CS559 Hw 2

I pledge my honor that I have abided by the Stevens Honor System

Dimitrios Haralampopoulos

2023-02-27

Please follow the below instructions when you submit the assignment.

- **Do not use any package/tool for implementing the algorithms; You can use packages for matrix/vector operations, data processing, or cross-validation**.
- You shall submit a zip file named Assignment2_LastName_FirstName.zip which contains:
  - a pdf file contains all your solutions for the written part
  - python files (jupyter notebook or .py files)

(20 points) Please download the "processed.cleveland.data" from Heart-disease data set in the UCI Machine Learning repository and implement a binary Fisher's Lin- ear Discriminant Analysis to distinguish no-heart disease (0) from heart disease(1 – 4) and report your results. Please read "heart-disease.names" for the explanation of features (13 features are used). Split data into training (80%) and test (20%). Write down each step of your solution. You need to choose a decision boundary and classify the test samples based on the decision boundary you learned from the training data. Please report the data distributions (e.g., how many samples are no-heart disease and how many are heart disease). Then report your results on the accuracy, recall, precision, and F1 (assuming heart disease samples are positive samples) on the test data and plot the projected test samples using your learned w.

(50 points) Please download the breast cancer data set from UCI Machine Learning repository. You can either use "breast-cancer-wisconsin.data" or "wdbc.data". Please check their corresponding ".names" files for the explanation of features and labels.

1. (10 pts) Show that the derivative of the error function in Logistic Regression with respect to $\mathbf{w}$ is:

$$\nabla_{\mathbf{w}} E(\mathbf{w}) = \sum_{n=1}^{N} (f(\mathbf{x}_n) - y_n)\mathbf{x}_n$$

$$p(\mathcal{C}_1|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi)$$

$$\frac{d\sigma}{da} = \sigma(1-\sigma)$$

$$p(\mathbf{t}|w) = \prod_{n=1}^{N} y_n^{t_n}\{1-y_n\}^{1-t_n}$$

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|w) = -\sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} \rightarrow$$

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\{t_n \ln \sigma(\mathbf{w}^T\phi_n) + (1-t_n)\ln(1-\sigma(\mathbf{w}^T\phi_n))\}$$

$$\frac{\partial}{\partial \mathbf{w}}E(\mathbf{w}) = \frac{\partial}{\partial \mathbf{w}}\left[-\sum_{n=1}^{N}\{t_n \ln \sigma(\mathbf{w}^T\phi_n) + (1-t_n)\ln(1-\sigma(\mathbf{w}^T\phi_n))\}\right] =$$

$$-\sum_{n=1}^{N}\{t_n \frac{\partial}{\partial \mathbf{w}} \ln \sigma(\mathbf{w}^T\phi_n) + (1-t_n)\frac{\partial}{\partial \mathbf{w}}\ln(1-\sigma(\mathbf{w}^T\phi_n))\}$$

Solve Derivatives separately to obtain: $\frac{\partial}{\partial \mathbf{w}} \ln \sigma(\mathbf{w}^T\phi_n) = \frac{1}{\sigma(\mathbf{w}^T\phi)}\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n)$ and $\frac{\partial}{\partial \mathbf{w}} \ln(1-\sigma(\mathbf{w}^T\phi_n)) = \frac{1}{1-\sigma(\mathbf{w}^T\phi_n)}\frac{\partial}{\partial \mathbf{w}}(1-\sigma(\mathbf{w}^T\phi_n)) = \frac{-1}{1-\sigma(\mathbf{w}^T\phi_n)}\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n)$ Plug back in to get:

$$-\sum_{n=1}^{N} t_n \frac{1}{\sigma(\mathbf{w}^T\phi_n)}\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n) + (1-t_n)\frac{-1}{1-\sigma(\mathbf{w}^T\phi_n)}\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n)$$

$$-\sum_{n=1}^{N}\left[\frac{t_n}{\sigma(\mathbf{w}^T\phi_n)} - \frac{(1-t_n)}{1-\sigma(\mathbf{w}^T\phi_n)}\right]\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n)$$

Using the chain rule we get:

$$\frac{\partial}{\partial \mathbf{w}}\sigma(\mathbf{w}^T\phi_n) = \frac{\partial}{\partial a_n}\sigma(a_n) * \frac{\partial}{\partial \mathbf{w}}\mathbf{w}^T\phi_n = [\sigma(a_n) * [1-\sigma(a_n)]] * [\phi_n]$$

Since $\frac{\partial}{\partial \mathbf{w}}[\sigma(a_n)] = \frac{\partial}{\partial \mathbf{w}}[\mathbf{w}^T\phi_n] = \phi_n$, we get:

$$\frac{\partial}{\partial \mathbf{w}}E(\mathbf{w}) = -\sum_{n=1}^{N}\left[\frac{t_n}{\sigma(\mathbf{w}^T\phi_n)} - \frac{(1-t_n)}{1-\sigma(\mathbf{w}^T\phi_n)}\right]\sigma(a_n)*[1-\sigma(a_n)]*\phi_n \rightarrow -\sum_{n=1}^{N}[t_n(1-\sigma(a_n)) - (1-t_n)(\sigma(a_n))]*\phi_n \rightarrow$$

$$-\sum_{n=1}^{N}[t_n(1-y_n) - (1-t_n)(y_n)] * \phi_n \rightarrow -\sum_{n=1}^{N}[t_n - t_n y_n - y_n + t_n y_n)] * \phi_n \rightarrow \sum_{n=1}^{N}[y_n - t_n] * \phi_n$$

This is, however, by the textbook. In the lecture notes, we treat y_n as $f(\mathbf{x}_n)$, t_n as y_n, and ϕ_n as $\mathbf{x}_n$, giving us the form:

$$\nabla_{\mathbf{w}}E(\mathbf{w}) = \sum_{n=1}^{N}[f(\mathbf{x}_n) - y_n]\mathbf{x}_n$$

2. (20 pts) Implement a logistic regression classifier with maximum likelihood (ML) estimator using Stochastic gradient descent and Mini-Batch gradient descent algorithms. Divide the data into training and test. Choose a proper learning rate. Use cross-validation on the training data to choose the best model and report the recall, precision, and accuracy on malignant class prediction (class label malignant is positive) on the test data using the best model. Write down each step of your solution.

3. (20 pts) Implement a probabilistic generative model (the one in our lecture) for this problem. Use cross-validation on the training data and report the recall, precision, and accuracy on malignant class prediction (class label malignant is positive) on the test data using the best model. Write down each step of your solution.

(20 points) From Project Gutenberg, we downloaded two files: The Adventures of Sherlock Holmes by Arthur Conan Doyle (pg1661.txt) and The Complete Works of Jane Austen (pg31100.txt). Please develop a multinomial Naive Bayes Classifier that will learn to classify the authors from a snippet of text into: Conan Doyle or Jane Austen. A multinomial Naive Bayes uses a feature vector $\mathbf{x} = \{x_1, ..., x_D\}$ as a histogram and model the posterior probability as:

$$p(C_k|\mathbf{x}) \propto p(C_k) \prod_{i=1}^{D} p(x_i|C_k)$$

where $p(x_i|C_k)$ can be estimated by the number of times word i was observed in class C_k plus a smoothing factor divided by the total number of words in C_k

In the test phase, given a new example $\mathbf{x}_t$, you can output the class assignment for this example by comparing $\log p(C_1|\mathbf{x}_t)$ and $\log p(C_2|\mathbf{x}_t)$. If $\log p(C_2|\mathbf{x}_t) > \log p(C_1|\mathbf{x}_t)$, assign C_2 to this example. You need to divide the data into training and test. For the words that appear in the test set but not in the training set, you can either ignore these words in the probability calculation or you can apply smoothing in $p(x_i|C_k)$ (e.g. Laplace smoothing).

You can apply some preprocesssing techniques such as removing stop-words and punctuation. You can remove the unrelated text in the beginning of each file. Make sure the test data has equal number of samples from Conan Doyle and Jane Austen.

Report accuracy on test data using your Naive Bayes classifier.

(10 points) Please prove that 1) the multinomial naive Bayes classifier in log-space essentially translates to a linear classifier. 2) Logistic regression is a linear classifier.

Multinomial Naive Bayes is defined as:

$$p(\mathcal{C}_k|\mathbf{x}) \propto p(\mathcal{C}_k) \prod_{i=1}^{D} p(\mathbf{x}_i|\mathcal{C}_k)$$

where each $p(\mathbf{x}_i|\mathcal{C})$ is of multinomial distribution (ex. Gaussian) and $\mathbf{x}_i$ is each feature of data $\mathbf{x}$.

$$H^{MAP} = \underset{i\epsilon\{+,-\}}{argmax}\ p(\mathcal{C}_k) \prod_{i=1}^{D} p(\mathbf{x}_i|\mathcal{C}_k)$$

We apply the natural logarithm to this to get:

$$H^{MAP} = \underset{i\epsilon\{+,-\}}{argmax}\ \ln\left(p(\mathcal{C}_k) \prod_{i=1}^{D} p(\mathbf{x}_i|\mathcal{C}_k)\right) \rightarrow \underset{i\epsilon\{+,-\}}{argmax}\ \ln p(\mathcal{C}_k) + \ln\left[\prod_{i=1}^{D} p(\mathbf{x}_i|\mathcal{C}_k)\right]$$

Using log properties we can say:

$$H^{MAP} = \underset{i\epsilon\{+,-\}}{argmax}\ \ln p(\mathcal{C}_k) + \left[\sum_{i=1}^{D} \ln p(\mathbf{x}_i|\mathcal{C}_k)\right]$$

We can see from here that H^{MAP} is linear in log-space, as a sum of the natural logarithms of the prior probability and the sum of class-conditional densities. The probabilities are all scalars between 0 and 1, and the natural logarithm of each will also return a scalar value. The linear combination of scalar values is equivalent to a linear scalar, meaning that the multinomial naive Bayes classifier is essentially a linear classifier in log-space.

Given the Logistic Regression likelihood function:

$$\mathcal{L}(\mathbf{w}) = \prod_{n=1}^{N} p(\mathcal{C}_1|\mathbf{x}_n)^{y_n} (1 - p(\mathcal{C}_1|\mathbf{x}_n))^{1-y_n}$$

where

$$p(\mathcal{C}|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + \omega_0) = f(\mathbf{x})$$

and

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

We look at the logistic sigmoid function's input, $a = \mathbf{w}^T\mathbf{x} + \omega_0$. The decision boundary for the logistic sigmoid function is usually $\mathcal{C}_1 < 0.5$ and $\mathcal{C}_2 > 0.5$, and that can be shown mathematically by setting our input as close to 0 as possible. We can see that if we graph the logistic sigmoid function, we have an x-intercept at $y = 0.5$ and the distinctive curve of the sigmoid flattens out the closer a gets to 0. Because of this feature of the logistic sigmoid function, we can essentially use it as a linear classifier with a decision boundary of 0.5.