Predicting Subreddits: Retail & Service

Trevor O'Donnell

Problem Statement

- I work for the Service Employees International Union, a labor union representing almost 1.9 million workers in over 100 occupations in the United States and Canada.
- The SEIU is considering whether to expand the scope of their organization, to include servers as well.
- They want to better understand what the server experience is, and the main factors that differentiate it from the server experience.
- **My job is to create a predictive model that will classify whether a post comes from either the retail or server subreddit based on the title and text of the post.**

Background Information

Retail workers vs Servers

Similarities: Customer service, cash-handling, working on feet

Differences: Location (stores & restaurants), Tips, Food & Beverages

Web Scraping

- Awesome function from Gwen Rathgeber
- Automates html requests and converts into pandas dataframe
- day windows = 25, n = 130
- sleep (6)
- #is_self – including empty text

```
def query_pushshift(subreddit, kind = 'submission', day_window = 25, n = 130):
    SUBFIELDS = ['title', 'selftext', 'subreddit', 'created_utc', 'author', 'num_comments', 'sc

    # establish base url and stem
    BASE_URL = f"https://api.pushshift.io/reddit/search/{kind}" # also known as the "API endpoi
    stem = f"{BASE_URL}?subreddit={subreddit}&size=100" # always pulling max of 500

    # instantiate empty list for temp storage
    posts = []

    # implement for loop with `time.sleep(2)`
    for i in range(1, n + 1): #setting the
        URL = "{}&after={}d".format(stem, day_window * i) #decalres 'URL' combining our stem UR
        print("Querying from: " + URL) #prints the source URL as it retrieves the data
        response = requests.get(URL) #retrieves information from URL location, stored in 'respo
        assert response.status_code == 200 #throws an error if URL is inaccessible, (if the boo
        mine = response.json()['data'] #user JavaScript encoder convert method
        df = pd.DataFrame.from_dict(mine) #converting to pandas dataframe
        posts.append(df) #creating a list from the dataframe,
        time.sleep(6) #this method puts five seconds between each iteration of the list

    # pd.concat storage list
    full = pd.concat(posts, sort=False)

    # if submission
    if kind == "submission":
        # select desired columns
        full = full[SUBFIELDS]
        # drop duplicates
        full.drop_duplicates(inplace = True)
        # select `is_self` == True
        #full = full.loc[full['is_self'] == True]

    # create `timestamp` column
    full['timestamp'] = full["created_utc"].map(dt.date.fromtimestamp)

    print("Query Complete!")
    return full
```

Meta data

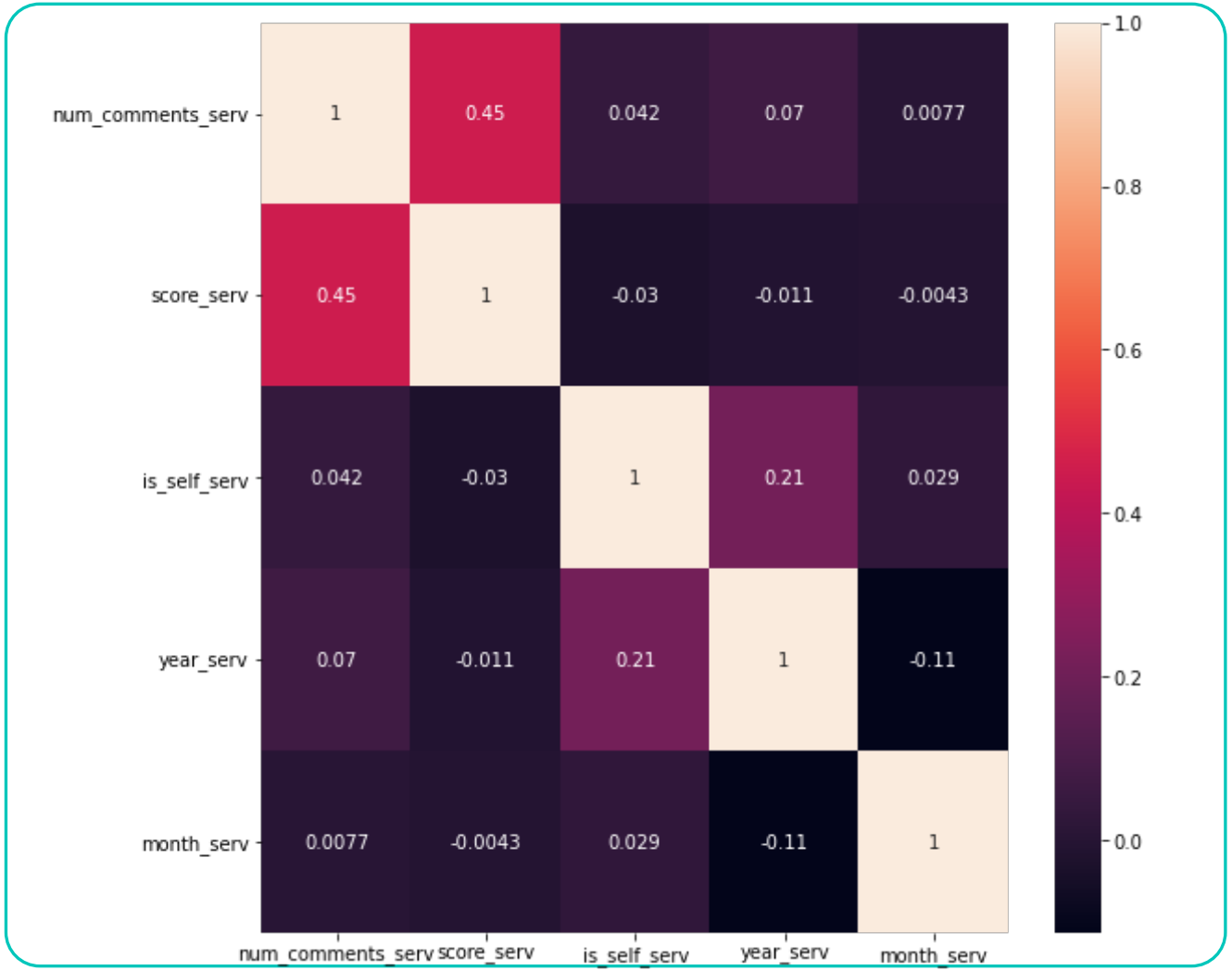
- Combined Title and Text
- Added month and time columns
- Exclusively Examined posts

  before Nov. 1st. 2012
- Merged and Separate CSV

- 11,177 Retail – 0 - .501278%
- 11,120 Server – 1 - **.498722%**
- 50% Baseline Score

EDA

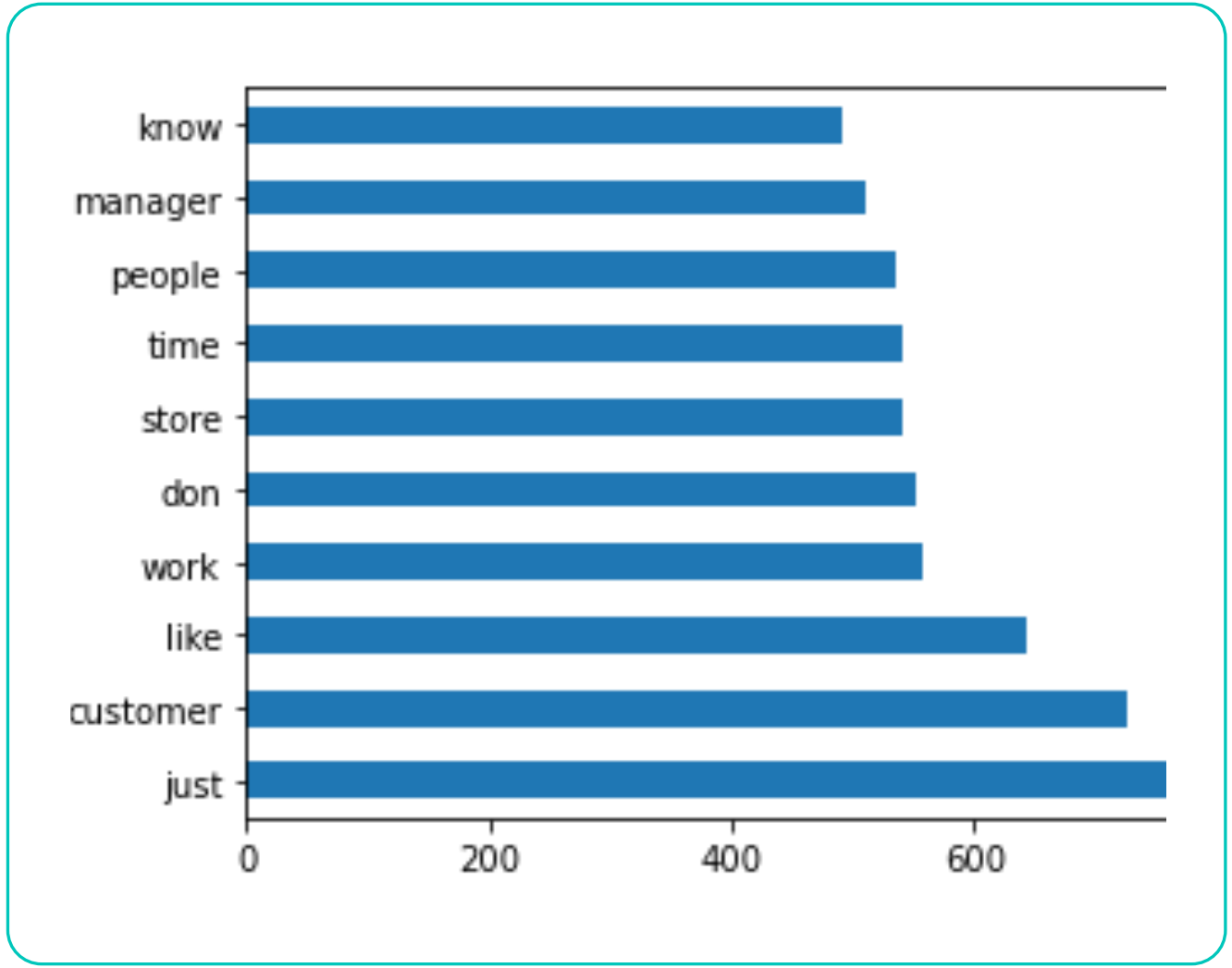
- Retail Author Rate: .70
- Server Author Rate: .65



NLP

- TFIDI:
  stop_words='english',
  strip_accents = 'ascii',
- Bad Differentiators:

10 most common words between both subreddits

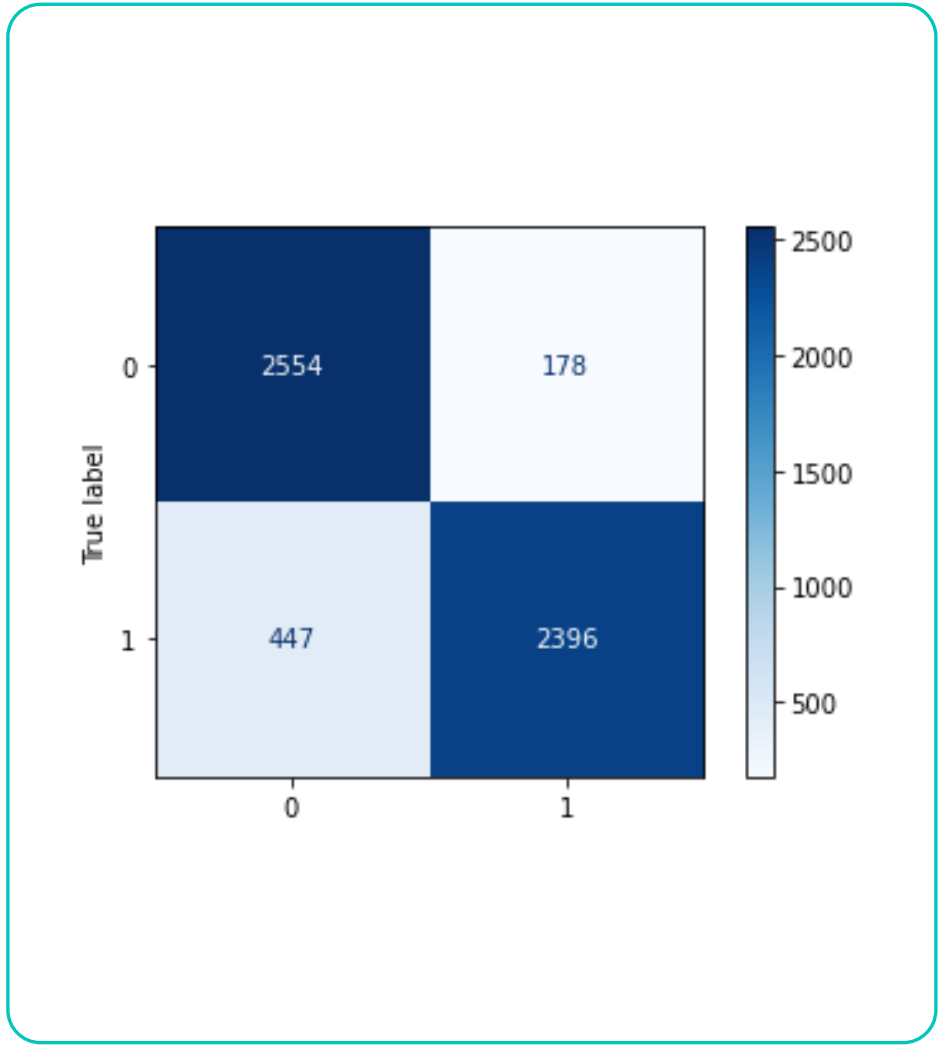


Version Name	Parameters	Train	Test
Log Reg 1	stop_words='english', strip_accents = 'ascii', l2	.9372	9104
Logistic Reg 2	stop_words='english', strip_accents = 'ascii' ngram = 2, min_df = .03, l2	.8913	.8644
KNN 1	With Scaling	.6253	.5397
KNN 2	No Scaling	.6370	.5580
KNN 3	With Scaling with_mean = False	.6253	.5397

Modeling

Confusion Matrix

- Recall Score: .8741%
- Precision: .9463% √
- Accuracy: .9104%



Interpreting coefficients

- Simple Nouns
- Exponentiated values
- We can say holding all else constant that including the word restaurant is 22,630 times more likely to be a post coming from a server.
- Not to mention “tip”, “bar”

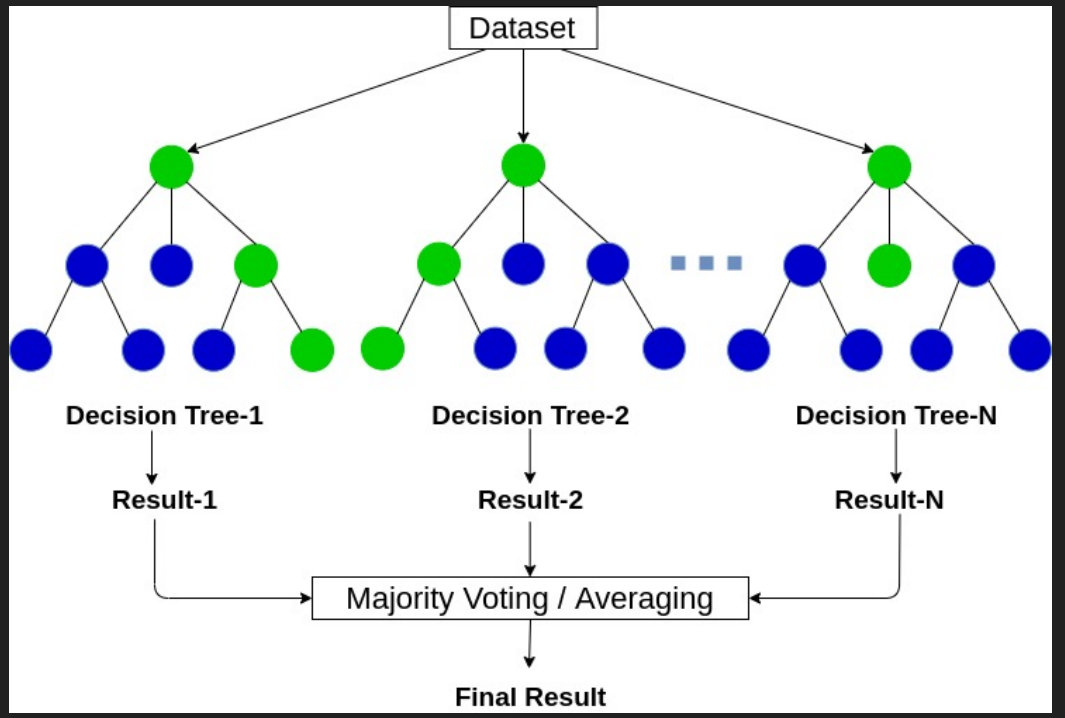
27500	restaurant	22360.267641
29158	server	12785.183319
32438	table	5027.068066
29170	servers	983.481728
33449	tip	639.902982
3930	bar	295.969239

Recommendations

- Do not include servers into SEIU
- Based on the coefficients, it seems like the server experience is myopic, conceited, opportunist, and possibly alcoholic.
- This does not align with the core values of industrious retail workers and could be a liability to our cause.

Project Future

- Finding the correct order
- Doing everything better:
- More NLP: porter-stemming, sentiment analysis
- Models: Native Bayesian, Random Forest
- Hyperparameter tuning: C, K
- Hyperparameter tuning: RandomSearchCV
- Metrics: F1_score



Thank you!

- Gwen Rathgeber
- Heather Robbins
- Dr. Anderson Prewitt
- TrainTestBritt