**Initial Draft:**

**Comparison between Multinomial Logistic Regression, Naïve Bayes and Neuron Network on Cardiac wave identification**

**by**

**Mohammad Mahmood**

**Phuong Nguyen**

**Introduction:**

Being able to predict whether a person’s what type of heart beat a person has is crucial in deciding whether that person need some medical assistance or not. As Mohammad has already been doing research on modeling the heart. We decided to create three machine learning algorithms to see if we will be able to predict what type of heart beat is occurring. We will then compare the performance of these three different algorithms to see which one works best.

Dataset we have is from Mohammad’s research on the computational modeling regarding the electrical and mechanical physiology of the cardiac tissue. Since this is a preliminary study and because of computational constraints, the data set doesn’t contain the propagation of voltage through the whole heart but rather just the small length of cardiac tissue. This data set generated from a MATLAB code based on the computational formulas by Luo Rudy model.

The data set contains a set of 190 voltages propagating through a 1.25cm cardiac tissue, the training set. So, it has 190 voltages for 121 features, these features are actually a group of 5 cells. Therefore, the size of trainset is 190 by 121(Xtrain). There is also a set of data with the same dimensions but different voltages, the test data (Xtest). All the Xtrain and Xtest data are of type float.

Moreover, each of the respective training and test data have a file classifying the type of cardiac wave it has (ytrain , ytest respectively) and the size is 190 by 3. For instance, for the first row (one heart beat) of the Xtrain set has a respective value of have 1 0 0 in the first row of ytrain. Classifying a normal heart beat.

The cardiac wave is being classified in to three different forms, normal (1 0 0), spiral (0 1 0), and wave break (0 0 1). The normal cardiac wave signifies the heart beat is normal everything is fine. The spiral wave signifies that heart is in tachycardia (heart rate is extremely high) and this could potentially lead to a ventricular fibrillation causing a myocardial infarction (heart attack). The wave break means that the heart is in the state of ventricular fibrillation causing a myocardial infarction. All the ytrain and ytest data are of type integers.

An example of the first row of the Xtrain and ytrain voltage data set is:

Xtrain:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| -72.824  (x1) | -69.066  (x2) | -57.866  (x3) | -43.88  (x4) | …….. | -82.846  (x121) |

ytrain:

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |

So, we are designing three algorithms that can help us classify the three different type of heart beats. Then comparing them to see which one performs better for classifying the type of heart beat. Furthermore, we will also test these algorithms on another set of data that has the displacement of the cardiac tissue (how much it stretches) in 190 heart beats. It has the same size (190 by the 121) and similar classification file (190 by 3). It also has a training and test set. We will then compare performance of the algorithms on displacement data set to the ones on the voltages. That will also potentially tell us whether voltage or displacement should be used for classification of the heart beats.

From the results obtained, in the future we can then proceed by doing the study on larger data set which is obtained from whole a heart or use an actual human hearts data obtained from a number of patients and instead of computer simulation. Potentially leading to a machine learning algorithm to detect the type of heart beat so that the patient receives the required medical attention in crucial time.

**Methodology:**

The main goal of our experiment is to classify the three different types of cardiac waves. Meaning that we multi-class classification problem(three classes in total). Therefore, the three algorithms we decided to implement were the following:

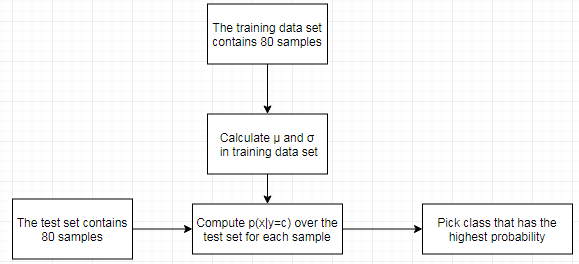
1. Naïve Bayes (generative model)
2. Multinomial logistic regression
3. Neural Network with one hidden layer

The first algorithm is Naïve Bayes. Based on strong independence assumption between features, Naïve Bayes is a generative approach for classification. One advantage of Naïve Bayes is when conditional independence assumption is actually true, a Naïve Bayes classifier will be likely to converge more quickly than discriminative algorithms. It means that the less training data is required.

As we have a continuous set of features, we assume a Gaussian distribution. Hence, the Naïve Bayes involves computing the mean and variance of feature j across the data points labeled with class c, as shown as below (from page 90 of the notes):

Then the prediction is made based on the following equation (Gaussian distribution):

With multi class classification, the classifier will make a decision () based on the highest probability value. The diagram for the Naïve Bayes algorithm is presented in…



*Figure 1: The simple diagram for Naïve Bayes algorithm*

**Algorithm**: Naïve Bayes algorithm

// Naïve Bayes

1: **(mean)** and **2(variance)**🡨 matrix j by c (j: number of columns, c: number of classes)

// Calculate and **2** from the training set

2: **for** j = 1, …, number columns

3: **for** c = 1,…, number of classes

4: Calculate and from the training set

// After that, making prediction over test set

5: **for** i =1, …, n (number of samples (rows))

6: **for** j = 1,…, number of columns

7**: for** c = 1,…, number of classes

8**:** Calculate

9: Pick class that has the highest probability value

The second algorithm to be implemented is Multinomial Logistic Regression. This algorithm is a classification method which generalizes logistic regression to multiclass problems. One advantage of multinomial logistic regression is that the model can be updated easily with new data using gradient descent method. In other words, the new training data is easier to be incorporated into the existing model.

The transfer function (inverse link) in this case is the SoftMax transfer.

The softmax transfer is (number of classes = k):

With the gradient:

In our problem, there are 3 discrete outcomes, the probability of the outcome for each sample is computed, as shown below:

Like Naïve Bayes, the multinomial logistic regression classifier picks class with the highest probability. The multinomial logistic regression algorithm will be:

**Algorithm**: Multinomial logistic regression

// Multinomial logistic regression with stochastic gradient descent

1: **w** 🡨 random vector

2: **for** i = 1, … number of epochs

3: Shuffle data points from 1,…,n

4: **for** j =1, …, n

5: g 🡨

6: **w** 🡨 **w** -

7: **return w**

**// Next**, make prediction

8: Compute

9: **for** i = 1, …, n

Pick the class with highest probability value (Choose the highest value of )

The third algorithm being implemented is a Neural network with single hidden layer. The neural network algorithm has ability to learn a non-linear model, which is important in the real world where most of the problems are non-linear. Like in our case the data is a classification type with no particular linearity. Another advantage of this algorithm is that after learning the training data, the model can be used to infer relationships between input data. However, there are some drawbacks making it difficult to optimize and understand the properties of the solutions. Overall, the neural network is a powerful algorithm, and we expect it to perform the best.

In the neural network with one hidden layer, the backpropagation is computed by first propagating forward:

And then propagating the gradient back:

**Algorithm**: Neural network with single hidden layer

// Neural network with single hidden layer with stochastic gradient descent

1: **w1** and **w2** 🡨 random vector

2: **for** i = 1, … number of epochs

3: Shuffle data points from 1,…,n

4: **for** j =1, …, n

**//** Calculate forward propagation

5: Calculate **h** and

// Calculate backward propagation

6: g1 🡨 and g2 🡨

7: **w1** 🡨 **w1** - and **w2** 🡨 **w2** -

8: **return w1** and **w2**

**// Next**, make prediction

In this experiment, the linear regression algorithm will also be implemented. Even though linear regression is a not a standard algorithm for classification, this algorithm is still chosen for a sanity check. In other words, other algorithms are expected to be able to outperform linear regression. To implement linear regression, we will use follow the guide lines in page 73 of the notes.

All the algorithms mentioned above will be coded in python with the help of packages like scikit and scipy.

**Evaluation:**

At the beginning we will determine the best meta-parameters for each algorithm. The training data has 190 samples and might be imbalanced, so we will use the stratified k-fold cross-validation to make sure all the specific folds of the data same ratio of the samples belonging to each class. K = 5 folds is chosen in this cross-validation. The training data set will be partitioned into 5 small groups but preserving the percentage of samples for each class. With stratified k-fold cross validation, each group is a good representative of the whole training set. This generates a better design both in terms of bias and variance.

Table 1: Groups after partitioning

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | 1 | 2 | 3 | 4 | 5 |
| Number of samples | 38 | 38 | 38 | 38 | 38 |

The methodology for tuning a meta parameter is presented as below:

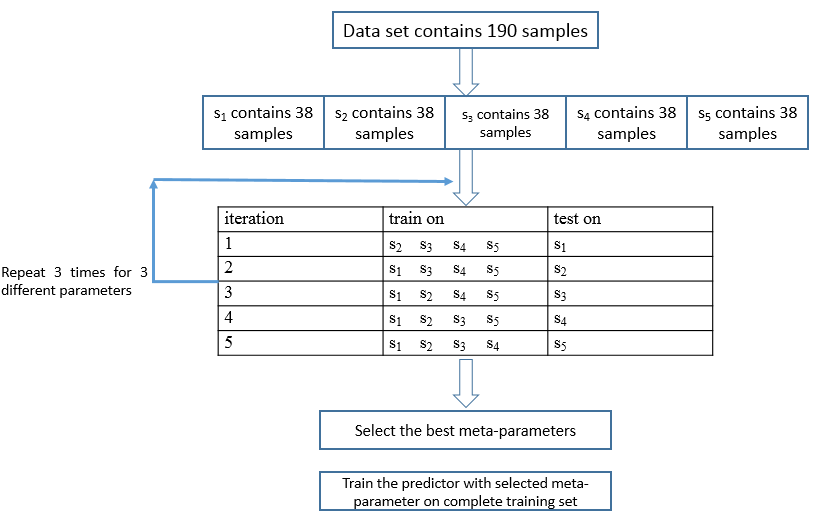


Figure 2: The diagram for stratified k-fold cross validation in this report

**1) Multinomial logistic regression:**

The meta-parameter chosen in stratified k-fold cross validation for multinomial logistic regression is step size. Three different values of step size chosen are 0.1, 0.01 and 0.001.

Table 2: Example Summary k-fold cross validation with multinomial logistic regression

|  |  |
| --- | --- |
| step size | average error |
| 0.1 |  |
| 0.01 |  |
| 0.001 |  |

**2) Neural network:**

The meta-parameter chosen in stratified k-fold cross validation for neural network is number of neurons in hidden layer. Three different values of number of neurons are 4, 16 and 32.

Table 3: Example Summary k-fold cross validation with multinomial logistic regression

|  |  |
| --- | --- |
| number of neurons in hidden layer | average error |
| 4 |  |
| 16 |  |
| 32 |  |

Table 4: Example Summary of the accuracies obtained from the best meta parameters

|  |  |  |  |
| --- | --- | --- | --- |
| iteration | train on | test on | correct |
| 1 |  |  | /38 |
| 2 |  |  | /38 |
| 3 |  |  | /38 |
| 4 |  |  | /38 |
| 5 |  |  | /38 |
| **Accuracy = /190 = %** | | | |

Table 5: Example of a Confusion matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | Actual class | | |
| Group 1 | Group 2 | Group 3 |
| Predicted class | Group 1 |  |  |  |
| Group 2 |  |  |  |
| Group 3 |  |  |  |

After finding optimal meta parameters from cross validation. We will run the three algorithms on test data set and then we will compute the following hyper parameters as performance measures:

1. Classification accuracy
2. Precision
3. Recall
4. Runtime of the algorithm

These hyper parameters were chosen because we have a classification problem.

**Statistical significance test:**

We will be performing paired t-test (two tailed) to check the statistical significance of our performance comparison between the three algorithms. Meaning we will check if each of our hyperparameters are statistically comparable or not. Assuming that our hyperparameters are normally distributed. We will have three null and alternative hypotheses for each hyperparameter. For instance, looking at the classification accuracy, our three null hypotheses(H0) will be:

* H01: accuracy of algorithm 1 is equal to the accuracy of algorithm 2
* H02: accuracy of algorithm 2 is equal to the accuracy of algorithm 3
* H03: accuracy of algorithm 1 is equal to the accuracy of algorithm 3

Our alternative hypothesis signifies that our algorithms are different.

* Ha1: accuracy of algorithm 1 is not equal to the accuracy of algorithm 2
* Ha2: accuracy of algorithm 2 is not equal to the accuracy of algorithm 3
* Ha3: accuracy of algorithm 1 is not equal to the accuracy of algorithm 3

We next calculate the t random variable based on the student t distribution. Compare it to the random variable T, relative to the computed statistic. If the probability that T is larger than the absolute value of t, like if the p = Pr (T > |t|) in a 95% confidence interval. Then we can conclude that the algorithms are different statistically and can reject the null hypothesis. Otherwise we can’t reject the null hypothesis, meaning that the two specific algorithms can’t be compared to one another based on that specific hyperparameter with a statistical significance.

Finally, after checking statistical significance, we then compare the three algorithms.

**Results and Discussion:**