Q0:

1. print(“Hello, World!”)
2. Sean Mebust, smebust20@amherst.edu, Sean Mebust, Noah Aube, naube20@amherst.edu, noah aube
3. Score: 28.14096

Q1:

The first thing our system does is gather both user posting data and social connections. In order to make the social connections data useful for our learner, we created a method that, for each user, will gather known data from each of that user’s connections and average them together. We thought this information, especially the average location of a user’s friends, would be helpful in predicting a user’s location as a user is likely to live geographically near, and have similar posting habits to, those with whom they are connected on social media. We combined this information with the known data on the user’s posting habits, giving us six additional features, for a total of ten features, from which to predict a user’s location.

Another datapoint we considered calculating was the timezone of various users. We thought that by using the UTC time at which users at various longitudes were most active, we could determine the local time when most people post around the world. We figured that we could use this time to predict the timezone (and therefore a narrow slice of longitudes) for users who’s locations were unknown. We ultimately decided against this, however, since simply training the model on the users most active hours had a very similar effect.

This data was not perfect, however, because many of the users that we were training our model on had no known location and were therefore skewing our model towards null island. To nullify this we tried a variety of methods, the most successful of which was also the simplest. When training our model we simply ignored all users for whom we lacked a location.

To choose the model, we decided to randomly split 75 percent of the points in posts_ train to train on, and use the other 25 percent to validate the model. We then tried a variety of different learners to see which would return the lowest mean squared error. Of these models (which included: k-nearest neighbors with different values of k, decision trees with different max-depths, support vector regressors with different kernels and other parameters, and linear regression) linear regression returned the lowest loss by a significant margin. We decided that since linear regression worked so much better than the other learners we did not need to use cross validation to confirm our pick.

In the end, we trained one model using all of ten features we gathered from the data, using linear regression to fit the model, in order to predict longitude and latitude simultaneously. We had also tried creating two models, one to predict longitude and the other to predict latitude, but we found this to be much less reliable than using one model for both, as they would often overfit one coordinate and underfit the other (for example, SVMs would often return 39.9 as the latitude for every point). Our resulting model had an average root mean squared error of about 27.7 on the randomly selected validation sets, a public score of about 28.1 on the test set, and a private score of about 35.1.

Q2:

One thing that we would do if given significantly more time to work on this problem would be to test how different subsets of features affect the model. Of the ten features we used to train the model, it is very likely that some are better indicators of location than others and therefore should be weighted higher when training. We did not consider this while creating our learner mostly due to lack of time, so it is likely that some of the features we used actually decreased our models accuracy. Having more time to test the effects of each feature would allow use to optimize our training data and, by extension, our model.

Another thing that we could do would be to try an ensemble method. This would allow us to train different learners on different features, allowing us to extract the best possible prediction from each feature. The additional layers of computation would create a more finely tuned model that would potentially be far more accurate than the simple learner we used.

Q3:

We believe login times would be very helpful in determining location for a very similar reason that most active posting times is helpful. Similar to what we mentioned about timezones in Q1, if we determine that people tend to login in around, say 8pm local time, the UTC time that a person tends to login at can be used to pretty accurately guess their longitude. Also, thinking astronomically, the sun sets at different times at different latitudes, so for example people may login earlier if they live further north. Having more features related to time is very helpful for predicting location, both mathematically and logically, because we define often time by location on earth.

If we could use a form of natural language processing on post content or links clicked, we could potentially glean some crucial information for predicting location. Analysing text to search for proper nouns can give major clues towards location, especially (and obviously) if place names are used. We could also train a learner to identify regional dialects to help narrow down location. These are just a couple of the many ways text could help inform a leaner.

Additionally, if given access to post content we could potentially gain helpful data from images posted. A learner could potentially be trained to identify buildings in images, locating where the photo was taken. Likewise, one could be trained to identify land features. One could even use pictures of the night sky to accurately and directly determine latitude.

Given more time and more data, we could see how a neural network could be made to analyze text and images in order to give a very accurate prediction of a person's location on Earth.