The Bachelor(ette): Will You Accept This Rose?

By Sarah Bashir, Ethan Ong, and Lindsey Tam

Abstract

Trying to find love in the 21st century? The Bachelor franchise can help! Twitter and the TV shows the Bachelor and the Bachelorette are key components of American culture. This projects seeks to better understand the relationship between these two media forms. Using text analysis, we explore the polarity of tweet content as well as associated words that people use to describe the show. These trends are visualized and vary by state, show season, and time. Using all of this information, we begin the process of creating a predictive model. Though the current state of the model yields uninterpretable results, we build the foundation of a potentially promising way to use Twitter data to predict the outcome of each show.

Data

This project utilizes two years worth of Twitter data, which was generated shared with use by Professor Mike Izbicki. Data is stored on a supercomputer as a series of JSON files but parsed into smaller CSVs using a Python script. Data for each season was extracted by looking at the dates between times contestants were announced and one week after the final episode was aired. For each season, a set of key words were used to identify relevant tweets. Key words included the twitter handle of each contestant, as well as 'thebachelor', 'thebachelorette', and 'bachelorabc'. Intially, words like 'bachelor' were included, but this led to high amounts of noise. By narrowing the key words to contestants' twitter handles, the data set became significnatly smaller, but we are more confident that the tweets are relevant to the show of interest. In addition to parsing tweets into geographic location, time, and tweet content, we were also able to measure the sentiment of each tweet. A package called TextBlob was used to assign sentiment from a scale of -1 to 1, with -1 being negative, 0 being neutral, and 1 being positive.

A critical problem of this method is dealing with contestants who do not have Twitter. This was extremely problematic for the Bachelorette 2018, because the winner of the season had no Twitter account. To compensate for this, we parsed a second dataset for this season. This is likely to be a noisy dataset, but the results of the model can be compared to see if the noise made a significant impact. Another flaw in the data was the varying size of each data set. For instance, the Bachelor 2018 was very large, consisting of 1,914 tweets. This is likely due to the fact that this season had a unique ending where the Bachelor ended up leaving the finalist for the runner-up in a very short time frame. Other seasons had very few, nearly 300 tweets. This is something to be mindful of when interpreting results.

Example of parsed twitter data from the Bachelor 2018:

```
cleaned_data <- read.csv(file = 'Bachelor_2018.csv')
head(cleaned_data)
```

```
##   X             city                           date polarity state
## 1 0 South Burlington Fri May 18 03:48:18 +0000 2018   0.0000    VT
## 2 1        Manhattan Thu May 17 18:40:39 +0000 2018  -0.0625    NY
## 3 2           Queens Thu May 17 22:43:36 +0000 2018   0.0000    NY
## 4 0            Hooks Sat May 19 01:48:48 +0000 2018   0.0000    TX
## 5 1            Hooks Sat May 19 01:50:59 +0000 2018   0.0000    TX
## 6 2      Lindenhurst Sat May 19 00:03:15 +0000 2018   0.3125    NY
##
## 1                                                                 Mike Johnson is not the her
## 2 This is the dumbest show in the world. IT's all fake everyone saying they love each other after sc
```

```
## 3                                                                                @JohnPaulWebe
## 4 @MarkWoodsmall @packersfan86 @mushwear @jimmyjamny @Babchik @EvCoRadio @jpwilson1982 @JohnPaulFAL (
## 5 @MarkWoodsmall @packersfan86 @mushwear @jimmyjamny @Babchik @EvCoRadio @jpwilson1982 @JohnPaulFAL (
## 6                 The most underrated Led Zeppelin member! John Baldwin aka John Paul Jones. Musical
```

For the visualizations, we created two main data sets. The reasoning behind creating the data sets was to simplify the process of the visualizations. Once the data was in the exact format that was needed for Tableau, it was straightforward to create interactive images. The first data set was very similar to the parsed data from twitter and included the same polarity, state, and text columns. We first took the data from each season and transformed it from the parsed twitter data. We wanted to assign the tweets to a week in the show: e.g. week 1, week 2, etc. In order to do this, the date needed to be in a format of Y-M-D. The code used to transform the date for the bachelor in 2018 is below:

```
bachelor_2018<-separate(bachelor_2018, date, into = c("day", "month", "date", "time1", "time2", "year")
#covert the months to numbers
bachelor_2018$month[which(bachelor_2018$month == "Dec")] = "12"
bachelor_2018$month[which(bachelor_2018$month == "Jan")] = "01"
bachelor_2018$month[which(bachelor_2018$month == "Feb")] = "02"
bachelor_2018$month[which(bachelor_2018$month == "Mar")] = "03"
#select the relevant variables for the new data set
bachelor_2018 <- bachelor_2018 %>%
  select(month, date, year, polarity, state, text)
#y-m-d format
form.dates <- ymd(paste(bachelor_2018$year, bachelor_2018$month, bachelor_2018$date, sep = "-"))
#eliminate previous month and year column
#overwrite dates column with new dates
bachelor_2018$date<-form.dates
bachelor_2018$month<-NULL
bachelor_2018$year<-NULL
```

The next step in creating the new dataset was to assign a week value according to the weeks in the season on wikipedia. To do this, we looked at the dates in specific time frames and assigned weeks based on those values. The process of finding the time frames had to, unfortunately, be done by hand and not using lubridate because some "weeks" actually stretched more than 7 days. The "week" assignment was according to wikipedia assignment, not actual weeks. The dates differed from season to season but the process for assigning each week in the bachelor 2018 is shown below:

```
#create and assign a weeks column based on dates taken from wikipedia
bachelor_2018<-bachelor_2018 %>%
  mutate(Weeks = case_when(
    between(date, "2017-12-09", "2017-12-31") ~ "pre-show",
    between(date, "2018-01-01", "2018-01-07") ~ "week 1",
    between(date, "2018-01-08", "2018-01-14") ~ "week 2",
    between(date, "2018-01-15", "2018-01-21") ~ "week 3",
    between(date, "2018-01-22", "2018-01-28") ~ "week 4",
    between(date, "2018-01-29", "2018-02-04") ~ "week 5",
    between(date, "2018-02-05", "2018-02-11") ~ "week 6",
    between(date, "2018-02-12", "2018-02-18") ~ "week 7",
    between(date, "2018-02-19", "2018-02-25") ~ "week 8",
    between(date, "2018-02-26", "2018-03-04") ~ "week 9",
    between(date, "2018-03-05", "2018-03-14") ~ "week 10"
  ))
```

Finally, we created a column that stored the name of the season and the finalized data for the season was written to a csv file. The process of these two actions is below:

```r
#create a variable that assigns the season to each row
bachelor_2018<-bachelor_2018 %>%
  mutate(Season = "Bachelor 2018")

#create a csv of the new, altered data
write.csv(bachelor_2018, file = "bachelor_2018.csv")
```

This process was repeated for the other 3 seasons of the bacehlor. 4 different csv's were output, each containing the data for the respective seasons. The final step was to combine all the data sets for each season into 1 large data set. The large, combined data set is the main data set used for the map and graph visualizations in Tableau. This data set allowed a for a simple way to filter visualizations by season and week, without very much additional manipulation in Tableau. The code to combine the data sets is below:

```r
#combine all rows of the 4 data sets into one large data set
all_Data<-do.call("rbind", list(bachelor_2018, bachelor_2019, bachelorette_2018, bachelorette_2019))

#save the large data set as a csv
write.csv(all_Data, file = "finalized_data.csv")
```

The next step is to prepare a data set that will be useful in creating a word cloud visualization in Tableau. We began with the data sets previously created that were separated into bachelor_2018, bachelor_2019, bachelorette_2018, and bachelorette_2019. We began by collapsing each the text column in each data set into a single cell. The next step was to separate each word in the cell and include the count. The words were then sorted by decreasing order of count. We looked at the words in each seasons and chose to exclude certain words that we didn't consider relevant in explaining the tweets. However, we were unable to completely eliminate all irrelevant words, specifically individual letters. For example, the letter "i" was unable to be eliminated from the word counts because every word that contained "i" would also be excluded. We later addressed the noisy words we had to leave in by removing them while creating the Tableau visualization. Once the noisy words were excluded, the word counts data was run again and output the top 200 entries for every season and all the seasons combined (5 total data sets). The 2 columns in each data set were the words and the count of each word. Only the top 200 entries were included in order to make the data set more manageable to work with in Tabeleau and for sake of relevancy. Below is the R code for creating the bachelor 2018 word count data:

```r
#collapse each text column in each data set into one cell
sentiment_bachelor_2018 <- paste(unlist(bachelor_2018$text), collapse = " ")

#cast the data into a data frame
data_bachelor_2018 <- data.frame(category = sample(sentiment_bachelor_2018, 100, replace = TRUE), string

#separate each word in the cell and get the number of occurences of each word
data_bachelor_2018 <- data_bachelor_2018 %>% unnest_tokens(word, category) %>%
  group_by(word) %>%
  count()

#sort the words by decreasing order of number of occurrences
data_bachelor_2018 <- data_bachelor_2018[order(data_bachelor_2018$n, decreasing = TRUE),]

#words not to include in word counts
bad_words <- c("the", "https", "t.co", "to", "a", "you", "and", "for", "of", "on", "that", "so", "my",

#new word counts excluding the irrelevant words
data_bachelor_2018<-data_bachelor_2018[!grepl(paste(bad_words,collapse="|"),data_bachelor_2018$word),]
```

```
#top 200 words
data_bachelor_2018<-head(data_bachelor_2018, 200)

#words and their counts for each file written to csv format
write.csv(data_bachelor_2018, file = "bachelor_2018_word_count.csv")
```

The above process was repeated for the other 4 data sets.

For the predictive model, we used a dataset that summarized contestant information of total tweet count and average tweet sentiment. We only looked at 3 seasons of data, which included the Bachelor (2018-2019) and the Bachelorette (2019). Example of predictive model data.

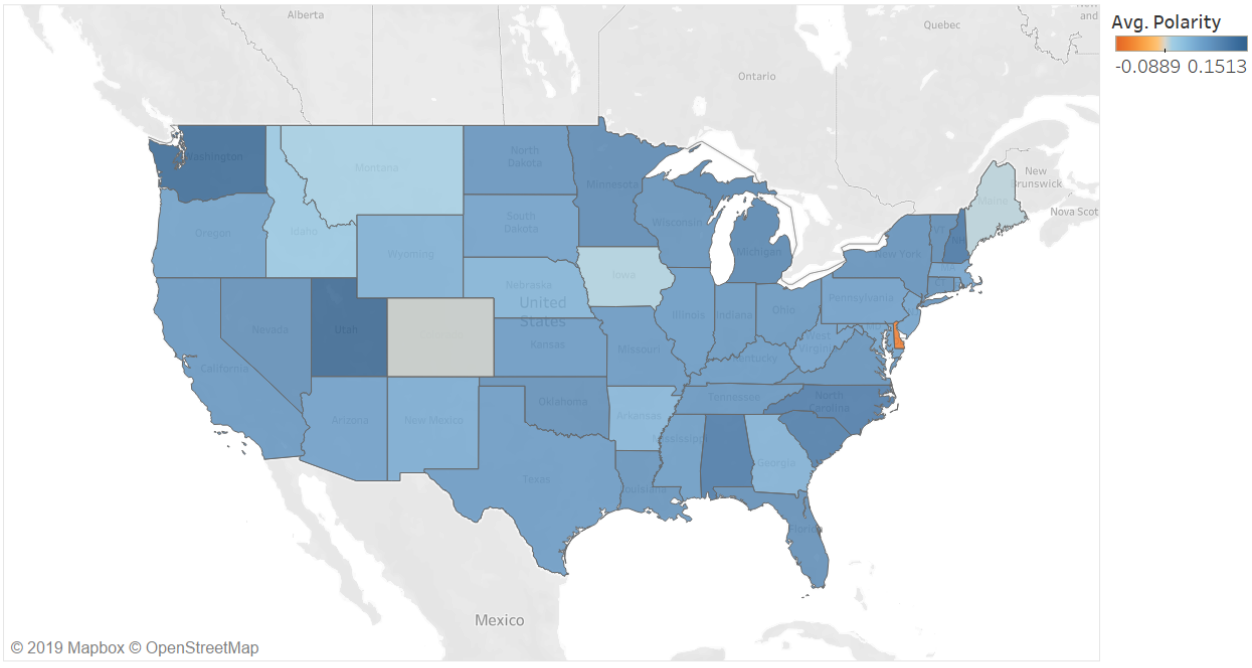
```
generated_data <- read.csv(file = 'model_data.csv')
head(generated_data)
```

```
##   X             X0        X1  X2 status
## 1 0        thebkoof 0.1840278 297 winner
## 2 1 laurenburnham91 0.1250230  66  loser
## 3 2  KendallPatrice 0.2456223 186  loser
## 4 3     tiarachel91 0.1430435 501  loser
## 5 4  whats_ur_sign_ 0.1433547 439  loser
## 6 5   seinnefleming 0.2112450  66  loser
```

Visualizations

The visualizations associated with this project were generated in Tableau and published on Tableau public.

Polarity by State

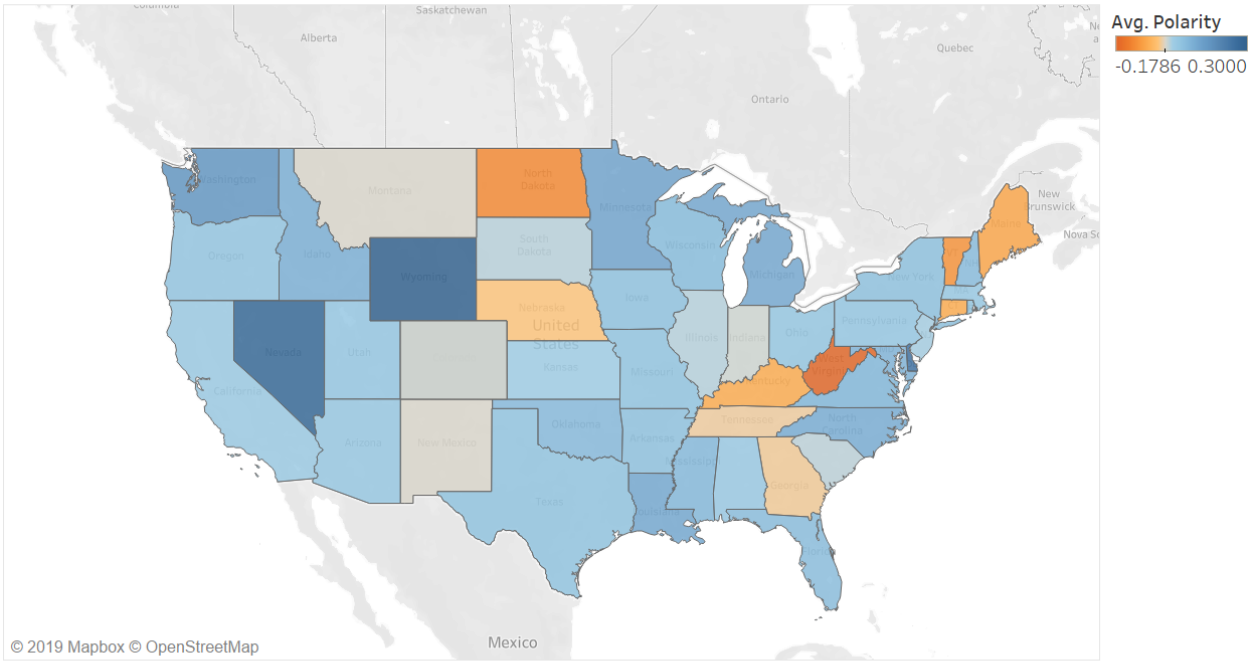


Map based on Longitude (generated) and Latitude (generated). Color shows average of Polarity. Details are shown for State. The data is filtered on Weeks, Season and Action (State). The Weeks filter keeps 11 of 11 members. The Season filter keeps Bachelor 2018, Bachelor 2019, Bachelorette 2018 and Bachelorette 2019. The Action (State) filter keeps 53 members.

For visualizations, we wanted to first understand how polarity varies occcurs across the United States.

Above is a map displaying average polarity of tweets related to The Bachelor/Bachelorette in 2018 and 2019 across all weeks the show is aired. We can see that, on average, tweet sentiment is generally positive across all states, indicated by the blue indicator color. On the other hand, across both shows during 2018 and 2019, it seems that Twitter sentiment in Delaware is on average negative, indicated by the red indicator color.

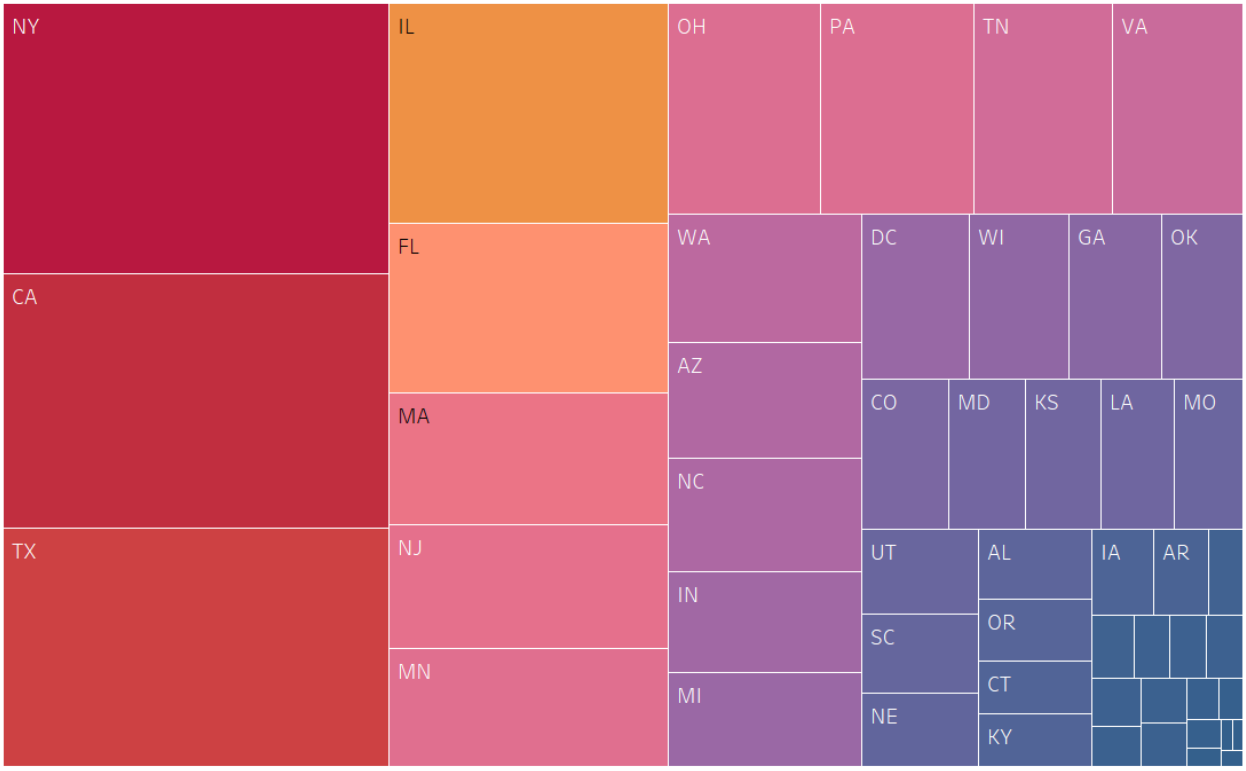
Polarity by State



Map based on Longitude (generated) and Latitude (generated). Color shows average of Polarity. Details are shown for State. The data is filtered on Weeks, Season and Action (State). The Weeks filter keeps week 10. The Season filter keeps Bachelor 2018. The Action (State) filter keeps 53 members.

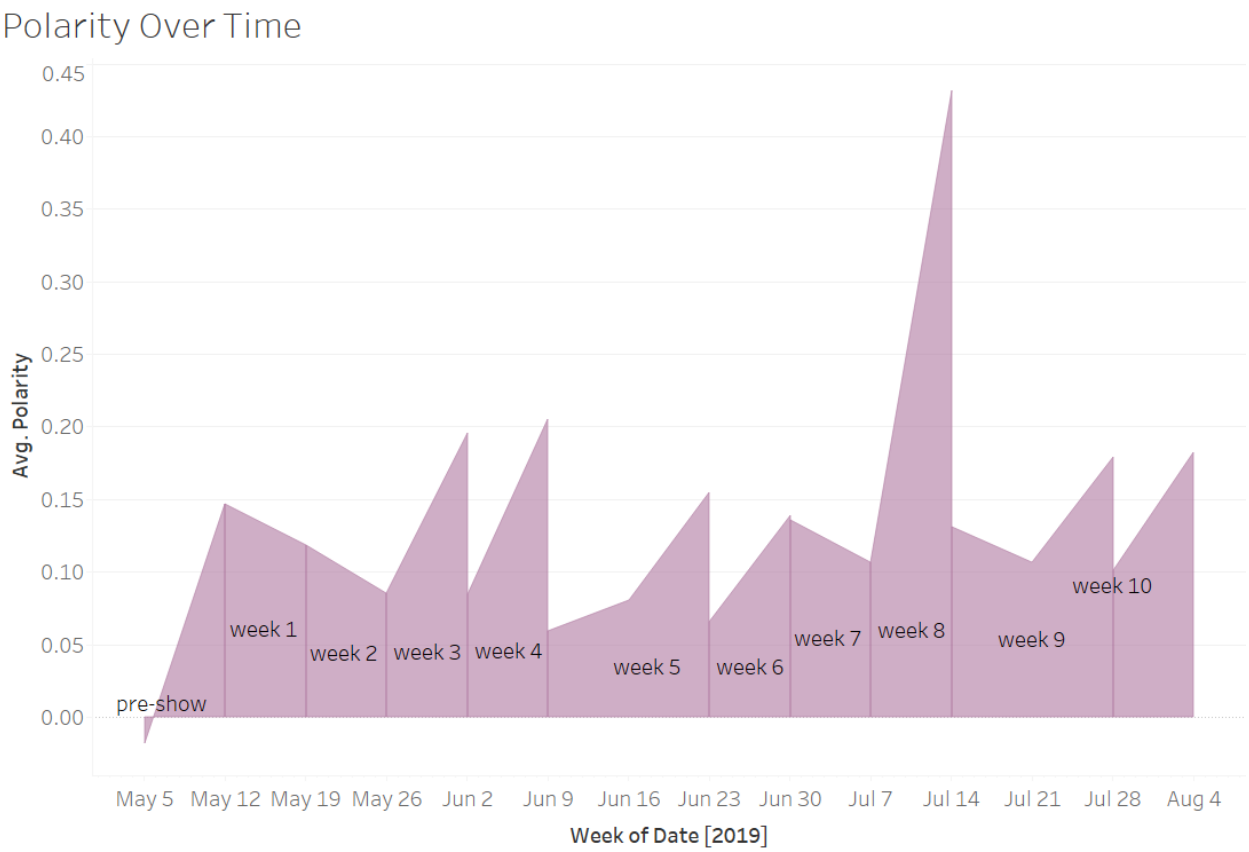
For our map visualizations, our online pulbished version has an interactive feature where the user can toggle the map's display by show season and week. Above is a map displaying average polarity of tweets related to The Bachelor in 2018 on the airing of the last episode and the following week after. We can see that, based on the deep blue indicator color, tweets about the show from Nevada and Wyoming, on average, had the most positive sentiment. On the other hand, tweets about the show from West Virginia and North Dakota, on average, had the most negative sentiment, indicated by the more saturated red indicator color.

Tweet Volume by State



State. Color shows sum of Number of Records. Size shows sum of Number of Records. The marks are labeled by State. The data is filtered on Weeks and Season. The Weeks filter keeps week 10. The Season filter keeps Bachelor 2018.

To accompany the map visualizations, we created an accompanying visualization that represents Tweet volume by state. On the online interactive version, the user can toggle show season and week to compare polarity by state and Tweet volume by state simultaneously. Above is a block graphic displaying Tweet volume for The Bachelor 2018 on the airing of the last episode and the following week after. We can see that, New York, California and Texas dominate in Tweet volume related to The Bachelor this particular week. This helps the user carefully consider the map visualizations by recognizing how many Tweets are considered for caluculating average polarity between states.



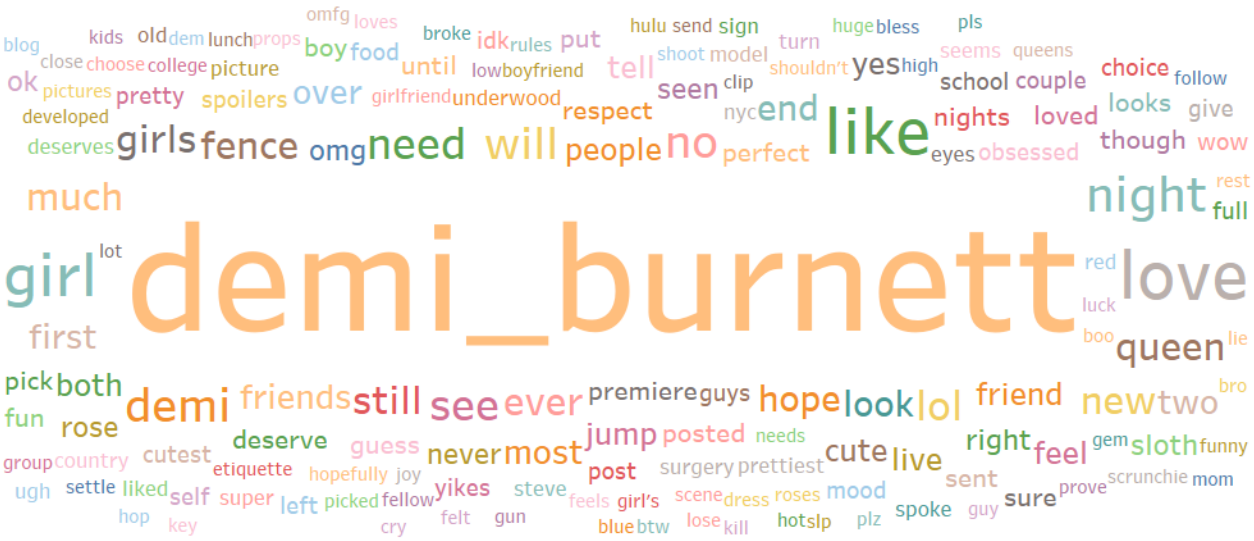
The plot of average of Polarity for Date Week. The marks are labeled by Weeks. The data is filtered on Season, which keeps Bachelorette 2019.

In understanding how polarity varied during the airing of each show season, we created a visualization that plots polarity as a function of time, which the user can filter by show season. Above is an area graphic displaying Tweet polarity across the United States for The Bachelorette 2019 during the entire airing of the show. Furthermore, each area graphic is sectioned by week. This enables the user to interpret polarity episode by episode and the direct impact of an episode. In the above graphic, we can see that sentiment was generally positive about the show with a noticeable spike in positive sentiment during week 8. From conducting brief market research, we conclude that viewers start to express stronger positive sentiment in their Tweets around week 8 due to contestants having more intimate conversations which is a catalyst for chemistry and drama.

Word Count Over 4 Seasons

Word. Color shows details about Word. Size shows sum of N. The data is filtered on N, Word (bachelor_2018_word_count), Word (bachelor_2019_word_count), Word (bachelorette_2018_word_count) and Word (bachelorette_2019_word_count). The N filter includes values greater than or equal to 100. The Word (bachelor_2018_word_count) filter keeps 153 of 200 members. The Word (bachelor_2019_word_count) filter keeps 161 of 200 members. The Word (bachelorette_2018_word_count) filter keeps 156 of 200 members. The Word (bachelorette_2019_word_count) filter keeps 172 of 200 members.

To understand what viewers are tweeting about the most, we created an interactive word cloud visualization that formats the most frequently used words in Tweets in a cloud format, excluding nonsensical and filler words. Above is a word cloud across all show seasons. We can see that words such as `like`, `love`, `need`, and `omg` are often used words in tweets, as indicated by their large size, related to The Bachelor and The Bachelorette. On the online interactive version, the user can also toggle the word cloud to only include words that have a certain minimum word count.



Word. Color shows details about Word. Size shows sum of N. The data is filtered on N, which includes values greater than or equal to 200. The view is filtered on Word, which keeps 161 of 200 members.

To follow up a word cloud that visualizes word frequencies across all show seasons, we also created individual word clouds by show season. Each word cloud also included the feature to toggle the word cloud to only include words that have a certain minimum word count. Above is a word cloud representing frequently used words about The Bachelor 2019. Words such as `demi_burnett`, `fence`, `sloth`, and `love` are some examples of words that are frequently used, as indicated by the word cloud. Furthermore, high word frequencies can help us determine notable moments in a particular show season. Further market research can easily be done to understand the causal and resultant factors of certain high word frequencies.

Ethical Considerations for predictive models

It is critical to point out that no interpretation should be made about these models. We were only able to train this model on 3 seasons worth of data (not 4 because one season's winner had no Twitter handle). A better model would have included data from all 23 seasons of The Bachelor and all 15 seasons of The Bachelorette. Although we have created a predictive model, that predicts the winner of a season based on sentiment, ethically, we must address the issue of training a model with few data points and drawing conclusions from it. The model likely has high bias with so little data and will not perform well on out-of-bag data. We are therefore unable to draw any meaningful conclusions from the model but are framing how this model should be formulated as an approach to how the model could look with sufficiently more seasons or more data (50+ seasons).

The 5 classification models

These models were implemented using Python's scikit learn library. A quick overview of each model is attached below:

- Naive Bayes is a supervised classification model that uses Bayes' theorem of conditional probability and also uses the 'naive' assumption that feautures are independent. In this case, the Naive Bayes model is trying to predict whether or not a contestant will given their tweet count and average sentiment, under the assumption that tweet count and sentiment are independent (which they are not). Therefore, as more data is inputted into a model, accuracy is expected to go down because the independence assumption does not always hold true, as in this case. In general, a Naive Bayes model creates overly-simplified assumptions, which make it a poor model to use on real world data. It is included in this project in order to compare performance with other models.
- Linear Discriminant Analysis is a supervised classification model that creates a linear decision boundary that seperates multiple classes. LDA has two main uses: dimensionality reduction and linear classification. For dimensionality reductions, LDA tries to project the data into another space/dimension that minmimzes distance from the mean while minimizing variance of each class; this projection can make the data linearly seperable in a lower space when it was not linearly serperable in its original space.
- Logistic Regression is a classification algorithim that generates the probability of binary dependent variables. In our case, the dependent variable is whether or not a contestant will win or not win based on total tweet count and avergae tweet sentiment. It is similar to linear regession, except the outcome is binary (not continuous) and the seperating boundary is 'S' shaped.
- Linear Support Vector Classification tries to linearly seperate training data into classes. A linear SVC can be though of as a SVM. with a linear kernel. Linear SVC tends to perform better with larger sample sizes and is more computationally efficient because it is using a linear kernel.
- Stochastic Gradient Descent Classifier is a linear classifier that utilizes SGD for training. This means that for each data point, the gradient (or minima) of loss is estimated. The SGD classifier has a similar loss function as compared to Logistic Regression, but the solver each one uses is different. Since Logistic Regression uses GD and not SGD, we expect the SGD classifier to be the most computationally efficient out of these 5 models. Like the Naive Bayes model, SGD Classifier is not commonly used in practice.

Model with 3 seasons worth of data

Based on the way dataset was created, the classification model will have 89 classes (contestants), 2 dimensions/features (avg sentiment + total tweets), and a binary outcome (winner or not winner).

The results are highly questionable, due to the extremely high accuracy. We therefore are extremely doubutful of the results but instead view the data as a positive signal for the potential of adding much more data to the model.

```
knitr::include_graphics("chart.png")
```

	name	accuracy
0	nb	[0.9444444444444444]
1	lda	[0.8888888888888888]
2	lr	[0.9444444444444444]
3	svc	[0.8888888888888888]
4	sgd	[0.9444444444444444]

Conclusions

We observe that although no long-term predictions can be drawn from 4 seasons of The Bachelor franchise data, we are still able to setup the foundations for a potential predictive process and look at long-term modeling possibilities (when using more data). We are also able to draw interesting conclusions through the use of interactive visuals. From the visuals, we gain a clear understanding on the geographic distribution of the tweets and how polarity varies across states. We also were able to visualize polarity as a function of time/episode week by show season. Lastly, we were also able to visualize the most frequently used words when discussing different seasons of The Bachelor/Bachelorette. From all of this information, we can evaluate how people on Twitter are feeling from season to season of the show, whether it is more negative or positive about a particular show season during a particular time. Ultimately, our conclusions drawn from polarity averages in each state is our most meaningful data.

Next Steps

Access to more seasons or more data would allow us to create even more insightful visualizations and a more reliable predictive model. This would require Twitter data from 2006 - 2017 (Twitter founded in 2006), to be available, which we could access with a paid developer account through Twitter. Furthermore, hyperparameters in each classification model can be tuned to yield better results. Additionally, looking at polarity in specific major cities could lead to more specific insights between more specific locations. In our current project, we looked at tweet volume by state; however, it may be more meaningful in the future to look at tweet volume as proportional to population of the state. Data from more seasons would greatly increase the number and quality of conclusions we are able to draw from The Bachelor franchise data.

New Insights

In order to complete this project, we had to learn how to interpret JSON data and parse the data into a CSV. Accessing data through a supercomputer (ssh-ing through the machine) was a major hurdle in intially collecting data and a new process that the team learned a lot from. Completing this project required gaining a better understanding of the 5 classification models mentioned previously mentioned. String manipulation was a new technique implemented when data wrangling. Furthermore, generating the visualizations required learning Tableau, which none of the team members had experience in was crucial to our most insightful findings.