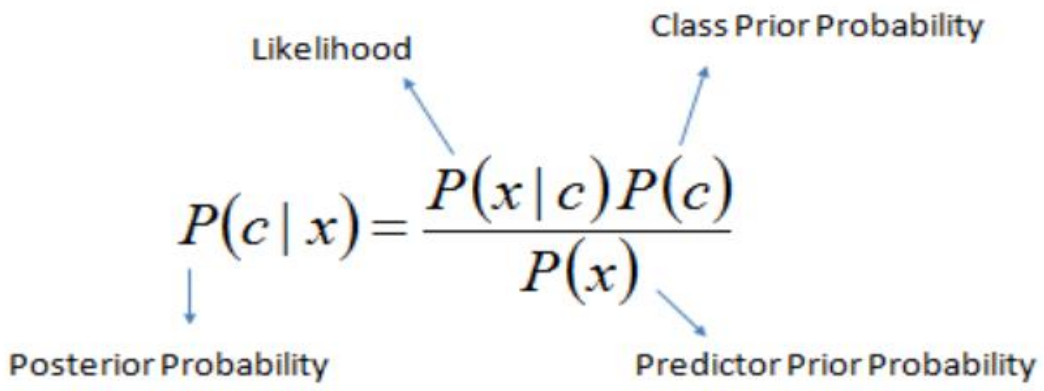
LAB-H: WRITE A PROGRAM TO DETECT SPAM & NOT SPAM USING NAIVE BAYES CLASSIFIER.

OBJECTIVE: To implement spam detection using naïve bayes classifier

THEORY:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Naive Bayes is a probabilistic algorithm based on the Bayes Theorem used for email spam filtering in data analytics. If you have an email account, we are sure that you have seen emails being categorised into different buckets and automatically being marked important, spam, promotions, etc.

FORMULA:

One of the most simple yet powerful classifier algorithms, Naive Bayes is based on Bayes' Theorem Formula with an assumption of independence among predictors. Given a Hypothesis A and evidence B, Bayes' Theorem calculator states that the relationship between the probability of Hypothesis before getting the evidence P(A) and the probability of the hypothesis after getting the evidence P(A|B) is:

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

Here:

- A, B = events
- P(A|B) = probability of A given B is true
- P(B|A) = probability of B given A is true
- P(A), P(B) = the independent probabilities of A and B

Detecting Email Spam

Modern spam filtering software continuously struggles to categorise the emails correctly. Unwanted spam & promotional communication is the toughest of them all. Spam communication algorithms must be iterated continuously since there is an ongoing battle between spam filtering software and anonymous spam & promotional mail senders. Naive Bayes Algorithm in data analytics forms the base for text filtering in Gmail, Yahoo Mail, Hotmail & all other platforms.

Like Naive Bayes, other classifier algorithms like Support Vector Machine, or Neural Network also get the job done! Before we begin, here is the dataset for you to download:

Email Spam Filtering Using Naive Bayes Algorithm
This would be a zipped file, attached in the email.

Usage Of Naive Bayes Algorithm:

- News Classification.
- Spam Filtering.
- Face Detection / Object detection.
- Medical Diagnosis.
- Weather Prediction, etc.

ALGORITHM:

STEP 1: Data Processing

STEP 2: Fitting the naïve bayes classifier to the training set.

STEP 3: Predicting the test set results**.**

PROGRAM:

```
train_spam = ['send us your password', 'review our website', 'send your password', 'send us your account']
train_ham = ['Your activity report','benefits physical activity', 'the importance vows']
test_emails = {'spam':['renew your password', 'renew your vows'], 'ham':['benefits of our account', 'the importance of physical activity']}
vocab_words_spam = []

for sentence in train_spam:
sentence_as_list = sentence.split()
for word in sentence_as_list:
vocab_words_spam.append(word)

print(vocab_words_spam)

vocab_unique_words_spam = list(dict.fromkeys(vocab_words_spam))
print(vocab_unique_words_spam)


dict_spamicity = {}
for w in vocab_unique_words_spam:
emails_with_w = 0 # counter
for sentence in train_spam:
if w in sentence:
emails_with_w += 1

print(f"Number of spam emails with the word {w}: {emails_with_w}")
total_spam = len(train_spam)
spamicity = (emails_with_w + 1) / (total_spam + 2)
print(f"Spamicity of the word '{w}': {spamicity} \n")
dict_spamicity[w.lower()] = spamicity
print(dict_spamicity)
```

OUTPUT:

```
PROBLEMS 18   OUTPUT   DEBUG CONSOLE   TERMINAL   SQL CONSOLE   Python Debug Console

(orange3) tilak@tilak:~/Desktop/ai$  cd /home/tilak/Desktop/ai ; /usr/bin/env /home/til
ak/anaconda3/envs/orange3/bin/python /home/tilak/.vscode/extensions/ms-python.python-20
21.8.1159798656/pythonFiles/lib/python/debugpy/launcher 39565 -- /home/tilak/Desktop/ai
/spam_filter.py
['send', 'us', 'your', 'password', 'review', 'our', 'website', 'send', 'your', 'passwor
d', 'send', 'us', 'your', 'account']
['send', 'us', 'your', 'password', 'review', 'our', 'website', 'account']
Number of spam emails with the word send: 3
Spamicity of the word 'send': 0.6666666666666666

Number of spam emails with the word us: 2
Spamicity of the word 'us': 0.5

Number of spam emails with the word your: 3
Spamicity of the word 'your': 0.6666666666666666

Number of spam emails with the word password: 2
Spamicity of the word 'password': 0.5

Number of spam emails with the word review: 1
Spamicity of the word 'review': 0.3333333333333333

Number of spam emails with the word our: 4
Spamicity of the word 'our': 0.8333333333333334

Number of spam emails with the word website: 1
Spamicity of the word 'website': 0.3333333333333333

Number of spam emails with the word account: 1
Spamicity of the word 'account': 0.3333333333333333

{'send': 0.6666666666666666, 'us': 0.5, 'your': 0.6666666666666666, 'password': 0.5, 'r
eview': 0.3333333333333333, 'our': 0.8333333333333334, 'website': 0.3333333333333333, '
account': 0.3333333333333333}
(orange3) tilak@tilak:~/Desktop/ai$
```

CONCLUSION

- Ability to handle an extremely large number of features (words)
- Very simple
- Rarely overfits