Predicting Death and Other Complications in Myocardial Infarction Patients

Mingxi He
Philippe Nguyen

Contents

1 Introduction and Motivation

În 2008, Cardiovascular diseases (CVD) accounted for 29% of all deaths in Canada, with over half being caused by coronary artery disease or myocardial infarction. [Canada, 2011]. In the province of Quebec, the prevalence of CVD in the elderly population is 19.4%. [Huynh et al., 2006]. A massive amount of research is being poured into figuring out effective ways to treat this condition that affects 1 in 5 of the senior population in la belle province.

One recent study involving 24 different hospitals investigated using multi-faceted knowledge translation interventions to increase the prescription of evidence-based medical therapy. We have acquired the data used by this study and will apply some machine learning techniques to classify patients into two classes after admittance into the hospital for acute coronary syndrome (ACS). Though the study was designed in order to test the impact of evidence-medical therapy on the long term outcome of patients after they've left the hospital, the data can also be used to predict short term mortality (death or complications in hospital) since these events were also recorded. This is an important task since physicians want to separate low-risk and high-risk patients in order to provide the best treatment for both groups. We will use machine learning methods in order to classify patients.

Each example in the dataset corresponds to a patient who had a diagnosis of STEMI (ST segment elevated myocardial infarction, a type of heart attack), NSTEMI (non-ST segment elevated myocardial infarction) or unstable angina. These are the three conditions that cause ACS. At the time of hospitalization, trained clinicians used electronic case report forms in order to input patient data. Over a hundred variables were recorded, such as age, sex, weight, cholesterol levels, whether the patient smoked and whether they had previous heart attacks or strokes. If a complication, such as stroke or cardiac arrest, occurred during the hospital stay, then the event was recorded. Since the main goal of the study was to test the long term outcome of treatments given for ACS patients, if the hospital stay was less than 48 hours, then the patient was not included into the dataset; this restriction should not affect the classification of short term complications. Though the majority of analysis is done for predicting death, the dataset also contains other outcomes such as stroke, cardiogenic shock, and other complications. Instead of only predicting death, we also train a classifier to predict the existence of any complication in a patient.

2 Approach

We used standard machine learning techniques that will allow us to report meaningful results, namely k-fold cross validation, with a training and testing set.

In order to deal with missing values, we assumed a naive bayes model where each feature only depends on its class (i.e. ”death or ”no death”/ ”complication” or ”no complication”). Standard

MLE estimates of the missing values in the training set prompted us to fill in these values with the class sample mean (since none of the class labels are ever missing, only the feature values). Missing values in the cross-validation/testing set received a different treatment. For these, we filled in the overall sample mean of the variable. This approach is justified by reasoning that if our classifier were to classify a newly admitted patient with missing values, it would not know their class, and therefore be unable to assign a class sample mean.

Next, there is a significant inconvenience in our data caused by the class imbalance. The low number of patients that have died (about 4.5%) implies that a trivial classifier that assigns the majority class (e.g. "no death") to every patient will have an accuracy of 95.5%. This is a highly undesirable property since a good classifier will be barely differentiable from a trivial classifier. There are several methods we implemented to combat the class imbalance. In addition to reporting the accuracy

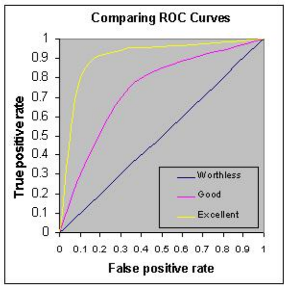
$$Accuracy = \frac{TP + TN}{TP + FP + FN + TP}$$

we will also report the F1 score [Van Rijsbergen, 1979]

$$F_1 = \frac{2 * TP}{2 * TP + FN + FP}$$

and "Area Under ROC Curve" [Hanley and McNeil, 1982]. Note that if the minority class is treated as positive, then the F1 score of the trivial classifier is 0. A classifier which chooses classes at chance, will have an ROC score of 0.5. For all measures a perfect score is a score of 1. The ROC score/Area under ROC curve is a standard method of comparing models which is widely used in medecine. It can be thought of as showing how well a classifier can discriminate between two distributions; in our case, it is the patients who died or had a complication and those who didn't. The ROC score corresponds to the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance; the higher ranking would correspond to a heightened risk of death or complication[Fawcett, 2006].

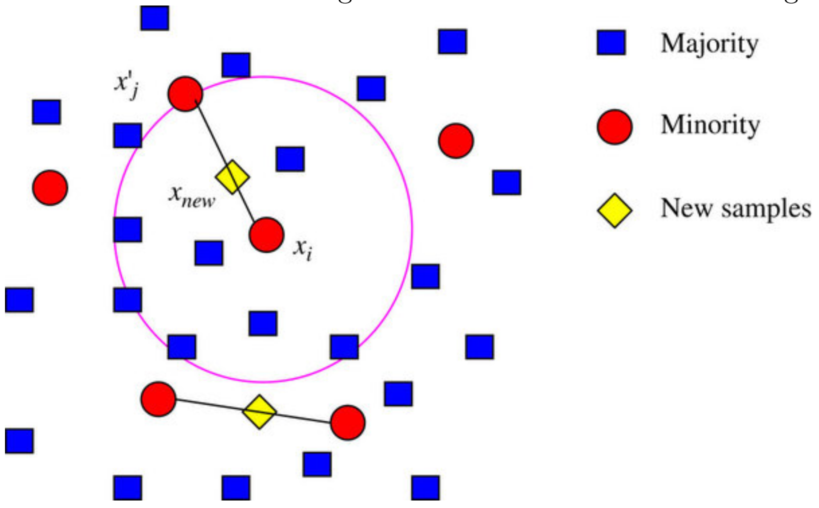
Figure 1: An example of different Area Under ROC Curves



Furthermore, we could implement over-sampling the minority class or under-sampling the majority class to discourage our classifier from assigning the majority class to all instances. The under-sampling tends to perform better than the over sampling [Chawla et al., 2002], but throwing away valuable data tends to be undesirable. In their 2002 paper [Chawla et al., 2002], Chawla et al. proposed the Synthetic Minority Over-sampling Technique (SMOTE) in order to combat the class imbalance problem. The detailed algorithm from their paper is attached below in figure 6 the appendix.

Essentially, the algorithm selects an instance from the minority class, selects its k-closest neighbors, interpolates a new point, and adds that point as an instance to the training set. k is chosen with the goal of creating a desired amount of new training instances. The number of new minority class instances to generate was a hyperparameter which could be selected using cross validation.

Figure 2: An illustration on the general idea behind the SMOTE algorithm.



Finally, we had to choose a classifier. We begun with a support vector machine - a maximum margin classifier which was introduced in class. Then we used a few other classifiers such as logistic regression. Fortunately, these were already implemented in python in the scikit-learn package. It was found that the other classifiers performed worse than the SVM in almost all cases, so the SVM was used for all of the analysis results.

3 Experimental Setup

As stated earlier, the data consists of patients who were admitted for acute coronary syndromes, which can be caused by STEMI, NSTEMI or unstable angina. The data was input manually by trained professionals. Some variables were numerical, some were strings, because of this, missing data was recorded in different ways. In order to use the sci-kit learn functions, the data needed to first be processed.

After acquiring the data, we processed the data in the following order:

1. Drop irrelevant variables. A comma separated list of variables from the data set that were kept can be found in the appendix.
2. Replace "YES/NO" variables with "1/0" variables.
3. Replace "diseasedarteries" and "typeoftrop" with binary variables.
4. Change the one factor variable (race) into binary indicator variables.
5. Recode missing data values into -1.
6. Manually replaced some instances of variables that were strings with -1.

For convenience, the R code that accomplishes this has been uploaded along with this report.

Once the data was processed, it can be separated into a matrix containing each of the variables (each column is a feature) and a matrix containing the complications. The complications matrix has columns corresponding to each possible complication such as stroke, shock or death. If we want to predict death, only the column corresponding to death is selected as the dependent variable. If we want to predict the occurrence of any complication, the response vector is selected by using the logical OR function over the columns (so if any complication occurs, the response value for that patient is 1, and 0 otherwise).

Now, the data is separated into a training and a test set. 1/5th of the dataset were used as test examples. They were randomly chosen, however, the ratio between the two classes was set to be the roughly the same between the training set and test set. A seed was used in separating the two sets so that test set was the same everytime. For every trained classifier, the test set was only used once to report a final result (all results shown in the results section are from predictions on this test set).

Every time we wish to train a classifier, we select the type of classifier, and the hyperparameter values that we want to use. We generate all combinations of the hyperparameter values. We use stratified 5-fold cross validation to generate 5 cross validation sets and 5 corresponding training sets.

With the full training set we find the best hyperparameter values:

- For every combination of hyperparameter values:
  - Apply stratified 5-fold cross validation. For every fold:
    - For the training set:
      - Fill in the missing values according to the class mean.
      - Apply SMOTE and/or undersampling.
      - (Normalization) Calculate the mean and variance of the training set. Subtract the mean and divide by the standard deviation to obtain zero mean and unit variance for each feature.

            - Train the classifier on the training set.

        - For the cross validation set:

            - Fill in missing values by taking using the overall mean of the training set.
            - (Normalization) For each feature subtract the mean and divide by the standard deviation calculated in the training set.
            - Predict the classes of the cross validation set. Report the accuracy measures for this fold.

    - Report the average accuracy, F score, and ROC score for this combination of hyperparameter values.

Once the accuracy measures for every combination of hyperparameter values has been collected, choose the hyperparameter combination which maximizes the wanted accuracy measure (accuracy, F score or ROC score).

For the best set of hyperparameter values, we predict on the held out test set:

- For the entire training set:
    - Fill in the missing values according to the class mean.
    - Apply SMOTE and/or undersampling.
    - (Normalization) Calculate the mean and variance of the training set. Subtract the mean and divide by the standard deviation to obtain zero mean and unit variance for each feature.
    - Train the classifier on the entire training set.
- On the test set:
    - Fill in missing values by taking using the overall mean of the training set.
    - (Normalization) For each feature subtract the mean and divide by the standard deviation calculated in the training set.
    - Predict the classes of the test set. Report the accuracy measures.

As a note, in a naive Bayes model, when we fill in the missing values for the cross-validation/testing sets, it would be more principled to use un-SMOTEd training set means since probability of each class is changed by SMOTE, thus unlike the true distribution. However, in practice, imputation using the SMOTEd(or undersampled) training set performs much better, likely because the classifier was fit on this set.

Though a known seed to ”randomly” divide the total data into the training set and test set, when the cross validation folds are generated there is no given seed. This leads to different cross validation folds on every iteration or run of the program. Thus the returned scores are slightly different on every run, but only by a small amount. While the test set always remained the same, the number of cross validation folds could also vary, however, 5-fold cross validation was used since changing the number of folds had little impact.

The use of principle component analysis (PCA) to reduce the dimensionality of the feature matrix (after normalization was applied) was used. However, the PCA did not significantly change the results of the SVM, so it was not applied for the values given in the results section.

Also, since ROC is a standard measure used for methods of patient stratification, for the purposes of the report we always choose the hyperparameter set which achieves the highest ROC score (on the cross validation sets) to run on the test set.

4 Results

We trained a support vector machine in order to predict short term (in hospital) mortality or complication. The results are shown in table 1. It is clear that the support vector machine is better at predicting death than it is at predicting complications. Though the set of complications consist of several major cardiac and cerebrovascular events, it would seem that the examples of death are more separable from non-death examples than the complications and non-complication examples are.

Table 1: SVM Predictions For Death an Complications

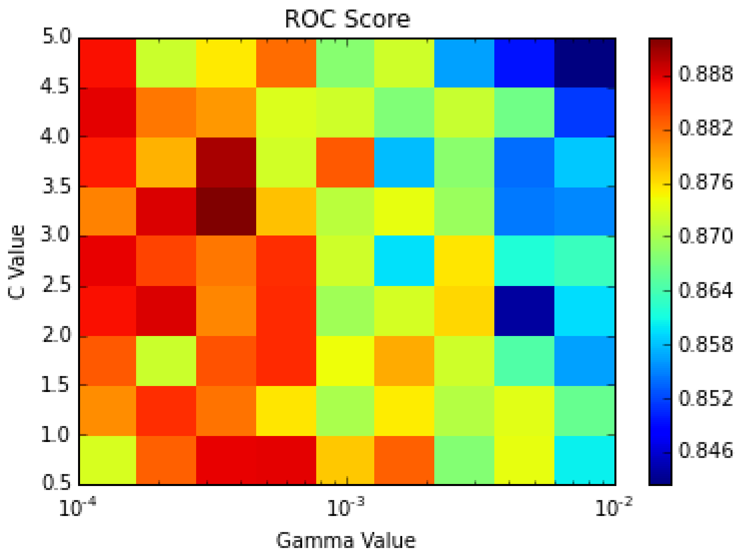
	Death	Complication
Accuracy	0.88	0.68
F-Score	0.44	0.59
ROC-Score	0.91	0.71

It became quickly clear as we were training classifiers that one which maximized the ROC measure would usually not do so well for the other measures (accuracy and F score). However, this is not a large concern since it is expected that the different scores represent different characteristics of the classifier. We choose to mainly use the ROC score since other classifiers in the field report their results with ROC score, which allows for easier comparison.

There were several parameters which could be selected when generating creating the SVM; such as the type of kernel (and associated scale factor) and the C penalty parameter. Furthermore, other hyperparameters which could be varied were the number of samples generated by SMOTE and the number of majority class examples to drop when undersampling.

In order to select the best hyperparameters, many combinations of hyperparameters were tried on the training/cross-validation set. The accuracy, F-score and ROC-score were averaged over the 5 cross validation folds in order to generate accuracy measures on the cross validation set. Only the hyperparameter set which provided the highest ROC score was chosen to be tried on the test set in order to report accuracy of the classifier. Figure 3 shows an example color plot of the hyperparameter sets and their relative averaged ROC scores on the cross validation set. The values of the penalty factor C of the SVM were varied linearly from 0.5 to 5, while the scale factor γ was varied logarithmically from 10^{-4} to 10^{-2}. As can be seen in figure 3, the parameter set with the highest cross validation score is [C = 3.0 and $\gamma = 0.000278$]. This parameter set had a ROC score of 0.892 averaged across the cross validation sets. The ROC score on the test set was 0.91.

Figure 3: ROC scores of the SVM Classifier with Different Parameter Settings



Firstly we assessed the methods used in order to rectify the problem of class imbalance. Table 2 shows the accuracy measures of the SVM using different methods. The "Classes Imbalanced" case left the training examples untouched. The "Undersampling only" case only removes examples of the majority class. The "SMOTE only" case uses the SMOTE algorithm to generate new samples for the minority class. The "both" case uses both SMOTE and undersampling. The number of examples generated by SMOTE and number of majority classes randomly removed by undersampling was always chosen by the highest average ROC score on the cross validation set. As can be seen in table 2, leaving the minority and majority classes imbalanced leads to a poor ROC score. It seems that using only undersampling, only SMOTE, and using both will all give similar ROC scores. SMOTE seems to do slightly worse than undersampling. Using both does slightly better than using either separately, however this requires to train the SVM with every single combination of the two varying parameters (The number of SMOTE examples generated and the number of majority class examples dropped by undersampling).

Table 2: Accuracy for Different Methods of Handling Class Imbalance

	Classes Imbalanced	Undersampling only	SMOTE only	Both
Accuracy	0.96	0.90	0.94	0.88
F-Score	0.54	0.45	0.56	0.44
ROC-Score	0.71	0.89	0.88	0.91

Table 3 shows the accuracy, F-score and ROC-score of different SVM kernels. For polynomial kernels, degrees up to d=4 are shown (higher degree polynomial kernels performed worse). It can be seen that linear SVMs and a 2nd or 4th degree polynomial kernel performs relatively poorly. The radial basis function kernel performs fairly well, and the degree = 3 polynomial also had a high ROC score. From this analysis it may seem that the polynomial kernel with degree 3 should be considered; however, it was found that the radial basis function kernel was more consistent as other factors were changed (such as using a restricted dataset, see table 4). So the radial basis function was used as the SVM kernel for the remainder of the analysis.

Table 3: Accuracy for Different SVM Kernels

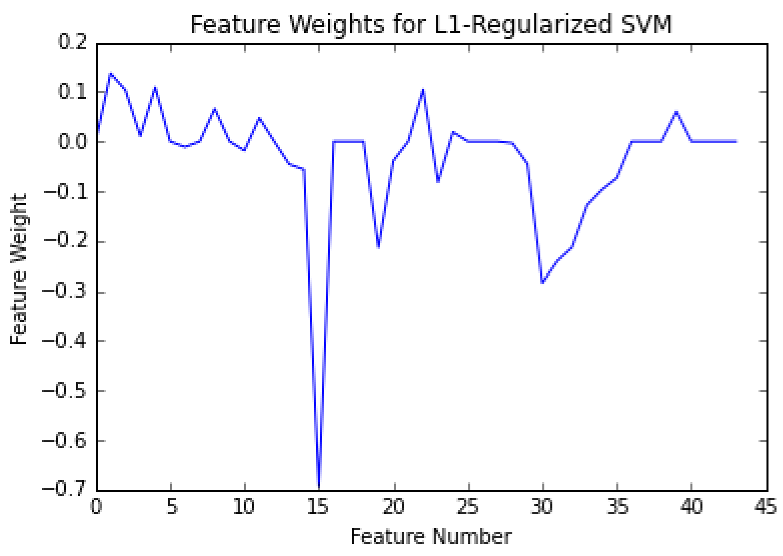
	Radial Basis Function	Linear	Poly (d = 2)	Poly (d = 3)	Poly (d = 4)
Accuracy	0.88	0.82	0.89	0.87	0.70
F-Score	0.44	0.32	0.38	0.41	0.21
ROC-Score	0.91	0.87	0.82	0.90	0.80

If we use a linear SVM, we can use L1 regularization to get important predictors of death. L1 regularization will enforce sparse weights, such that the unimportant features have near 0 weights. In figure 4, the weights for each of the L1-Regularized features is shown. Many of the weights are set to 0 or small numbers; however there are significantly non-zero. The greatest peak (feature number 15) is the level of baseline LDL(low density lipoprotein cholesterol). Another important feature whether the patient has had prior revascularization (treatment for ischemia, damaged blood vessels). These are variables that are known to lead to greater chance of death in patients with heart failure[Charach and Rubinstein, 2010][Eagle, 1997].

It should be noted that slightly changing the C and gamma parameters (which would still provide a SVM with a fairly high ROC score) would significantly change the weights for the features. For example, sometimes the weight for baseline LDL was small.

A comparison of the performance of our classifier is shown in table 4. The SVM we trained using the dataset is shown under the SVM (all features) column. Another predictive measure currently used is the GRACE score which uses predictive factors such as systolic blood pressure, creatinine levels, the presence of elevated or abnormal cardiac enzymes and other factors. The GRACE score reports a ROC score of 0.82 for predicting short term mortality [Huynh, 2013]. Our classifier performs better with a score of 0.91.

Figure 4: L1-Regularized Weight coefficients



Though our SVM classifier performs well in predicting patient outcome; as a clinical tool, it is useless since clinicians will not be able to immediately retrieve all of the same variables from the patient at the time of hospital admittance. In order for our classifier to have a clinical impact, we want to be able predict mortality before death occurs, so several of the variables in the dataset cannot be used since they cannot be retrieved in quickly in a clinical setting. The C-ACS Risk score uses only variables that can be obtained before hospital admittance or in the emergency room. These variables include age, sex, prior health issues, heart rate and systolic blood pressure [Huynh, 2013]. The C-ACS Risk score has a ROC score of 0.76 when predicting short term mortality. In order to compare the support vector machine classifier to the C-ACS score, we restricted the independent variables to only the ones used by the C-ACS. Furthermore, our dataset did not have any record of the patient's systolic blood pressure or the heart rate, so the SVM was trained without them. The SVM performed as well as the C-ACS score, with a ROC-Score of 0.76.

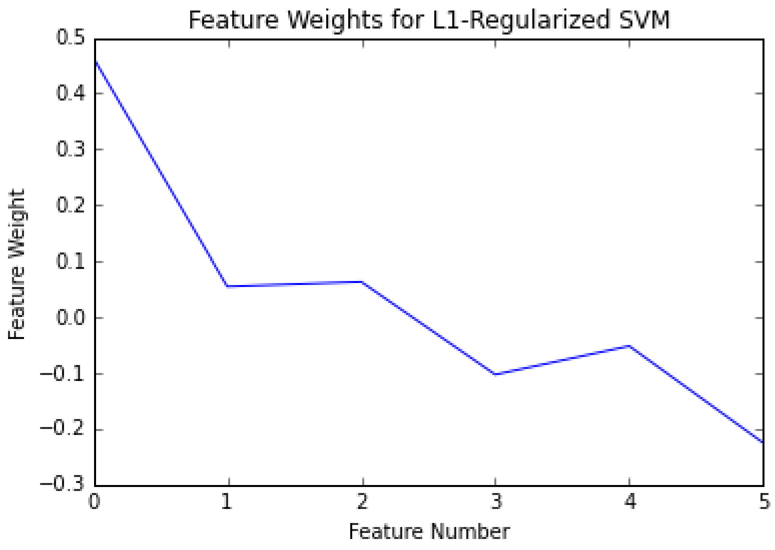
Table 4: Accuracy of Different Classification Methods

	SVM (all features)	GRACE	SVM (prehospital features)	C-ACS
ROC-Score	0.91	0.82	0.76	0.76

Similarly to the analysis done in figure 4, L1-regularization can be used to generate sparse weights for the prehospital/ER variables. The results are shown in figure 5.

In this case the L1-regularized SVM is just as good as the RBF SVM, and has a ROC score of 0.76. The two significantly non-zero features are feature numbers 0 and 5, which correspond to the patients age and whether they had prior stroke or transient ischemic attack (”mini stroke”). Unlike

Figure 5: L1-Regularized Weights for SVM Trained with Prehospital/ER variables



the SVMs trained on the full dataset, the relative weight coefficients did not significantly change for different hyperparameter settings.

5 Conclusion and Future Work

From our results we can see that the SVM classifier performs very well in predicting mortality. When the full dataset is used, the SVM does much better than the GRACE score in separating patients who died and those who didn't. When only "prehospital and emergency room accessible" variables were used the SVM performed just as well as the C-ACS Risk score, even though the SVM had to use fewer variables (since our dataset did not have access to some of these variables). Thus if systolic blood pressure and heart rate were included for the SVM to use like in the C-ACS score, then the SVM may perform better. However, despite performing just as well (or possibly performing better with more complete data), there are a few reasons as to why it would be more useful to use the C-ACS score in a clinical setting. Mainly, the SVM predictions are unintuitive, it is difficult to understand how each of the factors influence the SVM prediction.

The L1-regularized weights do give some insight on what variables are important, however this intuition is lost when considering the whole SVM, especially when using a kernel. For example, the GRACE score is fairly easy to compute by physicians; a value is assigned for each of the variables, and the values are summed. If the sum is below 100, then the patient is low risk, in hospital death risk is less than 1%; if the sum is above 171, then the patient is high risk and the chance of death is above 9 %. Similarly, the C-ACS score is even simpler. The C-ACS simply ranges from 0-4, with one point for each of the 4 variables that passes a certain threshold. It could be argued that the SVM is just as easy to use since physicians only need to plug the variable values into a computer, however, there is no doubt that the intuition is lost since the computer is doing all of the computation. Furthermore, using the SVM depends entirely on access to a computer.

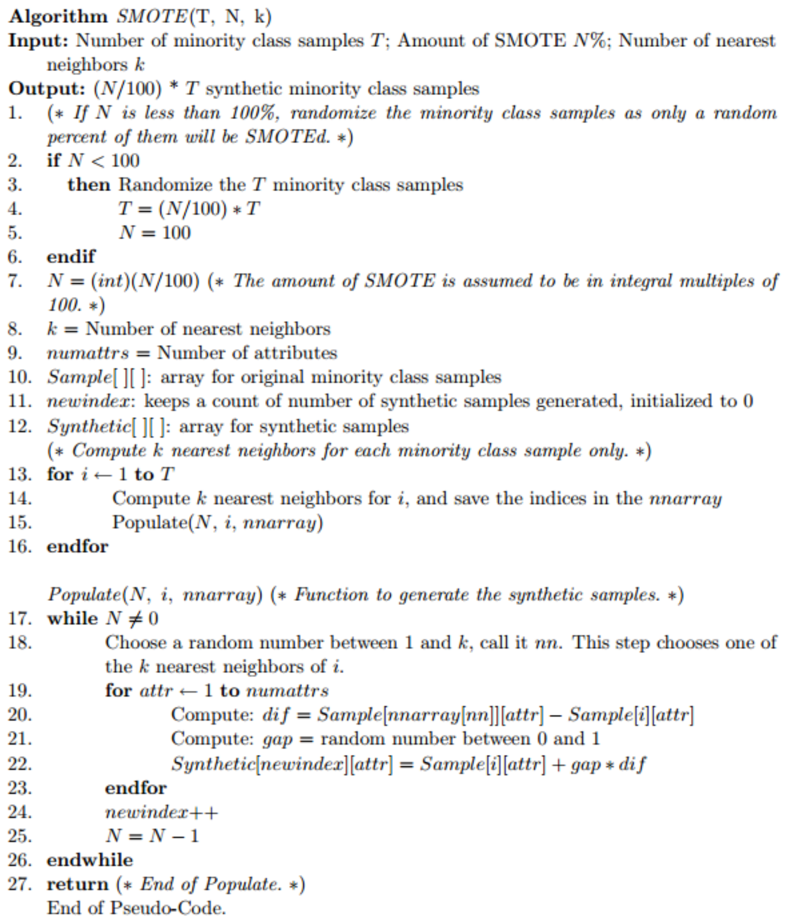
When it comes to filling in missing variables, we assumed a naive bayes approach where each variable is conditional only on the class. However, this is unrealistic. More advanced methods can be used to fill in missing values such as expectation maximization, which will require a more complex model of the probabilities and a better understanding of the dependencies between each of the feature. It is also possible to fill in missing values by weighted averaging of the example's nearest neighbors, this is called k-nn imputation.

Some criticisms made to the F1 score and the AUC score are that they do not take into account true negatives and false negatives. There have been other measures proposed, such as the Phi coefficient, Matthews correlation coefficient, Cohen's kappa [Powers, 2011], that may be better at capturing the "goodness" of a classifier. However, we choose to use the ROC score since it is a standard score to report across the field.

The SVM performs less well in predicting more general complications. There is still room for work on building a classifier which better predicts complications. This will likely require a better model for the occurences of complications, and possibly more data. The strong results obtained by using the SVM to stratisfy patients who died shows that machine learning methods may indeed be useful in predicting any complication. Hopefully this will be useful for physicians in the emergency rooms and help prevent future deaths and complications.

6 Appendix

Figure 6: The SMOTE algorithm from the paper by [Chawla et al., 2002]



```
Algorithm SMOTE(T, N, k)
Input: Number of minority class samples $T$; Amount of SMOTE $N\%$; Number of nearest
    neighbors $k$
Output: $(N/100) * T$ synthetic minority class samples
1.  (* If N is less than 100%, randomize the minority class samples as only a random
    percent of them will be SMOTEd. *)
2.  if $N < 100$
3.     then Randomize the $T$ minority class samples
4.          $T = (N/100) * T$
5.          $N = 100$
6.  endif
7.  $N = (int)(N/100)$ (* The amount of SMOTE is assumed to be in integral multiples of
    100. *)
8.  $k$ = Number of nearest neighbors
9.  $numattrs$ = Number of attributes
10. $Sample[\,][\,]$: array for original minority class samples
11. $newindex$: keeps a count of number of synthetic samples generated, initialized to 0
12. $Synthetic[\,][\,]$: array for synthetic samples
    (* Compute k nearest neighbors for each minority class sample only. *)
13. for $i \leftarrow 1$ to $T$
14.         Compute $k$ nearest neighbors for $i$, and save the indices in the $nnarray$
15.         Populate($N$, $i$, $nnarray$)
16. endfor

    Populate($N$, $i$, $nnarray$) (* Function to generate the synthetic samples. *)
17. while $N \neq 0$
18.        Choose a random number between 1 and $k$, call it $nn$. This step chooses one of
           the $k$ nearest neighbors of $i$.
19.        for $attr \leftarrow 1$ to $numattrs$
20.                Compute: $dif = Sample[nnarray[nn]][attr] - Sample[i][attr]$
21.                Compute: $gap$ = random number between 0 and 1
22.                $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
23.        endfor
24.        $newindex$++
25.        $N = N - 1$
26. endwhile
27. return (* End of Populate. *)
    End of Pseudo-Code.
```

Variables Kept: "hypothyroid", "age", "stemi", "mi", "femal", "weight", "race", "activesmoker", "asa", "antiplatelet", "antiplateletintravenous", "baselinecreat", "peakcreatinine", "baselinehemoglobin", "nadirhemoglobin", "baselineplatelets", "baselineldl", "baselinehdl", "typeoftrop", "troppeak", "ejectionfraction", "cath", "diseasedarteries", "angioplasty", "number.ofstents", "INHOSPCABG", "shock", "stroke", "mechanicalventilation", "chf", "cardiogenicshock", "ventriculararrythmia", "atrialfibrillation", "bradyarrythmia", "arrrythmia", "cardiacarrest", "timibleed", "gibleed", "infection", "death", "diabetes", "hypertension", "priorpci", "priorrevasc", "priorcabg", "priorcvatia", "antiplateletondischarge", "bbprescribed", "statinsprescribed", "aceinhibitorprescribed", "arbprescribed", "cablocker"

NA Strings: ""," ","NA", "UNK", "UNKNOWN", "ND", "#NULL!", "UK"

Relevant Software

Python - machine learning techniques we used were already implemented in the scikit-learn package. The NYAN python library was used for SMOTE implementation: https://github.com/blacklab/nyan
R - for data cleaning

References

[Canada, 2011] Canada, S. (2011). Mortality, summary list of causes.

[Charach and Rubinstein, 2010] Charach, Gideon, J. G. A. R. O. R. D. W. D. S. I. G. M. W. G. K. and Rubinstein, A. (2010). Baseline low-density lipoprotein cholesterol levels and outcome in patients with heart failure. *The American journal of cardiology*, 105(1):100–104.

[Chawla et al., 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16(1):321–357.

[Eagle, 1997] Eagle, Kim A., C. S. R. M. C. M. D. R. H. E. D. F. B. J. G. (1997). Cardiac risk of noncardiac surgery influence of coronary disease and type of surgery in 3368 operations. *Circulation*, 96(6):1882–1887.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

[Hanley and McNeil, 1982] Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36.

[Huynh, 2013] Huynh, Thao, S. K. A. Y. N. D. J. O. L. E. S. R. Y. (2013). Canada acute coronary syndrome risk score: A new risk score for early prognostication in acute coronary syndromes. *American heart journal*, 166(1):58–63.

[Huynh et al., 2006] Huynh, T., O'Loughlin, J., Lawrence, J., Schampaert, E., Rinfret, S., Afilalo, M., Kouz, S., Cantin, B., Nguyen, M., and Eisenberg, M. J. (2006). Delays to reperfusion therapy in acute st-segment elevation myocardial infarction: results from the ami-quebec study. *Canadian Medical Association Journal*, 175(12):1527–1532.

[Powers, 2011] Powers, D. M. (2011). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

[Van Rijsbergen, 1979] Van Rijsbergen, C. J. (1979). *Information Retrieval (2nd ed.)*.