COMP 551 Mini-Project 1

Christopher Scarvelis, 260689983
Nadia Blostein, 260804876
Vivian Eberle, 260745995

Abstract:

In this project, we implement a logistic regression model and a Naive Bayes model and study their respective accuracies, their ability to leverage larger amounts of data to improve their performance, as well as the impact of the learning rate on both accuracy and the amount of iterations for gradient descent in logistic regression. We train and test these models on four datasets using five-fold cross-validation and find that while the Naive Bayes classifier performs well on small training sets, the logistic model more effectively takes advantage of large training sets and typically outperforms Naive Bayes in such contexts.

Introduction

Naive Bayes and logistic regression are workhorse classification algorithms that have been extensively studied in the statistics literature. A logistic regression model assumes that given an instance described by a feature vector $x \in \mathbf{R^d}$, the corresponding label is given by the function:

$$f_w(x) = \sigma(w^T x) = \frac{1}{1 + e^{-w^T x}}$$

Convex optimization algorithms such as gradient descent are then used to solve for a weight vector $w \in \mathbf{R^d}$ that minimizes the cross-entropy loss of this model on a training set $(x_i, y_i)_{i=1}^n$.
On the other hand, a Naive Bayes model makes the strong assumption of feature independence conditional on the labels and uses Bayes' rule to solve for the posterior probability that a new instance belongs to a class c conditional on its associated features x. The parameters of a Naive Bayes model (namely the prior class probabilities and the parameters of the class-conditional distribution of each feature) are obtained in closed form from their maximum-likelihood estimates.
We train and test these two models on four datasets drawn from the UCI Machine Learning Repository: the Ionosphere dataset, the Adult dataset, the Wisconsin breast cancer dataset, and the Haberman cancer survival dataset. The Ionosphere dataset, in which one attempts to predict whether radar returns are "good" or "bad," was first analyzed by Sigillito et al.[1], who used it as an early demonstration of the effectiveness of multilayer neural networks for classification. The Adult dataset is a subset of 1994 US Census data from which one attempts to predict whether a given individual has an income over $50,000 per year. It was used by Kohavi[2] to carry out experiments on his proposed algorithm NBTree, a hybrid of decision-tree and Naive Bayes classifiers that is intended to scale to large datasets more favourably than a standard Naive Bayes classifier.
Our two freely-chosen datasets are the Wisconsin breast cancer dataset and the Haberman cancer survival dataset. The Wisconsin breast cancer set is a collection of clinical data reported by Dr. William H. Wolberg from 1989 to 1991. It saw its first machine learning application in Zhang [3]. The Haberman dataset consists of data collected at the UChicago Billings Hospital regarding the survival of patients who had received breast cancer surgeries. It was used as a example in Landwehr et al.[4] to demonstrate the application of three graphical methods for for assessing the fit of a logistic regression model.

Our results indicate that neither of the two classifiers achieves clearly superior performance (as measured by accuracy) on all datasets. We observe that the performance of Naive Bayes classifiers is relatively insensitive to the number of training examples used, while logistic regression tends to perform substantially better when it has access to large training sets. As expected, logistic regression achieves better accuracy when it is trained over a greater number of iterations, but our results show that it is difficult to choose learning rates to produce a substantial variation in the number of iterations for gradient descent: Either very few iterations are required before the change in the cost function falls below our chosen tolerances, or gradient descent ends by reaching a hard limit in the number of permitted iterations. Finally, training our logistic classifier using Armijo backtracking (a simple heuristic for automatically choosing the learning rate) leads to competitive performance on the two assigned datasets, but poor performance on our two freely-chosen datasets; this likely reflects difficulties in hyperparameter selection for Armijo backtracking.

Datasets

For **dataset 1**, column 1 was removed because its entries were the same across all features. Both according to the documentation and the fact that the dataset had no non-numerical columns, there were no missing values to handle. The output feature was binarized such that 1 = "good" and 0 = "bad". All of the data was converted into floats to avoid potential type-handling errors.

The graphs that characterize this dataset can be found in the **dataset_1/data_analysis** sub-directory. The features had a lot of variability (see the standard deviations in **mean_sd.png**). Most of the features had on average higher values in the good radar return group than in the bad one (see **mean_sd.png**). The 33 x 33 correlation matrix of all of the features with each other (see **correlation.png**) formed four visible clusters (see **dendrogram_cutoff.png**, where the red bar indicated the cut-off height). The frequency distributions of one randomly selected continuous feature per cluster were plotted (see **feature_{8, 16, 21, 29}.png**).

For **dataset 2**, all instances with missing features were removed. Categorical variables were converted into binary features using the pandas.get_dummies() function. Redundant columns were removed (for instance, it was not necessary to keep both the binary "Male and binary "Female columns). After columns were re-indexed, all of the data was converted into floating points to avoid type-handling errors.

The graphs that characterize this dataset can be found in the **dataset_2/data_analysis** sub-directory. The age feature in the > 50K class seems to follow a normal distribution (mean=44 $\pm$ 10 yrs). As expected, the age distribution in the ≤ 50K class is lower and has a positive skew (mean = 37 $\pm$ 13 yrs) (see **age.png**). No matter their annual income, a significant portion of the subjects work under private companies (see **work.png**). A significantly higher proportion of people with university degrees fall under the $>$50K class than under the $\leq$50K class. 91.18% of the instances in this dataset come from the United States (see **countries.png**). As shown by the violin plot of the final weight of each instance (see **fnlwgt.png**), the outliers differ significantly from the rest of the data. This feature needed to be scaled in order to avoid overflow with the logistic cost function.

For **dataset 3**, all instances with missing features were removed. The output feature was binarized such that 1 = "malignant tumour" and 0 = "benign tumour". All of the data was converted into floats to avoid potential type-handling errors. Column 0 (subject ID numbers) was removed since this feature was unnecessary for the model.

The graphs that characterize this dataset can be found in the **dataset_3/data_analysis** sub-directory. Malignant tumours had on average higher values across all features than benign tumours (see **mean_sd.png**). The 9 x 9 correlation matrix (see **correlation.png**) of all of the features with each other did not form clusters (see **clustered_correlation.png**).

For **dataset 4**, both according to the documentation and the fact that the dataset had no non-numerical columns, there were no missing values to handle. The output feature was binarized such that 1 = "the patient survived 5 years or longer and 0 = "the patient died within 5 years.

The graphs that characterize this dataset can be found in the **dataset_4/data_analysis** sub-directory. Features 0 and 1 were very similar across both classes. Feature 2 was on average higher in patients who died within 5 years (see **mean_sd.png**). None of the 3 features were correlated with each other (see **correlation.csv**).

Results

The following figures report a summary of our results regarding Naive Bayes and logistic regression. We start by looking at the convergence speed (number of iterations) for gradient descent in logistic regression, and its effect on accuracy. In fact, by increasing the learning rate of gradient descent, the number of iterations decrease, and accuracy tends to decrease. When plotting the iterations against the accuracy (Figure 1), we were able to find for which learning rates the number of iterations started peaking towards 1000 (maximal number of iterations). As such, we also reported the learning rate used when testing linear regression's accuracy (Figure 2). Figure 2 shows that compared to logistic regression (with the chosen learning rate), the mean accuracy of

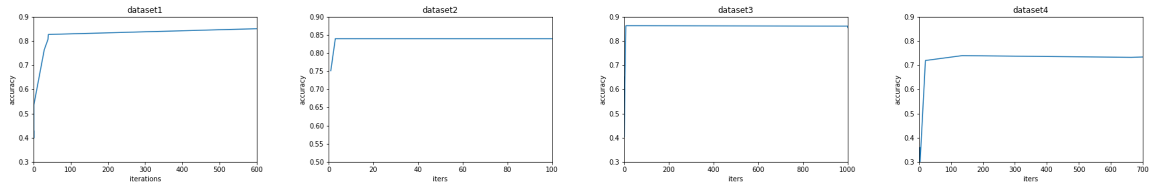
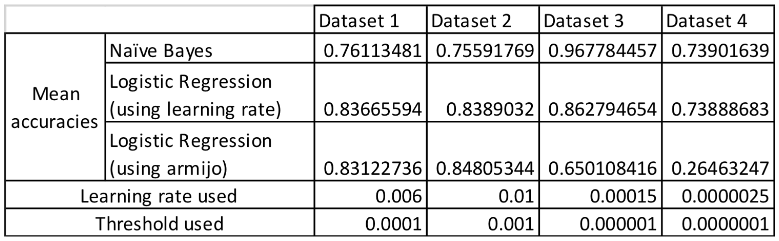


Figure 1: Datasets 1,2,3 and 4 plotting accuracy against iterations for logistic regression, using different learning rates



		Dataset 1	Dataset 2	Dataset 3	Dataset 4
Mean accuracies	Naïve Bayes	0.76113481	0.75591769	0.967784457	0.73901639
	Logistic Regression (using learning rate)	0.83665594	0.8389032	0.862794654	0.73888683
	Logistic Regression (using armijo)	0.83122736	0.84805344	0.650108416	0.26463247
Learning rate used		0.006	0.01	0.00015	0.0000025
Threshold used		0.0001	0.001	0.000001	0.0000001

Figure 2: Mean accuracies for Naive Bayes, logistic regression for datasets 1,2,3 and 4, with respective learning rates and threshold values used for linear regression.

Naive Bayes is lower in datasets 1 and 2, higher in dataset 3, and comparable in dataset 4.

In our analysis of Naive Bayes and logistic regression, we look at the effect of dataset size on the accuracy. Figures 3 and 4 show the plotting of dataset size and accuracy. Figures 5 and 6 show that of logistic regression with the learning rate (Figure 2), whereas Figures 7 and 8 show that of logistic regression using Armijo. We use 7 different sizes and two different ways to measure accuracy, namely: 1) isolating a test-set and reducing the size of the train-set, and 2) by using 5-fold cross validation on a smaller and smaller subset of the datasets.

Figure 3 and 4 show that when using Naive Bayes, accuracy does not seem to change much with a change in dataset size, for both methods of testing accuracy. When the dataset is at least approximately 100 instances large, Naive Bayes is not affected by size much.In fact, the most it varies is for dataset 4, the smallest dataset, where the size goes down to below 50 instances to under 300, and accuracy varies from 0.645 to 0.685 (Figure 3) and 0.71 to 0.75 (Figure 4).

Figures 5 and 6 show that when using linear regression, accuracy increases as the dataset sizes increase. The increase is most obvious in datasets 1,3 and 4, which are all substantially smaller than dataset 4, which sees a less drastic increase in accuracy, going from 0.830 to 0.836 accuracy (Figure 5), with 5000 to over 25000 instances. Comparatively, dataset 1 goes from 0.77 to 0.85 as the size of the dataset goes from below 100 to above 600 instances (Figure 5).

Figures 7 and 8 show that when using the Armijo rule, linear regression's accuracy has the same effect of increasing with dataset size, only if the Armijo rule works in the given datasets.

The Armijo rule is a simple heuristic for automatically choosing a learning rate for gradient descent. Given some smooth convex function $f : \mathbf{R}^\mathbf{d} \to \mathbf{R}$ that we seek to minimize and a descent direction d (we typically take $d := -\nabla f(x)$), the Armijo rule takes two hyperparameters $\beta, \sigma \in (0, 1)$ as input and chooses the following learning rate at a point x:

$$\alpha := \max_{l \in \mathbf{N_0}} \beta^l : f(x + \beta^l d) \leq f(x) + \beta^l \sigma \nabla f(x)^T d$$

In other words, the Armijo chooses the largest power of β such that a sufficient decrease condition is satisfied. This explain why datasets 3 and 4 do not work well with Armijo, as they have the lowest optimal learning rate, and the β xwould be too large. Also, since we were able to observe an adequate learning rate which was used to plot Figures 5 and 6, this might explain why Armijo curves are not as indicative relative to using the learning rates.

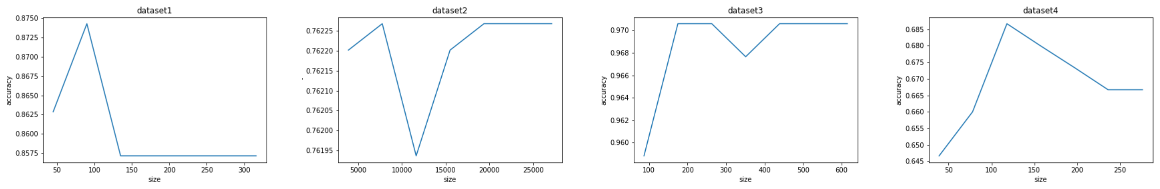


Figure 3: Datasets 1,2,3,4 plotting accuracy against dataset size for Naive Bayes

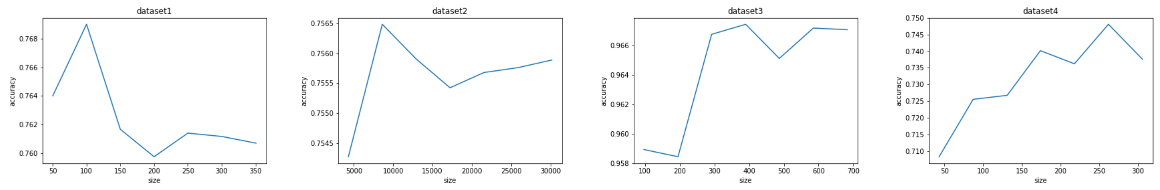


Figure 4: Datasets 1,2,3,4 plotting accuracy against dataset size for Naive Bayes, using 5-fold cross validation

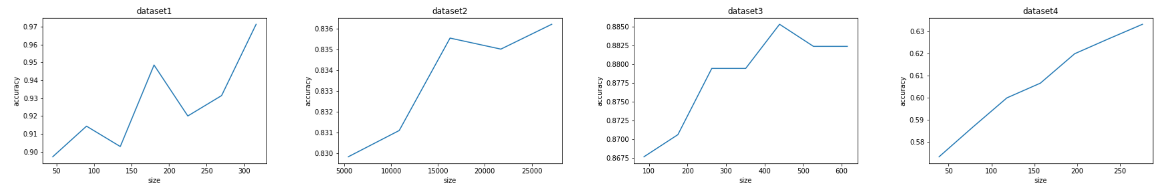


Figure 5: Datasets 1,2,3,4 plotting accuracy against dataset size for logistic regression, using their optimal learning rate and a train-test ratio

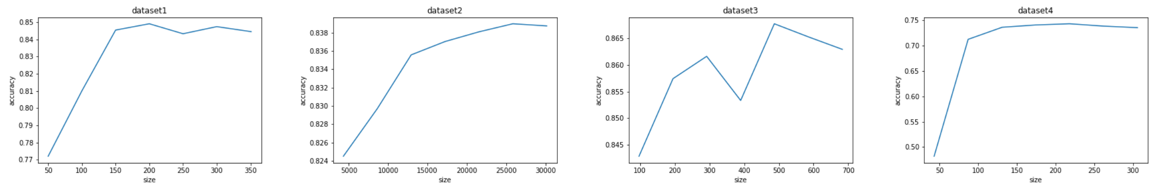


Figure 6: Datasets 1,2,3,4 plotting accuracy against dataset size for logistic regression using the optimal learning rate, and using 5-fold cross validation

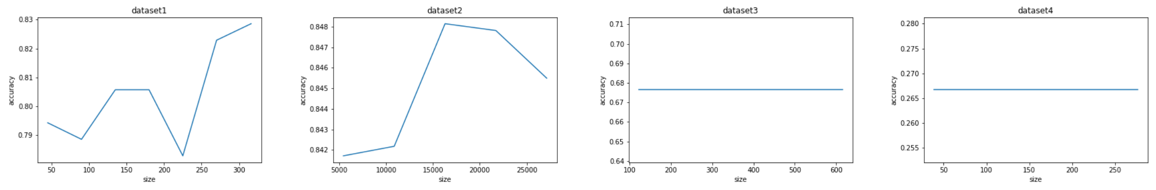


Figure 7: Datasets 1,2,3,4 plotting accuracy against dataset size for logistic regression, using Armijo fitting and a train-test ratio

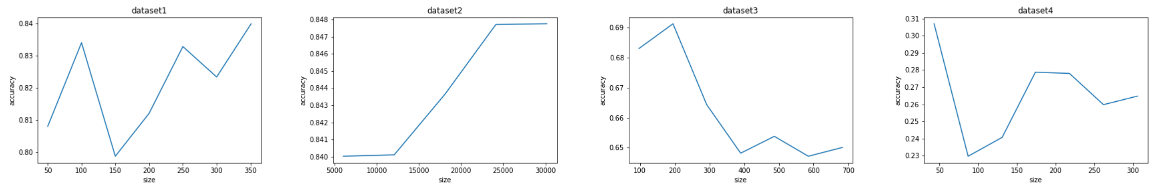


Figure 8: Datasets 1,2,3,4 plotting accuracy against dataset size for logistic regression using Armijo fitting, and using 5-fold Ccoss validation

Discussion and Conclusion

Perhaps the most important takeaway from this project is that neither logistic regression nor Naive Bayes are unequivocally superior classification methods. Either method may outperform the other on a suitable dataset: Naive Bayes performs notably better on the Wisconsin breast cancer dataset, whereas logistic regression is the stronger performer on the Ionosphere dataset (especially if it has access to the full set of training examples). This is in some sense a confirmation of the no-free-lunch theorem, which states that no single machine learning algorithm achieves good performance on all problems. As expected, giving either algorithm access to more training examples generally improves its performance (though this is not always clear for Naive Bayes). Similarly, allowing logistic regression to train through more iterations of gradient descent improves performance (likely by allowing the algorithm to reach a weight vector closer to the true global minimum of the cost).
Future work may explore the possibility of using kernel methods to embed the features in a high-dimensional space before training a logistic classifier. Logistic classification learns a linear decision boundary in feature space. Classes which are not linearly separable hence cannot be learned via logistic regression. However, sets of points are more likely to be linearly separable in high-dimensional spaces; this is formalized by Cover's theorem. [5] Kernel methods are one possible direction for accomplishing this, but it may be interesting to explore whether simply embedding each feature in a high-dimensional space via some function $\phi : \mathbf{R^{d_1}} \to \mathbf{R^{d_2}}$ for $d_2 > d_1$ before training a logistic classifier produces good results for data that is not linearly separable.

Statement of contributions

Christopher wrote the code for Task 2 as well as the introduction and discussion for this writeup, and assisted with debugging throughout the project. Nadia wrote the code for Task 1 and the dataset part of the writeup. Vivian wrote the code for Task 3 and the result part of the writeup.

References and Notes

[1] V. Sigillito, S. Wing, L. Hutton, K. Baker, *Johns Hopkins APL Technical Digest* (1989).

[2] R. Kohavi, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (1996).

[3] J. Zhang, *Proceedings of the Ninth International Machine Learning Conference*.

[4] J. Landwehr, D. Pregibon, A. Shoemaker, *Journal of the American Statistical Association* (1984).

[5] T. Cover, *IEEE Transactions on Electronic Computers* (1965).