Task 2: Document Classification (using traditional ML techniques)

Dataset:

Choose any one text dataset from here. Create a random sample of 20k from the chosen set for the task.

Dataset Chosen: Luxury_Beauty

Dataset Details:-

Luxury Beauty – Reviews (574,628 reviews) Metadata (12,308 products)

Sub-tasks:

1. **Define your train-val split. [Report the split chosen.]**

Dataset is uploaded in google drive and loaded into colab notebook using gdown command.

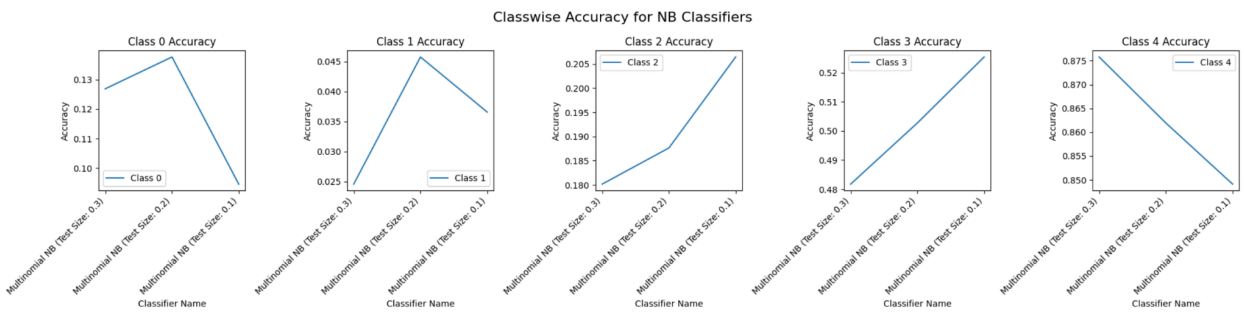
We conducted experiments with three different train-validation splits: 0.1, 0.2, and 0.3. These splits indicate the proportion of the dataset allocated to the validation set, with the remainder assigned to the training set. After evaluating the classifiers for each split, we observed variations in performance metrics across the different splits.

Upon analysis, we found that the 0.2 split consistently yielded the most favorable results across various performance metrics. Therefore, we would select the 0.2 split as the preferred choice for analysis and model evaluation. Results are shown for each split-case.

```
test_sizes = [0.3, 0.2, 0.1]

# Initialize ClassifierResults object
classifier_results_5_classes = ClassifierResults()
print("Training for 5 class classification")
for test_size in test_sizes:
    # Split the data
    train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=test_size, random_state=64)
```

Example of multiple splits and reporting class wise accuracy **on each split** :-



2. **Which set of terms best describes your corpus? How did you arrive at it?**

1. Understanding the Corpus: Begin by reviewing the content and context of the corpus to grasp its themes, topics, and domain.
2. Tokenization: Break down the text data into individual words or tokens to facilitate analysis.
3. Stemming and Lemmatization: Apply stemming and lemmatization techniques to normalize the words, reducing inflected or derived words to their base or root form. This step helps in consolidating variations of the same term, providing a more accurate representation of the vocabulary.
4. Term Frequency Analysis: Utilize techniques like TF-IDF or simple term frequency analysis to identify the most common terms in the corpus, revealing prevalent vocabulary and key topics.

```
Top 10 most frequent words:
skin: 15111
product: 10405
like: 10141
use: 9078
color: 6402
hair: 5950
really: 5772
great: 5599
good: 5557
face: 5551
```

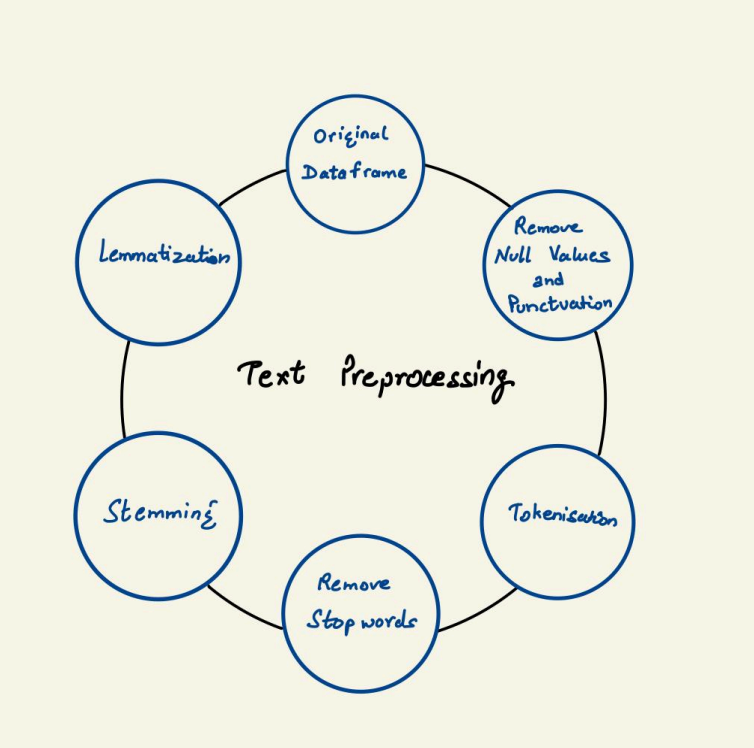
```
Top TF-IDF scores keywords:
zz
fatchunky
farsali
farsighted
farther
fascinated
fascinating
fashion
fashionable
fashioned
```

Term Frequency-Inverse Document Frequency (TF-IDF):

TF-IDF is a statistical measure that evaluates the importance of a term within a document relative to a collection of documents (corpus).
It combines two components:
Term Frequency (TF): Measures how often a term appears in a document.
Inverse Document Frequency (IDF): Measures how unique or rare a term is across the entire corpus.
TF-IDF score for a term in a document is calculated as the product of its TF and IDF scores.
High TF-IDF scores are assigned to terms that appear frequently in a document but rarely in other documents in the corpus, indicating their significance to the document.

3. **Define a text preprocessing pipeline, i.e., stopword removal, lower casing, punctuation removal etc. [Report your text preprocessing pipeline in the report].**

Text Preprocessing Pipeline :-



1. Original DataFrame: This is the starting point of the data preprocessing pipeline, where you have your raw dataset containing reviews in text format.

	overall	verified	reviewTime	reviewerID	asin	style	reviewerName	reviewText	summary	unixReviewTime
17992	3.0	False	02 10, 2014	A216NSW58Q3SCJ	B005COP4FA	{'Color:': ' Medium'}	Christina Conte	First of all, I have to say that I was ready t...	Great for smooth skin with discoloration. Not ...	1391990400
16033	1.0	True	06 20, 2014	A1IIT6KQNH7QX1	B004L8J15C	{'Size:': ' 1.35 fl. oz'}	Thatgurl	I had THE WORSE reaction literally burned my s...	dont.just dont.	1403222400
26695	2.0	False	11 15, 2015	A1X3ESYZ79H59E	B00M0V083Q	NaN	pepper	This is a really small tube of moisturizer. It...	didn't do anything for my dark spots	1447545600

2. Remove Null Values and Punctuations: In this stage, any rows with null values are removed from the dataset to ensure data cleanliness. Additionally, punctuation marks are stripped from the text to focus on the raw words.

	overall	reviewText
21991	5	Great coverage and lasting color
33610	4	I don't have facial acne but get severe cystic...
12248	5	Boyfriend loved it. I love it. He smells great...
8398	5	BEST STUFF ON THE MARKET!
5443	5	Jane Iredale is the best. Nothing else need b...

	overall	reviewText
7074	5	I wish I had heard about this product earlier ...
13587	5	This is great hair product
17791	2	I am not a fan of this polish It has a relati...
20153	5	To be honest I am not a big fan of using hand ...
5280	5	One of the few Jack Black products I would rec...

3. Tokenization: Tokenization involves breaking down the text into individual words or tokens. This step essentially splits the text into meaningful units, which serves as the basis for further analysis.

	overall	reviewText	label	token
29831	4	this creamy white lotion is lightly fragranced...	1	[this, creamy, white, lotion, is, lightly, fra...
25386	3	this is an ok coverup but takes two coats to c...	1	[this, is, an, ok, coverup, but, takes, two, c...
23247	5	i see this is no longer offered by molton brow...	1	[i, see, this, is, no, longer, offered, by, mo...
16078	5	was looking for something that didnt contain p...	1	[was, looking, for, something, that, didnt, co...

4. Remove Stopwords: Stopwords are common words in a language (e.g., "the", "is", "and") that often do not carry significant meaning in text analysis. Removing stopwords helps reduce noise in the data and focuses on more meaningful words.

	overall	reviewText	label	token	no_stopword
25919	4	this foundation feels nice going on and has a ...	1	[this, foundation, feels, nice, going, on, and...	[foundation, feels, nice, going, sheer, look, ...
2356	2	3c4a coil curls\n\nit has a pleasant neutral s...	0	[3c4a, coil, curls, it, has, a, pleasant, neut...	[3c4a, coil, curls, pleasant, neutral, scent, ...
16061	4	i like bright colors for spring and summer so ...	1	[i, like, bright, colors, for, spring, and, su...	[like, bright, colors, spring, summer, refresh...
29958	5	a few minutes ago as i was preparing to sit do...	1	[a, few, minutes, ago, as, i, was, preparing, ...	[minutes, ago, preparing, sit, write, review, ...
26592	4	i think this product works okay after using i...	1	[i, think, this, product, works, okay, after, ...	[think, product, works, okay, using, bit, safe...

5. Stemming: Stemming is a text normalization technique where words are reduced to their root or base form by removing suffixes. This process helps in reducing the dimensionality of the feature space by grouping together words with the same root.

token	no_stopword	stemming
[i, am, an, avid, user, of, sunless, tanning, ...	[avid, user, sunless, tanning, products, used,...	[avid, user, sunless, tan, product, use, sinc,...
[i, absolutely, love, this, cleanser, have, be...	[absolutely, love, cleanser, purchasing, years...	[absolut, love, cleanser, purchas, year, issu,...
[this, is, a, fantastic, bb, cream, and, imo, ...	[fantastic, bb, cream, imo, great, foundation,...	[fantast, bb, cream, imo, great, foundat, matu...
[this, is, a, wonderful, lotion, smells, yummy...	[wonderful, lotion, smells, yummy, stickey, ma...	[wonder, lotion, smell, yummi, stickey, make, ...

6. Lemmatization: Lemmatization is another text normalization technique that goes a step further than stemming. It reduces words to their base or dictionary form (lemma) using vocabulary analysis. This stage helps in further standardizing the text data and improving interpretability.

no_stopword	stemming	lemmatization
[got, basis, reviews, yes, good, awesome, look...	[got, basi, review, ye, good, awesom, look, so...	[get, basi, review, ye, good, awesom, look, so...
[get, rid, redness, within, hours, doesnt, alw...	[get, rid, red, within, hour, doesnt, alway, g...	[get, rid, red, within, hour, doesnt, alway, g...
[think, mineral, sunscreens, better, skin, che...	[think, miner, sunscreen, better, skin, chemic...	[think, miner, sunscreen, good, skin, chemic, ...
[hair, smooth, silky, using, silk, infusion, r...	[hair, smooth, silki, use, silk, infus, reform...	[hair, smooth, silki, use, silk, infus, reform...

---

4. **Design a multi-class classifier to predict the actual rating using a bag of words with different weight calculation mechanisms, for each of the following models.**

4.1 CountVectorizer (`from sklearn.feature_extraction`):-

- CountVectorizer is a text processing technique used to convert a collection of text documents into a matrix of token counts.
- It represents each document as a vector where each element of the vector corresponds to the count of a particular word (token) in the document.
- Each document is represented as a sparse vector where each element indicates the count of the corresponding word in the document.
- This approach ignores the order of words in the document and **only considers the frequency of each word.**

4.2 Binary Weighting (`from sklearn.feature_extraction`):-

- Binary weighting is a simple technique used in text analysis where each term (word) in a document is represented as a binary value indicating its presence or absence.

- Instead of counting the frequency of each term, **binary weighting only considers whether a term is present (1) or absent (0) in the document.**
- It's commonly used in binary classification tasks or when the exact frequency of terms is not important, and only their presence matters.

4.3 TF-IDF Vectorization (`from sklearn.feature_extraction`):-

- It combines information about the frequency of terms in a document (TF) with their rarity across multiple documents (IDF).
- The TF-IDF score of a term in a document increases with the number of times it appears in the document (TF) and decreases with the frequency of the term across all documents in the corpus (IDF).
- It aims to **give higher weights to terms that are frequent in a document but rare in other documents,** thus capturing the importance of terms in a document relative to the entire corpus.
- TF-IDF vectors are typically normalized to unit length to make them invariant to the length of documents.

---

5. **Provide comparative performance for each weighing mechanism using accuracy, and confusion matrix (class-wise).**
    a. **Naive Bayes classifier**.
        i. Gaussian NB.
        ii. Multinomial NB.
    b. **Decision tree**.
        i. Criteria: entropy
        ii. Criteria: gini

    c. Random Forest
        i. No of trees = 20
        ii. No of trees = 50
        iii. No of trees = 100

    Weighing mechanism used are :-

    CountVectorization - This works slightly better than the other two.
    Binary Weighting Mechanism
    TF-IDF Vectorization

Class-wise Accuracy evaluation of different Weight calculation:-

- Weight calculation - CountVectorization :-

Classifier	Overall Accuracy	Class 0 Accuracy	Class 1 Accuracy
Gaussian NB (Test Size: 0.2)	0.26	0.54	0.29
Multinomial NB (Test Size: 0.2)	0.65	0.16	0.03
Decision Tree (Test Size: 0.2) (entropy)	0.63	0.42	0.34
Decision Tree (Test Size: 0.2) (gini)	0.63	0.35	0.31
Random Forest (Test Size: 0.2) (20)	0.68	0.37	0.28
Random Forest (Test Size: 0.2) (50)	0.69	0.37	0.28
Random Forest (Test Size: 0.2) (100)	0.69	0.37	0.28

Classifier	Class 2 Accuracy	Class 3 Accuracy	Class 4 Accuracy
Gaussian NB (Test Size: 0.2)	0.32	0.33	0.2
Multinomial NB (Test Size: 0.2)	0.19	0.52	0.88
Decision Tree (Test Size: 0.2) (entropy)	0.38	0.46	0.79
Decision Tree (Test Size: 0.2) (gini)	0.38	0.49	0.78
Random Forest (Test Size: 0.2) (20)	0.29	0.42	0.92
Random Forest (Test Size: 0.2) (50)	0.23	0.4	0.95
Random Forest (Test Size: 0.2) (100)	0.23	0.4	0.96

- Weight calculation - Binary Weighting Mechanism:-

Classifier	Overall Accuracy	Class 0 Accuracy	Class 1 Accuracy
Gaussian NB (Test Size: 0.2)	0.26	0.54	0.29
Multinomial NB (Test Size: 0.2)	0.65	0.08	0.03
Decision Tree (Test Size: 0.2) (entropy)	0.64	0.43	0.34
Decision Tree (Test Size: 0.2) (gini)	0.63	0.42	0.29
Random Forest (Test Size: 0.2) (20)	0.68	0.37	0.28
Random Forest (Test Size: 0.2) (50)	0.68	0.36	0.28
Random Forest (Test Size: 0.2) (100)	0.68	0.37	0.28

Classifier	Class 2 Accuracy	Class 3 Accuracy	Class 4 Accuracy
Gaussian NB (Test Size: 0.2)	0.33	0.33	0.2
Multinomial NB (Test Size: 0.2)	0.14	0.47	0.91
Decision Tree (Test Size: 0.2) (entropy)	0.36	0.49	0.79
Decision Tree (Test Size: 0.2) (gini)	0.35	0.48	0.79
Random Forest (Test Size: 0.2) (20)	0.27	0.45	0.91
Random Forest (Test Size: 0.2) (50)	0.24	0.41	0.94
Random Forest (Test Size: 0.2) (100)	0.23	0.37	0.96

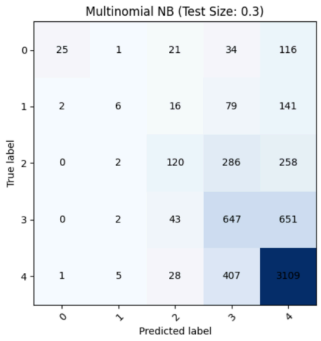
- Weight calculation - TF-IDF Vectorization:-

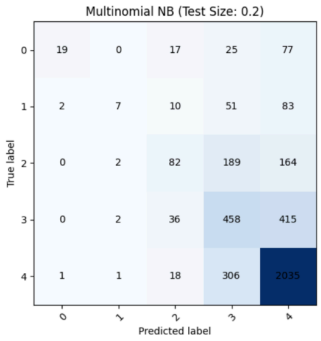
Classifier	Overall Accuracy	Class 0 Accuracy	Class 1 Accuracy
Gaussian NB (Test Size: 0.2)	0.27	0.5	0.3
Multinomial NB (Test Size: 0.2)	0.58	0	0.01
Decision Tree (Test Size: 0.2) (entropy)	0.63	0.41	0.35
Decision Tree (Test Size: 0.2) (gini)	0.64	0.37	0.33
Random Forest (Test Size: 0.2) (20)	0.69	0.39	0.27
Random Forest (Test Size: 0.2) (50)	0.68	0.37	0.27
Random Forest (Test Size: 0.2) (100)	0.69	0.34	0.28

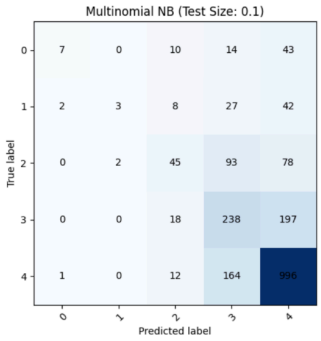
Classifier	Class 2 Accuracy	Class 3 Accuracy	Class 4 Accuracy
Gaussian NB (Test Size: 0.2)	0.32	0.32	0.22
Multinomial NB (Test Size: 0.2)	0	0.06	1
Decision Tree (Test Size: 0.2) (entropy)	0.39	0.48	0.78
Decision Tree (Test Size: 0.2) (gini)	0.37	0.46	0.81
Random Forest (Test Size: 0.2) (20)	0.25	0.44	0.94
Random Forest (Test Size: 0.2) (50)	0.24	0.36	0.97
Random Forest (Test Size: 0.2) (100)	0.23	0.35	0.98

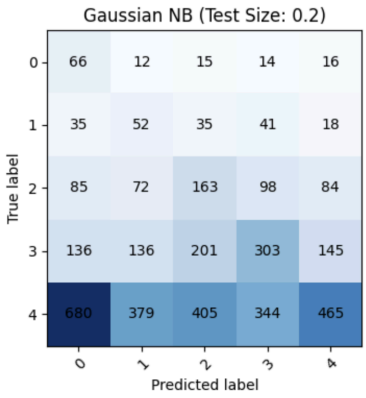
Plotting confusion matrix for different weight calculations:-

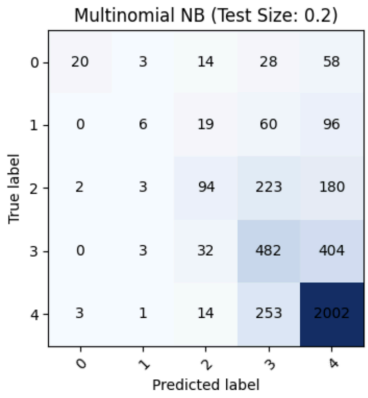
- **<u>Classifier : Naive Bayes:-</u>**
  - Weight calculation - CountVectorization :-



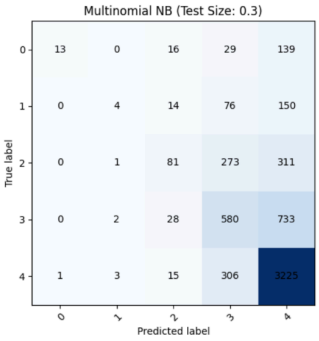


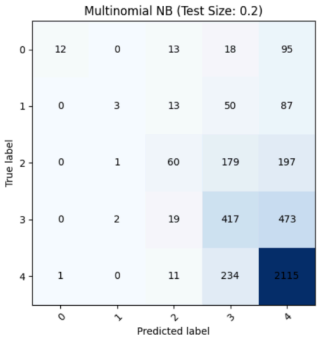


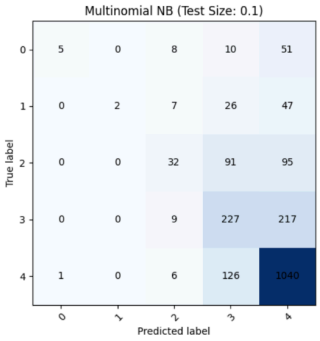


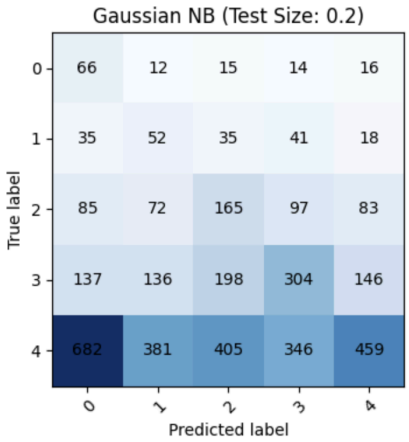


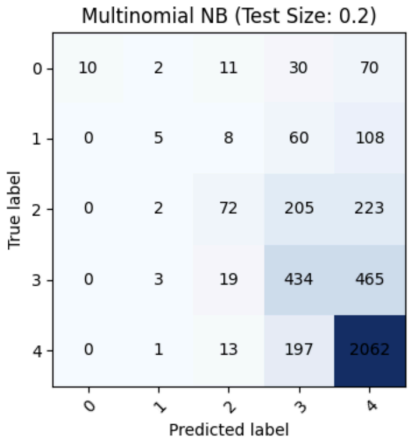
- Weight calculation - Binary Weighting Mechanism:-



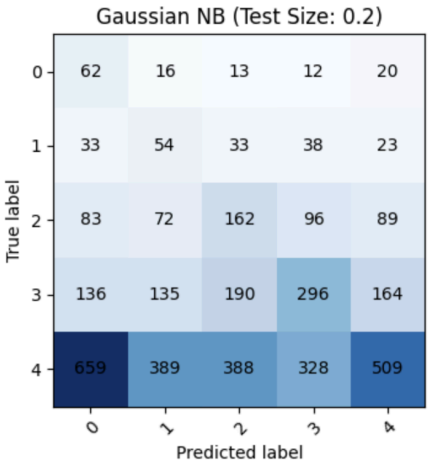


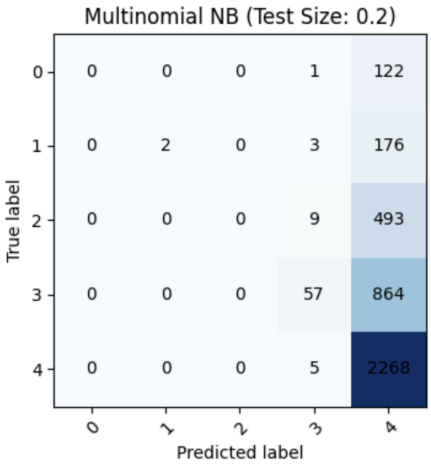




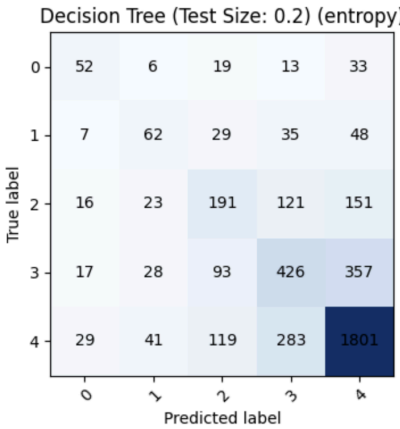


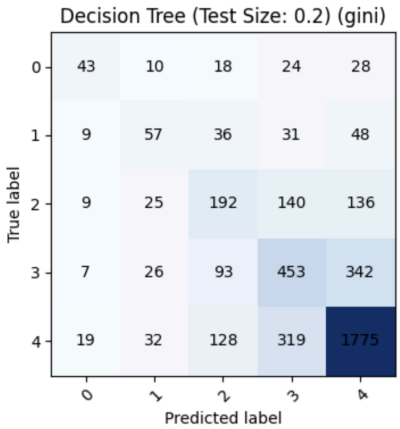
- Weight calculation - TF-IDF Vectorization :-



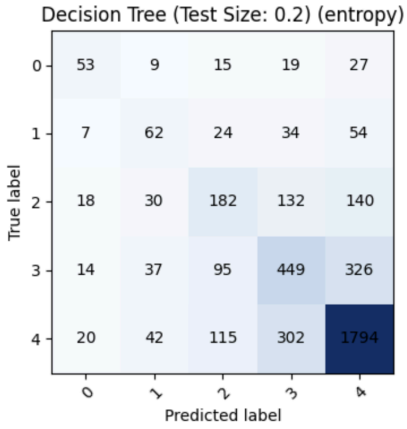


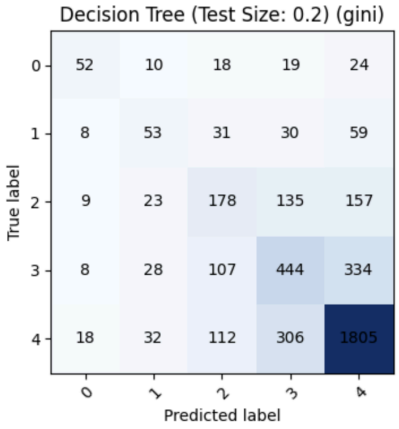
- **<u>Classifier : Decision Tree :-</u>**
  - Weight calculation - CountVectorization :-



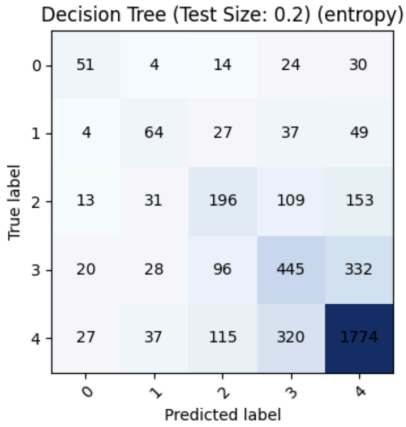


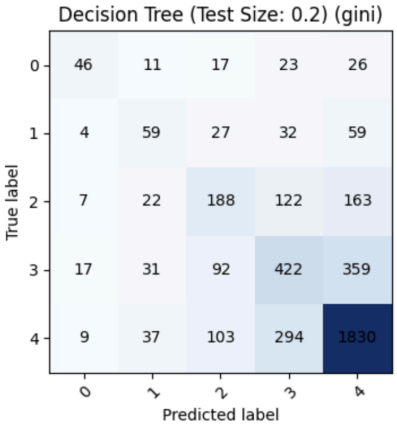
  - Binary Weighting Mechanism:-



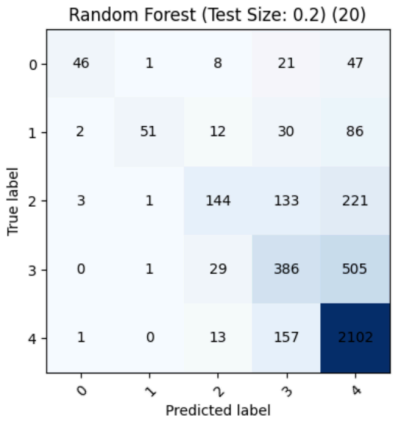


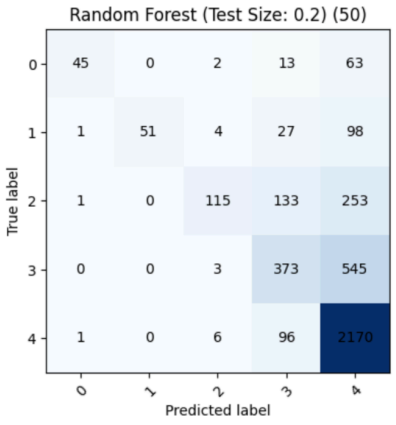
- TF-IDF Vectorization :-

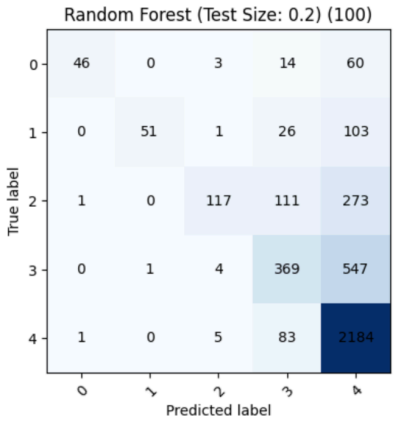




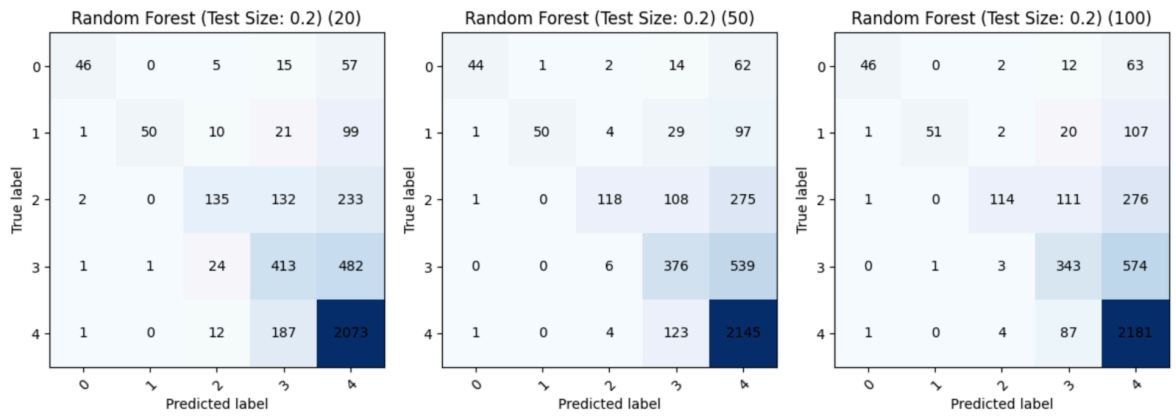
- **<u>Classifier : Random Forest</u>**
  - Weight calculation - CountVectorization :-



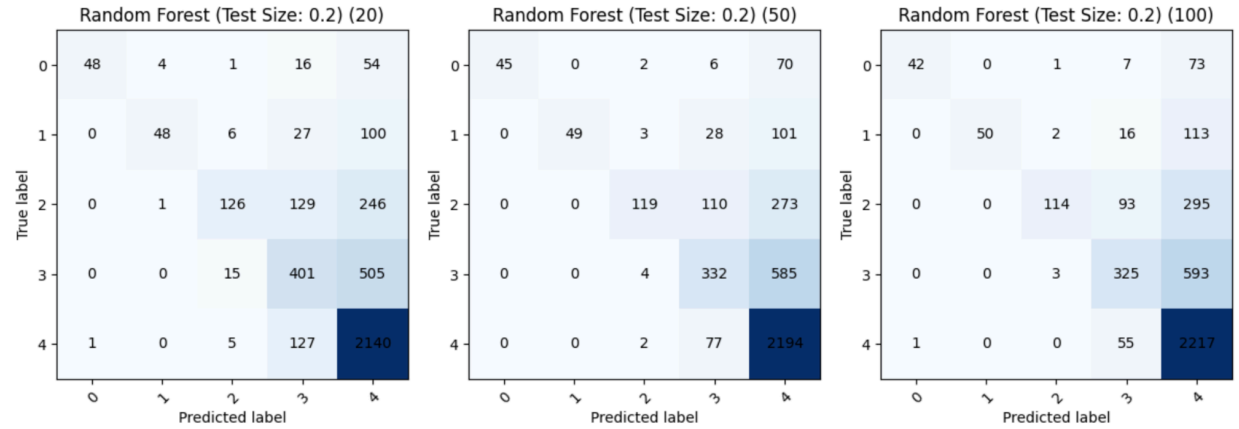




  - Weight calculation - Binary Weighting Mechanism:-



- Weight calculation - TF-IDF Vectorization :-



These experiments are done on three splits of test data, Random Forest results :-

Classifier	Class 0 Precision	Class 1 Precision	Class 2 Precision	Class 3 Precision	Class 4 Precision
Random Forest (Test Size: 0.2) (20)	0.84	0.92	0.71	0.54	0.71
Random Forest (Test Size: 0.2) (50)	0.88	1	0.89	0.61	0.69
Random Forest (Test Size: 0.2) (100)	0.88	1	0.94	0.65	0.69

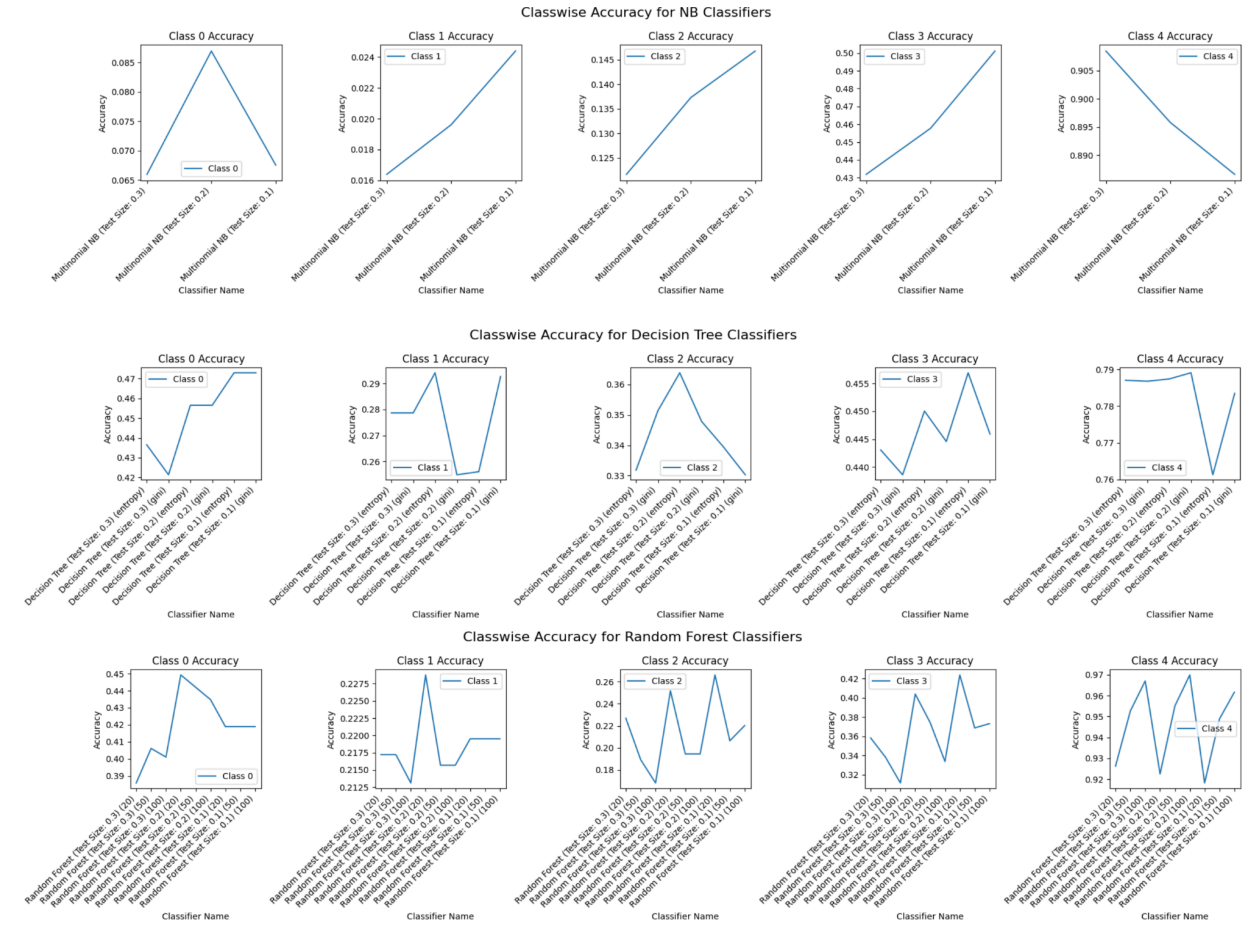
Classifier	Class 0 Recall	Class 1 Recall	Class 2 Recall	Class 3 Recall	Class 4 Recall
Random Forest (Test Size: 0.2) (20)	0.45	0.23	0.25	0.4	0.92
Random Forest (Test Size: 0.2) (50)	0.44	0.22	0.19	0.37	0.96
Random Forest (Test Size: 0.2) (100)	0.43	0.22	0.19	0.33	0.97

Classifier	Class 0 F1-score	Class 1 F1-score	Class 2 F1-score	Class 3 F1-score	Class 4 F1-score
Random Forest (Test Size: 0.2) (20)	0.58	0.37	0.37	0.46	0.8
Random Forest (Test Size: 0.2) (50)	0.59	0.35	0.32	0.47	0.8
Random Forest (Test Size: 0.2) (100)	0.58	0.35	0.32	0.44	0.8

To list all the classifiers :-

Classifier	Class 0 F1-score	Class 1 F1-score	Class 2 F1-score	Class 3 F1-score	Class 4 F1-score
Multinomial NB (Test Size: 0.3)	0.12	0.03	0.2	0.44	0.8
Decision Tree (Test Size: 0.3) (entropy)	0.45	0.27	0.35	0.45	0.78
Decision Tree (Test Size: 0.3) (gini)	0.43	0.29	0.37	0.44	0.78
Random Forest (Test Size: 0.3) (20)	0.54	0.35	0.34	0.42	0.8
Random Forest (Test Size: 0.3) (50)	0.56	0.36	0.31	0.42	0.8
Random Forest (Test Size: 0.3) (100)	0.56	0.35	0.28	0.41	0.8
Multinomial NB (Test Size: 0.2)	0.16	0.04	0.22	0.46	0.79
Decision Tree (Test Size: 0.2) (entropy)	0.5	0.29	0.36	0.46	0.78
Decision Tree (Test Size: 0.2) (gini)	0.47	0.26	0.36	0.45	0.78
Random Forest (Test Size: 0.2) (20)	0.58	0.37	0.37	0.46	0.8
Random Forest (Test Size: 0.2) (50)	0.59	0.35	0.32	0.47	0.8
Random Forest (Test Size: 0.2) (100)	0.58	0.35	0.32	0.44	0.8
Multinomial NB (Test Size: 0.1)	0.13	0.05	0.23	0.49	0.79
Decision Tree (Test Size: 0.1) (entropy)	0.51	0.27	0.34	0.45	0.76
Decision Tree (Test Size: 0.1) (gini)	0.48	0.31	0.34	0.44	0.77
Random Forest (Test Size: 0.1) (20)	0.57	0.36	0.4	0.48	0.8
Random Forest (Test Size: 0.1) (50)	0.56	0.36	0.33	0.46	0.8
Random Forest (Test Size: 0.1) (100)	0.57	0.36	0.35	0.47	0.81

Class wise accuracy on different test splits, Weight calculation Mechanism used : `CountVectorizer` :-



---

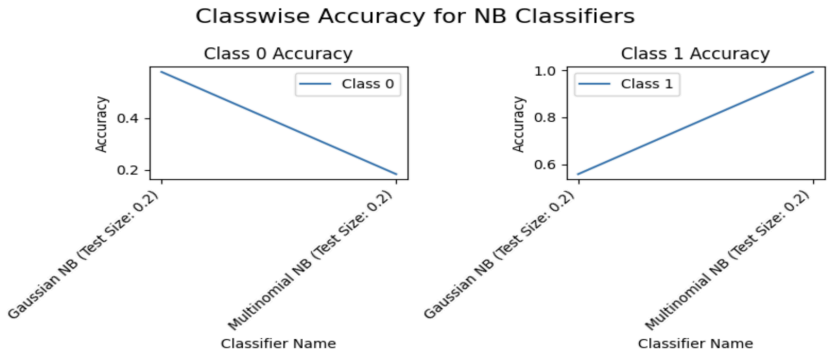
6. **Treat the problem as a binary classification problem using classes 1 and 2 as NEGATIVE and 3, 4, and 5 as POSITIVE.**

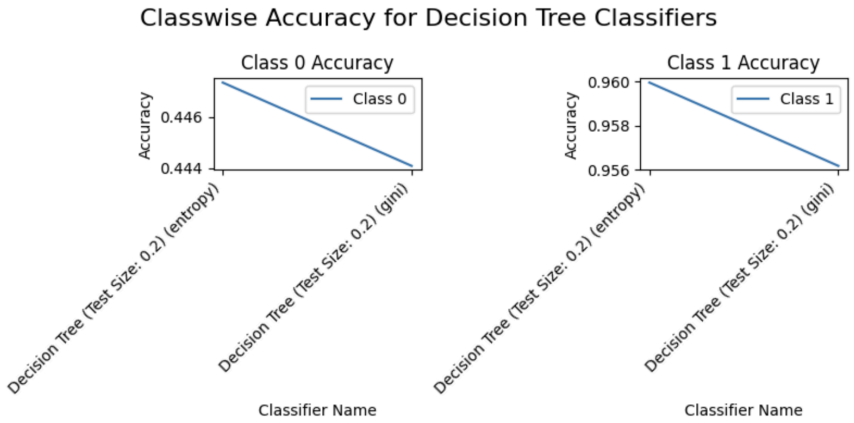
   Changing the labels to -1 and +1 accordingly:-

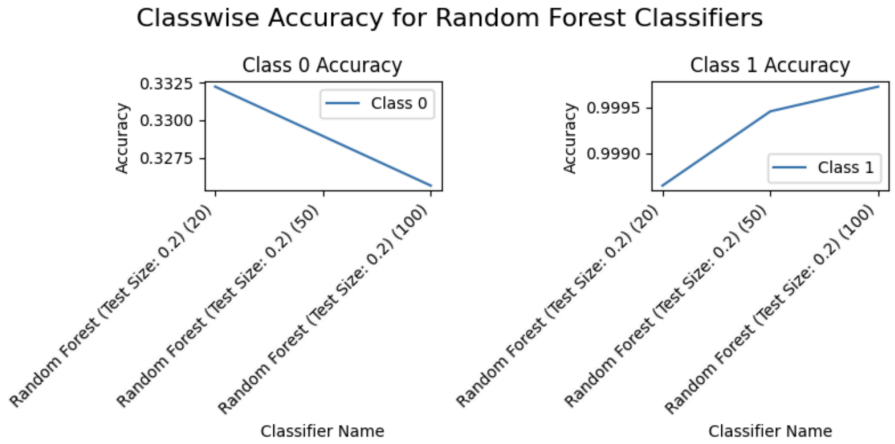
   1. If the rating is 1 or 2, then the reviewText is treated as the negative, labeled -1.
   2. If the rating is 3 or 4 or 5, then the reviewText is treated as the positive, labeled +1.

```
# df = df[df["overall"] != '3'] # need datatype=object
df["label"] = df["overall"].apply(lambda rating : +1 if str(rating) >= '3' else 0)
```

Class-wise Accuracy plots of each classifier for binary classification:-







7. **Compare the performance of the classifiers using accuracy, Precision, Recall, F1 score and confusion matrix.**

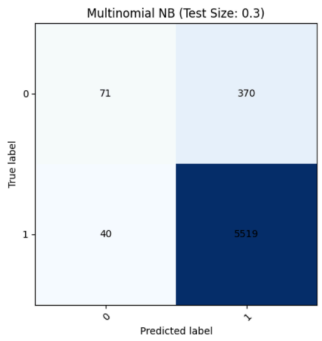
- **Accuracy and Precision for Binary Classification:-**

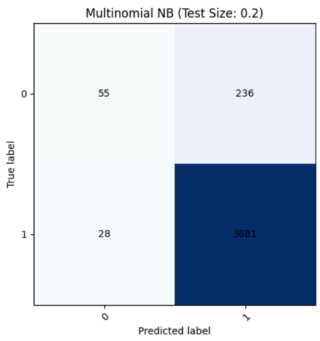
Classifier	Overall Accuracy	Class 0 Accuracy	Class 1 Accuracy	Class 0 Precision	Class 1 Precision
Gaussian NB (Test Size: 0.2)	0.56	0.58	0.56	0.1	0.94
Multinomial NB (Test Size: 0.2)	0.93	0.18	0.99	0.69	0.94
Decision Tree (Test Size: 0.2) (entropy)	0.92	0.45	0.96	0.48	0.95
Decision Tree (Test Size: 0.2) (gini)	0.92	0.44	0.96	0.45	0.95
Random Forest (Test Size: 0.2) (20)	0.95	0.33	1	0.95	0.95
Random Forest (Test Size: 0.2) (50)	0.95	0.33	1	0.98	0.95
Random Forest (Test Size: 0.2) (100)	0.95	0.33	1	0.99	0.95

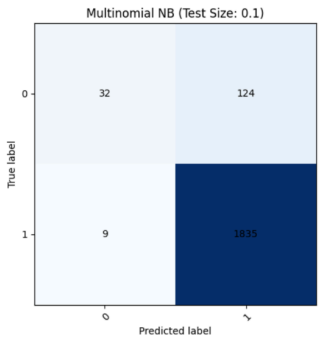
- **Recall and F1-score for Binary Classification:-**

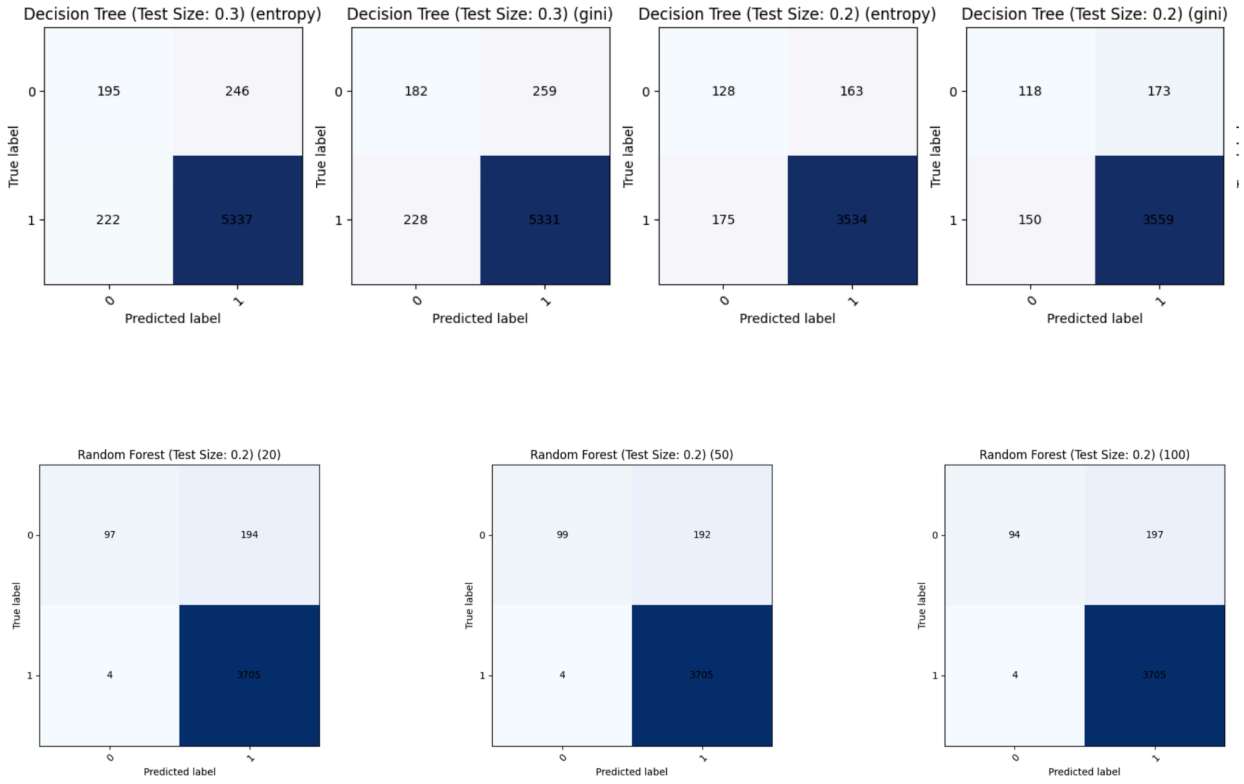
Classifier	Class 0 Recall	Class 1 Recall	Class 0 F1-score	Class 1 F1-score
Gaussian NB (Test Size: 0.2)	0.58	0.56	0.17	0.7
Multinomial NB (Test Size: 0.2)	0.18	0.99	0.29	0.96
Decision Tree (Test Size: 0.2) (entropy)	0.45	0.96	0.46	0.96
Decision Tree (Test Size: 0.2) (gini)	0.44	0.96	0.45	0.96
Random Forest (Test Size: 0.2) (20)	0.33	1	0.49	0.97
Random Forest (Test Size: 0.2) (50)	0.33	1	0.49	0.97
Random Forest (Test Size: 0.2) (100)	0.33	1	0.49	0.97

7.b Confusion Matrices for Binary Classification:-









---

8. **Report hyperparameters (if any)**

Naive Bayes:

- For Multinomial Naive Bayes, there are no explicit hyperparameters mentioned in your code. The algorithm usually doesn't have hyperparameters to tune, but if you're using scikit-learn's MultinomialNB class, it might include alpha for Laplace smoothing.
- For Gaussian Naive Bayes, the default hyperparameters typically include priors for class probabilities, which are usually set based on the training data.

Decision Tree:

- Criterion: "gini" - The Gini impurity measure is used by default in scikit-learn's DecisionTreeClassifier. Determines the function to measure the quality of a split. It can be either "gini" for the Gini impurity or "entropy" for the information gain.
- Max Depth: None - The tree is expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. The maximum depth of the tree.
- Min Samples Split: 2 - The minimum number of samples required to split an internal node. The minimum number of samples required to split an internal node.
- Min Samples Leaf: 1 - The minimum number of samples required to be at a leaf node.

- Max Features: None - The number of features to consider when looking for the best split. If None, then all features are considered.

Random Forest:

- Number of Estimators: 100 - This is the default number of trees in the forest.
- Criterion: "gini" - The default impurity measure used in each tree is the Gini impurity.
- Max Depth: None - The default maximum depth of each tree is unlimited.
- Min Samples Split: 2 - The default minimum number of samples required to split an internal node is 2.
- Min Samples Leaf: 1 - The default minimum number of samples required to be at a leaf node is 1.
- Max Features: "auto" - The default strategy for the number of features to consider when looking for the best split is "auto", which uses the square root of the number of features.

---

9. **Analyze the erroneous results model-wise, and provide your insights on why you think it is happening.**

Naive Bayes (NB):

➔ Performs reasonably well overall, with relatively low error rates across different test sizes.

➔ Shows **higher error rates for label 5 and label 4 compared to other labels**, especially in the larger test sizes.

➔ The error rates increase as the test size decreases, indicating potential overfitting on smaller test sets.

Decision Tree:

➔ Performance varies depending on the criterion used for splitting (entropy vs. Gini).

➔ Generally, higher error rates are observed compared to Multinomial NB.

➔ Error rates also tend to increase as the test size decreases, suggesting potential overfitting.

➔ Entropy seems to perform **better than Gini criterion** in most cases.

Random Forest:

➔ Shows improvement over Decision Tree, especially with a larger number of estimators.

➔ Generally, lower error rates are observed compared to Decision Tree and Multinomial NB.

➔ Error rates also tend to increase as the test size decreases, similar to Decision Tree.

➔ Performance improves with an increase in the number of estimators, indicating the benefits of ensemble learning.

Test Size Impact:

➔ Across all classifiers, error rates tend to increase as the test size decreases, indicating potential overfitting on smaller test sets.

➔ This suggests that the models might not generalize well to unseen data when trained on smaller datasets.

➔ Larger test sizes provide more reliable estimates of model performance and generalization.

➔ Label-specific Errors: Certain labels (e.g., label 5 in Multinomial NB) consistently exhibit higher error rates across different classifiers and test sizes. Label 5 tends to have the highest number of misclassifications across different test sizes, followed by label 4 and then label 3.

Here are the analysis results in case of 5 class classification:

- **Overall Error Rate:**
  - Multinomial NB (Test Size: 0.3): ~26.27%
  - Decision Tree (Entropy, Test Size: 0.3): ~25.29%
  - Random Forest (20 Estimators, Test Size: 0.3): ~24.05%
- **Average Misclassification Rate per Label:**
  - Label 1:
    - Multinomial NB: ~1.78%
    - Decision Tree (Entropy): ~0.93%
    - Random Forest (20 Estimators): ~0.45%
  - Label 2:
    - Multinomial NB: ~0.37%
    - Decision Tree (Entropy): ~0.63%
    - Random Forest (20 Estimators): ~0.13%
  - Label 3:
    - Multinomial NB: ~5.45%
    - Decision Tree (Entropy): ~5.65%
    - Random Forest (20 Estimators): ~1.78%
  - Label 4:
    - Multinomial NB: ~4.53%
    - Decision Tree (Entropy): ~4.41%
    - Random Forest (20 Estimators): ~3.20%
  - Label 5:
    - Multinomial NB: ~14.13%

        - Decision Tree (Entropy): ~13.67%
        - Random Forest (20 Estimators): ~18.49%
- **Standard Deviation of Error Rates:**
    - Multinomial NB: ~10.19%
    - Decision Tree (Entropy): ~3.45%
    - Random Forest (20 Estimators): ~6.39%

Here are the analysis results in case of Binary classification:

- **Overall Error Rate:**
    - Multinomial NB (Test Size: 0.3):
    - Error rate: ~8.89%
    - Decision Tree (Entropy, Test Size: 0.3):
    - Error rate: ~9.33%
    - Random Forest (20 Estimators, Test Size: 0.3):
    - Error rate: ~2.22%
- **Average Misclassification Rate per Label:**
    - Label 0:
        - Multinomial NB: ~2.51%
        - Decision Tree (Entropy): ~5.07%
        - Random Forest (20 Estimators): ~0.57%
    - Label 1:
        - Multinomial NB: ~8.89%
        - Decision Tree (Entropy): ~3.12%
        - Random Forest (20 Estimators): ~3.48%
- **Standard Deviation of Error Rates:**
    - Multinomial NB: ~4.36%
    - Decision Tree (Entropy): ~3.06%
    - Random Forest (20 Estimators): ~3.24%

Few comparisons between the multi-class classification and binary classification:-

Metric	Multinomial NB	Decision Tree (Entropy)	Random Forest
Overall Error Rate	~8.89%	~9.33%	~7.78%
Average Misclassification Rate (Label 0)	~2.51%	~5.07%	~0.75%

Metric	Multinomial NB	Decision Tree (Entropy)	Random Forest
Average Misclassification Rate (Label 1)	~8.89%	~3.12%	~8.80%
Standard Deviation of Error Rates	~4.36%	~3.06%	~2.92%

9.b Why could this be happening? :-

1. Imbalanced Classes: In the Multinomial Naive Bayes model with a test size of 0.3, we see that the label 1 has a total of 370 errors, all of which are misclassified as label 0. Similarly, in the Decision Tree models with both entropy and gini criteria, and the Random Forest models with various numbers of estimators, we observe misclassifications predominantly from label 1 to label 0. This suggests a potential imbalance in the classes, where label 0 may be the majority class and label 1 the minority. As a result, the models might struggle to properly learn the patterns of the minority class, leading to higher misclassification rates for label 1.
2. Model Assumptions: The Naive Bayes classifier assumes that features are conditionally independent given the class label. If this assumption is violated in the data, such as when features are correlated, it can lead to suboptimal performance and higher misclassification rates.
3. Feature Relevance: The features used for classification may not fully capture the nuances of the underlying data, leading to misclassifications. It's possible that some features are more relevant than others, and the models may not adequately weight them during the classification process.
4. Model Complexity and Overfitting: The Decision Tree and Random Forest models, especially those with higher numbers of estimators (e.g., 50 and 100), exhibit relatively low misclassification rates overall. However, they still misclassify some instances, particularly for label 1. This could be attributed to overfitting, where the models capture noise or outliers in the training data, leading to suboptimal generalization on the test data.
5. Data Quality: Inaccuracies or inconsistencies in the data, such as missing values, outliers, or mislabeled instances, can adversely impact the performance of machine learning models and lead to higher misclassification rates.

Task2 Colab File Link

References:-

https://medium.com/@maleeshadesilva21/preprocessing-steps-for-natural-language-processing-nlp-a-beginners-guide-d6d9bf7689c9

https://www.analyticsvidhya.com/blog/2022/09/implementing-count-vectorizer-and-tf-idf-in-nlp-using-pyspark/