Final Presentation

Anna - Emre - Francis - Irina

About Our Project

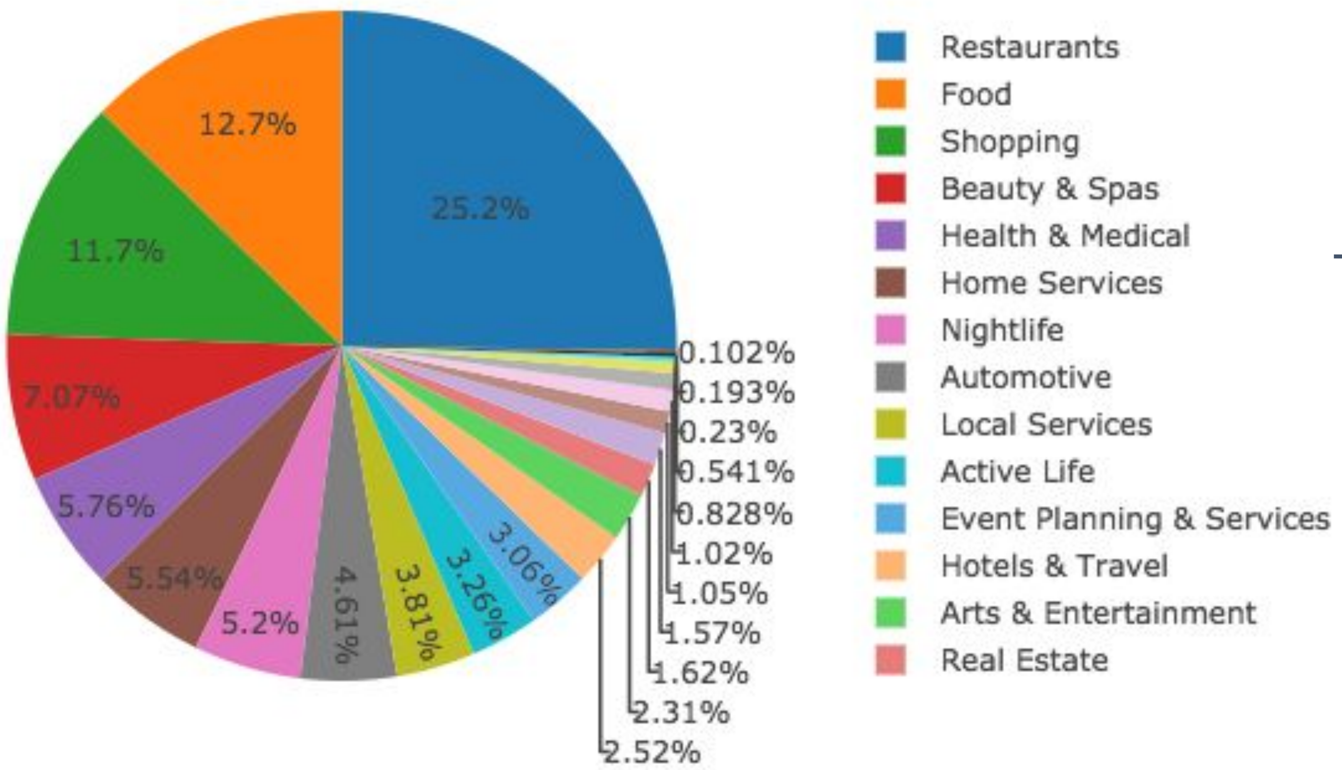
yelp

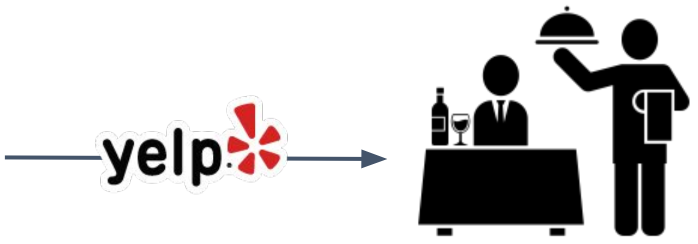
Big Dataset

Analyze **Reviews** and visualize the **Findings**

Apply machine learning to **Generate Insights** for businesses

Target Group

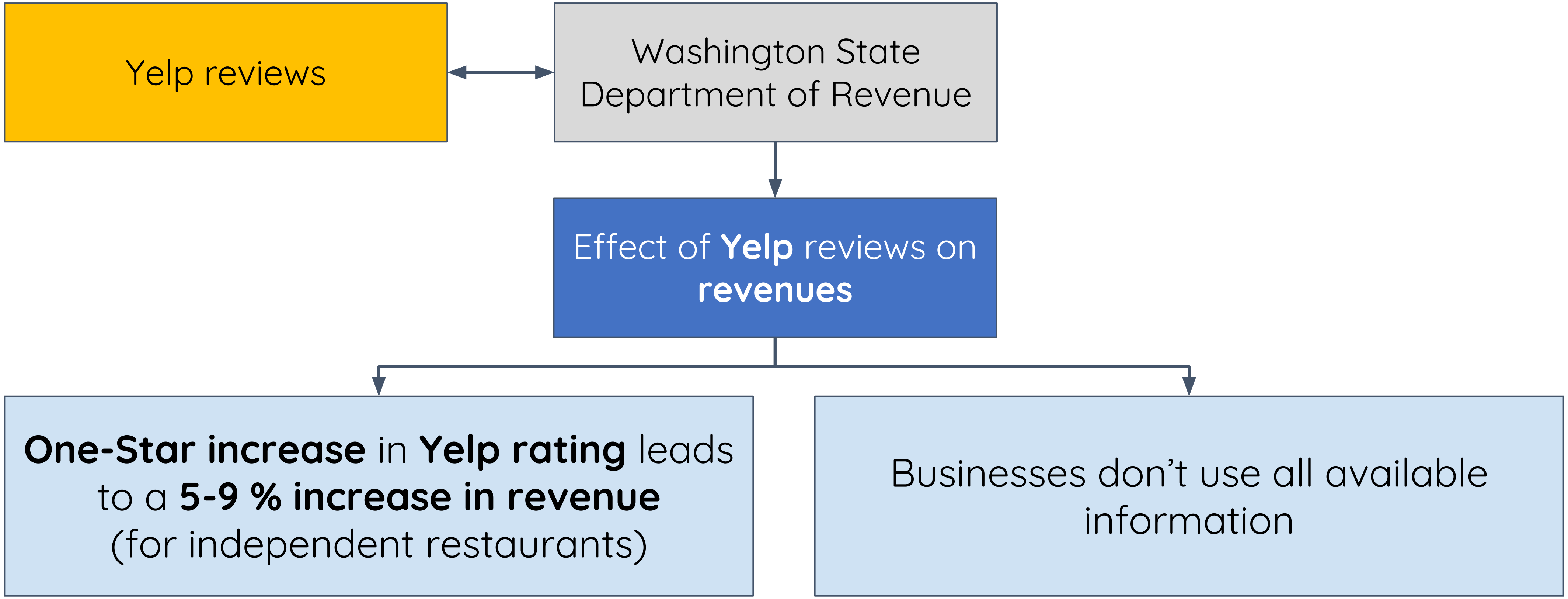




Independent restaurants

Source: *Harvard Business School https://www.hbs.edu/faculty/Pages/item.aspx?num=41233*

Impact of Yelp Reviews on Businesses



Source: *Harvard Business School https://www.hbs.edu/faculty/Pages/item.aspx?num=41233*

Scope: Overview of Main Results

Offer good food and service to the customers

Maximize profits

Improve company image

Identify fake reviews

Review rating prediction

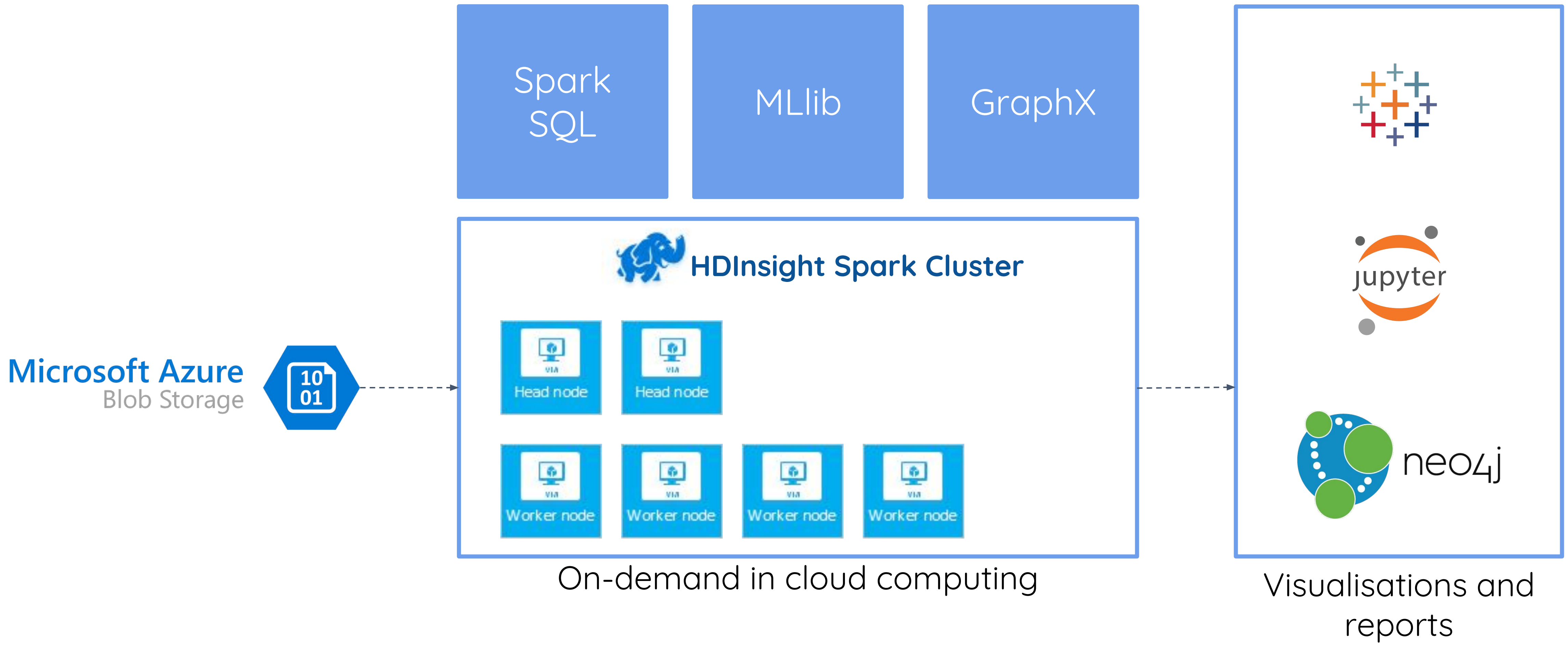
Find your competitors on a map

Find most important users to your business

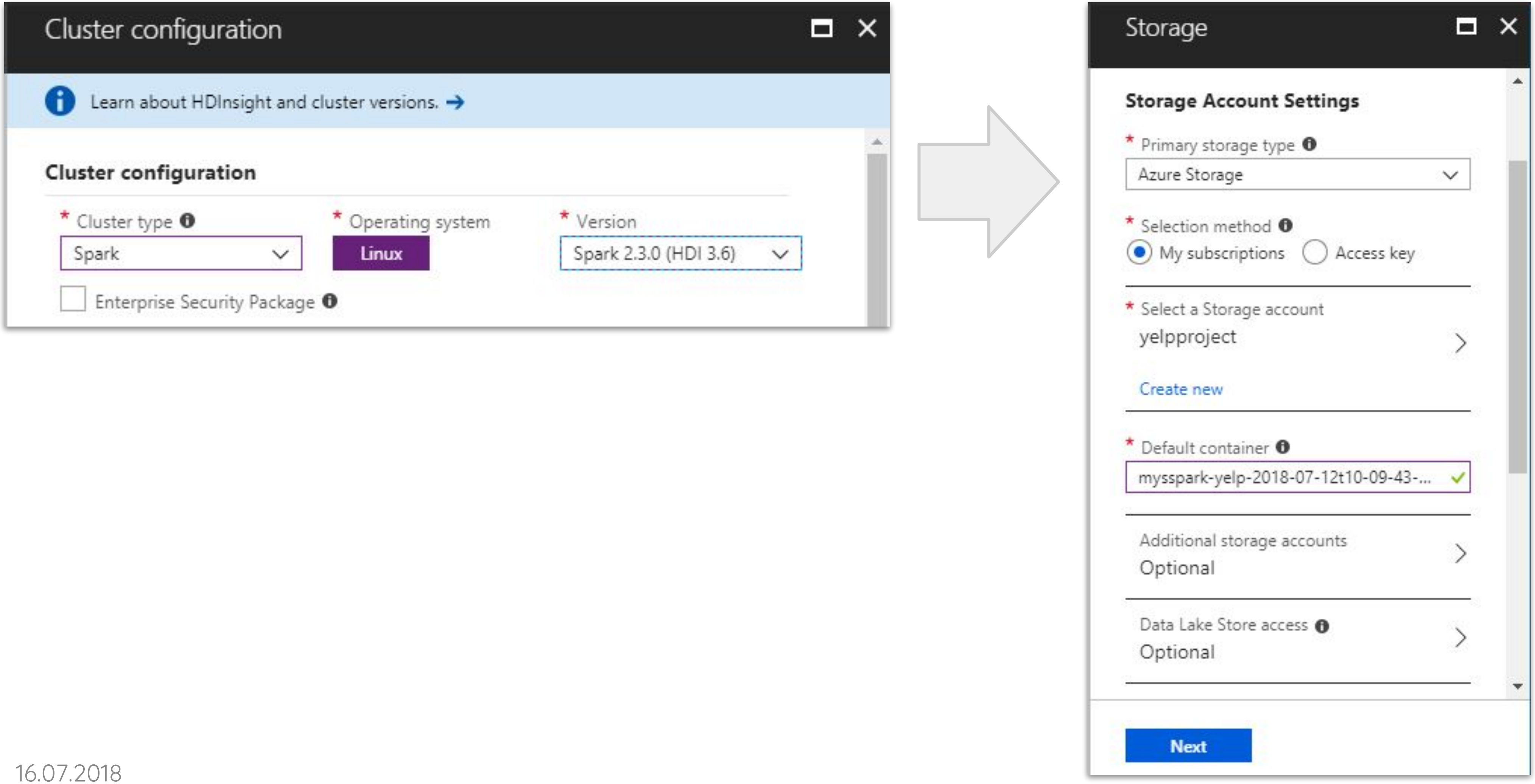
Sentiment Analysis & Topic Modeling

Increase revenues

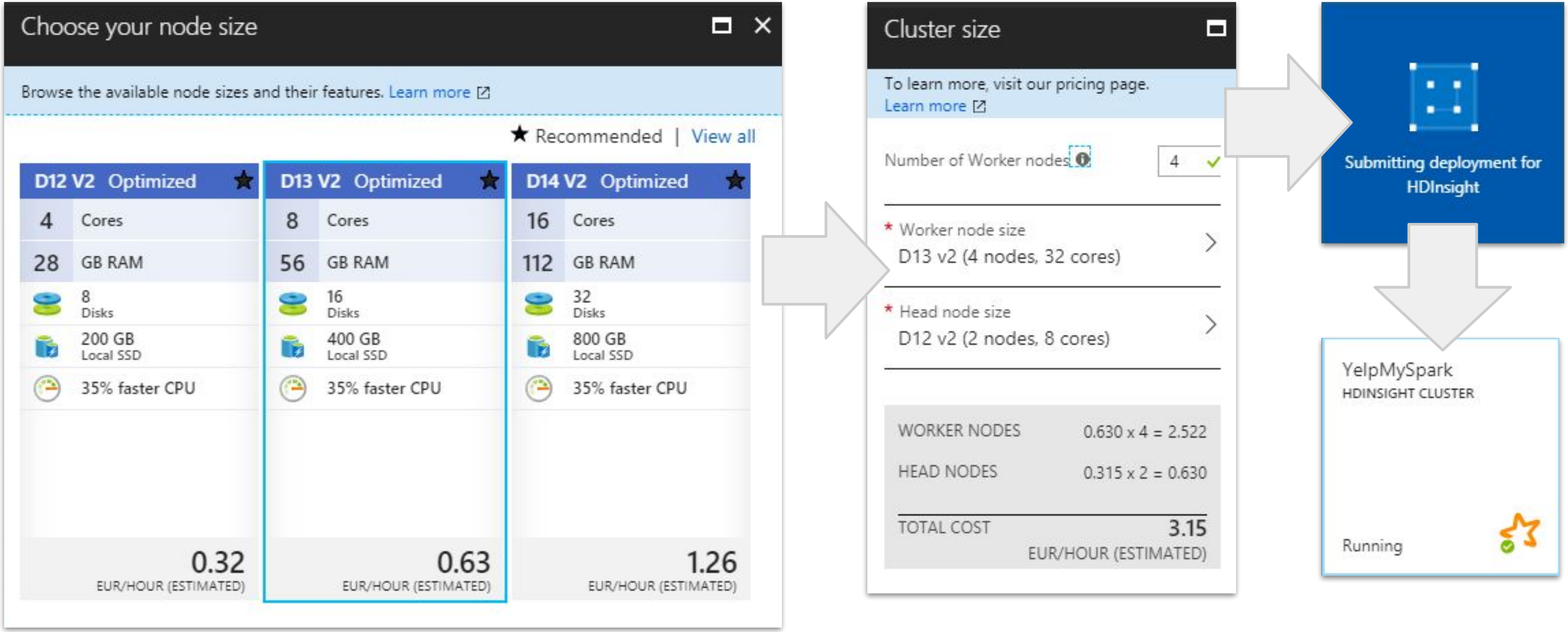
Working Environment I: Microsoft Azure



Cluster Configuration: Type & Storage



Cluster Configuration: Size & Pricing



Working Environment II: Databricks

Spark SQL

MLlib

databricks

Databricks file system - **DBFS**

Clusters

Databricks Cluster

Databricks Runtime Version

4.1 (includes Apache Spark 2.3.0, Scala 2.11)

Python Version

3

Driver Type

Community Optimized 6.0 GB Memory, 0.88 Cores, 1 DBU

On-demand in cloud computing

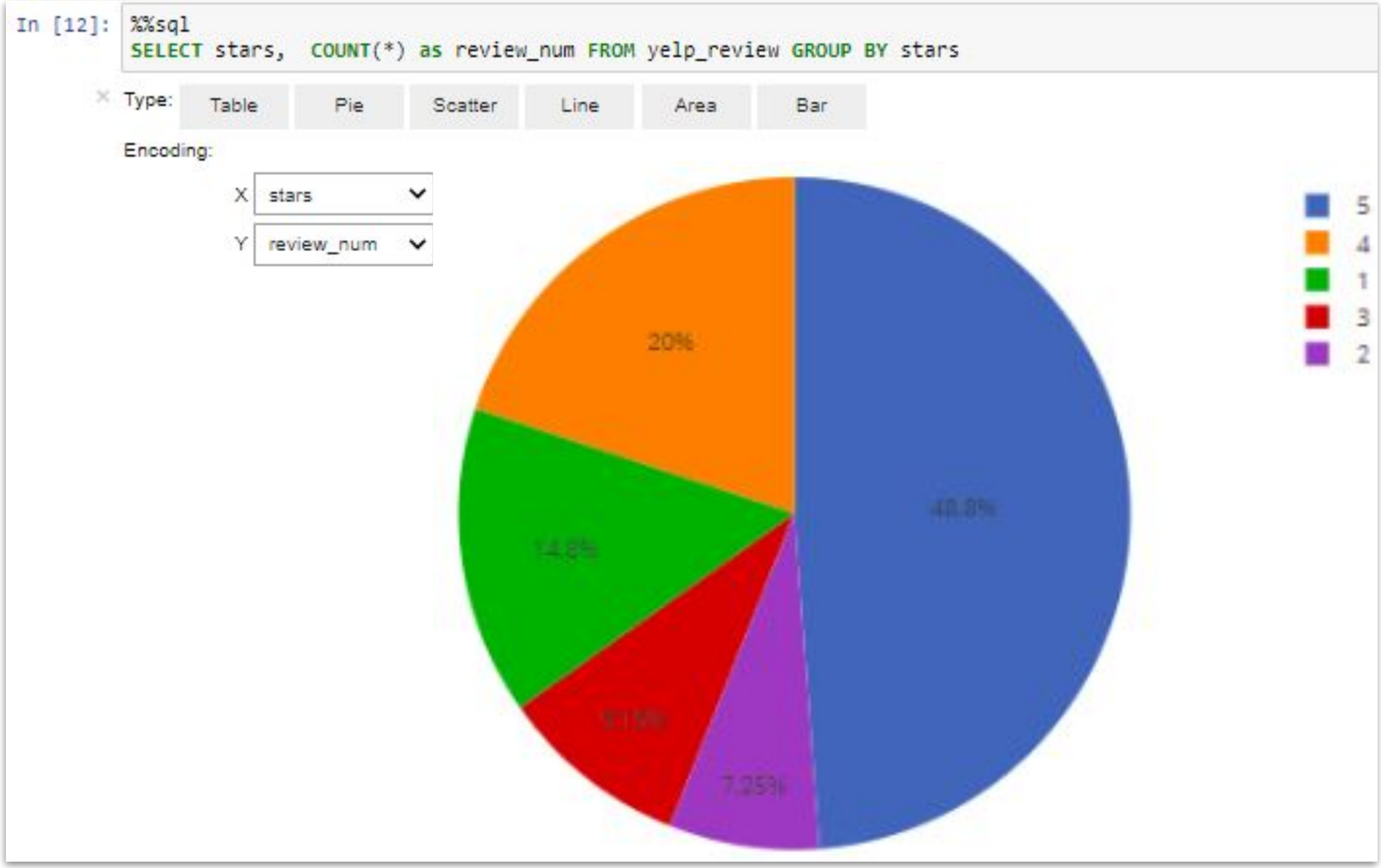
Workspace

Databricks Notebook

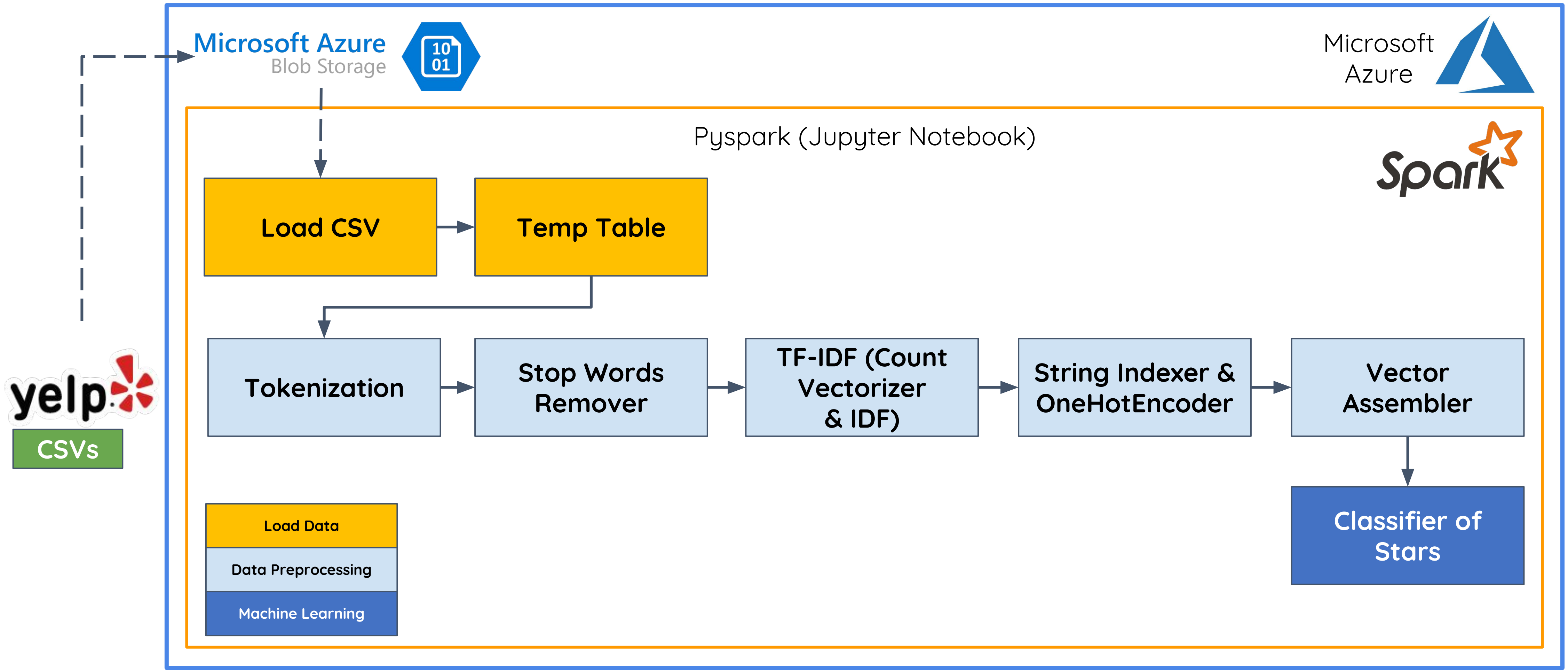
Visualisations and reports

Star rating prediction 

Star rating distribution: Spark SQL



Data Pipeline: Classifiers



Review Rating Prediction: Business Value

Feedback cards in a restaurant

Company's website

YELP reviews

Google reviews

Single data storage including all review sources

Applying predictive model to assign a Star rating

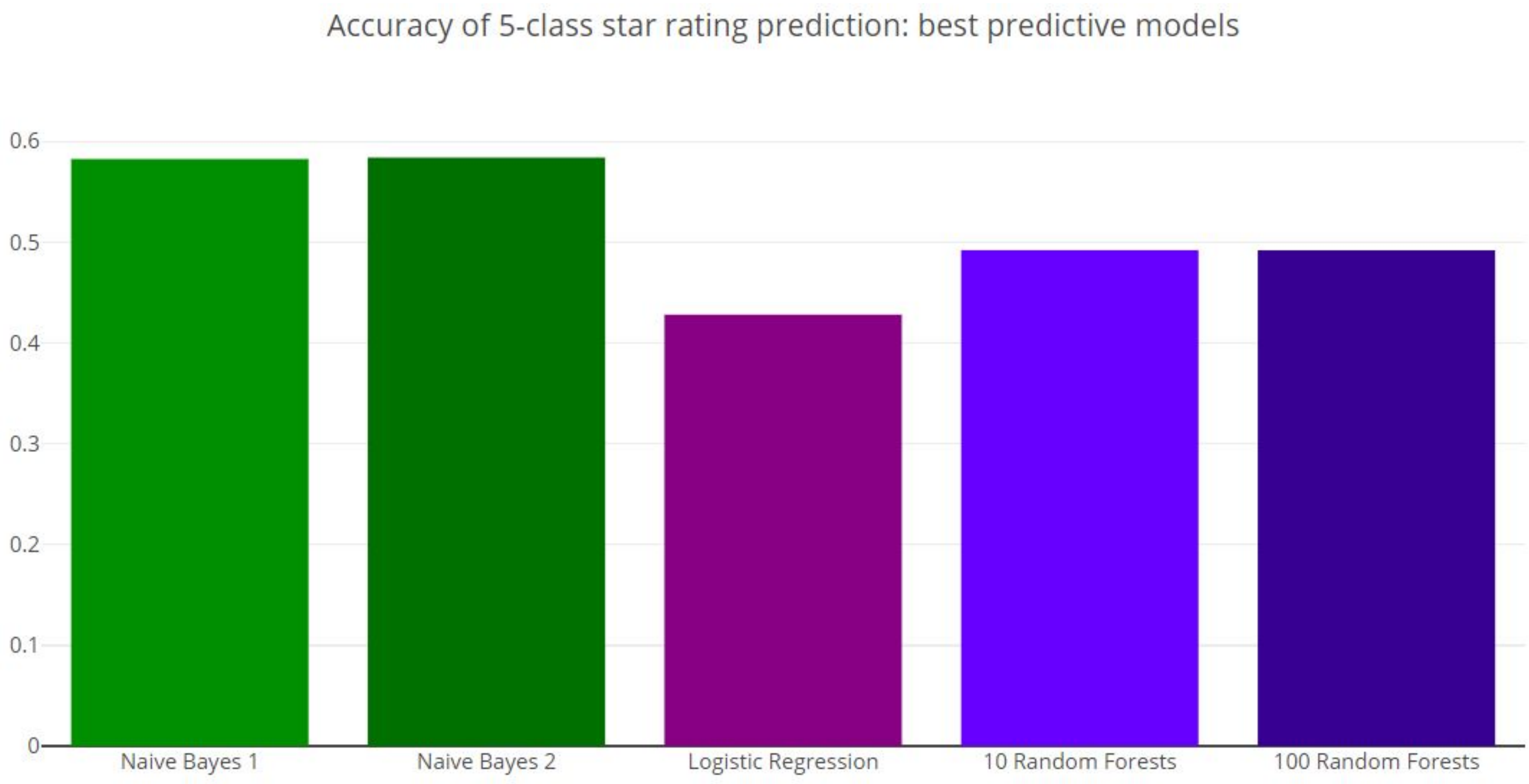
Evaluation

Naive Bayes - the best model

```
tokenizer = Tokenizer(inputCol="text", outputCol="token_text")
stopremove = StopWordsRemover(inputCol='token_text',outputCol='stop_tokens')
count_vec = CountVectorizer(inputCol='stop_tokens',outputCol='c_vec')
idf = IDF(inputCol="c_vec", outputCol="tf_idf")
stars_to_label = StringIndexer(inputCol='stars', outputCol='label')
feature_vector = VectorAssembler(inputCols=['tf_idf','text_length', 'useful', 'funny', 'cool'], outputCol='features')

nb = NaiveBaye
# smooting = 2

# Create a pipeline
data_prep_pipe = Pipeline(stages=[stars_to_label, tokenizer, stopremove, count_vec, idf, feature_vector])
cleaner = data_prep_pipe.fit(review)
clean_data = cleaner.transform(review)
clean_data = clean_data.select(['label','features'])

(train, test) = clean_data.randomSplit([0.7,0.3], 1234)
stars_predictor = nb.fit(train)
predictions = stars_predictor.transform(test)

from pyspark.ml.evaluation import MulticlassClassificationEvaluator
acc_eval = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
acc_nb = acc_eval.evaluate(predictions)
print("Accuracy of the model at predicting stars rating is: {}".forma

Accuracy of the model at predicting stars rating is: 0.58379025299897
```

```
NaiveBayes(smoothing=2.0, modelType="multinomial")
```

```
0.5837902529989701
```

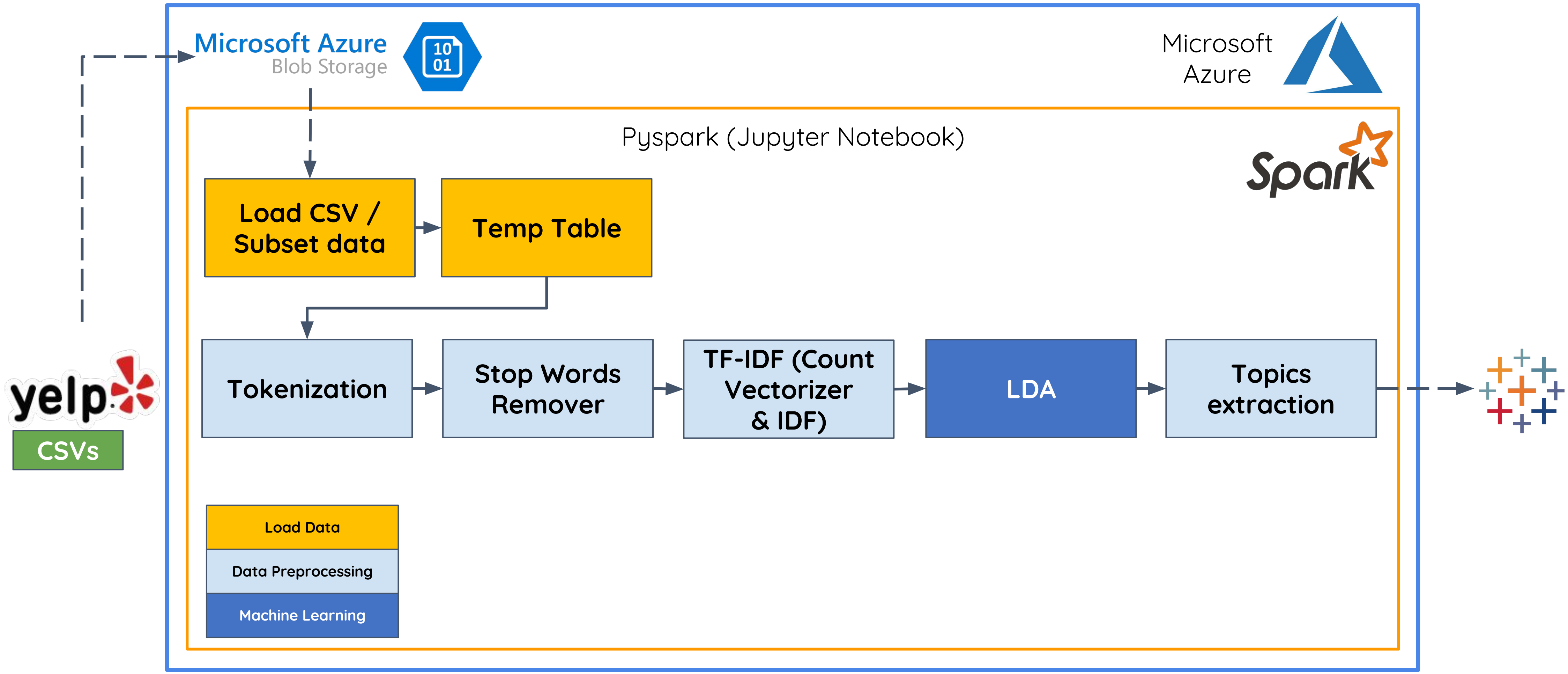

Accuracy of 5-class star rating prediction: best predictive models
0.6
0.5
0.4
0.3
0.2
0.1
0
Naive Bayes 1
Naive Bayes 2
Logistic Regression
10 Random Forests
100 Random Forests

Best Predictive Model

Naive Bayes classifier

Customer preferences via LDA clustering 🍽️

Data Pipeline: LDA



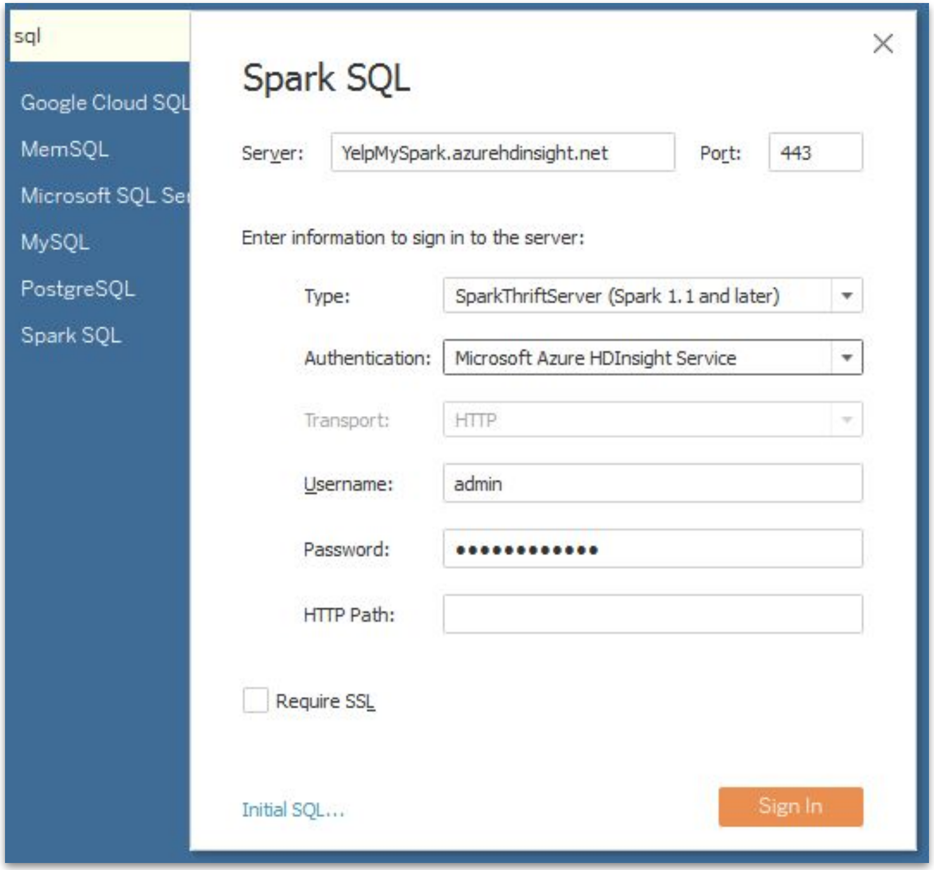
Implementation

```
# Generate 10 Topics:
lda = LDA(k=10, seed=1234,  optimizer='online', featuresCol="features")
ldamodel = lda.fit(rescaledData)
```

```
In [316]: pd.options.display.max_rows=1000
          print(cities_df)
```

```
         City     Term  Topic    Weight
0     Toronto    ramen      1  0.012112
1     Toronto    wings      1  0.007514
2     Toronto    great      1  0.006947
3     Toronto     beer      1  0.006540
4     Toronto    steak      1  0.005970
5     Toronto      tea      2  0.010772
6     Toronto    store      2  0.006031
7     Toronto      get      2  0.005988
8     Toronto    place      2  0.005309
9     Toronto     like      2  0.005053
10    Toronto    pizza      3  0.015631
11    Toronto     fish      3  0.010228
12    Toronto    tacos      3  0.007007
13    Toronto     good      3  0.006363
14    Toronto    vegan      3  0.006210
15    Toronto    sushi      4  0.009223
16    Toronto    great      4  0.007776
17    Toronto     food      4  0.007011
18    Toronto  service      4  0.006892
```

Tableau Configuration



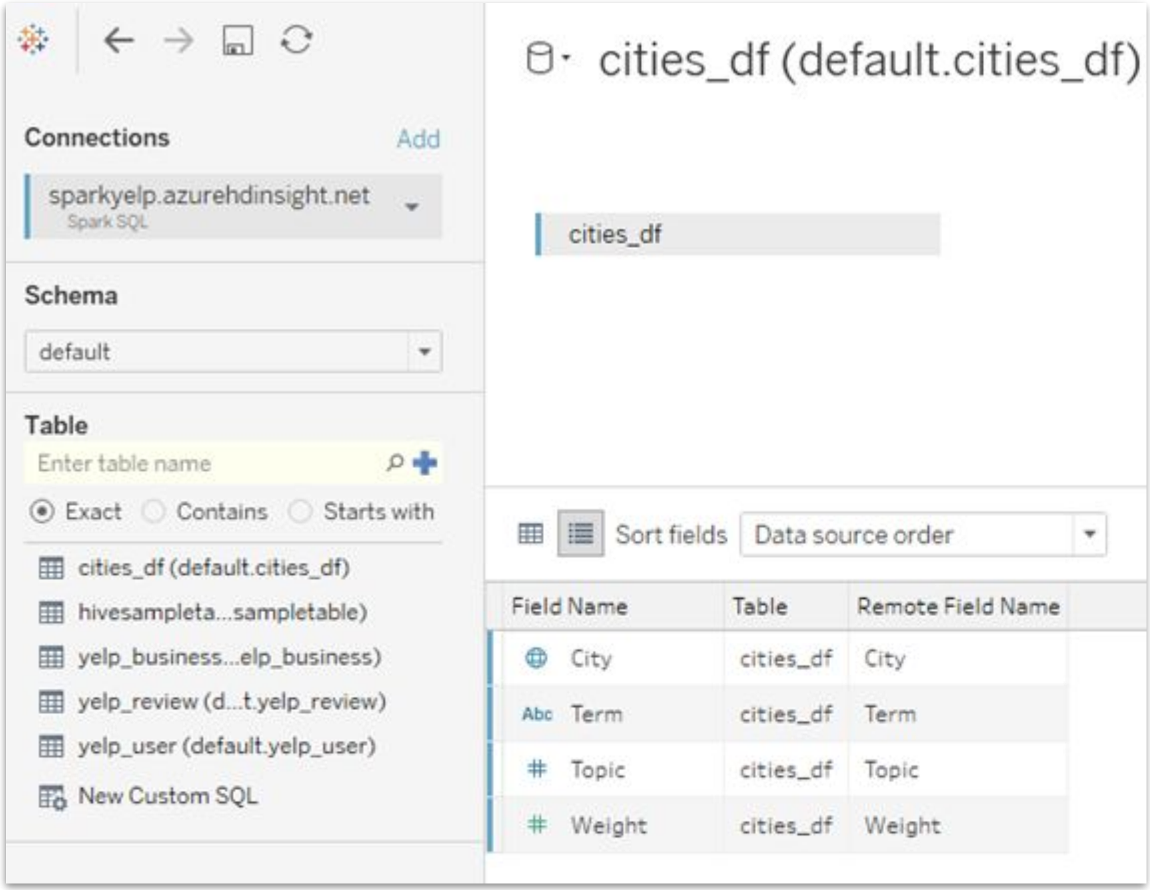
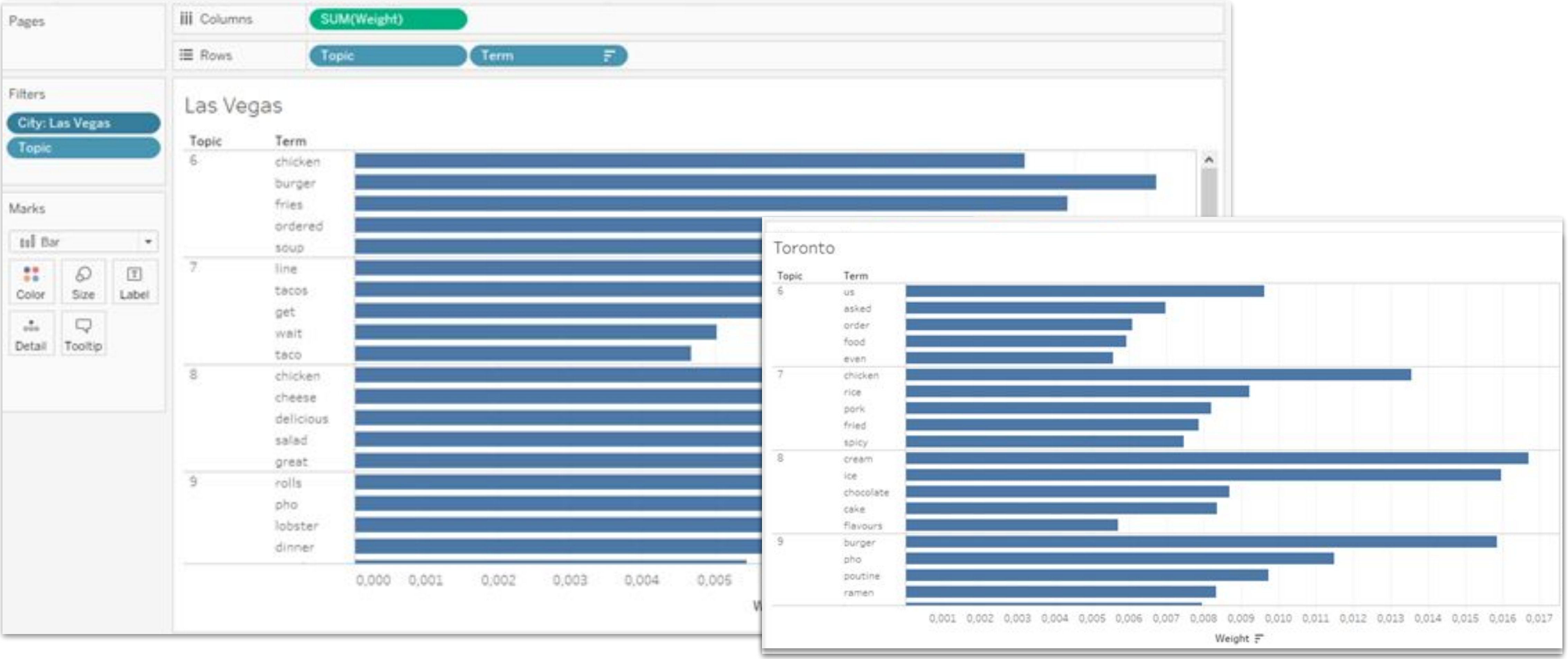
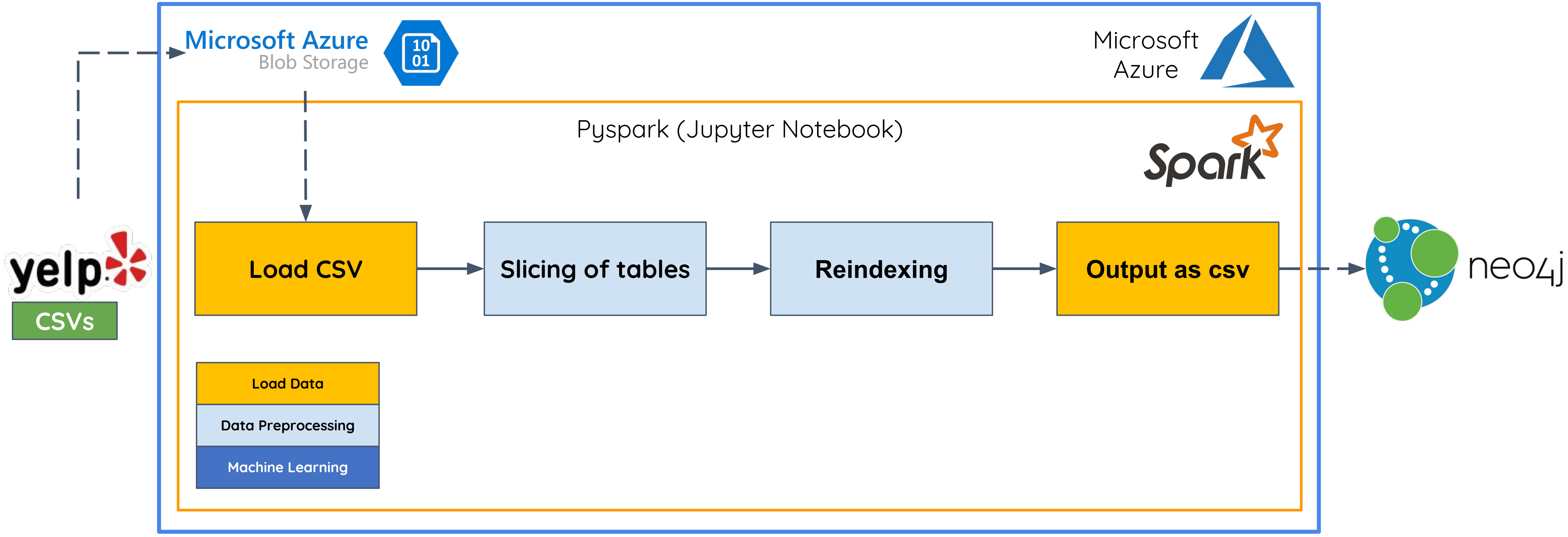


Tableau visualizations



Filtering fake reviews

Data Pipeline: Graph



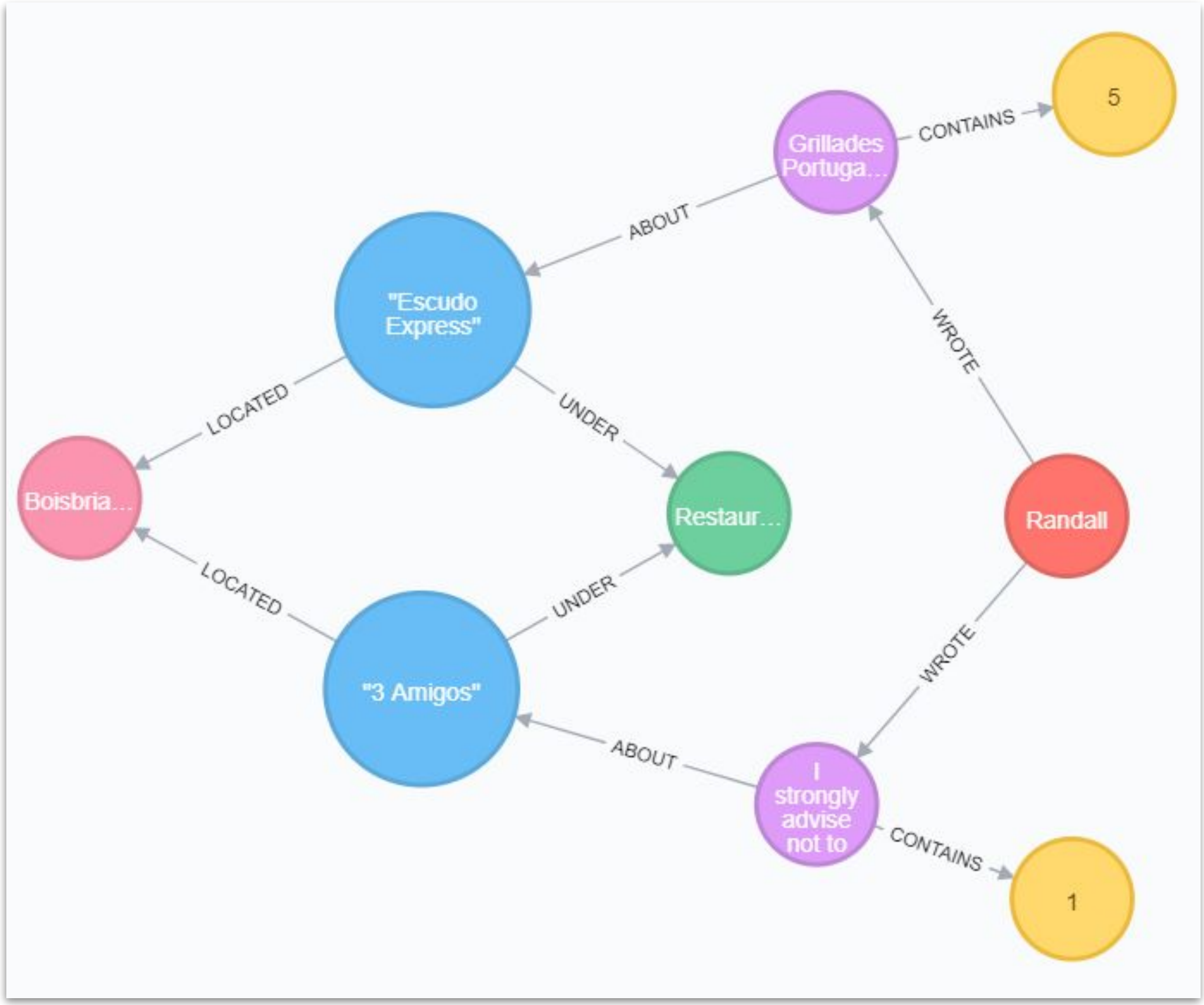
What percentage of Yelp reviews are **fake**?

According to
the Harvard Business School,
around 20% of Yelp **reviews**
have been potentially
manipulated

Assumptions

Employees who work at competing businesses in the same area **will rate** their **own business** with a **5 star** review, and rate **competing businesses** with a **1 star** review

Potential fake reviews



The user with nickname **Randall** gave two extreme **comments** to two restaurants: one with **5 stars** & another with **1 star**

Two restaurants are in **same city** and **category**

Results

```
$ match (u:User)-[rW:WROTE]->(r:review)-[rA:ABOUT]->(b:Business), (u)-[rW1:WROTE]->(r1:review)-[rA1:ABOU…
```

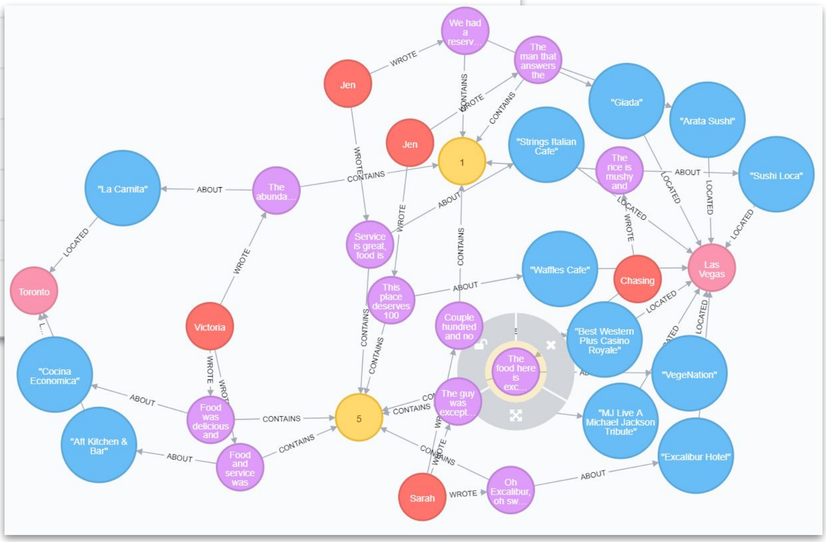
Table

Text

Code

u.user_id	u.name	u.review_count
"Kg6PUClchJGMUFjGvvTB0w"	"Jen"	7
"RcPnFSNnCUTAxxnurrPzqg"	"Jen"	9
"p8feOvowjtSSQXKX3du1Tg"	"Kim"	4
"jKLyOafO-KdzurHgeD6y3Q"	"J"	6
"uTK7i6UvzAvhZaJyLEFjJQ"	"Shayna"	6
"tLa80KOdezpUgcOQFdqtaA"	"Cynthia"	7
"QhchN0LAzZ00cJVEZFX_-g"	"Eric"	3
"J43y_5fgbc-JFRzPs2trqQ"	"Cesar"	3
"Yla-asW5l0lzdOfpwiuYXQ"	"Angel"	8
"fGS0d36wWtSTI2J2KILXig"	"Alysia"	5

Started streaming 10 records after 770 ms and completed after 933 ms.

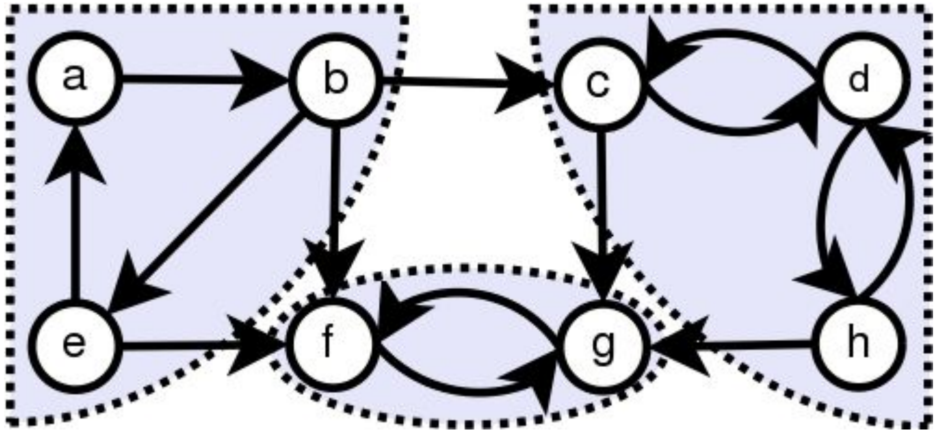


1,082 reviews written by **196 users** can be filtered

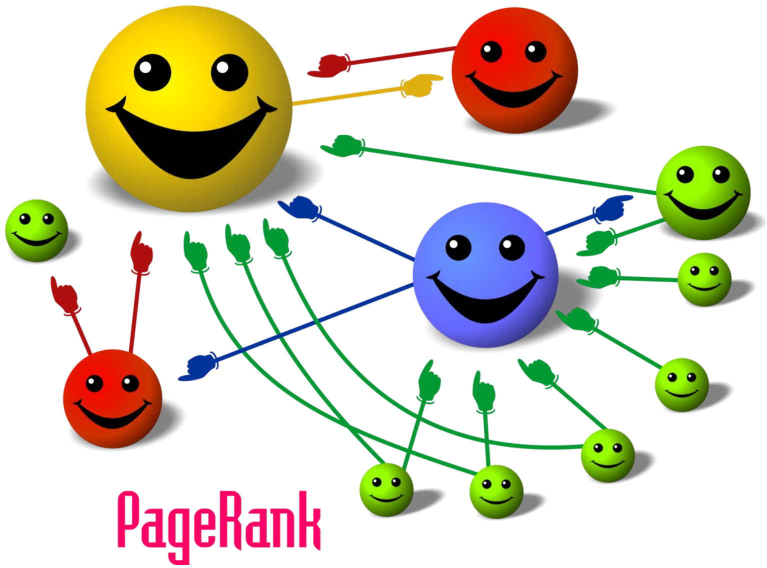
Identifying Influencers

Which **consumers** are the **most important** to your business?

Algorithms

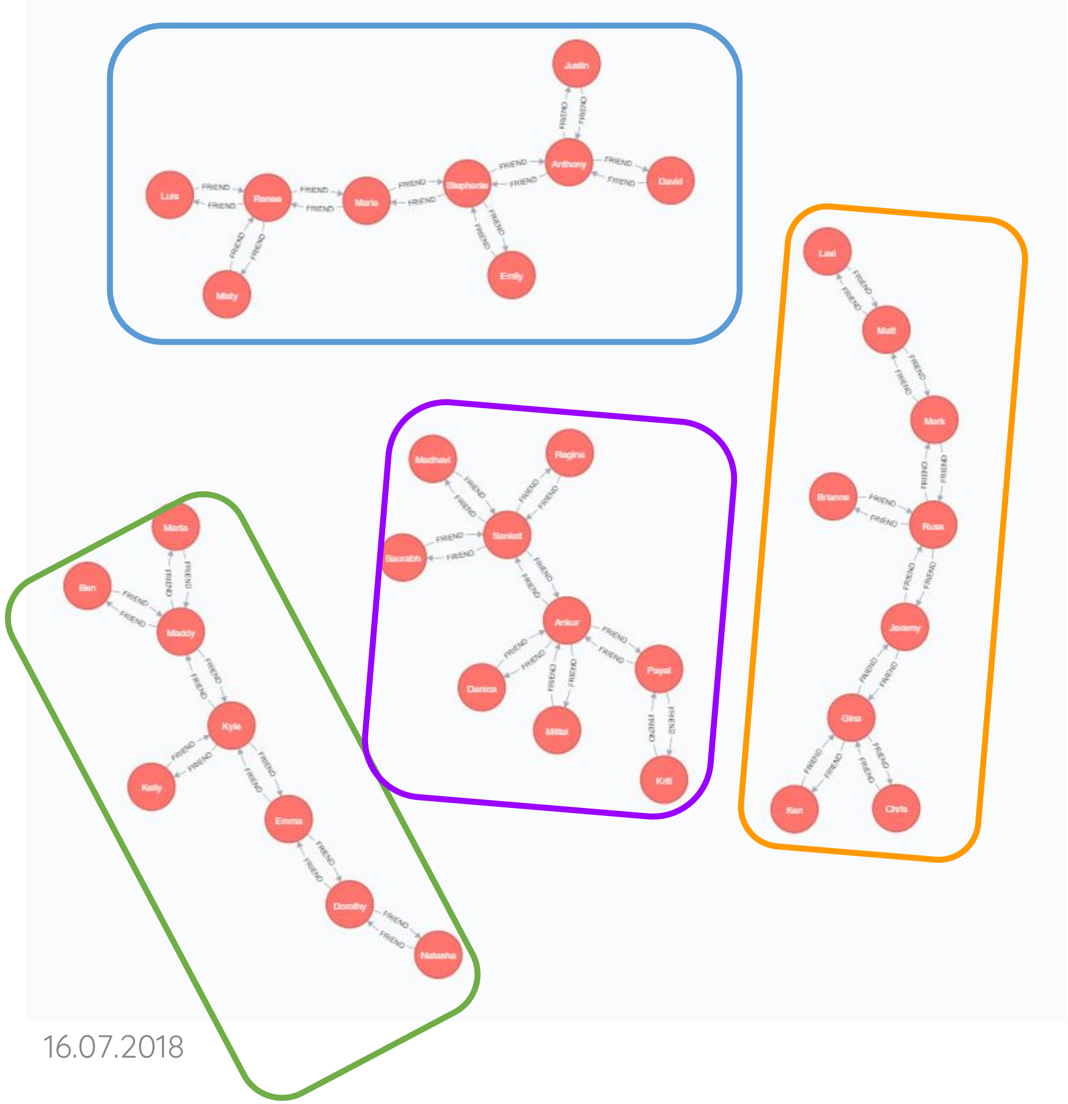


Strongly Connected Components



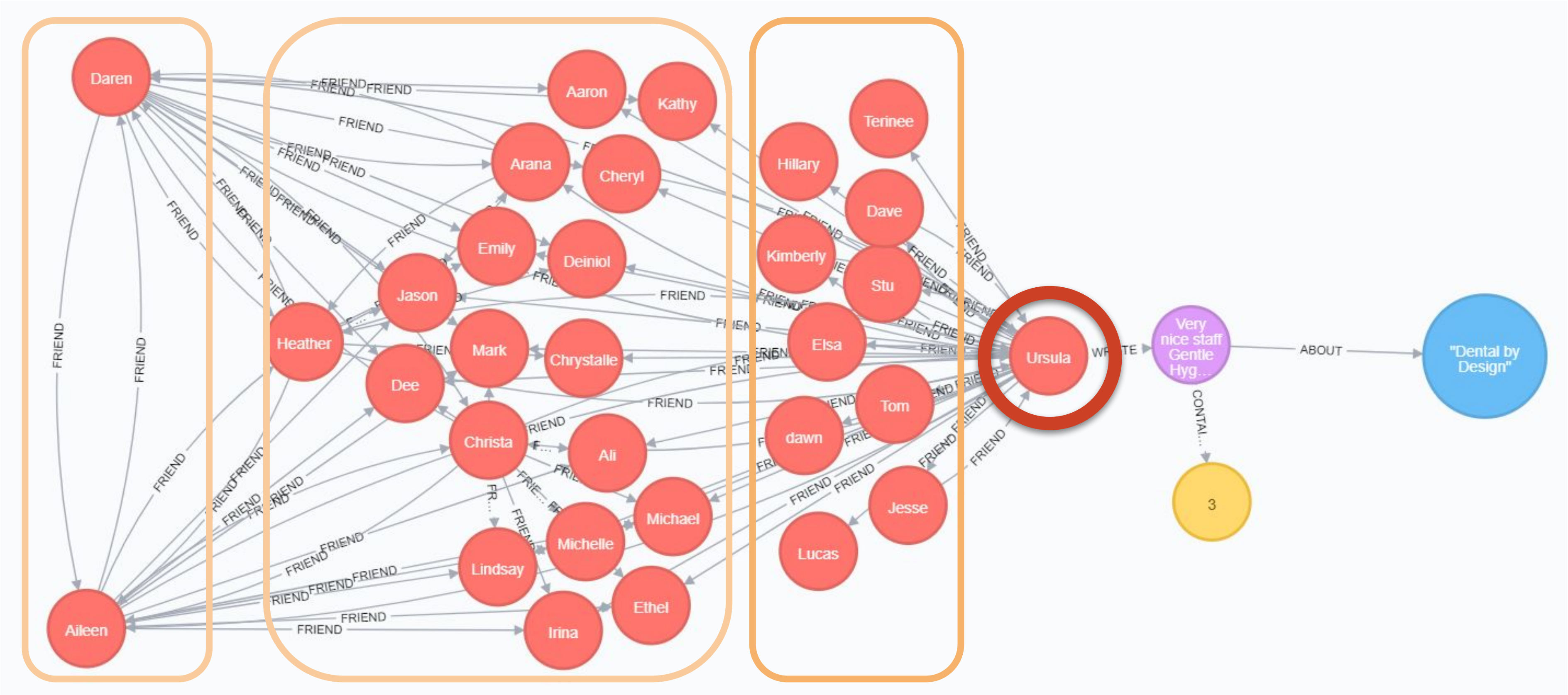
PageRank

Strongly Connected Components



- Strongly connected components is a graph algorithm that finds group of nodes
- In this project this algorithm was used to partition users into groups
- Small independent businesses can make use of the algorithm to identify important user groups

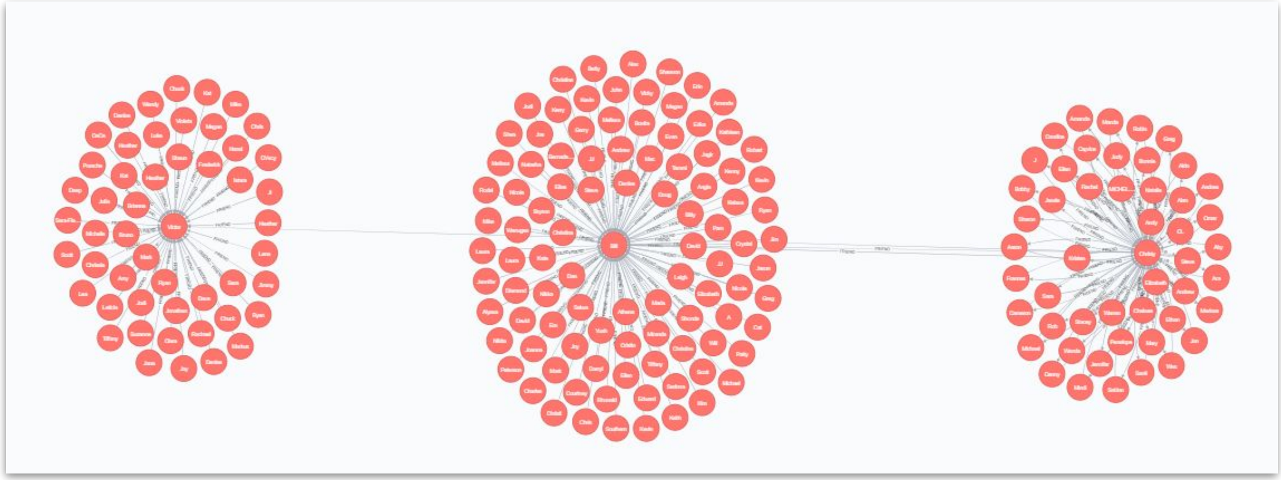
Strongly Connected Components - Example Result



Ursula can help the business to directly reach **10 users as a group**, and after these 10 users can reach **more users in other groups.**

PageRank

$$PR(A) = 1 - d + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right).$$



PageRank - Results

```
$ MATCH (n:User)-[r:WROTE]->(m:review)-[r1:ABOUT]->(b:Business), (m)-[r2:CONTAINS]->(s:Star), (n)-[r3:FR…
```

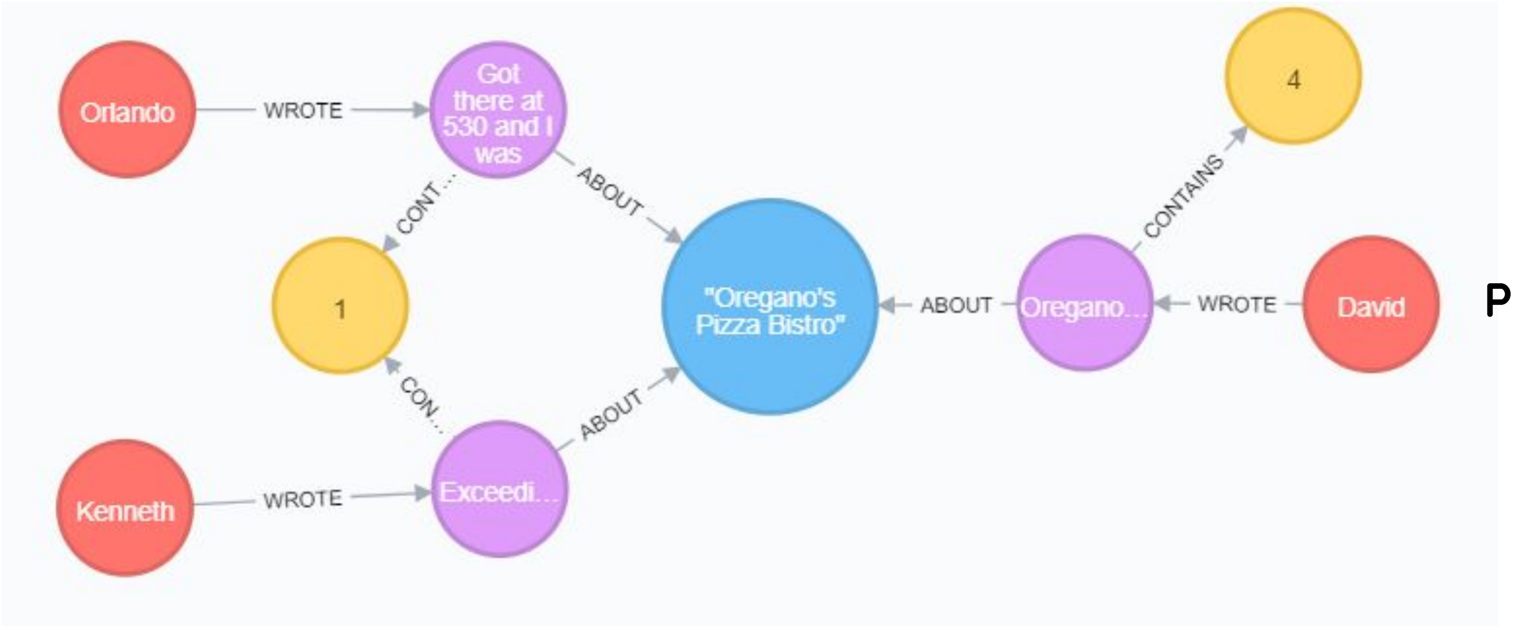
n.name	n.userpagerank	n.review_count	friendcount
"Deb"	2.0204675	384	820
"Nydia"	2.0137694999999995	400	1746
"Ken"	1.9980785000000003	724	249
"Michael"	1.3867840000000002	927	882
"Amy"	1.3749265000000004	357	3189
"Alicia"	1.201314	419	208

PageRank gives a different point of view on the user importance

PageRank - Results

PR: 0.43

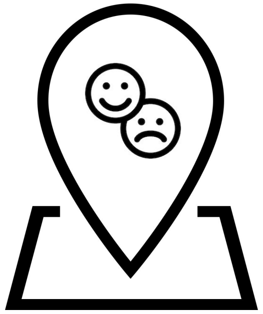
PR: 0.77



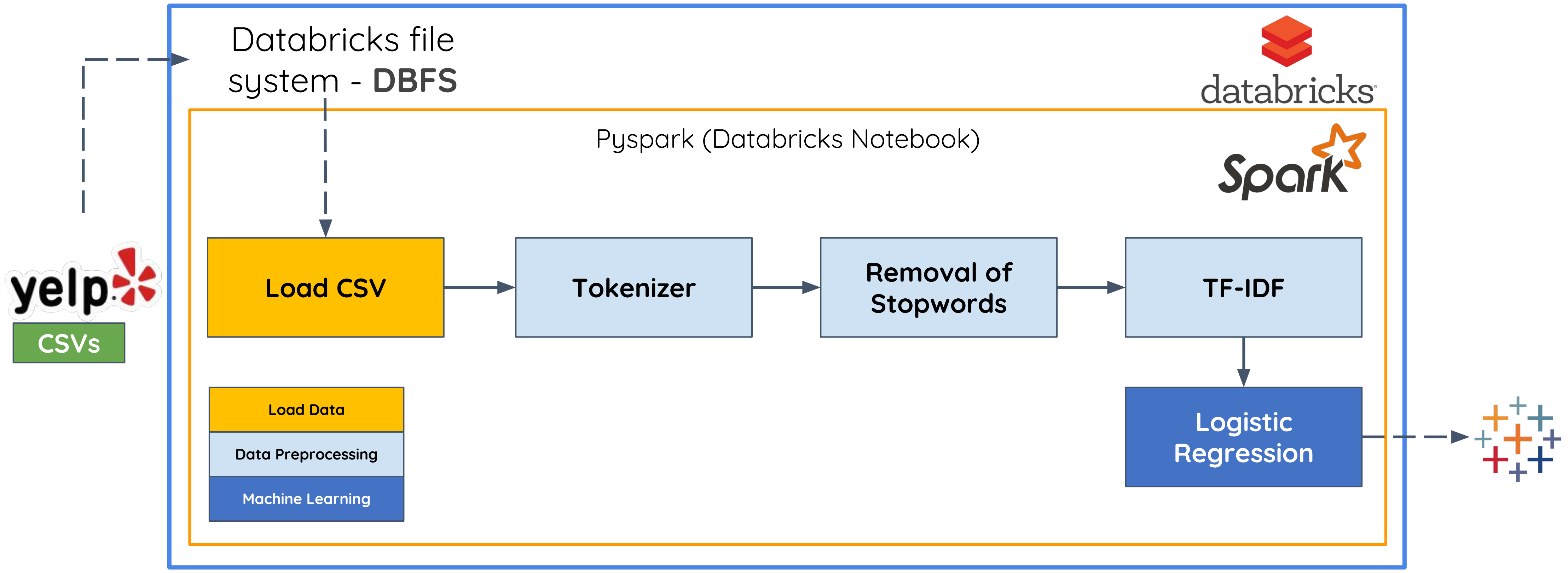
PR: 0.40

Users **Orlando** and **Kenneth** both **gave 1 star**.
If business wants to **improve** the **reputation**, then **to which user** business should **respond first** when resources are limited?

Sentiment map



Data Pipeline: Sentiment Analysis



Sentiment Analysis with Logistic Regression

Cmd 60

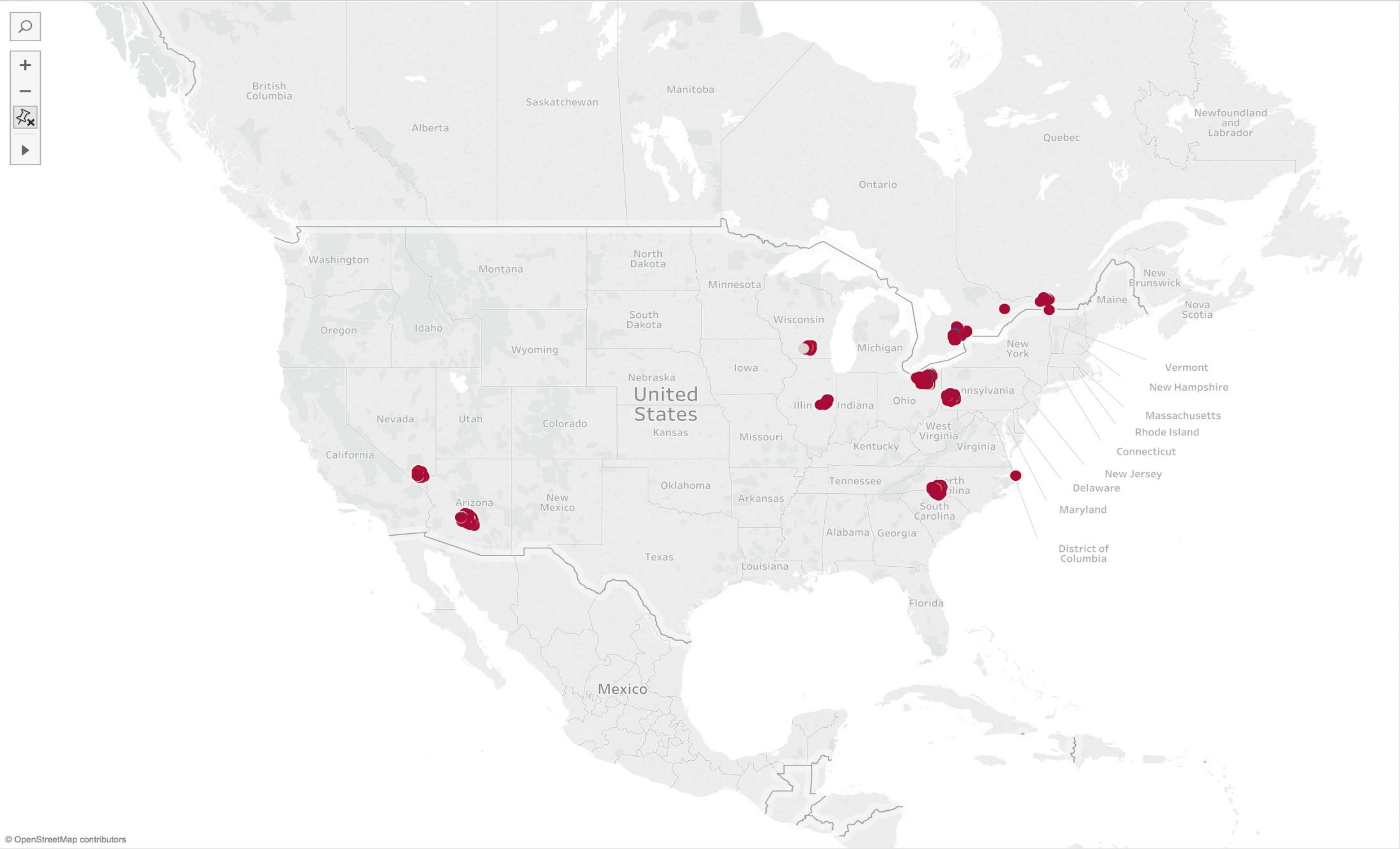
```
#elastic net regularization to avoid overfitting
lambda_par = 0.02
alpha_par = 0.3
en_lr = LogisticRegression().\
        setLabelCol('stars').\
        setFeaturesCol('tfidf').\
        setRegParam(lambda_par).\
        setMaxIter(100).\
        setElasticNetParam(alpha_par)
```

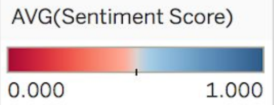
Cmd 74

```
#our best model has a accuracy of 88.9%
best_model = all_models[best_model_idx]
accuracies[best_model_idx]
```

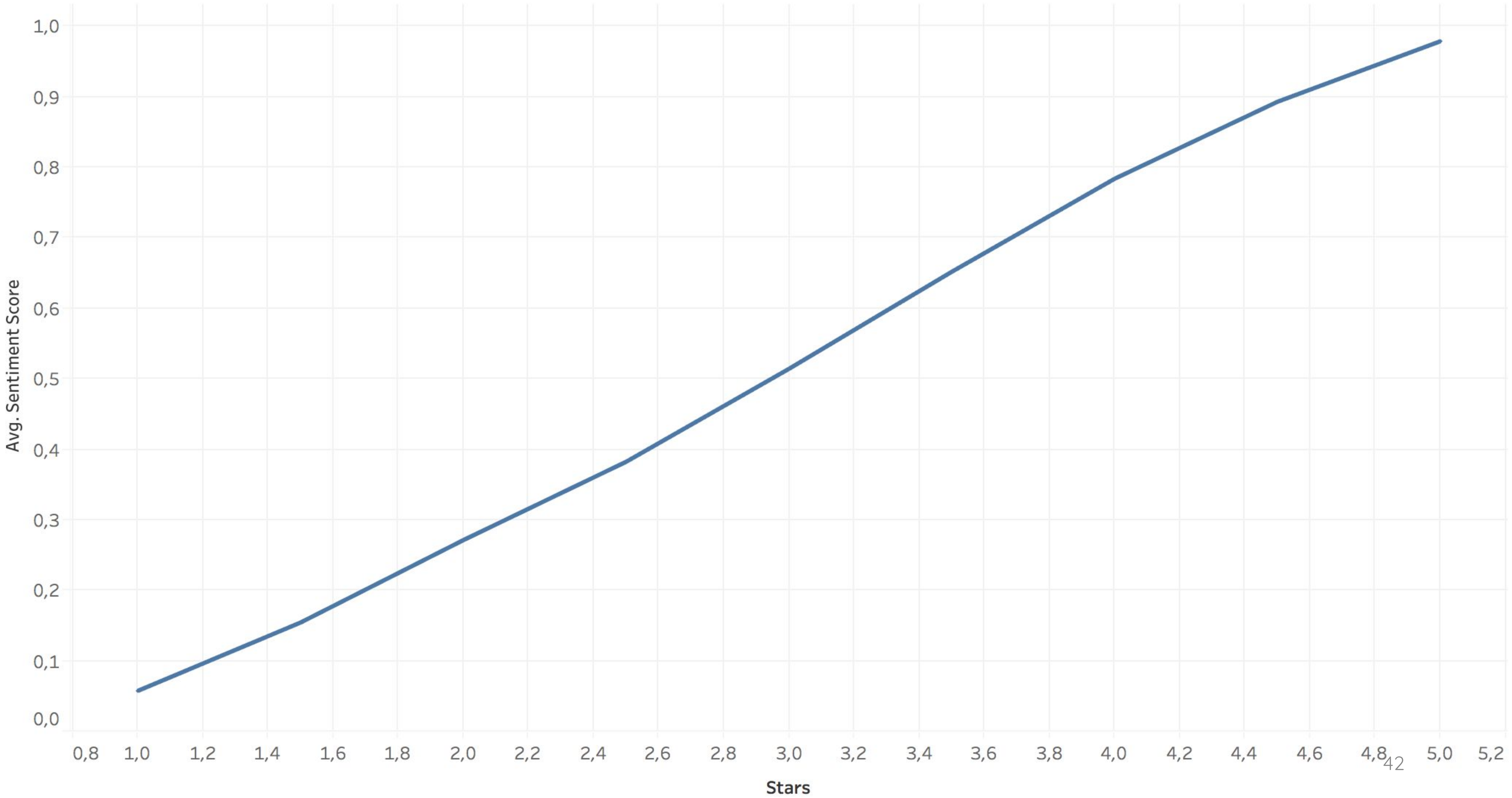
```
Out[129]: 0.8894220196325396
```

Average Sentiment when stars=1

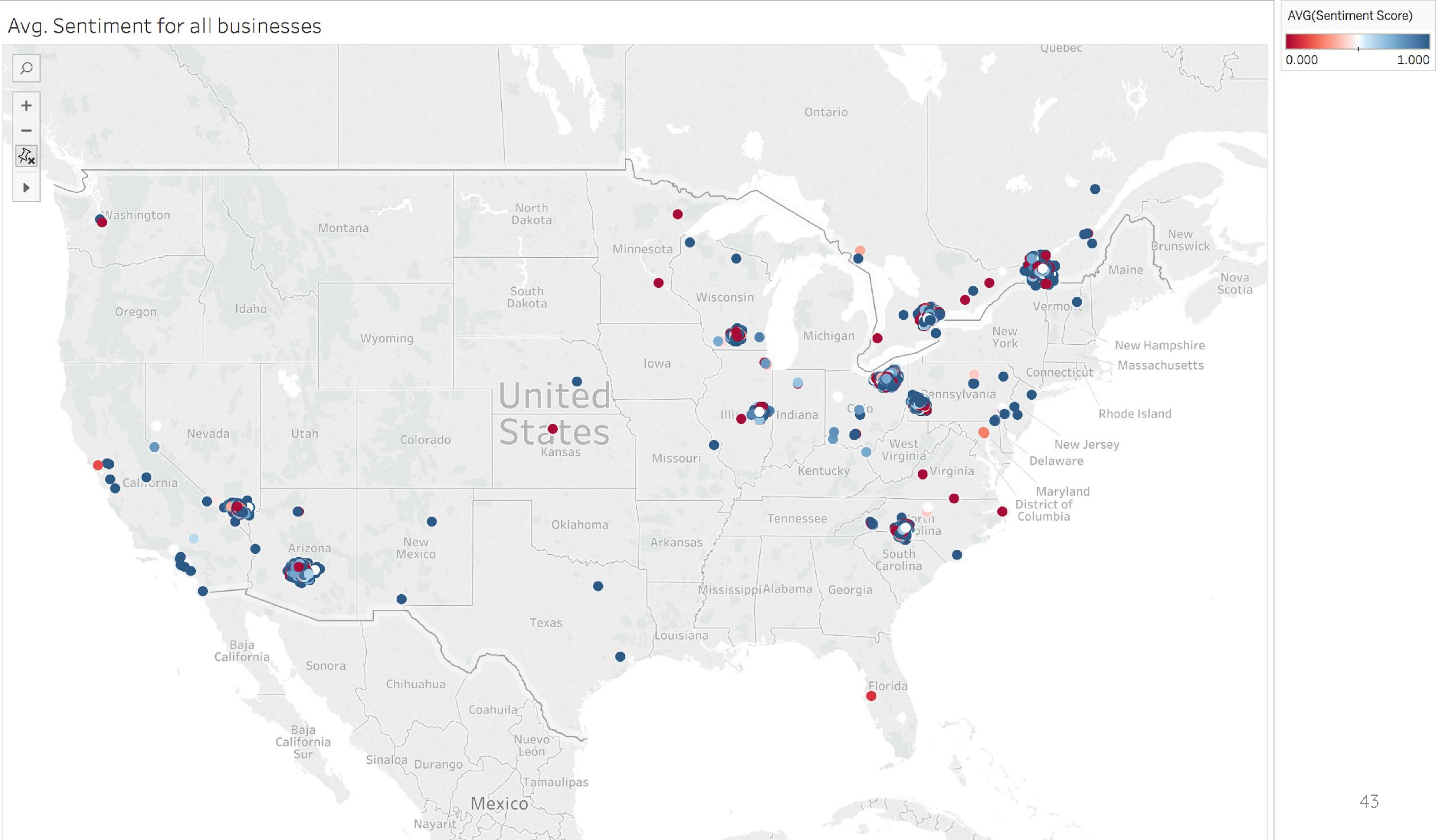




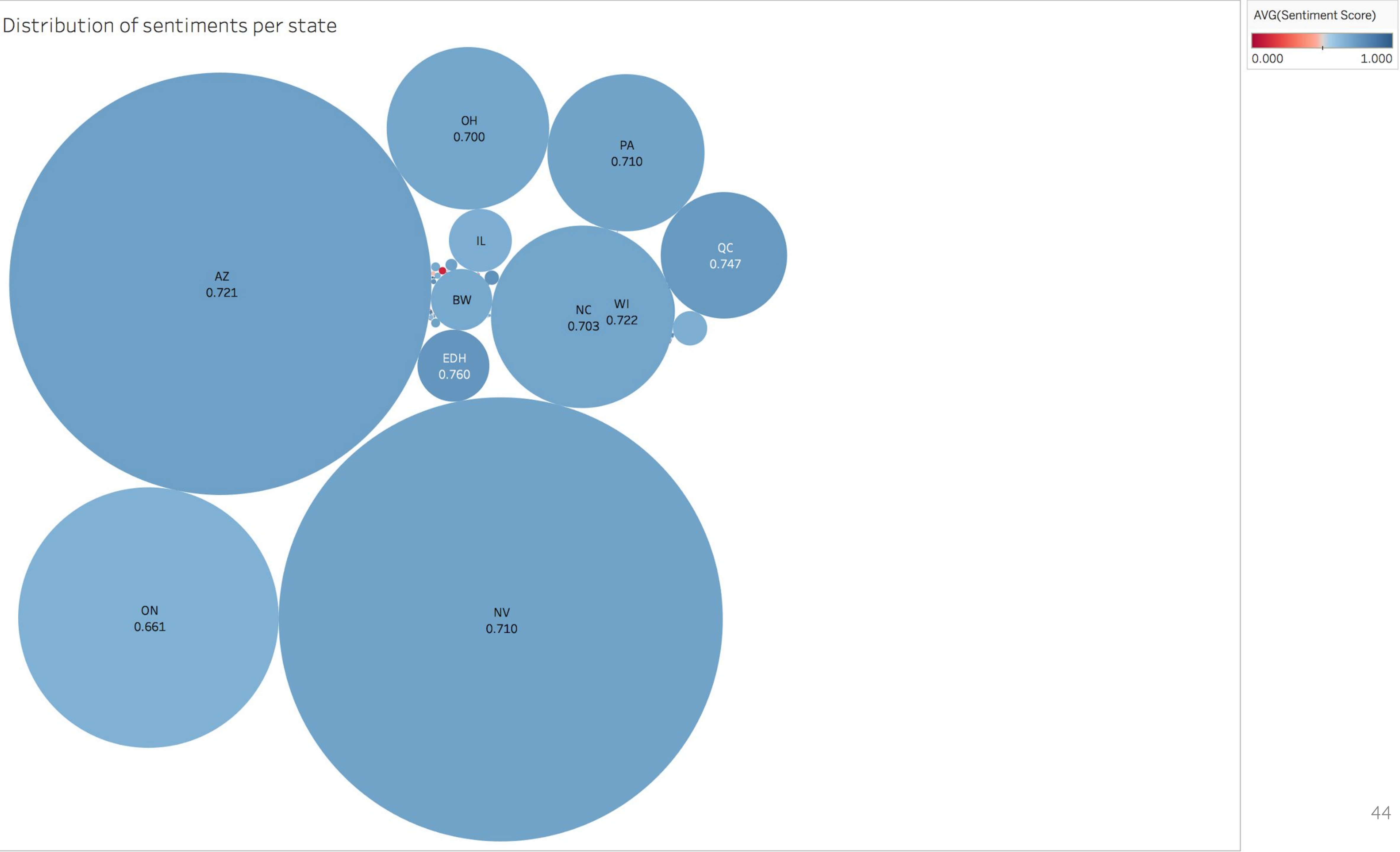
Average Sentiment per stars



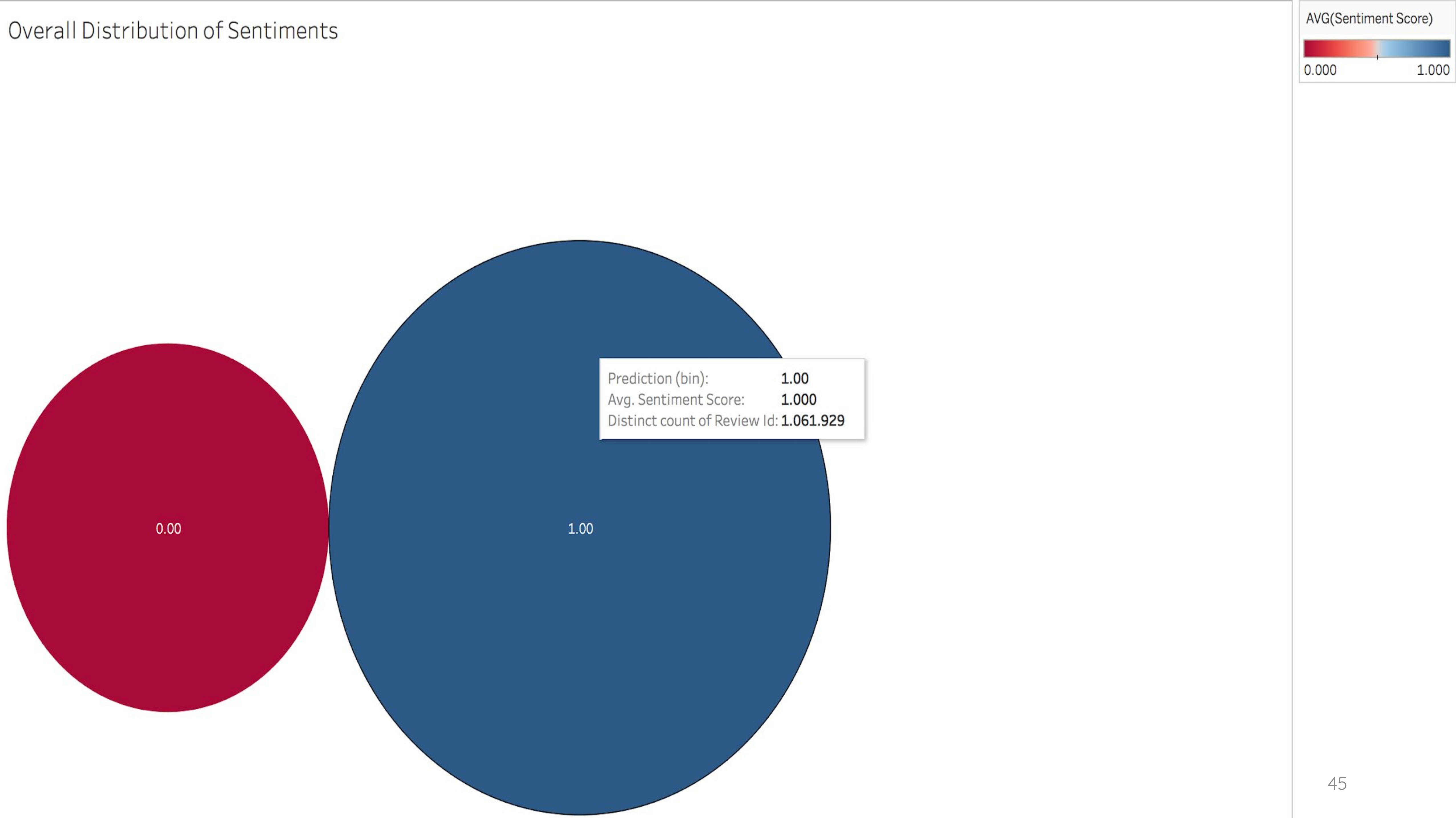
Avg. Sentiment for all businesses



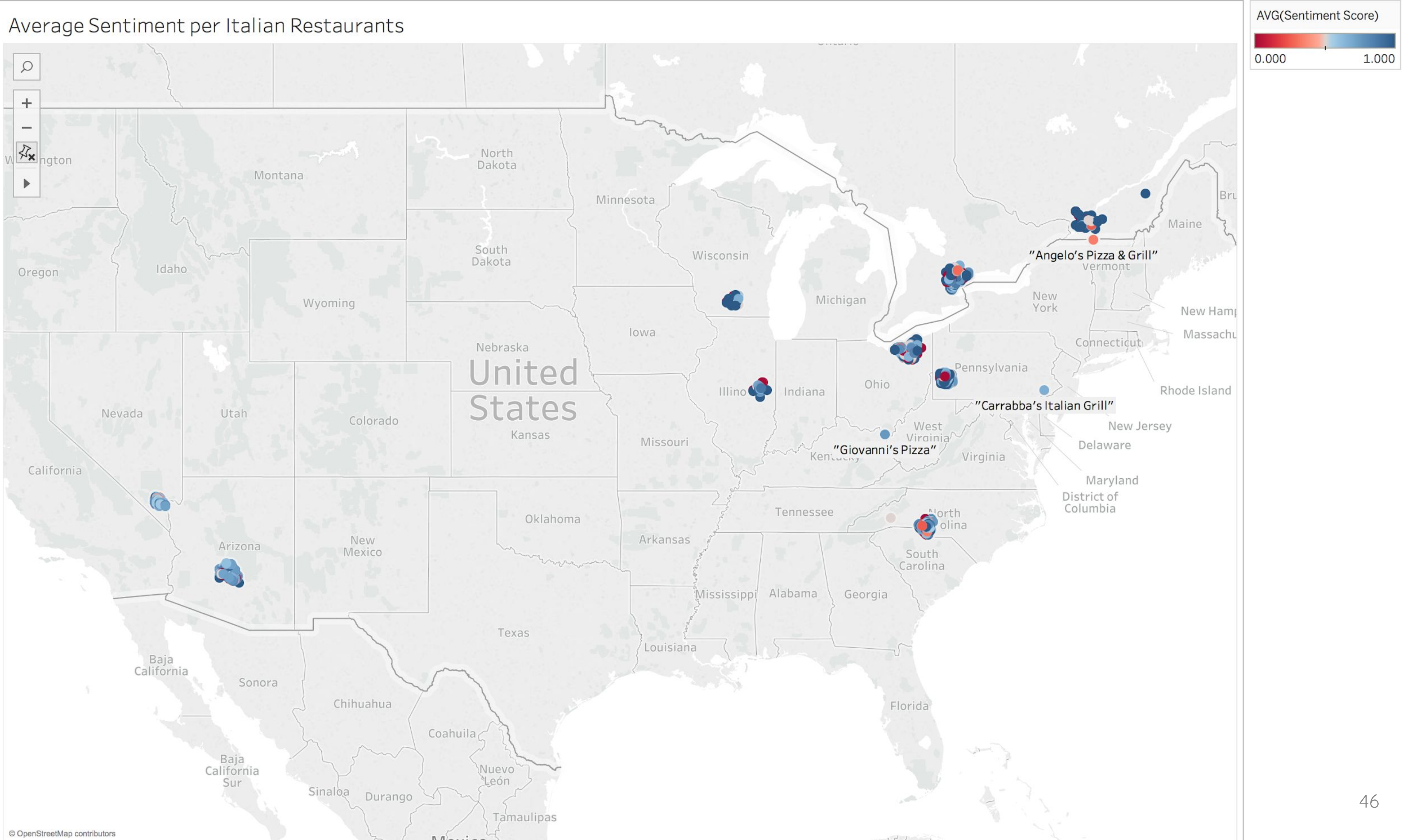
Distribution of sentiments per state



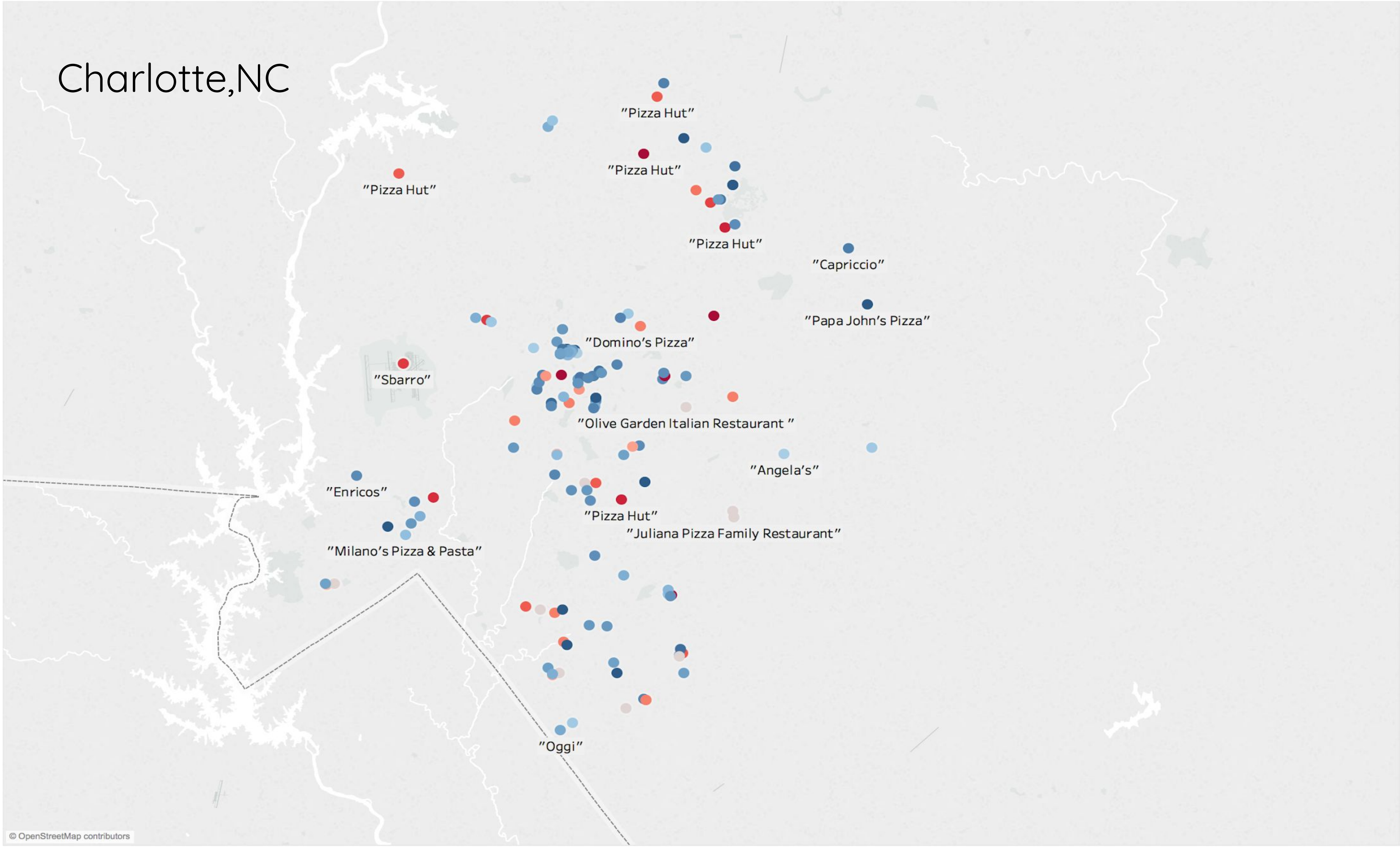
Overall Distribution of Sentiments

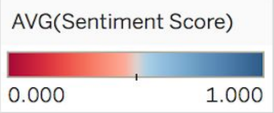


Average Sentiment per Italian Restaurants



Average Sentiment per Italian Restaurants



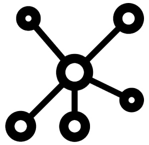


Timeline & Milestones

Timeline	Milestone	Difficulties & Solutions
11.04.18	Ideas brainstorming	Yelp dataset selected
18.05.18	Kick-off presentation	Roadmap, scope & technology selection
29.05.18	First prototype	Local machine: Hadoop cluster & Spark on Ubuntu -> very slow -> moved to Spark cluster on Azure
13.06.18	Review presentation	Azure HDInsight Neo4j Databricks for processing small tables < 2Gb
09.07.18	Intermediate technical solution & Final visualization	Tableau connected to HDInsight Spark Cluster
16.07.18	Final presentation	Finalizing all results

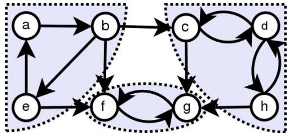
Sources

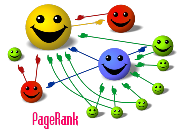
Images:

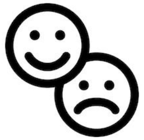
- http://www.theyelphelpers.com/yelp-reviews/ 
- https://www.yelp.de/berlin 
- https://www.yelp.com/brand
- https://buildazure.com/2017/09/25/microsoft-azure-gets-a-new-logo-and-a-manifesto/
- https://venturebeat.com/2017/06/06/databricks-brings-deep-learning-to-apache-spark/
- http://fondazionesvp.it/cosa-facciamo/network-icon-large/ 

Sources

Images:

- https://commons.wikimedia.org/w/index.php?curid=2776582 

- https://commons.wikimedia.org/w/index.php?curid=647584 

- https://www.shutterstock.com/image-vector/customer-satisfaction-happy-sad-unhappy-smiley-431184508?src=ZDsqkYtofCTlDtDSskOm9A-1-3 
- https://www.kisspng.com/png-business-hotel-restaurant-horeca-apni-rasoi-family-4596657/