Alex Hunsader
Lexie Kemp-Dahlberg

Project 2 Report

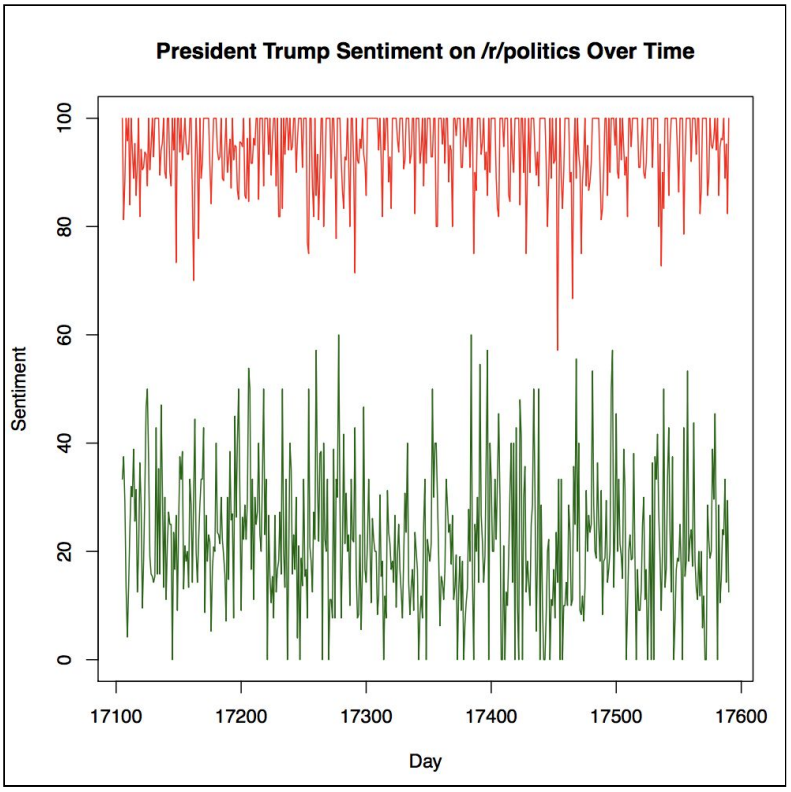

Figure 1: Time series plot of the percentage of comments with positive sentiment (green) and negative sentiment (red) based on the day the comment was posted

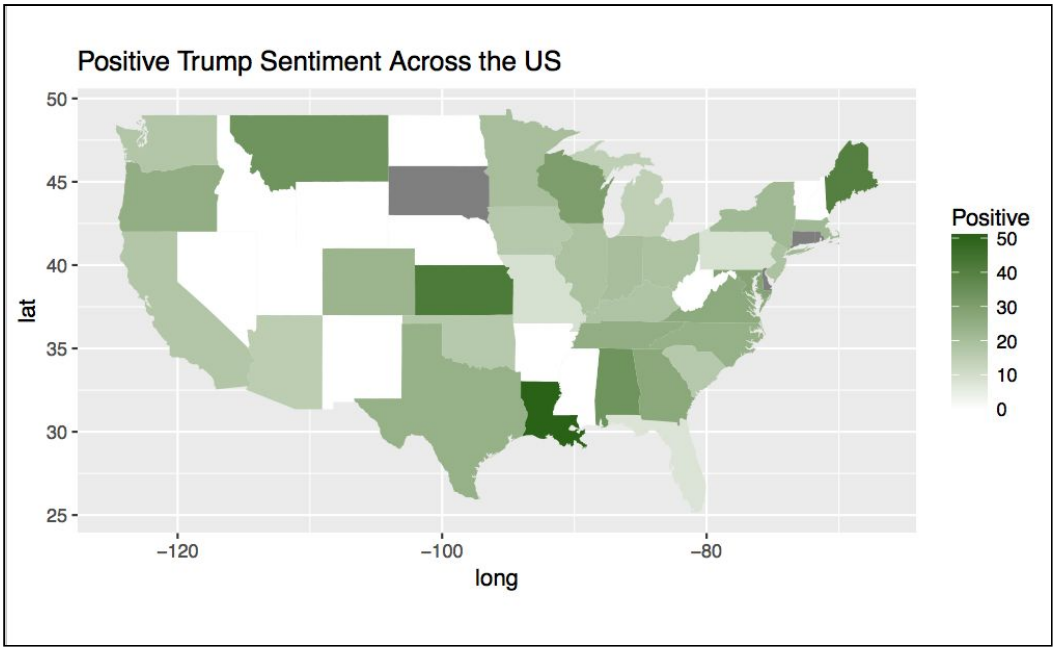

Figure 2: Map of positive sentiment by state

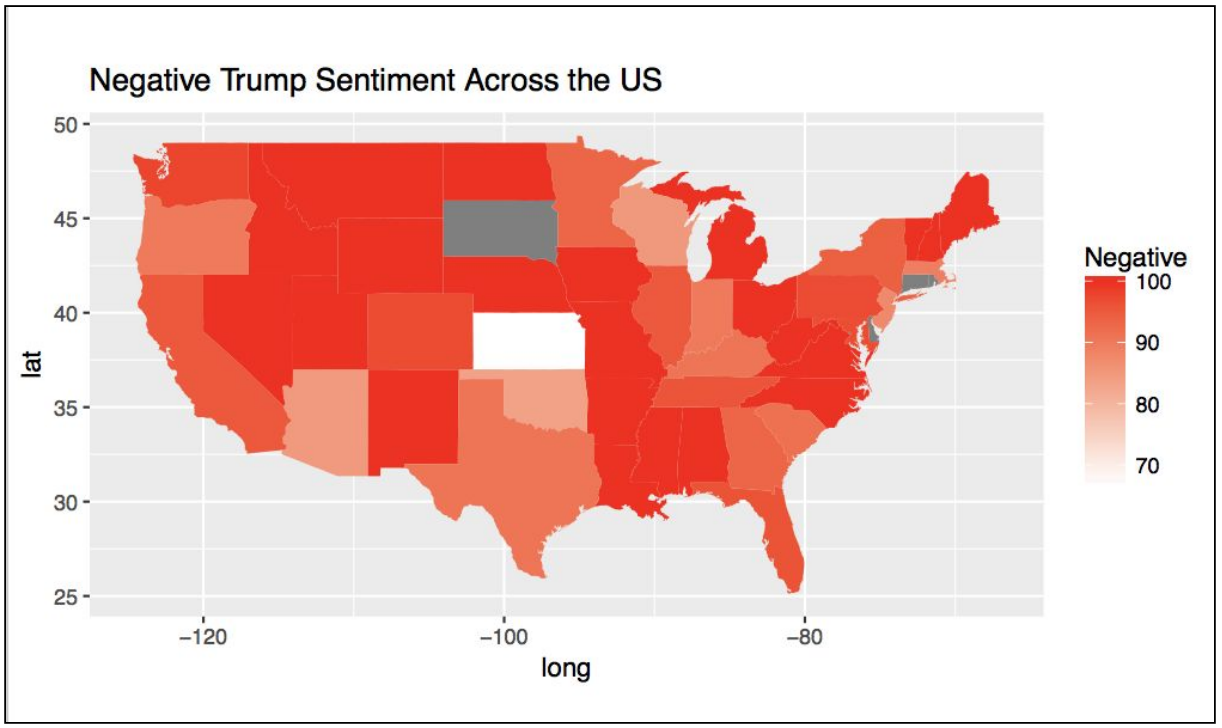

Figure 3: Map of negative sentiment by state

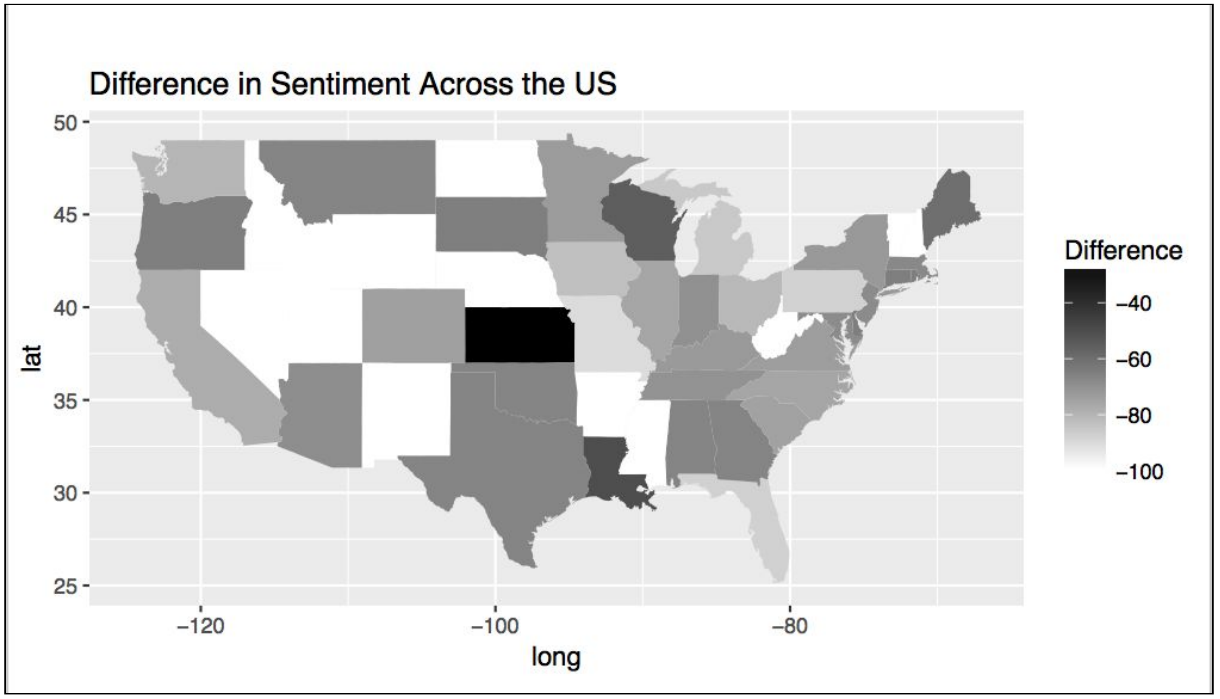

Figure 4: Map of difference in positive and negative sentiment by state

title	neg
Carter Page, Man at the Center of #ReleasetheMemo, Speaks	100.0
Electoral College must reject Trump unless he sells his business, top lawyers for Bush and Obama say	100.0
KING: Conservatives don't hate a golfing President, but they hated an uppity negro golfing President	100.0
Claims of voter fraud in North Carolina turned out to be merely a Facebook joke.	100.0
Macron to award U.S. climate scientists with ‘Make Our Planet Great Again’ research grants	100.0
Donald Trump tells Sean Hannity ‘my CIA speech was a 10 and everybody loved it’	100.0
Trump Humiliated Jeff Sessions After Mueller Appointment	100.0
This Trump real estate deal looks awfully like criminal tax fraud	100.0
Trump's Presidency Is Shaping Up to Be an American Tragedy	100.0
Discussion Thread: GA-06 and SC-05 Special Elections	100.0

Figure 5: Top ten most negative stories

title	pos
Russia says ready to retaliate after U.S. talks end without deal	100.0
Michael Wolff: Jared Kushner and President Trump will throw each other under bus	100.0
CNN anchor: 'Zero chance' journalists will 'move on' from Flynn scandal	100.0
Macron to award U.S. climate scientists with ‘Make Our Planet Great Again’ research grants	100.0
2017 Presidential Inauguration Megathread	100.0
This Trump real estate deal looks awfully like criminal tax fraud	100.0
The Electoral College Was Meant to Stop Men Like Trump From Being President	100.0
Oliver Stone’s “The Putin Interviews” is a litmus test for progressives	100.0
Reporter asks Sarah Huckabee Sanders if she has ever been sexually harassed	100.0
A terrifying axis of idiocy: Kim Jong-un baits Trump, who falls for it	100.0

Figure 6: Top ten most positive stories

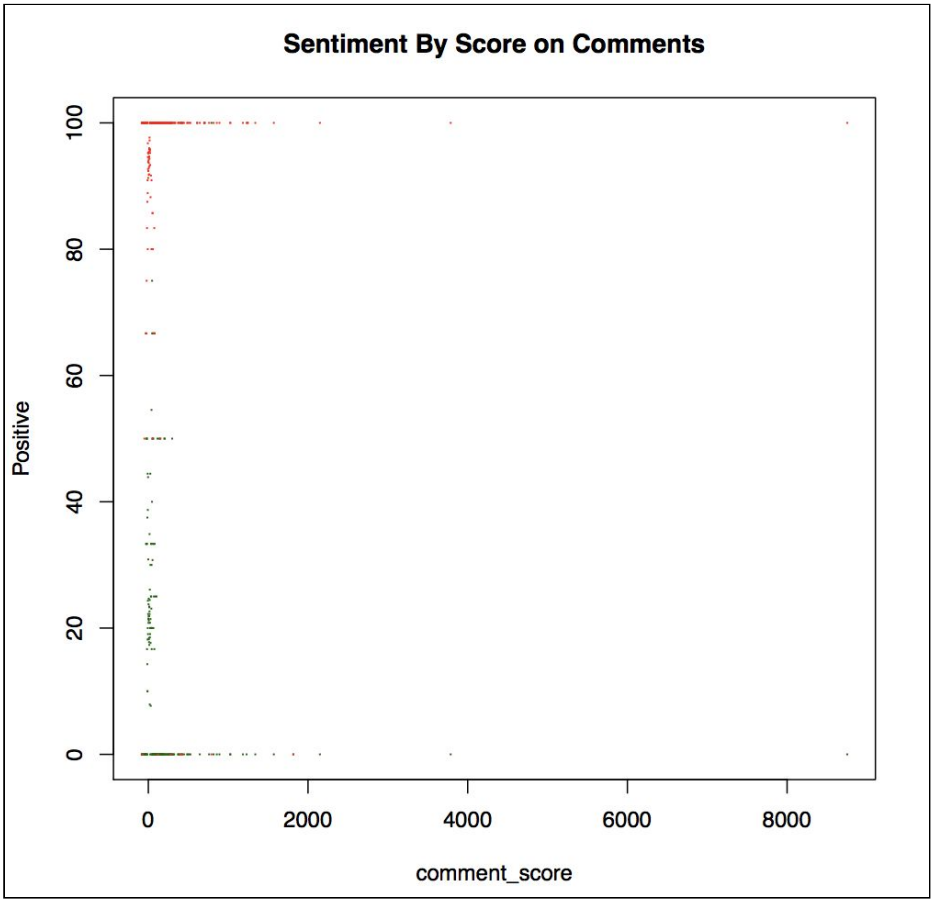

Figure 7: Scatter plot of the percentage of comments with positive sentiment (green) and negative sentiment (red) based on comment score

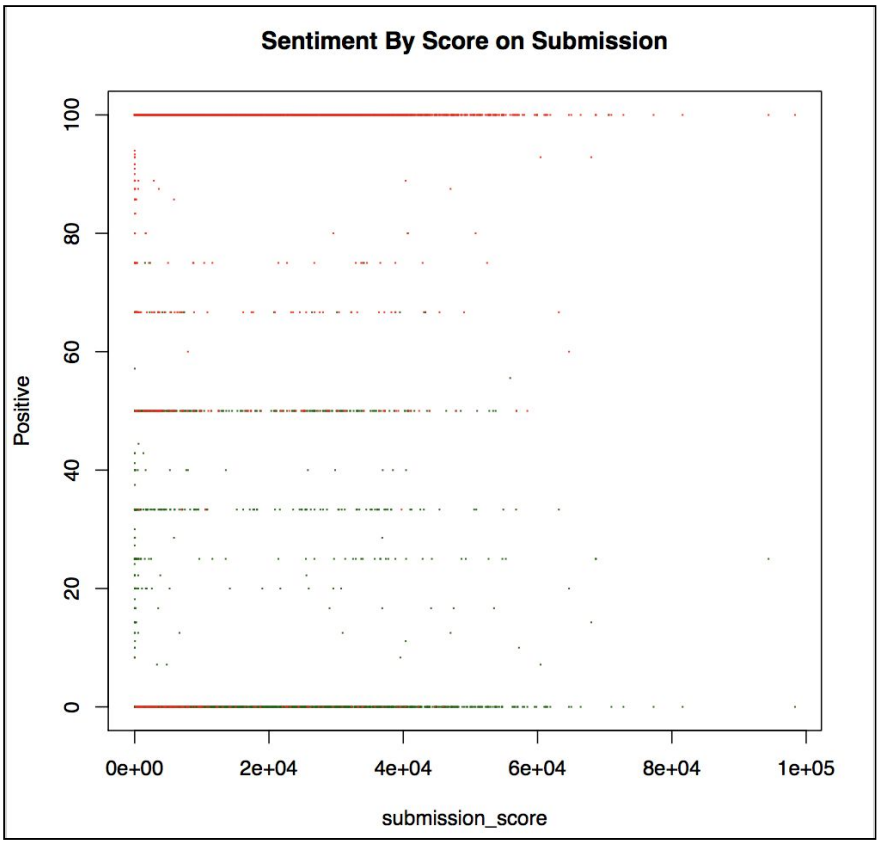

Figure 8: Scatter plot of the percentage of comments with positive sentiment (green) and negative sentiment (red) based on submission score

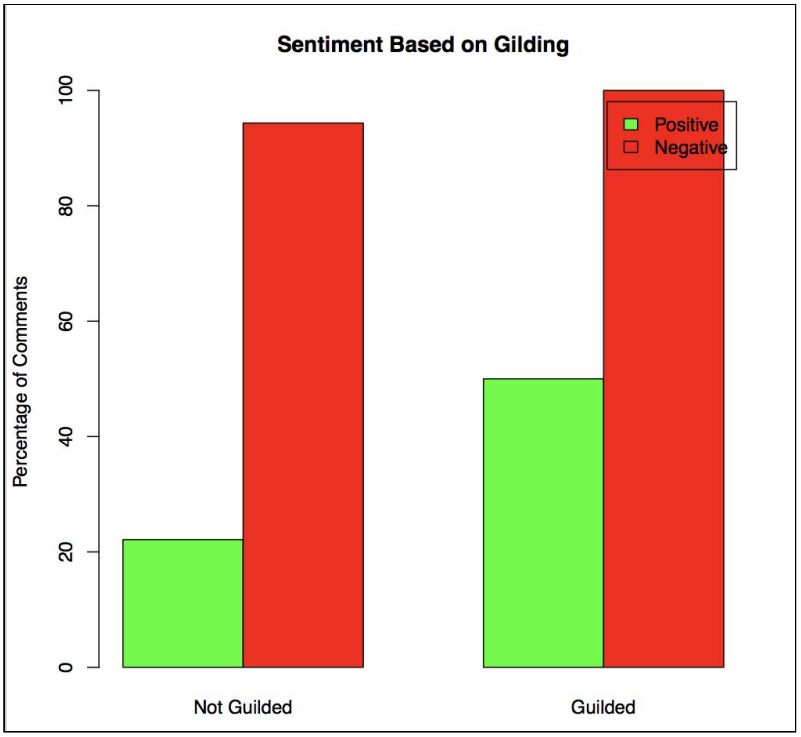

Figure 9: Bar graph of the percentage of comments with positive sentiment (green) and negative sentiment (red) based on whether or not the comment was gilded

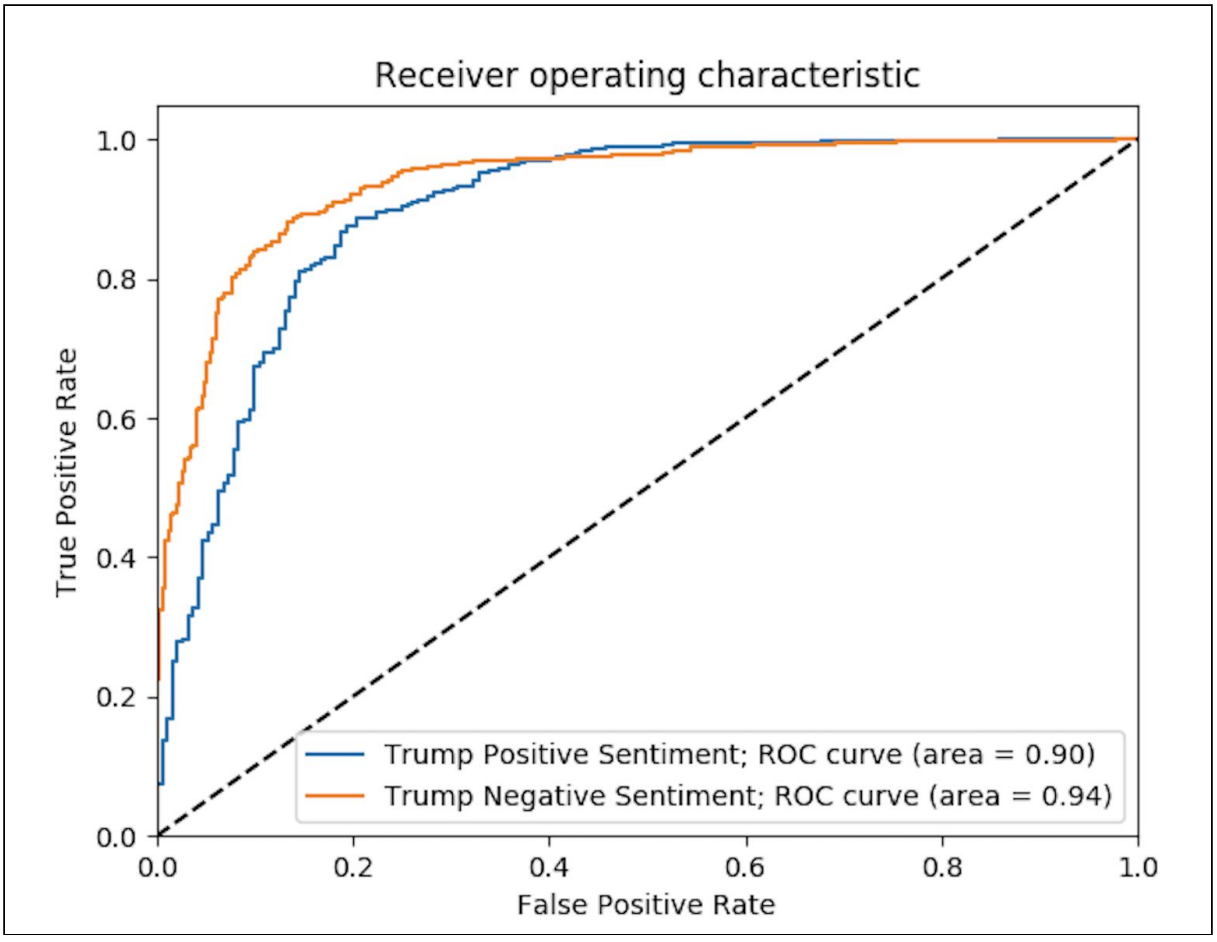

Figure 10: ROC curve to show the accuracy of our predictions for positive sentiment (blue) and negative sentiment (orange)

Overall, there were many more negative comments than positive comments about President Trump. There was not much of a trend over the days, but every day had a much higher percentage of negative comments than positive comment, with many days having 100% of the comments be marked as negative. There was also not much of a trend by comment score. There was a slight trend by submission score in that there were many negative comments on submissions with higher scores. From the US graphs, most of the states had high negative sentiment with no clear region having a lower negative sentiment. There were a couple states (New Mexico, Nebraska, and New Hampshire) that had very high positive sentiment, but otherwise there were no clear trends for positive sentiment. The states also had a very high negative sentiment, in fact, these were the states with the least difference in positive and negative sentiment. We also looked at correlation between gilded comments and sentiment and found that there is not much of a correlation, this could be due to very few comments being gilded.

Positive ROC Score: 0.90
Negative ROC Score: 0.94

The maximum score is 1 which would indicate a perfect model, so we can conclude that our model is relatively decent. This data is printed in our code. The graph (Figure 10) of the ROC curves is generated in our code. This uses the sklearn package.

Questions:

1) Functional Dependencies in labeled_data.csv:
- Input_id -> {labeldem, labelgop, labeldjt}

2) The comments data is not normalized. It contains redundant user data. For every comment a user makes, many attributes about that user are listed. In addition, there is redundant subreddit data. It would be better to decompose the data into a comment table, a user table, and a subreddit table and link the information about a user to his or her comment through the author attribute and the subreddit of the comment through the subreddit id. The user table would contain the attributes author and can_guild, the subreddit table would contain the attributes subreddit_id and subreddit, and the comment table would contain the attributes author, subreddit_id, and the other attributes not in the user or subreddit tables.

The collector of the data may have stored it in this way because it would be easier for people to see all the data in one place for each comment and was not intended for people to be doing inserts or updates that could jeopardize data integrity.

In addition, there is not that much redundant data since can_guild and subreddit do not contain that much information. The collector could have decided that needing a little extra storage space was better than the costs of joining normalized tables (which can be very slow).

3)

```
sqlContext.sql('select sanitize(body) as san, if(trump = 1, 1, 0) as positive, if(trump = -1, 1, 0) as negative from comments inner join labeled on comments.id = labeled.id').explain()

== Physical Plan ==
*(3) Project [pythonUDF0#198 AS san#192, if ((cast(trump#19 as int) = 1)) 1 else 0 AS positive#193, if ((cast(trump#19 as int) = -1)) 1 else 0 AS negative#194]
+- BatchEvalPython [sanitize(body#26)], [body#26, trump#19, pythonUDF0#198]
   +- *(2) Project [body#26, trump#19]
      +- *(2) BroadcastHashJoin [id#36], [id#18], Inner, BuildRight
         :- *(2) Project [body#26, id#36]
         :  +- *(2) Filter isnotnull(id#36)
         :     +- *(2) FileScan parquet [body#26,id#36] Batched: true, Format: Parquet, Location: InMemoryFileIndex[file:/media/sf_vm-shared/comments_data.parquet], PartitionFilters: [], PushedFilters: [IsNotNull(id)], ReadSchema: struct<body:string,id:string>
         +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
            +- *(1) Project [Input.id#10 AS id#18, labeldjt#13 AS trump#19]
               +- *(1) Filter isnotnull(Input.id#10)
```

+- *(1) FileScan csv [Input.id#10,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,labeldjt:string>

This join is done as a hash join, and more specifically a broadcast hash join. This is done when one of the tables is able to fit completely in RAM. In this case, that table is sent to each of the nodes and the other table is divided amongst the nodes (in this case the single node is my computer). In addition, the udf (sanitize) is then evaluated on each of the entries' body column. The if statements on the trump column are also evaluated. Then these calculations are projected to their final names.