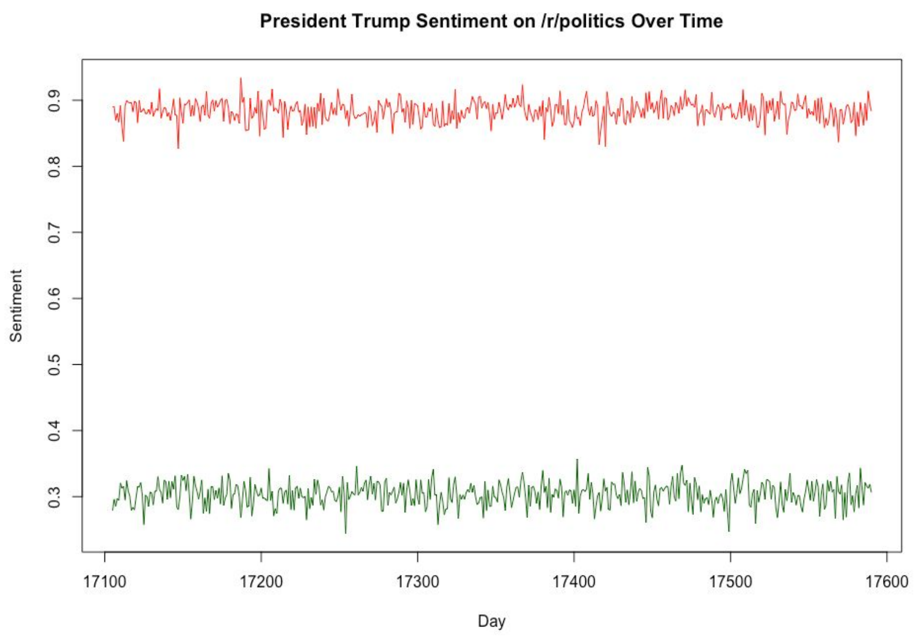
Project 2 Report

CS 143 Introduction to Database

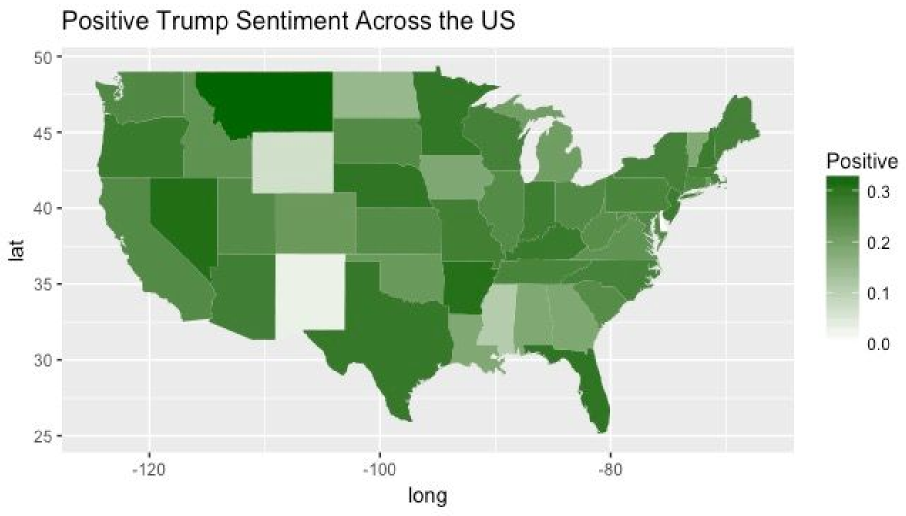
Zhouyang Xue 104629708
Yizhu Zhang 504577340
Yang Li 904642975
Sihan Min 504807176

Positive and Negative Sentiment



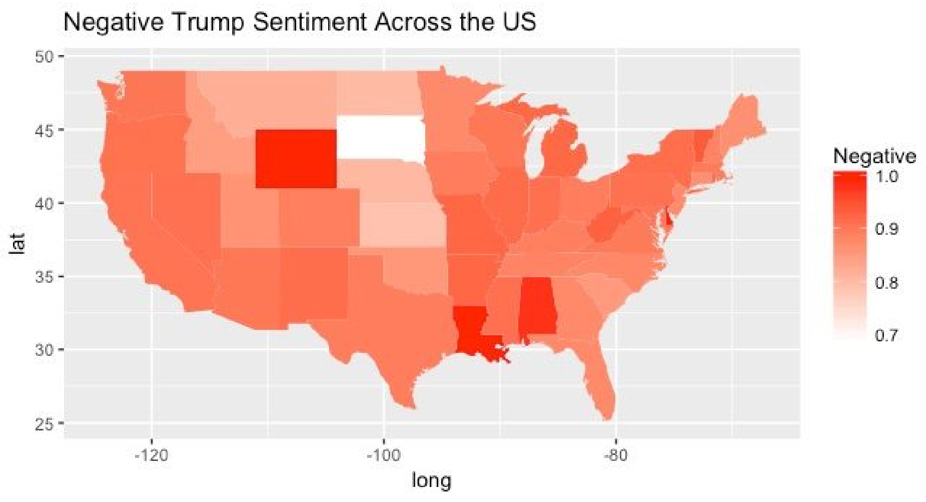
The table shows that from the certain period of time, the attitudes to Trump does not change a lot. Around 80%-90% people post negative attitudes to Trump and only 30% people post positive attitudes to Trump. In the graph, the red curve showcases the negative comment sentiment and the green curve indicates the positive comment sentiment.

Positive Sentiment across United States



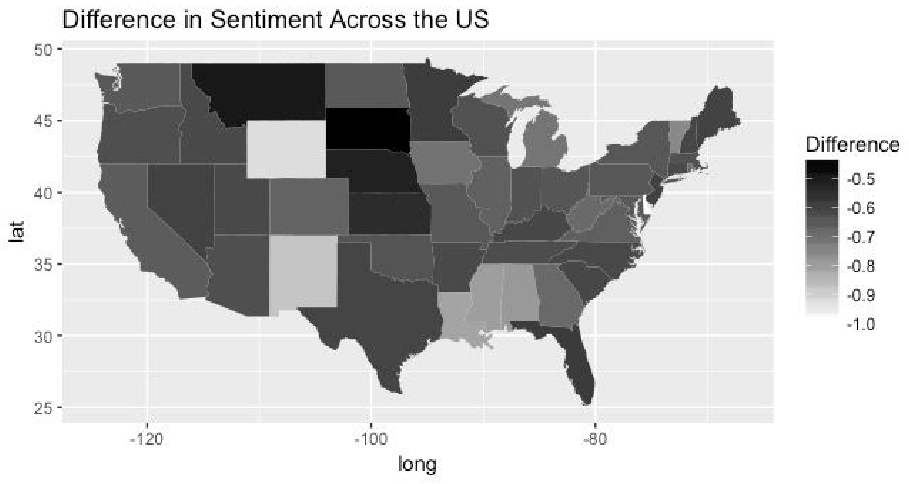
As the graph demonstrates, some particular states have a dominantly high percentage of positive commenters, such as Florida, Montana, and Texas. In the graph, the darker the color is, the higher a portion of positive comments in that state.

Negative Sentiment across United States



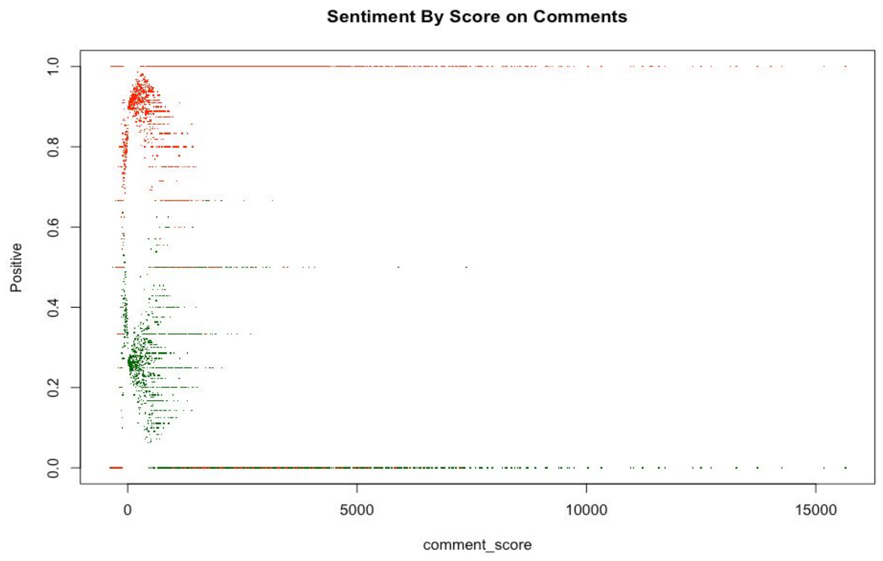
As the graph demonstrates, some particular states have a dominantly large percentage of negative commenters, such as Wyoming, Alabama, and Louisiana. In the graph, the darker the color is, the higher a portion of negative comments in that state.

Difference across United States



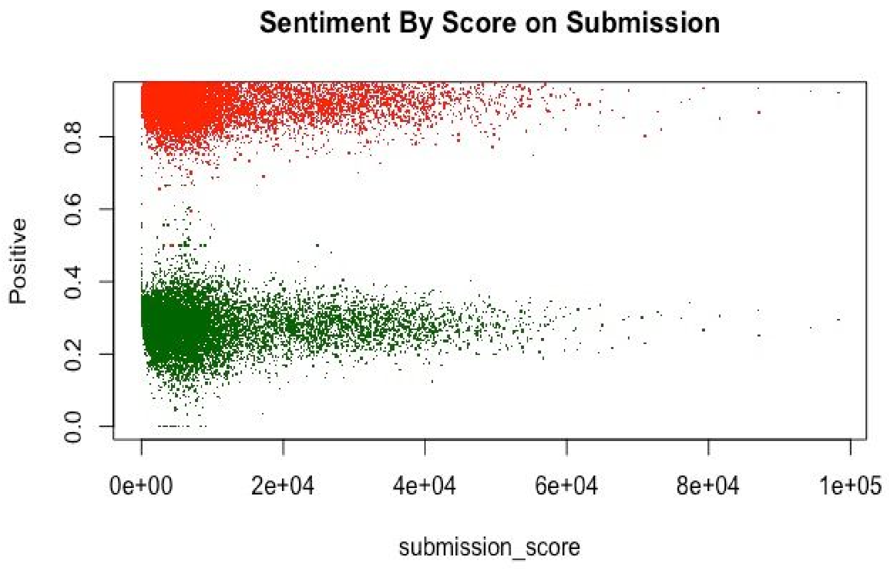
This graph shows the difference between %Positive and %Negative in each state. The reason why the graph looks like this is that overall there are significantly more negative comments than positive comments.

Sentiment by Comment Score



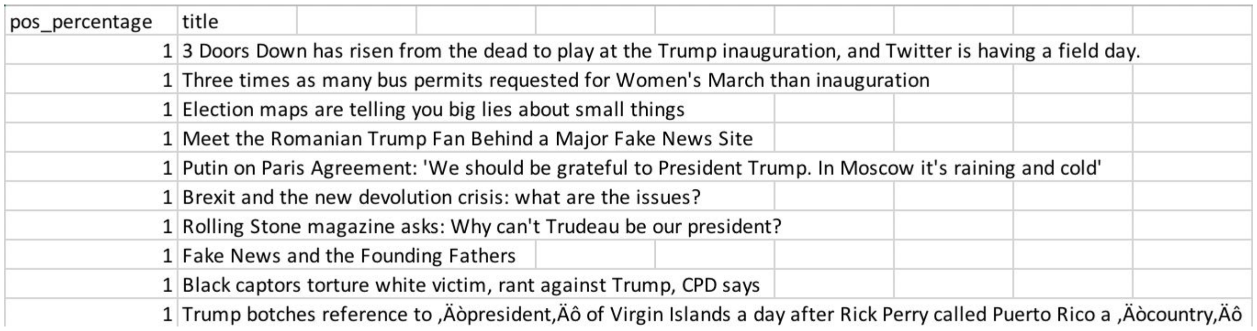
As the graph demonstrates, the higher the comment score, the more extreme the attitudes towards, either positive or negative attitude. However, comment score is not good to be used as a feature because there is no typical relationship between score and sentiment distribution.

Sentiment by Submission Score



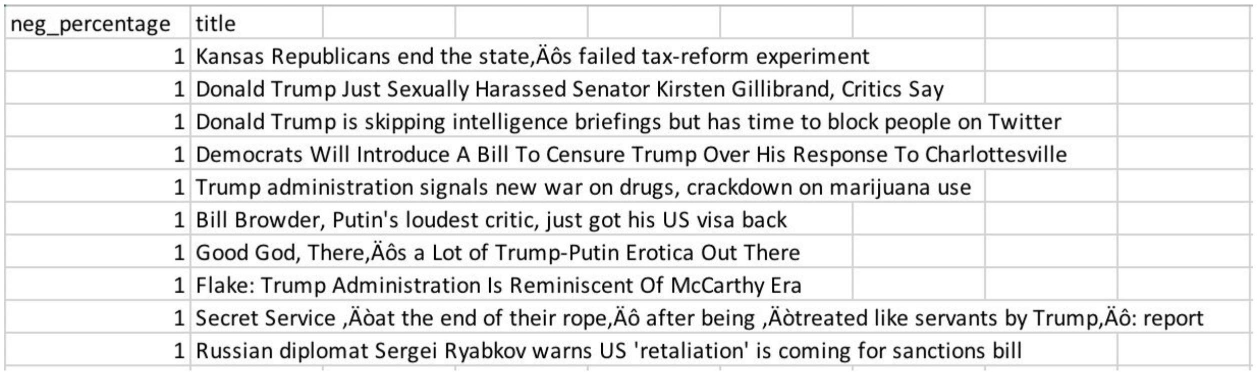
We can see that most of the comments are about stories with low submission score, between 0 to 2000. As the submission score increases, the attitudes become more concentrated around 30% percent positive and 90% negative.

Top Ten Positive Stories



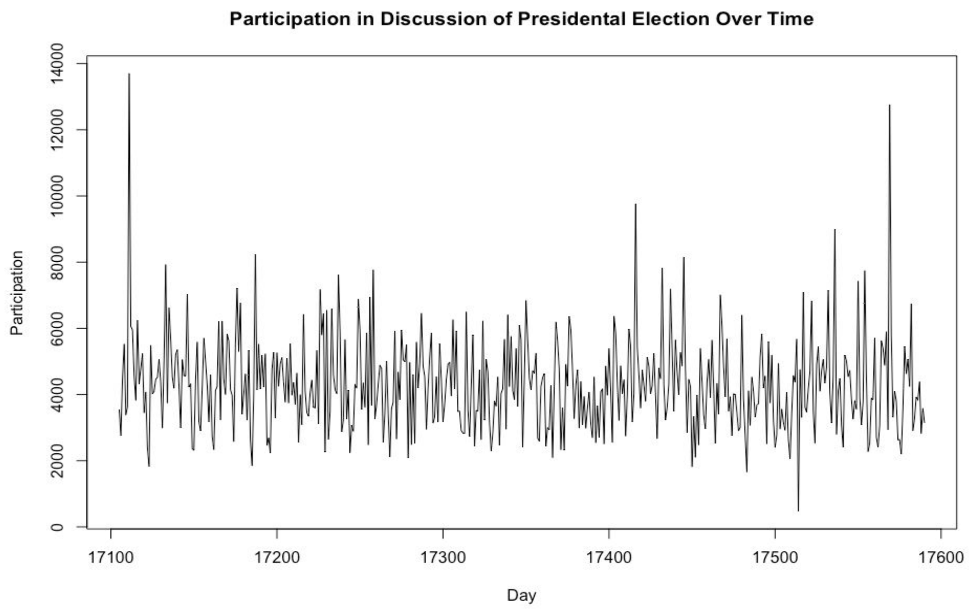
pos_percentage	title
1	3 Doors Down has risen from the dead to play at the Trump inauguration, and Twitter is having a field day.
1	Three times as many bus permits requested for Women's March than inauguration
1	Election maps are telling you big lies about small things
1	Meet the Romanian Trump Fan Behind a Major Fake News Site
1	Putin on Paris Agreement: 'We should be grateful to President Trump. In Moscow it's raining and cold'
1	Brexit and the new devolution crisis: what are the issues?
1	Rolling Stone magazine asks: Why can't Trudeau be our president?
1	Fake News and the Founding Fathers
1	Black captors torture white victim, rant against Trump, CPD says
1	Trump botches reference to ‚Äòpresident‚Äô of Virgin Islands a day after Rick Perry called Puerto Rico a ‚Äòcountry‚Äô

Top Ten Negative Stories



neg_percentage	title
1	Kansas Republicans end the state‚Äôs failed tax-reform experiment
1	Donald Trump Just Sexually Harassed Senator Kirsten Gillibrand, Critics Say
1	Donald Trump is skipping intelligence briefings but has time to block people on Twitter
1	Democrats Will Introduce A Bill To Censure Trump Over His Response To Charlottesville
1	Trump administration signals new war on drugs, crackdown on marijuana use
1	Bill Browder, Putin's loudest critic, just got his US visa back
1	Good God, There‚Äôs a Lot of Trump-Putin Erotica Out There
1	Flake: Trump Administration Is Reminiscent Of McCarthy Era
1	Secret Service ‚Äòat the end of their rope‚Äô after being ‚Äòtreated like servants by Trump‚Äô: report
1	Russian diplomat Sergei Ryabkov warns US 'retaliation' is coming for sanctions bill

Extra Plot: Participation Over Time



This graph measures the total number of comments made per day. It shows people’s participation in the issue of presidential election over time. The great fluctuation in the the amount is probably because of the outbreaks of certain news.

Summary

Based on our discovery, overall, President Donald Trump receives an overwhelming negative comments across the United States. In detail, Around 80%-90% people post with negative attitude to Trump and only 30% people post positive attitude. However, attitudes towards Donald Trump are not evenly distributed across the States. Certain states have significantly higher portion of positive comments and vise versa. For instance, states of Florida, Montana, and Texas have higher percentage of positive commenters compared to other states, and states of Wyoming, Alabama, and Louisiana have higher percentage of negative commenters. In terms of time, public attitude towards Donald Trump, although fluctuating, does not vary much over time. Finally, the higher the comment score, the more extreme the attitudes towards, either positive or negative attitude. As the submission score increases, the attitudes become more concentrated around 30% percent positive and 90% negative.

Short Answer

QUESTION 1: Write the functional dependencies implied by the data.

FD = {Input_id -> labeldem, Input_id -> labelgop, Input_id -> labeldjt}

QUESTION 2: Does the data frame look normalized? In other words, is the data frame free of redundancies that might affect insert/update integrity? If not, how would we decompose it? Why do you believe the collector of the data stored it in this way?

The data frame is not normalized. It has redundant data. For example, the user data such as "author_flair_text", "author_flair_css_classString", "author_cakeday" should all be decomposed to another table for "author". The collector store data in this way in order to achieve efficiency. As joining two huge tables require a lot resource, it might be preferrable to save all columns into one table.

QUESTION 3: Explain what Spark SQL is doing during the join. Which join algorithm does Spark seem to be using?

Rerun the join in Task 2.

Output:

```
== Parsed Logical Plan ==
'Project ['id, 'body, 'labeldem, 'labelgop, 'labeldjt]
+- 'Join Inner, ('comments.id = 'labeled_data.Input_id)
   :- 'UnresolvedRelation `comments`
   +- 'UnresolvedRelation `labeled_data`
```

```
== Analyzed Logical Plan ==
id: string, body: string, labeldem: int, labelgop: int, labeldjt: int
Project [id#32, body#22, labeldem#11, labelgop#12, labeldjt#13]
+- Join Inner, (id#32 = Input_id#10)
   :- SubqueryAlias comments
   :  +-
Relation[author#18,author_cakeday#19,author_flair_css_class#20,author_flair_text#21,
body#22,can_gild#23,can_mod_post#24,collapsed#25,collapsed_reason#26,controvers
iality#27L,created_utc#28L,distinguished#29,edited#30,gilded#31L,id#32,is_submitter#
33,link_id#34,parent_id#35,permalink#36,retrieved_on#37L,score#38L,stickied#39,subr
eddit#40,subreddit_id#41,subreddit_type#42] parquet
   +- SubqueryAlias labeled_data
      +- Relation[Input_id#10,labeldem#11,labelgop#12,labeldjt#13] csv

== Optimized Logical Plan ==
Project [id#32, body#22, labeldem#11, labelgop#12, labeldjt#13]
+- Join Inner, (id#32 = Input_id#10)
   :- Project [body#22, id#32]
   :  +- Filter isnotnull(id#32)
   :     +-
Relation[author#18,author_cakeday#19,author_flair_css_class#20,author_flair_text#21,
body#22,can_gild#23,can_mod_post#24,collapsed#25,collapsed_reason#26,controvers
iality#27L,created_utc#28L,distinguished#29,edited#30,gilded#31L,id#32,is_submitter#
33,link_id#34,parent_id#35,permalink#36,retrieved_on#37L,score#38L,stickied#39,subr
eddit#40,subreddit_id#41,subreddit_type#42] parquet
   +- Filter isnotnull(Input_id#10)
      +- Relation[Input_id#10,labeldem#11,labelgop#12,labeldjt#13] csv

== Physical Plan ==
*(2) Project [id#32, body#22, labeldem#11, labelgop#12, labeldjt#13]
+- *(2) BroadcastHashJoin [id#32], [Input_id#10], Inner, BuildRight
   :- *(2) Project [body#22, id#32]
   :  +- *(2) Filter isnotnull(id#32)
   :     +- *(2) FileScan parquet [body#22,id#32] 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 [Input_id#10, labeldem#11, labelgop#12, labeldjt#13]
      +- *(1) Filter isnotnull(Input_id#10)
         +- *(1) FileScan csv [Input_id#10,labeldem#11,labelgop#12,labeldjt#13]
Batched: false, Format: CSV, Location:
InMemoryFileIndex[file:/media/sf_vm-shared/labeled_data.csv], PartitionFilters: [],
PushedFilters: [IsNotNull(Input_id)], ReadSchema:
struct<Input_id:string,labeldem:int,labelgop:int,labeldjt:int>
```

Explanation: In the output, the optimized logical path showed spark first runs join, than projection and than selection. I think because the projection can filters out multiple columns and the columns are in the select operation, so the query optimize by first project to get small table with smaller columns then select. In the physical plan we can see the spark uses Hash join.