**IMPLEMENTATION**

**MODULES:**

* Data Collection
* Dataset
* Data Preparation
* Model Selection
* Analyze and Prediction
* Accuracy on test set
* Saving the Trained Model

**MODULES DESCSRIPTION:**

**Data Collection:**

In the first module, we develop the Data Collection process. This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.

There are several techniques to collect the data, like web scraping, manual interventions. Our dataset is located in the model folder. The dataset is referred from the popular dataset repository called kaggle. The following is the link of the dataset.

Kaggle Link:

<https://www.kaggle.com/datasets/jayaprakashpondy/social-media-comment-text>

**Dataset:**

The dataset consists of 159572 individual data. There are 8 columns in the dataset, which are described below.

Id: Id number

comment\_text: comment text

toxic: 0 or 1

severe\_toxic: 0 or 1

obscene: 0 or 1

threat: 0 or 1

insult: 0 or 1

identity\_hate: 0 or 1

**Data Preparation:**

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, and data type conversions, etc.)

Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis

Split into training and evaluation sets

**Model Selection:**

We used Random Forest Classifier machine learning algorithm.

*The Random Forests Algorithm*

Let’s understand the algorithm in layman’s terms. Suppose you want to go on a trip and you would like to travel to a place which you will enjoy.

So what do you do to find a place that you will like? You can search online, read reviews on travel blogs and portals, or you can also ask your friends.

Let’s suppose you have decided to ask your friends, and talked with them about their past travel experience to various places. You will get some recommendations from every friend. Now you have to make a list of those recommended places. Then, you ask them to vote (or select one best place for the trip) from the list of recommended places you made. The place with the highest number of votes will be your final choice for the trip.

In the above decision process, there are two parts. First, asking your friends about their individual travel experience and getting one recommendation out of multiple places they have visited. This part is like using the decision tree algorithm. Here, each friend makes a selection of the places he or she has visited so far.

The second part, after collecting all the recommendations, is the voting procedure for selecting the best place in the list of recommendations. This whole process of getting recommendations from friends and voting on them to find the best place is known as the random forests algorithm.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

*How does the algorithm work?*

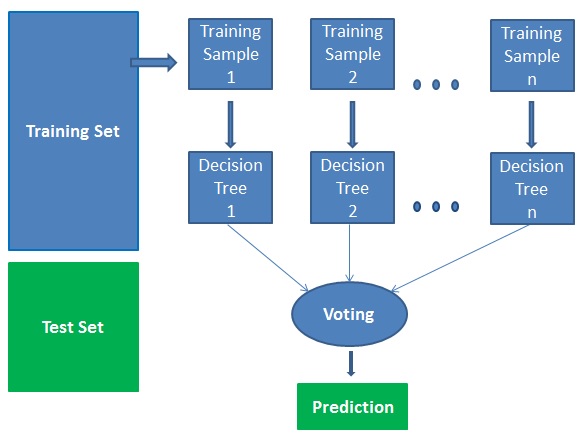
It works in four steps:

Select random samples from a given dataset.

Construct a decision tree for each sample and get a prediction result from each decision tree.

Perform a vote for each predicted result.

Select the prediction result with the most votes as the final prediction.



*Finding important features*

Random forests also offer a good feature selection indicator. Scikit-learn provides an extra variable with the model, which shows the relative importance or contribution of each feature in the prediction. It automatically computes the relevance score of each feature in the training phase. Then it scales the relevance down so that the sum of all scores is 1.

This score will help you choose the most important features and drop the least important ones for model building.

Random forest uses gini importance or mean decrease in impurity (MDI) to calculate the importance of each feature. Gini importance is also known as the total decrease in node impurity. This is how much the model fit or accuracy decreases when you drop a variable. The larger the decrease, the more significant the variable is. Here, the mean decrease is a significant parameter for variable selection. The Gini index can describe the overall explanatory power of the variables.

**Analyze and Prediction:**

In the actual dataset, we chose only 7 features :

comment\_text: comment text

toxic: 0 or 1

severe\_toxic: 0 or 1

obscene: 0 or 1

threat: 0 or 1

insult: 0 or 1

identity\_hate: 0 or 1

**Accuracy on test set:**

We got an accuracy of (toxic) 0.838055,

We got an accuracy of (severe\_toxic) 0.934874,

We got an accuracy of (obscene) 0.909091,

We got an accuracy of (insult) 0.883993,

We got an accuracy of (threat) 0.795539,

We got an accuracy of (identity\_hate) 0.768448 on test set so we implemented this algorithm.

**Saving the Trained Model:**

Once you’re confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle.

Make sure you have pickle installed in your environment.

Next, let’s import the module and dump the model into .pkl file