# Can We Predict Whether a User Will Retweet the Personalvictory Hashtag?

*Alexander Egorenkov*

*Wednesday, August 12, 2015*

## Summary

We use a predictive modeling approach to find Twitter users who may be likely to retweet the #personvictory hashtag in order to infer characteristics about the users of www.personalvictories.com. The site reposts Twitter statuses that contain the #personalvictory hashtag.

We use the Twitter API to collect information about Twitter users, including posters, retweeters, and random users. From this information we predict whether a given user will retweet the hashtag. Furthermore, we can find characteristic among users that make them more likely to retweet the hashtag.

## Data Summary

We collected 3,672 #personalvictory tweets posted from 2009 to 2015. To our knowledge, this is the entirety of relevant tweets. From 2009 to 2015 there were only 285 users who retweeted #personalvictory, giving #personalvictory tweets at 7.7% chance of being retweeted. We also collect 1,014 random users with some constraints to make them comparable to the retweeters. For the total set of users we collected 126,211 tweets to gain a sense of their writing style and typical content.

## Data Collection

The data were collected using a mix of manual search and data access through the Twitter API. The data collection process can be broken into the following steps: manual search for #personalvictory statuses through the Twitter website, collection of user IDs for retweeters of the statuses, generation of random user IDs, collection of user data for the collected IDs, and finally, the collection of the 200 most recent tweets for every user in our sample.

### The Twitter API

Twitter maintains two APIs, the streaming API, used to process incoming tweets in real-time, and the search API, used to access resources created in the past. Since there are only one or two #personalvictory tweets per day, data collection through the streaming API would take too long for the scope of this project. Instead, we rely on the search API to access resources from 2009 to July 2015.

We primarily used four resources from the Twitter search API:

- GET application/rate_limit_status (180 requests/15 minutes)
- GET statuses/retweets/:id (15 requests/15 minutes)
- GET users/lookup (180 requests/15 minutes)
- GET statuses/user_timeline (180 requests/15 minutes)

The resource, application/rate_limit_status, was used to control the pace of the data collection process to avoid exceeding Twitter's rate limits. This same process can be used to search for new users who may be likely to retweet #personalvictory tweets. The resource, statuses/retweets/:id was used to identify retweeters of #personalvictory tweets. Statuses/retweets/:id will not return more than the 100 most recent retweeters, but in this case, no status had more than 100 retweets. Users/lookup finds common information, such as the number of followers, about a user when given a user ID. This information is largely unused in the project, but can be used to improve the accuracy of the predictive algorithm in future iterations. The most used resource is statuses/user_timeline which returns a user's 3,000 most recent tweets, 200 tweets per request. The bulk of the data in this project comes from GET request to this resource.

## The Data Collection Process

### Manual Search for #personalvictory statuses

To get the initial listing of statuses, we simply used the built-in search on the Twitter website using the search term "#personalvictory". This search return statuses with case-insensitive inclusions of the hashtag. To avoid looking only at popular tweets, we collected data from "live" listings instead of "top" listings.

The web page returned by the search contains the status, status ID, user ID of the poster, number of times the status was favorited, number of times the status was retweeted, time the status was posted, as well as other potentially useful data such as the poster's avatar. The retrieve this information we used a simple HTML parser. This process can't be automated any further as it would breach Twitter's current terms of service.

### Retweeter IDs

It is possible to collect IDs of retweeters once you have access to the status IDs. This can be difficult to do in real-time since the rate-limit for this resource is quite low. Even in this initial project, the retweeter IDs became a bottleneck. To take full advantage of the Twitter API you need to use non-blocking code that takes advantage of either multithreading, message passing, or asynchronous dataflow.

### Random IDs

To generate random IDs we took the the minimum and maximum user ID number from the list of posters. With this range established, we generated uniformly random numbers between the minimum and maximum. The assumption is that user IDs are generated sequentially as users sign-up; as is the default behavior in Twitter's web framework of choice, django. This helps assure us that the predictive algorithm works based off of relevant information and not merely account creation date.

It should be apparent that a random ID is not guaranteed to return a user. Only about 30% of the random requests actually returned a user, but there is no discernable pattern and we believe our sample is random.

### User Data

With a set of user IDs, it is possible to collect user information. The resource returns the account creation date, number of tweets favorited, number of tweets retweeted, number of followers, number of friends, the most recent user tweet, various geographic information, and various profile specific information. A full description of the resource can be found in the Twitter API documentation.

While all data was preserved during the collection phase, great care was taken to prevent leakage, the unintentional spilling of predictive information into the algorithm training process. In this case, it was important to assure that no random users where also retweeters.

### User Timeline Data

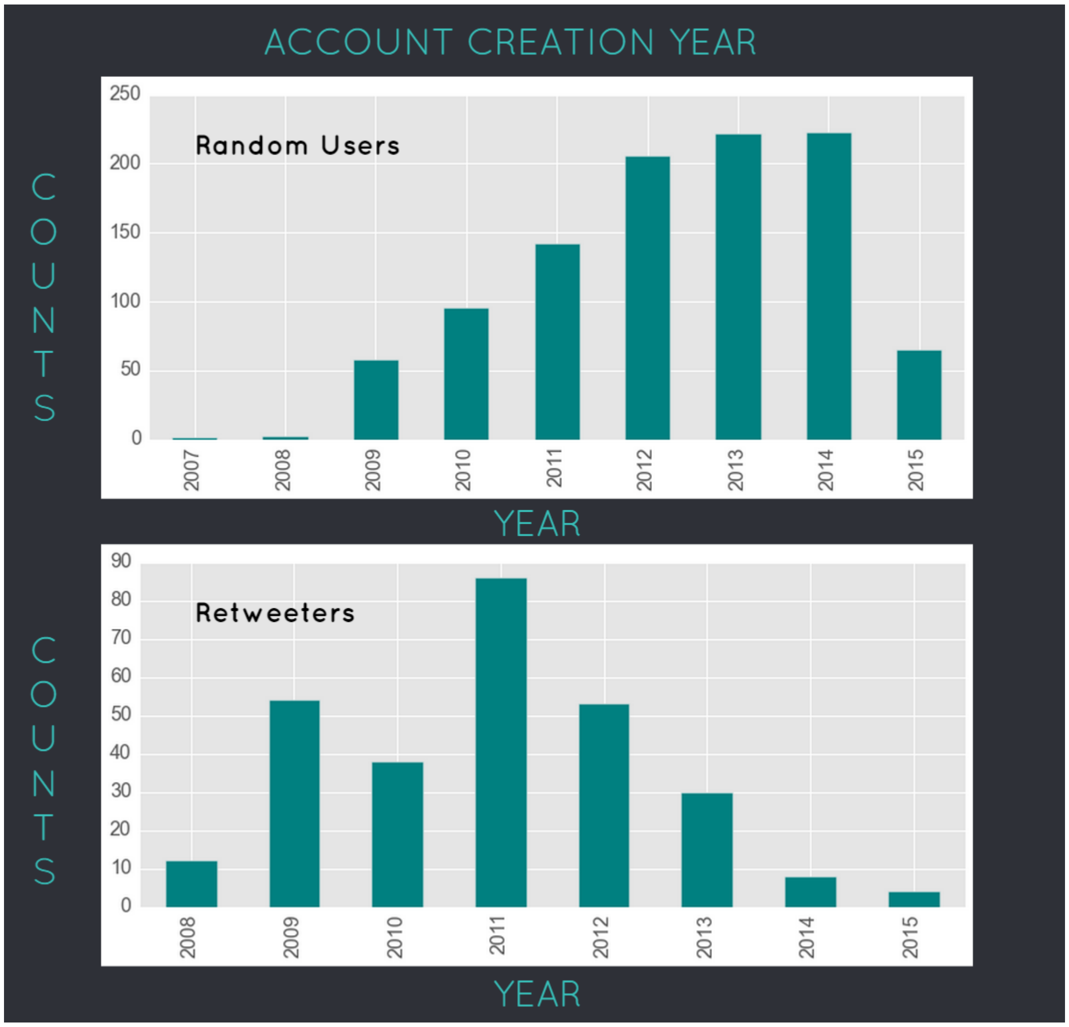
The user timeline resource, when given a user ID, returns information about the 200 most recent statuses. This information includes status text, creation date, links in the tweet, hashtags in the tweet, how many times the tweet was retweeted, and various other information that can be found in the Twitter documentation. We primarily only use the text data from this resource.

Here we also address potential leakage issues. We eliminate statuses that are themselves #personalvictory statuses and at times we eliminate retweets altogether to test the robustness of the algorithm.

## Data Exploration

### Account information

We start our exploration by looking at a handful of user characteristics to get a sense of what characteristics the predictive algorithm will use.

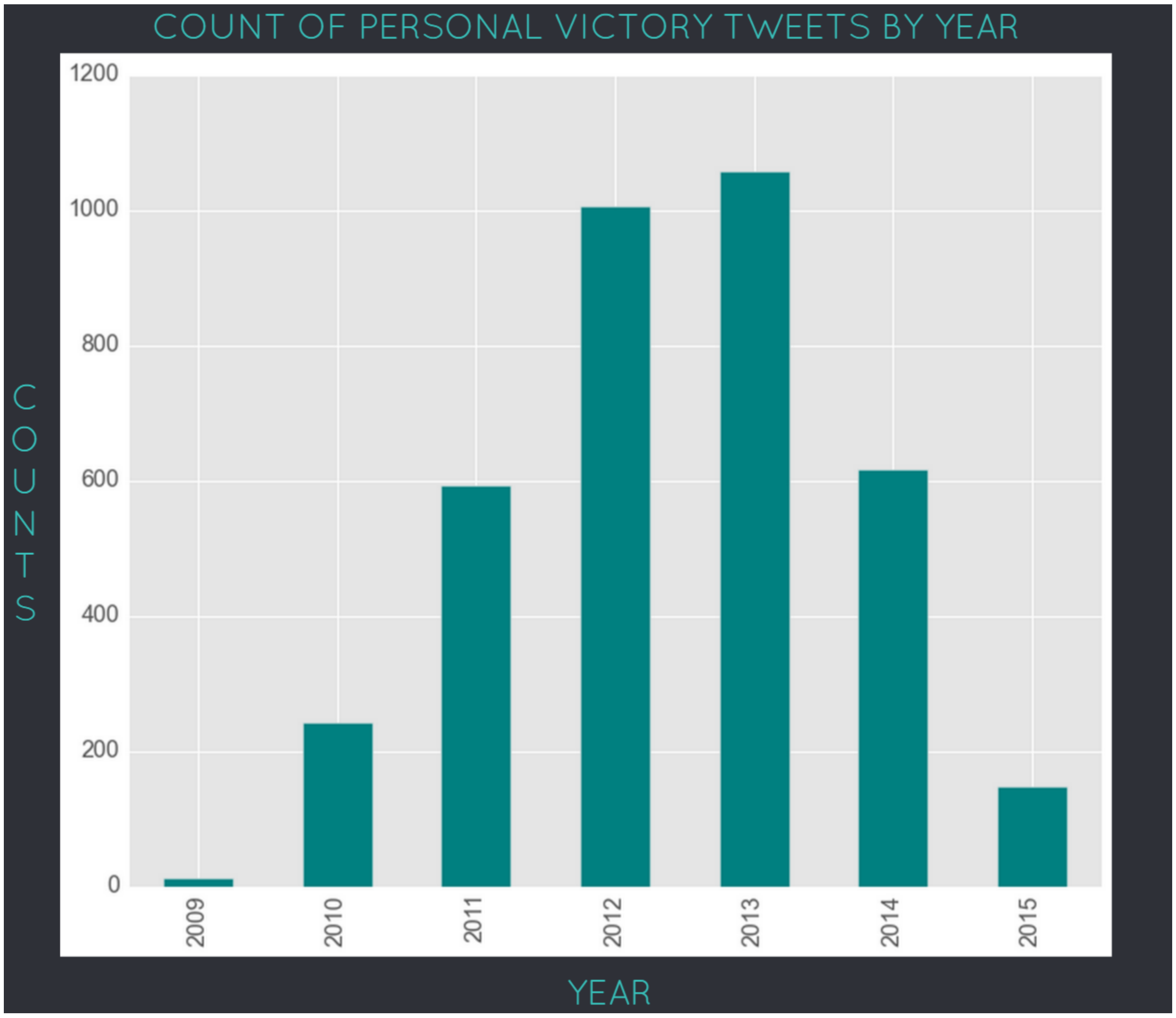


Random user account creation and retweet account creation varies slightly. This is because random users represent the general Twitter trend, while it is likely that retweeters were more active when the #personalvictory hashtag was more popular. It's possible that the predictive algorithm will use this information by associating other hashtags that were popular in the same time period.

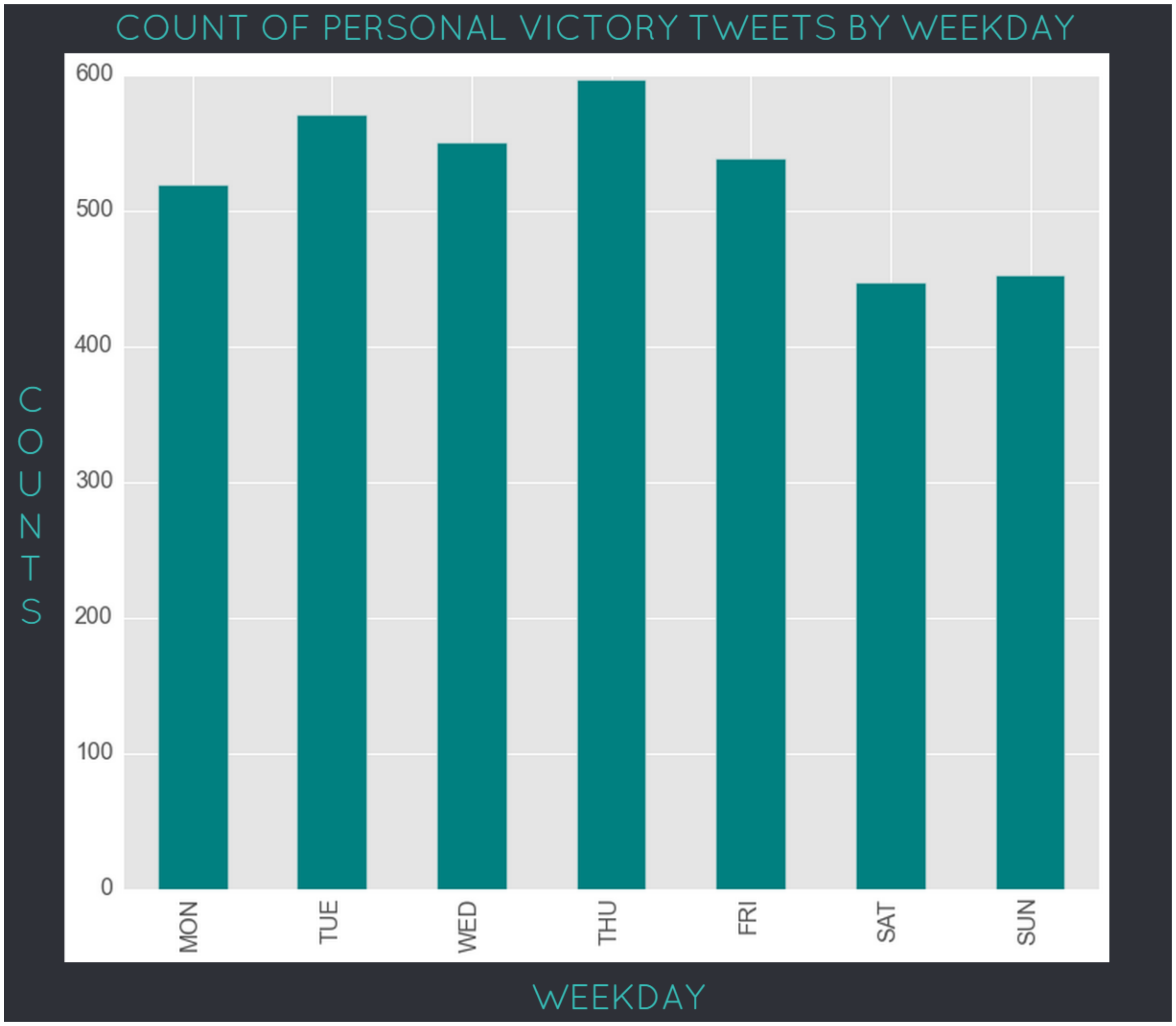
| User Type | Favourites Count | Followers Count | Friends Count |
|---|---|---|---|
| Random | 241.706070 | 182.233909 | 97.958595 |
| Retweeter | 4600.053381 | 2056.151408 | 1421.263158 |

Random users are generally much less active than the retweeters, this seems to be both because retweeters are more active in general and by definition; if you have at least one status you will be more active than a completely inactive account. It's unsatisfying to predict on account activity, so we try several version of the predictive algorithm where we remove inactive user, users with too few friends, or users with too few posts. Generally, the accuracy of the algorithm is unaffected as we vary these parameters.

## Tweet information



The pattern of how many #personalvictory posts were made and have many retweeting accounts were created in a given year seems to be related. It's also worth noting that the use of the hashtag seems to be declining. We do not look at general Twitter trends, so we do not know if this reflects a decline in Twitter use.



It seems that more #personalvictory posts are made during the weekday. This is intuitive when you read the statuses. Many personal victories are about work related success, monetary discounts on mid-day treats, and happy hour.

The most common tokens once we remove common english stop words in #personalvictory messages are:

| | | |
|---|---|---|
| personalvictory | work | going |
| just | went | people |
| today | actually | don |
| got | rt | finished |
| time | know | home |
| com | night | way |
| finally | right | instagram |
| didn | lol | won |
| day | week | think |
| ve | class | year |
| http | yes | run |
| twitter | morning | minutes |
| did | proud | took |
| like | managed | win |
| pic | getting | love |

| | | |
|---|---|---|
| good | feel | game |
| tonight | lost | |

We find this set of tokens representative of what we found while reading the statuses. #Personalvictory is included in every tweet by definition. Many of the keywords are related to just having an event take place such as “just”, “today”, “morning”, or “tonight”. The tokens “work”, “class”, and “run” relate to the most common topics of #personalvictory tweets. A handful of tokens relate to the posters sense of relief of amusement. Finally, “http” just gets read in from the html representation of the hashtag.

## Preprocessing and Feature Selection

Our goal for this project is to demonstrate that an effective predictive algorithm can be made for the task at hand using a simple and mostly transparent approach. To do this we decide not to include most of our data and focus only on the feature we find most applicable and interesting, text. For every user we collected up to the 200 of their most recently posted tweets. We essentially treat this wall of tweets as a single document associated with each user. Eventually, we transform these document into a set of keyword counts associated with each user, also known as a document term matrix. Using these keyword counts, our predictive algorithm can classify whether a user is random or a retweeter. Ultimately, we decide not to use keyword counts and instead just look for the presence of a keyword on a user’s timeline, an approach that we found more effective.

We also took some care to avoid leakage. In each document, we eliminate any tweet that has a status associated with a #personalvictory tweet or contain the hashtag itself. We also experiment with excluding retweets altogether.

Finally, within the modeling processes we tuned pre-processing to find an ideal document term matrix. We found that the ideal case was to break a document into less than 150 tokens, test for token presence rather than token counts, remove common english stopword, and remove all tokens that show up less than 15 times in our entire sample. We also consider pairs of words that occur commonly and include them as potential tokens, but largely they are unused by the predictive model. We also use a version of this matrix where we allow several thousand tokens, that particular model does not perform as well, but produced tokens that are easier to comprehend.

## Modeling Process

We only focused on one simple model, Naive Bayes, which learns the probability of a user being a retweeter given that they used a particular keyword, and then uses the probabilities for all keyword to determine which class a user falls into. There are several other appropriate models that could be used, but Naive Bayes is often used for text problems and is generally a good start with a simple interpretation. We could further enhance the accuracy of our model by using various ensembling techniques and text processing, but we achieve a good result without sacrificing interpretability.

We made model selection choices within a train/test split framework. To make sure that our model will generalize to new data, we train it on only 80% of the data we have on hand and hold out 20% for testing. A model that generalizes well will perform about as well on the testing data as in the training data.

To determine how well a model performs during a test we use the accuracy and area under curve metrics. Accuracy simply counts whether we correctly determined whether a user was random or a retweeter and divides by the number of total user in the test sample. The failure of accuracy in this case is that it does not account for class imbalance. Only 22% of our sample is made up of retweeters, so a model that merely guesses the most likely class will have an accuracy of 78%. The AUC (area under curve) is a broader metric that is not affected by class imbalance.

On each test, we adjust the pre-processing phase and some parameters of the Naive Bayes algorithm to find a combination that works well.

## Results

Our final model achieves an accuracy of 92% and an AUC of .94, both of which are a significant improvement over the null model that merely guesses the most common class. In fact, this score is so good, that we purposely tried to give the model several unfavorable conditions to make sure we weren't leaking information. Specifically, we removed all retweets from the sample and we removed users who had few friends or were likely to be inactive. These modifications greatly reduced sample size, but the algorithm performance was only minorly affected.

A secondary goal was to find a document term matrix which contained predictive features. In our best performing model, the 50 most discriminative tokens were:

| | | |
|---|---|---|
| http | want | thanks |
| rt | ve | come |
| https | think | work |
| just | make | game |
| like | best | way |
| amp | night | let |
| love | got | man |
| don | really | video |
| day | happy | follow |
| time | need | gt |
| new | going | world |
| today | great | la |
| good | ll | live |
| people | lol | en |
| know | right | que |
| http rt | tonight | el |
| life | thank | |

These tokens show the words that are most likely to be used by retweeters while simultaneously are the least likely to be used by random users. Overall, we don't find this list to be intuitive and would need to take a closer look at the original data to determine the appropriate context. The one except is that the "http", "rt", "https" are associated with retweets and one would reason that user who have a habit of retweeting are more likely to retweet a #personalvictory status. If we eliminate all retweets from the sample, we get back a very similar list of tokens except for these three tokens.

Some models that displayed slightly worse performance gave a much more intuitive document term matrix, we show the top 30 discriminative tokens:

| | |
|---|---|
| lmao | drive |
| drunk | basically |
| makeup | fitness |
| entire | af |
| point | relationship |
| missing | energy |
| literally | opportunity |
| pictures | fault |
| thoughts | interesting |
| sports | mind |
| thankful | extra |
| fell | eat |
| automatically | pretty sure |
| tbt | gun |

| | |
|---|---|
| conversation | glad |

## Extensions

We can use this approach for customer acquisition, by searching for prospective readers. The API terms are broad enough to allow this use and much of the webscraping framework is already prepared. The predictive algorithm needs to be further tested, but can be used in combination with the scraper to identify users. With some work the model may be able to work outside of Twitter and find prospective users through forum postings, reddit, blog comments, and other online media. However, this does needs great care because of the unique form of Twitter statuses and the general difficulties that are presented when changing the context of a predictive algorithm.

The predictive algorithm itself can be greatly improved with fine tuning and ensembling technique.

We can further explore user characteristics to guide our notion of who these users are and what kind of products they may be interested in.

We can extend beyond retweeters and look at friends, followers, and the poster themselves. The most directly useful information comes from understanding the retweeters because they have read the personal victory status and have demonstrated interest in the status, but the other users may offer information about personal networks and how information propagates.

Similarly, we can find central personalities responsible for having personal victory messages propagate. One example is @girlposts who was highly retweeted.

We can reverse the approach and attempt to find traits associated with the most popular personal victory statuses. It's likely that popularity is mostly based on the user's followers and friends, but it may be possible to find more generalizable information.

We can attempt to cluster or group statuses or users. This also seems unlikely because there were only 285 retweeters, which is not a great deal of information.

## Conclusion

We can successfully predict which users are likely to retweet a #personalvictory status. We suspect that these users would also be interested in similar content.

Despite the apparent success of the predictive algorithm, we feel there needs to be more testing before putting it in any type of production setting. Similarly, this report largely glazed over in depth exploration of the users. There is a wealth of data collected that can be used to answer the broad question "Who is likely to read www.personalvictories.com?"