Now in machine learning there are different processes before actually creating a model

1. Data collection
2. Data pre-processing
3. Feature extraction
4. Model creation or intialization
5. Evaluation
6. Updating
7. Deploying
8. Maintaince

NOW FORsentiment\_model.ipynb:

First of all I imported the libraries I used for the creation of the models

- then I loaded the dataset(ecoprocessed) which was inside a folder named “data”

What “data.head()” does is that it shows some of the dataset starting from the head…. something I like doing to see what the dataset looks like call it visualization, Data visualization

Next was something about null columns:

This is to show the details of my dataset, things like :

If there are null dataset well that’s what I was looking for, because if there are null values (empty spaces, columns) in the dataset I will affect the model during training

Next is train\_test\_split:

This is just when this becomes beautiful

Now in you dataset what I used was “review”, “divison”

The way train test split works is that:

It splits the data set into two parts which are

“train”, “test” I gave the testsize to be = 0.2 which is 20

What that means is the the remaining 80 % will be used by the model to learn patterns, insights, just like the way we read books after that we used the remaing 20% “test” to check for the accuracy of the dataset because the model hasn’t seen that 20% before so lets say its going for a test or exam

Next you’ll see Countvectorizer and multinomial initialization

For countvectorizer it makes text processing easy, it converts raw text into numerical formats machines learning algorithms can understand in our case the algorithm is (multinomial)

For multinomial :

| Text | Sentiment |

|---------------------------------------------------|---------------|

| This movie is fantastic! | positive |

| I hated every minute of it. | negative |

| The acting was superb, but the plot was weak. | mixed |

| An absolute masterpiece. | positive |

This is a text dataset:

In the text:

1. Multinomial Naive Bayes analyzes this dataset to learn the statistical properties of 2.the words in positive and negative movie reviews.

3.It calculates the frequency of each word in both types of reviews.

4.It also computes the prior probabilities of positive and negative reviews.

Now:

The algorithm constructs a vocabulary by collecting all unique words from the dataset. In this case, our vocabulary might include words like "fantastic", "hated", "acting", "superb", "plot", "weak", "absolute", "masterpiece".

Calculating Probabilities:

For each word in the vocabulary, the algorithm calculates the probability of observing that word given the sentiment (positive or negative). For example:

("fantastic"∣"positive")=12P("fantastic"∣"positive")=21​ because the word "fantastic" appears once in positive reviews out of a total of two positive reviews.

("hated"∣"negative")=11P("hated"∣"negative")=11​ because the word "hated" appears once in negative reviews out of a total of one negative review.

It also calculates the prior probabilities of positive and negative reviews.

Classification Phase:

Suppose we have a new, unseen movie review: "The plot was engaging, but the acting fell flat."

Multinomial Naive Bayes calculates the likelihood of this review belonging to each sentiment category based on the frequencies of words in the review.

It combines this likelihood with the prior probabilities of positive and negative reviews to compute the posterior probabilities of each sentiment category.

The class with the highest posterior probability is chosen as the predicted sentiment for the review.

Prediction:

Based on the computed posterior probabilities, Multinomial Naive Bayes predicts whether the new review has positive or negative sentiment.

This is how multinomial Naive Bayes works

Next is X\_test\_vectorised, what this does is after initializing the model I have to converts text data into numerical formats

The next line is we used the remaining 20 percent to make predictions

Note: the reason we did this is because we transformed the training dataset so therefore we have to transform the test dataset

Next line Is simple we print accuracy

Save the model to a file named “Sentiment\_Model\_CountVectorizer.pkl”

And save the vectorizer as well to a file named “CountVectorizer.pkl”

This next part is simple we run test on the model we created by making new inputs,

We made it possible so that we will be able to see how much “positive”, or “neutral”, or “negative” an input can be.