CSE 575 Statistical Machine Learning

Project Part 1

Saianirud Reddy Nellore | snellore@asu.edu | ASU ID: 1219495602

The given dataset, Fashion-MNIST, contains a training set of size 12000 and a testing set of size 2000. The training set contains 6000 samples of T-shirts and 6000 samples of trousers. In the same way, the testing set contains 1000 samples of T-shirts and 1000 samples of trousers.

The data is given in the form of a dictionary. So, we loaded the data using SciPy and stored it in the following variables.

trX– samples in the training dataset

trY – labels of corresponding samples in the training dataset.

tsX – samples in the testing dataset

tsY – labels of corresponding samples in the testing dataset.

Feature Extraction:

Each sample is an image of size 28*28 = 784 pixels. i.e., each row in the trX contains 784 values. Instead of using the 784 values as features, we reduce them down to 2 features by extracting the mean and standard deviation of each sample.

Mean:

$$\mu = \sum_{i=1}^{n} x_i / n$$

Standard Deviation:

$$\sigma = \sqrt{\sum_{i=1}^{n} \frac{(x_i - \mu)^2}{n}}$$

where n represents the number of values and x_i represents the i^{th} value in each sample.

NumPy is used to calculate these values directly in the form of a matrix.

The values for the features in trX:

1. $Feature1\ (\mu) = [0.42315926 \quad 0.14336735 \quad \dots \quad 0.1890006 \quad 0.21389056]$
2. $Feature2\ (\sigma) = [0.39541744 \quad 0.19489707 \quad \dots \quad 0.32555838 \quad 0.36485942]$

where 1st column represents 1st image, 2nd column represents 2nd image, and so on.

In the same way, 2 features are extracted for each image in the testing data.

Using the two extracted features, mean and standard deviation, from the training data we are going to train the model using Naïve Bayes classification and Logistic regression and perform classification on testing data.

Estimating the parameters:

We assume that the two features are independent, and is drawn from a 2D normal distribution. So, each feature represents a 1D normal distribution. We separate the features according to the class and use the above formulas for mean and standard deviation to estimate the parameters for each feature within each class, where n represents the number of values and x_i represents the i^{th} value in each feature in each class. There are 2 classes in the problem, Class 0 and Class 1.

Estimated values of the parameters for Class 0:

$$\begin{aligned} & & feature1 & \quad feature2 \\ Mean\ (\mu) & = [0.32560777 & 0.32003609] \\ Standard\ Deviation\ (\sigma) & = [0.11337491 & 0.08798281] \end{aligned}$$

Estimated values of the parameters for Class 1:

$$\begin{aligned} & & feature1 & \quad feature2 \\ Mean\ (\mu) & = [0.22290531 & 0.33394171] \\ Standard\ Deviation\ (\sigma) & = [0.05695101 & 0.05703229] \end{aligned}$$

Naïve Bayes:

In Naïve Bayes classification, we assume that the two features are independent of each other i.e.,

$p(x/y) = p(x_1, x_2, \dots, x_d/y) = \prod_{i=1}^{d} p(x_i/y)$, where d is the number of features.

To classify the sample as Class 0 or Class 1, we need to get the probability $p(y/x)$ for each class. From bayes theorem,

$p(y/x) = \frac{p(x/y) * p(y)}{p(x)} \propto p(x/y) * p(y)$ as $p(x)$ will be same for $y = 0$ and $y = 1$ for a given sample and does not affect $p(y/x)$.

As $p(y/x) \propto p(x/y) * p(y)$, the sample will be classified into the class with high $p(x/y) * p(y)$ value.

Gaussian distribution is given by,

$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$, where μ is the mean and σ is the standard deviation of the values of x.

How the distributions are used in classifying a testing sample:

We will have two distributions in each class, one for each feature. We assume that the features are independent of each other and calculate the following probabilities for each class.

$p(x_i/y = k) = \frac{1}{\sqrt{2\pi}\,\sigma_{ik}} e^{-\frac{1}{2}\left(\frac{x_i-\mu_{ik}}{\sigma_{ik}}\right)^2}$ for each feature, where x_i represents i[th] feature and μ_{ik}, σ_{ik} represents mean and standard deviation of i[th] feature in class k.

$$p(y = k) = \frac{number\ of\ samples\ labeled\ class\ k}{total\ number\ of\ samples}$$

The predicted label in Naïve Bayes classifier:

$\hat{y} = argmax_y\, p(y) \prod_{i=1}^{d} p(x_i/y)$, where d is the number of features in testing data.

$$Accuracy = \frac{number\ of\ labels\ classified\ correctly}{total\ number\ of\ labels}$$

Accuracy of Naïve Bayes classification,

Accuracy overall is 83.15%

Accuracy for class $y = 0$ is 78.4%

Accuracy for class $y = 1$ is 87.9%

Logistic Regression:

In logistic regression, we assume that the data is linearly separable. The model is represented by the dot product of the weighted vector w and the feature vector x.

$w = (w_0, w_1, w_2, \dots\dots, w_d)$, $x = (1, x_1, x_2, \dots\dots, x_d)$, where w is the weighted vector, x is the feature set of each sample and d is the number of features.

We add 1 as an extra feature in the feature set to get the intercept w_0 after the dot product.

Dot product = $w_0 + \sum_{i=1}^{d} w_i x_i = w^T x$, where w^T is the transpose of matrix w.

Here, we directly model $p(y/x)$ and assume that $p(y/x)$ takes the form of a sigmoid function for better understanding. We pass the model $w^T x$ to the sigmoid function, which gives us the probability $p(y/x)$.

Sigmoid function:

$$\sigma(t) = \frac{1}{1+e^{-t}}, \text{ where } t = w^T x$$

$$p(y = 0/x) = \frac{1}{1+e^{w^T x}}, p(y = 1/x) = \frac{e^{w^T X}}{1+e^{w^T x}}$$

So, $p(y/x) = (\sigma(w^T x))^y \left(1 - \sigma(w^T x)\right)^{1-y}$, where $y = 0 \; or \; 1$

$$L(w) = p(y^1, y^2, \dots, y^n / x^1, x^2, \dots, x^n, w) = \prod_{i=1}^{n} p(y^i / x^i, w)$$

$$l(w) = logL(w) = \sum_{i=1}^{n} (\sigma(w^T x^i))^{y^i} \left(1 - \sigma(w^T x^i)\right)^{1-y^i}$$

$$= \sum_{i=1}^{n} y^i \log\left(\sigma(w^T x^i)\right) + (1 - y^i) \, log\left(1 - \sigma(w^T x^i)\right)$$

$$= \sum_{i=1}^{n} y^i (\mathrm{w^T} x^i) - \log\left(1 + \mathrm{e^{w^T} x^i}\right), \text{where n is the number of samples}$$

$$\widehat{w} = argmax_w \, l(w)$$

We use the gradient ascent method to get the w parameters for better accuracy.

Gradient Ascent:

In this method, we assume w to be all zeroes initially and gradually increase the w by adding some gradient to it until it converges.

$w = w + \eta \nabla_w l(w)$, where η (> 0) is called the learning rate and $\nabla_w l(w)$ is the differentiation of $l(w)$ with respect to w.

$$\nabla_w l(w) = \sum_{i=1}^{n} x^i (y^i - \sigma(w^T x^i))$$

First, we calculate the error in the predicted value, then calculate the gradient which is the dot product of the error vector and the feature vector, and then update the w vector by adding it with the product of learning rate and the gradient. Repeat the procedure until w vector converges.

Once the model is trained, we can get the probability for each sample of testing data by taking the dot product of w vector and the feature vector and passing it to the sigmoid function.

It will be classified as Class 0 if the probability is less than 0.5, otherwise, it will be classified as Class 1.

Taking learning rate = 0.01 and iterations = 10000, we get,

$w = [-17.98540149 \quad -219.28955939 \quad 232.38002755]$

$$Accuracy = \frac{number\ of\ labels\ classified\ correctly}{total\ number\ of\ labels}$$

Accuracy for Logistic Regression:

Accuracy overall: 92.25%

Accuracy for class $y = 0$: 92.8%

Accuracy for class $y = 1$: 91.7%