be a yelp star value that we are most confident with.

What is exactly the function of the tool that you would like to develop?

Given a menu as input, provide a yelp star prediction based on a model created from our corpus of menus and reviews.

Do you have some rough idea about how the target function might be achieved?

The program can be run via commandline or API. We will collect menus and corresponding yelp reviews and split them into training sets to tune our model.

What is the minimum goal to be achieved during this semester?

Given a menu as input, provide a yelp star prediction.

Gathering the corpus

The description of the elements contained within the data set is:

For the business listings -

- `slug` - a unique portion of the url, typically dashed business name for http://{host}.com/path/{slug}
- `categories` - the type of foods the restaurant serves, eg Pizza, Chinese, Mexican
- `distance` - the distance from the centroid of the zip code
- `name` - the name of the restaurant

- `price_level` - a yelp categorical price level $= under $10. $$=11-30. $$$=31-60. $$$$= over 61 USD
- `rating` - **predictor** the yelp rating associated with a restaurnat
- `review_count` - the number of reviews that were taken to establish the rating
- `url` - the url from yelp
- `lat` -the latitude of the location of the restaurant
- `lng` - the longitude of the location of the restaurant
- `Sp1` - a spacer (not used)
- `type` - the type - all `natural` words
- `homeurl` - the path portion of the url
- `resource_id1` - a resource id used in yelp specific api
- `resource_id2` - a resource id used in yelp specific api
- `lat2` - the latitude again resulting from a join within scraper
- `lng2` - the longitude again resulting from a join within scraper

For the menuitems-

- `slug` - a unique portion of the url, typically dashed business name for http://{host}.com/path/{slug}
- `title` - a menuitem title eg `Chicken Caesar Salad`
- `description` - the longer description of the title eg `Grilled chicken, romaine, Parmesan, tomatoes and Caesar dressing`
- `price` - the price of the menuitem eg `7.99`

Sample of Chicago Data

You can download a smaller subset of the chicago data that we used to initially test. The entire data.zip file is provided above for the full data set that was used in the project.
chicago-menu.csv
chicago-summary.csv

Setting up the Database

Why `sqlite3` database?
Here is a good article that explains sqlite vs pandas and performance gains.
In addition joining is much faster and we can do more complex things within `sqlite3`.

```
# clone repo
git clone https://lab.textdata.org/jamesmb3/foodcasting
# change dir
cd foodcasting
# setup sql lite to setup database
sqlite3 sql-lite-cache/foodcasting.db
```

Run the create-db-yelp.sql script from within sqlite3

```
sqlite> .read create-db-yelp.sql
```

Rebuilding local data

- Clean tables

```
(base) > $ sqlite3 sql-lite-cache foodcasting.db
      SQLite version 3.30.0 2019-10-04 15:03:17
```

```
sqlite> DROP TABLE summary;
sqlite> DROP TABLE menu;
sqlite> .read create-db-yelp.sql
sqlite> .quit
```

- Import compressed summary from shell

```
gunzip -c ./data/summary.csv.gz | sqlite3 -csv -separator ',' sql-lite-cache/foodcasting.db '
```

- Import compressed menu from shell

```
gunzip -c ./data/menu.csv.gz | sqlite3 -csv -separator ',' sql-lite-cache/foodcasting.db '.im
```

- Verify counts

```
(base) > $ sqlite3 sql-lite-cache/foodcasting.db
SQLite version 3.30.0 2019-10-04 15:03:17

sqlite> select count(*) from summary;
65480
sqlite> select count(*) from menu;
2244981
sqlite> .quit
```

- Optional - import the census data used to determine top 1000 zip codes.

```
gunzip -c scrapy_yelp/data/population_by_zip_2010.csv.gz | sqlite3 -csv -separator ',' sql-li
```

Image ocr with tesserocr

```
brew install tesseract
pip install tesserocr
pip install Pillow
```

Use the notebook ocr/menu-ocr and run each step.

API endpoint set up and configuration

Using a Sagemaker deployed model, a lambda was created to invoke the endpoint.

The lambda code can be viewed at

```
/aws/lambda_function.py
```

A public facing API endpoint was configured to access the lambda, allowing the public to access our model with a simple POST request, with the data in the request body. (Strip or delimit commas from the menu text). The body contains a data object of “menu text, category, restaurant name, price, review count”

https://q3di7y5jsj.execute-api.us-east-1.amazonaws.com/yelp-test/yelp-menu
Body:

```
{"data":"Wine $60 steak $65 pasta,Steak,Genes steakhouse,$$$$$,1000"}
```

Get prediction of Menu Using curl

In the presentation we use different requests to get the values of rating of the menu being.

```
curl --header "Content-Type: application/json" \
  --request POST \
  --data '{"data":"Whopper $5 Fries $2 Shake $4,Fast Food Burgers,Burger King,1,100"}' \
  https://q3di7y5jsj.execute-api.us-east-1.amazonaws.com/yelp-test/yelp-menu
```

> 3.9464221000671387%

```
curl --header "Content-Type: application/json" \
  --request POST \
  --data '{"data":"Whopper $5 Fries $2 Shake $4,Pizza,Pizza,5,1000"}' \
  https://q3di7y5jsj.execute-api.us-east-1.amazonaws.com/yelp-test/yelp-menu
```

> 4.177654743194558%

Acknowledgements

This report acknowledges Professor ChengXiang Zhai and the wonderful TA’s for their advice and expertise in within Intelligent Information Systems and `CS410 Text Information Systems`.