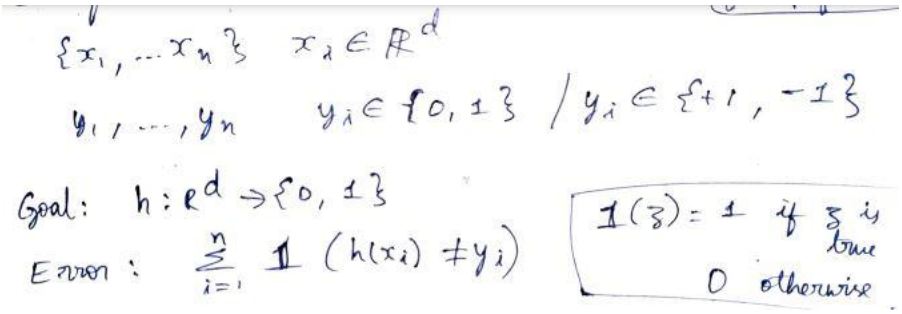
PRML ASSIGNMENT 3 REPORT
SPAM OR HAM?

ADARSHA K S
CS22S017

Classification is a Supervised Learning Problem in Machine Learning, in which given the data and supervision, we need to classify data to different classes.In classification, if we need to classify to only two classes, then we call such problem as Binary Classification problem. Spam or Ham is an Binary Classification Problem. Here we classify the given test instances, here emails, into two different classes-Spam or Ham based on the observations previously done from the training instances.



IMPLEMENTATION DETAILS OF SPAM CLASSIFIER

The dataset used : https://www.kaggle.com/datasets/venky73/spam-mails-dataset

This Dataset contains 3 columns:

label	text	label_num

The label column specifies if the email on that row is ‘spam’ or ‘ham’. The text column contains the email and the label_num contains either 1 or 0, for Spam or Ham respectively

Preprocessing the data :
The noise present in email, like redundant or unnecessary characters and jumbled up words has to be processed before features were extracted.
Methods performed in this implementation are:
1.Any duplicate data was removed from the dataset.
2.Text data is converted to lowercase for consistency.
3.Noise like special characters, numbers and punctuations are removed from the email data.
4.Frequently occurring words such as ‘the’, ‘it’ etc. are removed as they are not prominent features for distinguishing spam and non-spam emails.
5.All the words were stemmed using stemming libraries.Stemming is the process of reducing inflected words to their word stem. eg: Changed to change.

The words that were present in all the emails of the dataset were stored into a dictionary which is used later to construct a vector for each email in the dataset.

For Feature Extraction, Technique called TF-IDF vectorization of the document was used to convert each email into |V| sized vectors, where |V| = vocabulary size.
Each document can be represented as a vector of weights wij, where wij is the weight for the i-th email's j-th word, where j ranges from {1, 2, … |V|}.

D(i) = [w(i1), w(i2), w(i3), … w(i|V|)]

w(ij) = tf (ij) * idf (if the word is in the i-th email);0 (otherwise)

tf = number of occurrences of the j-th word in the i-th email
idf = log(Total number of emails / No of emails containing the j-th word).

After these steps,each document is represented as a |V| dimensional vector where the weights corresponding to each word in the email is kept at the index of the word in the vocabulary.

The training data constructed from all these vectors is a n * |V| matrix where each row is a datapoint xi in |V| dimensions and each column is a feature for the data.

The details of Mathematics behind the chosen algorithm is explained in following page.

(pto)

Naive Baye's Algorithm with Gaussian Likelihood Assumption is used in the algorithm.

wkt the density function for a multivariate Gaussian is given by $p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)}$

This is used as likelihood function. for

The log of the posterior using this function as likelihood, by Baye's Rule:

$$\log(L(\mu_k,\Sigma)) = \sum_{i=1}^{|V|}\left[-\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma|) - \frac{1}{2}(x^i-\mu_k)^T\Sigma^{-1}(x^i-\mu_k) + P(y^i)\right]$$

Maximizing log likelihood is same as maximizing the probability of a class given the ~~data~~ posterior probability for data.

The MLE estimates for the prior & means can be used from the Gaussian Discriminant Analysis.

$$\mu_0 = \frac{\sum_{i=1}^{|V|} \mathbb{1}(y^i=0)x^i}{\sum_{i=1}^{|V|}\mathbb{1}(y^i=0)} \qquad \mu_1 = \frac{\sum_{i=1}^{|V|}\mathbb{1}(y^i=1)x^i}{\sum_{i=1}^{|V|}\mathbb{1}(y^i=1)}$$

μ_0: Mean for class 0 μ_1 - Mean for class 1.

Prior for each class, $\pi_c = \frac{N_c}{N}$

As Naive Bayes assumes independence of each feature, the covariance matrix is easy to be calculated as it is a diagonal matrix with the diagonal entries as the variance of each feature.

Parameters Estimated:

Means and the covariances are estimated and stored and used for the posterior probability computation of each test data. The class with the maximum log of posterior probability is chosen as the predicted class.

Result Observed:

```
Training data Accuracy:
0.9891761876127481
Predictions for the emails in test folder :
[0 1]
```

After training, 1 easy ham and 1 easy spam emails have been tested and the model made the correct classification. These emails are used from url :
https://spamassassin.apache.org/old/publiccorpus/

Further by adding any number of email to be classified in test folder, we can observe the result, if it is spam or not.