Detecting and Predicting Happiness

Group ID#: 10
Group Name: Moodbusters
Topic: Mood Detection and Prediction

Yuhai Li, 104844760
John Kim, 504511858
Martin Verde, 804423412
Richard Yu, 304464688
Taylor Caulfield, 404773737

CCS Concepts

• Information systems→Information retrieval→ Retrieval tasks and goals → Sentiment Analysis • Proper nouns: People, technologies and companies →Technologies →TREC→Twitter

Keywords

California; Sentiment analysis; Twitter

1. INTRODUCTION

Sentiment analysis, the automated extraction of emotional data from text or speech **(Kharde & Sonawane, 2016)**, has long been a favored tool for its ability to gauge consumers' opinions and feelings en masse with minimal input from the analyst's end. It is often performed on Twitter and Facebook data to gain insight into how the public views a particular brand or a political candidate, or to measure the trust placed into a particular financial stock **(Feldman, 2013)**. As such, we feel that sentiment analysis would be the perfect tool for answering a sociological question such as ours.

In this project, we aim to explore how the Thanksgiving holiday affect individual's mood. For the sake of simplicity, we are choosing to only explore how people in California are affected by the holiday's approach and passing. In addition, we want to see if an individual's mood is affected by the day of the week and the outside temperature.

We will first gather tweets from Twitter, which can serve as a proxy to individuals' thoughts and the general public mood **(Bollen, 2011)**. Next, we will perform sentiment analysis on each tweet to determine whether it is happy or sad. Then, we will split the tweets up by date and tally up the number of happy and sad tweets for each day to determine how happy or sad individuals in different regions of California felt on each day in November. Finally, we will analyze the trends in the total numbers of happy and sad tweets and compare those trends with recent events to answer our question.

2. ABSTRACT

Our goal with this project was to apply sentiment analysis to a large number of Tweets to extrapolate how the Thanksgiving holiday impact individual's mood in roughly a three-week period. To apply sentiment analysis, we first roughly assigned labels using a dictionary of positive and negative words and by scanning the tweet for emoji's that may be indicative of the Tweeter's mood. Out of the three models we chose to predict a Tweet's mood: k-nearest neighbors, support vector machine, and logistic regression, only the k-nearest neighbors approach provided a meaningful prediction model. In addition, we found that positivity in Tweets increases from the week before Thanksgiving to the day of, but falls off afterward.

3. PROBLEM DEFINITION

Before we can perform sentiment analysis on our tweets, we will need to write and train a fitting model that will predict whether a given tweet is positive or negative. We will also need to amass a large set of tweets (at least 1,000,000 tweets) to train our fitting model and to obtain a representative picture of Californians' thoughts. Additionally, we will have to extract only the relevant information from each tweet and convert that information into a format that makes sense to our fitting model. Writing down the information from a million tweets by hand is infeasible, so we will be automating the downloading and extraction of Tweets.

As such, our approach will roughly consist of the following steps:

- Designing and deploying a Twitter crawler to automatically download tweets
- Preprocessing the tweets to obtain information that our fitting model can read
- Designing and training our fitting model with a subset of our downloaded tweets to hone its ability to predict happiness/sadness
- Running our fitting model on another subset of our tweets to obtain happiness/sadness levels for each day
- Analyzing the levels and cross-referencing them with current events to determine the influence of the Thanksgiving holiday on Californian's moods relative to other events
- Writing a report of our findings

4. DATA PREPARATION

We wrote and deployed a Twitter crawler written in Python that used the Twitter streaming API to grab the tweets from users located in California. This crawler ran constantly throughout from November 14th to December 6th. The resulting tweets were then cleaned to extract their relevant information. We then used this information to get the user behind the tweet and the temperature information associated with the tweet. Finally, all the information for each tweet was packaged into a JSON file that could be read by our model (which will be described in the next section).

The following information was extracted or inferred from each tweet.

- Location
    - We eliminated tweets from individuals that do not reside in California.
    - We also separated individuals by their location within California to observe how different regions may have different moods throughout Thanksgiving season.
    - Due to the concentration of tweets around certain areas, we chose to define three main location categories: Los Angeles, San Francisco, and other.
- Date
    - The tweets are separated into bins for each day for our experimentation phase. Tweets were acquired from the date range 11/14 – 12/6 with the exception of the following dates: 11/17-11/19, 11/22, 11/28, 12/5
- Words
    - The presence of certain words in a tweet can hint at the tweeter's mood at the time of writing. For example, a person will usually only use the word "hooray" when they are feeling happy, or the word "sucks" when they are not in a good mood.
    - Because we do not care about the ordering of the words in the tweet, the text body of a tweet can be represented as a "bag of words." This is a vector of integers, with each index corresponding to a particular word. The value at each index is the number of times the corresponding word appears in the tweet. Representing text in "bags of words" allows us to quickly aggregate their word counts, allowing us to isolate the words that occur most commonly throughout our corpus of tweets. In the end, we chose not to use this metric, as it slowed down preprocessing, training, and predicting too much.
- Temperature
    - Certain studies hint that weather and temperature patterns have been shown to influence mood levels. Elevated temperatures, higher barometric pressure, low humidity, and clear skies seem to correlate with a positive increase in mood. However, the actual details tended to vary between studies, and certain other studies found no correlation between weather and mood (Lucas & Lawless, 2013). In addition, since we are only focusing on three main location categories, we have found our weather information gathered to be largely the same, so we have decided to discount weather in our model. We may still want to take temperature into consideration when trying to train our model, but if we do so, we will not want to place too much weight in it.
    - Since Twitter doesn't provide weather or temperature information from their APIs, we look up the weather and temperature information from the Metaweather API based on their location. (MetaWeather, 2017)
    - Any tweet whose temperature information could not be obtained was eliminated.
- Day of the Week
    - Average mood levels can vary with the day of the week. Mood levels tended to rise starting on Friday, peak on the weekends, and decline starting on Monday. (Mislove, Lehmann, Yong-Yeol, Onnela, & Rosenquist, 2010)
- Time
    - Average mood levels can also vary with the time of the day. Mood levels peaked in the early morning and late evening hours, with a dip in between that bottomed out at noontime and in the middle of the night. (Mislove, Lehmann, Yong-Yeol, Onnela, & Rosenquist, 2010)
- Number of Followers
- Number of Retweets
    - Individuals with a large number of followers whose tweets boast a high number of retweets tend to have a large online presence and field of influence. In addition, followers of a particular user will often agree with that user in their opinions and beliefs. Thus, if a particular user feels happy or sad about a particular event or person, their followers may feel the same way. As such, we feel that such individuals are representative of a larger portion of the population than individuals with fewer followers whose tweets are not retweeted as much, and we will give more weight to their tweets.

5. METHODS

To classify our data, we will use a model that can label any incoming tweet as being either positive or negative. We have formulated three approaches for the design of our model, and we will test the effectiveness of two of these approaches in our experiments.

5.1 Logistic Regression Approach

In this approach, we teach our model how to detect positive or negative tweets with prior knowledge. Using logistic regression, our model discovers the optimal weights for modelling a pre-labeled database of tweets that are known to be positive or negative. Then, when our training model is fed a tweet, it can compare the tweet contents with its knowledge base to determine whether the tweet is more likely to be positive or negative.

- The following definitions are used:
    - We defined our set of words of interest from two sources.
        - Our first source was the Opinion Lexicon, a database of 6800 words compiled by Bing Liu and Minqing Hu. This database of 6800 words is divided into two sets, one listing words associated with a positive sentiment and another listing those associated with a negative

sentiment. (Liu, Hu, & Cheng, 2005)
        - We also consider the most commonly-used emojis (which will be considered words for the purposes of this paper, as they usually capture the writer's sentiments), as listed by the emoji-emotion database. This database contains 118 emojis with polarity scores for each emoji (ranging from -4 to 4, with lower numbers signifying negativity). (Wormer, 2017)
    - We define an input data point to be the data from an individual tweet. It consists of the following variables:
        - Day of the week, represented in one-hot encoding (with one variable for each of the seven days of the week)
        - Location, represented in one-hot encoding (with one variable for each of our three considered location categories)
        - Temperature, represented as a float
        - Time, represented as a float
        - One variable for each word in our list of words of interest, represented as an integer denoting the frequency of that word in the given tweet
        - The number of followers, represented as an integer
        - The number of retweets, represented as an integer
    - We define two classes of tweets: positive or negative. The positive class is associated with the value 1, while the negative class is associated with the value 0.
    - We define a weight vector with weights corresponding to each of the variables. The initial values for the day, temperature, and time weights will be initialized by hand based on the studies stated in the previous section. We will also initialize some of the word weights based on our own judgments. Positive weight values mean that a variable is associated with happiness, while negative weight values mean that a variable is associated with sadness. There is no weight associated with the "influence variable."
    - The output of our model's fitting algorithm will be a float restricted to the range [0,1]. If the output for a given tweet is below 0.5, the tweet is put into class 0 (a.k.a negative). Otherwise, it is put into class 1 (a.k.a positive)
    - Our objective is to minimize the mean squared error function.
- The main hurdle to this approach would be acquiring a pre-labeled set of training data of adequate size. Creating our own pre-labeled set by reading tweets and using our own judgment to label them as happy or sad is too slow and would not yield a sufficiently-sized training set.

5.2 SVM Approach

In this approach, we create a hyperplane in a feature space that can classify tweets given their position in the feature space. To discover this hyperplane, we map a set of pre-labeled tweets into a feature space with a number of dimensions equal to the number of variables in an input data point. Then, we identify the support vectors in the feature space and run those points through a SVM (with soft margins) to calculate the hyperplane that best approximates the division of our pre-labeled data into positive and negative categories. We can use this hyperplane to classify any unlabeled tweets by mapping the tweet into our feature space and seeing which side of the hyperplane it falls on.

- The following definitions are used:
    - The words of interest, variables of the input data points and the weights are defined the same way as they are in the logistic regression-based approach
    - The negative class is now associated with the value -1 instead of 0 (the positive class is still associated with the value 1).
    - The output of our model will be a float. If this float is positive for a given tweet, we consider the tweet to be positive. Otherwise, we consider the tweet to be negative.
- Like the logistic regression approach, we require a pre-labeled set of data points to use this approach. Compared with the logistic regression approach, the SVM approach would take longer to train and would be more memory-intensive (especially since we are working with a large data set). However, once trained, it can classify incoming tweets more quickly and accurately.

5.3 K-Nearest Neighbors Approach

Like the previous approach, we map a set a pre-labeled tweets into a feature space. Unlike the SVM approach, however, we directly use the pre-labeled tweets to classify unlabeled data instead of creating a hyperplane. An unlabeled data is classified according to the majority vote of the k nearest tweets.

- The following definitions are used:
    - The words of interest and variables of the input data points are defined the same way as they are in the logistic regression-based approach
    - We define our distance function (to determine how close a given tweet is to another) to be the Euclidean distance function
- As with the two previous approaches, this approach require pre-labeled data. K-nearest neighbors is generally easy to implement and can be robust to noise, but the results can be skewed by certain variables that we do not place as much value in (such as temperature or location).

5.4 K-Means Clustering Approach

In this approach, we try to extrapolate what constitutes a positive tweet or a negative tweet without any external guidance. Like the previous two approaches, pre-labeled tweets are mapped into a feature space. However, instead of comparing each tweet with an external template, we try to discover "clusters" in our collected tweets that correspond to happiness and sadness, respectively.

- The following definitions are used:
    - The words of interest, variables of the input data points and the weights are defined the same way as they are in the logistic regression-based approach
    - We define our distance function between two points to be the Euclidean distance function
- As this approach is a form of unsupervised learning, we would not need a pre-labeled set of training data as we did with the logistic regression approach. However, our resulting model may not be as accurate. In addition, our training process could potentially take much more time than the logistic regression approach as we may perform many loops of the entire k-means clustering algorithm.

6. EXPERIMENTS DESIGN AND EVALUATION

In this section, we will describe our implementations of the logistic regression, SVM, and K-nearest neighbor approaches described in Section 4 (we have decided to not pursue the K-means clustering approach due to implementation difficulties and time constraints). Then, we will compare the approaches in terms of accuracy and running time to determine which approach appears to be more effective.

6.1 Experiment Design

Both the logistic regression and K-means clustering approaches were implemented with Python 3 in a Windows environment.

6.1.1 Logistic Regression Approach

Our training model keeps one copy of the weight vector defined in Section 4.1, which we will denote as w. It accepts a feature vector as input and calculates an output label using logistic regression.

As we were unable to find a pre-labeled set of tweets of sufficient size, and because labelling tweets by hand would be too cumbersome, we decide to automate the generation of a training set with the following method:

- We create a training model initialized with a weight vector containing all of the weights described in the weights definition in Section 4.1.
- For each word that originated from the Opinion Lexicon, we assign its corresponding weight variable in the weight vector to 1 if it was contained in the positive sentiment list, or -1 if it was contained in the negative sentiment list.
- For each emoji listed in the emoji-emotion database, we initialize its weight to its associated polarity score divided by 4 (to restrain it to the range [-1,1]).
- Then, we selected tweets that could be labeled from each of the 22 time periods covered by our data and ran these tweets once through our training model to obtain an output value for each tweet.
- If the output value was closer to 1, we labeled the tweet with class 1 (positive). Otherwise, we labeled the tweet with class 0 (negative).

We ended up with 12,772 positive tweets and 6,246 negative tweets, for a total of 19,018 tweets in our pre-labeled set.

We convert each tweet into a feature vector for our training model as follows:

- The tweet is cleaned and its information extracted using the procedure described in the previous section, yielding the tweet's text date, location, time, and weather information.
- We create a feature vector with all of the variables described in the input data point definition in Section 4.1. All variables are initialized to 0.
- For each of the following types of information: weather, time, date, and location, we set the variable corresponding to the information's category to 1.
- For each word in the tweet, we increment its corresponding word variable by 1 if it exists.

We trained and tested our model as follows:

- We split our pre-labeled set into training and validation sets using 5-fold cross validation.
- For each fold, we perform the following procedure:
- We create a weight vector with all of the weights described in the weights definition in Section 4.1. All weights are initialized to 0.
- We iterated the following procedure until convergence:
    - We run the tweets in the training set through the training model to obtain output values for each pre-labeled tweet.
    - Using MLE, we compare each tweet's output value to its actual label to determine how well our model predicts our pre-labeled set.
    - We use the Newton-Raphson method to update the weights to try to model our pre-labeled set better in the next iteration.
- We run the tweets in the validation set through our training model and record our fold's accuracy (the fraction of those tweets that were correctly predicted).
- We average the accuracies from each of the folds to determine how accurate our model is.

6.1.2 SVM Approach

Our training model accepts a feature vector and determines its output label using a hyperplane it learned from training.

We use the same pre-labeled set of tweets that was described in section 5.1.1. Tweets are also prepared for the training model in the same manner as was done in section 5.1.1.

We trained and tested our model as follows:

- We discovered the support vectors in our pre-labeled data set
- We discovered a dividing hyperplane with our support vectors using a SVM
- We split our pre-labeled set into training and validation sets using 5-fold cross validation to evaluate accuracy.

6.1.3 K-Nearest Neighbor Approach

For this approach, we set k = 5 (so we considered the 5 nearest neighbors when classifying a data point).

Again, we use the same pre-labeled tweets and input preparation method described in section 5.1.1. 5-fold cross validation is used to evaluate accuracy as was done with the SVM approach.

6.2 Evaluation

6.2.1 Model Testing

Listed below are the training data accuracies, 5-fold cross validation accuracies, and training time for each of our three models.

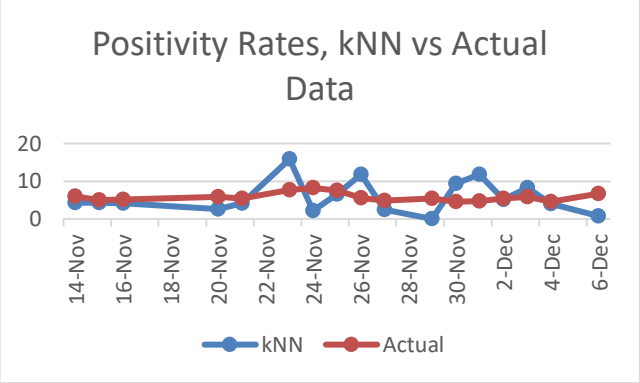
Model	Training Data Accuracy	5-fold Cross Validation Accuracy	Training Speed (in seconds)
LogReg	0.671574298033 44204	0.6458051732218 9579	~5
SVM	0.797139552003 36528	0.5583623747245 5641	~3
kNN	0.671889788621 30611	0.6412298379602 6023	~30

In terms of training speed, SVM performed the fastest, while kNN was by far the slowest. With regards to the training data, SVM was the most accurate while logistic regression was the least accurate. With regards to 5-fold cross validation, logistic regression was the most accurate while SVM was the least accurate. Overall, we feel that for our given data, the logistic regression and SVM models are fairly evenly matched, while the kNN model is not ideal.

6.2.2 Results

To measure how happiness and sadness changed over time, we first split our unlabeled data set by day. For each day's subset, we calculate the positivity rate (the ratio of the number of tweets classified as positive to the number of tweets classified as negative) from the testing data and from the data labelled by each of the three models. We also calculate the precision, recall, f-measure, and support for each subset. An excel sheet with these values for each data set is included with this report.

Below is a chart showing the positivity rates predicted by the kNN model compared to the positivity rates of the testing data. Certain dates (11/22, 11/29, have multiple subsets. In this case, their positivity rates were average together for the purposes of this chart.



While the kNN model produced reasonable output, the logistic regression and SVM models did not yield usable data on the positivity rates. The logistic regression model predicted a ratio of 1 for all dates, while the SVM model either predicted 1 or an extraordinarily high ratio (of at least 1000). Thus, the rates from the logistic regression and SVM models are not shown.

According to our testing data, the positivity rate peaked on November 24, with low points around the 11/16 and 12/4. However, the curve remained relatively smooth, suggesting that people's overall mood did not seem to change too rapidly with current events.

According to the kNN model, the positivity rate experienced high peaks around 11/23, 11/26, and 12/1, and dipped considerably on 11/24 and 11/29. In general, the curve predicted by the kNN model varied wildly and did not follow the curve predicted by the testing data. This curve suggests that people were heavily influenced by current events.

However, both charts seem to suggest that …

7. SCHEDULE

The timeline of our project is listed below:

- Designing and deploying a Twitter crawler to obtain tweets - Week 5-6
- Preprocessing the tweets - Week 6-7
- Implementing bag of words for converting tweets for model usage – Week 8
- Implementing models for both considered approaches - Week 8-10
- Running our models on data to obtain results - Week 10
- Analyzing the results in a report - Week 10

Our work was divided as follows:

Task	Persons Involved
Writing Crawler	Yuhai Li
Obtaining Tweets	Martin Verde/Richard Yu
Preprocessing Tweets	Yuhai Li/Martin Verde

Implementing Bag of Words	Taylor Caulfield/John Kim
Implementing Logisitc Regression	John Kim
Implementing SVM	Martin Verde
Implementing kNN	Martin Verde
Obtaining Results	Richard Yu
Writing report	Taylor Caulfield/ Richard Yu

8. RELATED WORK

Personal opinions and expressions have been examined and cross-referenced for information even before the advent of computers. For example, in the 5th century BCE, officials in the Greek city of Athens used the votes of its citizens to determine how popular a particular policy was. However, this process only came to be computerized in the form of sentiment analysis in the 1990s with the computational linguistic community's foray into text subjectivity analysis. With the advent of the Web, the field of sentiment analysis has exploded. (Mäntylä, Graziotin, & Kuutila, 2018) Since then, a multitude of different approaches to performing sentiment analysis have been developed, such as machine-learning based approaches and lexicon-based approaches (Kharde & Sonawane, 2016).

Many techniques have been developed and evaluated with respect to sentiment analysis on Twitter data. These include model-building methods such as Naïve Bayes N-modelling and Support Vector Machines. Initially, sentiment analysis on Twitter data was focused on binary classification, but it has since expanded into multi-class classification. (Kharde & Sonawane, 2016)

Sentiment analysis has many real-world applications, such as answering research questions, informing business practices, and improving user-application interactions. For example, sentiment analysis has been run on product reviews and feedback to obtain favorability data on various products. This data is then be used by business to improve their products or to formulate new promotional campaigns. It has also been used to identify abusive comments on social media websites for removal, as abusive comments often carry a heavy negative sentiment. (Kharde & Sonawane, 2016)

9. CONCLUSION

Our goal in creating these predictors was to create a model to predict a Twitter user's mood based on his or her tweet. With the model we create, we would then be able to feed in Tweets from a large number of users and find the mood of a large number of users. For our project specifically, we could use this data to observe how mood changes over time from before Thanksgiving to afterward. Furthermore, we could continue to use this model to observe how other major events affect the mood of people in an area.

While we were successful in creating a model using the kNN approach, the data from our SVM model and our Logistic Regression model did not yield a useable predictor. Perhaps these models could have given a successful predictor if we had included the words in the Tweet as another feature in our data. Originally, we had intended to include the words in the tweet as feature data using a "bag of words" method. Had we included this in our model, the linear classifiers would have been weighed by words included in a Tweet and their frequency. Ultimately, we chose not to use this approach, as creating the list of words we would consider significant would have taken too much time and space, as we have to scan through the tens of thousands of tweets and save the words. We would then have to scan through every tweet again to record their frequency. Furthermore, our feature vector would have been orders of magnitude larger than they were, which leads to much more time both training our models and predicting moods.

When looking through our collected data, we noticed that all of the Tweets had a retweet count of zero. While it is common for Tweets to have zero retweets, it is impossible that every one of the thousand Tweets we collected had zero retweets. We discovered this was because we were only collecting streaming data, and each Tweet initially has zero retweets. We believe that if we were to collect older tweets that contain retweets, we would have been able create more robust sentiment predictors.

10. References

Bollen, J. (2011). Modeling Public Mood and Emotion: Twitter Sentiment and Socio-Economic Phenomena. *ICWSM.* Retrieved from https://dl.acm.org/citation.cfm?id=2436274

Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM, 56*(4), 82-89. Retrieved from https://dl.acm.org/citation.cfm?id=2436274

Kharde, V. A., & Sonawane, S. S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. *International Journal of Computer Applications, 139*(11), 5-15. Retrieved from https://arxiv.org/ftp/arxiv/papers/1601/1601.06971.pdf

Liu, B., Hu, M., & Cheng, J. (2005). Opinion Observer: Analyzing and Comparing Opinions on the Web. *Proceedings of the 14th International World Wide Web conference.* Chiba, Japan.

Lucas, R. E., & Lawless, N. M. (2013). Does life seem better on a sunny day? Examining the association between daily weather conditions and life satisfaction judgments. *Journal of Personality and Social Psychology, 104*(5), 872-884. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3695376/pdf/nihms476943.pdf

Mäntylä, M. V., Graziotin, D., & Kuutila, M. (2018). The Evolution of Sentiment Analysis - A Review of Research Topics, Venues, and Top Cited Papers. *Computer Science Review, 27*, 16-32. Retrieved from https://arxiv.org/ftp/arxiv/papers/1612/1612.01556.pdf

Mislove, A., Lehmann, S., Yong-Yeol, A., Onnela, J.-P., & Rosenquist, J. N. (2010). *Pulse of the Nation: U.S. Mood Throughout the Day inferred from Twitter*. Retrieved from https://mislove.org/twittermood/

Roberts, K., Roach, M. A., Johnson, J., Guthrie, J., & Harabagiu, S. M. (2012). EmpaTweet: Annotating and Detecting Emotions on Twitter. *LREC.* Retrieved from http://www.hlt.utdallas.edu/~kirk/publications/robertsLREC2012_2.pdf