Learning about Machine Learning: An Extended Assignment to Classify Twitter Accounts

Eni Mustafaraj and **Scott D. Anderson**
Computer Science Department
Wellesley College
Wellesley, MA 02481
emustafa@wellesley.edu
scott.anderson@acm.org

Abstract

We describe a four-week series of assignments in an undergraduate AI course at a liberal arts college developing a supervised learning solution to the problem of classifying Twitter accounts as either a person account or a non-person account (e.g. organization or spambot). This problem employs real data in an ongoing research project by the first author, yet is accessible to students with limited programming expertise. The students were able to experience a complete cycle of creating a machine learning solution: exploring raw data, creating a training set, engineering features, comparing different classifiers, evaluating the results, and performing error analysis. We received positive feedback from the students and intend to refine the assignment and make it available (together with the created training data) for use by the research community.

Introduction

The Artificial Intelligence course offered in our liberal arts college is an elective, intermediate-level course for students majoring in computer science, neuroscience, and cognitive science. Students are expected to have completed at least two courses in computer science: one course in programming concepts (CS1) and one in data structures (CS2). The diverse audience of students (2 sophomores, 5 juniors, 4 seniors) and their non-equal programming background and skills makes it difficult to develop a course with a focus on the implementation of algorithms.

Our solution was to pursue a mixed approach that included reading papers, understanding the working of algorithms, implementing parts of them, and using existing packages to test how algorithms work to solve concrete problems. We decided to teach Python to the students (five of them didn't have previous exposure), in order to make use of the Python based library NLTK (`http://www.nltk.org/`) and the excellent book that explains its use for natural language processing (Bird, Klein, and Loper 2009) and other AI related topics (such as reasoning or automatic theorem proving).

An early decision in creating this AI course was to emphasize machine learning. Core AI topics (vision, robotics, games, and natural-language processing) were surveyed in the first four weeks, but the rest of the course was built around the discussion of machine learning algorithms and ways to incorporate them into the students' final projects. We were guided by two goals: 1) give the students a complete experience of building a machine learning solution to a problem, and 2) make this experience part of an existing, larger research project so that the students had the satisfaction of contributing to something real.

We designed a series of assignments and in-class activities to explore different concepts and build up confidence, so that students would be willing to tackle increasingly difficult challenges from week to week. An example of a concept introduced early in the course is the problem of acquiring labeled data needed for supervised learning. For example, we discussed a classifier to predict the gender of an author (Koppel, Argamon, and Shimoni 2002). The students were instructed (as part of the first assignment) to think of how to collect other corpora that would enable the learning of such a gender classifier and possible new uses for it. The challenge was to find ways to gather examples of writings (by female and male authors) together with the correct label.

Because advancement in new supervised learning tasks is often hindered by the lack of training data, a few publicly-funded organizations — such as NIST (`http://trec.nist.gov/`) or PASCAL (`http://pascallin2.ecs.soton.ac.uk/`) — have invested in the creation of high-quality training sets and made them available to researchers who participate in challenges designed to measure the success of new approaches. Other organizations, such as DARPA, use challenges to engage the research community in tackling new and difficult problems. We discussed the DARPA Urban Challenge (`http://www.darpa.mil/grandchallenge/`) as well as PASCAL's Visual Object Classes challenge (`http://pascallin.ecs.soton.ac.uk/challenges/VOC/`). In this way, students became familiar with the notion of using challenges to spur research in an area, as well as the need for a common corpus that is used by all research groups to test their algorithms.

During the semester, our students performed a series of activities inspired by these challenges: exploring whether a supervised learning algorithm can learn to determine whether a Twitter account belongs to a person. In the following sections, we detail the different phases of this challenge. A summary of the timeline is as follows:

phase	day	
1	0	Explain Twitter and its data format
2	3	Task: update Wiki with proposed features
3	4	Explain research context in class
4	6	Task: labeling tweets as to person vs non-person
	11	Explain decision trees in class
	11	Due: manually labeled tweets
5	12	Task: determine features to classify tweets
6	20	In-class competition for best accuracy with decision trees
7	27	Explore decision trees and data analysis using Orange
8	33	Task: teams submit analysis and evaluation of features and results

Phase 1: Twitter

There are several reasons why Twitter has attracted the attention of researchers. Unlike Facebook (the largest social networking website), Twitter is about broadcasting thoughts, information, and ideas in 140 character chunks called "tweets" to whoever happens to listen. Therefore, Twitter does not enforce reciprocal ties as Facebook does, Twitter users do not have the privacy expectations of Facebook users, and most Twitter users have public profiles and streams of tweets that can be inspected by everyone. Twitter's membership has been growing steadily and the volume of daily tweets amounts to several million (Paczkowski 2010). However, the most appealing aspect of Twitter is its powerful and very easy to use APIs, which allow easy access to all public data and content that are generated daily by Twitter users. Twitter does not have restrictions on what developers choose to do with this content, and therefore a very rich eco-system of third-party apps has flourished around this data, offering new and better ways to engage with it.

On the other hand, Facebook provides better user profiles than Twitter. Facebook users are interested in creating accounts that reflect their real-life identity, in order to be recognized by their social groups. Additionally, Facebook distinguishes between pages belonging to individuals and pages belonging to groups, organizations, companies, events, etc. In Twitter there is no such distinction.

Data Collection

The data used for this assignment was collected on August 13, 2010, using the Twitter Streaming API. Over 20 hours, a script collected approximately 3.4 million tweets, containing some of the most frequent English functional words ('of', 'the', 'to', 'and', 'this', 'that'), to ensure the acquisition of English-language tweets. We then randomly selected 12,000 tweets to create a training and testing set.

Twitter Data

The tweets collected through the Twitter API are formatted as JSON objects that contain several ⟨ attribute, value⟩ pairs. Some of these attributes describe the tweet itself and others the account from which the tweet was sent. Every tweet in our corpus has 45 ⟨ attribute, value ⟩ pairs. Table 1 shows some examples.

Phase 2: Proposing Features

In one class session, we provided a brief description of Twitter, its API, and types of Twitter accounts. Afterwards students were given the following task: explore random Twitter accounts and describe five features that can be used by a machine learning classifier to distinguish between accounts of different types. As a background reading on understanding features, we assigned section 6.1 and 6.2 from (Bird, Klein, and Loper 2009). Students worked individually on this task, but once they completed it, they posted their features to the class wiki. Some of the features they discovered and the explanations are shown in Table 2.

Phase 3: Modeling Users of Twitter

The first author is involved in a larger research project that studies the relationship between the Web and politics in the United States. That project has shown that Twitter can have a profound effect on U.S. politics, such as being instrumental in the election of Senator Scott Brown of Massachusetts. Other research studies have shown that Twitter can be used for such tasks as predicting elections (Tumasjan et al. 2010), inferring the political bias of media outlets (Golbeck and Hansen 2010), extracting the sentiment of public opinions (O'Connor et al. 2010), or predicting changes in the stock market (Bollen, Mao, and Zeng 2010).

Our research on U.S. politics is hindered by the fact that Twitter does not distinguish between accounts belonging to real people versus corporations or spambots. If we want to use Twitter data for insight into voter opinions, we need to distinguish these different kinds of Twitter accounts. Machine learning can help with this. We start with separating accounts belonging to persons from non-persons. We can then apply other machine learning algorithms to try to infer demographic features such as age, gender, and political orientation. Discussing this larger research context in class went a long way towards motivating the students on the value of machine learning.

Phase 4: Data Labeling

Data exploration while preparing for these assignments indicated several things: 1) there are many more tweets from persons than non-persons, 2) the class of non-persons is difficult to define, 3) our corpus contained foreign-language tweets (despite the filtering for English words), 4) some of the accounts were suspended (our data was collected in August and the labeling was done in October). This led us to a set of eight different labels: person [p], private account (a person who doesn't allow access to the tweets) [pa], spammer [s], organization [o], commercial entity [c], foreign account [f], not existing (suspended by Twitter) [x], and "don't know" (if we were not sure how to label an account) [d]. During a lab session, the students started labeling accounts, in order to achieve a common agreement on how to apply labels. Students were instructed to continue the labeling

Attribute	Value
text	Give your pet the best in joint health with Kenzen Pet Joint ... http://bit.ly/cuMXvw
created_at	Fri Aug 13 08:09:19 +0000 2010
source	twitterfeed
statuses_count	7988
description	Pet owner interested in natural health treatments for our pets. No chemicals and drugs
friends_count	2239

Table 1: Some ⟨attribute,value⟩ pairs from a tweet. The top three pairs belong to the tweet description, the bottom three pairs to the account description.

Feature	Feature Explanation
Repeating patterns in tweets	a lot of non-person twitter accounts, especially newspapers, have tweets that all follow a pattern: a short description followed by a link
Bio description	People seem to have bios that are a list of descriptive phrases about themselves, and are not very formal, whereas companies, or people posting in a more official capacity (reporters or something) have bios that are longer, and tend to be formal sentences
Topic analysis	People do different things every day, whereas bots or corporate accounts only tweet about one topic (i.e. love, football, their product, etc)
Internet language	'LOL', 'lulz', 'ppl', 'u r', emoticons, will more commonly be used by people
Links	If tweets consistently link to one website and interact very little with other twitters, then it's probably not a person. (Matches the pattern of newsfeeds.)

Table 2: Features that students discovered by analysing Twitter accounts in the exploratory phase.

	person	non-person	person%
Training	5451	1773	75.45
Testing	618	206	75.00

Table 3: Distribution of accounts by class

outside class and whenever possible to share the task with friends.

As a common application and database for the labeled tweets, we gave the students a Google Spreadsheet with links to the Twitter accounts and a drop-down menu to choose the labels. This procedure involved five clicks and wasted time, inspiring one student to write a web-based app where the whole list of accounts is imported, the content of the Twitter account is displayed on one part of the screen for inspection, and labels assigned with one click. The tool is open-source and can be re-purposed for other Twitter-related data labeling.

In total, 8500 accounts were labeled by students and 1000 by a research assistant. The first set was compiled by the instructor and redistributed to all the teams as training data. The second set was used for testing; students did not have access to it while they engineered features for learning. From the labeled data we removed the instances labeled as "foreign accounts" and "don't know." Because we wanted just two categories, we mapped [p] and [pa] to "person," and the remaining labels to "non-person." The distribution of accounts by class for the training and test data is shown in Table 3.

Phase 5: Feature Engineering

The students were given the training set of raw Twitter data and instructed to do two things: 1) write feature extractor functions to transform raw data into feature vectors that can be used by a supervised classifier, and 2) write a function to calculate the information gain of a feature. Then, they could use the information gain function to select the ten best features that will ultimately go into their training set of feature vectors. Students worked in pairs to complete this assignment and used Python to code feature extractor functions. An example feature extractor function written by students is shown in Figure 1.

In designing this feature extraction task, we wrestled with the decision whether the students should implement their own version of the decision tree learning algorithm, but eventually decided to use the available implementation in the Orange suite (http://www.ailab.si/orange/). Orange is an open source data visualization and analysis platform that can be used by novices (through its visual interface based on widgets) and experts (through Python scripts). During the competition, we used its scripting capabilities to access the implementation of the decision tree classifier (a modified version of Quinlan's C4.5, written in C++ for efficiency, but wrapped in Python).

Phase 6: In-class Competition

All teams had to submit their feature extraction code to a designated server where the training and testing files (with raw data) were residing (but the teams could not access them). We had written several scripts that did the following: a) visited every team drop folder to find the latest submitted

```
def location_feature(tweet):
    """
        Checks whether a user has a
        value 'location.' Accounts
        labeled 'p' are more likely
        to have defined this value.
    """
    location = tweet['user']['location']
    hasLocation = "false"

    if location != None:
        hasLocation = "true"

    # Return feature_name : value
    return {'location': hasLocation}
```

Figure 1: Sample student feature extractor function

file and executed it to create a training and testing set with the team features, b) learned the Orange tree classifier from the training set and evaluated it on the testing set, and c) sorted the accuracy results of all teams and projected them on the board. We noticed the following issues in class:

- one team had engineered features that gave 95% accuracy on the training data, but only 75% on the testing data. We had the chance to discuss overfitting and how important it was that the teams did not have access to the testing data.
- one team had written code that excluded a part of the training and test examples programmatically and that caused their accuracy to be 12 percentage points higher than that of the other teams. We had to change the code for evaluating the accuracy to reflect the real size of testing data.

During class time, teams continued making changes to their code to increase their accuracy score. Whenever a team would submit a new version, there was general anticipation of changes in the leader board and that seemed to create an enjoyable atmosphere. We allowed the students to continue this process of iteration in the following days by automating the process of submission, execution, and accuracy score comparison.

Figure 2 shows the accuracy one week after the competition.

Phase 7: Explore Orange Trees

In the week following the competition, we spent a lab working with the visual programming tools offered by Orange for evaluating classification results. For example, a scenario like the one shown in Figure 3 (and discussed in the next section) performs the following steps: 1) the training data are fed to the two different classifiers — a majority classifier and a classification tree; 2) the test learner uses the learned classifiers to evaluate their accuracy either on the training data or on the testing data; 3) the classification results can be further explored in the confusion matrix.

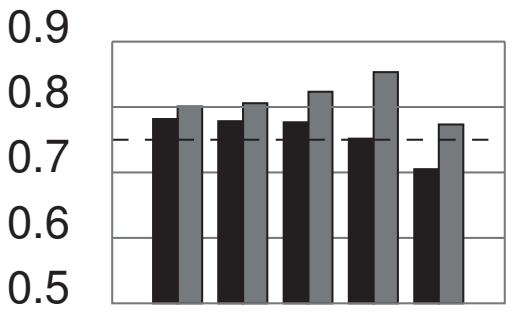


Figure 2: Accuracy of the five student teams on the training data (gray bar) and the testing data (black bar). Teams are ordered based on the accuracy on the testing data. Baseline accuracy is 75%.

Phase 8: Analysis and Evaluation

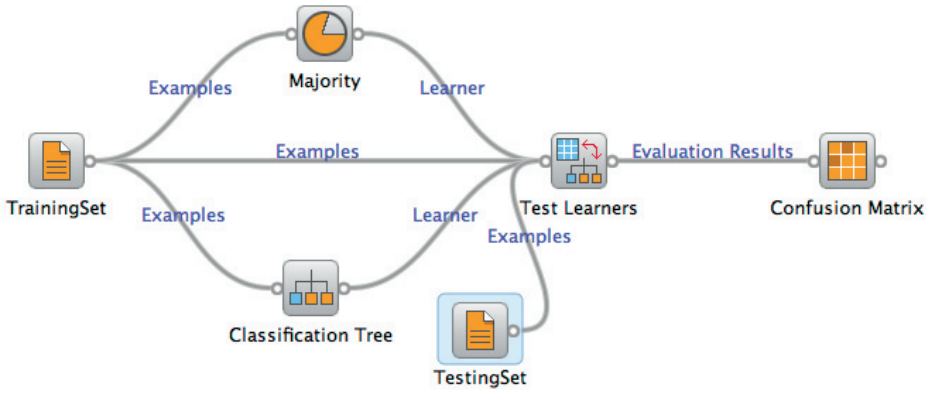
As a last part of the assignment, the students were instructed to perform the following tasks:

- compare the accuracy of the tree classifier learned from their feature vectors with that of the majority classifier
- compare the information gain values calculated by their code with that calculated by Orange. In this way, they could make sure that their code was correct. If not, they would need to revise the code
- compare the values of information gain and gain ratio and understand why it is better to use the latter as a splitting criterion
- inspect 50 classification errors and make a hypothesis about the cause of errors

By comparing their learned classifiers with the majority classifier (which assigns the label of the majority class to every instance — for this data, it labels every tweet as a "person," and is right 75% of the time), students were able to see the gain from learning. This was made clearer by displaying the results in a confusion matrix, as shown at the right of Figure 3. While the classifier does a good job of classifying person accounts (96.7% correctly classified), it has certainly not learned well to classify non-person accounts (72.6% of non-persons were incorrectly classified as persons). The error analysis allowed the students to investigate potential reasons for this failure. One of the most intriguing hypotheses, which we are currently investigating, has to do with the suspended accounts. Our training corpus has 782 such accounts, which we labeled as non-person, under the assumption that Twitter suspended them for spamming activities. However, the messages by these accounts that are in our corpus give the impression of being written by persons.

In fact, a recent study focusing on detecting spam in Twitter (Yardi et al. 2010) concluded that spammer accounts are very sophisticated and are able to mimic with high credibility legitimate user accounts. We think that this is what our data is showing and that more ingenious features and more data might be necessary to resolve the issue.

Another hypothesis has to do with the subjectivity in the labeling process, especially since this was a collaborative



	np	p	
np	486	1287	1773
p	181	5270	5451
	667	6557	7224

	np	p
np	27.4	72.6
p	3.3	96.7

Figure 3: Comparing the classification results between a tree classifier and the majority classifier.

labeling effort. Some students thought that many examples were labeled incorrectly, and that this was a reason for errors in the learning. We were able to measure Cohen's kappa coefficient for two annotators (it captures the inter-annotator agreement) and that was 0.84. While this shows a high agreement, it is not perfect (a score of 1) and inter-annotator agreement is certainly worth addressing.

On-going Activities

Since the students created this new dataset for a classification problem and dedicated time to understanding the data and engineering features, we are using it for other learning activities in class. After learning about Naïve Bayes classifiers and their use for text categorization, students are implementing the classifier to apply it to two free text features: the tweet text and the account description. Other planned activities for the rest of the semester include experiments with boosted classifiers (Schapire and Singer 1999) and semi-supervised learning (Abney 2008). The goal is to compare different classifiers and their performance.

Student Feedback

After completing the last assignment (error analysis and evaluation), the students filled out an anonymous online questionnaire giving us feedback on the series of assignments. They began filling out the questionnaire during the last ten minutes of lab (we left early) so as to increase the response rate. We got responses from 9 out of 11 people in class, and it was clear from the length of some of the free responses that some students took extra time to tell us more.

The questionnaire began with quantitative questions on the educational value of different aspects of the assignments, and one question on how much they enjoyed the in-class competition. Figure 4 summarizes the results. We think these results show that:

- They did not like labeling the data, but they grudgingly agreed that it informed their choice of features.
- They liked the feature engineering best, with evaluation and analysis second.

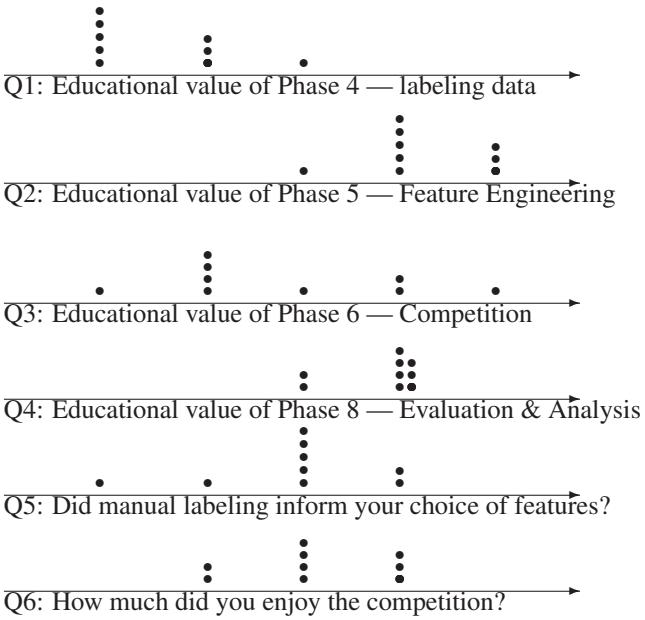


Figure 4: Student valuation of aspects of the assignments. Since there are only nine responses, we show the actual data. Responses range from 1 (least) to 5 (most).

- The educational value of the competition had the least agreement: some thought it very valuable, others very little. Somewhat surprisingly, there was substantial agreement on whether they *liked* it, with nobody hating or loving it.

We also asked several free-response questions about the assignments. Aspects they found valuable include:

- feature engineering is valuable and interesting (eight of nine mentioned this)
- seeing what features others came up with
- competition against majority classifier (rather than each other)

- working in pairs
- modifying code in lab, during the competition

The only aspect of the assignments that more than one person mentioned as being "less valuable" was the manual labeling: 8 of 9 mentioned this. We were not surprised to see that the manual labeling was nearly universally disliked. However, it was reassuring to read comments that showed that the students understood why manual labeling was important, both in terms of seeing the data that the classifier was dealing with and having data for the classifier to train on. They observed that there were diminishing returns to the insights gained, and they felt that the need for speed (to finish their quota) reduced the opportunity for reflection on the manual labeling. We found this to be thoughtful and useful feedback.

Next, we asked them for suggestions for improvement. There were only two noteworthy suggestions, which we will certainly consider in the future: access to more than one tweet from an individual Twitter account, and creating their own classifier to use in the competition.

Finally, we asked the students whether they would consider using supervised learning in future work. Here, we thought the overall feedback was fairly positive, but we will let them say it in their own words. Here are some selected comments:

- "I think [it's] cool and useful — but probably won't use it myself, because it requires supervision/the creation of a corpus; which seems too time-intensive to be useful."
- "If I had a problem for which it was an appropriate solution, yes."
- "I think that it seems like a pretty good way to solve problems. The caveat is that you'd need an army of poor grad students (or undergrads) to label the corpora for you."
- "I find supervised learning fascinating. I would almost definitely use it if an appropriate problem arose."
- "Only if I knew there were useful features."
- "We have considered using it for our final project and I would use it again. It felt like a magical thing that only geniuses could do before, so I am happy that I am able to see that I can do it and it is not so hard."

The last comment is particularly gratifying.

Lessons Learned

Involving undergraduate students in current research projects is not trivial. The inherent uncertainty of every research project is not something that undergraduate students expect in their assignments, which usually deal with well-established textbook problems. With this assignment we took a risk and the result is encouraging, though we would change several things to improve it:

1. reduce the task of manual labeling, retaining some of it for the learning experience, and outsource the remainder to services such as Amazon Mechanical Turk.
2. use Orange from the beginning and encourage students to use its interactive tree builder functionality to explore the value of engineered features.
3. allow students to use external sources of data (such as more tweets or access to the Web).
4. postpone the later phases of the assignment so that the students can experiment with several classifiers.

One of the positive outcomes of this assignment was the fact that two teams chose as a final project a topic that involves data from Twitter. One team is creating a game that will be populated with information extracted from the Twitter accounts of the players and another team is building a committee of classifiers to predict the political orientation of Twitter accounts based on different types of information.

Acknowledgments

We are grateful to the students of CS232 from Fall 2010, and particularly for their feedback on this assignment. We also appreciate the comments from the anonymous reviewers.

References

Abney, S. 2008. *Semi-supervised Learning for Computational Linguistics*. Chapman & Hall/CRc.

Bird, S.; Klein, E.; and Loper, E. 2009. *Natural Language Processing with Python*. O'Reilly Media.

Bollen, J.; Mao, H.; and Zeng, X. J. 2010. Twitter mood predicts the stock market. arXiv:1010.3003v1 [cs.CE].

Golbeck, J., and Hansen, D. L. 2010. Computing political preference among twitter followers. In *Tech Report HCIL-2010-20*. http://www.cs.umd.edu/localphp/hcil/tech-reports-search.php?number=2010% -20.

Koppel, M.; Argamon, S.; and Shimoni, A. R. 2002. Automatically categorizing written text by author gender. In *Literary Linguistic Computing*, volume 17, 401–412. Oxford University Press.

O'Connor, B.; Balasubramanyan, R.; Routledge, B. R.; and Smith, N. A. 2010. From tweets to polls: Linking text sentiment to public opinion time series. In *Proc. of 4th ICWSM*, 122–129. AAAI Press.

Paczkowski, J. 2010. Nearly two billion tweets a month; giddy foursquare mayors suspected. http://digitaldaily.allthingsd.com/20100609/2-billion-tweets-a-month/.

Schapire, R., and Singer, Y. 1999. Improved boosting algorithms using confidence rated predictions. In *Machine Learning*, volume 37, 297:336.

Tumasjan, A.; Sprenger, T.; Sandner, P. G.; and Welpe, I. M. 2010. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proc. of 4th ICWSM*, 178–185. AAAI Press.

Yardi, S.; Romero, D.; Schoenebeck, G.; and Boyd, D. 2010. Detecting spam in a twitter network. In *First Monday*. http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/% 2793/2431.