Spamerly

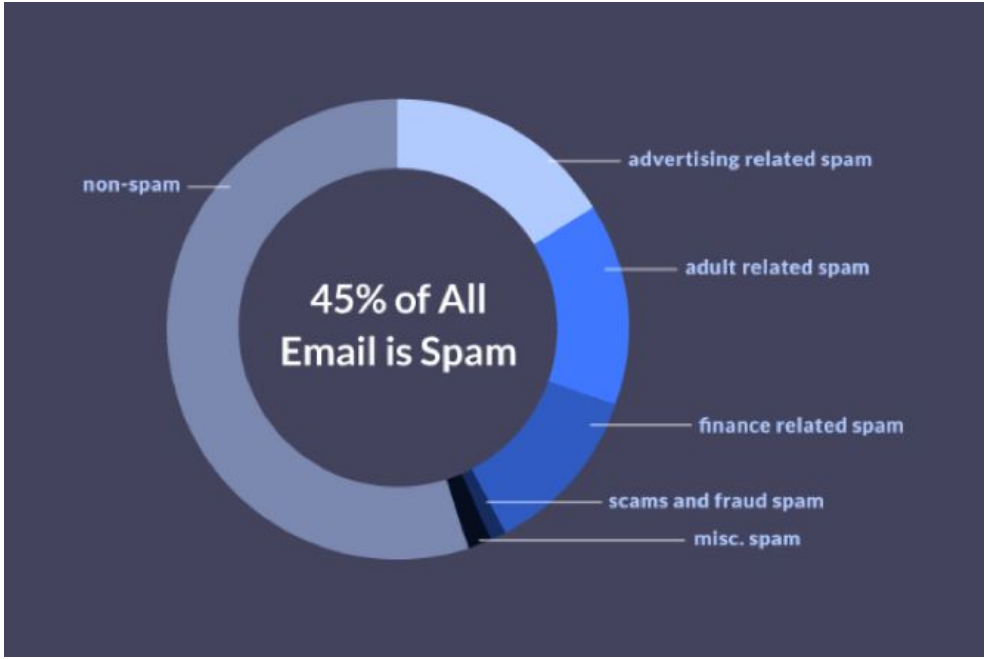
- Naresh
- Sandip
- Dhwani
- Shambo
- Prashanthi

Outline

- Motivation
- Data Preprocessing
- SVM
- Neural Network
- Bayesian
- Results

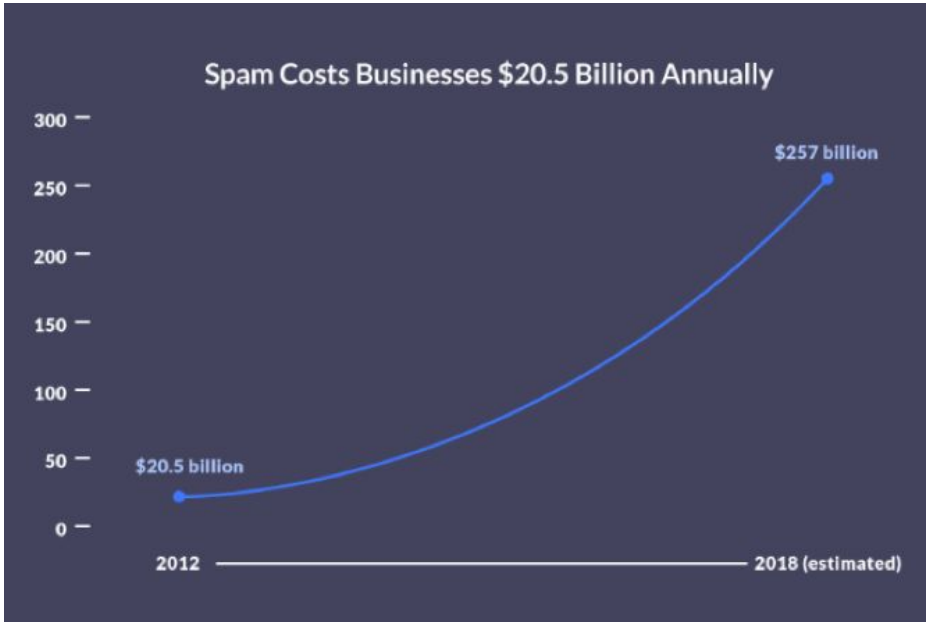
14.5 billion

spam messages per day



Source: https://www.propellercrm.com/blog/email-spam-statistics

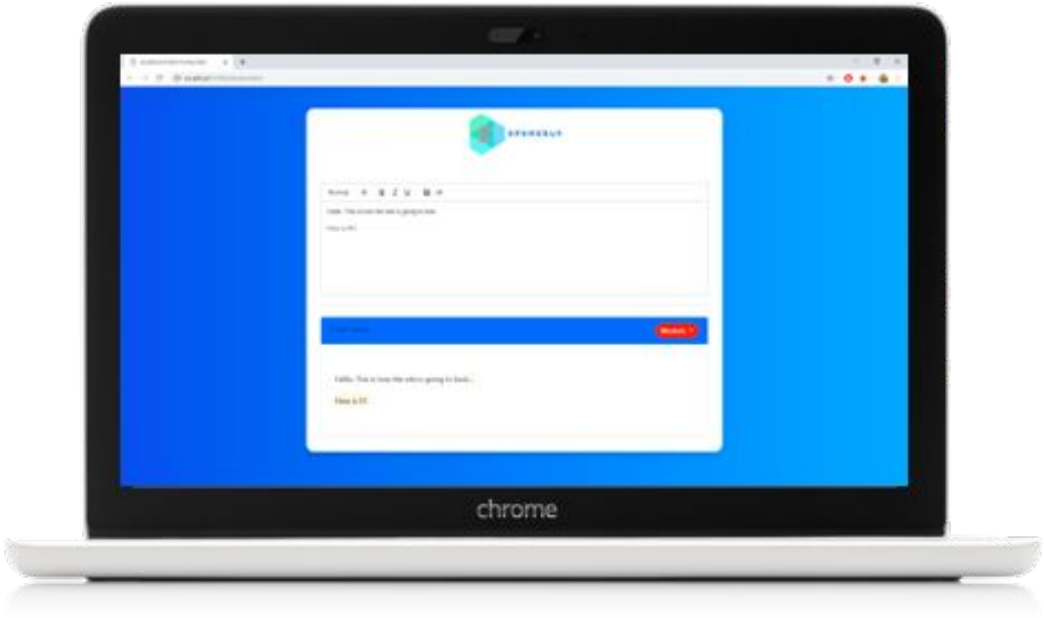
SPAM ≠ SCAM



Source: https://www.propellercrm.com/blog/email-spam-statistics

Introducing

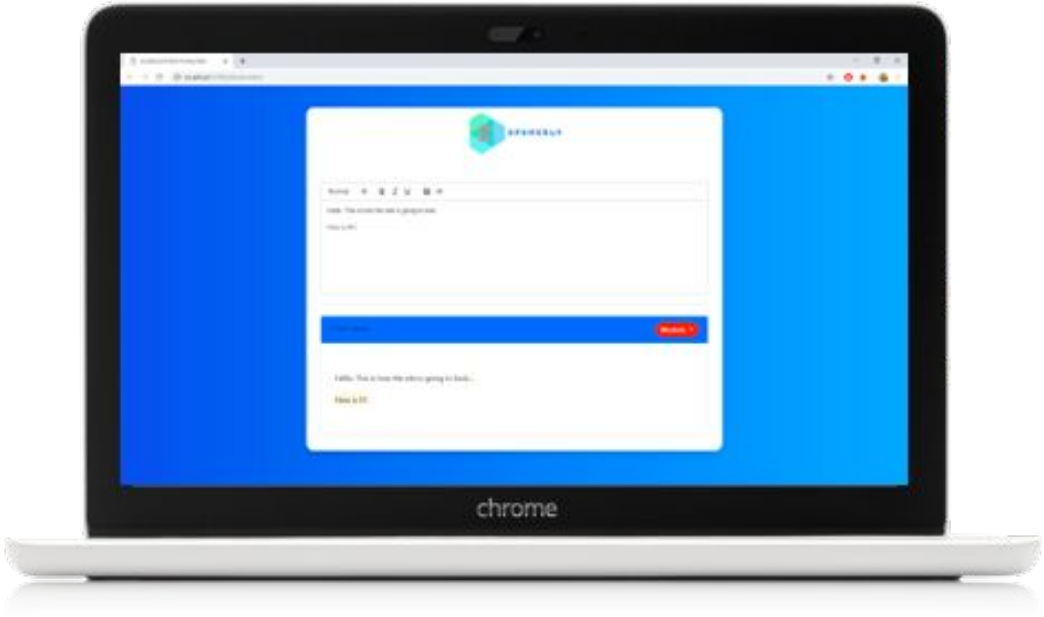
Spamerly

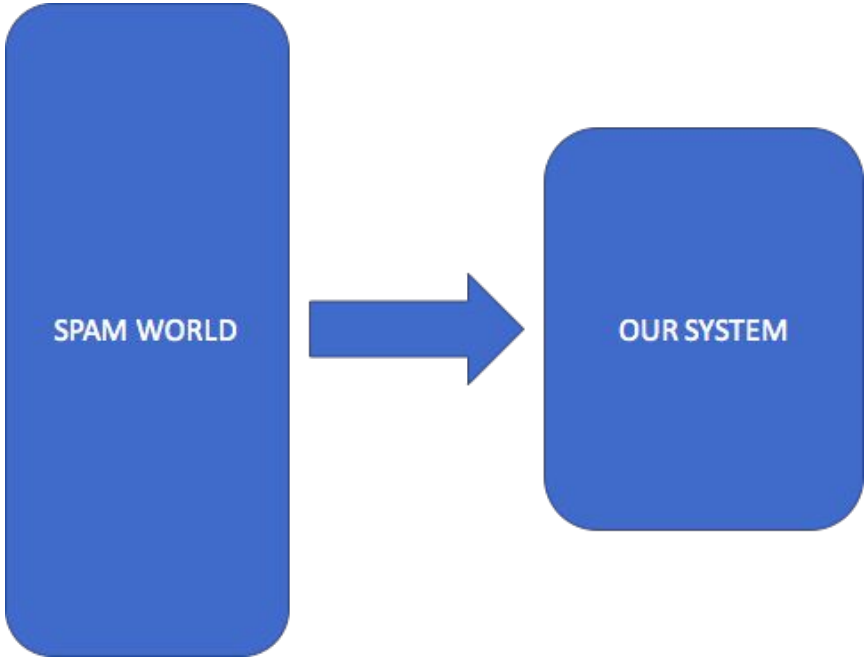


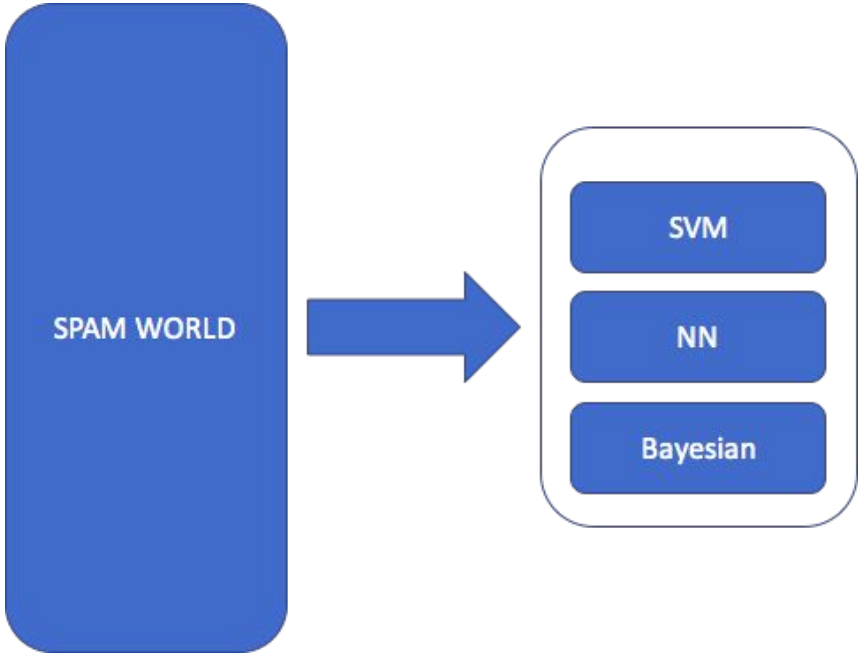
Simple

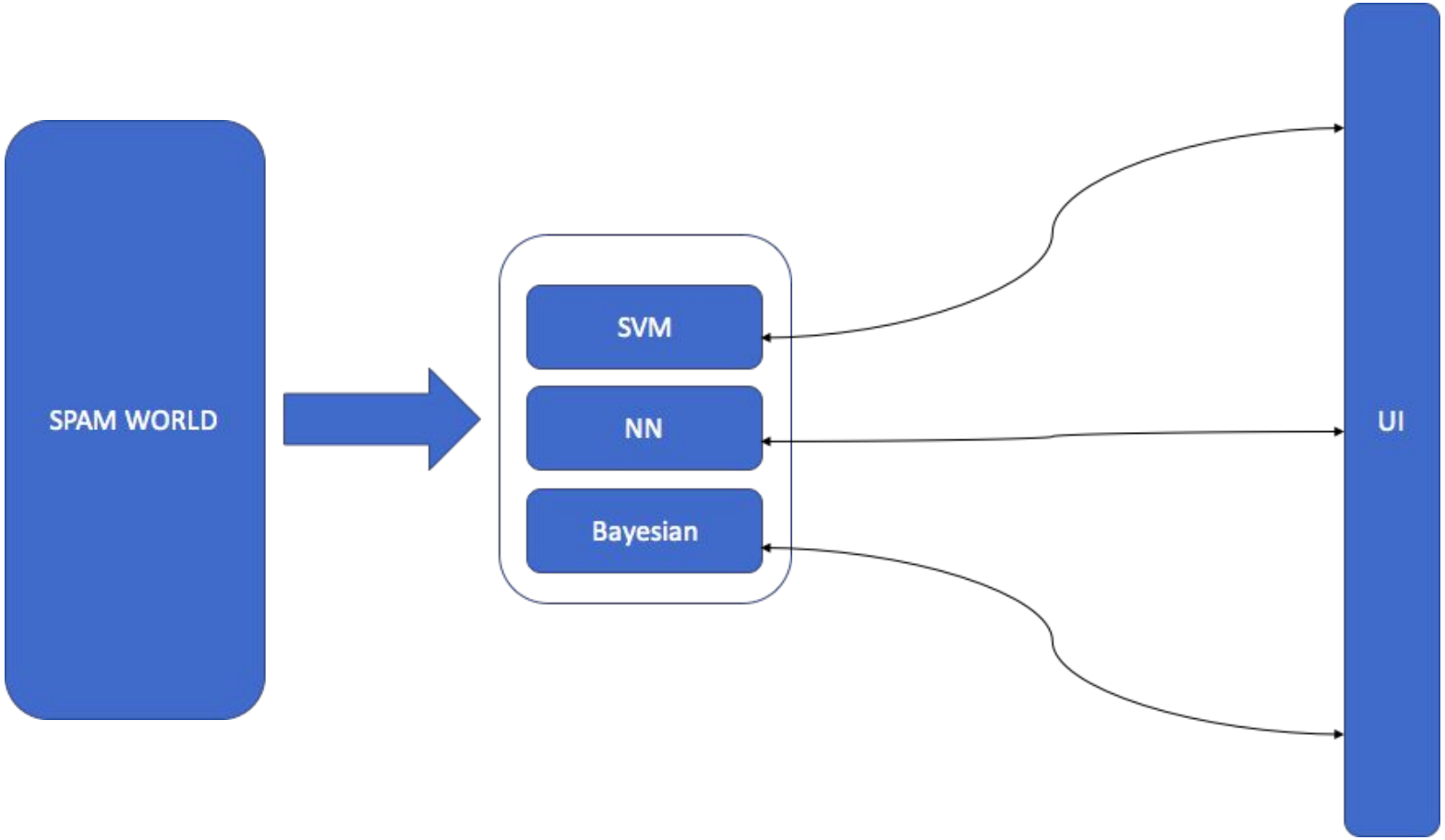
Versatile

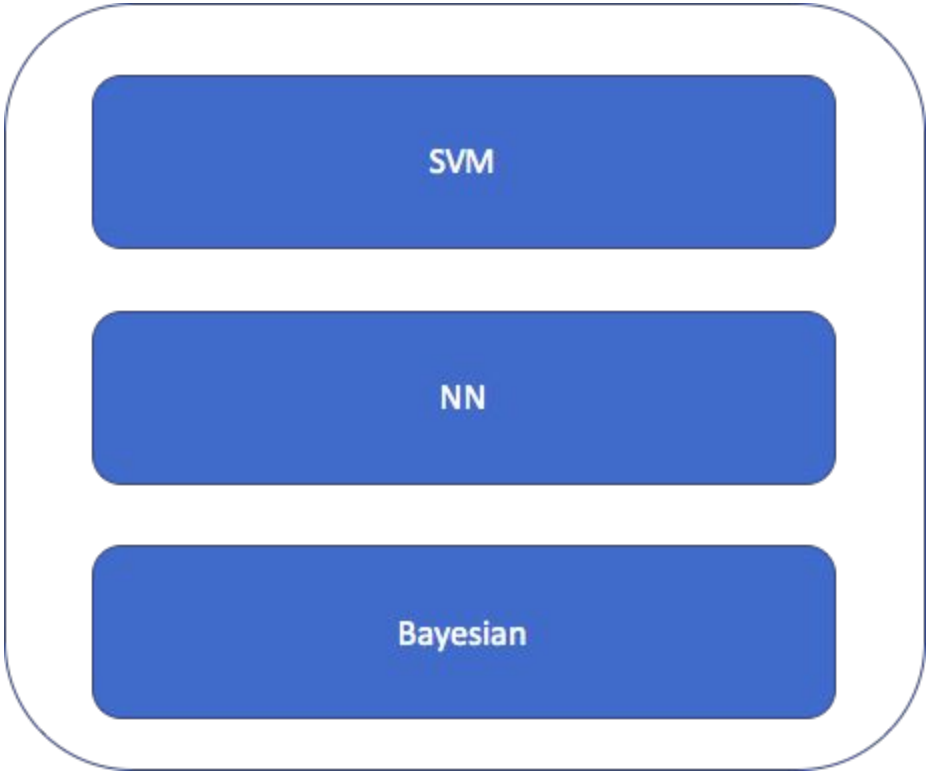
Effective

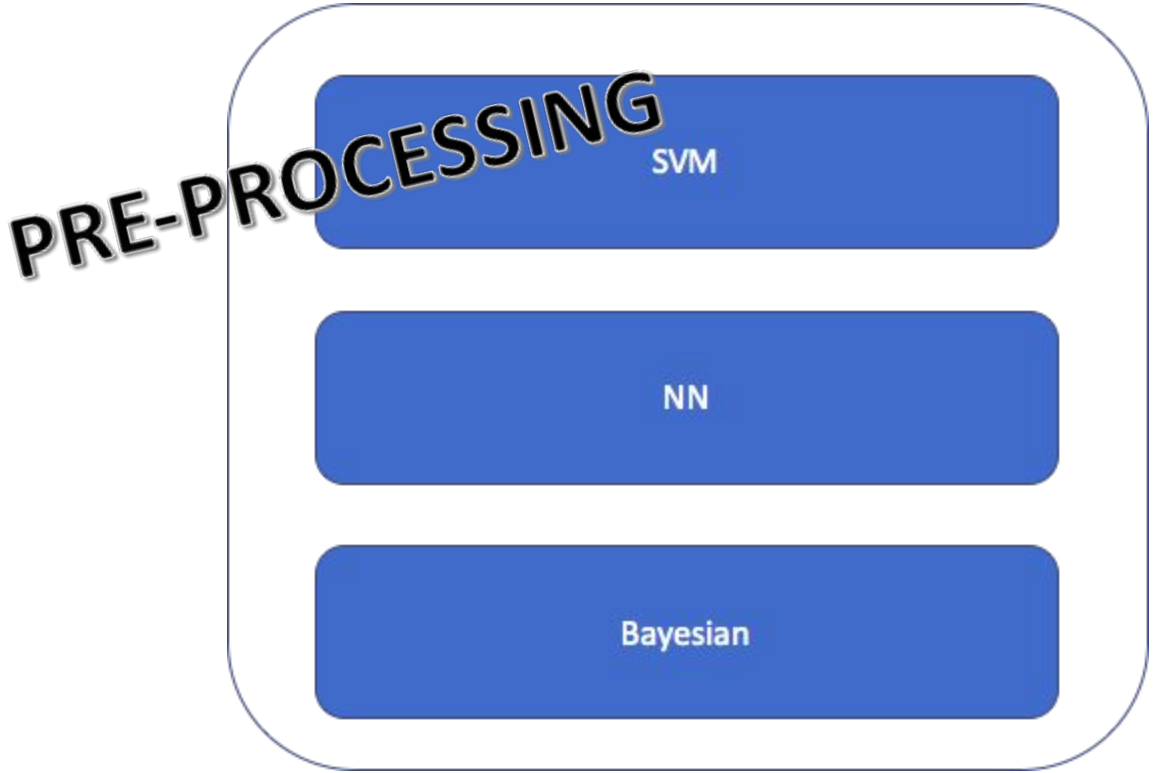


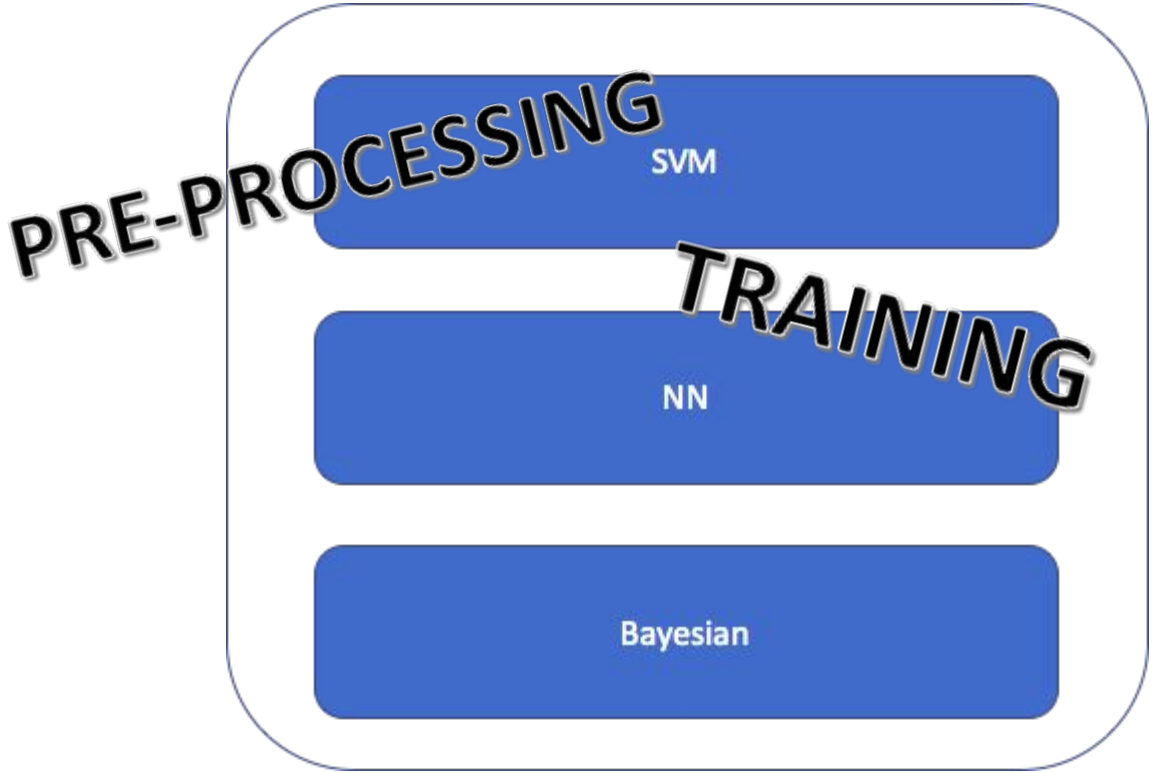

SPAM WORLD
OUR SYSTEM

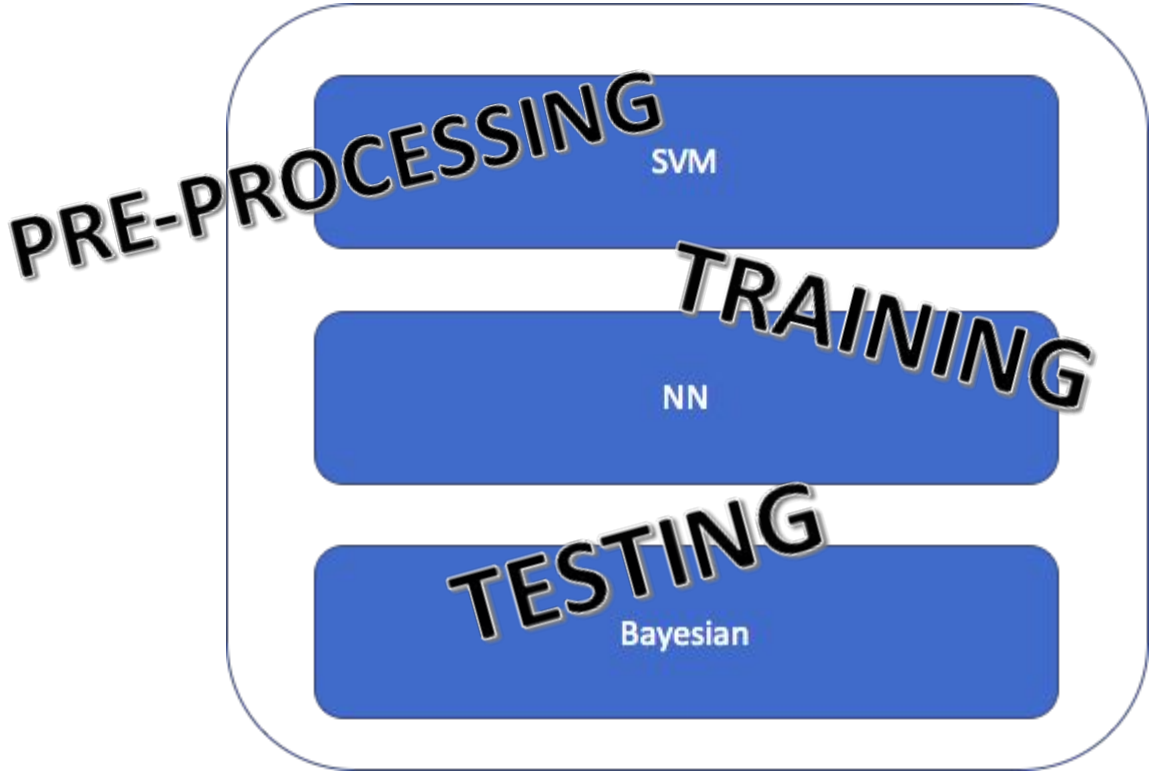

SPAM WORLD
SVM
NN
Bayesian


SPAM WORLD
SVM
NN
Bayesian
UI


SVM
NN
Bayesian


PRE-PROCESSING
SVM
NN
Bayesian


PRE-PROCESSING
TRAINING
SVM
NN
Bayesian


PRE-PROCESSING
SVM
TRAINING
NN
TESTING
Bayesian

Data Preprocessing - NLP

- Extracted the text data into data frame . (Email spam and spam emails extracted from gmail)
- Applied word tokenization on the extracted raw data using natural language processing toolkit module.
- Removed stop words from text.
- Removed some common special characters as (“, “, “.”, “-”).
- Used this processed data for feature extraction.

Data Preprocessing - SVM

- Feature extraction using information gain.
- Applied the concept of information gain from information theory
- Information gain have been used as feature ranking technique for numerous text applications.
- Information gain = expected information from the required class - information of an attribute to split this class

$Info(D) = -\sum p \log(p)$

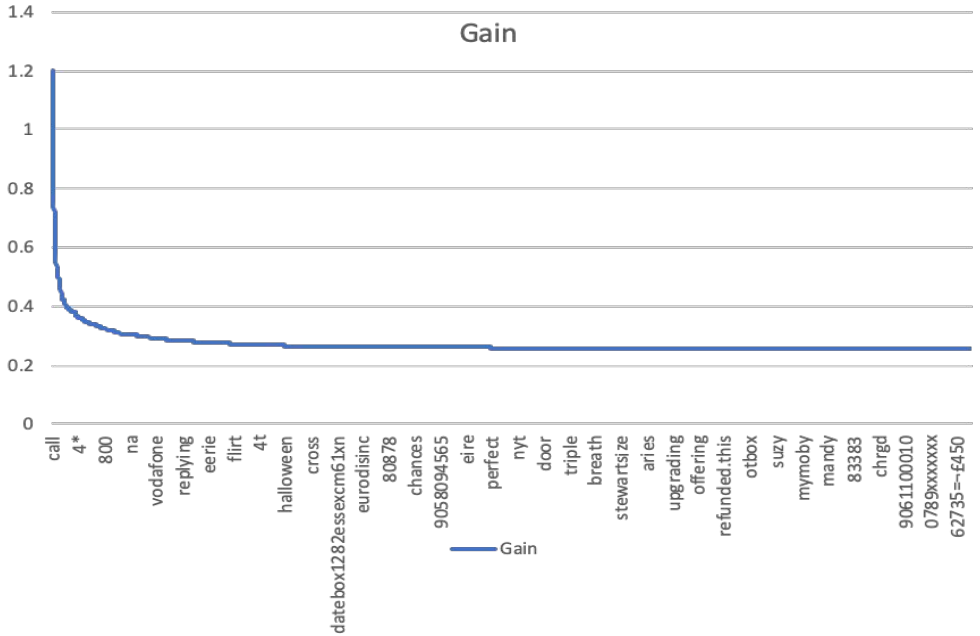
$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

$$Gain(A) = Info(D) - Info_A(D)$$

- Words extracted from spam class emails/messages. Each word considered as feature.
- Features with highest information gain considered as features for spam classification.

Information Gain

- Extracted 1000 features with highest information gain.
- Also, tested with different range of features ranked based on the information gain



Support Vector Machine

- Widely used in text classification applications .
- Designed Support Vector Machine algorithm from scratch.
- Applied spam classification on the SVM algorithm with Linear kernel.
- Weights and bias information calculated for support vector machine Algorithm.

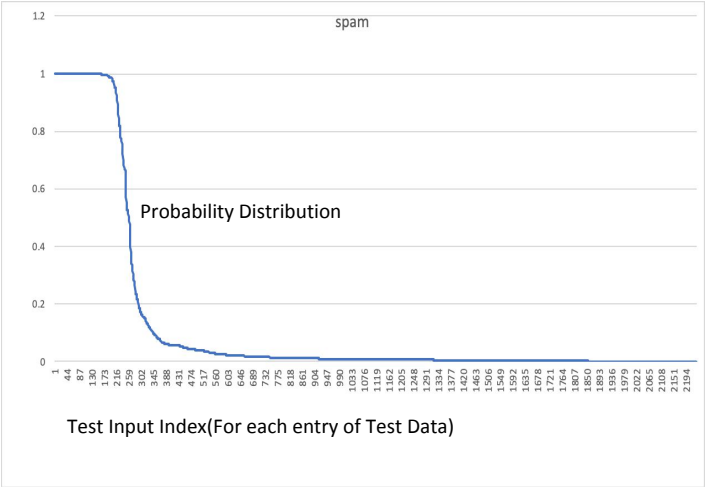
$wTx + b \geq 0$ for di = +1 $wTx + b < 0$ for di = –1

– w is a weight vector– x is input vector – b is bias

- Compared results with existing models in python .
- Tested on real time data using user interface.

SVM Result:

- Results verified based on the metrics accuracy and confusion Matrix.
- Accuracy 87% (Compared metrics with Sklearn SVM Model result).
- Accuracy is almost same.
- Graph shows the Probability Distribution of SVM classification For Spam



Neural Network

- Vectorization of text is using TF-IDF.
- TF(Term frequency) - Frequency of the words in text corpus.
- IDF(Inverse Document frequency) - Computes weights of rare words.
- Combining these to get TF-IDF score for each word in the corpus.

Neural Network

- **Forward-Backward Propagation :**

Parameters				
Epoch	*Learning Rate*	*Input Neurons*	*Hidden Layers*	*Hidden Layer Neurons*
1000	0.05	Same as Features Count	1	1500

Data Preprocessing - Bayesian

- Bayes Rule :

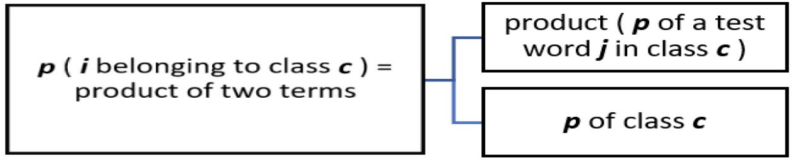
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

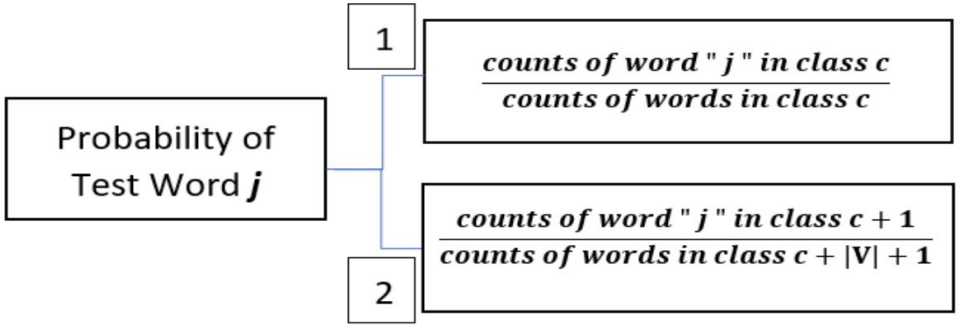
Roadmap to finding probability to predict label of a text

- Preparing the Model :
  - Data Preprocessing
  - Feature Engineering
    - Bag of Words(Spam and Not Spam)
  - Training it on the model
  - Accuracy : 91% (Good Right?)
- Finding Probability of a Test text
  - Preprocessing of text
  - Tokenization

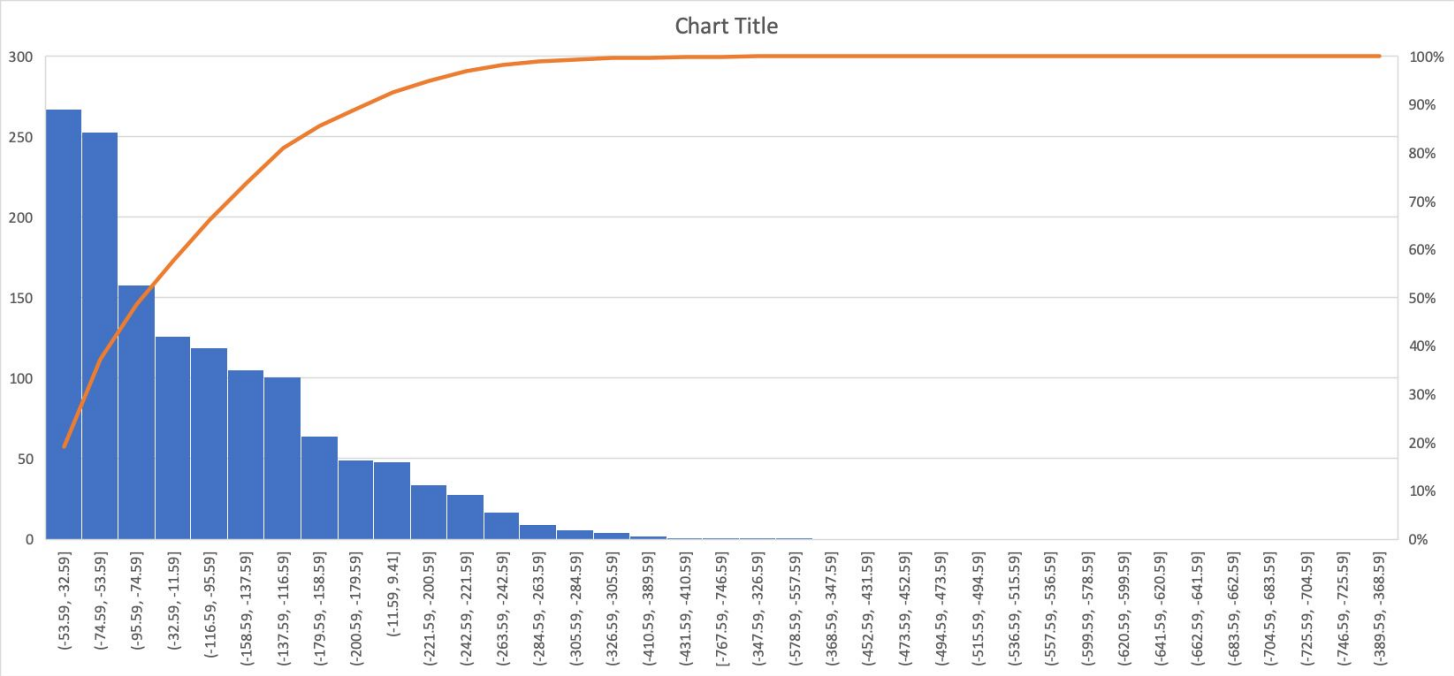
- Using Probability to predict label



$$p \text{ of class } c = \frac{\textit{total number of training examples belonging to class } c}{\textit{total number of examples in the training set}}$$

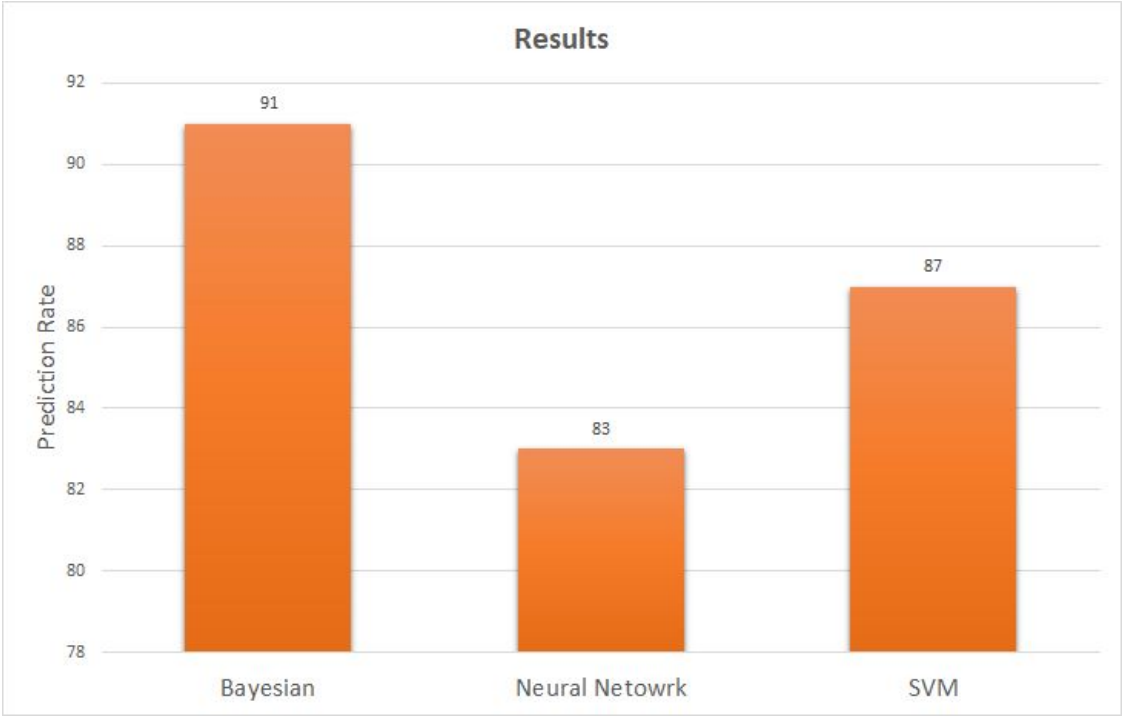


- Label for the text is decided depending on the max probability of the two classes



Probability scale of the train data to decide the level of spam - High, Medium, Low and No Spam

Results


Results
Prediction Rate
92
90
88
86
84
82
80
78
91
83
87
Bayesian
Neural Netowrk
SVM

Data Set Considerations 1

1. **Lingspam**
   - Public dataset which contains *Spam and Ham Mails*.
   - Thousands of *text corpus* classified as Spam or not.
2. **Gmail Spam**
   - To *increase* the Spam count in the dataset and to predict more text.
   - We created a tool to acquire spam mails from mailbox.

Issues:

1. *HTML & Image:* Most modern mails are either image or in HTML format.
2. Text Corpus and not classified by Line - which we do not need.

Data Set Considerations 2

Our Requirement:

- Single Sentences classified as Spam or not.

SMS Spam Collection

- SMS text classified as Spam or not.

Applications

- **Filtering:**
  - Remove spam messages or emails.

- **Detection:**
  - Detect if a text will be filtered as spam or not.
  - Allow the genuine sender to know, if the message will end up in Junk folder.

Demo