Methodology for Sparse Classification Learning: Arrhythmia

Lee Tanenbaum, Stanford University

Introduction

Cardiac arrhythmias represent changes in the human heart's normal rhythm, which maybe be either immediately fatal or, if sustained over a long duration, cause irreparable cardiac damage.

Therefore, I propose to study EKG data from the UCI Arrhythmia Dataset[1] with a machine learning approach to improve machine classification performance to assist doctors in their diagnosis.

This dataset contains 452 samples, each sample containing 279 features about patient physiology, divided into 'healthy', 11 non-empty arrhythmia classes, and one 'other' class distributed as below:

Class	1	2	3	4	5	6	7	8	9	10	11	12	13
samples	245	44	15	15	13	25	3	2	9	50	4	5	22

This problem poses four hurdles that learning models must account for, mainly that there are 13 classes, a heavily imbalanced class distribution, many features relative to number of training examples, and missing data points.

The methodology I use here is applicable to other problems with similar difficulties so my goal is to both improve classification accuracy to improve medical preventative care as well as to explore these challenges more thoroughly and help set precedence for future research.

Ensembles:

Ensemble voting method creates many decision processes, varying either the training parameters or model structure between individual learned constituents, and then takes the mode average of its constituent's votes to make a prediction. Ensembles have been shown to perform best results when there is significant diversity among constituent models. This is maximized when using a variety of strong learning algorithms rather than simplifying models to promote diversity due to lack of individual model accuracy. Therefore, for my classification ensemble I will attempt to build strong classifiers based on different models, run with feature sets derived from different feature selection algorithms that will accurately represent the data while maintaining variation in structure from each other so as to have varying sources of error. As such, creating an ensemble for arrhythmia classification can be reduced to an analysis of learning models and feature selection algorithms that can be adapted to handle multiple imbalanced classes with many features relative to number of examples.

Measurements:

I want to represent a measure of performance of my model. As opposed to accuracy which doesn't take into account imbalance in data, I am interested in the F-score of the minority classes as a way to represent performance. Due to the multiclass setting, I will therefore calculate the average of the F-score of the classes throughout the distribution to represent performance in a meaningful way.

I will also report overall accuracy for ease of readability.

All of the performance measurements are reported as 10-fold cross validation averages.

Design Process

Such datasets require a non-standard learning approach. The process I followed was as follows.

- Create models and measure training error to select multiclass models of sufficient complexity
- Explore feature selection to reduce overfitting
- Measure validation f-score and graph confusion matrixes to guide progress.
- Create an ensemble of models and feature sets.
- Review the performance of the ensemble was comparable to the individual classifiers.
- Exploring the ensemble voting procedure, I notice many misclassifications are unanimous.
- Given that cardiologists can recognize arrhythmias based on these features, I create more relevant features. Here, I analyzed the confusion matrix and focused on the greatest sources of misclassifications.
- To give an example of assisting the feature selection with medical research, consider:
    - To predict arrhythmia types by vector angle structures I calculated a feature to represent if $\angle QRS$ is between -30 and 90, if ∠T is near ∠QRS, and if ∠P isbetween 0 and 90, and to map all combinations to different values:

$$\boldsymbol{Feature} = \left(\mathbf{1} + \boldsymbol{e}^{-\left(\frac{\angle QRS - 30}{60}\right)^{4}}\right)\left(\mathbf{1/2} + \boldsymbol{e}^{-\left(\frac{\angle T - \angle QRS}{30}\right)^{4}}\right)\left(\mathbf{1} - \mathbf{2}\boldsymbol{e}^{-\left(\frac{\angle P - 45}{45}\right)^{4}}\right)$$

Feature selection:

First, I must address the problem of missing data points. I choose to impute the missing data points with the nearest neighbor of non-missing features. If its nearest neighbor is also missing that value, I instead replace the value with the mean value for the dataset.

I attempted to perform Principal Component Analysis on the data to extract a small set of features representative of a large percent of the variation in the dataset, however I did not see strong performance from this algorithm. This seems intuitive: the examples differing from the standard direction of variation are, by definition, those representatives of cardiac abnormalities.

For my solution, I performed backwards search on the minimum Redundancy Maximum Relevancy (mRMR) of the data [2]. This is a measure of the mutual information of a feature with the class label minus its average mutual information with other features.

The mRMR search algorithm is therefore to iteratively remove features based on

$$Estimated\ Worst\ Feature = \min_{S}\left[\frac{1}{|S|}\sum_{f_i \in S} I(f_i;c) - \frac{1}{|S|^2}\sum_{f_i,f_j \in S} I\left(f_i;f_j\right)\right]$$

The other feature selection algorithm I employ in the ensemble is the more random undersampling approach of the gini index of a Random Forest classifier. During RF classification, the gini index of a variable 'i' is the average difference between the impurity of the descendants of the tree split at nodes with variable 'i' minus the impurity of the nodes to be split with variable 'i'. For the case of 'm' total features, and with 'f_i' representing the fraction of items with the value 'i' in the set, this becomes

$$Impurity = 1 - \sum_{i=1}^{m} f_i^2$$

Notice that the impurity is therefore 0 (minimum) when all cases under the node fall into the same classification category.

I also performed forward search by adding one feature at a time that increased most the average f-score validation performance of a SVM. I believe SVMs to be a good choice due to high classification performance and the deterministic aspect of creating a SVM decision boundary, which none of the other models used here have.

Also note that I stop feature selection when there is a relatively large number, 40-80, features remaining, which is larger than the recommended maximum of 1/10th of the number of samples. This is to be expected, however, because of the existence of 13 classes, which requires more information encoded into the feature set than a small number of features would contain. I suggest that future research look into the relation between sample size and number of classes to distinguish to make predictive ranges for the optimal number of features for classification.

Models

Trees, Random Forest (RF)

A basic strategy for performing multi-class classification would be to create decision trees that branch based on feature values being greater than or equal to some threshold, and to predict class labels based on the percentage of training data in each leaf of a decision tree. This could be improved for imbalanced datasets by normalizing the probabilities of data, and further improved by boosting, by creating an ensemble of such trees and sampling more from classes with poor prediction values or bagging by randomly selecting samples from the training set to create ensembles of decision trees. Based on existence of many features relative to the number of examples, I chose an ensemble of decision trees with randomly selected features for decision nodes, which is called Random Forest (RF) [3].

One elegant fact about the Random Forest algorithm is the lack of parameters to adjust to fit the dataset. In particular, the main selections needed to be made are the number of trees, which has little significance as long as enough trees are selected to sample the population, and the number of parameters chosen, or depth, of each tree, which I have chosen to be $\sqrt{p}$ for a dataset with p features.

There are many ways to vary a Random Forest to handle extremely imbalanced datasets, from manipulating misclassification costs to sampling techniques. I chose to sample uniformly between classes to balance the class probabilities. Because I believe it is more important not to misclassify one of the few extreme arrhythmia class cases than misclassify a healthy patient, I wish to also apply a lower cutoff for the minimum number of votes required to end an ensemble search and select a class, and my cutoff deviation is proportional to the extremity of the minority class imbalance.

One-vs-Rest Naïve bayes (NB) with Gaussian smoothing, uniform prior probabilities, and imbalanced misclassification costs

NB calculates the most likely class for an example by treating conditional probabilities of features given class labels as independent and maximizing the likelihood of the example.

This model is designed initially for discrete random variables, and cannot represent in its initial form the continuous random variables in arrhythmia readings. Therefore, I treat the variables as Gaussian distributions to calculate the conditional probabilities. To explain this, for a variable x, let μ_c be the mean of the values in x associated with class c, and let σ_c be the variance of the values in x associated with class c. Therefore, the probability distribution of some value given a class p(x=v|c) can be computed by plugging v into the equation for as $N(\mu_c, \sigma_c^2)$:

$$p(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

To extrapolate this binary classification model to multi-class for this arrhythmia data, I use a one-vs-rest model where I compute binary classification between one class vs the rest of the classes. I compute this individually for every class, and then select the class with the greatest likelihood.

If there are K classes, and $f_k(x)$ is the likelihood of class k versus the set of all other classes, I chose class

$$\hat{y} = \underset{k\in 1\ldots K}{\operatorname{argmax}} f_k(x)$$

This allows us to apply binary classification models like the NB model to multiclass settings. Also, the NB algorithm needs to handle the imbalanced class distributions, which can be done either through creating an ensemble with majority classes under-sampled, or can run once with the minority classes oversampled to create balance. I chose to oversample the minority classes rather than create an ensemble of undersampled majority classes to help runtime of training and testing at no noticeable hit to performance to create balanced prior probabilities, as well as to restrict misclassifications of minority classes further with heavier misclassification costs for minority classes.

Therefore, the Naïve Bayes decision boundary between class 1 and 2 with misclassification costs C_i proportional to $\frac{1}{n_i}$ the number of examples of class I in the true distribution,

$$C_i = \frac{1}{n_i} \qquad C_2 \prod_{i=1}^{n} p(x_i|y = 1) = C_1 \prod_{i=1}^{n} p(x_i|y = 2)$$

Note that this does not take into account prior probabilities of the classes, because I sampled uniformly from all classes.

Note that NB classifiers are strongly dependent on direct correlations between features and class labels because they cannot model dependent relations between variables. Therefore, NB with a one-vs-rest modification was an initial model attempted and discarded due to not reaching even a high training-set accuracy. However, once I improved feature selection and had fewer, stronger individual features the NB performance improved dramatically.

One-vs-Rest Support Vector Machine(SVM) with regularization and a kernel threshold offset, and imbalanced misclassification costs

SVMs create decision boundaries between two classes. I can use this binary model, and again consider comparing each individual class against the group of all other classes. After training, I have decision boundaries based on the maximally confident one-vs-rest decision classifier in that space.

Kernel Threshold Offset and Misclassification Costs

This model suffers a similar problem to the NB in that a one-vs-rest SVM will strongly favor majority classes over minority classes. Solving this problem by creating an ensemble of SVMs with under-sampled majority classes would be too time-intensive, and oversampling minority classes has little effect on SVM models. Instead, I handle imbalanced class distributions by modeling imbalanced misclassification costs and implementing a kernel threshold offset.

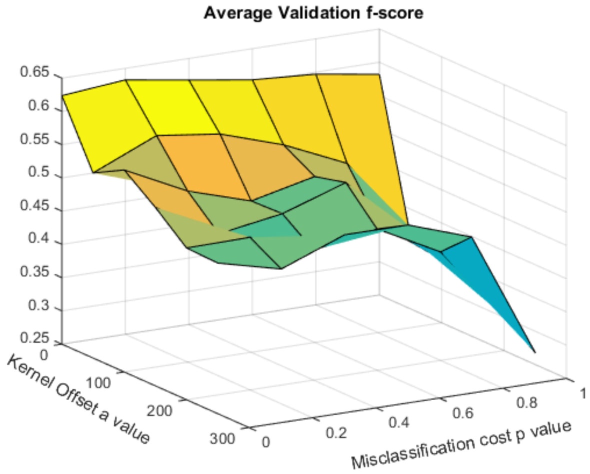
I offset the SVM kernel decision boundary in the direction of the under-represented minority classes based on the magnitude of the imbalance.

The kernel threshold offset is calculated as such: Let f(x) be the decision function for an example x. A new sample x will be assigned to the positive class if $f(x) \geq 0$ and the negative class if $f(x) < 0$, and let us assume without loss of generality that minority classes are the positive side of the boundary and majority classes the negative. Chen et al. [4] and Lin[5] showed that adjusting the decision threshold can increase minority class prediction accuracy. Therefore, we adjust the decision threshold to be offset by Θ, and replace the misclassification costs with a weighted misclassification cost, where n_+ and n_- represent the sizes of the majority and minority classes, $Cost_+$ and $Cost_-$ are misclassification costs inversely proportional to n_+ and n_-, and 'a' is a constant to modify global magnitude of these adjustments for all classes:

$$\theta = -1 + 2\frac{n_+ - n_-}{n_+ + n_- + 2a} \qquad Cost\sum_{i=1}^{n} \xi_i \rightarrow Cost_+\sum_{i\in I_+} \xi_i + Cost_-\sum_{i\in I_-} \xi_i \qquad \text{Where } Cost_i = \frac{1}{n_i^p}$$

and selecting the values of 'p' and 'a' that yield the greatest average f-score validation performance.

Varying these two parameters can be viewed by the following graph and explained by the following table:



Notice the following general regions that I expect, because both smaller 'a' values and larger 'p' values both increase minority class prediction likelihood.

	P cost exponent large	P cost exponent small
Low 'a' offset value	Always predict minority classes	Predict classes evenly and more accurately
High 'a' offset value	Predict classes evenly but inaccurately	Always predict majority class

This demonstrates that the SVM with threshold offset actually performs best with its costs approximately proportional to $\frac{1}{{n_i}^{1/8}}$, and with its threshold offset value a=0.

SVM regulation parameter C

I have found that the SVM gets better validation performance by decreasing the regularization parameter C to increase the regularization to inhibit overfitting. To add the regularization term C, the SVM hypothesis becomes:

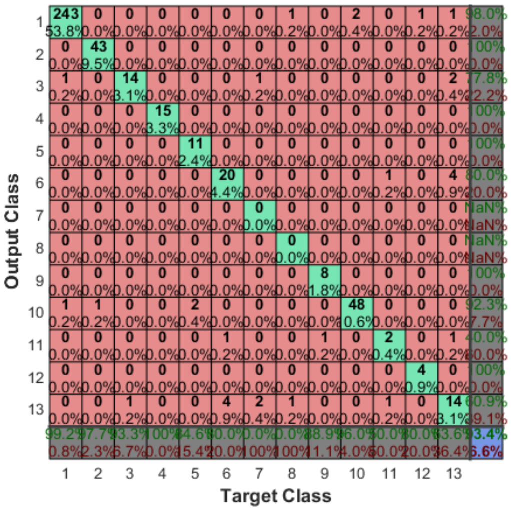
$$\min_{\theta} C\left(\sum_{i=1}^{m} y^i cost_1(\theta^T x^i) + y^i cost_0(\theta^T x^i)\right) + \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$$

Neural network with regularization(removed from ensemble)

Neural networks have been studied extensively on this dataset, and I confirmed their high performance. However, publications each select many variables for their NN, such as forward transfer function, back propagation method, structure, and many other parameters, and it is little understood why different structures would perform better or worse on this dataset.

Therefore, I remove NN from the ensemble as more research is needed to optimize and understand relevant tradeoffs in their application.

Results



Ablative analysis

Removed	Accuracy	F-score
Nothing	93%	.72
Feature Extraction	86%	.63
Feature Ensemble	86%/85%/83%	.60/.61/.54
Feature Selection	.66	.26
Model Ensemble & Parameter tuning	.54/.54/.69 SVM/NB/RF	.05/.05/.32 SVM/NB/RF

Note that a majority of the improvements came from feature manipulation and adapting models, and therefore I spent a majority of my time exploring this area of research due to the large number of features.

Individual model-feature set pair accuracies and f-scores

Accuracy/f-score	Gini	mRMR	SVM
SVM	81%/.60	80%/.56	83%/.62
NB	80%/.71	77%/.70	N/A
RF	84%/.62	83%/.61	80%/.65
Ensemble	93%/.72		

Note that NB with SVM feature search was not included in the ensemble. This combination performed poorly, presumably because SVM feature search detects dependent correlations to class labels which NB cannot represent.

Conclusions and Future Work

For classifying datasets that learning models were not designed to fit, one must:

- Adapt models to fit data structure
    - For ideas on how to adapt a model, one should either work through the derivation of the model and consider all assumptions and altering significant calculations, or look up model transformations in previous publications and consider applying variations to their project.
- Take meaningful measures of performance
- Study the many layers of the classification process to diagnose error causes
- Understand expectations for individual module performance

Future work should focus on

- Understanding model adaptations to imbalanced data, both for effectiveness at classifying this dataset and to extrapolate to general recommendations.
- Research could expand the feature set further using polynomial regression or performing other feature selection techniques such as lasso or ridge regression on the expanded feature set.
- Pay close attention to increasing classification accuracy for the ‘other’ class, the greatest current source of error

References:

1. Dataset, ECG Arrhythmia. "UCI Repository of Machine Learning Databases."*Available from:< ftp. ics. uci. edu/pub/machine-learningdatabases/>(accessed May 2009)* (1998).
2. Ding, Chris, and Hanchuan Peng. "Minimum redundancy feature selection from microarray gene expression data." Journal of bioinformatics and computational biology 3.02 (2005): 185-205.
3. Breiman, Leo. "Random forests." Machine learning 45.1 (2001): 5-32.
4. Chen JJ, Tsai CA, Moon H, et al. “Decision threshold adjustment in class prediction.” SAR QSAR Environ Res 2006;17:337-52.
5. Lin, Wei-Jiun, and James J. Chen. "Class-imbalanced classifiers for high-dimensional data." Briefings in bioinformatics (2012): bbs006.