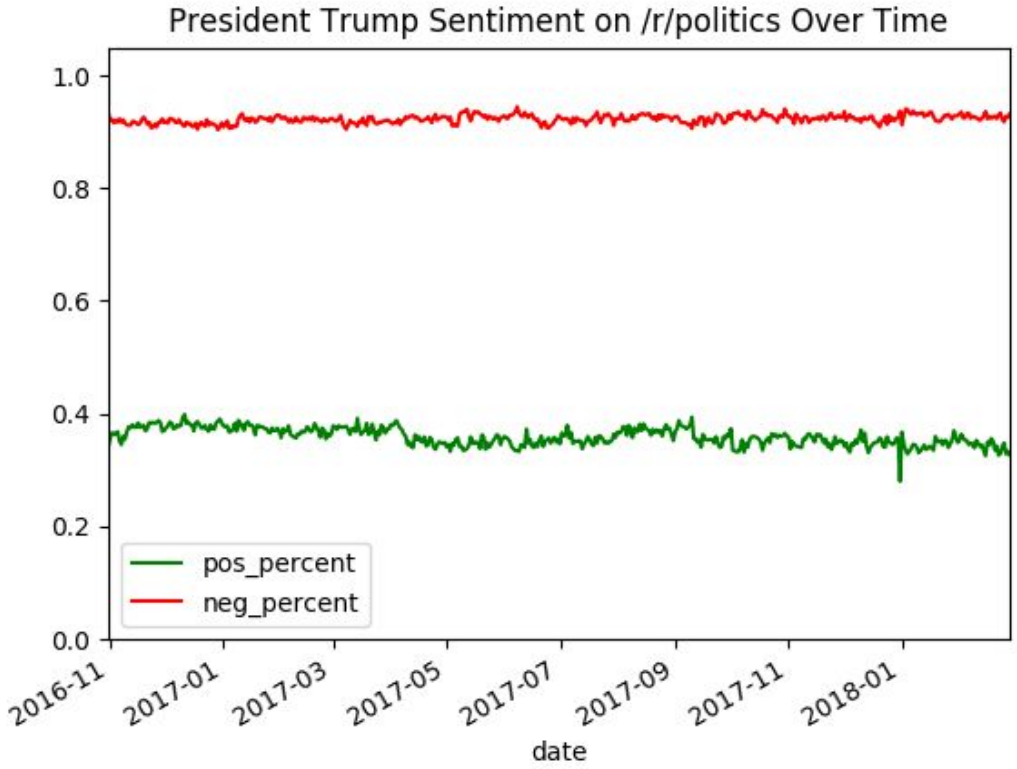
Project 2: Reddit Models
CS143 - Database Systems

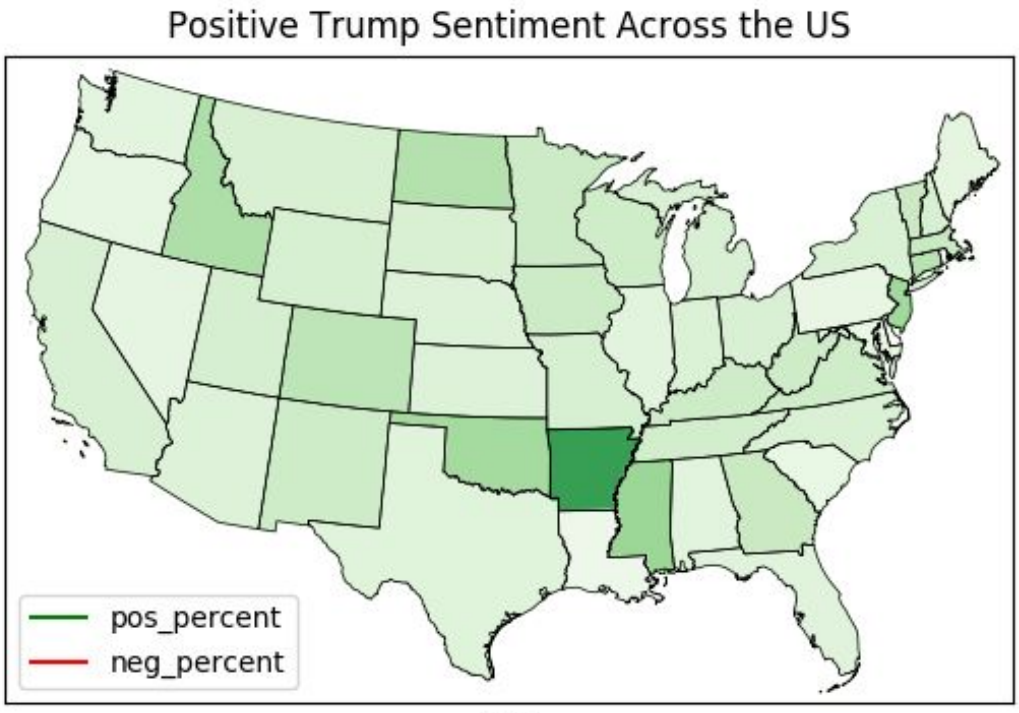
Raymond Lin, 304937942
Raymond Hoong, 904604520

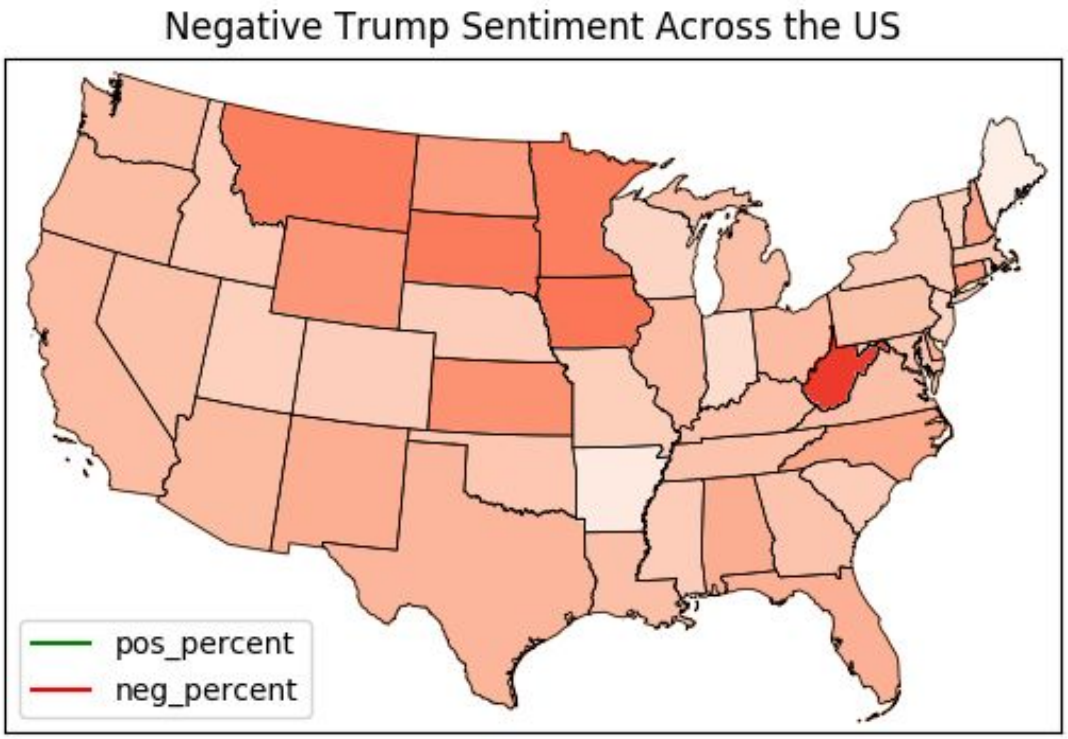
Data Plots:

1. Time series plot (by day) of positive and negative sentiment for Trump:



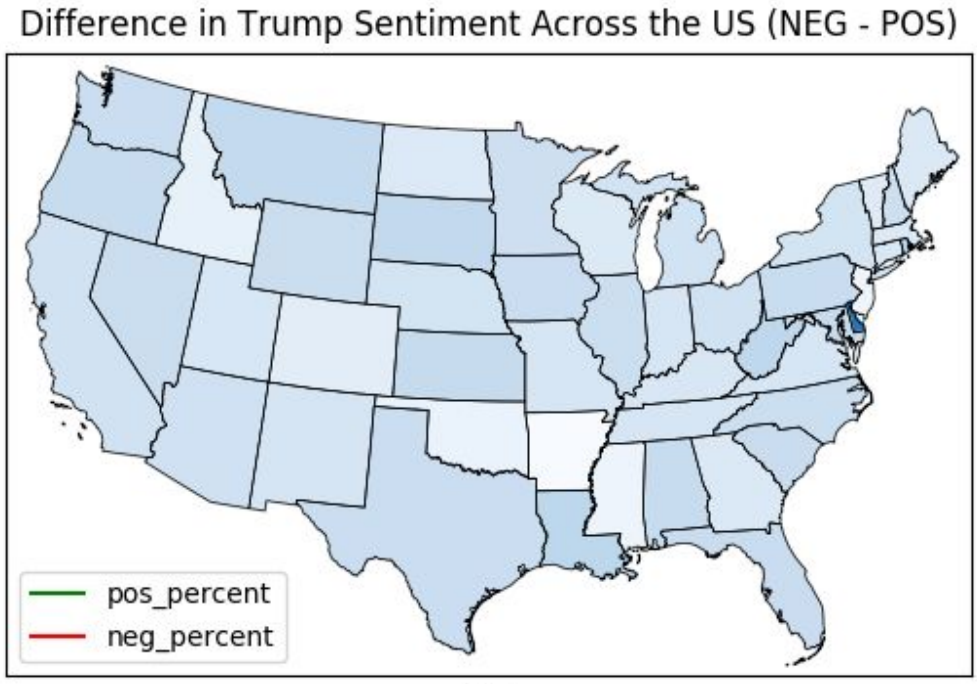
2. Percentages of positive and negative sentiments (by state) for Trump:





3. Difference of percentages between positive and negative sentiments (by state) for Trump:

Because the percentage of negative sentiments for Trump was much higher than the percentage of positive sentiments, we computed the difference: **% Negative - % Positive** instead of the reverse, as stated in the project specifications.



4. Top 10 positive and negative stories

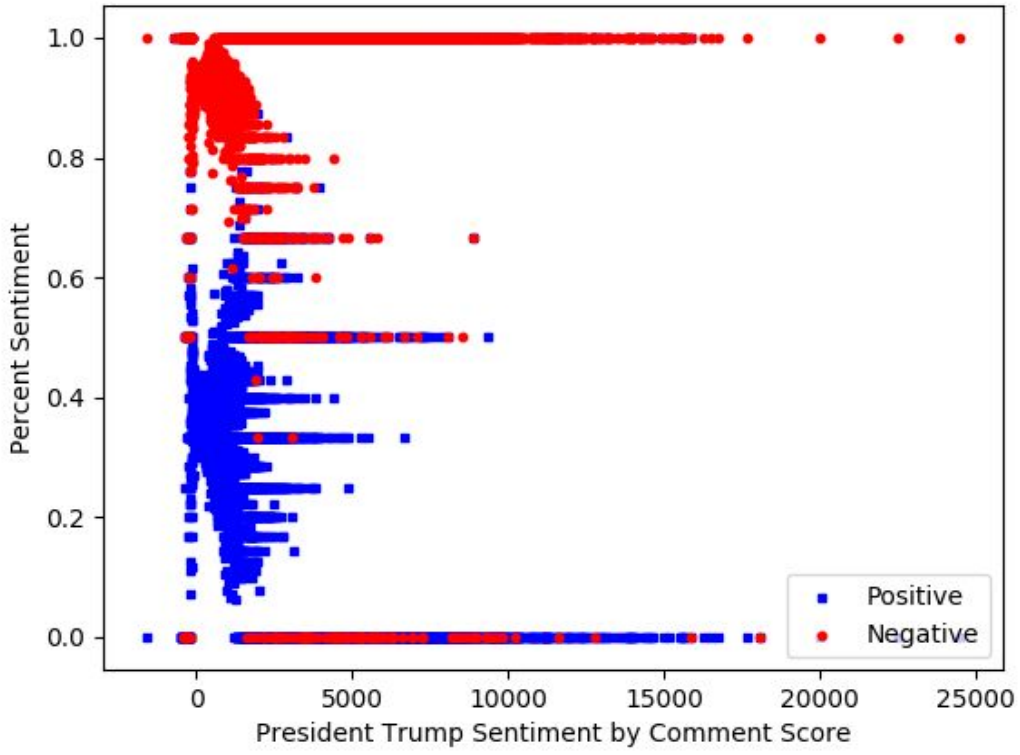
Top 10 Positive Stories

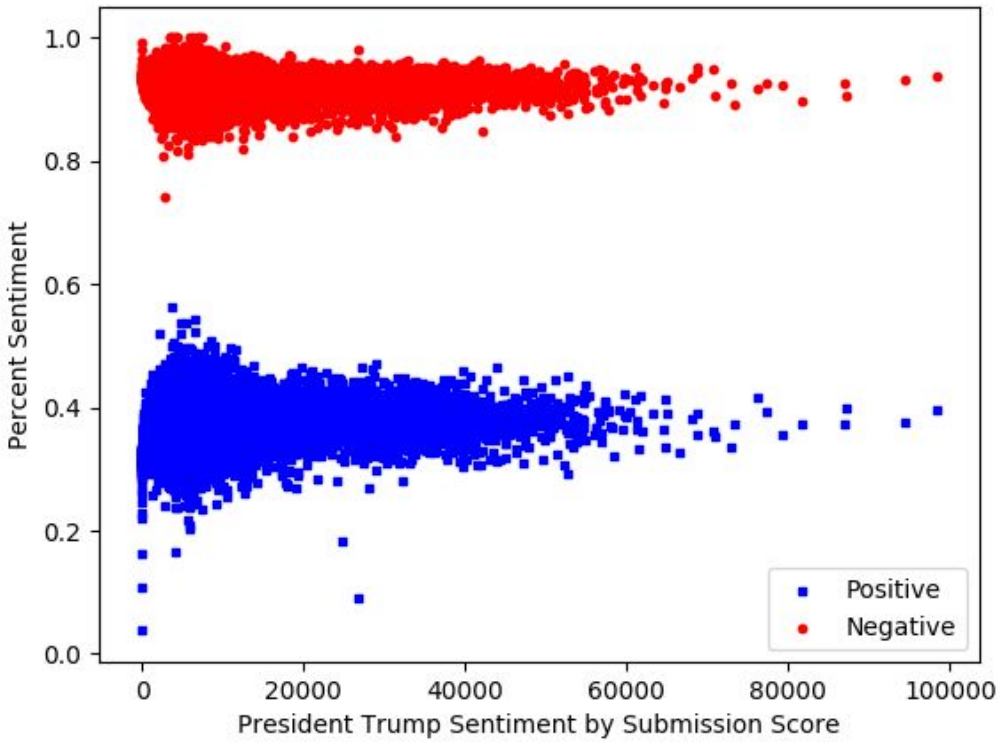
submission_id	title	pos_percent	neg_percent
5eavyi	Welcome to the Brave New (Trumpolitical/Trumponomic) World	1	1
5nbtuh	4chan trolls make it on classified CIA report about Trump 'golden showers'. When will bias news that has 0 veri	1	1
5b8hqh	Debbie Wasserman Schultz - VICE News Tonight on HBO (Full Interview)	1	0.6666666666666666
5pjdi8	Donald Trump leads standing ovation for Hillary and Bill Clinton	1	1
5vtb0u	The Coded Language of For-Profit Colleges	1	1
5dajp4	Clinton's Defeat Means Lost Chances for Many Women on the Rise	1	1
5y30q5	Trump's federal hiring freeze is hitting military families hard	1	1
5gd0cl	Green Party Drops Bid for Statewide Pennsylvania Recount	1	1
5r327p	DC police surveillance cameras were infected with ransomware before inauguration	1	1
6vz4jy	Domestic crisis further clouds stand-off with Kim Jung-un	1	1

Top 10 Negative Stories

submission_id	title	pos_percent	neg_percent
69majy	Reaction to @Raul_Labrador saying \I do not believe that healthcare is a basic human right\" at a town hall meeting"	0	1
6ocr0u	John McCain, US senator, found to have brain tumour	0	1
6ann2f	Comey Asked for More Prosecutor Resources for Russia Probe	0.35714285714285715	1
5bj0ka	Arkansas Ballot Typo Puts 'Liar' in Hillary Clinton's Name	0.25	1
6bwfw1	Fox News Founder Roger Ailes Is Dead	0.6666666666666666	1
5qlq0y	Blagojevich daughter: 'Spineless' Obama has 'broken my heart'	0.2	1
6d59on	Top Russian Officials Discussed How to Influence Trump Aides Last Summer	0	1
5zvn5o	The First Climate Model Turns 50, And Predicted Global Warming Almost Perfectly	0.5	1
6ddrkl	Montana special election results: live updates	0.8333333333333334	1
62olt0	Kansas passes Medicaid expansion bill despite governor's objection	0	1

5. Two scatterplots of comment scores and submission scores





6. Analysis of plots:

Looking at these plots, it is clear that the comments on the subreddit /r/politics do not think too highly of President Donald Trump. From the start of Trump's presidency in 2016 to last year of 2018, the percentages of positive and negative comments towards Trump has relatively remained the same. Comments with positive sentiments towards Trump makes up about 40% of all comments from /r/politics, while the negative comments made up over 90% of all comments. Note that the proportion in the percentages is skewed because comments were allowed to be labeled as both negative and positive. Focusing our attention on the maps, we can observe that sentiments toward Trump varies state by state. Many redditors from the state of Arkansas hold a positive view of Trump, evident by the dark green shading of the state in the first map. In general, states in the South seem more likely to carry positive sentiments towards Trump. Contrastingly, states in other regions tend to be more disapproving of the President. This is clearly demonstrated in the negative-sentiments map, which colors states in the West and Midwest, and North-East with darker shades of red. Another interesting finding is the opposition towards those who support Trump. In the lists provided from the most positive and negative stories, submissions that have high positive sentiments towards Trump also contain high negative sentiments. However, the reverse is not true as the most negative submissions have relatively low positive sentiments. This means that anti-Trump redditors are much more strong-minded and zealous than their counterparts. Any

pro-Trump posts are heavily met with opposition, while anti-Trump posts are generally in consensus.

Questions:

1. There are 3 functional dependencies:
   input_id -> labeldem
   input_id -> labelgop
   input_id -> labeldjt
   The id of the comment is the candidate key, as it uniquely identifies the tuple. Each of the other three columns depend only on the id of the comment.

2. The comments dataframe does not look normalized. There are different pieces of information in the table, such as author, comment, post that the comment is attached to, subreddit etc… We could decompose it by the following:
   -a table about the author, containing:
      -author
      -author_cakeday
      -can_gild
   -a table about the comment, containing:
      -id
      -author
      -body
      -edited
      -gilded
      -created_utc
      -collapsed
      -collapsed_reason
      -controversiality
      -parent_id
      -permalink
      -subreddit_id
      -distinguished
      -retrieved_on
      -score
      -stickied
   -a table about the parent or submissions post, containing:
      -link_id
      -subreddit_id
   -a table about the post's subreddit, containing:
      -subreddit

-subreddit_id
-subreddit_type

-a table about both the author and the post, containing:
-author
-link_id
-author_flair_css_class
-author_flair_text
-can_mod_post
-is_submitter

The collector of the data probably kept the table denormalized because denormalized tables work better with data warehouses and OLAP. Since there is a lot of data, people may work with the data in data warehouses or OLAP systems.

3. Query Plan:

```
== Physical Plan ==
*(3) Project [pythonUDF0#193 AS strArr#178, _c3#173 AS trump#179, CASE WHEN (cast(_c3#173 as int) = 1) THEN 1 ELSE 0 END AS poslabel#180, CASE WHEN (cast(_c3#173 as int) = -1) THEN 1 ELSE 0 END AS neglabel#181]
+- BatchEvalPython [makeStringArr(makeNGrams(body#4))], [_c3#173, body#4, pythonUDF0#193]
   +- *(2) Project [_c3#173, body#4]
      +- *(2) BroadcastHashJoin [id#14], [_c0#170], Inner, BuildRight
         :- *(2) Project [id#14, body#4]
         :  +- *(2) Filter isnotnull(id#14)
         :     +- *(2) FileScan parquet [body#4,id#14] Batched: true, Format: Parquet, Location: InMemoryFileIndex[file:/media/sf_vm-shared/comments.parquet], PartitionFilters: [], PushedFilters: [IsNotNull(id)], ReadSchema: struct<body:string,id:string>
         +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
            +- *(1) Project [_c0#170, _c3#173]
               +- *(1) Filter isnotnull(_c0#170)
                  +- *(1) FileScan parquet [_c0#170,_c3#173] Batched: true, Format: Parquet, Location: InMemoryFileIndex[file:/media/sf_vm-shared/labeled_data.parquet], PartitionFilters: [], PushedFilters: [IsNotNull(_c0)], ReadSchema: struct<_c0:string,_c3:string>
```

Spark seems to be using the broadcast hash join. We first hash the keys or, in this case, the ‘id’, of the larger RDD, or the ‘comments’ dataframe. Then we partition the larger RDD, or ‘comments’ dataframe, by hash value to different worker nodes. Then we take

the smaller RDD or ‘labeled_data’ dataframe and broadcast it to all of the worker nodes. On each node, we compute the hash of the keys in ‘labeled_data’. Of the hash values that match the ones in the worker node, we must manually compare the raw ‘id’, because of hash collisions.