Health Net Data Project

Joe Rummel

As Americans become older and more obese, health care is going to become more and more costly. Can we use data to help mitigate some of the costs? Hospital stays are the most costly expense for a health insurer. More than 71 million individuals in the United States are admitted to hospitals each year, according to the latest survey from the American Hospital Association. Studies have concluded that in 2006 well over $30 billion was spent on unnecessary hospital admissions. If we could predict who was going to be spending time in the hospital, we could render them assistance beforehand, helping avoid the hospital stay and saving the insurer money. By using claims data from a health insurer, I believe I can make those predictions.

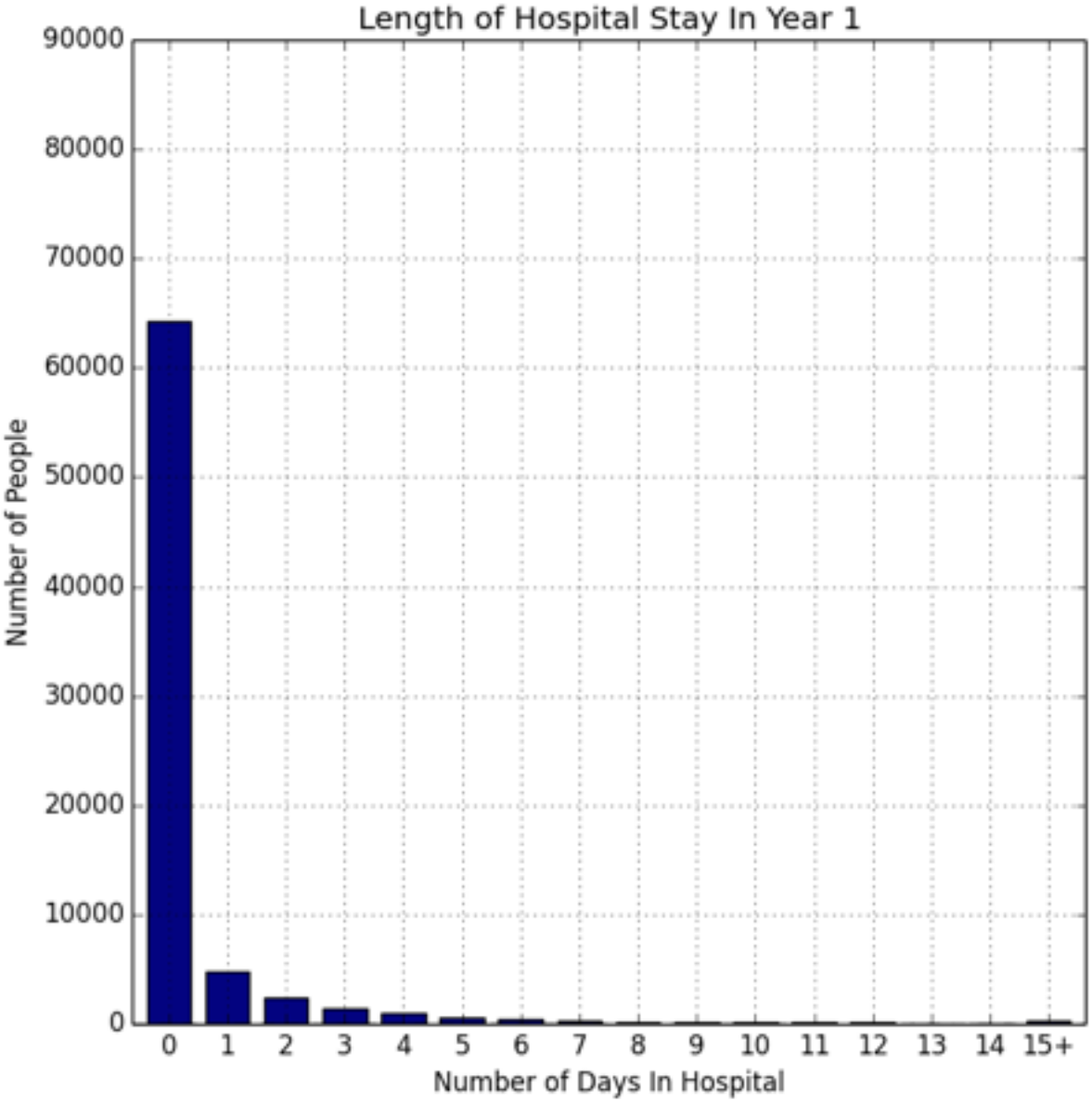
The data for this project came from and old Kaggle competition called "Heritage Health Prize" (http://www.heritagehealthprize.com/c/hhp). The Objective is to use the data from one year to predict the hospital stays for the next year. (i.e. Use Year 1 data to predict Year 2 hospital stays). The data comes in several .csv files:

- Claims.csv contains information on all the claims made at a health insurer for three years
- DrugCount.csv contains the count of unique prescription drugs filled over 3 years
- LabCount.csv contains the count of unique laboratory and pathology tests over 3 years
- Lookup PrimaryConditionGroup.csv provides a description of the different groups of medical conditions a patient could have
- Lookup ProcedureGroup.csv contains a description of the different procedures a patient may need to go through
- Members.csv contains the member ID, sex, and age of each patient
- DaysInHospital_Y2.csv and DaysInHospital_Y3.csv contains the member ID and number of days spent in the hospital during year 2 and year 3.
- Target.csv contains the member ID of the members that I am going to predict Year 4 hospital stays for.

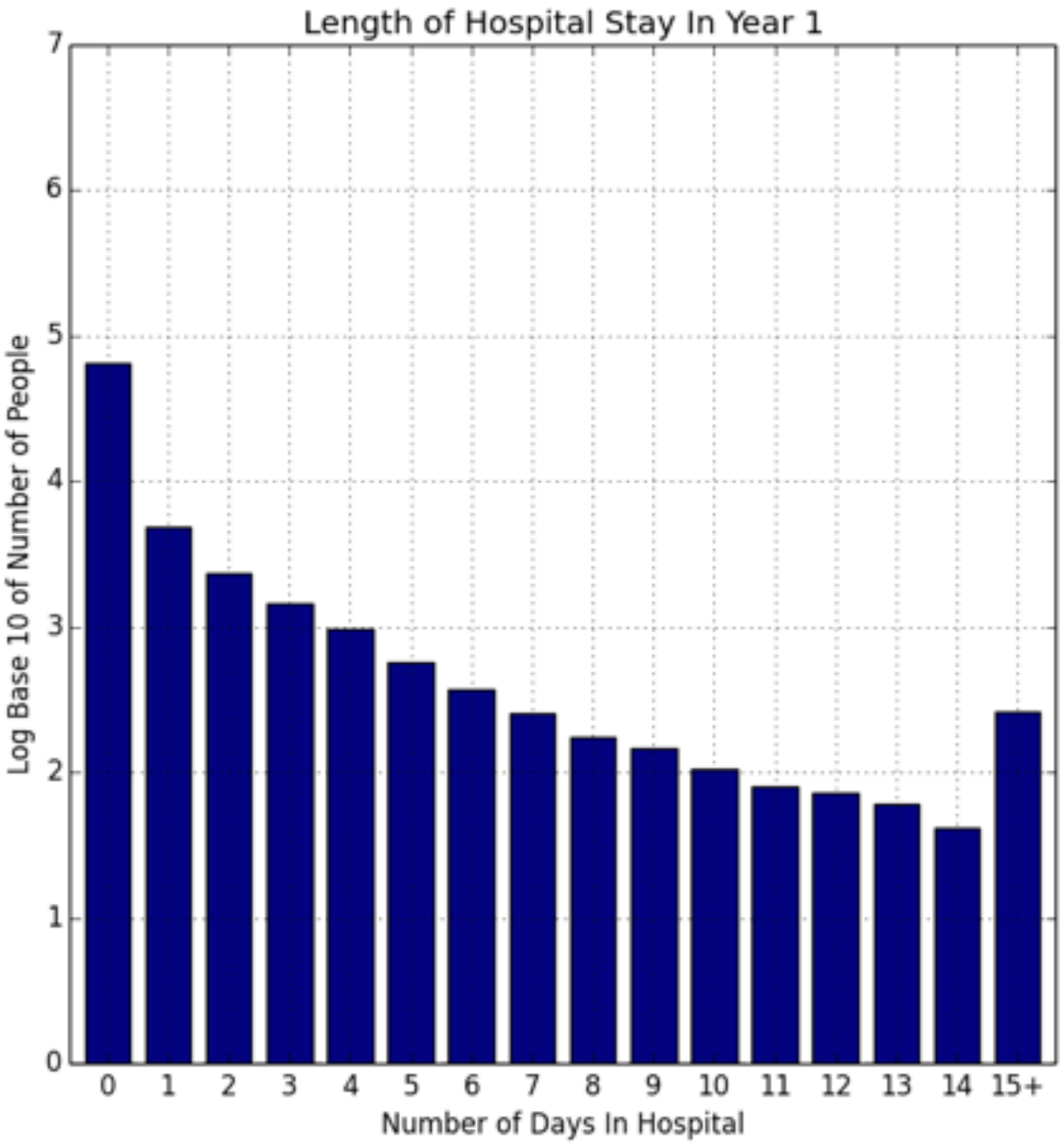
I decided to use Year 1 data as my training data with DaysInHospital_Y2.csv being the target for the training data and Year 2 data as the test data with DaysInHospital_Y3.csv as the target for the test data. My model created DaysInHospital_Y4.csv, which contains the predicted days in the hospital that my model outputted based on the Year 3 data. I created a pSQL database to store all of the data, with each .csv file being a separate table in the database. I decided on SQL because of its ability to do JOINS between tables, which I needed to do a lot of. All code was written in Python.

Exploratory Data Analysis

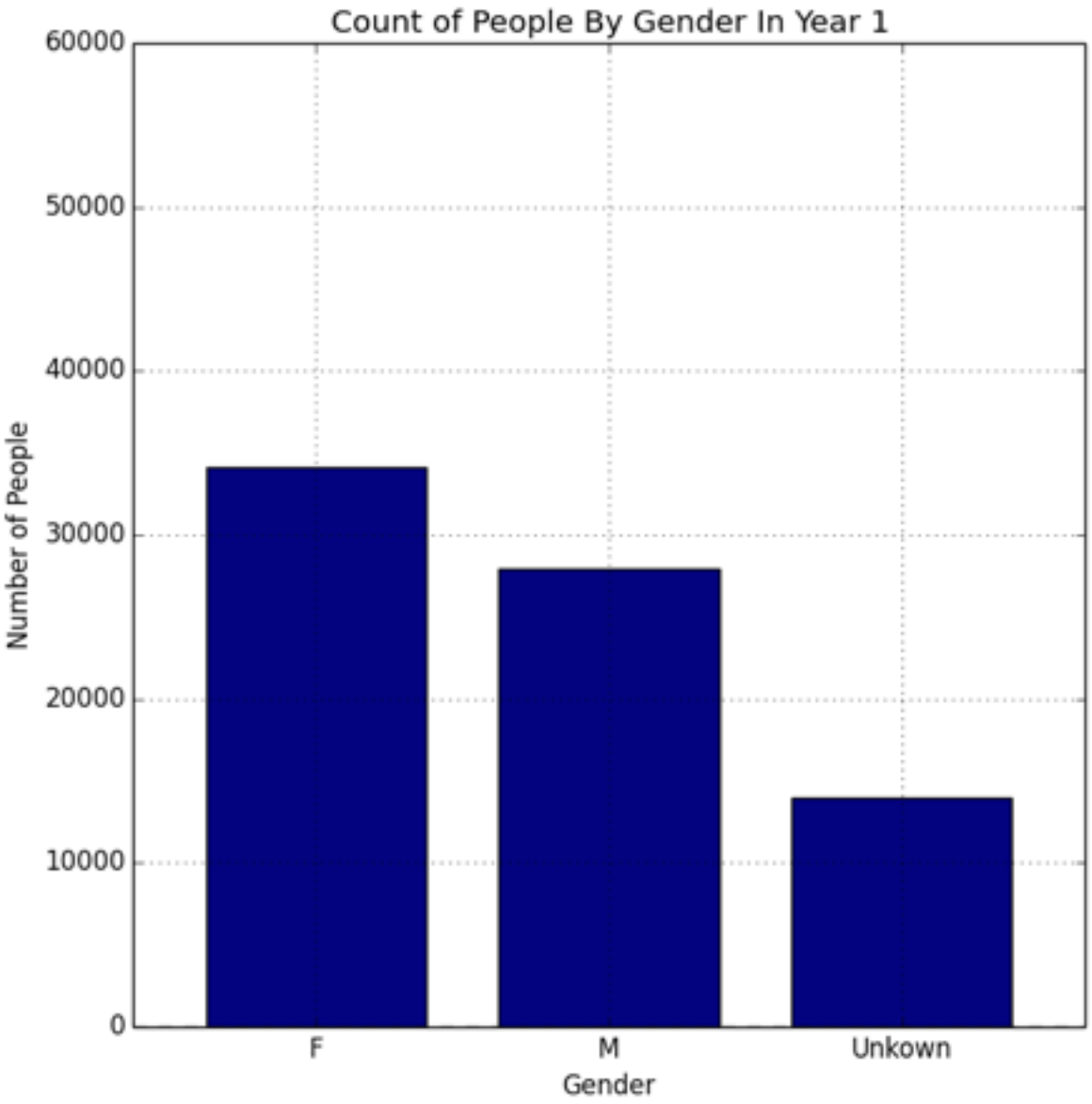
Once all of the data was in the database, I performed Exploratory Data Analysis to learn what was in the data. The first thing I wanted to see was how many people spent time in the hospital, which is in the **Length of Hospital Stay in Year 1** chart.



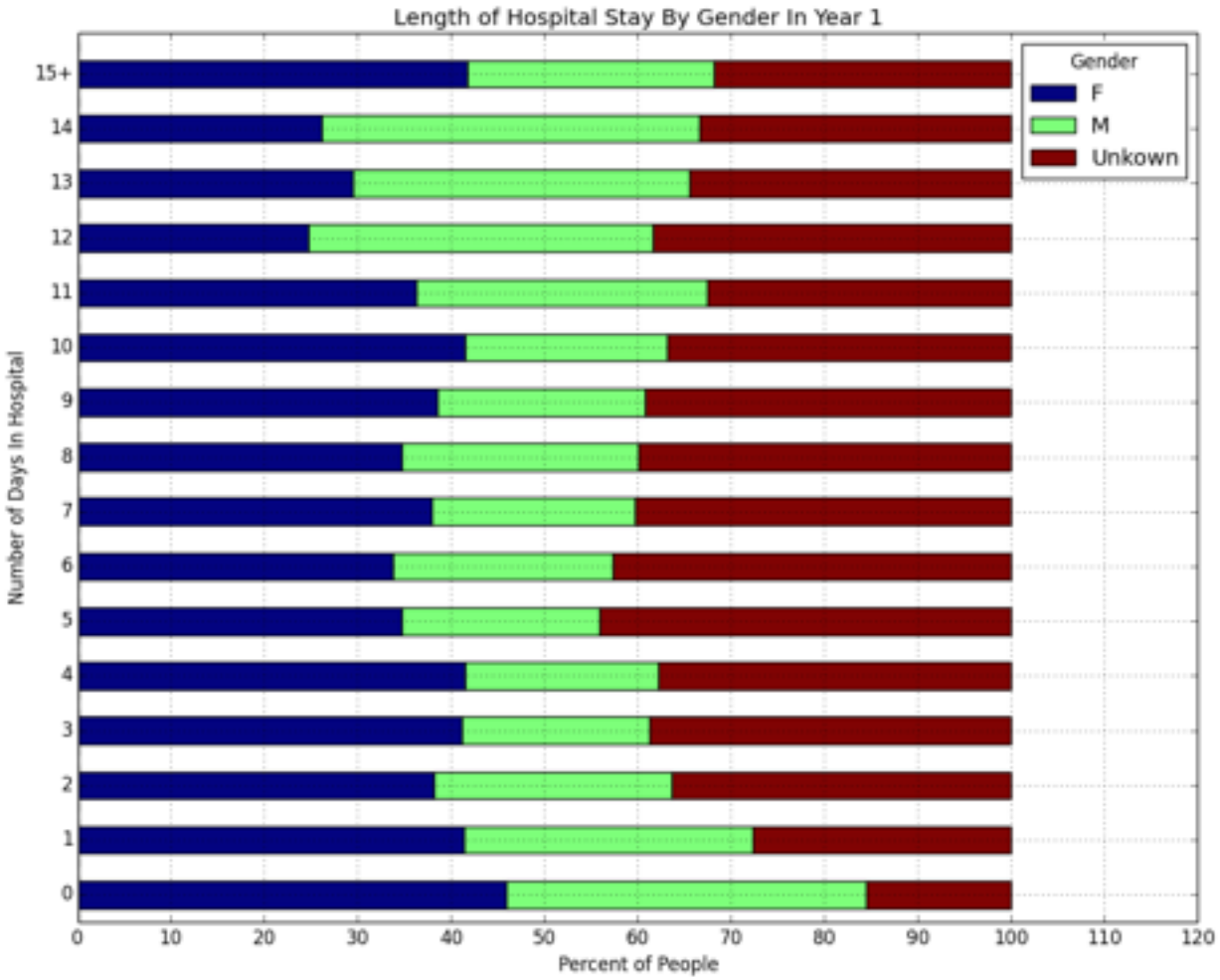
The data is very heavily skewed toward zero days in the hospital. Right away, I knew that a linear model, such as logistic regression, would not work on this data because the data is not anywhere close to being normalized. I then created a chart that used the log10 of the number of people in the y-axis so that I could see how the data trends. As I expected, the number of people falls as days in the hospital increase, except for a small uptick at 15+.



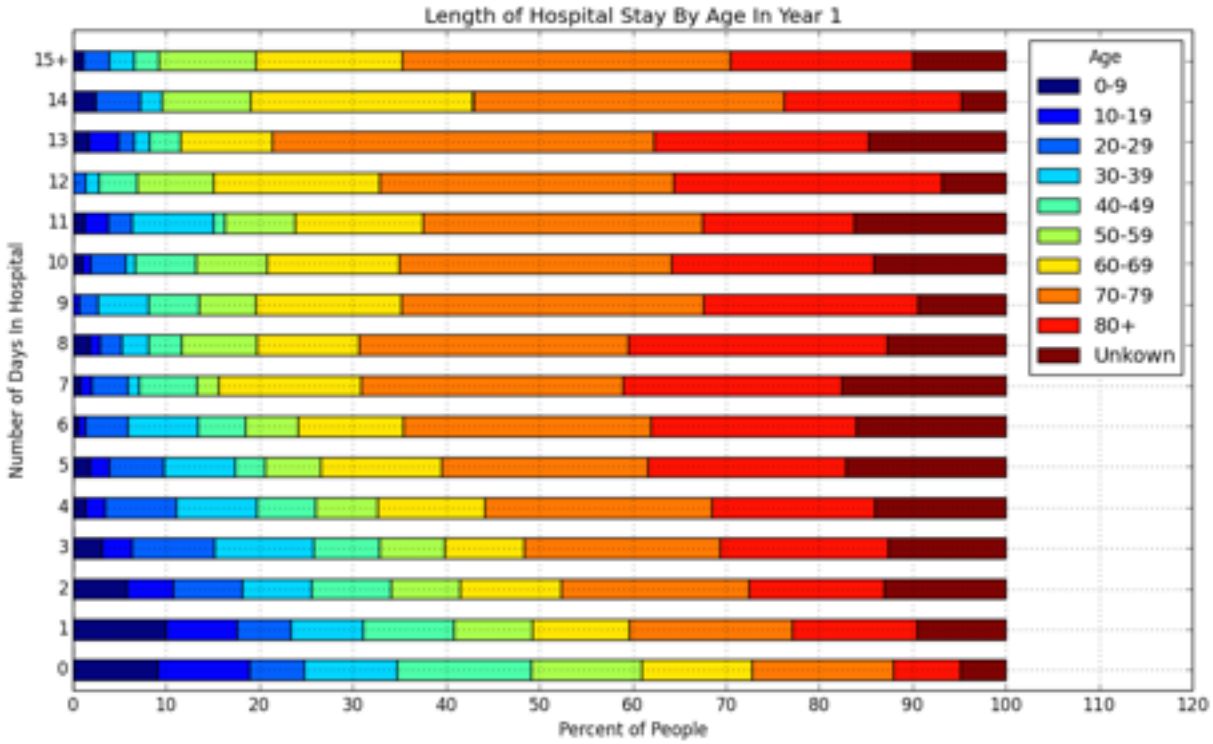
Next, I wanted to see what other trends I could find in the data. I decided to look at the gender of the patients and see if the data was skewed toward male or female, so I created the **Count of People By Gender in Year 1** chart.



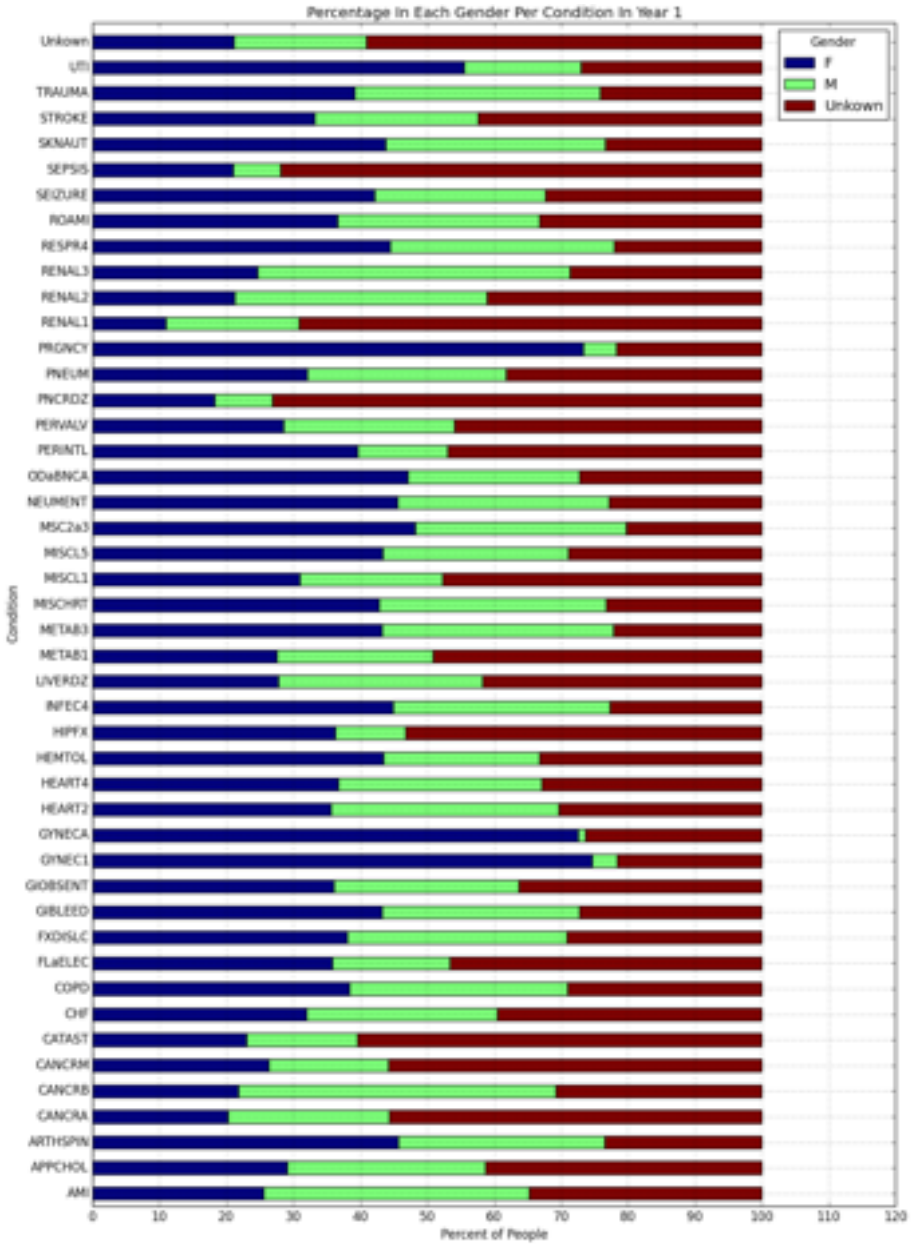
There were more females than males, but not a lot more. The alarming thing I take away from the **Count of People By Gender in Year 1** chart is that there were many patients in which the gender was not specified in the data. I could have just thrown out the data from those patients since knowing whether they are male or female is important and only about 17% of the patients had unknown gender. However, the next chart, **Length of Hospital Stay By Gender in Year 1** shows that the gender was unknown on about 15% of the patients who spent zero days in the hospital, but those patients accounted for a much larger percentage of the people who did spend time in the hospital (about 30-40%). Therefore, throwing them out would skew the data even more toward zero days in the hospital, so I had to keep that data and have three categories for gender: male, female, and unknown.



The next parameter I looked at was age. I expected that older people would spend more time in the hospital than younger people and the chart **Length of Hospital Stay By Age in Year 1** does show that trend. People under 60 years old accounted for about 60% of the people who spent zero days in the hospital, but people over 60 years old accounted for about 70% of the people who spend 15 or more days in the hospital. Like with gender, there were people with unknown age. About 5% of the people who spent 0 days in the hospital had an unknown age, but about 15% of people who spent time in the hospital had an unknown age. Therefore, we need to leave the patients with unknown age in the data lest we skew the data even more toward zero days in the hospital.



Being that the data had unknowns for gender and age, I decided to investigate the conditions and see if I could find anything else that was obviously wrong with the data (i.e. pregnant men). Indeed, on the chart **Percentage in Each Gender Per Condition in Year 1**, not only were there men being treated for pregnancy (PRGNCY), but there were men with female breast cancer (GYNECA) and men with gynecological issues (GYNEC1). Though I did not make a chart for Age vs. Gender vs. Condition, I did find that there were women over 60 years old who were being treated for pregnancy and perinatal period (PERINTL), which is defined by the World Health Organization as 22 weeks of gestation to 7 days after birth of a baby. There were also men being treated for perinatal period, but since perinatal period covers disorders of the newborn of either sex as well as maternal disorders, then it would make sense that a male in the 0-9 age range could fall under the perinatal period condition group. However, the data had older males falling under the perinatal period condition, which is clearly bad data. The condition data would have to be cleaned up as part of the data munging process. There was a Data_Dictionary_release3.pdf from the Kaggle competition that describes what each of the condition groups are. There could be other conditions mislabeled in this data, but I would need a domain expert to help me find them.



Data Munging

Once data exploration was done, it was time to determine what parameters my model would use and how to munge the data. I decided to use gender, age, conditions, number of drugs prescribed, number of lab tests done, and number of claims submitted by a patient as the predictive parameters I would use for my model. I chose those parameters mainly by intuition from my limited domain knowledge. The winning teams for this Kaggle competition had doctors on their teams to help with the domain knowledge, but I didn't have a doctor to work with, so I went with my intuition. I then had to munge the data from the parameters in order to get them in a format where they could work with a model.

The gender, age, and claims parameters were the easiest when it came to data munging. Gender already was broken out into “M” and “F” categories, so all I had to do is replace the blanks with “Unknown”. Age already had “0-9”, “10-19”, “20-29”, “30-39”, “40-49”, “50-59”, “60-69”, “70-79”, and “80+”. Once again, all I had to do was replace the blanks with “Unknown”. For claims, I just needed to count how many times a member id appeared in the claims table in the SQL database for a particular year. Because a member could have had more than one condition he/she was being treated for and they may have had more than one occasion during a year where they received prescription drugs or had lab tests done, it would take a little more work to munge those parameters.

The number of drugs prescribed was broken out into “1”, “2”, “3”, “4”, “5”, “6”, and “7+” categories. However, a patient may have gotten 4 drugs on one visit and 6 drugs the next, so how did I deal with that? I took the average of drugs prescribed per visit, rounded to the nearest integer, and used that number for the number of drugs prescribed parameter. What to do with “7+” was the only problem. If I made “7+” equal to 7 when calculating the mean, then the only way the mean would end up being “7+” is if the patient received “7+” drugs at every visit. Therefore, I decided to make “7+” equal to 9 so that a patient who had four visits to the doctor with a prescription count of “7+”, “7+”, “7+”, and “1” for those visits, the mean would turn out to be “7+” instead of “6”. I used the same method for the number of lab tests parameter, except that lab tests could go up to “10+” and I made “10+” equal to 12. Also, if a member ID was not listed at all in the lab count table or the drug count table, then I used “0” as the category for that parameter.

Figuring out what to do with the conditions parameter was the biggest challenge. A patient could be treated for multiple conditions in the same year, so how do I represent that? There are about 44 condition groups, so I decided to represent the conditions as a binary number with 44 bits, where each bit represents a condition. A condition gets a “1” if the patient has that condition and a “0” if he/she does not. The next question is what order should I list those conditions, as bits on the left will cause the binary number to have a much higher value. I needed to weight the conditions. I checked the Kaggle competition and found the Milestone 1 winners, Market Makers, had a doctor on their team and that doctor assigned Admission Risk values to each condition, with 5 being the highest risk and 1 being the lowest risk. I set the bits for the binary number based on the admission risk. Conditions with 5 for admission risk would be to the left of the binary string and conditions with a 1 for admission risk would be to the right. Therefore, if a patient had a condition that was high risk for hospital admission, then the binary number would be high. Of course, before doing all of this I had to first eliminate the bad data I discussed earlier (pregnant men, pregnant women over 60, etc.). After all the munging was complete, I put the clean data into its own .csv file for the model to use. The training data, the test data, and the data the predictions were to be made on each had their own .csv file.

The Model

The final task was determining what model I would use to predict how many days a patient will spend in the hospital in Year 4. I decided to go with a categorical model, since number of days in the hospital is categorical (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15+). Since I do have target data for the training and test data, then I would need a supervised algorithm. I have already mentioned that a linear model would not work because of how skewed the data is. I tried a logistic regression just to see what would happen and the results were not good, as expected.

10- Fold Cross Validation for Logistic Regression:
[0.84521305 0.84521305 0.84521305 0.84521305 0.84521305 0.84521305
0.84521305 0.84521305 0.84532421 0.84519269]

Logistic Regression Accuracy:
0.849807517323

The cross validation and accuracy both looked pretty good at around 85%, but let's dig deeper:

Metrics Report for Logistic Regression:

	precision	recall	f1-score	support
0	0.85	1.00	0.92	60706
1	0.00	0.00	0.00	4464
2	0.00	0.00	0.00	2182
3	0.00	0.00	0.00	1429
4	0.00	0.00	0.00	842
5	0.00	0.00	0.00	528
6	0.00	0.00	0.00	287
7	0.00	0.00	0.00	218
8	0.00	0.00	0.00	143
9	0.00	0.00	0.00	115
10	0.00	0.00	0.00	103
11	0.00	0.00	0.00	65
12	0.00	0.00	0.00	62
13	0.00	0.00	0.00	50
14	0.00	0.00	0.00	23
15+	0.00	0.00	0.00	218
avg / total	0.72	0.85	0.78	71435

Basically, the model was 85% accurate because it predicted "0" for everything and 85% of the data was zero days in the hospital. I decided a decision tree would be the way to go, mainly because it handles non-linear data better than most other classifiers. To make my model even more accurate, I used a random forest. I tried a random forest of 5, 10, 15, and 20 decision trees and did a 10-fold cross validation. At 5, the cross validation was much worse than at 10, but 15 and 20 were not any better than 10,

though they took much longer to run, so I decided to use a random forest of 10 decision trees. Also, random forests have an option where you can give parameters a weight, so if the data is skewed, as this data is, you can give more weight to the parameters that have less data. Since “0” was by far the most, I weighed that as a 1 and every other parameter was weighed based on how many times smaller its dataset was then “0”. For example, there were 60,706 people who spent 0 days in the hospital in the test data (year 2). There were 4464 who spent 1 day in the hospital, which is about 12 times less, so 1 day would get a weight of 12. Here are the results from the 10-fold cross validation and the accuracy of the random forest:

Random Forest Cross Validation:
[0.82496055 0.8229879 0.81917412 0.81772751 0.82114676 0.81693845
0.81772751 0.81917412 0.81967644 0.81573063]

Random Forest Accuracy:
0.796430321271

Metrics Report For Random Forest:

	precision	recall	f1-score	support
0	0.85	0.93	0.89	60706
1	0.06	0.03	0.05	4464
2	0.05	0.02	0.03	2182
3	0.02	0.01	0.01	1429
4	0.03	0.01	0.02	842
5	0.05	0.01	0.02	528
6	0.01	0.00	0.00	287
7	0.01	0.00	0.01	218
8	0.01	0.01	0.01	143
9	0.00	0.00	0.00	115
10	0.00	0.00	0.00	103
11	0.00	0.00	0.00	65
12	0.00	0.00	0.00	62
13	0.00	0.00	0.00	50
14	0.00	0.00	0.00	23
15+	0.02	0.00	0.01	218
avg / total	0.73	0.80	0.76	71435

The model does a very good job predicting if a patient will spend 0 days in the hospital (.93 recall, which is excellent), but it is very bad at predicting if a patient will spend any more than 0 days in the hospital. However, at least it did get some good predictions rather than predicting everything is “0”. Finally, I decided to try Singular Valued Decomposition dimension reduction to see if that would help the accuracy. I did not have a lot of dimensions to begin with, so I was skeptical dimension reduction would help and I was right. Dimension reduction made it worse:

Random Forest With Dimension Reduction Cross Validation:
[0.80865334 0.80733824 0.80746975 0.80628617 0.80746975 0.81128353
  0.80996844 0.80576013 0.80704985 0.80704985]

Implementation Plan

This entire project is written in python, so it is just a matter of running some python scripts to implement this model. First, the data would need to be put into the SQL database by running populateClaimsTable.py, populateDrugLabTable.py, populateGroupTables.py, and populateMemberTable.py. The weights for the conditions need to be added to the database, so cond_weights_insert.py would need run. The target data would need to be put into the database by running populateDaysTable.py. After the database has the data, dataMunging.py to munge the data and create a trainingData.csv, testData.csv, and predictData.csv file. Last, run model.py, which will export the predictions into a file called DaysInHospital_Y4.csv.

Conclusion

With such skewed data, random forest was likely the best classifier, but it still did not do very well. Further investigation could be done to find the reasons for this. First, the fact that I did not have much domain knowledge likely hurt my ability to choose the right predictors. There were several parameters in the data I left out of the model because I did not think they were pertinent, but with the right domain expertise, I may have found that they were pertinent. Also, handling severely skewed data is another avenue to look at. Are there better classifiers out there for handling skewed data like this? More than likely, some kind of ensemble using several different classifiers would yield better results. Better data would have helped as well. I was able to find several errors and omissions in the data, but who knows how many I didn't find. As the population ages and waistlines grow, projects like this one could be of great benefit to insurance companies and patients alike and I look forward to seeing more research done using health care data in the future.