605.649 Programming Project 4: Logistic Regression and Adaline

Christopher El-Khouri

1. Introduction

In the field of machine learning, linear models are the simplest of algorithms and function by generating a line of best fit to successfully predict the value of the target variable using the feature attributes. A typical linear model has the following form:

$$(1) \qquad f(x) = w_0 + \sum_{i=1}^{d} w_i x_i = w_0 + \boldsymbol{w}_i^T \boldsymbol{x}$$

Our main focus for this project will be on linear classifiers, classification algorithms where linear modeling is used to predict the class of given input data. We have previously experimented with linear classifiers in Project 1: Winnow-2 and Naïve Bayes, here we will be analyzing the performance of the Logistic Regression classifier and the Adaline classifier.

Logistic Regression is a supervised linear machine learning algorithm that uses the logistic distribution to classify a dependent variable. The detailed history of how logistic regression was developed is documented by J.S. Cramer (2002), with origins dating back to the 1800s in the topics of population growth. In equation (2), $P(C_i|x)$ refers to the probability that input feature data *x* belongs to a class C_i.

Taking $f(x)$ from (1):

$$(2) \qquad P(C_i|x) = \frac{\exp(f(x_i))}{\sum_{j=1}^{K} \exp\left(f(x_j)\right)}, i = 1, \dots, K$$

Adaline, which is short for Adaptive Linear Neuron, is a supervised linear neural network algorithm created by Bernard Widrow (1960). It is a 2-layer network, where the first layer consists of input distribution nodes, and the second layer consists of weighted summation nodes. In Figure 1, the Σ function takes the form of Equation (1). The weighted sum is calculated and the error is recorded and used to modify the weights.

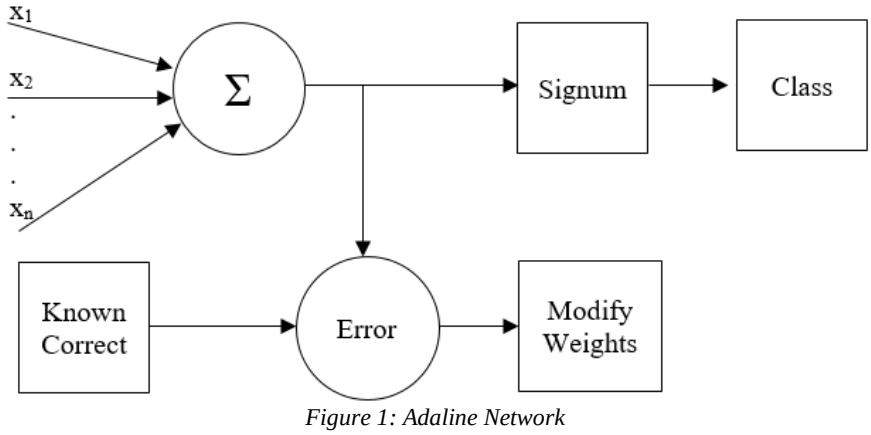

Figure 1: Adaline Network

We will be comparing the performance of the two different algorithms on 5 different datasets. We hypothesize that the accuracy and runtime of the outputs produced by the Logistic Regression algorithm will be superior to the accuracy and runtime of the outputs by Adaline. The accuracy of our classification problems will be calculated using the classification error, which is the percentage of incorrectly classified points.

In the next section, we will describe the technical and theoretical basis of our algorithms, in section 3 we will list the datasets involved in this experiment, section 4 will present and compare the results, and in section 5 we will discuss the significance of our experiment and compare the performance with the other algorithms previously encountered in the course with the same datasets.

2. Algorithms and Experimental Methods

Logistic Regression

The Logistic Regression algorithm used in this project is based on the following pseudocode:

For $i = 1, \ldots, K$
  For $j = 0, \ldots, d$
    $w_{ij} \leftarrow \text{rand}(-0.01, 0.01)$
Repeat
  For $i = 1, \ldots, K$
    For $j = 0, \ldots, d$
      $\Delta w_{ij} \leftarrow 0$
  For $t = 1, \ldots, N$
    For $i = 1, \ldots, K$
      $o_i \leftarrow 0$
      For $j = 0, \ldots, d$
        $o_i \leftarrow o_i + w_{ij} x_j^t$
    For $i = 1, \ldots, K$
      $y_i \leftarrow \exp(o_i) / \sum_k \exp(o_k)$
    For $i = 1, \ldots, K$
      For $j = 0, \ldots, d$
        $\Delta w_{ij} \leftarrow \Delta w_{ij} + (r_i^t - y_i) x_j^t$
  For $i = 1, \ldots, K$
    For $j = 0, \ldots, d$
      $w_{ij} \leftarrow w_{ij} + \eta \Delta w_{ij}$
Until convergence

Figure 2: Logistic Regression pseudocode

In figure 2, the dataset being inputted into the algorithm consists of d features, K classes, and N data points. **r** takes a value of 1 if $\boldsymbol{r}_i^t$ is equal to class C_i and 0 otherwise, x_0^t is taken as 1. Gradient descent is being implemented on an incremental basis as the dataset is being traversed row by row.

This algorithm has been modified for our project by defining the convergence as the maximum error difference between the previous values of the coefficients w and the new values to be less than 10^{-2}. Furthermore, the implementation of gradient descent on the evaluation of w has been modified from incremental updating to batch updating by taking the average of the Δw_{ij} values of a complete pass over the dataset. Batch updating has been implemented to enhance processing time when training the model. The modified algorithm is shown below:

LogisticRegression(**df**(**X**, **r**), **n**):
Where:
df: A data frame containing d features X, and target variable r with K classes
n: The learning rate upon which the coefficients w are updated

For $i = 1, \dots, K$
    For $j = 0, \dots, d$
        $w_{ij} = \text{rand}(-0.01, 0.01)$
While $\text{err} > 10^{-2}$:
    $\text{oldw} = w$
    For $i = 1, \dots, K$
        For $j = 0, \dots, d$
            $\Delta w_{ij} = 0$
    For $i = 1, \dots, K$
        $o_i = 0$
        For $j = 0, \dots, d$
            $o_i = o_i + w_{ij}\mathbf{X}_j$
    For $i = 1, \dots, K$
        $y_i = \frac{\exp(o_i)}{\sum_K \exp(o_K)}$
    For $i = 1, \dots, K$
        For $j = 0, \dots, d$
            $\Delta w_{ij} = \Delta w_{ij} + \text{mean}\left((\mathbf{r}_i - \mathbf{y}_i)\mathbf{X}_j\right)$
    For $i = 1, \dots, K$
        For $j = 0, \dots, d$
            $w_{ij} = w_{ij} + n\Delta w_{ij}$
    $\text{err} = \max\left(\left|\frac{w - \text{oldw}}{\text{oldw}}\right|\right)$

Tuning

The Logistic Regression model is initially tuned by extracting 10% of the data to be used as a tuning set to tune the learning rate n. The algorithm above is then run on the tuning set while varying the values of n from 1 to 0.01. The largest n value that results in the convergence of the weighted coefficients w in the least amount of epochs is chosen as the learning rate to be used to train our model.

5-fold cross-validation

After extracting 10% of our dataset for tuning, the remaining 90% of the data underwent 5-fold cross-validation. The data is split into five parts (or folds) and the model is trained on each of the folds accordingly.

In each run, one of the five folds is used as a test set and the remaining four folds are concatenated and used as the training set. The model is trained by running the algorithm above with the tuned n on the training set and using the converged w to predict the class values of the test set.

$$y_i = \hat{P}(C_i|\boldsymbol{x}) = \frac{\exp(\boldsymbol{w}_i^T \boldsymbol{x} + w_{i0})}{\sum_{j=1}^{K} \exp(\boldsymbol{w}_j^T \boldsymbol{x} + w_{j0})}, i = 1, \dots, K$$

The class *i* that results in the highest value of *y* is selected. The performance of the model is determined by calculating the classification errors of each of the folds by dividing the number of incorrect predictions over the total size of the folds, then calculating the mean of the errors.

Adaline

The Adaline algorithm used in this project is based on the following pseudocode:

Adaline(**df**(**X**, **r**), **n**):
Where:
df: A data frame containing d features X, and target variable r with K classes
n: The learning rate upon which the coefficients w are updated

For i = 1, … , K
    For j = 0, … , d
        $w_{ij} = 0$
While err > 10^{-2}:
    oldw = w
    For i = 1, … , K
        For j = 0, … , d
            $\Delta w_{ij} = 0$
    For i = 1, … , K
        $u_i = 0$
        For j = 0, … , d
            $u_i = u_i + w_{ij}\mathbf{X}_j$
    For i = 1, … , K
        For j = 0, … , d
            $\Delta w_{ij} = \Delta w_{ij} + \text{mean}\left((\mathbf{r}_i - \mathbf{u}_i)\mathbf{X}_j\right)$
    For i = 1, … , K
        For j = 0, … , d
            $w_{ij} = w_{ij} + n\Delta w_{ij}$
    $\text{err} = \max\left(\left|\frac{w - \text{oldw}}{\text{oldw}}\right|\right)$

In the algorithm above, the dataset being inputted into the algorithm consists of d features and K classes. **r** takes a value of 1 if $\boldsymbol{r}_i^t$ is equal to class C_i and -1 otherwise, x_0^t is taken as 1. Convergence is achieved when the maximum error difference between the previous values of the coefficients w and the new values to be less than 10^{-2}.

Furthermore, the learning rule of the coefficients follows the LMS (least mean square) algorithm and the Widrow-Hoff learning rule and this is implemented using batch updating by taking the average of the Δw_{ij} values of a complete pass over the dataset.

5-fold cross-validation

5-fold cross-validation is implemented for training and fitting our model, the data is split into five parts (or folds) and the model is trained on each of the folds accordingly. In each run, one of the five folds is used as a test set and the remaining four folds are concatenated and used as the training set. The algorithm runs on the training set while varying the values of n from 1 to 0.01. The weighted coefficients w resulting from the largest n value that results in the convergence in the least amount of epochs are chosen as coefficients to test our data.

$$u_i = \boldsymbol{w}_i^T \boldsymbol{x} + w_{i0}$$
$$i = 1, \dots, K$$

A one vs all approach was implemented, where the class *i* that results in the highest value of u is selected. The performance of the model is determined by calculating the classification errors of each of the folds by dividing the number of incorrect predictions over the total size of the folds, then calculating the mean of the errors.

3. Datasets

The following datasets were used in this project:

1. Breast Cancer
2. Glass
3. Iris
4. Soybean (Small)
5. Vote

Dataset 1: Breast Cancer

This breast cancer database was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg. The dataset contains 699 points and 2 classes: Benign (denoted by '2') and Malignant (denoted by '4'). Including the class variable 'Class', the data contains a total of 11 features. 16 missing values need pre-processing and are all under the feature 'Bare Nuclei'. We will fill those missing values by separating the data into the 2 different classes and producing sampled data based on the median of the 'Bare Nuclei' feature of the 2 classes. The median was chosen due to the distribution of 'Bare Nuclei' data shown below:

Table 1: Statistical Description of Bare Nuclei Data by Class

Class=2	
	bare_nuclei
count	444
mean	1.34
std	1.18
min	1
25%	1
50%	1
75%	1
max	10

Class=4	
	bare_nuclei
count	239
mean	7.62
std	3.12
min	1
25%	5
50%	10
75%	10
max	10

Based on the table above, class 2's distribution shows that at least 75% of the data have a value of 1, class 4's distribution shows that at least 50% of the data have a value of 10. The mean values are 1.34 and 7.62 respectively, sampling the data based on the mean values will not give accurate simulations due to the values of 'Bare Nuclei' being round figures and since the data is skewed.

The feature attributes were scaled to have their values between -1 and 1. This was done by performing the following transformation:

$$X_{std} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

$$X_{scaled} = 2X_{std} - 1$$

Dataset 2: Glass
This study of the classification of types of glass was motivated by a criminological investigation. The dataset contains 214 instances, 6 classes, and 11 columns. The feature attribute values were scaled to have their values between -1 and 1.

Dataset 3: Iris
The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. The feature attribute values were scaled to have their values between -1 and 1.

Dataset 4: Soybean (Small)
The dataset consists of a small subset of the original soybean database. The dataset contains 47 instances, 4 classes, and 35 columns. Columns 10, 12-18, and 28-33 contained only 1 unique value and were dropped before modeling. The feature attribute values were scaled to have their values between -1 and 1.

Dataset 5: Vote
This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac. The dataset contains 435 instances, 2 classes, and 17 columns. The attribute values were discretized by giving 'y' values a value of 1, 'n' values a value of 0, and '?' values which represent 'abstain' were given a value of -1.

4. Results

Dataset 1: Breast Cancer (Classification)
Modeling the Breast Cancer dataset with our algorithms we get the following:

Table 2: Dataset 1 Results

Dataset 1: Breast Cancer						
N=699, K=2, d=9						
Fold	Logistic Regression			Adaline		
	n	Epochs	Classification Error	n	Epochs	Classification Error
1	1	48	2.38	0.1	34	3.57
2	1	45	3.97	0.1	35	4.29
3	1	46	2.40	0.1	34	3.60
4	1	48	5.56	0.1	122	4.29
5	1	49	4.76	0.1	36	5.00
Average	**1**	**47**	**3.81**	**0.1**	**52**	**4.15**

Dataset 2: Glass
Modeling the Glass dataset with our algorithms we get the following:

Table 3: Dataset 2 Results

Dataset 2: Glass						
N=214, K=6, d=10						
Fold	Logistic Regression			Adaline		
	n	Epochs	Classification Error	n	Epochs	Classification Error
1	1	152	12.82	0.01	789	39.53
2	1	366	18.42	0.1	417	34.88
3	1	325	17.95	0.01	499	28.57
4	1	523	10.53	0.1	447	23.26
5	1	329	17.95	0.1	686	32.56
Average	**1**	**339**	**15.53**	**0.06**	**568**	**31.76**

Dataset 3: Iris
Modeling the Iris dataset with our algorithms we get the following:

Table 4: Dataset 3 Results

Dataset 3: Iris						
N=150, K=3, d=4						
Fold	Logistic Regression			Adaline		
	n	Epochs	Classification Error	n	Epochs	Classification Error
1	1	109	14.81	1	339	16.67
2	1	97	7.41	1	94	10.00
3	1	123	3.70	1	212	20.00
4	1	115	7.41	1	150	16.67
5	1	261	7.41	1	254	23.33
Average	**1**	**141**	**8.15**	**1**	**210**	**17.33**

Dataset 4: Soybean (Small)
Modeling the Soybean (Small) dataset with our algorithms we get the following:

Table 5: Dataset 4 Results

Dataset 4: Soybean (Small)						
N=47, K=4, d=21						
Fold	Logistic Regression			Adaline		
	n	Epochs	Classification Error	n	Epochs	Classification Error
1	1	63	0.00	0.1	1793	0.00
2	1	175	0.00	0.1	824	10.00
3	1	129	0.00	0.1	2556	0.00
4	1	423	0.00	0.1	626	0.00
5	1	73	0.00	0.1	1677	11.11
Average	**1**	**173**	**0.00**	**0.1**	**1495**	**4.22**

Dataset 5: Vote
Modeling the Vote dataset with our algorithms we get the following:

Table 6: Dataset 5 Results

Dataset 5: Vote						
N=435, K=2, d=16						
Fold	Logistic Regression			Adaline		
	n	Epochs	Classification Error	n	Epochs	Classification Error
1	1	179	7.69	0.1	266	2.30
2	1	161	6.41	0.1	256	4.60
3	1	176	3.80	0.1	301	5.75
4	1	343	2.56	0.1	160	6.90
5	1	123	6.41	0.1	215	6.90
Average	**1**	**196**	**5.37**	**0.1**	**240**	**5.29**

Summary
Producing a summary table of our results:

Table 7: Results Summary

Dataset	**N**	**K**	**d**	**Logistic Regression**			**Adaline**		
				n	Epochs	Error	n	Epochs	Error
Dataset 1	699	2	9	1	47	3.81	0.1	52	4.15
Dataset 2	214	6	10	1	339	15.53	0.06	568	31.76
Dataset 3	150	3	4	1	141	8.15	1	210	17.33
Dataset 4	47	4	21	1	173	0	0.1	1495	4.22
Dataset 5	435	2	16	1	196	5.37	0.1	240	5.29
Average				**1**	**179**	**6.57**	**0.272**	**513**	**12.55**

5. Discussion

Referring back to our hypothesis that the accuracy and runtime of the models produced by the Logistic Regression algorithm will be superior to Adaline, it seems that our hypothesis stands regarding Datasets 1-4 with lower error values and lower required epochs resulting from the Logistic Regression models. Dataset 5, the Vote dataset, produced a similar yet a lower mean error with the Adaline model.

Looking at Table 7, on average, our Logistic Regression model resulted in higher learning rates with lower epochs. There may be correlations to be found between the size of the dataset (N), the number of classes (K), and the number of dimensions (d) with the way that the algorithms perform. In both algorithms, more epochs were required for datasets with larger K and d values, however, further research into this is required with conclusive results leading to more optimum model selection methods.

6. Conclusion

In conclusion, it seems that our hypothesis regarding running time stood in all datasets. As for model accuracy, our hypothesis stood for all datasets except for Dataset 5. However, I would not rush to completely void it, perhaps improving the method on how the learning rate was selected in our Logistic Regression algorithm may confirm our hypothesis. With our current tuning method in selecting the learning rate, all our learning rates were selected as 1, perhaps modifying the learning rate in Dataset 5 for Logistic Regression may have produced a model with superior accuracy to Adaline.

Comparing the results of our Naïve Bayes classifier previously assessed in Project 1 done on the same datasets:

Table 8: Comparison with Naive Bayes

Dataset	Error		
	Logistic Regression	Adaline	Naïve Bayes
Dataset 1	3.81	4.15	3.2
Dataset 2	15.53	31.76	36.8
Dataset 3	8.15	17.33	22.2
Dataset 4	0	4.22	0
Dataset 5	5.37	5.29	11.5
Average	**6.57**	**12.55**	**14.74**

Both Logistic Regression and Adaline outperformed Naïve Bayes in all the datasets except for Dataset 1 which had about the same performance.

7. References

J.S. Cramer.
The Origins of Logistic Regression.
Tinbergen Institute Working Paper, 2002-119/4, 2002.

Bernard Widrow.
An Adaptive "Adaline" Neuron Using Chemical "Memistors".
Technical Report 1553-2, Information Systems Laboratory, Stanford University, Stanford, California, 1960.