**01EMAIL SPAM DETECTION USING SUPERVISED MACHINE LEARNING**

**SEABORN-**

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and colour palettes to make statistical plots more attractive. It is built on the top of [matplotlib](https://www.geeksforgeeks.org/python-introduction-matplotlib/) library and also closely integrated to the data structures from [pandas](https://www.geeksforgeeks.org/introduction-to-pandas-in-python/).  
Seaborn aims to make visualization the central part of exploring and understanding data. It provides dataset-oriented APIs, so that we can switch between different visual representations for same variables for better understanding of dataset.

* **Relational plots:** This plot is used to understand the relation between two variables.
* [**Categorical plots:**](https://www.geeksforgeeks.org/seaborn-categorical-plots/)This plot deals with categorical variables and how they can be visualized.
* [**Distribution plots:**](https://www.geeksforgeeks.org/seaborn-distribution-plots/)This plot is used for examining univariate and bivariate distributions
* [**Regression plots:**](https://www.geeksforgeeks.org/seaborn-regression-plots/)The regression plots in seaborn are primarily intended to add a visual guide that helps to emphasize patterns in a dataset during exploratory data analyses.
* [**Matrix plots:**](https://www.geeksforgeeks.org/ml-matrix-plots-in-seaborn/) A matrix plot is an array of scatterplots.
* **Multi-plot grids:**It is an useful approach is to draw multiple instances of the same plot on different subsets of the dataset.

**MATPLOTLIB-**

**Matplotlib**is easy to use and an amazing visualizing library in Python. It is built on NumPy arrays and designed to work with the broader SciPy stack and consists of several plots like line, bar, scatter, histogram, etc.

**HEAD-**

Of all data set displays first 5 rows

**TAIL-**

Of all data set displays last 5 rows

**.MAP-**

Python's map() is a built-in function that allows you to process and transform all the items in an iterable without using an explicit for loop, a technique commonly known as mapping. map() is useful when you need to apply a transformation function to each item in an iterable and transform them into a new iterable.

**.DESCRIBE-**

Pandas **describe()** is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. When this method is applied to a series of string, it returns a different output which is shown in the examples below.

**COUNT, UNIQUE, TOP, FREQ-**

When String is passed prints this.

When Numerical data is passed prints this along with mean, quartiles etc.

The freq is tied to the top immediately above it. top is the most frequent element, freq is how many times that most frequent element was seen.

**WORDCLOUD-**

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

For generating word cloud in Python, modules needed are – matplotlib, pandas and wordcloud.

**str-**

converting the series data into string for computational tasks

**BAG OF WORDS-**

Bag of words is a [Natural Language Processing](https://www.mygreatlearning.com/blog/natural-language-processing-tutorial/) technique of text modelling. In technical terms, we can say that it is a method of feature extraction with text data. This approach is a simple and flexible way of extracting features from documents.

A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order. It is called a “bag” of words because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

**pprint-**

The **pprint** module provides a capability to “pretty-print” arbitrary Python data structures in a well-formatted and more readable way!

**SKLEARN -**

**Important features of scikit-learn:**

* Simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, etc.
* Accessible to everybody and reusable in various contexts.
* Built on the top of NumPy, SciPy, and matplotlib.
* Open source, commercially usable – BSD license.

**COUNTVECTORIZER()-**

(AN INBUILT FUNCTION OF SK LEARN)

**CountVectorizer**is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such texts, and we wish to convert each word in each text into vectors (for using in further text analysis).

Provides an integer ID to each data

**.fit()-**

When you call fit method it estimates the best representative function for the the data points (could be a line, polynomial or discrete borders around). With that representation, you can calculate new data points.

**.get\_feature\_names()-**

To get the list of words which have been categorize as features

**.toarray()-**

converts in array

**train\_test\_split()-**

If size=0.2 //Ratio for Testing then testing data= 20% and training data is 80%

Returns 4 parameters x\_train, x\_test, y\_train, y\_test

If Random state: 10 , x\_train will not change for first 10 values otherwise it will change every time the code is executed

**LINEAR REGRESSION-**

Then we will fit our data

**.predict:**

Python predict() function enables us to predict the labels of the data values on the basis of the trained model.

Syntax: model.predict(data) The predict() function accepts only a single argument which is usually the data to be tested.

**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.

The dataset is divided into two parts, namely, **feature matrix** and the **response vector**.

* Feature matrix contains all the vectors(rows) of dataset in which each vector consists of the value of **dependent features**. In above dataset, features are ‘Outlook’, ‘Temperature’, ‘Humidity’ and ‘Windy’.
* Response vector contains the value of **class variable**(prediction or output) for each row of feature matrix. In above dataset, the class variable name is ‘Play golf’.

The fundamental Naive Bayes assumption is that each feature makes an:

* independent
* equal

contribution to the outcome.

With relation to our dataset, this concept can be understood as:

* We assume that no pair of features are dependent. For example, the temperature being ‘Hot’ has nothing to do with the humidity or the outlook being ‘Rainy’ has no effect on the winds. Hence, the features are assumed to be **independent**.
* Secondly, each feature is given the same weight(or importance). For example, knowing only temperature and humidity alone can’t predict the outcome accurately. None of the attributes is irrelevant and assumed to be contributing **equally** to the outcome.

**What is Multinomialnb?**

The Multinomial Naive Bayes algorithm is **a Bayesian learning approach popular in Natural Language Processing (NLP)**. The program guesses the tag of a text, such as an email or a newspaper story, using the Bayes theorem. It calculates each tag's likelihood for a given sample and outputs the tag with the greatest chance

**What is metrics in sklearn?**

Classification metrics. The sklearn. metrics module **implements several loss, score, and utility functions to measure classification performance**. Some metrics might require probability estimates of the positive class, confidence values, or binary decisions values.

**What is metrics accuracy\_score?**

**Accuracy classification score**. In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y\_true.

**What is Precision\_score?**

The precision is **the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives**. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative. The best value is 1 and the worst value is 0.

**How do you get a recall score?**

Compute the recall. **The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives**. The recall is intuitively the ability of the classifier to find all the positive samples. The best value is 1 and the worst value is 0.

**What is F1 score Sklearn?**

f1\_score. Compute the F1 score, also known as balanced F-score or F-measure. In the multi-class and multi-label case, this is **the average of the F1 score of each class with weighting depending on the average parameter**.

**RANDOM FOREST:**

The Random Forest or Random Decision Forest is a supervised Machine learning algorithm used for classification, regression, and other tasks using decision trees.  
The Random Forest classifier creates a set of decision trees from a randomly selected subset of the training set. It is basically *a set of decision trees (DT) from a randomly selected subset of the training set and then It collects the votes from different decision trees to decide the final prediction.*  
In this classification algorithm, we will use IRIS flower datasets to train and test the model. We will build a model to classify the type of flower.

**What is Randomforestclassifier Sklearn?**

A random forest classifier. A random forest is **a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting**.

**What is Sklearn ensemble?**

The sklearn. ensemble module **includes two averaging algorithms based on randomized decision trees: the RandomForest algorithm and the Extra-Trees method**. Both algorithms are perturb-and-combine techniques [B1998] specifically designed for trees.

**What is Sklearn SVM?**

Support vector machines (SVMs) are **a set of supervised learning methods used for classification, regression and outliers detection**. The advantages of support vector machines are: Effective in high dimensional spaces. Still effective in cases where number of dimensions is greater than the number of samples.

**What is SVC in Sklearn?**

It is **C-support vector classification** whose implementation is based on libsvm. The module used by scikit-learn is sklearn.

**What is sklearn Linear\_model?**

linear\_model is **a class of the sklearn module if contain different functions for performing machine learning with linear models**. The term linear model implies that the model is specified as a linear combination of features.

**What is logistic regression sklearn?**

Scikit-Learn. Logistic Regression is **a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable**. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).

**MATPLOTLIB:**

**Pyplot** is a Matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python and the advantage of being free and open-source. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. The various plots we can utilize using Pyplot are **Line Plot**, **Histogram**, **Scatter**, **3D Plot**, **Image**, **Contour**, and **Polar**.

**What is NumPy?**  
NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.  
Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

**What are xticks in Python?**

xticks() function is used to **get or set the current tick locations and labels of the x-axis**. It passes no arguments to return the current values without modifying them.

**What is a confusion matrix Python?**

A confusion matrix is **a tabular summary of the number of correct and incorrect predictions made by a classifier**. It can be used to evaluate the performance of a classification model through the calculation of performance metrics like accuracy, precision, recall, and F1-score.

**What is classification report in Python?**

It is **a performance evaluation metric in machine learning which is used to show the precision, recall, F1 Score, and support score of your trained classification model**.

**What is tkinter class?**

**Tkinter** is the most commonly used library for developing GUI (Graphical User Interface) in Python. It is a standard Python interface to the Tk GUI toolkit shipped with Python. As Tk and Tkinter are available on most of the Unix platforms as well as on the Windows system, developing GUI applications with Tkinter becomes the fastest and easiest.

**What is Title in tkinter?**

The title in tkinter refers to **a name assigned to the application window**. It is mostly found on the top of the application. For this, we can use the title() function. We create a widget object using the Tk() function and use the title() function to add a title to this window.

**Some other features of tkinter**

Tkinter provides some methods with the help of which we can get the current screen height and width.  
Following methods can be used to decide height and width :

winfo\_screenheight() // Returns screen height in **pixels**

winfo\_screenmmheight() // Returns screen height in **mm**

winfo\_screenwidth() // Returns screen width in **pixels**

winfo\_screenmmwidth() // Returns screen width in **mm**

**geometry**

Tkinter provides many methods; one of them is the **geometry()** method. This method is used to set the dimensions of the [Tkinter](https://www.geeksforgeeks.org/python-gui-tkinter/) window and is used to set the position of the main window on the user’s desktop.

**Pack()**

The Pack geometry manager packs widgets relative to the earlier widget. Tkinter literally packs all the widgets one after the other in a window.  We can use options like **fill**, **expand**, and **side** to control this geometry manager.  
Compared to the **grid** manager, the **pack** manager is somewhat limited, but it’s much easier to use in a few, but quite common situations:

* Put a widget inside a frame (or any other container widget), and have it fill the entire frame
* Place a number of widgets on top of each other
* Place a number of widgets side by side

**Explode in Pie:**

Exploding, or pulling out, the chosen slice from the rest of the pie. Focusing on one or more slices using a Pie of Pie chart. Focusing on one or more slices using a Bar of Pie chart.

**Autopct:**

As autopctis a function used to label the wedges with their numeric value, you can write there any label or format items quantity with it as you need. The easiest approach for me to show percentage label is using lambda: