Operating Report: Chicago 2017

# Workflow

1. Download the datasets from Divvy’s website and from Yelp’s. There is a pre-processed Divvy dataset available on Kaggle but I chose to use Divvy’s raw data and process it myself. I used the Kaggle dataset only to extract weather historical weather data (I have previously used Weather Underground, but they have recently removed free access to their API).
2. Divvy provides a live JSON feed of data from their stations. This is the only source of GPS data available (except for the Kaggle dataset). There are a few stations in the 2017 data which are not included in the JSON feed, so I assume these stations have been removed. They were not popular stations and otherwise did not come up in my analysis, so I removed them from analyses which required GPS information.
3. Data munging can be followed along in the included Jupyter notebook.
4. Top 5 stations with the most starts:
   1. Streeter Dr & Grand Ave; 97,571 starts
   2. Lake Shore Dr & Monroe St; 53,400 starts
   3. Canal St & Adams St; 50,911 starts
   4. Clinton St & Washington Blvd; 49,832 starts
   5. Theater on the Lake; 47, 908 starts
5. Trip duration by user type
   1. Customer (24-hour pass) median trip duration: 1351 seconds (23 minutes)
   2. Dependent (Explore pass) median trip duration: 804 seconds (13 minutes)
   3. Subscriber (annual pass) median trip duration: 572 seconds (10 minutes)
6. Most popular trips based on start and stop stations
   1. Lake Shore Dr & Monroe St – Streeter Dr & Grand Ave
   2. Streeter Dr & Grand Ave – Streeter Dr & Grand Ave (same start and stop station)
   3. Streeter Dr & Grand Ave – Theater on the Lake
   4. Streeter Dr & Grand Ave – Lake Shore Dr & North Blvd
   5. Lake Shore Dr & North Blvd – Streeter Dr & Grand Ave
   6. Streeter Dr & Grand Ave – Lake Shore Drive and Monroe St
   7. Theater on the Lake – Streeter Dr & Grand Ave
   8. Lake Shore Drive and Monroe St – Lake Shore Drive and Monroe St (same start and stop station)
   9. Streeter Dr & Grand Ave – Michigan Ave & Oak St
   10. Streeter Dr & Grand Ave – Millennium Park
7. Rider Performance by gender and age. I binned the ages into generational categories for this analysis. Additionally, I excluded ages below 10 and above 80. There are a very small handful of ages in the dataset between 0 and 10 but quite a few at 0 exactly. Also, there were quite a few ages above 100. Both ages at 0 and above 100 seem very suspect but because they were a small portion of the data, excluded them seemed the proper approach.
   1. Both by median speed and average trip distance, Millennials were the highest achievers followed by Gen Xers and Baby Boomers with Gen Z picking up the rear.
   2. By median speed, the Silent Generation was the slowest but by average trip distance, the Silent Generation beat Generation Z as well as Baby Boomers. Interestingly, women of the oldest generation had the longest overall trip distance.
   3. Men rode faster than women on balance, but women rode further than men.
8. Busiest bike in Chicago in 2017. Statistics were calculated in a Tableau file using a CSV output.
   1. Bike ID: 2565
   2. Rides: 1,489
   3. Minutes ridden: 22,526 (375 hours)
   4. Miles ridden: 2, 175
   5. Average speed: 5.79 mph
9. Top 10 and Bottom 10 restaurants in Illinois by check-ins. I included only restaurants which were marked as open in the dataset and also excluded restaurants with no check-ins, simply because there were many of them and I wanted to be sure we were looking at real restaurants.
   1. Top 10
      1. Sakanaya
      2. Black Dog Smoke & Ale House
      3. DESTIHL Restaurant and Brew Works
      4. Seven Saints
      5. Golden Harbor Authentic Chinese Cuisine
      6. Maize Mexican Grill
      7. Meijer
      8. Café Kopi
      9. Courier Café
      10. Big Grove Tavern
   2. Bottom 10
      1. The Spice Box
      2. Monicals Pizza
      3. Tasty Fish Chicken & Grill
      4. Taco Bell
      5. Subway
      6. Red Cape Hot Pot
      7. Nanjing Bistro
      8. Main Street Belly Deli
      9. Little Caesar’s Pizza
      10. Dairy Queen
10. For the Top 10 and Bottom 10, calculate average star rating and sentiment. I used the VADER package for python to analyze sentiment. VADER stands for Valence Aware Dictionary for sEntiment Reasoning; it is specifically tuned to handle social media posts, so I thought it would be good with the Yelp dataset.

|  |  |  |
| --- | --- | --- |
| Name | Stars | Sentiment |
| Sakanaya | 4.414 | 4.651 |
| Black Dog Smoke & Ale House | 4.409 | 4.680 |
| Maize Mexican Grill | 4.160 | 4.324 |
| Golden Harbor Authentic Chinese Cuisine | 4.303 | 4.051 |
| Seven Saints | 4.417 | 4.434 |
| Cafe Kopi | 3.842 | 4.498 |
| DESTIHL Restaurant & Brew Works | 4.019 | 4.492 |
| Courier Cafe | 4.368 | 4.695 |
| Big Grove Tavern | 2.936 | 3.882 |
| Meijer | 3.167 | 3.633 |
| Dairy Queen | 1.000 | 0.751 |
| Little Caesars Pizza | 2.333 | 2.070 |
| Main Street Belly Deli | 4.667 | 4.852 |
| Monicals Pizza | 5.000 | 4.804 |
| Nanjing Bistro | 3.000 | 4.828 |
| Red Cape Hot Pot | 3.400 | 3.456 |
| Subway | 5.000 | 4.870 |
| Taco Bell | 1.000 | 1.996 |
| Tasty Fish Chicken & Grill | 1.750 | 2.477 |
| The Spice Box | 5.000 | 4.927 |

1. Top 10 cuisine types based on number of restaurants and number of check-ins. For this section, I looked at every entry in the Category column of the Yelp dataset. I included only business which had “Restaurant” as the category. Then I sorted all of the categories by frequency and by hand selected those categories which were ethnic cuisines. For example, “Pizza” and “Sandwiches” category came up, but I didn’t classify them as Cuisine per se and so did not include them here. The order below is by number of restaurants. See the presentation for the order by check-ins.
   1. American (Traditional)
   2. Italian
   3. Mexican
   4. American (New)
   5. Chinese
   6. Japanese
   7. Mediterranean
   8. Asian Fusion
   9. Thai
   10. Indian
2. What are the most popular keywords or adjectives reviewers use for each cuisine? See the presentation for the word clouds and the Jupyter Notebook for the coding procedure. Almost every cuisine’s most popular descriptor is “Good.” “Great” and “Delicious” also come up a lot. Indian and Korean are exceptions: their most popular adjectives are the “Indian” and “Korean” respectively.
3. Are there any temporal trends?
   1. Check-ins are highest on the weekends and drop during the week (with a small pick-up on Wednesday).
   2. Check-ins are lowest in the winter and highest in the summer.
   3. Most holidays feature fewer check-ins, except for New Year’s Eve, Election Day, and President’s Day. There is a spike on August 12th, International Youth Day, but I doubt that holiday is the reason for the spike, which remains unexplained. I searched and could not find any historical events which occurred on that day. Similarly, for June 8th, where there is a drop which does not coincide with any major event or holiday I could find. I found it interesting that check-ins on Valentine’s Day are low, however check-ins do rise above the norm for the weekends immediately before and after.
4. Trip duration prediction model.

The easiest (and possibly the most accurate) method would be to simply plug in to Google’s Maps API which will provide a road-by-road route as well as an estimated duration, based partly on the time of day and current conditions. Google used to offer this for free (rate-limited, of course) but has recently started charging even for academic use. Therefore, I built my own model.

I used direct line Trip Duration as an input, because the more accurate method of following roads is quite a bit more advanced than the scope of this project. I also wanted to include weather information and went to Weather Underground, my go-to API for weather data. However, they, too, have recently gone under a paywall so I downloaded the Divvy dataset from Yelp, which does include weather data, and joined the weather data into my processed data. I also looked at government statistics regarding traffic flows in Chicago and added Rush Hour as a feature in my data. I dummified data for day-of-week, month, user type, weather events, and weather conditions.

I used Scikit Learn’s linear regression model as well as its train-test-split function, so I could measure accuracy without any bias. My final accuracy was 69.5%, not very good but not bad at all for an initial pass. And after plotting the data (see the presentation) it is clear that a high-accuracy model would not be possible; there’s just too much noise in the data. Some people ride quickly from Point A to Point B, while others take their time (up to several hours!). In this model, I excluded trips over an hour and less than 200 meters, which helped a good deal (not least by removing those trips from Point A to Point A which really through off the predictions!)

1. Correlation between review length and star rating.

I find no correlation between review length and start rating. The regression model shows a slight negatively-sloping line comparing the two variables, but the P-value is 0.0 which means we must reject the null hypothesis that the data is correlated. In short, the model would predict a review length of ~700 characters regardless of how many stars were awarded.

1. Topic modeling

I performed LDA analysis on the text of the reviews. Processing time became a serious constraint in this part of the project. To help, I took a random sample of 50,000 reviews instead of the more than 600,000 available in total. I removed stop words and lemmatized in order to reduce the sparsity of my matrix and only include meaningful words. Additionally, I removed all punctuation. I did not build bigrams and trigrams, in order to reduce processing time, but feel this is a good area for improvement. Finally, I built LDA models using a number of topics of 5, 10, 15, 20, 30, 40, 50, and 75, and stored the perplexity at each value. Using this, I could plot the perplexity vs number of features and look for an elbow. Unfortunately, I don’t feel I captured the elbow. I need a finer-grained selection of topics numbers and also need to expand the set, possibly up to 300.

What I have now is a list of words (which can be seen in the Jupyter Notebook) belonging to each of 50 topics (I chose 50 nearly arbitrarily, although the elbow plot does show a slight bend at this point). The next step would be to have an SME go through these words and manually name each cluster, depending upon the vocabulary. Probable clusters might include café, pizza, quiet, date, etc.