Comparing Naive Bayes and Logistic Regression

Ujjwal, Kumar
McGill University

Furaha, Damien
McGill University

Mobassera, Zaman
McGill University

{ujjwal.kumar, furaha.damien, mobassera.zaman}@mail.mcgill.ca

Abstract

This project provides an analysis of discriminative and generative machine learning as typified by logistic regression and naïve Bayes. In the work, we develop the naïve Bayes and logistic regression machine learning algorithms from scratch. We use k-fold cross validation to fine tune parameters of the models and carry out model evaluation. The aim of this investigation is to evaluate the performance of both models on a range of corpora with varying characteristics, including size of the corpora, and attribute types, whether continuous or categorical. Using four benchmark datasets, we observe that while the overall performances of the models are high, there is a negative trend with the increase in corpora size for both models. The comparison and validation of the outcomes of each model were achieved using statistical evaluation measures. Overall, naïve Bayes shows good performance on smaller datasets and outperforms logistic regression. However, on larger datasets[1], logistic regression depicts a superior performance. While K-fold cross validation helped in parameter tuning, we found that increasing the size of the the training did not always improve the accuracy of our models. The investigation shows that it is not possible to say which model is the best as their distinct performances depend on the corpora used.

1 Introduction

Several studies have compared linear classification techniques in machine learning. These studies[2] have shown that naïve Bayes and logistic regression can lead to two distinct regimes in terms of which algorithm performs better as corpora sizes vary.

The performance of the algorithms depends on a wide range of factors. Ranging from the size and structure of the corpora being used to the parameters used in tuning the models themselves. Because of this, it is important to understand what model works better when building solution for a particular application. In this project, we aim to understand the variation in performance of naïve Bayes and logistic regression depending on the dataset being used. We also explore different settings that could enhance or diminish the performance of these models.

In our investigation we use four datasets of varying sizes and complexity to compare the performances of these two algorithms. We vary the number of training, testing and validation sets in a quest to measure the optimal performance of both models on each corpus. For each corpora, we explore what attributes contribute more to the training and performance of the models. We perform analysis on the dataset and experimentally remove and re-add features based on standard deviation and covariance, while keeping track of the models' perfomance on the respective datasets. We split our respect dataset into training, validation and testing sets in validation and evaluation of our model. We compare the performance of the model on the validation and the testing sets to help in parameter tuning and prevent overfitting. Using evaluation measures like accuracy, confusion matrix, recall[...] like O. L. Mangasarian et al.[3], we observe that naïve Bayes classifier has a slightly higher overall performance than logistic regression in cases where the models are evaluated on smaller datasets.

1.1 Logistic Regression

Logistic regression models the probabilities for classification problems with two possible outcomes. It's an extension of the linear regression model for classification problems. Instead of fitting a straight line or hyper plane, the logistic regression model uses the logistic functions to squeeze the output of linear equation between 0 and 1. The Logistic Function is defined as:

$$Logistic(x) = \frac{1}{1 + e^{-x)}}$$

The hypothesis of logistic regression tends to limit the cost function between 0 and 1. Therefore, linear functions fail to represent it as it can be a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

$$0 \leq h_\theta(x) \leq 1$$

In order to map predicted values to probabilities, we use the logistic function. The function maps any real value into another value between 0 and 1. In machine learning, we use Logistic to map predictions to probabilities.

Logistic Regression becomes a classification technique only when a decision threshold is brought into the picture. The setting of threshold value is a very important aspect of logistic regression and is dependent on the classification problem itself.
Hypothesis Representation :

$$\sigma(Z) = \sigma(\beta_0 + \beta_1 X)$$

we expect that our hypothesis will gives values between 0 and 1.

$$h\theta(x) = \sigma(Z)$$

$$h\theta(X) = \frac{1}{(1 + \exp -(\beta_0 + \beta_1 X))}$$

we expect our classifier to give us a set of outputs or classes based on probability when we pass the inputs through a prediction function and returns a probability score between 0 and 1. **Cost function** The cost function represents optimization objective i.e. we create a cost function and minimize it so that we can develop an accurate model with minimum error.

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) if y = 1 \\ -log(1 - h_\theta(x) if y = 0 \end{cases} \tag{1}$$

when implementing logistic regression, our job is to learn weights for the construction of decision boundary, so that predicted target values are approximately equal to the test target. To learn the weight parameters we define the the above cost function which we should use to train the logistic regression model. A cost function is an estimator of how good or bad our model is in predicting the known output in general. For the purpose of calculating gradient we have to simplify the above equation to the one given below

$$\frac{-1}{m}[\sum_{i=1}^{m} y^i * log * h_\theta(x^i) + (1 - y^i) log(1 - h_\theta(x^i)]$$

Gradient Descent To reduce the cost value we have to use Gradient Descent. The main goal of Gradient Descent is to *minimize the cost value* $J(\theta)$

1.2 Naive Bayes

Naive Bayes is a generative linear classification algorithm used in binary and multi-class classification problems. The algorithm makes a strong assumption and rather than calculating the probability of each feature, the features are assumed to be conditionally independent to each other with respect to their class labels. Despite the fact that this is a very strong assumption, naïve Bayes generally does well when compared to other complex models[4]

$$P(C_k|X) = \frac{P(C_k)P(x|C_k)}{P(x)}$$

Being based on conditional probability, naïve Bayes uses Bayes' theorem to compute the maximum likelihood of an instance having a particular class label. Given the instance labels, the algorithm finds the maximum likelihood that the instance belongs to a certain class assuming that the instance features are conditionally independent.

$$P(x|C_k) = \Pi_{d=1}^{D} P(x_d|C_K)$$

Where, $P(x|C_k)$ is the likelihood of instance x belonging to class C_k given that the features 1 to d of X belong to the same class. It is common to maximize the logarithm likelihood instead to prevent underflow of probabilities.

2 Datasets

In this project, we used a total of four distinct datasets. For Dataset 1 (Ionosphere), the aim is to predict whether a radar return from ionosphere is 'good' or 'bad'. This radar data was collected by a system in Goose Bay, Labrador. The Ionosphere Dataset[5] has 351 instances. Each instance has 34 continuous attributes and a label of "good' or "bad". For Dataset 2 (Adult Data Set): also known as "Census Income" dataset, the goal is to anticipate whether the income exceeds 50K/yr based on census data. This dataset has 48842 instances. Unlike the first set, the instances have a combination of continuous and categorical attributes. We also operated on a Beast Cancer dataset that classifies 699 instances as "1", "2" or "3" with each instance having 10 continuous attributes. Lastly, we worked on a Lung Cancer dataset with 32 attributes classified as "B" or "M", each having 56 continuous attributes.

2.1 Pre-processing and clean up

Data pre-processing involves transforming raw data into an understandable format. In real world, data are generally incomplete. Therefore, after acquiring the datasets, we started off by looking for missing values to avoid drawing an inaccurate inference about the data. We handled the null values and removed all instances with missing or malformed features from the four datasets. For example, instances with '?' in dataset 2. Since machine learning models are based on mathematical equations, we only want to deal with numbers. Consequently, we used One-Hot Encoding to deal with categorical variables.

2.2 Data analysis

In addition to malformed or missing features, data can also be noisy and inconsistent. It is always a safer course of action to compute basic statistics on the data to understand it better. As a result, we calculated some stats, such as, mean, standard deviation and IQR of the attributes. For each dataset, we used count plots and histograms to visualize the distributions of features grouped by class. Moreover, computing the correlations between the features gave us a better insight into which features have a greater influence on the class label.

3 Experimental Methods

In order to investigate the performance of Naive Bayes against Logistic regression, we developed the models from scratch and performed tests on the models repeatedly. Depending on respective datasets, we tuned the input parameters to the model.

3.1 Model Design

3.1.1 Libraries used

In building the models, we used python's built in libraries numpy and pandas to load the datasets into arrays. We did not import any third party libraries like Sklearn or NLTK. Other libraries that we used were csv , random , pandas and math.

3.1.2 Building Naive Bayes

The naïve Bayes model implemented for this experiment uses the Bayes' theorem to calculate the logarithmic likelihood of instances belonging to a class. We use Gaussian probabilities for respective features to calculate likelihood of class labels. To prevent underflow of the logarithmic function, features with zero standard deviation are removed from the datasets. A parameter ϵ to the function.

$$P(x|C_k) = \Pi_{d=1}^{D} P(x_d|C_K)$$

Introducing Log likelihood means that if we want to know which class($\hat{y}$) x belongs to,

$$\hat{y} = argmax_{k \in (1...|c|)} P(C_k) \Pi_{i=1}^{n} P(x_i|C_k)$$

taking log likelihood, this becomes:

$$\hat{y} = max_{k \in (1...|c|)} (log(P(C_k)+\epsilon) + \sum_{i=1}^{n} log(P(x_i|C_k)+\epsilon))$$

The parameter ϵ is set at e^{-100} so that it sorely serves to prevent under flowing of the log function in cases of zero variance amongst features.
The key function of the model is the naive_bayes function that calls a fit function to fit the training set and then uses the testing set to predict class labels. We use K-fold cross validation to evaluate the performance of our model on unseen dataset. This helps in tuning K to maximize the accuracy of the model.

3.1.3 Building Logistic Regression

Logistic regression is statistical method for predicting binary classes. Logistic regression hypothesis generalizes from the linear regression hypothesis in that is uses the logistic function where it predicts the probability of occurrence of a binary event utilizing a logistic function

$$Logistic(x) = \frac{1}{1 + e^{-x)}}$$

The logistic function also known as Sigmoid function has Asymptotes at 0 and 1, and it crosses the y-axis at 1 and 0.5. Linear Regression Equation :

$$y = \beta_0(0) + \beta_1 X_1 + ... + \beta_n(X_n)$$

where y is dependent variable , target variable , and

$$x_1, .., x_n$$

are features of given instance. we then apply Sigmoid function on linear regression as follows :

$$p = \frac{1}{1 + e^{-(\beta_0(0)+\beta_1 X_1 + ... + \beta_n(X_n))}}$$

The target variable in logistic regression follows Bernoulli Distribution and Estimation is done through maximum likelihood. Our Main Goal is to learn the weights i.e. $\beta_0, \beta_1, ..., \beta_n$ to predict our target variable. Logistic Regression Model takes in Feature Matrix, which has features of any given instance as columns and rows as different instance. In Every iteration we updates the learned weights based using the Cost function given below :

$$\frac{-1}{m}[\sum_{i=1}^{m} y^i * log * h_\theta(x^i) + (1 - y^i) log(1 - h_\theta(x^i)]$$

The cost function has to be minimize in order to minimize the error in learned weights and this is done using Gradient Descent method. The key function in Logistic Regression model is that it calls a fit function to learn weights and then uses Test set to predict class labels and calculate the accuracy of the model. We use K-fold cross validation to evaluate the performance of our model on unseen data set. This helps in tuning K to maximize the accuracy of the model.

3.2 Experimental Settings and Model Parameters

Despite there being a large performance cost of some datasets on the models, we run tests repeatedly. Given that the dataset had features of different characteristics, we parameterized the models depending on the dataset being tested. We trained the models and used K-fold cross validation multiple times and used the best obtained parameters to run our testing sets.

For naïve Bayes, 10-fold cross validation achieved the best mean accuracy except for the ionosphere dataset[5] which had k=300 as its best cross-validation parameter. Despite setting $\epsilon = e^{-100}$ in maximizing log likelihood, we continuously varied it and noted the value that give our model the best performance.

Similarly, For Logistic Regression, 10 fold cross validation with learning rate set to 10^{-2} , $epoch = 300$ and taking $\epsilon = e^{-6}$ for all datasets achieved the best mean accuracy. decreasing from $\epsilon = e^{-6}$ to $\epsilon = e^{-20}$ decrease the mean accuracy on data set "ionosphere" from around 83% to 74% and for "Breast Cancer" from around 50% to 38%. "Lung Cancer" data set seems to be unaffected with the changes in ϵ values.

3.3 Test Conditions

In conducting the experiments,test conditions were set depending on the dataset. For all the datasets but lung Cancer, the dataset was split into 80%, 10% and 10% for training, validation and testing respectively. The testing set was left unused until the testing phase.

4 Results

For all the datasets used, the primary evaluation measure was accuracy for both the validation set and the testing set. We also obtained the confusion matrices for the datasets.

4.1 Logistic Regression Results

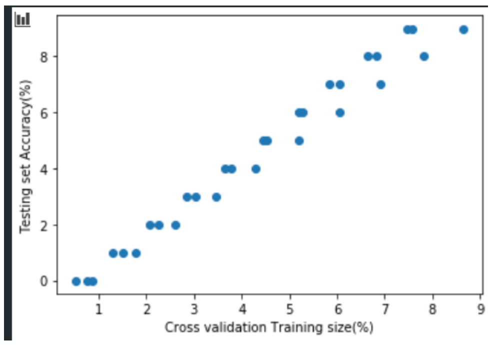
The "Ionosphere" dataset had 351 instances. We trained the model using learning rate 10^{-2}. We used 10% for testing the model using the learned weights. We used $K = 10$ for cross validation and $epoch = 700$ and we obtained mean accuracy of around $83.92\% \approx 84\%$ on the test set. Decreasing the $epoch$ to ≤ 700 or 700 decreases the mean accuracy for the data set during cross validation.
In "Lung Cancer" dataset, we trained the model with learning rate $= 10^{-2}$, and $epoch = 700$ with $k = 10$ for cross validation. We obtained the mean accuracy of around $\approx 50.0\%$. Changing hyper parameters values did not have much effect on the mean accuracy of the model. Using the learning rate 0.1 resulted in lower mean accuracy.

In "Breast Cancer" dataset, we used 10% of the dataset for testing the model using the learned weights. We used learning rate $= 10^{-2}$, $epoch = 700$ $\epsilon = e^{-6}$ for training the model. We achieved the mean accuracy of $61.41\% \approx= 61\%$ during cross validation.

In "Census Income" dataset, we used 10% of the dataset for testing the model using the learned weights. We used learning rate $= 10^{-2}$, $epoch = 700$ and $\epsilon = e^{-6}$ for training the model and achieved mean accuracy of $\approx 74\%$ during cross validation.
To further understand the performance of the model on the datasets, we obtained the confusion matrices of the datasets.

	IS	AD	BC	LC
TP	36	0	0	0
FP	1	683	42	0
TN	0	2036	12	0
FN	0	0	0	5

Table 1: Depiction of confusion matrices on the datasets for logistic regression model
IS = Ionosphere, **AD** Adult, **BC** = Breast Cancer, **LC** = Lung Cancer



From the given plot we can summarise that with every iteration of model the accuracy of the model increase linearly

4.2 Naive Bayes Results

The Ionosphere dataset had 351 instances. We used 10% for testing the model. Using K = 300 for cross validation, we obtained an 88.00% mean accuracy on the validation set and a 94.29% accuracy on the testing set. Applying the model to Dataset 2, we trained on 80% of the 30162 data points and tested on 10%. The model achieved mean accuracy of 72.70% on the validation set and 72.34% on the testing set. The lung cancer data set which had 27 data points after prepossessing was split in the same manner, obtaining 90.00% and 100% accuracies on the validation and testing sets, respectively. Finally, we run the experiment on the breast cancer dataset. Using K=20 from cross validation resulted in a 92.32% accuracy on validation set and 83.53% on testing set.
To further understand the performance of the model on the datasets, we obtained the confusion matrices for the datasets.

	IS	AD	BC	LC
Test size	35	3016	56	2
TP	15	1401	33	2
FP	0	0	0	0
TN	18	781	11	0
FN	2	834	12	0

Table 1: Depiction of confusion matrices on the datasets
IS = Ionosphere, **AD** Adult, **BC** = Breast Cancer, **LC** = Lung Cancer

We also looked at the performance of the model with respect to the training set size. In particular, we tested this on the breast cancer data set and observed how the accuracy of the prediction changed when we tested using an unseen testing set. The graph below shows how the performance changed.

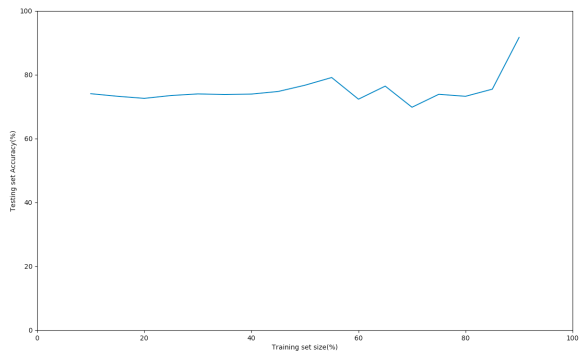


Fig 1: effects on training set size on prediction accuracy

We observe that we get the highest accuracy (92%) when we use 90 percent of the dataset for training.

5 Discussion and Conclusion

Both naïve Bayes and logistic regression can be used for classification of the data but they excel in different aspects. Naïve Bayes being a generative model, estimates a joint probability, given feature x and the label y, from the training data. Whereas, logistic regression being a discriminative model, estimates the probability $p(y|x)$ directly from the training data by minimizing error. Naïve Bayes places a strong constraint on the features of a given instance by assuming all the features being conditionally independent to each other. Therefore, if some of the features are strongly dependant on each other, thnn prediction might be poor. But in real world example this strong constraint followed by naïve Bayes model seem to work in most situations. On the other hand, logistic regression splits feature space linearly and does not put conditional independence constraint on the features of the given instance.
Both models seem to have some limitations. Naïve Bayes works well even with less training data, as the estimates are based on the joint density function. However, Logistic regression model might over fit the data, if given a small dataset to work with. We can follow various approaches to optimize the results.
for example: With naïve Bayes, when the training data size is less relative to the features, the information on prior probabilities helps in improving the results.
And with logistic regression model, when the training size is less relative to the features, Lasso and Ridge regression will help in improving the results.

6 Statement of Contributions

Ujjwal Kumar implemented logistic regression and ran experiments. Furaha Damien worked on Naive Bayes and ran experiments. Mobassera Zaman did data analysis and preprocessing. All of us worked on the report.

References

[1] O. L. Mangasarian and W. H. Wolberg: ”Cancer diagnosis via linear programming”, SIAM News, Volume 23, Number 5, September 1990, pp 1 18.

[2] Andrew Y. Ng , Michael I. Jordan , On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes

[3] Paraskevas Tsangaratos, Ioanna Ilia,Comparison of a logistic regression and Naïve Bayes classifier in landslide susceptibility assessments: The influence of models complexity and training dataset size, CATENA, Volume 145, 2016, Pages 164-179,

[4] I. Rish, An empirical study of the naïve Bayes classifier, T.J. Watson Research Center

[5] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.