# 

# 

# **MACHINE LEARNING LABORATORY**

# 

# **COURSE CODE : UE15CS356**

# **Mini Project Report**

# **Topic : Classifying emails and acquiring insights**

# 

# 

# **TEAM MEMBERS**

# **Bhavana Naga Sai Malepaty 01FB15ECS068**

# **Brindavana Sachidanand 01FB15ECS072**

# **Chaitra M.V. 01FB15ECS073**

# **Chinmayi P.S. 01FB15ECS078**

# 

# 

# 

# **Classifying emails and acquiring insights**

**Problem Statement:**

Emails are considered as official documents in communication among users. Email-categorization improves user efficiency in managing the inbox and help users extract data from the same. Machine-generated emails create more than 90% of the e-traffic.

Machine generated emails, being highly structured, make it easier to extract data from them, as all mails have a similar format.

The emails are classified into the following categories:

1.Company Business, Strategy

2.Purely Personal

3.Personal but in professional context (e.g., it was good working with you)

4.Logistic Arrangements (meeting scheduling,technical support)

5.Employment arrangements (job seeking, hiring, recommendations etc)

6.Document editing/checking (collaboration)

7.Empty message (due to missing attachment)

8.Empty message

**Contents:**

* Project name/Problem Statement and Group info
* ML techniques - Survey
* Dataset details
* Design of Model
* Results
* Concluding Remarks
* Conclusion
* References

**ML Techniques:**

1. **k-Nearest Neighbours Algorithm:**

The k-nearest neighbours algorithm (k-NN) is anon-parametric method used forclassification and regression. In both cases, the input consists of the *k* closest training examples in thefeature space . The output depends on whether *k*-NN is used for classification or regression

In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors .

* **Brute Force:** The most naive neighbor search implementation involves the brute-force computation of distances between all pairs of points in the dataset. Brute force search or exhaustive search is a very general problem solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem statement. The distance metrics used for Brute Force in our model are Cosine, Euclidean, Manhattan, Minkowski, and Chebyshev for N samples in D dimensions, this approach scales as O[D N^2].
* **K-D Tree:** K-D tree structures attempt to reduce the required number of distance calculations by efficiently encoding aggregate distance information for the sample. KD tree is a binary tree structure which recursively partitions the parameter space along the data axes, dividing it into nested orthotropic regions into which data points are filed. The construction of a Kd tree is very fast: because partitioning is performed only along the data axes, no D-dimensional distances need to be computed. Once constructed, the nearest neighbor of a query point can be determined with only O[\log(N)] distance computations. The distance metrics used for K-D tree in our model are Euclidean, Manhattan, Minkowski, and Chebyshev
* **Ball Tree:** To address the inefficiencies of KD Trees in higher dimensions, the ball tree data structure was developed. Where KD trees partition data along Cartesian axes, ball trees partition data in a series of nesting hyper-spheres. This makes tree construction more costly than that of the KD tree, but results in a data structure which can be very efficient on highly-structured data, even in very high dimensions.A ball tree recursively divides the data into nodes defined by a centroid C and radius r, such that each point in the node lies within the hyper-sphere defined by r and C. The distance metrics used for ball tree in our model are Euclidean, Manhattan, Minkowski, and Chebyshev

**2. Support Vector Machines**

Support Vector Machines aresupervised learning models with associated learningalgorithms that analyze data used forclassification andregression analysis.

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier . An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. The kernel functions implemented in our model are:

* Linear SVM
* Polynomial SVM
* Radial SVM
* Sigmoid SVM

**3. Naive Bayes Classifier**

Naive Bayes classifiers are a collection of classification algorithms based on Bayes’ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

* **Gaussian Naive Bayes -** In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also calledNormal Distribution. When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values
* **Multinomial Naive Bayes -** Feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution. This is the event model typically used for document classification.

**4. Logistic Regression**

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

The goal of logistic regression is to find the best fitting (yet biologically reasonable) model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables

**Dataset details:**

**Dataset name** : Enron Email Data Set

**Source** : https://www.cs.cmu.edu/~./enron/

**Attributes count and Attributes list:**

Message-ID:

Date:

From:

To:

Subject:

Cc:

Mime-Version:

Content-Type:

Content-Transfer-Encoding:

Bcc:

X-From:

X-To:

X-cc:

X-bcc:

X-Folder:

X-Origin:

X-FileName:

**Class count : 8**

**Instance count : 1709**

**Data Preprocessing:**

All lines of all files is processed to categorize the data correctly into the corresponding attributes column. A CSV file is created which has a column per attribute and a row per email file. The entire contents of the email is placed under the contents header of the column. The classification of each row (mail) had to be added separately to the data set.

**Feature Extraction:**

Content features include features derived from message subject and body. A TFIDF, short for **term frequency–inverse document frequency** is created from each of the email contents .

TFIDF is a numerical statistic (generally a matrix ) that is intended to reflect the importance of a word to a document (here email) in a collection or corpus.

To create this, the top keywords of the emails are required. And only the most frequently occuring words are used for to build the TFIDF. Else, the matrix will be quite sparse, causing unnecessary computations.

Each row represents an email, and each column represents a word. Each cell in the matrix contains an integer between 0 and infinity which is the count of how many times a given word occured in a given email.

The next step is to implement a technique called Latent Semantic Analysis(LSA), LSA is required to create a vector representation of each email. During this process, stop words are removed.

Imbalanced classes put “accuracy” out of business. This causes a limitation in the dataset, where a disproportionate ratio of observations in each class is present. Hence upsampling is done to balance the classes and improve accuracy significantly .

**Design of the model**

**Architectural Design:**

The models built are:

1. K Nearest Neighbours : Under this, we have 3 different algorithms like Brute, KD Tree and Ball KD tree (each having different distance metrics)
2. Support Vector Machines: In Support Vector Machines, we have different types based on the choice of kernels like linear, polynomial, radial and sigmoid.
3. Naive Bayes: In Naive Bayes we are implementing two types i.e., Multinomial Naive Bayes and Gaussian Naive Bayes
4. Logistic Regression: Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome.

**Detailed Design:**

import pickle

import time

import pandas as pd

import numpy as np

import math

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.decomposition import TruncatedSVD

from sklearn.pipeline import make\_pipeline

from sklearn.preprocessing import Normalizer

from sklearn.model\_selection import train\_test\_split

from sklearn import model\_selection

from sklearn import svm

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, roc\_curve, auc

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

from sklearn.naive\_bayes import GaussianNB

from sklearn.naive\_bayes import MultinomialNB

from sklearn.decomposition import NMF

from sklearn.linear\_model import LogisticRegression

import matplotlib.pyplot as plt

**Results :**

**Classification Accuracy and its Limitations:**

Almost all ML techniques used will have some limitations. Some are listed below:

**i) SVM -** The limitation lies in the the choice of kernels. Although SVMs have good generalization performance, they can be slow in test phase. From a practical point of view perhaps the most serious problem with SVMs is the high algorithmic complexity and extensive memory requirements of the required quadratic programming in large-scale tasks.

**ii) Multinomial Naive Bayes** which is a specific case in Naive Bayes usually takes longer time for training.

**iii) KNN** algorithms suffer from the curse of dimensionality since they include distance metrics.

**iv) Logistic regression** would "overfit" the model, meaning that it overstates the accuracy of its predictions.

v) The **data set** had imbalanced classes, which affects accuracy significantly.

vi) In **LSA,** representation is dense, making it difficult to index based on individual dimensions.

LSA is a linear model, and hence not the best solution to handle non - linear dependencies. The latent topic dimension cannot be chosen to arbitrary numbers. It depends on the rank of the matrix, so can't go beyond that.

**Confusion Matrix**

**Euclidean knn**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 131 | 4 | 35 | 27 | 4 | 24 | 9 | 7 |
| 0 | 250 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 8 | 211 | 6 | 4 | 4 | 0 | 0 |
| 37 | 5 | 14 | 132 | 6 | 12 | 9 | 2 |
| 4 | 4 | 5 | 6 | 239 | 0 | 0 | 0 |
| 26 | 0 | 9 | 19 | 14 | 186 | 3 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 247 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 237 |

**Linear SVM**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 161 | 1 | 17 | 29 | 4 | 26 | 3 | 0 |
| 0 | 250 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 14 | 162 | 15 | 15 | 1 | 0 | 0 |
| 32 | 0 | 21 | 144 | 5 | 9 | 5 | 1 |
| 9 | 0 | 0 | 13 | 231 | 0 | 5 | 0 |
| 24 | 0 | 14 | 31 | 17 | 167 | 5 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 247 | 0 |
| 16 | 0 | 0 | 10 | 0 | 0 | 0 | 211 |

**Multinomial Naive Bayes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 122 | 1 | 28 | 29 | 8 | 39 | 5 | 12 |
| 0 | 244 | 5 | 0 | 0 | 0 | 0 | 0 |
| 52 | 3 | 102 | 37 | 18 | 29 | 0 | 0 |
| 40 | 0 | 32 | 117 | 16 | 39 | 2 | 2 |
| 8 | 0 | 17 | 15 | 194 | 2 | 11 | 0 |
| 24 | 0 | 6 | 25 | 6 | 162 | 2 | 0 |
| 0 | 0 | 0 | 0 | 0 | 19 | 216 | 13 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 235 |

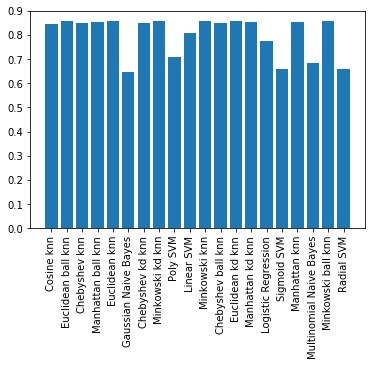
**Logistic Regression**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 111 | 2 | 22 | 35 | 9 | 36 | 7 | 22 |
| 0 | 244 | 5 | 0 | 0 | 0 | 0 | 0 |
| 40 | 11 | 104 | 40 | 28 | 10 | 5 | 3 |
| 21 | 2 | 26 | 144 | 17 | 28 | 6 | 4 |
| 4 | 1 | 12 | 18 | 196 | 5 | 11 | 0 |
| 12 | 2 | 7 | 22 | 12 | 164 | 5 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 235 | 13 |
| 0 | 0 | 0 | 16 | 0 | 0 | 0 | 234 |

**Accuracy, Error Rate, Precision, Recall and F1-score of various models**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl no** | **Technique** | **Accuracy** | **Error**  **Rate** | **Precision** | **Recall** | **F1-score** |
| 1 | Cosine KNN | 82.889 | 17.110 | 0.83 | 0.83 | 0.83 |
| 2 | Euclidean KNN | 83.658 | 16.342 | 0.83 | 0.84 | 0.83 |
| 3 | Manhattan KNN | 83.453 | 16.547 | 0.83 | 0.83 | 0.83 |
| 4 | Minkowski KNN | 83.658 | 16.342 | 0.83 | 0.84 | 0.83 |
| 5 | Chebyshev KNN | 83.914 | 16.086 | 0.83 | 0.84 | 0.83 |
| 6 | Euclidean KD KNN | 83.555 | 16.445 | 0.83 | 0.84 | 0.83 |
| 7 | Manhattan KD KNN | 83.248 | 16.752 | 0.82 | 0.83 | 0.83 |
| 8 | Minkowski KD KNN | 83.555 | 16.445 | 0.83 | 0.84 | 0.83 |
| 9 | Chebyshev KD KNN | 83.811 | 16.189 | 0.83 | 0.84 | 0.83 |
| 10 | Euclidean Ball KNN | 83.607 | 16.393 | 0.83 | 0.84 | 0.83 |
| 11 | Manhattan Ball KNN | 83.299 | 16.701 | 0.82 | 0.83 | 0.83 |
| 12 | Minkowski Ball KNN | 83.607 | 16.393 | 0.83 | 0.84 | 0.83 |
| 13 | Chebyshev Ball KNN | 83.811 | 16.189 | 0.83 | 0.84 | 0.83 |
| 14 | Linear SVM | 80.584 | 19.416 | 0.81 | 0.81 | 0.81 |
| 15 | Polynomial SVM | 11.117 | 88.883 | 0.01 | 0.11 | 0.02 |
| 16 | Radial SVM | 61.578 | 38.422 | 0.68 | 0.62 | 0.61 |
| 17 | Sigmoid SVM | 61.578 | 38.422 | 0.68 | 0.62 | 0.61 |
| 18 | Gaussian Naive Bayes | 65.523 | 34.477 | 0.65 | 0.66 | 0.65 |
| 19 | Multinomial Naive  Bayes | 71.311 | 28.688 | 0.71 | 0.71 | 0.71 |
| 20 | Logistic Regression | 73.361 | 26.639 | 0.72 | 0.73 | 0.73 |

**Concluding Remarks**

**Bar plot showing accuracies of various models**

**Conclusion**

We have a labelled dataset of emails with 8 categories. We built various models to classify emails.

It turns out that KNN (where all methods are equally good ) is the best model as it has the highest accuracy around 83% .

Linear SVM is the best SVM classifier with an accuracy of around 80% - almost as good as the KNN techniques.

Next best is logistic regression with around 73% accuracy.

Multinomial Naive Bayes has proved to be better than Gaussian Naive Bayes with an accuracy of 71% .

**References**

* <https://towardsdatascience.com/how-i-used-machine-learning-to-classify-emails-and-turn-them-into-insights-efed37c1e66>