

INDIAN INSTITUTE OF TECHNOLOGY DHARWAD



Lab 2

CS 412: Statistical Pattern Recognition

MEMBERS

GANESH SAMARTH

MANDEEP BAWA

SV PRAVEEN

Link to Code - [GitHub Repo bayes-classifiers](#)



Solution 1

MNIST Dataset

It is a dataset containing handwritten digits $\{0, 1, \dots, 9\}$ and their labels. Each image is in a 28×28 pixel format. It has a training set of 60,000 examples and a test set of 10,000 examples. The digits are size-normalized and centered in a fixed-size image.

Training Dataset -> Images(60000, 28, 28), Labels(60000,1)

Testing Dataset -> Images(10000, 28, 28), Labels(10000,1)

Distribution of classes in Training set

class_label	frequency	priors
0	5923	0.098717
1	6742	0.112367
2	5958	0.099300
3	6131	0.102183
4	5842	0.097367
5	5421	0.090350
6	5918	0.098633
7	6265	0.104417
8	5851	0.097517
9	5949	0.099150

Multivariate Gaussian Distribution

Multivariate Gaussian distribution is a generalization of the one dimensional normal distribution to higher dimensions. In this distribution, we would be able to capture the interrelations between the pixels of an image.

Here, we built a 784 dimensional gaussian distribution.

For a sample $\mathbf{x}(x_1, x_2, \dots, x_{784})$, its density is given by

$$p(\mathbf{x}; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp \left(-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right)$$

where μ is the mean vector and Σ is the correlation matrix for the class.

Preprocessing

Dataset is normalized to 0 mean and unit variance.

Unknown Parameters per class

Mean (μ), Variance (Σ)

We used Maximum Likelihood Estimates(MLE) to calculate unknown parameters. It will try to maximize the likelihood of the data given the parameters.

$$\max_{\theta} P(D/\theta)$$

where D represents the data and $\theta = (\mu, \Sigma)$

Maximum Likelihood Estimates

For each class i, we get the MLE estimates by using

$$\mu_i = E(X_i)$$

$$\Sigma_i = E((X_i - \mu_i) \cdot (X_i - \mu_i)^T)$$

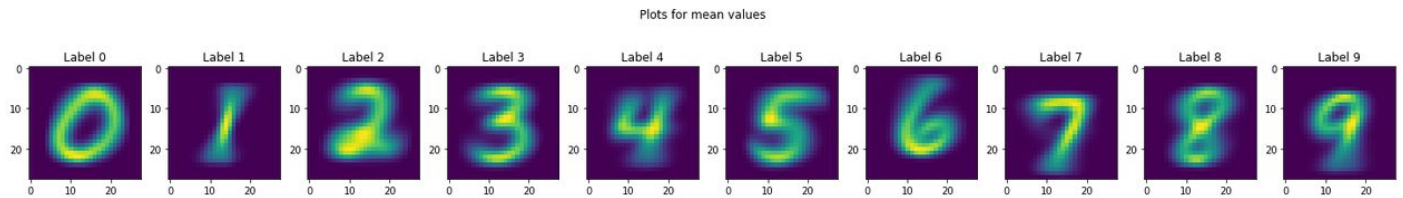
where X_i is the subset of X having label i

Since many of the diagonal entries in the covariance matrix can be zero due to sparse representation of digits. We regularized the covariance matrix so that it can be invertible

$$\Sigma_i = \Sigma_i + \text{diag}()$$

Estimated Means

After getting the MLE estimates, we can plot the mean values



Classification Step

For each example in the test set, we compare the posterior probabilities between the classes. The one having the maximum value is predicted as a label for that example.

$$label_{predicted} = \text{argmax}_i \text{prior}_i * \log(f_i(x))$$

where i take values from 0 to 9

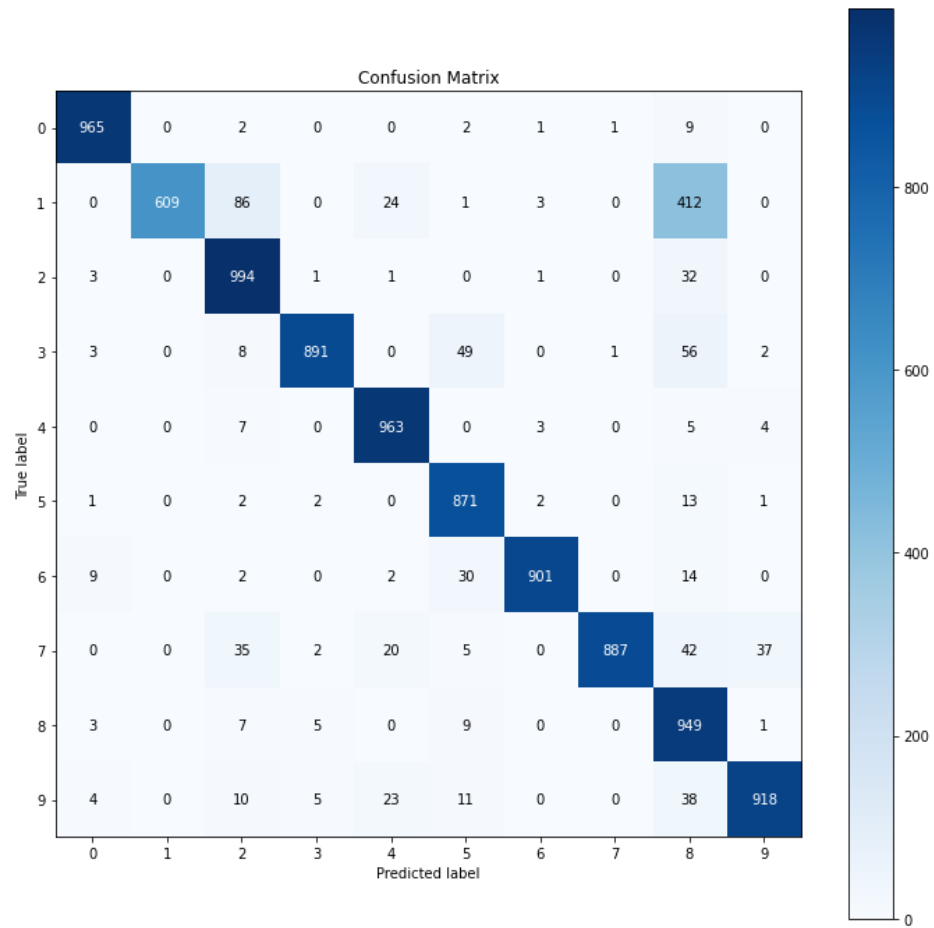
Results

Training dataset was used to model the class conditional densities. After modeling the densities, a Bayesian classifier was used to predict the labels for the test set.

Overall Accuracy

Accuracy on Test set = 89.48%

Confusion matrix



Classification Results per class

	precision	recall	f1-score	support
0	0.98	0.98	0.98	980
1	1.00	0.54	0.70	1135
2	0.86	0.96	0.91	1032
3	0.98	0.88	0.93	1010
4	0.93	0.98	0.96	982
5	0.89	0.98	0.93	892
6	0.99	0.94	0.96	958
7	1.00	0.86	0.93	1028
8	0.60	0.97	0.75	974
9	0.95	0.91	0.93	1009
accuracy			0.89	10000
macro avg	0.92	0.90	0.90	10000
weighted avg	0.92	0.89	0.89	10000

Inference

1. From the confusion matrix and also from the detailed results it is evident that our classifier misclassified 412 images as 8 which have true label 1
2. However, we can also see that we do not have any false positives for 1. So, we can conclude that our classifier has captured 1 correctly though not fully.
3. Comparing f-1 score between classes, we find it to be maximum for class 0. So, class 0 is modelled much better compared to other classes.
4. Overall accuracy is 89.48%, indicating that our classifier has been able to model the class conditional multivariate densities correctly.

Multivariate Exponential Distribution

Exponential distribution is the probability distribution of the time between events in the poisson point process.

Exponential distribution for the univariate variable is given by

$$f(x; \lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

Now, for modeling multivariate densities we assume that the density distribution of one pixel in an image is independent of the distributions of other pixels. So, the joint distribution of a particular class can be written as the product of the individual densities.

$$F(X, \lambda) = \prod_{i=1}^{784} f(x_i, \lambda_i)$$

Preprocessing

MinMax Scaling is used on all dataset to get values in range from 0 to 1, which is the requirement for exponential density estimation.

Unknown Parameters per class

Each class has a unknown parameter λ which is a 784-dimensional vector.

Maximum Likelihood Estimates

Suppose for class j , we have n_j entries in our dataset
Then λ for class j can be modelled as

$$\lambda_k = \frac{n_j}{\sum_{i=1}^{n_j} x_{ik} + \alpha}$$

where λ_k is the k^{th} dimension of λ , $\alpha > 0$ to eliminate the inf case

Classification Step

For each example in the test set, we compare the posterior probabilities between the classes. The one having the maximum value is predicted as a label for that example.

$$label_{predicted} = \operatorname{argmax}_i \text{prior}_i * \log(f_i(x))$$

where i take values from 0 to 9

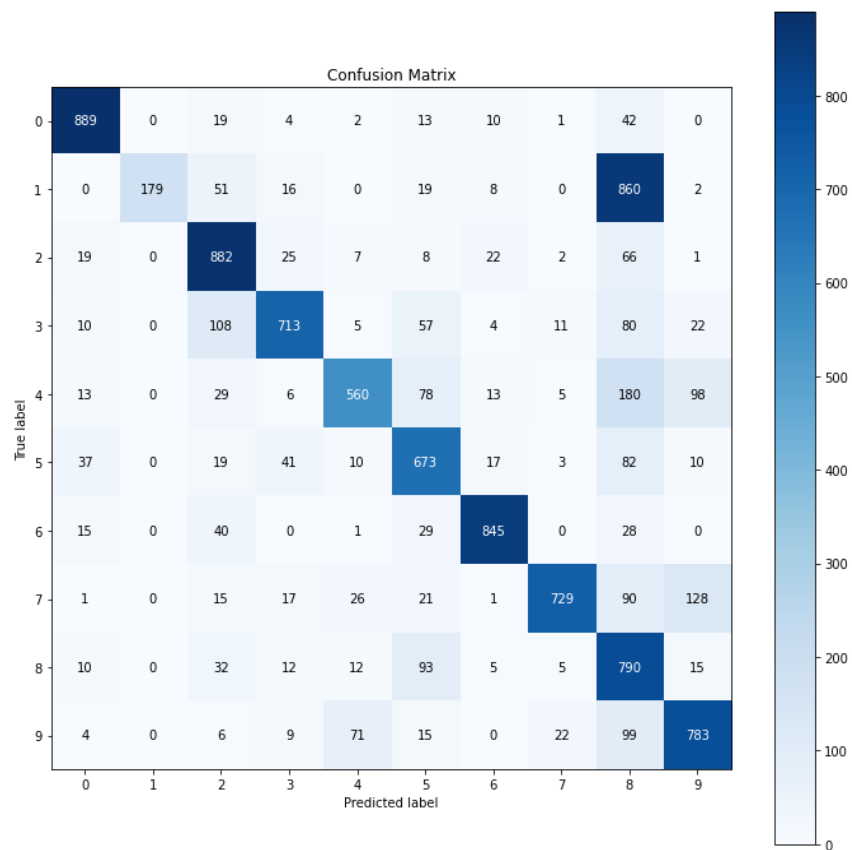
Results

Training dataset was used to model the class conditional densities. After modeling the densities, a Bayesian classifier was used to predict the labels for the test set.

Overall Accuracy

Accuracy on Test set = 70.43%

Confusion matrix



Classification Results per class

	precision	recall	f1-score	support
0	0.89	0.91	0.90	980
1	1.00	0.16	0.27	1135
2	0.73	0.85	0.79	1032
3	0.85	0.71	0.77	1010
4	0.81	0.57	0.67	982
5	0.67	0.75	0.71	892
6	0.91	0.88	0.90	958
7	0.94	0.71	0.81	1028
8	0.34	0.81	0.48	974
9	0.74	0.78	0.76	1009
accuracy			0.70	10000
macro avg	0.79	0.71	0.71	10000
weighted avg	0.79	0.70	0.70	10000

Inference

1. Here also, we can see that class 1 is misclassified as 8 but the error rate is much higher than the multivariate gaussian bayesian classifier.
2. As we assumed that densities within the class are independent, which was not the case in gaussian estimation. So, our classifier was not able to capture interrelation between the densities within the class.
3. Comparing f-1 scores between classes, we can find that class 0 and 6 have been modeled much better than other classes.
4. Overall accuracy is 70.43%, indicating that our classifier has been able to capture the class conditional multivariate densities.

Multinomial Naive Bayes Classifier for Newsgroup 20 Dataset

I. Dataset

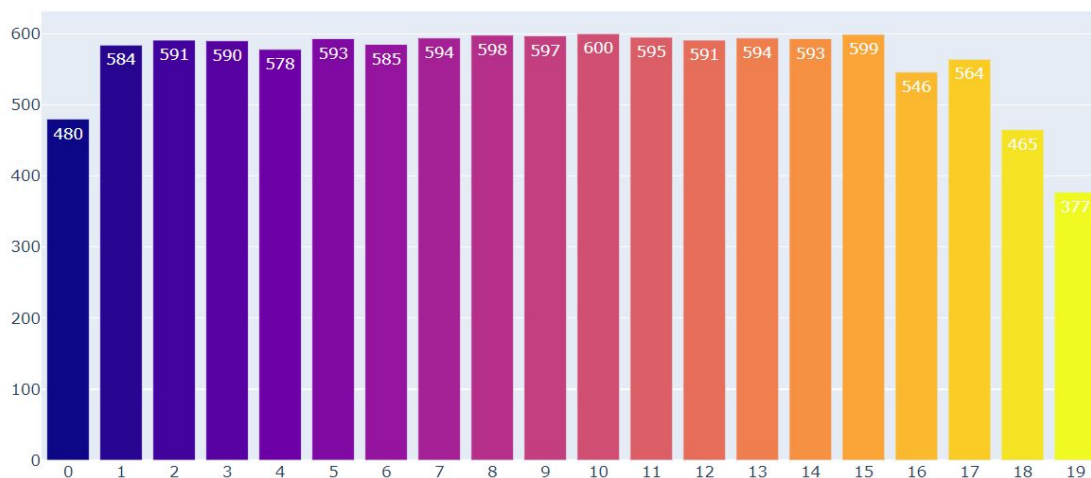
The 20 newsgroups dataset comprises 18846 newsgroups posts on 20 topics including -

0	alt.atheism	11	sci.crypt
1	comp.graphics	12	sci.electronics
2	comp.os.ms-windows.misc	13	sci.med
3	comp.sys.ibm.pc.hardware	14	sci.space
4	comp.sys.mac.hardware	15	soc.religion.christian
5	comp.windows.x	16	talk.politics.guns
6	misc.forsale	17	talk.politics.mideast
7	rec.autos	18	talk.politics.misc
8	rec.motorcycles	19	talk.religion.misc
9	rec.sport.baseball		

We split the dataset into a train-test split to validate the performance of our classifier.

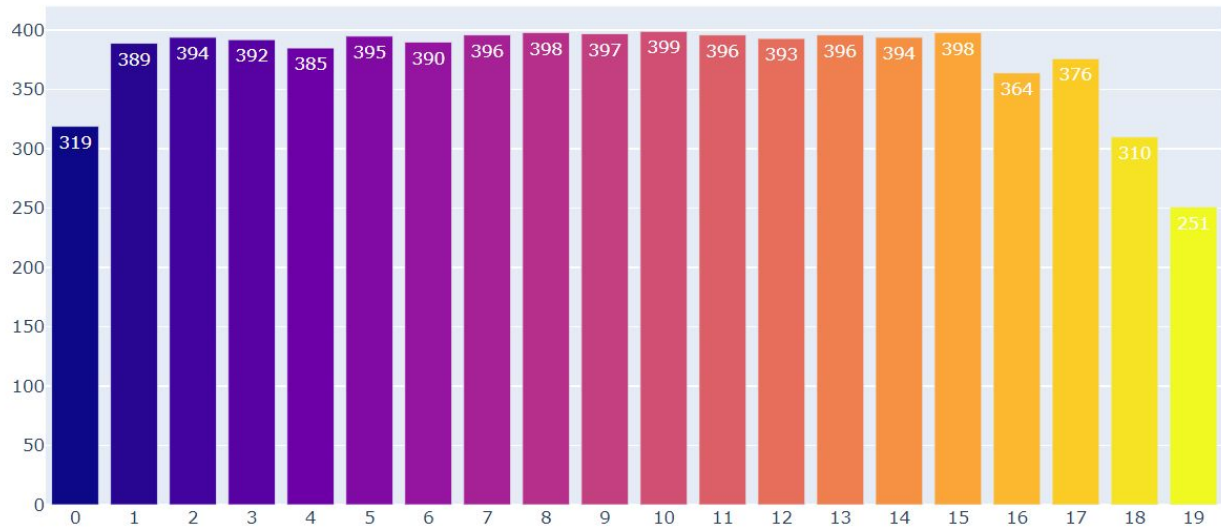
Train split: contains 11314 documents. The number of documents per class in this split is summarized by the figure below.

Number of documents per class (train split)



Test split: contains 7532 documents. The number of documents per class in this split is summarized by the figure below.

Number of documents per class (test split)



We can see that the splits are nearly balanced, although some classes such as class 1, 18 and 19 have fewer examples in total.

II. Pre-processing

Since we cannot feed text directly into the multinomial bayes classifier, the first step is to represent the document as a numerical feature vector. A popular representation is the Bag-of-words approach or the TF-IDF representation as described below. In both these approaches we assume that the position of a word does not matter.

1. Bag-of Words

A document is a string of text. In this algorithm, the first step is to tokenize a sentence by separating on white spaces or punctuation. We then generate a vocabulary over all the documents and assign an index to each word. Next, given a document we count the number of occurrences for each word in it and assign that number to the corresponding word index in the feature vector. If a word doesn't occur then the corresponding entry would be 0.

Example: "The man told the other man to come in."

The	man	told	other	to	come	in
2	2	1	1	1	1	1

*assuming vocabulary consists of only words in this document.

2. TF-IDF

Often words like "the", "a", etc. add little extra meaning to a document, but their large frequencies of occurrence may shadow the occurrence of other words that add more relevant context to the document.

Tf-idf means term-frequency times inverse document-frequency.

$$\text{tf-idf}(t,d) = \text{tf}(t,d) \times \text{idf}(t)$$

, where t is the term and d is the document.

$$\text{idf}(t) = \log \frac{n}{1+\text{df}(t)}.$$

, where n is the total number of documents and df(t) is the number of documents that contain term t.

III. Multinomial Naive Bayes Approach

1. First, we calculate class prior probabilities.

$$P_i = \frac{\text{number of documents in class}_i}{\text{total number of documents}}$$

2. Next, we calculate the class conditional probability of the j'th word in vocabulary given class i. This is calculated for all words in the vocabulary across all classes.

$$p(x_j/c_i) = \frac{(\text{no. of occurrences of } x_j \text{ in documents belonging to } c_i) + \alpha}{(\text{no. of occurrences of } x_{i=1,2,\dots,V} \text{ in documents belonging to } c_i) + |V|\alpha}$$

, where c_i represents i'th class, α is the laplace smoothing coefficient for numerical stability and $|V|$ is the number of words in the vocabulary.

3. Given a document $d = \{x_1, x_2, \dots, x_V\}$, we calculate the a posteriori probability.

$$p(c_i/d) = \frac{p(d/c_i) \times p(c_i)}{p(d)}$$

$$p(c_i/d) = \frac{p(x_1 x_2 \dots x_V / c_i) \times p(c_i)}{p(d)}$$

$$p(c_i/d) = \frac{p(x_1/c_i) \times p(x_2/c_i) \times \dots \times p(x_V/c_i) \times p(c_i)}{p(d)} \quad (\text{assume independence of } j\text{'th feature})$$

4. Finally, the predicted class is given by the class with the maximum a posterior probability.

$$\operatorname{argmax} p(c_i/d) = \operatorname{argmax} \frac{p(x_1/c_i) \times p(x_2/c_i) \times \dots \times p(x_V/c_i) \times p(c_i)}{p(d)}$$

$$\operatorname{argmax} p(c_i/d) = \operatorname{argmax} p(x_1/c_i) \times p(x_2/c_i) \times \dots \times p(x_V/c_i) \times p(c_i)$$

IV. Results

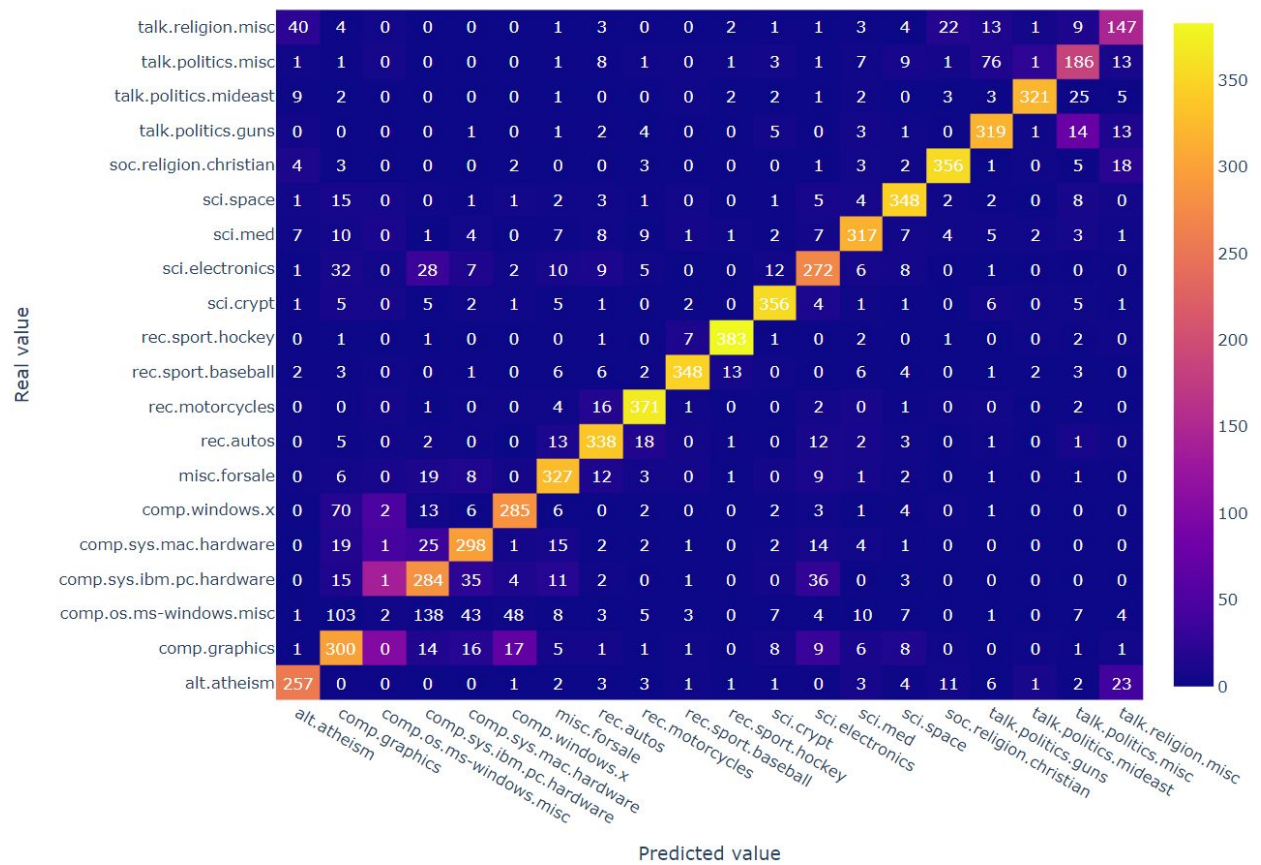
Test set results

Feature	Bag of Words	TF-IDF
Accuracy	0.79939	0.83258
F1 - Score	0.78198	0.82707

Train set results (*too verify the model)

Feature	Bag of Words	TF-IDF
Accuracy	0.96818	0.99779
F1 - Score	0.96603	0.99783

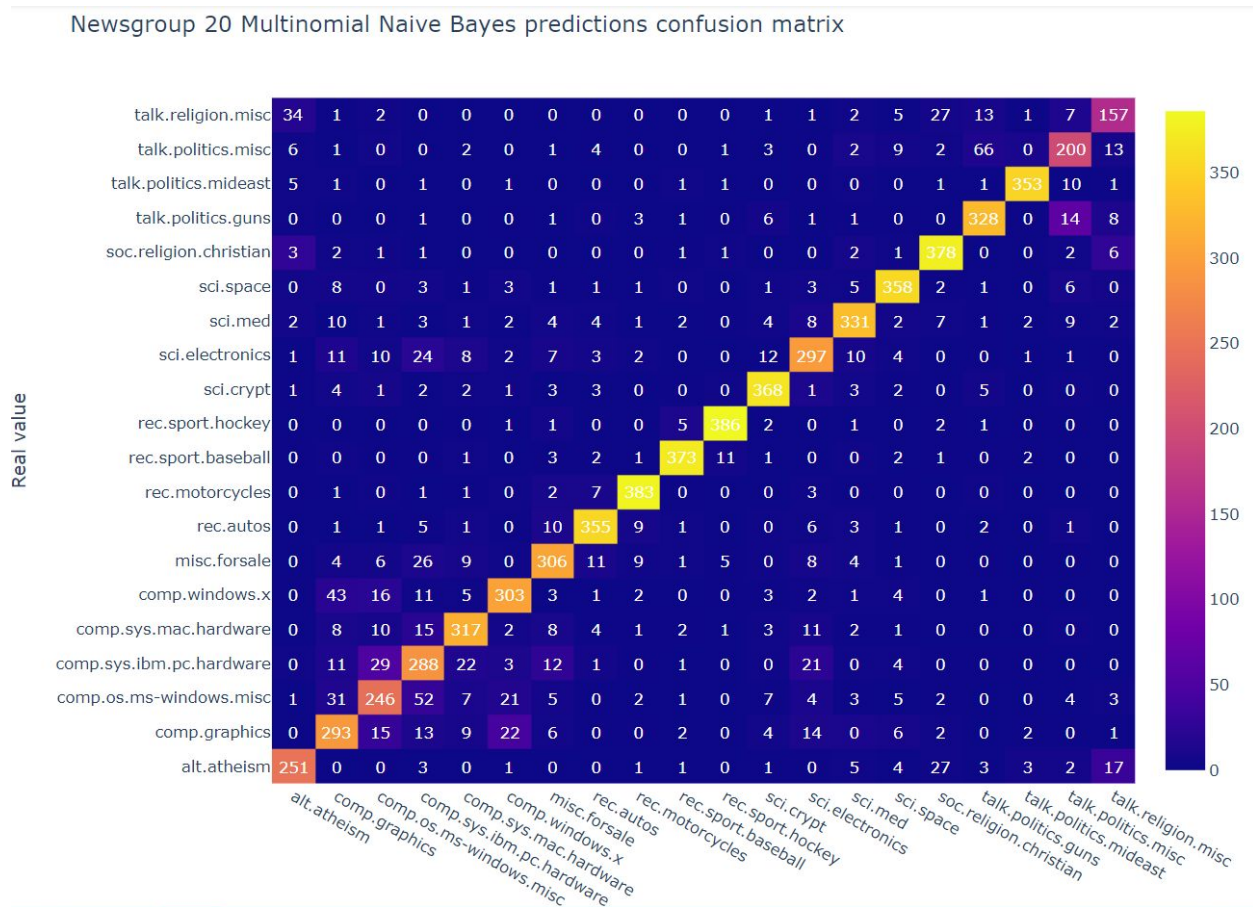
Confusion matrix (Bag of Words)



Observations

- It is interesting to see that 40 documents belonging to talk.religion.misc are misclassified as alt.atheism.
- Similarly, comp.os.ms-windows.misc is commonly confused with other classes like comp.graphics, comp.windows.x and comp.sys.ibm.pc.hardware.
- 76 documents in talks.politics.misc are misclassified as talks.politics.guns.
- Although the subjects are related, it is difficult for the model to capture the exact context based on the bag-of-words approach.

Confusion matrix (TF-IDF)



The confusion matrix shows a clear but slight improvement over using the bag of words approach.

Sklearn results

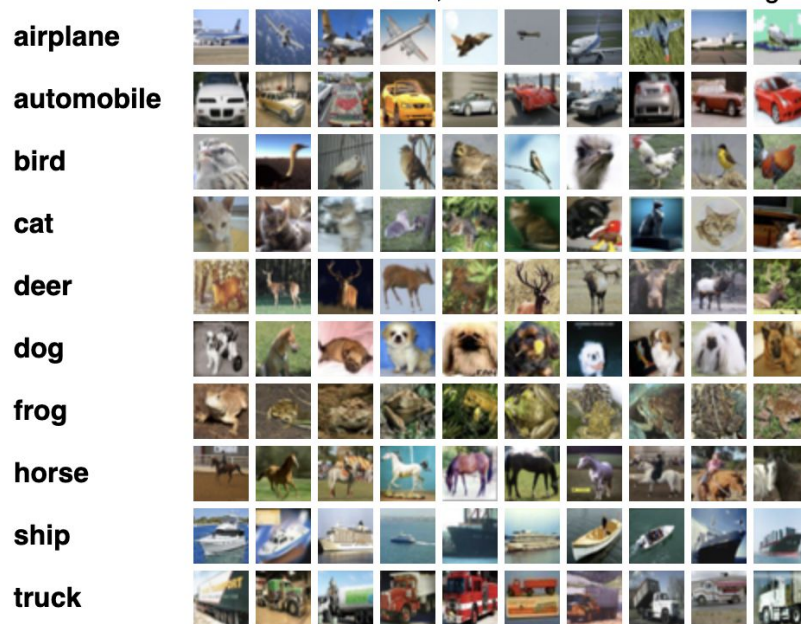
Feature	Bag of Words	TF-IDF
Accuracy	0.80390	0.83523
F1 - Score	0.78520	0.82906

V. Conclusion

TF-IDF and Bag of words don't take into account the position of the words but still the multinomial bayes classifier is able to perform well. When comparing the two, we find that using TF-IDF outperforms the bag of words approach, suggesting that it is better to penalize words that occur commonly across documents. Using the multinomial bayes classifier with TF-IDF, we are able to achieve an accuracy of 83.258% on the test set. Compared with sklearn the differences are subtle with our model performing nearly as well. Minor differences could be due to the fact that sklearn is able to use all words in the documents due to their efficient sparse matrix implementation, while we cap it at a slightly lesser number.

CIFAR-10 dataset

CIFAR-10 is a very popular Computer Vision dataset consisting of 60000 images each of size 32x32. These images are spread across 10 classes and have been divided into the train set consisting of 50000 images and a test set containing 10000 images.



For this task, we consider a binary classification problem by taking two classes from the dataset.

Data Exploration and Preprocessing

There are 5000 training images and 1000 testing images for each class and hence CIFAR-10 is a completely balanced dataset.

- Normalized by dividing by 255 to ensure pixel values are between 0 and 1
- Convert into gray-scale and flatten the image to generate a 1024 image representation.

Multivariate Gaussian Modelling

To model the class conditional densities for each class, we assume the data to be in the form of a multivariate gaussian distribution with mean μ and a covariance matrix Σ estimates.

Assuming that each training data images are sampled independently, we determine the density for each class according to the following equation

Given $D = \{x_1, x_2, x_3, \dots\}$

$$p(D/c) = p(x/c) \quad \forall \quad x \in D$$

Where $p(x/c)$ is the independent class conditional densities, given by

$$p(\mathbf{x}; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp \left(-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right)$$

To determine the estimates, we use MLE estimation

$$\mu_i = E(X_i) = \sum x_i/n$$

$$\Sigma_i = E((X_i - \mu_i) \cdot (X_i - \mu_i)^T)$$

μ is a 1024 dimension vector representing the mean of the training images

Σ is a 1024x1024 matrix which represents the variation among pixels.

Since, both of these classes have individual pixel variations with airplanes and cats occurring in various positions in the image, the mean estimate obtained does not display any physical representation of either of the class.

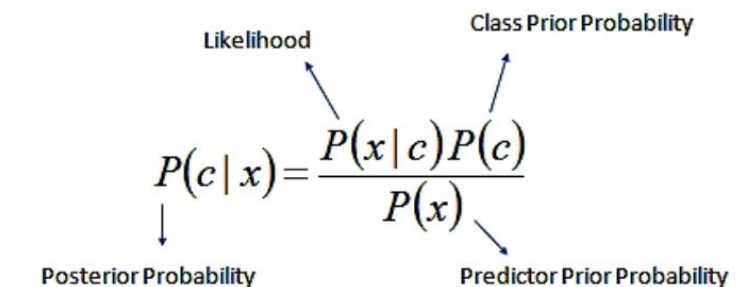
Priors:

Since both the classes are equally balanced, the priors for both classes are equal to 0.5

Bayes Classifier

We implement the Bayes Classifier, to determine the class of a given image by computing and comparing the posteriors obtained for each class.

The posteriors are computed according to the following equation



The diagram shows the equation $P(c | x) = \frac{P(x | c)P(c)}{P(x)}$ with four labels and arrows: 'Likelihood' points to $P(x | c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c | x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$

$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$

The Bayes classifier predicts the class with the higher posterior probability.

Laplace Smoothing and Logarithmic density

Since, the class conditional density, requires the computation of the inverse and determinant of the covariance matrix, there arise some issues, since many times, the covariance matrix is singular causing mathematical uncertainty. This could probably arise due to some values in the diagonal becoming zero.

In Laplace smoothing, we add a regularization parameter to the covariance matrix to make it non-singular.

This regularizer parameter α is determined by trial and error.

We found that higher dimensional matrices require higher α values to avoid singularity evident from our experiments on MNIST and CIFAR10 where, MNIST being a 784 dimensional matrix, required an α close to 0.01 but in case of CIFAR-10 we need at least of 0.5.

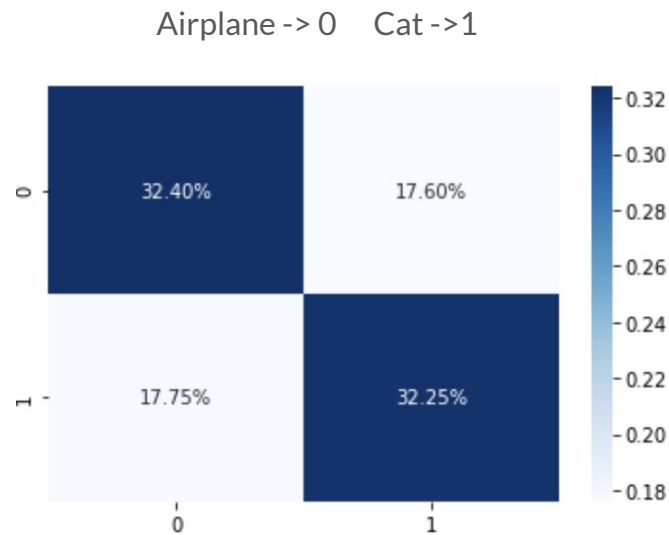
$$\Sigma = \Sigma + \text{diag}()$$

Furthermore, the computed determinant and inverse are considerably small and hence all computations are performed on log-scale.

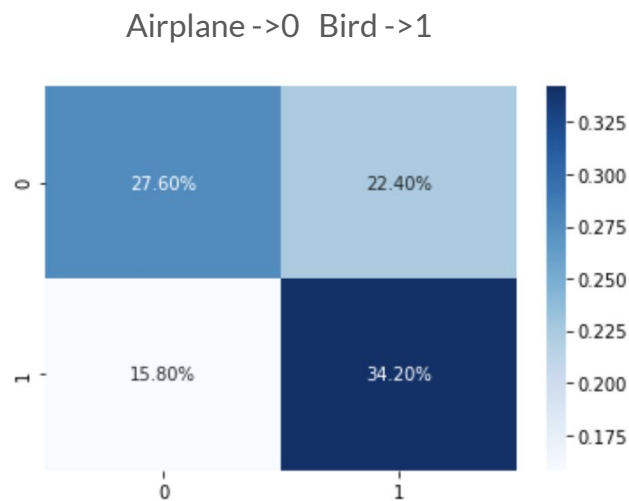
$$\begin{aligned}\log p(C_k | \mathbf{x}) &\propto \log \left(p(C_k) \prod_{i=1}^n p_{ki}^{x_i} \right) \\ &= \log p(C_k) + \sum_{i=1}^n x_i \cdot \log p_{ki}\end{aligned}$$

Results

We obtain the overall accuracy for the airplane v/s cat classes using a multivariate gaussian modelling to be 64.65% .



Using the same classifier model for the classes airplane v/s bird , we obtain an overall accuracy of 61.8%



Multivariate Exponential Modelling

Exponential distribution is one of the famous modelling distributions mainly describing events occurring in the positive real axis. The univariate distribution is as given below-

$$f(x; \lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

λ being the parameter of the distribution.

This is determined by maximizing the log-likelihood function

$$l(\lambda; x_1, \dots, x_n) = n \ln(\lambda) - \lambda \sum_{j=1}^n x_j$$

We obtain the estimate to be the inverse of the sample mean

$$\hat{\lambda}_n = \frac{n}{\sum_{j=1}^n x_j}$$

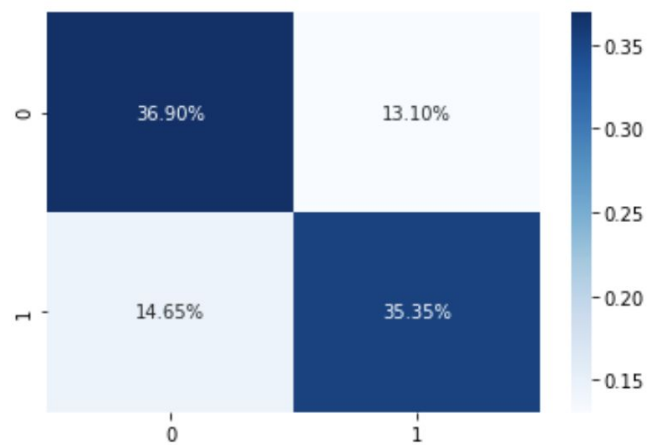
While modelling the dataset as an exponential distribution, we consider each pixel to be an independent variable and hence, the class conditional densities are obtained by a product of individual pixel distributions.

$$F(X, \lambda) = \prod_{i=1}^{1024} f(x_i, \lambda_i)$$

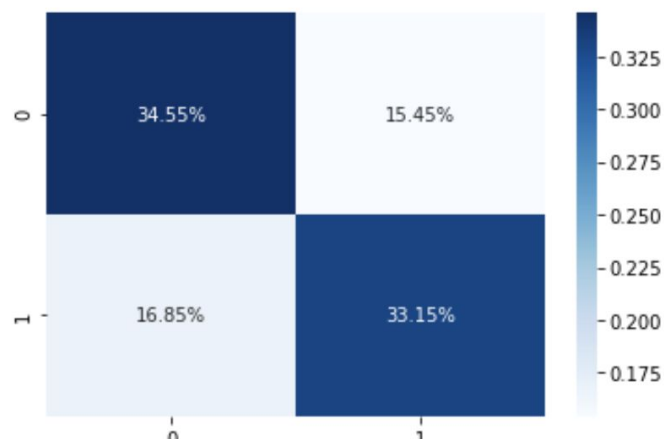
We implement the Bayes classifier by computing the posterior probabilities in log-scale. Since the priors are equal, their effects cancel out.

Results

We obtain the overall accuracy to be 72.25% for



In case of airplane v/s bird classifier, we obtain the overall accuracy to be 67.7%



Inferences

- CIFAR-10 is one of the most difficult datasets, owing to the very small image size and sometimes cannot be humanly identified.
- Exponential modelling seemed to be performing better than the gaussian modelling. This might have been due to high skewness in the data.
- Regularization parameters are required to ensure non-singularity in covariance matrices. This parameter increases with the increase in matrix size
- Building a classifier over similar classes led to decrease in accuracy as evident in case of airplane v/s bird and airplane v/s cat. Since birds could be mistaken as airplanes. However the distinction between airplanes and cats are more evident.