Categorizing Spam Using K-nearest Neighbors

CPE 466: Knowledge Discovery from Data
Dr. Stanchev

Team Justice For Harambe
Chris Opperwall, Stephen Daily, Kim Arre, Dennis Wong

Introduction

For our final project, we used the k-nearest neighbors learning algorithm to classify a data set of forum posts as either legitimate posts or spam posts, and analyzed how well our algorithm worked by comparing our results to the human verified results.

Input Data

The data set we chose to use is based off of the online repair website, iFixit.com. The site provides repair guides and answer forums to anyone intending to fix personal devices. iFixit's answer forums keep a record of reviewed spam posts which are available to moderators of the site. Spam decisions are archived in a database for training an automated spam detection service.

Using entries from this database, we constructed two datasets: human verified spam posts and human verified not-spam, or ham, posts.

For each post, we pre-processed the data to only include:

- Whether or not a post had a title
- The number of times "free" or "download" was present in the body of the post
- The number of times "watch" or "movie" appeared in the body of the post
- The ratio of alphabetical characters to the total character count of the post

We obtained a total of 2500 entries for human verified spam posts, and 2100 entries for ham posts for a total of 4600 entries of data.

The Algorithm

K-nearest neighbors is a classification algorithm that determines which category input data should fall under by analyzing the k items from the dataset that are most closely related. It then determines the average category that the neighbors classify as and assigns the input data to that category.

We then randomly separated ⅔ of our data into the training set and reserved the remaining ⅓ of it to be our testing set. The purpose of the training set was to have enough neighbors to compare

the testing data to. The separation of these sets allowed us to assess how accurate our algorithm was, since we knew definitively what they should have classified as.

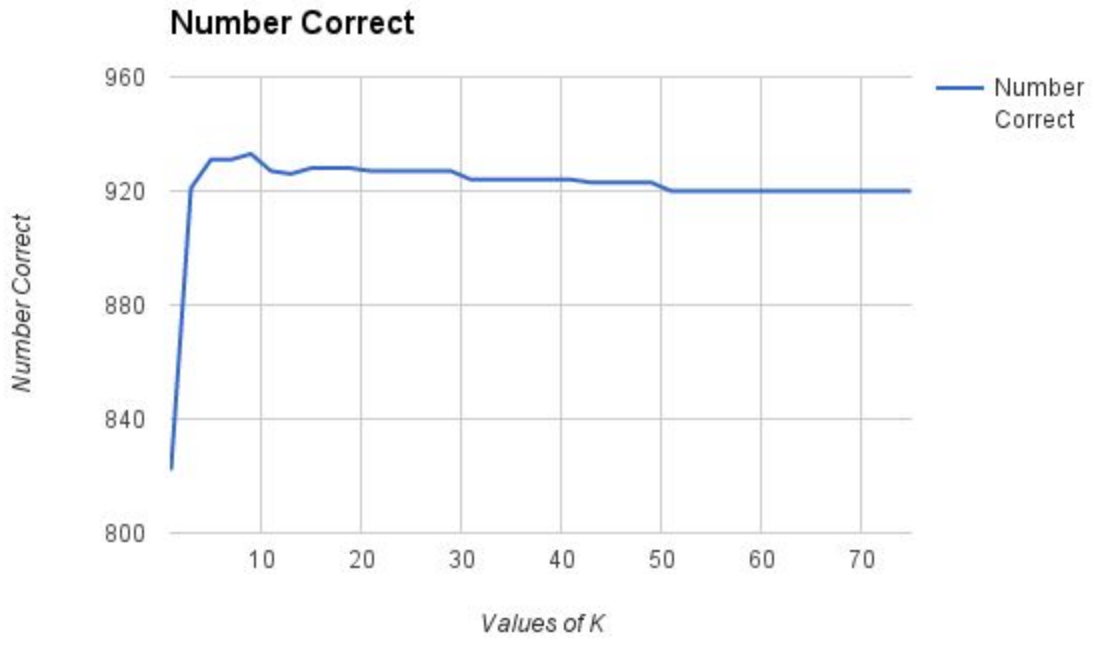
The algorithm itself was relatively straightforward. After preprocessing the data, each line was input and parsed into vectors stored within a class. That class also held the human classifier for that vector. The vectors were then randomly shuffled and split into the above proportions. Then each of the vectors in the testing set were used as input.

The program calculated the distance between each vector in the training data and the input vector using the euclidian distance formula. Afterwards, the vectors were sorted based on their distance (smallest first) and the first k vectors were selected. The majority classifier was returned and that was the result. Since there were only two classifiers, the program was relatively simple.

The output for the program was the algorithm's classification of the post, either spam or ham, and the value of k that was used.

Results

We ran this program for multiple values of k and found that it maxed at a value of k = 9. Our highest precision was 933 / 1514 correct predictions which is only about 60% accurate. The results plateau and steadily decrease as the number of k increases.



We believe that the accuracy of our data was hindered by the lack of diversity in the training data. In our pre-processing, we didn't have a lot of different values. If you look at the preprocessed data, you'll see that the majority of the data has the same first three numbers. The fourth number was suppose to be a representation of the number of english letters, but ended up being very close in difference.

In order to get more accurate results the data pre-processing would need to be altered to allow for more discernible attributes to have a wider range of values.

Usefulness

The reason this project could potentially be useful is that if iFixit were to move in the direction of implementing a system like this for their forum, it could help cut down the time that human moderators spend on assessing if posts are spam or not. If a set of criteria was found that turned out to be accurate enough, allowing an algorithm like this to automatically cut out the obviously spam posts would help the moderators spend more time manually classifying posts that are harder to determine.

Team Member Contributions

Stephen: Designed and implemented the algorithm. Results of the algorithm. Aided in document writing, specifically the results and algorithm details

Chris: Wrote the preprocessor that translates post texts into vectors based on four different criteria. Wrote testing code to determine TP, FP, FN, TN values of the test runs. Helped pick the attributes to translate data into.

Kim: Aided in analysis of whether or not the algorithm was a success, and wrote the final report document on the project procedure, approach, and analysis. Created most of the presentation.

Dennis: Aided in the creation of the presentation. Helped in brainstorming and picking attributes to use. Assisted in proofreading and layout of document and presentation.