KOÇ UNIVERSITY

DASC521: Introduction to Machine Learning

Fall 2021

Instructor: Assc. Prof. Mehmet Gönen

Homework 02- Naïve Bayes' Classifier

Gamze Keçibaş- 60211

30.10.2021

There are two main parts in the project as parameter estimation by Naïve Bayes' Classifier and check confusion matrices creating by training dataset and test dataset. During the project, numpy library is preferred for data processing because it is much faster than pandas library and it allows for mathematical operations. Firstly, required libraries, provided images and their labels are imported. There are 35000 samples in the provided set. First 30000 images in the set are selected for training and last 5000 images are used to test of the model. Shapes of using all sets are provided below:

```
Total set of images (35000, 784)
Total set of labels (35000,)
Training set shape:  (30000, 784)
Training label set shape:  (30000,)
Test set shape:  (5000, 784)
Test label set shape:  (5000,)
```

Figure 1: Size of using datasets

Train dataset is checked before the parameter estimation step. Understanding data is important to evaluate the results. Thus, first 20 clothes in the training set are plotted with additional step by reshape method.

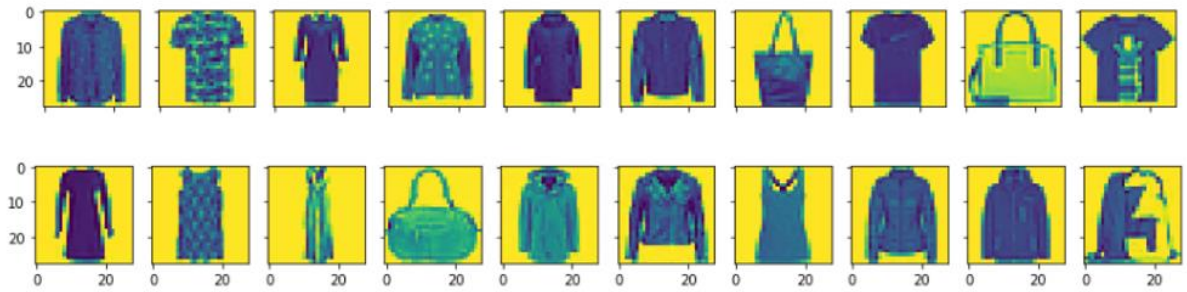

Figure 2: First 20 images in the training set

After data processing, means (Eq. 1), standard deviations (Eq. 2), and priors are calculated for each 5 classes. They are large arrays, so just first 10 and last 10 elements are printed. These three parameters are presented below where N is 30000, k is class label in the case. There are methods for mean and standard deviation in numpy library and they are preferred in the script.

$$\widehat{\mu_{k,l}} = \sum_{l=1}^{784} x_{k,l}/N \quad (1)$$

$$\widehat{\sigma_{k,l}} = \sum_{l=1}^{784} [x_{k,l} - \widehat{\mu_{k,l}}]/N \quad (2)$$

```
Sample priors

Prior probability of Class [1]:  0.2
Prior probability of Class [2]:  0.2
Prior probability of Class [3]:  0.2
Prior probability of Class [4]:  0.2
Prior probability of Class [5]:  0.2

Total probability=  1.0
```

Figure 3: Priors of 5 classes

```
Sample means

Size of sample means 5 x 784

Class [1]:
[254.99866667 254.98416667 254.85616667 254.66733333 254.54466667
 254.274      253.36283333 249.56366667 239.67583333 221.92416667]
...
[164.412      175.08283333 192.00516667 228.442      248.39266667
 253.13216667 254.2365     254.679      254.87816667 254.95933333]

Class [2]:
[254.99733333 254.99733333 254.9965     254.99416667 254.8705
 254.6405     254.08       252.97883333 249.87383333 233.35216667]
...
[187.578      218.94733333 241.22766667 250.99266667 253.73433333
 254.58233333 254.9045     254.96883333 254.99216667 254.98866667]

Class [3]:
[254.99933333 254.99933333 254.99233333 254.9765     254.87966667
 254.8475     254.7205     254.36316667 253.53116667 250.57233333]
...
[227.62333333 237.97533333 243.63633333 212.32866667 184.74433333
 199.76416667 233.05633333 251.52483333 254.4725     254.97483333]

Class [4]:
[254.99666667 254.98983333 254.91416667 254.69216667 254.18916667
 253.7985     252.88433333 249.064      241.3685     232.68933333]
...
[213.8055     227.44016667 235.85516667 232.03566667 221.2215
 229.0815     242.63233333 252.39516667 254.44166667 254.93666667]

Class [5]:
[254.999      254.98433333 254.93783333 254.7725     254.497
 254.20933333 254.032      253.79416667 253.663      253.40466667]
...
[218.43933333 219.97466667 222.59066667 226.64633333 232.916
 240.977      247.39783333 250.673      253.23333333 254.79083333]
```

Figure 4: Samples mean for each classes

```
Sample Standard Deviations

Size of sample standard deviations 5 x 784

Class [1]:
[ 0.09127736  0.25609108  1.31090756  3.80543465  5.27948907  6.97889132
 10.7720867  20.90887244 37.4438435  52.51224063]
...
[61.33922282 62.55887338 62.97645703 47.27240882 24.22176321 11.38112613
  7.69720086  5.29826629  3.9117332   1.93959091]

Class [2]:
[ 0.2065419   0.2065419   0.2163818   0.23050518  1.9811772   5.61972061
  8.94167769 14.1133643  21.42771372 41.32216288]
...
[64.07292654 53.12548879 34.92773255 17.69346243  9.93552991  4.41681121
  2.2767037   1.04076669  0.47057267  0.70062226]

Class [3]:
[ 0.05163547  0.04081939  0.16002465  0.21667429  2.82179374  2.85731408
  3.42870915  5.59427773 10.23928848 20.04369646]
...
[51.14423189 41.00799587 31.13997024 62.72873593 76.11316773 67.65612721
 43.2080528  18.43665868  6.7881694   1.1061344 ]

Class [4]:
[ 0.18436076  0.21617116  1.81046936  4.66455485  8.35111066 10.40547441
 13.11758189 22.03743566 34.98902267 45.99750159]
...
[52.18877596 46.80806647 41.9973871  44.40049994 56.68462553 49.48725618
 31.93485277 15.67799977  6.34549162  1.79971911]

Class [5]:
[ 0.04471018  0.64582342  3.03248555  4.68370335  7.48273508  9.21241081
 10.4029151  11.9006078  12.15466019 12.63741449]
...
[65.31681498 64.01413405 61.67456347 58.18092689 52.16745739 41.66458293
 30.85720708 23.62576428 13.9167006   4.4727787 ]
```

Figure 5: Samples standard deviations for each classes

End of the parameter estimation, score functions are designed by Naive Bayes' Classifier algorithm to calculate score functions.

$$g_c(x) = \sum_{i=1}^{N}[(-\frac{1}{2}\log(2\pi) - \log(\widehat{\mu_{i,c}}) - \frac{1}{2}\frac{(x_{i,c}-\widehat{\mu_{i,c}})^2}{2\widehat{\sigma_{i,c}}^2})] + \log \hat{P}(y = c) \quad (3)$$

When score functions are determined, maximum result is selected for labeling with argmax method in numpy library. Additionally, *safelog* function is defined to avoid gradient vanishing problem. The train dataset is used to evaluate the trained model. The confusion matrix is presented below. Pandas library is used to obtain confusion matrix:

Confusion Matrix of the Train Data:

y_truth	1.0	2.0	3.0	4.0	5.0
y_pred					
1	4436	583	16	1103	20
2	224	4035	173	74	96
3	123	775	4704	1867	33
4	971	574	933	2450	102
5	246	33	174	506	5749

Figure 6: Estimations with Train Dataset

Finally, the classifier model is tested by test dataset. It is confusion matrix is presented in Figure 7.

Confusion Matrix of the Test Data:

y_truth	1.0	2.0	3.0	4.0	5.0
y_pred					
1	736	91	0	199	3
2	45	711	23	13	18
3	19	112	814	289	4
4	143	79	135	416	20
5	57	7	28	83	955

Figure 7: Estimations with Test Dataset