Analysis and Comparison of Logistic Regression and Naive Bayes classifier

Andrew Cheng, Chris Rafiaa, Stephanie Li

Abstract

Machine learning algorithms are becoming a crucial part of making meaningful predictions given a large amount of data. Considering a set of relevant attributes, we can make predictions about the income of a person, authenticate banknotes, classify radar returns, and estimate the likelihood of breast cancer recurrence. We implemented logistic regression and naive Bayes algorithms to learn from such data sets. Our algorithms were designed for predicting binary classes problems, as opposed to continuous or categorical. The accuracy of the algorithms were achieved by training and validating our model via 5-fold cross-validation on 90% of the initial data and testing our final models on the remaining unseen 10% of the data. Based on our algorithms, we report that naive Bayes tends to perform better on smaller data sets, whereas logistic regression achieves higher accuracies on larger data sets. The naive Bayes algorithm is also significantly more efficient than logistic regression. To optimize the logistic regression model, we used methods including L1 regularization for automated feature selection and full-batch gradient descent to optimize the weights. Finally, we explored the effect on model accuracy of using constant learning rates compared to learning rates with momentum.

Background

Logistic regression and naive Bayes classifiers are machine learning algorithms used to predict the class to which a data point belongs. Although both algorithms can be used to classify data into any number of classes, we restricted the scope of this project to binary classification.

Binary logistic regression falls under the discriminative learning framework as it learns an optimal linear boundary to predict the class of a new data point. In other words, given N observations and D features, logistic regression assumes the linearity of independent variables and log-odds:

$$f_w(x_i) = \log(\frac{y_i}{1-y_i}) = w_0 + \sum_{j=1}^{D} w_j x_{ij}$$

and our goal is to find a set of weights $\mathbf{w} = \{w_0 \cdots w_D\}^T$ such that it maximizes the log-likelihood:

$$L(\mathbf{w}) = \sum_{i=1}^{N} y_i \log(\sigma_{\mathbf{w}}(x_i)) + (1-y_i)\log(1-\sigma_{\mathbf{w}}(x_i)).$$

where $\sigma_{\mathbf{w}}(x) = \frac{1}{1+\exp{(-\mathbf{w}^T x)}}$ is the sigmoid function which is the inverse of log-odds. Equivalently, we can find weights that minimize $-L(\mathbf{w})$, which is known as the cross entropy loss function and is a convex function. In addition, the L1 norm, $||\mathbf{w}||_1^1$, which is known to induce sparsity in models, is also convex. By the fact that the sum of two convex functions is convex, we have that the optimization problem,

$$argmin_{\mathbf{w}}(-L(\mathbf{w}) + \lambda||\mathbf{w}||_1^1) \tag{1}$$

is convex for any $\lambda \in \mathcal{R}$. This means that full-batch gradient descent will, in theory, discover the optimal set of weights to minimize (1), i.e. minimize the cross entropy loss of logistic regression with L1 regularization. However, in practice, gradient descent steps may oscillate and take many iterations to converge resulting in expensive computational costs. A natural solution is to inject momentum into the gradient updates:

$$\Delta\mathbf{w}^{(t)} \leftarrow \beta\Delta\mathbf{w}^{(t-1)} + (1-\beta)\nabla(J(\mathbf{w^{(t)}})$$

$$\mathbf{w}^{(t)} \leftarrow \mathbf{w}^{(t-1)} + \alpha\Delta\mathbf{w}^{(t)}$$

$$\Delta\mathbf{w}^{(t)} = \sum_{t=1}^{T} \beta^{T-t}(1-\beta)J(\mathbf{w}^{(t)} \tag{2}$$

where J is the cost function, $\beta \in (0,1)$, and (2) represents the exponential moving average which drives future gradient updates based on previous ones. This will effectively reduce the oscillatory behaviour of gradient descent thereby reducing computational costs. In our project, we utilized both L1 regularization and momentum-based learning rates to automate feature selection and as reduce the number of iterations required to converge to the optimal weights, respectively.

Comparatively, naive Bayes is a method of generative classification which seeks to learn the joint distribution of features. The model assumes that the features are conditionally independent of the class (i.e. $x_i \perp\!\!\!\perp x_j|y$), which allows us to reduce the number of model parameters to the number of features. Without this assumption, calculating the joint distribution would require 2^D parameters, where D is the number of features. Therefore, the naive Bayes assumption is especially useful for data sets with high dimensionality but may perform poorly on data sets that violate this assumption. Given a datapoint, $X = (x_1 \cdots x_D)$ with D features, we choose its label $y \in \{0,1\}$ based on the following equation:

$$y = argmax_{y\in\{0,1\}}\left(\log P(Y=y) + \sum_{j=1}^{D} P(X_j = x_j|Y=y)\right)$$

where the first term is the prior and the second summand term is known as the likelihood term. There are two types of parameters the naive Bayes model needs to learn. The first type is the maximum likelihood estimator of the prior probabilities of each class, given by $\frac{N_c}{N}$, where N_c is the number of data points in class c and N is the total number of points. Since we are doing binary classification, we assumed that the probability of a given class follows a Bernoulli distribution. The second type of parameter is the maximum conditional likelihood estimator which is learned for each feature and differs depending on the distribution of a given feature. Bernoulli naive Bayes is appropriate for one-hot encoded categorical or binary features where $w_{c,d} = \frac{N_{y=c,x_d=1}}{N_{y=c}}$. For continuous features, Gaussian naive Bayes is more appropriate and requires two parameters per feature: $\mu_{d,y} = \frac{1}{N_c}\sum_{n=1}^{N} x_d^{(n)} y_c^{(n)}$ and $\sigma_{d,y} = \frac{1}{N_c}\sum_{n=1}^{N} y_c^{(n)}(x_d^{(n)} - \mu_{d,y})^2$. It is necessary to calculate the parameters differently depending on the feature type because while we can assume the categorical features follow a binomial distribution, or a Bernoulli distribution when using one-hot encoding, we can also assume that continuous features will obey a Gaussian distribution. By collecting these parameters, we can compute the posterior likelihood of observing a class given a set of input features and classify data points according to the maximal likelihood between the two classes.

Lastly, k-fold cross validation is a resampling method used to estimate the test error rate. K-fold is preferred over leave-one-out cross validation (LOOCV) because it often gives more accurate estimates and lower variance (Witten, et al. 2017). Also, this method is preferred over the train-validate-test split approach for two main reasons. Firstly, k-fold cross validation is less likely to overestimate the test error as it allows our model to be trained on a larger fraction of the data set thereby reducing bias. Secondly, k-fold cross validation reduces the variance of our estimation of the testing error.

Task Description

In this project, we explored the performance of full-batch gradient descent logistic regression with L1 regularization and momentum based learning rates as well as a mixed Bernoulli-Gaussian naive Bayes model. The goal was to successfully implement both models, find the optimal hyper-parameters of logistic regression using L1 regularization and momentum based learning rates, and compare the performance and run-time of the two algorithms. In addition, we compared the convergence behaviour of gradient descent when using a constant learning rate versus a momentum-based one. Finally, we investigated the effect of standardizing the data of continuous features, as well as removing features that were highly correlated.

Data Sets

Four data sets were used for training and testing these algorithms: Adult, Banknote Authentication, Ionosphere, and Breast Cancer data sets. All the data sets were first processed by removing the rows with invalid entries, such as missing values or illogical entries such as question marks.

The Adult data set is drawn from U.S. census data that comprises six categorical and eight integer attributes with 48842 data points. We removed 6,465 data points with missing values as part of the data prepossessing. Additionally, we found that education number and education are fully correlated due to the fact that education number is an integer representation of the education category. Therefore, we removed the education attribute to avoid overfitting our model. We also removed the factor levels "Never-worked" and "armed-forces" from the workclass and occupation attributes because these attributes were false in all data points. One-hot encoding was used to transform categorical variables into binary variables. The fnlwgt attribute, which illustrates the number of citizens the census believes a data point represents, was also removed because using those values to weigh the samples did not improve the accuracy in either of our models.

The UCI Banknote Authentication data set, which comprises five continuous variables describing images of banknotes. There are 1372 data points that are not missing values. According to the Pearson correlation test, there exists moderate correlation (-0.79) between skewness and variance, and between class and variance (-0.72). We decided to implement L1 regularization to deal with any potential multicolinearity rather than manually removing the attributes from the data set.

The Ionosphere data set includes 351 instances with 34 continuous attributes. Attribute a02 had value 0 for all instances, so we determined it was not useful in training the model. Attribute a15 was shown to be correlated with attribute a13 with a Pearson correlation coefficient greater than 0.8, so we also removed it from the data.

Finally, the Breast Cancer data set includes 286 instances with nine attributes. Nine instances were removed for missing or illogical entries before training the models. We did not make any changes to the attributes.

After we cleaned the data, we computed the summary statistics table for all four data sets. However, only the statistics table of the Adult data set stood out (see Table 5 in the appendix). In particular, the maximum values of capital-gain and hours-per-week are 99999 and 99, respectively, thus illustrating strong possibility that there may be values greater than 9999 and 99 that were capped at these maxima.

Results

Optimal Hyperparameters

Data set	Beta	Learning Rate	Lambda	Iterations
Adult	0.9	$6x10^{-3}$	0.0	1750
Banknote	0.9	$3x10^{-5}$	0.5	8000
Ionosphere	0.9	$3x10^{-5}$	0.5	10,000
Breast Cancer	0.9	$3x10^{-5}$	0.5	400

Table 1: The hyperparameters reported above were found by testing a wide array of the parameters and choosing the ones that gave the highest accuracy without greatly compromising runtime. Beta is the hyperparameter used for adding momentum to update our learning rates. Lambda is the hyperparameter of the L1 regularization. We also tuned the stopping criteria threshold (not shown), which was set to 1e-7 for all models.

5-Fold Cross-Validated Percentage Accuracies

Data set	Logistic Regression	Naïve Bayes
Adult	80.15	83.57
Banknote	95.66	83.82
Ionosphere	86.51	81.27
Breast Cancer	71.00	73.47

Table 2: The accuracies reported above are an average over 5-fold cross-validation on the training set (90% of the data points) after hyperparameter tuning.

Logistic Regression: 5-Fold Cross-Validated Training and Test Percentage Accuracies

Percentage of training set used	100%		85%		70%		65%	
Data set	Train	Test	Train	Test	Train	Test	Train	Test
Adult	77.63	81.04	78.65	83.86	84.29	83.99	85.35	83.83
Banknote	95.70	97.10	95.51	97.10	95.23	97.10	95.66	97.10
Ionosphere	85.79	82.86	86.45	85.71	86.93	82.86	87.93	80.00
Breast Cancer	71.50	78.57	70.89	78.57	70.42	78.57	73.02	78.57

Naïve Bayes: 5-Fold Cross-Validated Training and Test Percentage Accuracies

Percentage of training set used	100%		85%		70%		65%	
Data set	Train	Test	Train	Test	Train	Test	Train	Test
Adult	83.60	83.06	83.79	83.00	83.73	83.00	83.78	83.13
Banknote	83.74	83.33	83.06	83.33	83.14	83.33	82.75	84.78
Ionosphere	80.63	85.71	83.02	88.57	84.09	88.57	84.00	85.71
Breast Cancer	72.65	78.57	76.67	67.86	73.53	67.86	73.75	71.43

Tables 3 and 4: To generate the above tables, we set aside 10% of the data points in each data set to use as a test test. Using the other 90% of the data points, we computed the average 5-fold cross-validated accuracy on 100%, 85%, 70%, and 65% of the training set (reported in the Train columns). The Test columns report the average cross-validated accuracies on the Test sets.

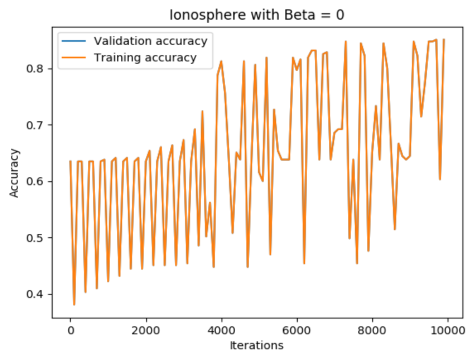

Figure 1

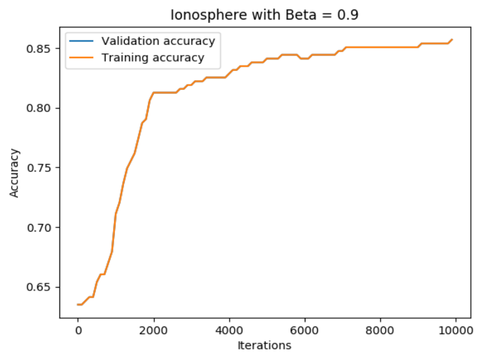

Figure 2

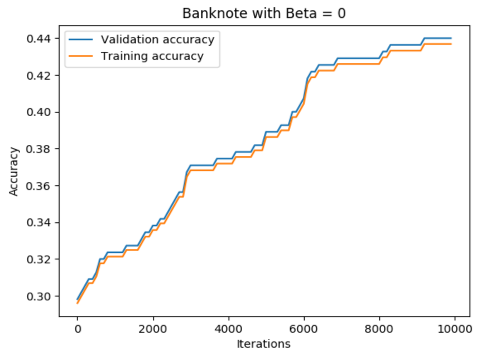

Figure 3

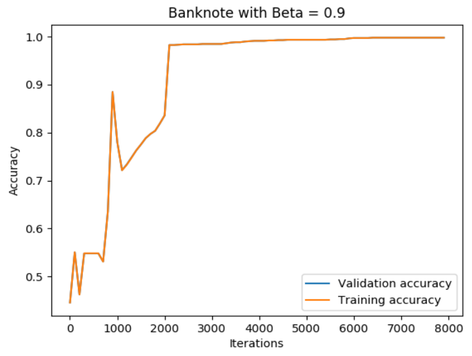

Figure 4

Discussion and Conclusions

Figures 1 to 4 illustrate the effect of incorporating momentum by setting $\beta = 0.9$ on the validation accuracy per iteration of gradient descent with all other hyperparameters fixed. In other words, these figures show us the effect of using a momentum-based learning rate on the trend of gradient descent convergence to the optimal weights. By comparing figures 1 and 2, we observe that injecting momentum has a dramatic effect when learning the Ionosphere data set. Without momentum, the accuracy has an increasing trend with oscillatory behaviour, preventing convergence to the global optimum in a reasonable number of iterations. However, once we include momentum, we observe that we eliminate the oscillatory behaviour, enabling our model to learn quicker. In addition, we found that including momentum when training the model on the Banknote data set allowed for quicker convergence, even when there was no initial oscillatory behaviour. This illustrates that momentum can improve the rate of convergence even in cases where oscillatory behaviour is absent. The trends of the accuracy per iteration of Adult and Breast Cancer data sets exhibit similar learning rate curves.

Furthermore, we noticed that using L1 regularization greatly improved the accuracy of our logistic regression model because it aided in feature selection. This was especially useful in the Ionosphere data set because it set the weight of feature a09 to 3.1e-06, effectively eliminating a redundant feature.

The Adult data set initially contained weights corresponding to each data point to indicate the number of people each instance represented. However, in our implementation of naive Bayes, we decided against including the weights of the data points because it did not improve the model accuracy. This was verified using the naive Bayes algorithm implemented in Scikit-Learn.

It is also evident that the naive Bayes did not perform as well on the Breast Cancer data set compared to the other data sets. This is likely because the naive Bayes assumption that the features are conditionally independent given the class is likely not true of the Breast Cancer features. For example, it is possible that parameters such as the degree of malignancy, tumour size, and the number of invasive nodes could be dependent on one another. If the naive Bayes assumption were significantly incorrect, this would make the model a poor fit for this data set.

To assess the final model, we computed the accuracy using the test set, which was not used in hyperparameter tuning, training, or cross validation. We also reported the cross-validated training accuracy. Both of these are shown in Tables 3 and 4. We found that the testing accuracy was generally higher than the cross-validated training accuracy. This may be because the test sets were smaller than the validation sets, as we see a gradual decrease in the difference between the Train and Test columns as the percentage of the training set used to train the model decreases from 100 to 65%.

Across the four data sets, we found that in practice, the logistic regression model took longer to train than the naive Bayes, especially for the Adult data set which had the most data points and features. This is expected because we ran k-fold cross-validation with full-batch gradient descent in the logistic regression algorithm, yielding a runtime of $\mathcal{O}(KNDI)$, where I is the predefined maximum number of iterations for gradient. In contrast, calculating the maximal likelihood of naive Bayes has a closed-form solution that can be found in $\mathcal{O}(ND)$ time, making it much faster than many machine learning algorithms including logistic regression.

Additionally, we see that in general, the naive Bayes model performs better for small data sets, while logistic regression has better performance on large data sets. This is exhibited by the higher accuracy achieved by our logistic regression model on the Banknote data set, which has a high number of data points (Tables 2, 3, and 4). Conversely, we see that the naive Bayes model has higher testing accuracy than the logistic regression model for the Ionosphere data set, which has relatively few data points (Tables 3 and 4).

Although we achieved high accuracies with our models, we believe that these models could be further improved by implementing additional strategies. For the gradient descent used to find the optimal weights in logistic regression, we could have used a decaying learning rate satisfying the Robbins Monro condition to increase rate or the likelihood of convergence. Alternatively, Adagrad would allow for a different learning rate for each parameter, which may improve the logistic regression model accuracy. There are also algorithms such as semi-Naive Bayes, which account for the dependence of input features (Fei Zheng and Geoffrey I. Webb, 2005). Such algorithms could improve the naive Bayes model accuracy on data sets with dependent features such as Breast Cancer.

Future Work

During the experiment, we found that the performance of logistic regression largely depended on the selected hyperparameters. Manually adjusting the hyperparameters is very time-consuming, and even if done precisely, lacks theoretical justification which naturally leads to a ”trial-and-error” setting. One could try implementing an automatic algorithm that adjusts the learning rate and momentum proposed by Lancewicki and Kopru (2019). Furthermore, several additional learning rates could be explored such as the Adaptive learning rate, RMS prop, or a sequence of learning rates satisfying the Robbins-Monro criteria. The rate of convergence to the optimal weights could be explored by creating a cross-entropy vs. iteration plot showing the convergence trends of these learning rates across all four data sets. (Mention how we can learn a nonlinear transformation for data that is not linearly separable) Another future direction this project could go in is generalizing our algorithms to work for problems with categorical or continuous data, rather than only for binary classification problems.

Statement of Contributions

Data pre-processing and hyperparameter tuning was done by Andrew Cheng and Chris Rafiaa. Logistic regression was implemented by Andrew Cheng and Naive Bayes by Stephanie Li. The experiments and the report write up was completed by all three members.

Citations

James, G., Witten, D., Hastie, T., Tibshirani, R. (2017). An introduction to statistical learning: with applications in R.

Lancewicki, Tomer & Kopru, Selcuk. (2019). Automatic and Simultaneous Adjustment of Learning Rate and Momentum for Stochastic Gradient Descent.

Zheng, F., Webb, G. (2005). A comparative study of Semi-naive Bayes methods in classification learning. In S. J. Simoff, G. J. Williams, J. Galloway, & I. Kolyshkina (Eds.), Proceedings of the 4th Australasian Data Mining Conference (pp. 141 - 156). Sydney NSW Australia: University of Technology Sydney.

Appendix

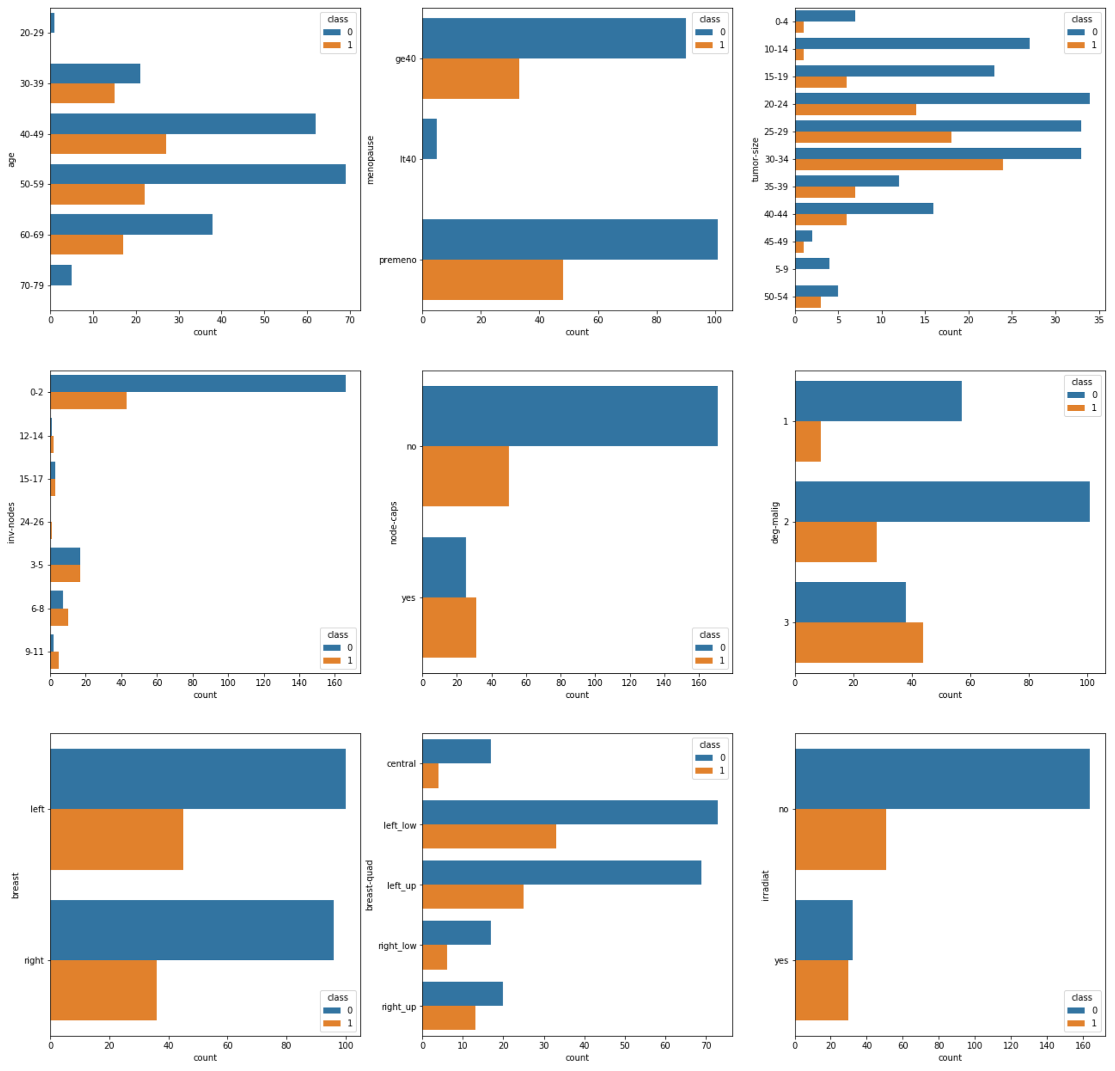

Figure 5: Distribution of the Breast Cancer data set

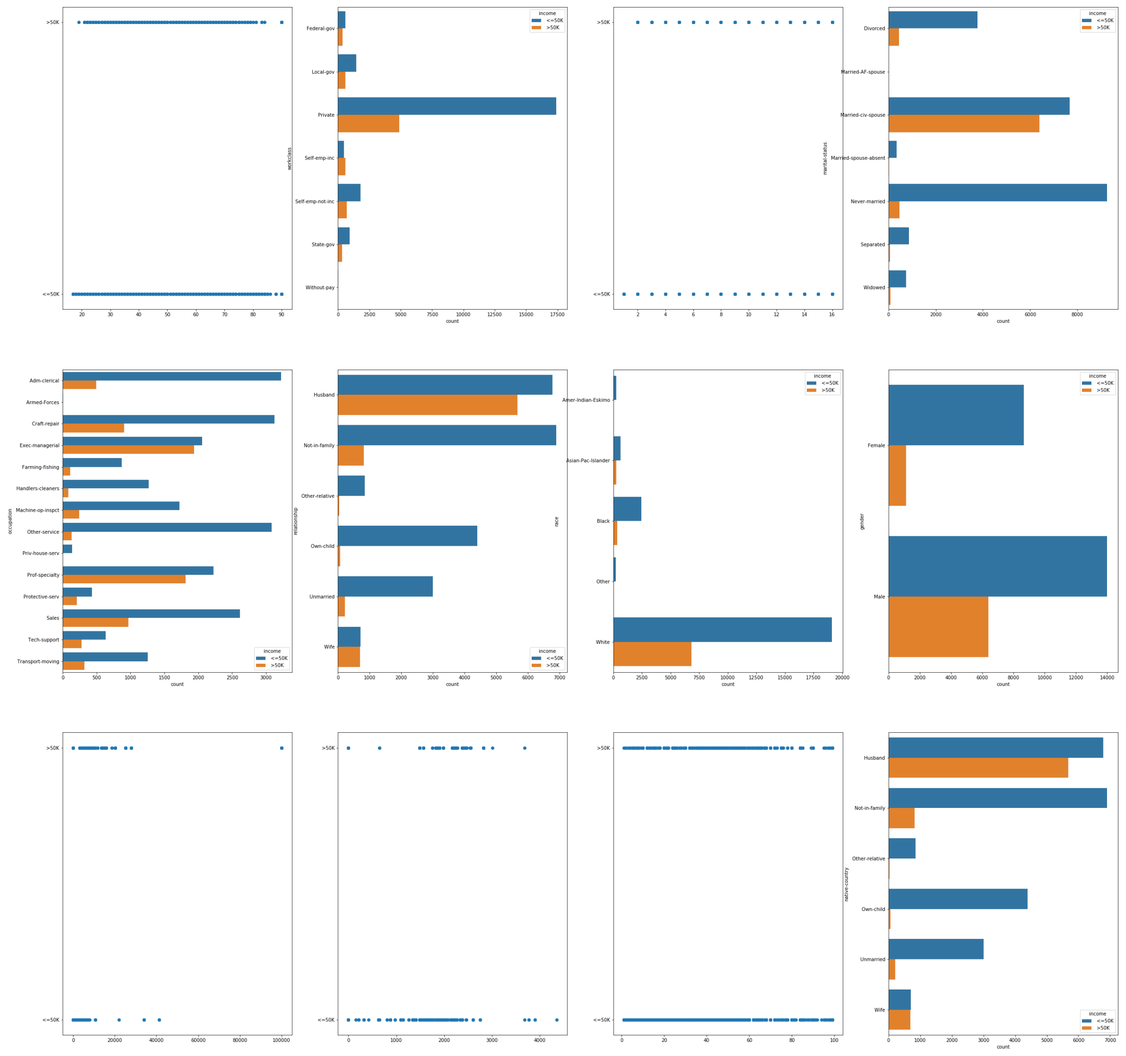

Figure 6: Distribution of the Adult data set

	age	educational-num	capital-gain	capital-loss	hours-per-week
count	30162.000000	30162.000000	30162.000000	30162.000000	30162.000000
mean	38.437902	10.121312	1092.007858	88.372489	40.931238
std	13.134665	2.549995	7406.346497	404.298370	11.979984
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	47.000000	13.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	4356.000000	99.000000

Table 5: Statistical Summary of the Adult data set