# Capstone Project

# Machine Learning Engineer Nanodegree

Richard Dunlap

August 17th, 2016

# I. Definition

## Project Overview

In this project, we explore the ability to predict the likelihood of a patient dying in an emergency room visit using data from the patient's healthcare records associated with the visit; this is one of several questions suggested for a Healthcare specialization capstone project for Udacity's Machine Learning nanodegree course. Since the situation being modeled is an emergency room setting, where there is not time to acquire detailed historical medical records for the patient, the analysis is based strictly on data found in the record of the ER visit. The data analyzed is drawn from MIMIC-III, a freely accessible critical care database based on actual hospital records that have been de-identified, or anonymized, for research purposes.[1]

## Problem Statement

The MIMIC-III dataset provides data for patients admitted to critical care units at Beth Israel Deaconess Medical Center in Boston, Massachusetts over more than a decade. While some patients were admitted multiple times, each item of data is associated not just with a specific patient, but with a specific visit. A variety of types of data are available, including:

- Admissions data, including the date of admission and how the patient came to be admitted to the hospital.
- Demographic data collected during the visit, such as the patient's age and gender.
- Billing data, including codes for diagnoses recorded and procedures performed

[1] MIMIC-III, a freely accessible critical care database. Johnson AEW, Pollard TJ, Shen L, Lehman L, Feng M, Ghassemi M, Moody B, Szolovits P, Celi LA, and Mark RG. Scientific Data (2016). DOI: 10.1038/sdata.2016.35. Available from: http://www.nature.com/articles/sdata201635

- Results of various laboratory tests, including microbiology cultures
- Chart data, including medications administered
- Bedside monitoring data, including vital signs and alarms

For each patient, a flag is recorded indicating whether or not the patient died during the visit. In this project, we are specifically interested in analyzing the data for emergency room admissions to predict the value of this flag based on the patient records. While some researchers have approached the problem of predicting mortality rates by looking at specific diagnoses[2], we will be using a more data-centric supervised machine learning approach that will allow the data to guide us to the salient features.

Our approach will include the following steps:

- Data exploration and cleaning: In this step, we will analyze the dataset to look for features that may be of particular interest. In addition, we will look for irregularities in the data that will need to be dealt with before applying machine learning techniques.
- Initial machine learning experimentation: In this step, we will explore the behavior of multiple machine learning models acting on the dataset, in order to get an initial feel for which ones may work best.
- Final tuning of the machine learning model: In this last step, we will select one or two specific machine learning models and tune the parameters of the model(s) to achieve the best performance on our training set.

## Metrics

To measure the performance of our models, we consider two metrics:

- Accuracy: Accuracy simply measures how often the model correctly predicts whether or not a patient died based on the data. To calculate this, we divide the number of successful predictions made by the model by the total number of predictions made.

$$accuracy = \frac{correct\ predictions}{total\ trials}$$

[2] See, for example, Cause of Emergency Department Mortality; a Case-control Study. Hossein Alimohammadi, Farahnaz Bidarizerehpoosh, Farzaneh Mirmohammadi, Ali Shahrami, Kamran Heidari, Anita Sabzghabaie, Shahram Keikha. Emerg (Tehran) 2014 Winter; 2(1): 30–35.

- F1 score: The F1 score is a combination of two other metrics: precision and recall.
    - Precision measures how often the model generates a false positive by incorrectly predicting that someone will die, and will be calculated by dividing the number of correct predictions of mortality by the total number of predictions of mortality.

$$precision = \frac{correct\ positives}{correct\ positives + false\ positives}$$

    - Recall measures how often the model generates a false negative by incorrectly predicting that someone will live, and will be calculated by dividing the number of correct predictions of mortality by the total actual instances of mortality.

$$recall = \frac{correct\ positives}{correct\ positives + false\ negatives}$$

    The F1 score will then be calculated by taking the harmonic mean of these measures:

$$F1\ score = \frac{2 * precision * recall}{precision + recall}$$

    Note that while accuracy treats correct positives and correct negatives the same, the F1 score ignores correct negatives.

In general, the F1 score gives us a better measure for the performance of a supervised learning algorithm than accuracy; in particular, if two models have the same accuracy, the one that performs best on achieves a better balance between the two types of errors (false positives and false negatives) will have a better score. In addition, the focus of the F1 score on positive values is useful when the positive value is the rarer value of the two, as is the case here. Thus, the F1 score is the metric that we will use to measure performance.

As a brief aside, it is worth discussing whether a balance is what we are really looking to achieve here. Let's think about the ramifications of false positives and false negatives. If we assume that the results of this classification might be being used as part of a triage process to determine which patients should be prioritized for limited bed space, then a false negative means that a patient at risk for becoming critically ill might not receive needed care in a timely fashion. However, false positives mean that we may be using limited resources on patients at

lower risk, which denies us the ability to use them on patients truly at higher risk. Thus, we need to find a balance between the two extremes, and the F1 score helps us achieve that.[3]

# II. Analysis

## Data Exploration

For this project, we will restrict ourselves to two types of available data from the MIMIC III set – demographic data and diagnosis data – as a means of keeping the problem tractable for initial analysis. The following tables from the MIMIC III data set will be used:

- PATIENTS: This table contains one row for each patient in the data set. This table captures data that is consistent across all visits made by this patient. The following fields will be of use:
    - SUBJECT_ID: A unique identifier for the patient.
    - GENDER: Whether the patient is male or female
    - DOB: The date of birth of the patient
- ADMISSIONS: This table contains one row for each patient visit. The following fields will be of use:
    - HADM_ID: A unique identifier for the visit
    - SUBJECT_ID: The patient associated with this visit
    - ADMITTIME: The date and time of admission
    - ADMISSION_LOCATION: Where the patient entered the hospital. For this study, we will specifically look at rows with a value of EMERGENCY ROOM ADMIT
    - RELIGION, MARITAL_STATUS, ETHNICITY: Self explanatory – as part of the study, we'll explore whether these make a difference.
    - DIAGNOSIS: While there is a diagnosis field in this table, it is a free form text field not suitable for analysis
- DIAGNOSES_ICD: This table contains one row for each diagnosis code associated with a patient during a visit
    - HADM_ID: The visit during which the code was recorded
    - SUBJECT_ID: The patient associated with the visit and diagnosis

[3] Note that this logic might be reversed in a wartime or disaster setting, where the most critically ill patients may receive minimal care to ease their suffering. However, the need for balance still comes through.

- ICD9_CODE: The ICD-9 code representing the diagnosis. This value can be used to lookup more information in the dictionary table D_ICD_DIAGNOSES

One useful field missing from the data is the age of the patient at the time of admission. This can be computed from the provided DOB and ADMITTIME fields. When performing this computation, we find that a number of the patients have computed ages of around 300 years old. Referring to the Deidentification section of the MIMIC III report referenced earlier, we find that these are all patients with ages greater than 89 that have had their dates of birth shifted to comply with HIPAA privacy regulations; in particular, there is no guarantee that the relative order of the ages of two patients remains the same. Thus, for our analysis, we will set all of these patients to have age 91.

We will address other potential issues with the demographic data in the Data Visualization section below.

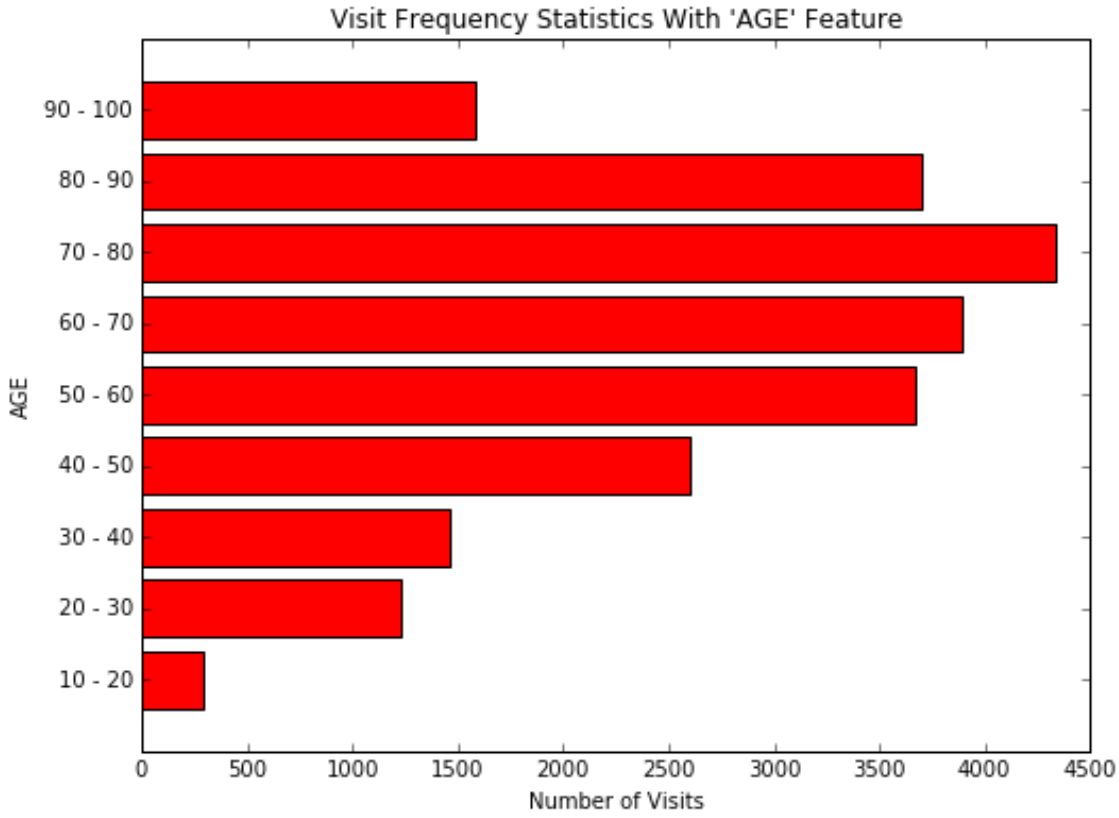
The following are some basic comparative statistics between the MIMIC-III data set as a whole and the data set we will be considering, where ADMISSION_LOCATION is EMERGENCY ROOM ADMIT. Note that the MIMIC-III numbers presented here differ from those in the MIMIC-III report; those summary statistics looked only at the patient population over 16.

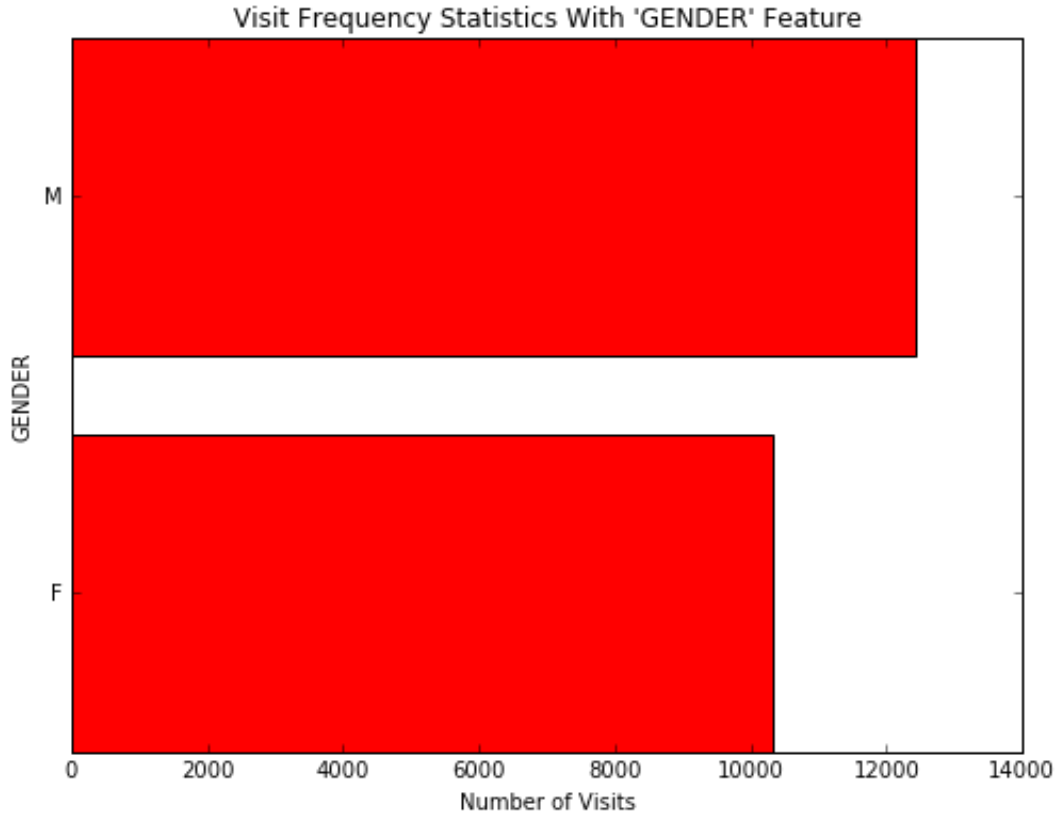
| Statistic | MIMIC-III | ER subset |
|---|---|---|
| Total visits | 58976 | 22754 |
| Total unique patients | 46520 | 17971 |
| Mortality rate | 0.099 | 0.136 |
| Mean age | 54.7 | 62.9 |
| Percentage of males | 55.9 | 54.6 |

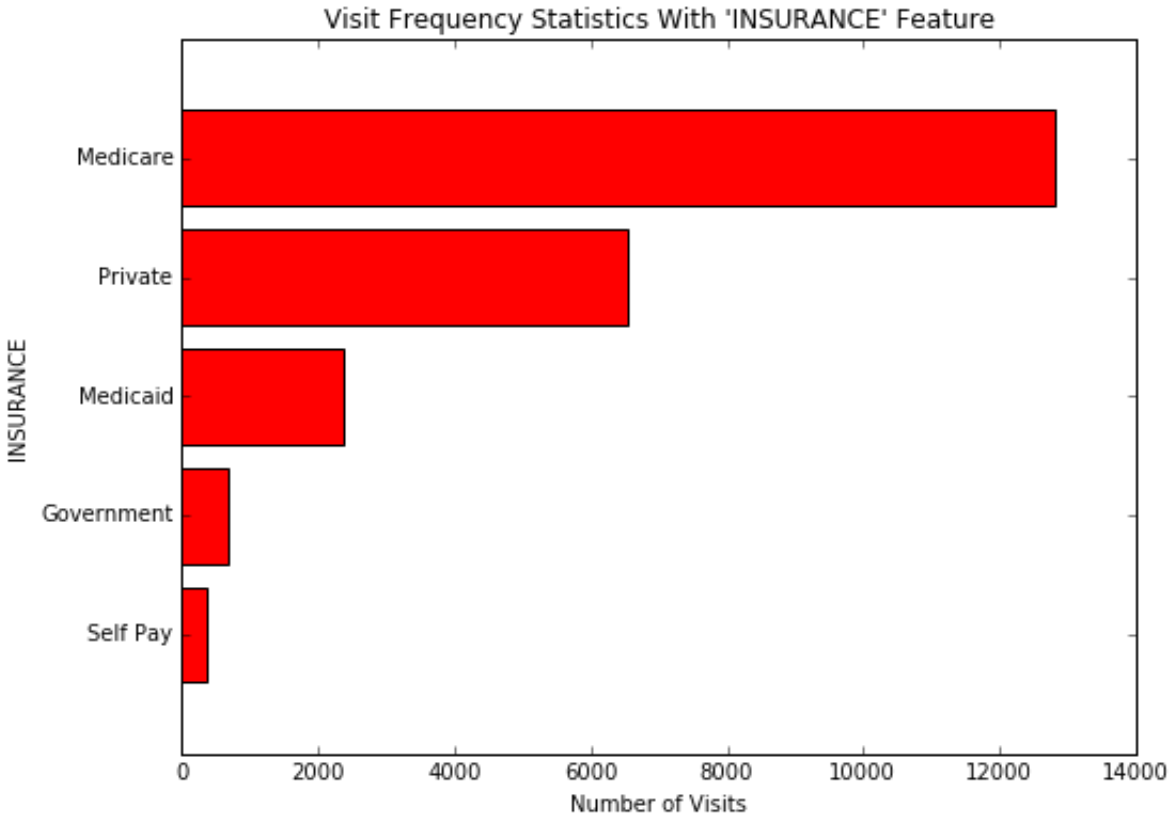
## Exploratory Visualization

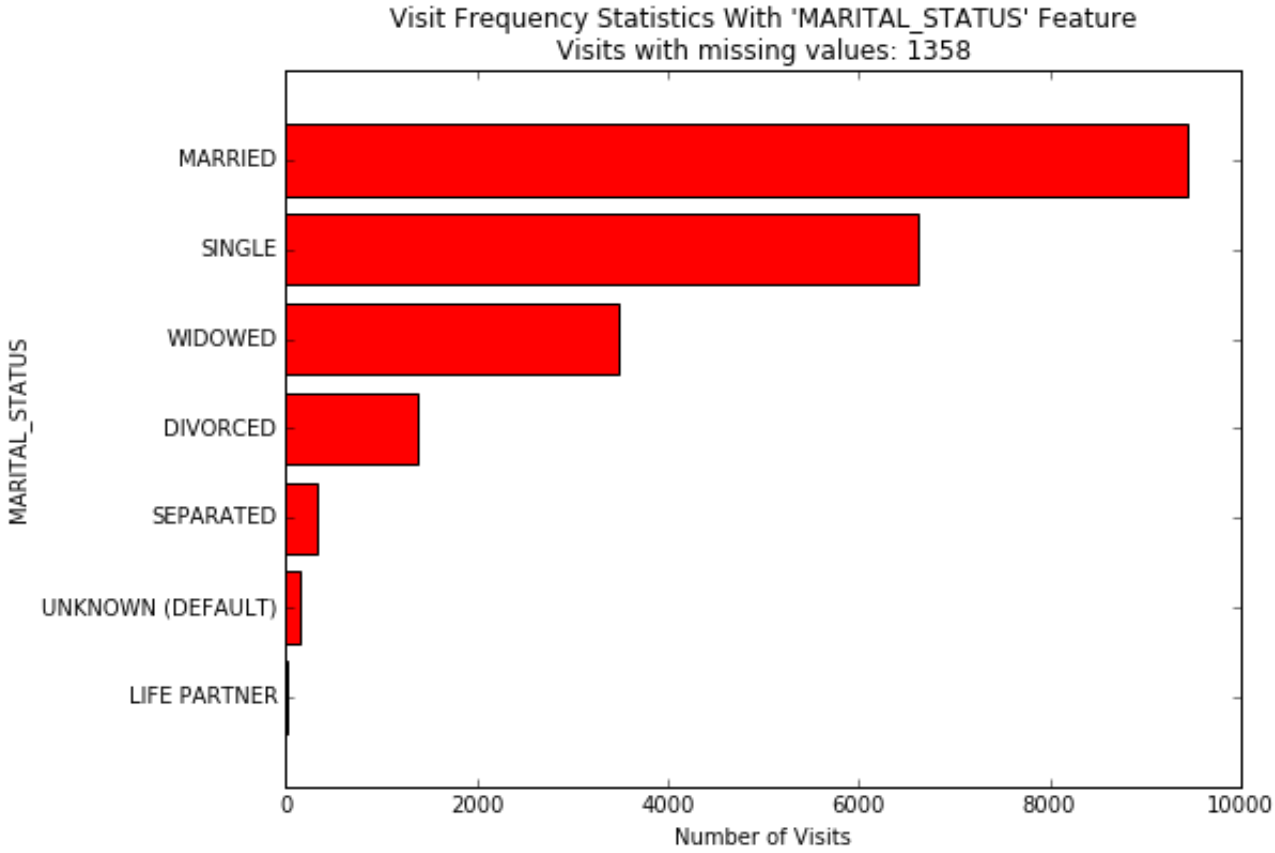
### *Demographic Data*

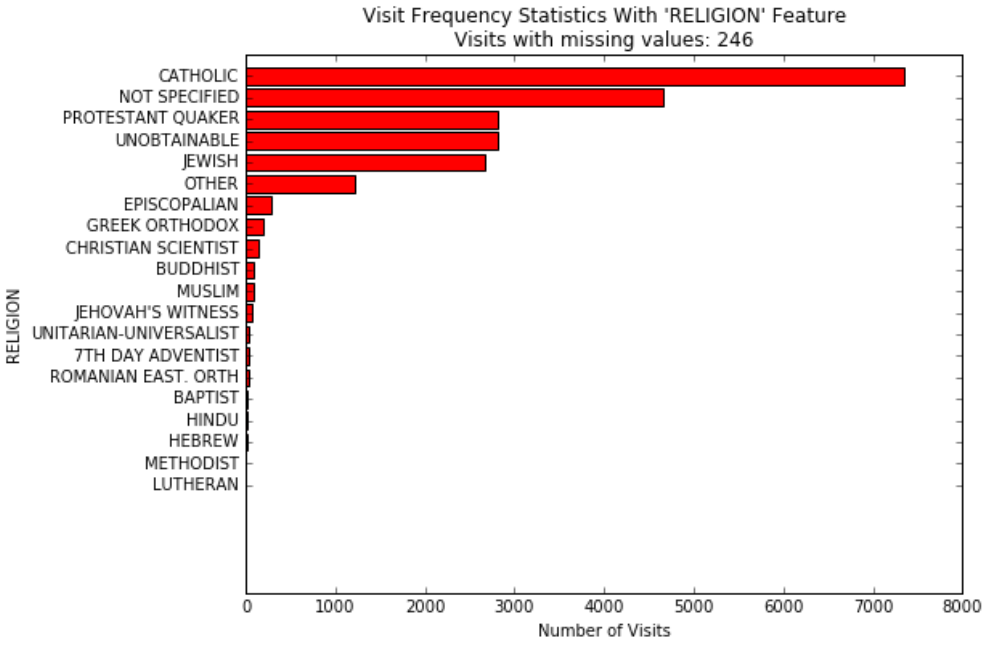
We start by looking at the demographic data. The following graphs of frequency statistics give us an idea of what our patient population looks like demographically.

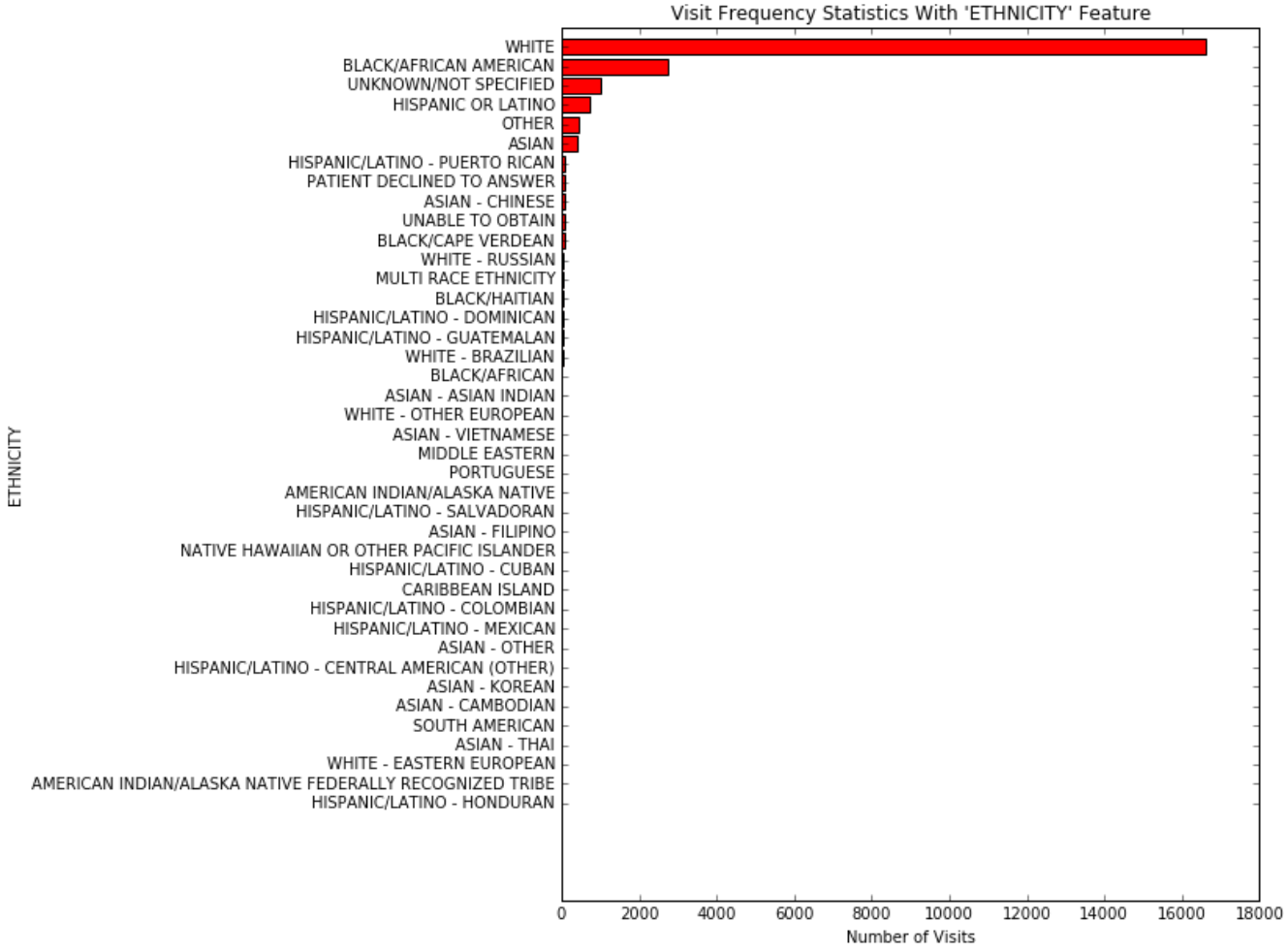

Visit Frequency Statistics With 'AGE' Feature
AGE
90 - 100
80 - 90
70 - 80
60 - 70
50 - 60
40 - 50
30 - 40
20 - 30
10 - 20
0
500
1000
1500
2000
2500
3000
3500
4000
4500
Number of Visits


Visit Frequency Statistics With 'GENDER' Feature
GENDER
M
F
0
2000
4000
6000
8000
10000
12000
14000
Number of Visits


Visit Frequency Statistics With 'INSURANCE' Feature
INSURANCE
Medicare
Private
Medicaid
Government
Self Pay
0
2000
4000
6000
8000
10000
12000
14000
Number of Visits


Visit Frequency Statistics With 'MARITAL_STATUS' Feature
Visits with missing values: 1358
MARITAL_STATUS
MARRIED
SINGLE
WIDOWED
DIVORCED
SEPARATED
UNKNOWN (DEFAULT)
LIFE PARTNER
0
2000
4000
6000
8000
10000
Number of Visits

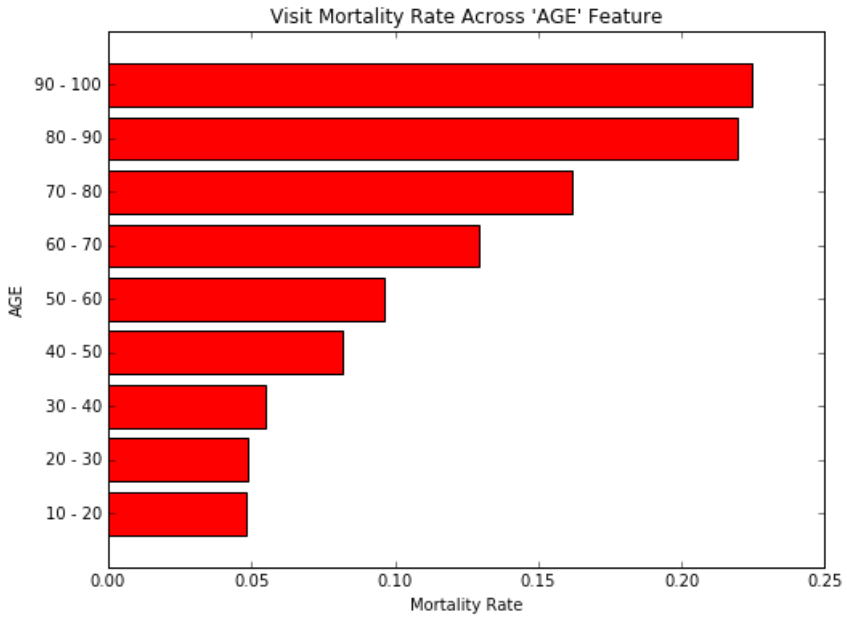

Visit Frequency Statistics With 'RELIGION' Feature
Visits with missing values: 246
CATHOLIC
NOT SPECIFIED
PROTESTANT QUAKER
UNOBTAINABLE
JEWISH
OTHER
EPISCOPALIAN
GREEK ORTHODOX
CHRISTIAN SCIENTIST
BUDDHIST
MUSLIM
JEHOVAH'S WITNESS
UNITARIAN-UNIVERSALIST
7TH DAY ADVENTIST
ROMANIAN EAST. ORTH
BAPTIST
HINDU
HEBREW
METHODIST
LUTHERAN
RELIGION
0
1000
2000
3000
4000
5000
6000
7000
8000
Number of Visits

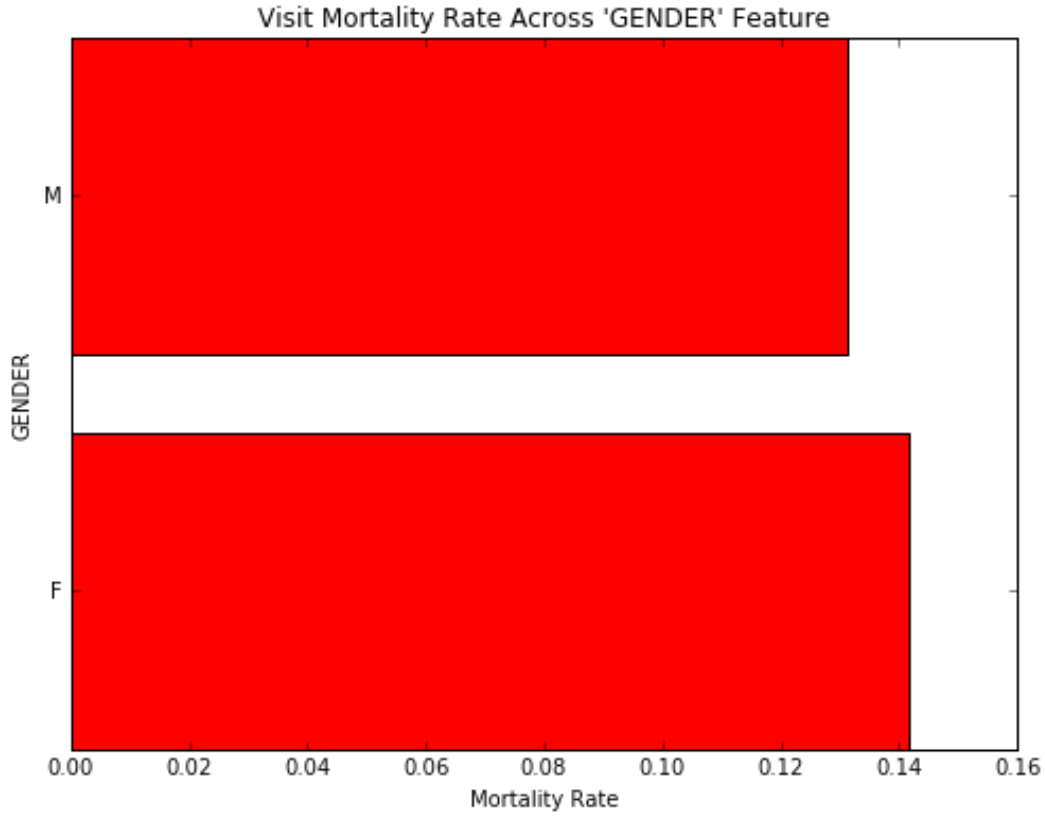

Visit Frequency Statistics With 'ETHNICITY' Feature
WHITE
BLACK/AFRICAN AMERICAN
UNKNOWN/NOT SPECIFIED
HISPANIC OR LATINO
OTHER
ASIAN
HISPANIC/LATINO - PUERTO RICAN
PATIENT DECLINED TO ANSWER
ASIAN - CHINESE
UNABLE TO OBTAIN
BLACK/CAPE VERDEAN
WHITE - RUSSIAN
MULTI RACE ETHNICITY
BLACK/HAITIAN
HISPANIC/LATINO - DOMINICAN
HISPANIC/LATINO - GUATEMALAN
WHITE - BRAZILIAN
BLACK/AFRICAN
ASIAN - ASIAN INDIAN
WHITE - OTHER EUROPEAN
ASIAN - VIETNAMESE
MIDDLE EASTERN
PORTUGUESE
AMERICAN INDIAN/ALASKA NATIVE
HISPANIC/LATINO - SALVADORAN
ASIAN - FILIPINO
NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER
HISPANIC/LATINO - CUBAN
CARIBBEAN ISLAND
HISPANIC/LATINO - COLOMBIAN
HISPANIC/LATINO - MEXICAN
ASIAN - OTHER
HISPANIC/LATINO - CENTRAL AMERICAN (OTHER)
ASIAN - KOREAN
ASIAN - CAMBODIAN
SOUTH AMERICAN
ASIAN - THAI
WHITE - EASTERN EUROPEAN
AMERICAN INDIAN/ALASKA NATIVE FEDERALLY RECOGNIZED TRIBE
HISPANIC/LATINO - HONDURAN
ETHNICITY
0
2000
4000
6000
8000
10000
12000
14000
16000
18000
Number of Visits

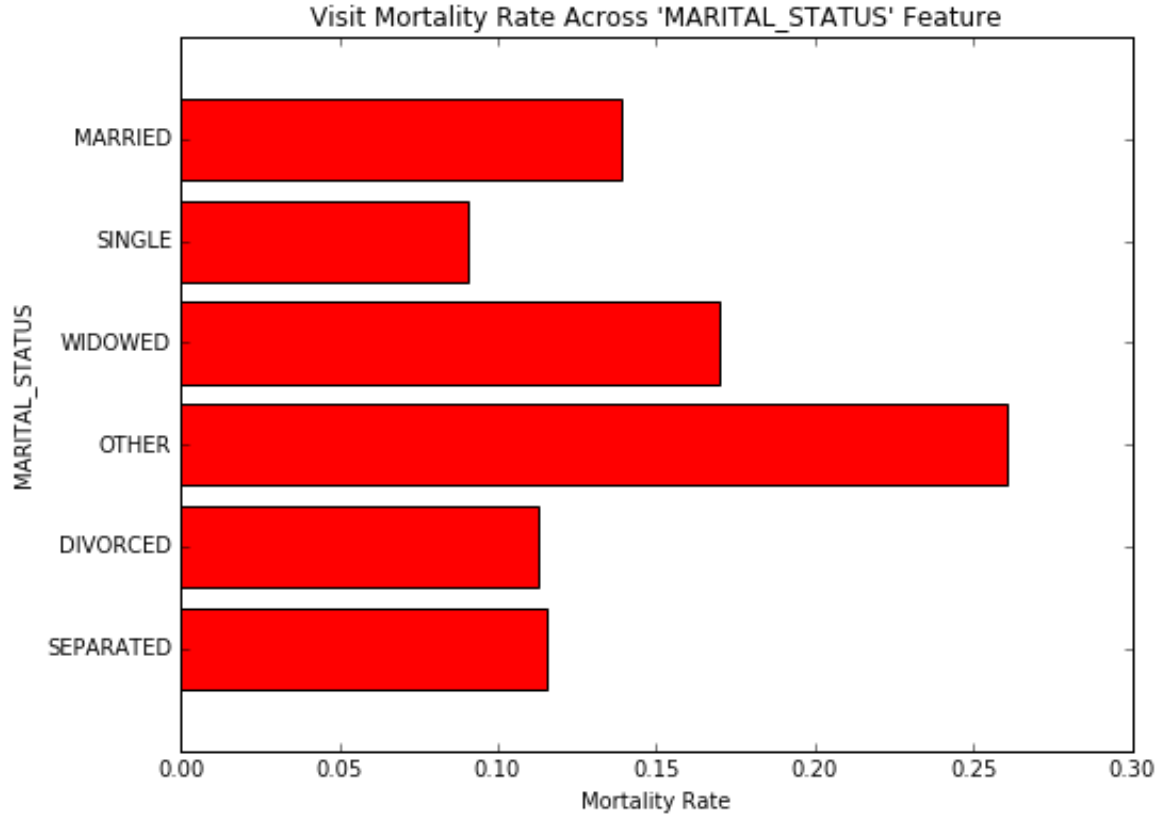
Besides giving us a feel for our population, the charts reveal a few issues with the data:

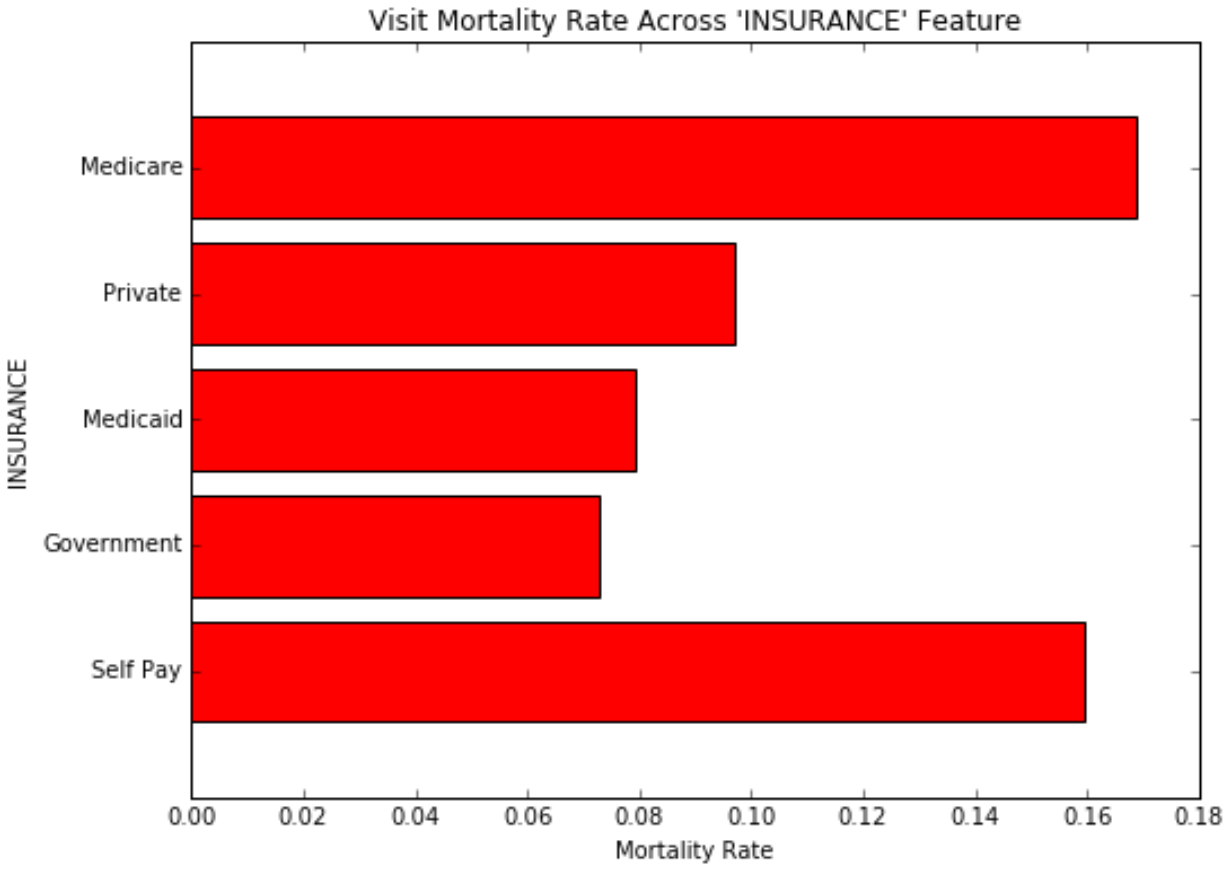
- Two of the dimensions, ETHNICITY and RELIGION, have long tails of infrequent values. This has a potential to be a problem for our analysis – for instance, if we found that we had only one patient of a certain ethnicity, and that patient died, we might come to the clearly erroneous conclusion that all patients of that ethnicity were doomed to die in the hospital. In addition, these additional vales will increase the number of dimensions our machine learning algorithm must deal with later. Thus, we should consider only values for the ETHNICITY field that occur sufficiently often in our data set; a threshold of 1% leaves us with seven values for RELIGION and six values for ETHNICITY. Looking at the other dimensions, this standard also trims the number of values for MARITAL_STATUS to five. Note that ETHNICITY and RELIGION have the catch-all value OTHER; it makes sense to group the infrequent values into this category, and we will add that value for MARITAL_STATUS as well.
- ETHNICITY and RELIGION also have various values representing the inability to obtain the data; these could indicate the data was simply not recorded or could indicate that conditions made it impossible to acquire the data (for instance, it is difficult to discover the religion of an unconscious, unaccompanied patient). In addition, both MARTIAL_STATUS and RELIGION have rows where the value is missing altogether. We will merge these values into the OTHER value as well.

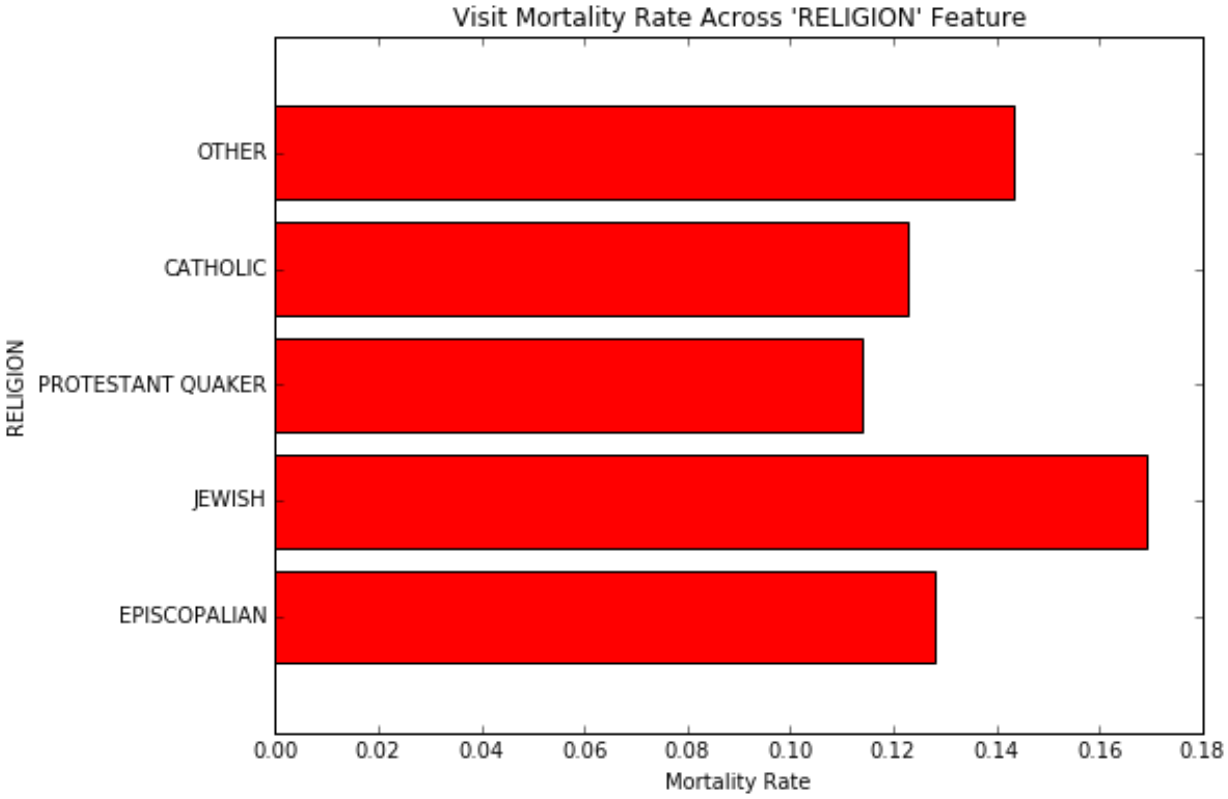
After this data cleanup, we now look at the mortality rate broken down across these dimensions.

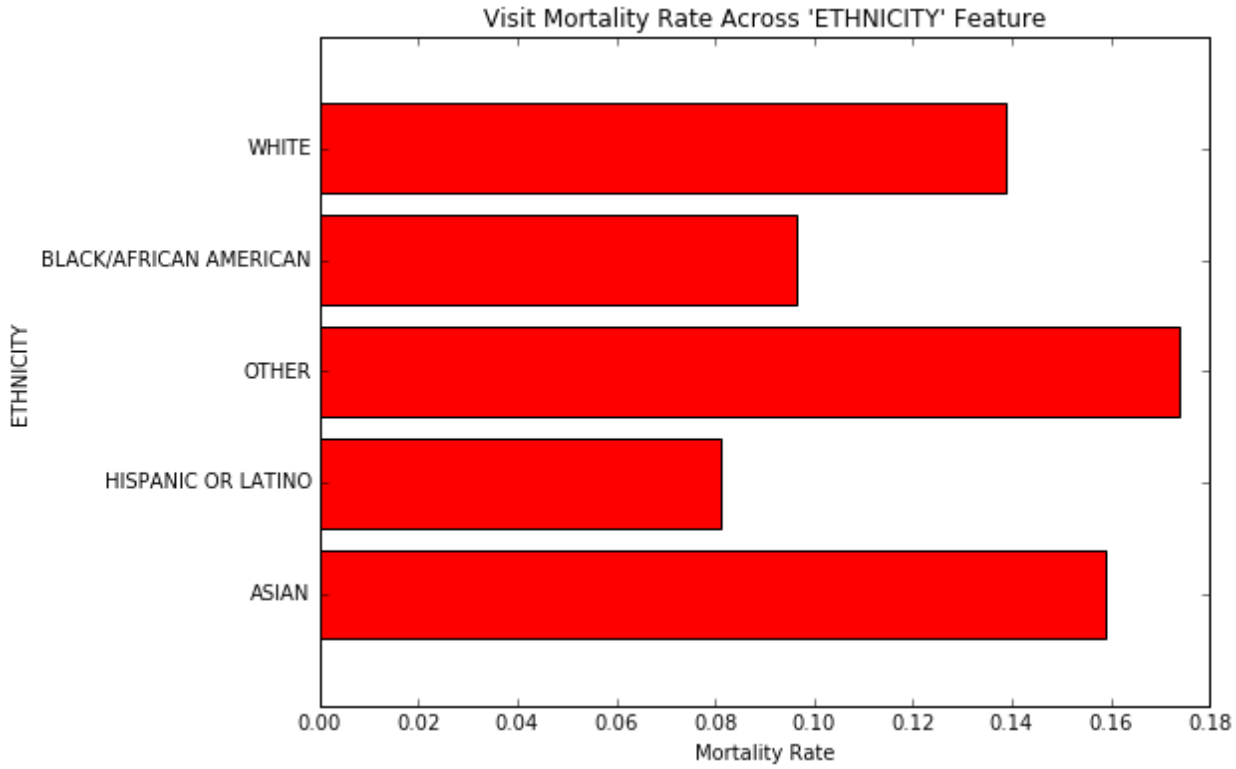



Visit Mortality Rate Across 'GENDER' Feature
GENDER
M
F
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
Mortality Rate


Visit Mortality Rate Across 'MARITAL_STATUS' Feature
MARITAL_STATUS
MARRIED
SINGLE
WIDOWED
OTHER
DIVORCED
SEPARATED
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Mortality Rate

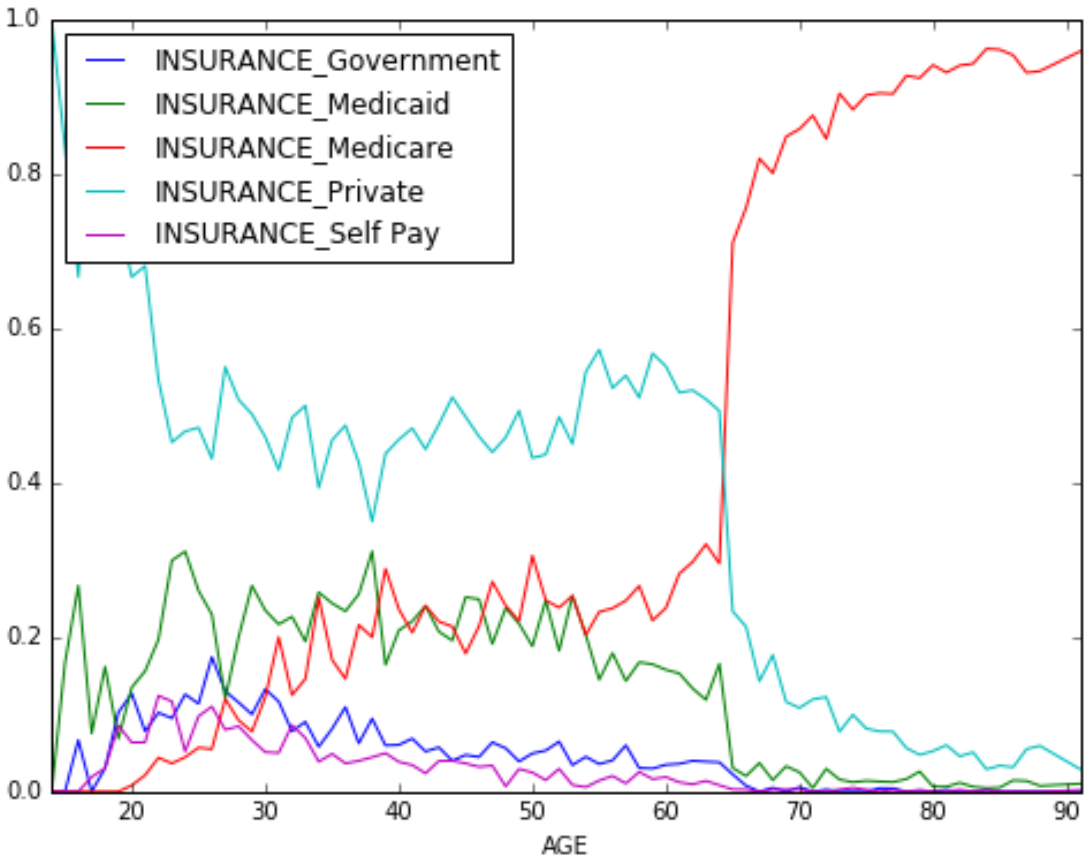

Visit Mortality Rate Across 'INSURANCE' Feature
INSURANCE
Medicare
Private
Medicaid
Government
Self Pay
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0.18
Mortality Rate


Visit Mortality Rate Across 'RELIGION' Feature
RELIGION
OTHER
CATHOLIC
PROTESTANT QUAKER
JEWISH
EPISCOPALIAN
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0.18
Mortality Rate




When we look at the graphs, we see that GENDER does not have a significant impact on mortality rate in and of itself; however, since we cannot dismiss the fact that it may be important in combination with other variables, we should leave it in place. On the other hand, AGE and mortality rate have a clear correlation; as patients grow older, the mortality rate increases. This makes intuitive sense.

The amount of variation in mortality rates across the other dimensions is interesting, and since we have chosen to only look at values for those dimensions that occur with some frequency, it is hard to dismiss what we are seeing as simply noise in the data. One hypothesis worth considering is that these other dimensions themselves correlate with age, and this hypothesis has some merit. For example, if we look at insurance with respect to age, we find that the percentage of Medicare patients is much higher for older patients; thus, we would expect Medicare patients to have a higher mortality rate by virtue of the fact that they are in general older.




However, this does not offer a complete explanation – for example, Self Pay patients also had an unusually high mortality rate, and we see in the graph above that the percentage of Self Pay patients declines with age. If we look at a pivot table of mortality rates by AGE and INSURANCE, we see that while we've explained some variation by correlation with AGE, the population of Self Pay patients still stands out as unusual.

| INSURANCE | Government | Medicaid | Medicare | Private | Self Pay |
|---|---|---|---|---|---|
| **AGE_BIN** | | | | | |
| **1** | 0.000 | 0.030 | NaN | 0.057 | 0.000 |
| **2** | 0.050 | 0.041 | 0.031 | 0.046 | 0.094 |
| **3** | 0.065 | 0.038 | 0.046 | 0.055 | 0.149 |
| **4** | 0.059 | 0.087 | 0.082 | 0.082 | 0.075 |
| **5** | 0.062 | 0.078 | 0.105 | 0.095 | 0.290 |
| **6** | 0.151 | 0.116 | 0.131 | 0.125 | 0.321 |
| **7** | 0.286 | 0.153 | 0.163 | 0.143 | 0.429 |
| **8** | 0.500 | 0.167 | 0.218 | 0.241 | 1.000 |
| **9** | NaN | 0.250 | 0.221 | 0.326 | 0.333 |

Graphs and pivot tables can be generated for the other dimensions in the accompanying code, and the conclusion is the same; correcting for the AGE dimension explains some of the variation, but not all of it. Some of these dimensions are highly unlikely to in and of themselves have a direct import on mortality rate; however, it is likely they are standing in for hidden socioeconomic variables that may in turn relate to the overall health of the patient even before he or she came to the ER. Therefore, we will leave them in as inputs for the learner.

### *Diagnosis Data*

We now turn to looking at the diagnosis data. There are 651047 diagnosis records in the MIMIC-III data set, containing 6984 unique ICD-9 codes; of those codes, 3333 are associated with at least one in-hospital death. Of these records, 258409 are associated with admissions through the ER, containing 5254 unique ICD-9 codes; of those codes, 2399 are associated with at least one in-hospital death.

At this point, we will turn our focus specifically to those diagnosis codes found in ER records. The most commonly found codes are summarized in the table below.

| ICD9_CODE | Visits | Deaths | Mortality Rate | LONG_TITLE |
|---|---|---|---|---|
| 4019 | 8247 | 1076 | 0.130 | Unspecified essential hypertension |
| 4280 | 6000 | 1000 | 0.167 | Congestive heart failure, unspecified |
| 42731 | 5210 | 1020 | 0.196 | Atrial fibrillation |
| 5849 | 4746 | 926 | 0.195 | Acute kidney failure, unspecified |
| 51881 | 3924 | 1136 | 0.290 | Acute respiratory failure |
| 25000 | 3704 | 573 | 0.155 | Diabetes mellitus without mention of complicat... |
| 41401 | 3572 | 402 | 0.113 | Coronary atherosclerosis of native coronary ar... |
| 5990 | 3389 | 434 | 0.128 | Urinary tract infection, site not specified |
| 486 | 2587 | 537 | 0.208 | Pneumonia, organism unspecified |
| 2724 | 2479 | 240 | 0.097 | Other and unspecified hyperlipidemia |

Note that there are differences between this list and the list of most frequent codes found in the MIMIC-III report. This is not strictly due to looking at only ER admissions; the MIMIC-III report of mpost common diagnoses looks only at primary diagnoses (SEQ_NUM = 1) and the codes associated with ICU stays.

The codes with the highest mortality rates are:

| ICD9_CODE | Visits | Deaths | Mortality Rate | LONG_TITLE |
|---|---|---|---|---|
| 6023 | 1 | 1 | 1 | Dysplasia of prostate |
| 43381 | 1 | 1 | 1 | Occlusion and stenosis of other specified prec... |
| 71895 | 1 | 1 | 1 | Unspecified derangement of joint, pelvic regio... |
| E9053 | 1 | 1 | 1 | Sting of hornets, wasps, and bees causing pois... |
| 31230 | 1 | 1 | 1 | Impulse control disorder, unspecified |
| 1726 | 1 | 1 | 1 | Malignant melanoma of skin of upper limb, incl... |
| 74512 | 1 | 1 | 1 | Corrected transposition of great vessels |
| 81381 | 1 | 1 | 1 | Closed fracture of unspecified part of radius ... |
| 85309 | 1 | 1 | 1 | Other and unspecified intracranial hemorrhage ... |
| 3431 | 1 | 1 | 1 | Congenital hemiplegia |

As with the demographic data, we have an outliers issue. There are three approaches to consider:

- Does it help if we limit ourselves to the primary diagnosis, as in the MIMIC-III report? Unfortunately, the answer is no; there are still many singleton values even in the primary diagnosis codes. This would also prevent us from finding interactions between diagnoses that would allow us to better predict mortality.
- Can we use the same approach as with the demographic data, where we limit ourselves to diagnoses occurring in at least 1% of the visits? This doesn't quite work, as there are only nine diagnosis codes occurring that frequently. But the idea is right if we adjust the threshold. If we look only at codes occurring at least 100 times in the ER data, we end up with 446 unique codes. The highest mortality rates are shown in the table below.

| ICD9_CODE | Visits | Deaths | Mortality Rate | LONG_TITLE |
|---|---|---|---|---|
| V667 | 405 | 298 | 0.736 | Encounter for palliative care |
| 2866 | 199 | 125 | 0.628 | Defibrination syndrome |
| 4275 | 599 | 364 | 0.608 | Cardiac arrest |
| 3481 | 219 | 129 | 0.589 | Anoxic brain damage |
| 5724 | 103 | 53 | 0.515 | Hepatorenal syndrome |
| 570 | 385 | 190 | 0.494 | Acute and subacute necrosis of liver |
| 41091 | 148 | 66 | 0.446 | Acute myocardial infarction of unspecified sit... |
| 78551 | 400 | 176 | 0.440 | Cardiogenic shock |
| 1972 | 123 | 54 | 0.439 | Secondary malignant neoplasm of pleura |
| 42741 | 206 | 90 | 0.437 | Ventricular fibrillation |

- A last idea to consider is collapsing the individual diagnosis codes into groups of codes. A table in the MIMIC-III report[4] suggests a breakdown into ten different groups; evaluating the mortality rate amongst those ten groups, we find values ranging from a low of .031 to a high of .240. We would expect the loss of granularity from using this grouping to be too much for the learner to overcome; however, this grouping might work well as a supplement to keep some of the data lost by looking at the top 446 codes.

## Algorithms and Techniques

We are working with a labeled data set, where each row of data will represent a single visit and include a binary value indicating whether or not the patient lived or died, and we would like our system to be able to apply labels to new sets of patient data. Therefore, it is appropriate to consider supervised classification algorithms. We will start off by considering a variety of algorithms and, based on initial results, choose one or two to tune further. The algorithms that we will implement are:

- Naïve Bayes – this uses the data to attempt to learn an underlying probabilistic model. If it works, it has the advantages of letting us comparatively rank patients based on their relative risks and providing us with a model that we can mine for explanations of the decision.
- Decision Tree – this technique learns a tree of rules that fit the training data. The result is a binary yes or no answer, but, like Naïve Bayes, we can get an understandable explanation of the decision from the classifier.
- K-Nearest Neighbors – this technique searches the provided training data and looks for those entries most like the data that we are trying to classify. Again, we can get a explanation from the system, but now it will be based on showing the comparable patient records considered.
- AdaBoost – this ensemble algorithm constructs a learner by iteratively accumulating a sequence of weak learners – in this case, we will be using one-step decision trees as the weak learners. The success or failure of each weak learner to classify a record is used to weaken or strengthen that record's contribution to choosing new learners in future

[4] http://www.nature.com/articles/sdata201635/tables/2

iterations of the process. Unlike the other algorithms, this algorithm is rather opaque when it comes to mining the rationale for the decision in a human understandable way.

- Random Forest – this is another ensemble algorithm based on decision trees. Again, a classifier is built up iteratively by accumulating a sequence of decision trees; in this case, each decision tree is built using a training set drawn randomly with replacement from the overall training set and considers a subset of features randomly drawn from the overall set. Once again, we end up with an opaquer result than the first three algorithms.

A support vector classifier algorithm was also considered, but some early tests revealed that it was significantly slower on this data set without improving the results.

For the initial phase, we will be using the default values for each of these classifiers as specified in the sklearn implementations. For cross validation purposes, the data will be divided into a training set and a test set – the test set will comprise 20% of the overall data.

## Benchmark

As mentioned earlier, we will be using the F1 score to evaluate our classifiers. As a benchmark, let us consider two naïve classifiers:

- Consider a random classifier, where each record is assigned a mortality flag of 1 with probability equal to the mortality rate for the set (.115). Another way to express this is that we choose a random subset of the data, where each record has a probability of being included in the subset equal to the mortality rate, and then assign the members of this subset a mortality flag value of 1. This classifier will, on average, have both a precision and a recall equal to the mortality rate. To see why this is so, consider the following:
    - Precision: Precision is the ratio between the correct positive predictions and the total positive predictions. If we choose a random subset of the records and assign them a positive prediction, then the percentage of actual deaths in this subset (i.e. the correct predictions) will, on average, be the same as the percentage of actual deaths in the data set as a whole (i.e. the mortality rate).
    - Recall: Recall is the ratio between the correct positive predictions and the total actual positive values. When choosing our random subset, each actual positive value has a probability of being placed in the subset equal to the mortality rate. Thus, the ratio of

actual values in the subset to actual values in the set as a whole will, on average, be this same probability, the mortality rate.

When the precision and recall are equal, the F1 score is equal to the same number – thus the F1 score will, on average, be .115.

- Consider a classifier where each record is assigned a mortality flag value of 1; i.e. everyone is classified as dying. It is easy to see that this classifier has a precision equal to the mortality rate, a recall of 1, and a F1 score of .206.

Therefore, we will consider a benchmark score of .206 as a minimum standard for our classifiers.

# III. Methodology

## Data Preprocessing

The following data pre-processing steps were taken on the data; some of these were discussed in detail in the Data Exploration section above.

### *Demographic Data*

- The PATIENTS_DATA_TABLE and ADMISSIONS_DATA_TABLE were both read in using pandas. The pandas CSV parsing method attempted to convert the HADM_ID field into integers; since in later steps in the processing, pandas subsequently converted them into floats, this was suppressed by forcing the field to be read in as a string.
- The PATIENTS and ADMISSIONS table were merged on the field SUBJECT_ID; this resulted in a data set with one row for each hospital visit.
- An AGE field was added to each record based on the DOB and ADMITTIME fields. Where this resulted in a value > 300, due to deidentification, the value was uniformly set to 91.
- The data was filtered down to emergency room visits, based on the criterion ADMISSION_LOCATION = ‘EMERGENCY ROOM ADMIT’.
- Values in the MARTIAL_STATUS, ETHNICITY and RELIGION fields that occurred in less than 1% of the ER visits, as well as all values of ‘None’, were replaced with the value ‘OTHER’

- The data set was reduced to the HADM_ID, HOSPITAL_EXPIRE_FLAG, AGE, GENDER, ETHNICITY, INSURANCE, MARTIAL_STATUS and RELIGION fields.
- The pandas get_dummies function was used to replace the GENDER, ETHNICITY, INSURANCE, MARITAL_STATUS and RELIGION fields with binary-valued columns for each of the remaining values. This was a necessity for some of the learning algorithms.

### *Diagnosis Data*

- The DIAGNOSIS_ICD_DATA_TABLE was read in, again with the HADM_ID field being forced to a string.
- The data was trimmed down to ER admissions by only considering HADM_ID fields found in the ER demographic data.
- The data set was reduced to the HADM_ID and ICD9_CODE fields.
- Three records were found to have no ICD9_CODE value – they were removed from the data set.
- A subset of the data containing only codes that appeared more than 100 times in the ER data (er_frequent_diagnosis_data) was derived.
- The code groupings from the MIMIC-III data set were used to create a data set with the addition of a DIAGNOSIS_GROUP field with values ranging from ‘A’ to ‘J’, but with no ICD9_CODE (er_grouped_diagnosis_data).
- The pandas get_dummies function was used to replace the ICD9_CODE and DIAGNOSIS_GROUP fields in the respective data sets with binary-valued columns.
- The er_grouped_diagnosis_data and er_frequent_diagnosis_data sets were merged back together on HADM_ID using an outer join. The null values resulting from the full outer join were replaced with values of 0. This results in a data set with one row for each ER admission for which diagnosis data is available.

## Implementation

The five classifiers discussed earlier are coded using their implementations in sklearn, with default values; all classifiers that support a random_state parameter have that value set to 38. The data is split with the sklearn.cross_validation.train_test_split function, with test_size set to 0.2 and random_state set to 42. The sklearn.metrics.f1score function is used to score each classifier.

The experiment was performed in three phases:

- Demographic data alone
- Diagnoses data alone, with labels drawn from the demographic data by using the HADM_ID field
- Demographic data and diagnosis data together, merged on HADM_ID

The F1-scores from the various trials are shown in the table below.

| | Demographic | Diagnoses | Combined |
|---|---|---|---|
| **Naïve Bayes** | 0 | 0.028 | 0.052 |
| **Decision Tree** | 0.185 | 0.395 | 0.397 |
| **KNN** | 0.047 | 0.029 | 0.048 |
| **AdaBoost** | 0 | 0.449 | 0.428 |
| **RandomForest** | 0.105 | 0.342 | 0.255 |

Some observations to make:

- None of the classifiers did better than the defined benchmark on demographic data alone. In fact, the GaussianNB classifier did so poorly that it did not predict **any** positive values; by definition, this meant the precision and thus the F1 score were not well-defined, and the sklearn scorer assigned both a value of 0.0.
- On diagnoses data, Naïve Bayes and KNN both fell short of the benchmark again; Decision Tree, AdaBoost and RandomForest, however, all outperformed our naïve benchmark classifier.
- On the combined data, Naïve Bayes and KNN improved, but not enough to improve over the benchmark. Decision Tree, AdaBoost and Random Forest all outperformed the benchmark again; interestingly, though, AdaBoost and Random Forest did worse with the additional data.

The last is an interesting result, particularly in light of the fact that there *did* appear to be a few trends regarding mortality rate in the demographic data. There are two possibilities to consider:

- The demographic data may be so noisy that it contributes negatively to the analysis.
- The demographic data may have a contribution when combined with the diagnosis data, but tuning of the machine learner is necessary to make best use of the data.

Most of the complications in the implementation did not come from implementing the classifiers themselves or building helper functions for that analysis, outside of the usual debugging when writing code; in fact, the Jupyter Notebook IDE simplified this process when compared with

other languages. The complications that surfaced from trial runs of the experiment had to do with data issues that were revealed. The need to force HADM_ID to be loaded as a string was noted above; some other items of note:

- Comparing statistics from trial runs of the analysis with the statistics in the MIMIC-III paper revealed the inadvertent use of the wrong flag for patient death (EXPIRE_FLAG, which indicates if the patient has died, instead of HOSPITAL_EXPIRE_FLAG, which looks at whether the patient died specifically during the hospital visit). Not too surprisingly, the demographic data classifier performed better with this flag, since it was able to note that older patients were more likely to have died already, whether in or out of the hospital. On the other hand, the diagnosis data classifier performed worse.
- Trial runs of generating dummy columns in the data also forced the issue of cutting down the dimensionality of the diagnoses data (discussed earlier) to be addressed early on. The problem was not with the accuracy of the classifiers – rather, it was with the computer being used running out of memory. Memory issues were also why a decision was made early on to not look at the 28 GB table of chart events.
- Finally, not so much a complication but a result of note is that the GaussianNB classifier performed so poorly on demographic data alone that it

## Refinement

Based on the results from the preceding section, we will refine the AdaBoost and Decision Tree classifiers further by tuning their parameters using the sklearn GridSearchCV function. In addition, we will look at tuning on the diagnoses set alone vs. tuning on the combined set, to see if that makes a difference. We will tune the following parameters for each classifier:

- AdaBoost: n_estimators (default = 50) and learning_rate (default = 1.0)
- Decision Tree: criterion (default = ‘gini’), min_samples_split (default = 2) and max_features (default = ‘sqrt’)

In each pass, the GridSearchCV function will be looking for the parameter set that achieves the highest mean F1 score on the training data. In the tables below, we will note both this score and the F1 score achieved on the test data.

### *AdaBoost, combined data*

The AdaBoost classifier on the combined data was tuned over four passes; the parameters used and best results for each pass are summarized in the following table. The mild regression downwards of the F1 score on the test data in the final step indicates that we may be starting to overfit the training data.

| n_estimators | learning_rate | best | CV F1 score | Test F1 score |
|---|---|---|---|---|
| 50-300 (increments of 50) | 0.2 - 1.0 (increments of 0.2) | 300, 0.8 | 0.473 | 0.475 |
| 300-500 (increments of 50) | 0.6 - 1.0 (increments of 0.2) | 500, 1.0 | 0.485 | 0.487 |
| 500-750 (increments of 50) | 0.6 - 1.0 (increments of 0.2) | 700, 1.0 | 0.485 | 0.495 |
| 650-750 (increments of 10) | 0.8 - 1.0 (increments of 0.05) | 730, 0.9 | 0.489 | 0.480 |

### *AdaBoost, diagnosis data only*

The AdaBoost classifier on the diagnosis data alone was tuned over two passes; the parameters used and best results for each pass are summarized in the following table.

| n_estimators | learning_rate | best | CV F1 score | Test F1 score |
|---|---|---|---|---|
| 50-300 (increments of 50) | 0.2 - 1.0 (increments of 0.2) | 250, 1.0 | 0.459 | 0.473 |
| 200-300 (increments of 10) | 0.8 - 1.0 Iincrements of 0.05) | 220, 1.0 | 0.459 | 0.476 |

### *Decision Tree, combined data*

The Decision Tree classifier on the combined data was tuned over two passes; the parameters used and best results for each pass are summarized in the following table.

| min_samples_splt | criterion | max_features | best | CV F1 score | Test F1 score |
|---|---|---|---|---|---|
| 2-20 (increments of 2) | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 18, None | 0.407 | 0.420 |
| 17-19 | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 18, None | 0.407 | 0.420 |

### *Decision Tree, diagnosis data only*

The Decision Tree classifier on the diagnosis data alone was tuned over four passes; the parameters used and best results for each pass are summarized in the following table.

| min_samples_splt | criterion | max_features | best | CV F1 score | Test F1 score |
|---|---|---|---|---|---|
| 2-20 (increments of 2) | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 20, None | 0.398 | 0.395 |
| 20-40 (increments of 2) | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 40, None | 0.403 | 0.407 |
| 40-60 (increments of 2) | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 40, None | 0.403 | 0.407 |
| 38-42 | ‘gini’, ‘entropy’ | 'sqrt', 'log2', None | 'entropy', 40, None | 0.403 | 0.407 |

# IV. Results

## Model Evaluation and Validation

In looking at the ability to predict the likelihood of a patient dying in an emergency room visit using data from the patient’s healthcare records associated with the visit, we tested a variety of supervised machine learning algorithms on the data, using a consistent split between training and testing data across all of the algorithms. This was followed by parameter tuning of the classifiers of the best two candidates. In addition, based on some interesting results, we considered whether or not the demographic data helped or hindered our analysis.

We have found that, of the supervised learning models considered, a tuned AdaBoost classifier achieves the best results. The demographic data by itself does not lead to useful results, however, it does lead to a small improvement when combined with diagnoses data, as compared to looking at the diagnoses data alone. This improvement comes at a performance cost, however – the AdaBoost classifier using combined data requires 730 weak estimators to achieve best results, while the AdaBoost classifier using only diagnoses data requires only 230 weak estimators.

One question we have not yet considered is the robustness of the resulting classifier. To consider this, we ran a sequence of 100 experiments using the final AdaBoost classifier, this time removing the random_state parameter from both the classifier and the training vs. testing split. The mean F1 score of the group on the test data was .487, with a standard deviation of .018; the minimum F1 score was .444 and the maximum was .526, and the middle two quartiles ranged from .474 to .500. Thus, our result was fairly typical; we can expect that the model will perform similarly on other data sets.

### Justification

Given that the F1 score of our classifier is still well below 1, it is hard to say that we have "solved the problem". However, we established a benchmark F1 score of .205 by looking at a classifier that assumes every patients dies in the hospital. Against this benchmark, we have an improved result, since our final classifier has an F1 score of .480 on the test data. It is also informative to look at the precision and recall of the classifier.

- The precision of our tuned classifier on the test data set was .661 – this means that of those patients identified by the classifier as at risk for dying in the hospital, nearly two-thirds did die, while the other one-third were false positives. Given that the mortality rate for the ER patients was only .136, we have achieved a precision five times better than random selection or the "everyone dies" classifier. This seems to be a good result.
- The recall of our tuned classifier is .377. This isn't as great a result; only slightly more than one-third of the time did we correctly identify that someone who was going to die based on the records. However, this is still over twice as good as random selection.

The precision and recall results suggest that, if this system were being used in a decision-aiding capacity, we would want to pay more attention to positive results than negative ones. It would also be interesting to see how the precision and recall of the system compare to a domain expert; would a medical professional looking at the data have the same difficulties in classifying the patients? This would be a different way of evaluating the effectiveness of the system.

We have at least established the thesis that machine learning algorithms can provide a positive contribution to this problem. There are a number of avenues that could be pursued that might improve our results further; those will be discussed in the Improvements section.

# V. Conclusion

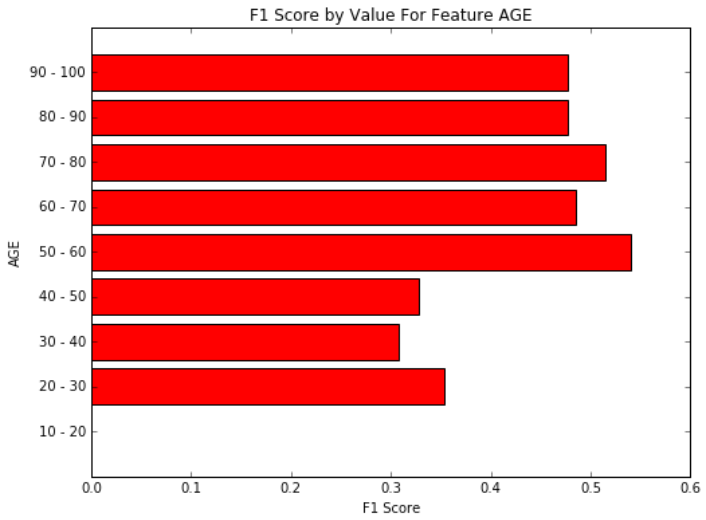
### Free-Form Visualization

In our original data exploration, we noted a few interesting characteristics of the relationship between the data and mortality rate:

- As age rose, so did the mortality rate.
- Independent of age, being a Self Pay patient appeared significant.
- Finally, we noted the diagnoses with the top 10 mortality rates.

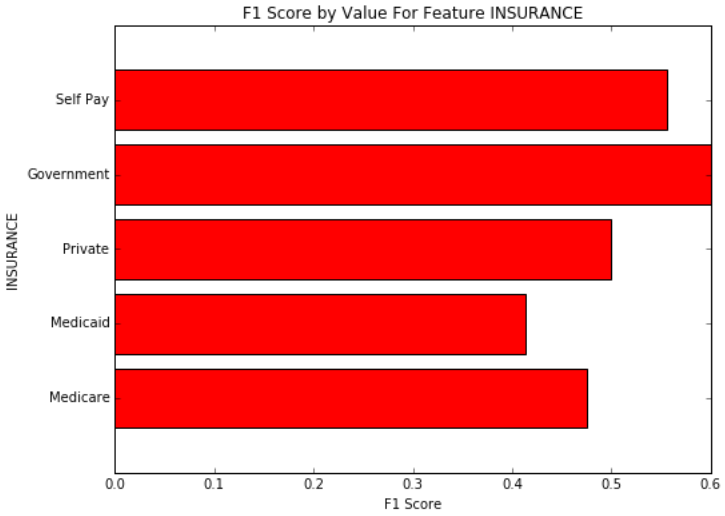
It would be interesting to see how our AdaBoost classifier does in each of these situations; does it do a better job predicting the result for certain values of each dimension? To look at this, we will split the data into training and testing sets as before, but this time, when we test the data, we will pick out subsets based on the conditions above and look at the F1 scores of the subsets independently. The objective here is to see how well our classifier performs in these data regimes of interest.

For AGE, we can generate the following chart.



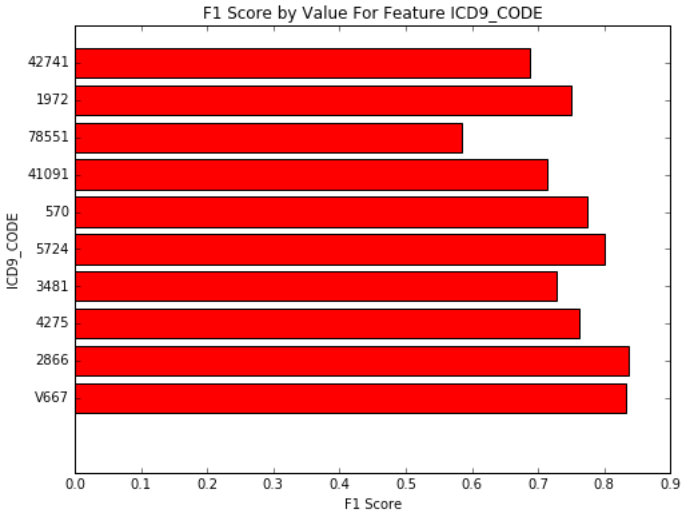

We note that the classifier has an F1 score of 0 for the youngest age range; given the relative infrequency of this range, this isn't a surprising result. More interesting is that the classifier performs poorly for patients aged from 20-50; some of this can again be explained by the frequency of data in the set, and some by the lower than average mortality rate in these ranges. Striking is that the classifier performed best in the age range 50-60, where the mortality rate still hovered around .10; this implies that we might want to give the classifier results more weight when looking at this group.

Turning to the INSURANCE feature, we can generate the following chart:




A striking note here is that the classifier does so well with the Government and Self Pay patients, despite the fact that these are the two most infrequent values in the data set. It would appear that the classifier is picking up some of what makes the Self Pay insurance category distinctive, and this might bear further scrutiny.

Finally, looking at the ICD-9 codes, we generate the following chart, for the ten diagnosis codes with the highest mortality rates.




We can see that the classifier does do a better job overall with these diagnoses. Some caution has to be considered in interpreting this chart; if we defined a naïve classifier that assigned a value of 1 to the expire flag for each of these, we would have a benchmark score significantly higher than .205. The table below summarizes these modified benchmarks for these codes.

| ICD9_CODE | Benchmark |
|---|---|
| V667 | 0.848 |
| 2866 | 0.771 |
| 4275 | 0.756 |
| 3481 | 0.741 |
| 5724 | 0.680 |
| 570 | 0.661 |
| 41091 | 0.617 |
| 78551 | 0.611 |
| 1972 | 0.610 |
| 42741 | 0.608 |

As we can see, we have a mixed bag of results – for some diagnosis codes, the classifier is clearly doing a better job of distinguishing between patients, whereas for others, the naïve classifier is outperforming. These could be considered areas for further research and improvement.

## Reflection

In this project, we have looked at the use of machine learning algorithms to study a question arising in healthcare; specifically, whether patient records can be used to predict mortality in an emergency room setting. To do this, we went through phases of data acquisition, data exploration and cleansing, initial machine learning experimentation and final tuning of the chosen machine learner.

We started with a copy of the MIMIC III data set, a deidentified data set based on actual hospital records used for research purposes, and read both the online documentation and the summary report describing the data set. Even though the data set has been deidentified, it is still sensitive in nature, so there was a slight delay in getting started while satisfying some certification requirements to gain access to the data.

We then made some decisions about which parts of the data to use and started to explore and clean the data. We ran into an issue with computing ages that was resolved by referring to the MIMIC-III report, and then turned to considering how to reduce the dimensionality of the data and handle clear dimensional outliers. This is the part of the project that would have benefited most from interaction with a domain expert in emergency room medicine. In particular, the diagnosis groupings used by the MIMIC-III authors might not be the best groupings for use in ER medicine, and there might have been other elements of the data that an ER physician could

have pointed out would be useful in the analysis. This was the area in which the most time was spent and was the most interesting to explore.

Next, we chose a number of different supervised machine learning algorithms presented in the course and applied them to the data using their default parameters as specific in the respective sklearn implementatons. A support vector classifier was considered, but between the size of the data set, the number of dimensions and the lack of linear separability, it did not perform well both in terms of speed and score, so it was dropped from the suite of experiments. There was some iteration between this phase and the preceding one; we first got the algorithms running on demographic data and then moved on to diagnosis data alone and the merged data set.

Finally, we took the two machine learners which stood out as the best and tuned them using a grid search methodology. This took several passes to find the right intervals in parameter space, as well as taking a fair amount of time on the clock; this section required the most patience, and provided a chance to work more on the report.

The final results were not “mind-blowing”, as it were, but were an improvement over the benchmarks considered. The use of machine learning on this problem is worth more study. The only caveat to using this technique in a general setting, particularly with AdaBoost, is the lack of an easy explanatory component; a physician or nurse using this as part of triage would like to know *why* the system reached the conclusions that it did, in order to validate the results against his or her own knowledge as a second opinion. But as a way to flag a patient for further consideration that might otherwise slip through, the technique has potential merit.

Finally – yes, this was a fun data set with which to work.

## Improvement

There are several directions we could go to possibly improve on the results. All of these generally involve finding ways to modify the data used for analysis.

One parameter of the technique used in this report that was not tuned is the threshold for diagnosis codes explicitly considered. Instead of only considering codes appearing 100 or more times, what happens if we only consider codes occurring even more or less frequently? The case for considering more codes is intuitive, but not necessarily beneficial; the more features we add, the more risk we have of running into the curse of dimensionality and the less effective our learner can become. Given that, in the final tuning step, our cross-validation F1 score and

testing F1 score starting to diverge, we may be on the edge of this issue already; as such, the possibility that fewer features would improve our results must be considered.

A related idea to the above would be finding a way to determine which diagnosis codes are the most significant in terms of separating the data by comparing results with and without each code. This is a computationally time-consuming approach if done naively, but there may be a way to mine these results from the tuned AdaBoost classifier. It may also be possible to derived this information directly from the AdaBoost classifier.

Finally, in this project, we have only used a portion of the data available in the MIMIC-III data set, and a clear direction to explore would be making use of more of the data available; the data on lab and microbiology tests and the chart data appear to be logical next steps. However, because of the volume of data, it would be critical to reduce the dimensionality of the diagnosis and demographic data used *and* to carefully choose the subset of the new data to consider;

# References

Hossein Alimohammadi, Farahnaz Bidarizerehpoosh, Farzaneh Mirmohammadi, Ali Shahrami, Kamran Heidari, Anita Sabzghabaie, Shahram Keikha. Cause of Emergency Department Mortality; a Case-control Study. Emerg (Tehran) 2014 Winter; 2(1): 30–35.

Johnson AEW, Pollard TJ, Shen L, Lehman L, Feng M, Ghassemi M, Moody B, Szolovits P, Celi LA, and Mark RG. MIMIC-III, a freely accessible critical care database. Scientific Data (2016). DOI: 10.1038/sdata.2016.35. Available from http://www.nature.com/articles/sdata201635