Training a Naive Bayes classifier using Laplace correction

ASSIGNMENT-2

CS60050-Machine-Learning

Group - 1
Haasita Pinnepu (19CS30021)
Piriya Sai Swapnika (19CS30035)

Abstract:

This report is submitted as part of the second assignment to the course CS60050 – Machine Learning, IIT Kharagpur. The dataset contained data records of the writing styles of various authors. The objective is to predict the presence of a specific word in the writings of an author using these records. We used Naïve Bayes classifiers, using Laplace correction, to perform the tasks. This paper briefly explains the procedure we followed to achieve the goal and the results obtained.

1. Assessing DataSet:

The dataset is provided to us as a CSV file. We accessed the data using pandas.read_csv API and created a *pandas.DataFrame* object for the dataset handling. This is done because it is easy to handle and manipulate data using such objects.

Fig1: Raw DataSet

id	text	author
id26305	This process, however, afforded me no means of ascertaining the dimensions of my dungeon; as I might make its circuit, and return to the point whence I set out, without being aware of the fact; so perfectly uniform seemed the wall.	EAP
id17569	It never once occurred to me that the fumbling might be a mere mistake.	HPL
id11008	In his left hand was a gold snuff box, from which, as he capered down the hill, cutting all manner of fantastic steps, he took snuff incessantly with an air of the greatest possible self satisfaction.	EAP
id27763	How lovely is spring As we looked from Windsor Terrace on the sixteen fertile counties spread beneath, speckled by happy cottages and wealthier towns, all looked as in former years, heart cheering and fair.	MWS
id12958	Finding nothing else, not even gold, the Superintendent abandoned his attempts; but a perplexed look occasionally steals over his countenance as he sits thinking at his desk.	HPL
id22965	A youth passed in solitude, my best years spent under your gentle and feminine fosterage, has so refined the groundwork of my character that I cannot overcome an intense distaste to the usual brutality exercised on board ship: I have never believed it to be necessary, and when I heard of a mariner equally noted for his kindliness of heart and the respect and obedience paid to him by his crew, I felt myself peculiarly fortunate in being able to secure his services.	MWS
id09674	The astronomer, perhaps, at this point, took refuge in the suggestion of non luminosity; and here analogy was suddenly let fall.	EAP
id13515	The surcingle hung in ribands from my body.	EAP

Fig2:DataSet after using pandas API

	id	text	author
0	id26305	This process, however, afforded me no means of...	EAP
1	id17569	It never once occurred to me that the fumbling...	HPL
2	id11008	In his left hand was a gold snuff box, from wh...	EAP
3	id27763	How lovely is spring As we looked from Windsor...	MWS
4	id12958	Finding nothing else, not even gold, the Super...	HPL
...	...	...	...
19574	id17718	I could have fancied, while I looked at it, th...	EAP
19575	id08973	The lids clenched themselves together as if in...	EAP
19576	id05267	Mais il faut agir that is to say, a Frenchman ...	EAP
19577	id17513	For an item of news like this, it strikes us i...	EAP
19578	id00393	He laid a gnarled claw on my shoulder, and it ...	HPL

19579 rows × 3 columns

2. Tokenization:

Word Tokenization is the most commonly used tokenization algorithm. It splits a piece of text into individual words based on a certain delimiter. Since each example is a raw text, we split the text into words. We used the python `re' package (re.findall(\[a-z0-9]+", text.tolower())) for this.

3. Removal of unwanted words:

Once tokenization was done, we removed all the uninformative words (also referred to as stopwords) like articles, prepositions, verbs etc using nltk. The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language.

Fig3:DataSet after removing stopwords

Ind	Type	Size	Value
0	str	4	aaem
1	str	2	ab
2	str	5	aback
3	str	5	abaft
4	str	7	abandon
5	str	9	abandoned
6	str	10	abandoning
7	str	11	abandonment
8	str	6	abaout
9	str	6	abased
10	str	9	abasement

4. Converting the dataset into a Scipy matrix:

In this step, we extracted features from the raw texts. We created an r*c *scipy.sparse.csr* matrix sA where r is the number of examples and c is the size of the vocabulary consisting of distinct words present in the dataset. Each row corresponds to an example of the dataset and each column corresponds to a word in the vocabulary. sA_{ij} = 1 if and only if the jth word is present in the text of the i example.

5. Training a Naive Bayes Classifier on the matrix:

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. All naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. The multinomial naïve Bayes classifier becomes a linear classifier when expressed in log-space.

$$\begin{aligned}\log p(C_k \mid \mathbf{x}) &\propto \log\left(p(C_k)\prod_{i=1}^{n} {p_{ki}}^{x_i}\right)\\ &= \log p(C_k) + \sum_{i=1}^{n} x_i \cdot \log p_{ki}\\ &= b + \mathbf{w}_k^\top \mathbf{x}\end{aligned}$$

where $b = \log p(C_k)$ and $w_{ki} = \log p_{ki}$.

We get the training_data and test_data by randomly splitting the dataset in a 70/30 ratio.

Metrics:

- ➔ True Positives (TP) are the Authors that truly wrote the string(sentence).
- ➔ True Negatives (TN) are the Authors that truly DID NOT write the string.
- ➔ False Positives (FP) are the Authors that have truly NOT written the string, but based on the test, they were falsely (False) denoted as the writers of it (Positives).
- ➔ False Negatives (FN) are the Authors that have truly written the string but based on the test, they were falsely (False) denoted that the DID NOT(Negative).

1. Accuracy:
It's the ratio of the correctly labelled subjects to the whole pool of subjects. Accuracy is the most intuitive one.
Accuracy = (TP+TN)/(TP+FP+FN+TN)

2. Precision:
Precision is the ratio of the correctly +ve labelled by our program to all +ve labelled.
Precision = TP/(TP+FP)

3. Sensitivity:
Sensitivity(recall) is the ratio of the correctly +ve labelled by our program to all who are true cases in reality.
Sensitivity = TP/(TP+FN)

4. F-Score
F Score considers both precision and recall. It is the harmonic mean(average) of the precision and recall.
F Score = 2*(Recall * Precision) / (Recall + Precision)

5. Specificity
Specificity is the correctly -ve labelled by the program.
Specificity = TN/(TN+FP)

6. Calculating the Performance Metrics of the model:

The **Test Accuracy** of the model: 31.306097183534508
The **95% confidence interval** of accuracy: 0.009876845932606494
The **Precision**: 0.20886513613653976
The **f-Score**: 0.20147102539402986
The **Sensitivity**: 0.21646695
The **Specificity:** 0.67535484

7. Calculating the Performance Metrics by Laplace Correction:

The **Test Accuracy** of the model: 83.25595790776849
The **95% confidence interval** of accuracy: 0.009103542185417632
The **Precision**: 0.837798567247146
The **f-Score**: 0.8325304246173398
The **Sensitivity**: 0.85233862
The **Specificity:** 0.87380645