Predicting Heart Attacks

Luyang Chen, Qi Cao, Sihua Li, Xiao Ju

Stanford University

lych, qcao, sihua, xju@stanford.edu

Abstract

This paper aims at a better understanding and application of machine learning in medical domain. In this paper, we modify three classical models for multiclass problems: Logistic Regression, Naive Bayes and SVM, and then implement them to predict cardiac arrhythmia based on patients' medical records. First, we use all features provided to build the models. By comparing the accuracy of different models, we find that Multiclass Naive Bayes and SVM models have better performance. Afterwards, we implement feature selection to improve the accuracy of prediction. Forward search procedure is used to choose different amount of features for each model. By comparing the accuracy between using all features and using features selected, we find that feature selection can significantly enhance the performance of our models.

I. Introduction

In the medical industry, machine learning algorithms can be used to diagnose some serious diseases. Among all diseases, cardiac arrhythmia is the top one cause of death in the world, claiming more lives than cancer and HIV combined. Thus, how to predict cardiac arrhythmia in real life is of great significance, both to research and application.

In this paper, we apply machine learning algorithms to predict cardiac arrhythmia based on a patient's medical record. We use the UCI Arrhythmia Data Set for both training and testing. We are provided with 452 clinical records of patients. Each record contains 279 attributes, such as age, sex, weight and information collected from ECG signals. The diagnosis of cardiac arrhythmia is divided into 16 classes. Class 1 refers to normal case. Class 2 to 15 represent different kinds of cardiac arrhythmia, such as Ischemic Changes, Old Anterior Myocardial Infarction, Supraventricular Premature Contraction, Right Bundle Branch Block and etc. Class 16 refers to the rest.

Our objective is to classify a patient into one class according to his or her clinical measurements. This paper applies Logistic Regression, Naive Bayes, and SVM algorithms to this realistic problem, compares their accuracy, and modifies the models to get the best possible results.

II. Features and Preprocessing

I. Extract valid columns

The dataset contains 279 columns representing 279 features, but some data in 5 columns are invalid or missing. So these 5 columns are completely ignored. We extract the rest 274 columns to form a new examples matrix for modeling.

II. Data discretization

After observing the raw data provided, we notice that some features have discrete values, such as age, sex, etc. while the other features have continuous, real values, such as the features extracted from ECG signals (e.g. amplitude of waves of each channel). Since we plan to use Naive Bayes algorithm, in which the input examples are discrete-valued, we have to turn all the raw data into discrete values before computing the probabilities. For the convenience of computation, we divide the values of each feature into 10 intervals. For the features that can only take two possible values 0 or 1, we consider 0 lies in the first interval and 1

lies in the 10^{th} interval. Then we convert each value into the order number of the interval it lies in. For example, if the height of one patient is 172cm and it lies in the 8th interval, which is $[164.5, 173.0)$, we assign 8 to the height of the patient. Thus, all features take values from 1 to 10.

III. Models

I. Logistic Regression and SVM

Since the supervised learning algorithms we have learnt can only classify examples into 2 classes, our first idea is to check whether an example lies in class 1 to 16 one by one until we find the class it lies in and then stop. We try both Logistic Regression and SVM.

First, we relabel class 1 to be 1 and class 2 to 16 to be -1. Then we can use Logistic Regression or SVM to get a decision boundary. If this decision boundary tells that a test example belongs to 1, we predict it belongs to class 1 and then stop. Otherwise, we relabel class 2 to be 1 and the other classes to be -1. We can get a new decision boundary, which tells whether the test example belongs to class 2 or not. We continue doing this until we find a class for this test example. If we cannot find a class for it after the first 15 trials, we predict it to be in class 16.

These algorithms can work, but there are two flaws. First, why should we start from class 1? If we start from a different class, we might get a different prediction. Second, why should we stop immediately after we get the first prediction. We can continue, pretending we haven't got a prediction. It is possible that two different decision boundaries tell an example belongs to class i and class j respectively. Which one should we believe? Due to these two flaws, we want to modify SVM later so that it becomes more reasonable and reliable when used for multiclass classifications.

II. Multiclass Naive Bayes

After data preprocessing, all the features can only take values from 1 to 10. Also, we would like to classify training set into 16 classes. To parameterize a multinomial over k outcomes, we can use k parameters $\phi_1, \cdots, \phi_k$ specifying the probability of each of the outcomes. We define ϕ_k and $\phi_{j,s|y=k}$ as follows:

$$\phi_k = P(y = k) \tag{1}$$

$$\phi_{j,s|y=k} = P(x_j = s|y = k) \tag{2}$$

$$\sum_{k=1}^{16} \phi_k = 1 \tag{3}$$

$$\sum_{s=1}^{10} \phi_{j,s|y=k} = 1 \tag{4}$$

We assume that all $x_j|y = k$ are independent. Then using formulas of conditional probability, we have the probability of the whole training set:

$$P(x, y) = \prod_{i=1}^{m} \Big(\prod_{j=1}^{n} \phi_{j,x_j^{(i)}|y=y^{(i)}} \Big) \phi_{y^{(i)}} \tag{5}$$

Maximizing $P(x, y)$ under the constraints (3) and (4), we get the following formulas:

$$\phi_k = \frac{\sum_{i=1}^{m} 1_{\{y^{(i)}=k\}}}{m} \tag{6}$$

$$\phi_{j,s|y=k} = \frac{\sum_{i=1}^{m} 1_{\{y^{(i)}=k\}} 1_{\{x_j^{(i)}=s\}}}{\sum_{i=1}^{m} 1_{\{y^{(i)}=k\}}} \tag{7}$$

And we can also use Laplace Smoothing to modify the formulas above.

To make predictions, we use Bayes formula:

$$P(y = k|x) = \frac{P(x|y = k)P(y = k)}{P(x)} \tag{8}$$

We predict y to be $\arg\max_k P(x|y = k)P(y = k)$.

III. Multiclass SVM

In the lecture, we talked about using SVM to classify our training examples into two classes. We want to modify the algorithm so that it can be used to make multiclass classifications.

First, we draw k hyperplanes in the feature space, so that the j^{th} hyperplane $\omega_j^T x + b_j = 0$

can separate training examples labelled j from those not labelled j. This can be easily achieved by using SVM algorithms. Each hyperplane tells whether a training example belongs to class j or not. However, chances are that the i^{th} hyperplane tells a training example belongs to class i while the j^{th} hyperplane tells it belongs to class j. We want to know which one is more reliable.

We define the geometric margin of a training example with respect to the j^{th} hyperplane as γ_j:

$$\gamma_j = \frac{|\omega_j^T x + b_j|}{\|\omega_j\|} \tag{9}$$

If γ_j is small, then $\omega_j^T x + b_j$ is likely to change its sign even if the j^{th} hyperplane changes a little bit. Therefore, if γ_j is larger, the conclusion the j^{th} hyperplane makes whether it belongs to class j is more reliable.

After we calculate all the geometric margins, we sort them from the largest to the smallest. We first look at the largest one j_1. If it tells this example belongs to class j_1, we trust it and assign $y = j_1$. If it tells this example doesn't belong to the class j_1, we also trust it and then we look at the next largest one j_2. If it tells this example belongs to the class j_2, we trust it and assign $y = j_2$. Otherwise, we look at the next largest one j_3, until we find some j and it tells this example belongs to class j. If none of all the hyperplanes tell this example belongs to some class, we predict it should be in the class with the smallest geometric margin.

Please look at the following simple example. If we consider 3 classes with 2D feature space, it is what the algorithm above attains. First, we use SVM to separate blue dots from the others with a blue dashed line, separate red dots from the others with a red dashed line and separate green dots from the others with a green dashed line. Then we draw three lines which are the angle bisectors and divide the plane into three areas. Now we want to classify the black dot. We find its geometric margin to the blue dashed line is the smallest. The red dashed line tells it does not belong to class 'RED' and the green dashed line tells it does not belong to class 'GREEN'. Therefore, we predict it to be in class 'BLUE'. We can easily check other cases and obtain the same results as the algorithm achieves, which is quite reasonable.

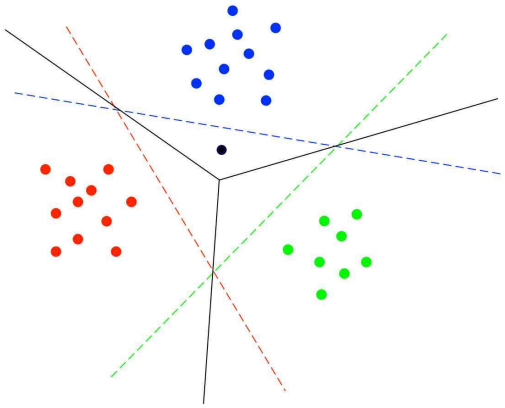

Figure 1: *Simple Example*

So we conclude our algorithm as follows:

Table 1: *Algorithm*

Pseudo Code

for j=1:16
    if $y^{(i)} = j$
        set $Y^{(i)} = 1$;
    else
        set $Y^{(i)} = -1$;
    end
    run SVM using (X, Y) to get ω_j and b_j;
end
for j=1:16
    calculate $\gamma_j = \frac{|\omega_j^T x + b_j|}{\|\omega_j\|}$;
end
sort γ_j from the largest to the smallest;
find the first j such that $\omega_j^T x + b_j > 0$;
if such j can be found
    assign $y = j$;
else
    assign $y = j_{16}$;
end

In addition, if the training examples are not well separable, we can also use SVM with

l_1 regularization. Just need to modify this algorithm and run SVM with l_1 regularization for 16 times to get 16 hyperplanes.

IV. Feature Selection and Results

For classification problems, the accuracy is a vital performance measurement of the classifier. To test the accuracy, 10-fold cross-validation technique is used in our experiments.

Since the results we obtain by using all features are unsatisfactory, we decide to implement feature selection to our above-mentioned models to improve the accuracy.

In class, three kinds of heuristic search procedures used for feature selection were introduced: forward search, backward search, and filter feature selection. We choose forward search because of its easy implementation and good performance.

For our four models, accuracy before feature selection, after feature selection and numbers of selected features are shown in the following table.

Table 2: *Results*

Model	Accuracy (before)	Number of Selected Features	Accuracy (after)
Logistic Regression	50.67%	42	68.22%
SVM	62.44%	50	75.56%
Multiclass Naive Bayes	64.28%	52	72.67%
Multiclass SVM	61.11%	—	—

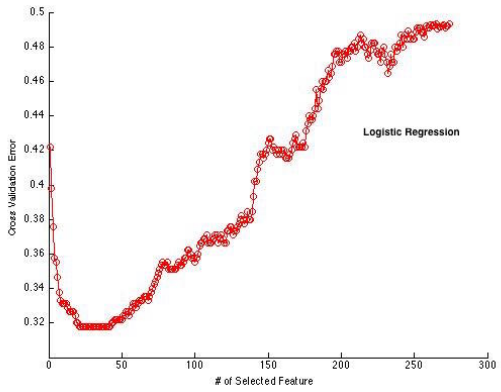

Figure 2: *Feature Selection for Logistic Regression*

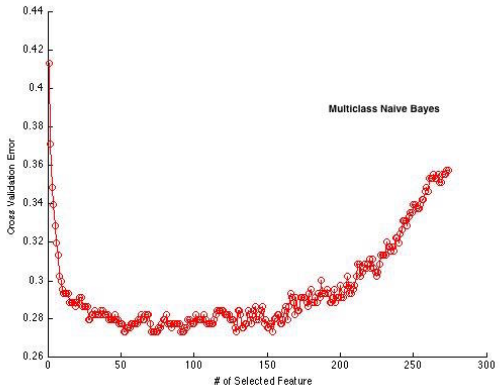

Figure 3: *Feature Selection for Multiclass Naive Bayes*

V. Discussion

In our project, before feature selection, Naive Bayes achieves lower cross validation error than SVM. While after feature selection, SVM achieves lower cross validation error than Naive Bayes. We think the problem may lie in the lack of enough training examples(475) and excessive amount of features(274). Just as we can see in Problem Set 2 when we use both Naive Bayes and SVM to classify spams and non-spams, SVM algorithm works better if there are more training examples. However, in this problem, only 475 training examples are available.

As shown in the Table 2, we didn't implement feature selection for Multiclass SVM. This is because all the codes are written by ourselves and they are not efficient enough to complete feature selection within acceptable time.

VI. Future

There are certainly rooms for improvement. First, we can extract new features. Since we are provided with ECG raw data, methods like FFT and wavelet decomposition can be used to gain new features that cannot be easily recognized in time domain. Then we need to give these features some physiological explanations. Moreover, we can also use deep learning methods to generate new features. Also, we can use PCA to reduce the dimension of feature space and figure out what features are informative.

References

[1] Guvenir, H. Altay, et al. "A supervised machine learning algorithm for arrhythmia analysis." Computers in Cardiology 1997. IEEE, 1997.

[2] Mishra, Binod Kumar, Prashant Lakkadwala, and Naveen Kumar Shrivastava. "Novel Approach to Predict Cardiovascular Disease Using Incremental SVM." Communication Systems and Network Technologies (CSNT), 2013 International Conference on. IEEE, 2013.