REPORT

Aim: Text Classification Model using various algorithms.

Objective: This report aims to assess and compare the performance of four distinct machine learning algorithms – Logistic Regression, Random Forest, Support Vector Classifier (SVC), and Naive Bayes – in the context of text classification using the 20 Newsgroups dataset.

The primary objectives include:

Algorithm Comparison: Evaluate the effectiveness of each algorithm in accurately categorizing text inputs into predefined news groups within the 20 Newsgroups dataset.

Performance Metrics: Measure and compare the accuracy, precision, and recall of the algorithms to ascertain their suitability for text classification tasks.

Real-World Applicability: Explore the practical implications of the text classification model by demonstrating its ability to assign relevant categories to user-provided textual inputs.

By achieving these objectives, we, through this report, intend to provide insights into the comparative performance of machine learning algorithms for text classification, aiding in the selection of an optimal algorithm for real-world applications in content categorization and information retrieval systems.

Code:

- This code analyzes the 20 Newsgroups dataset for text classification purposes.
- It divides the data into training and test sets and experiments with diverse classifiers like Multinomial Naive Bayes, Random Forest, SVC, and Logistic Regression.
- Each classifier is coupled with a TF-IDF vectorizer to train a model. Performance evaluation includes accuracy metrics, classification reports, and visualized confusion matrices.
- The code then identifies the most accurate model and re-trains it.
- Lastly, it integrates this superior model into a user interface using Gradio, allowing users to input text for category predictions.

The primary aim is to develop an accurate text classification tool for categorizing text inputs. It uses diverse machine learning algorithms, aiming to accurately categorize text inputs from the 20 Newsgroups dataset into defined categories.

The model:

Importing important required files

```
import warnings warnings.filterwarnings('ignore')
warnings.simplefilter('ignore') from sklearn.metrics import
confusion_matrix from transformers import pipeline
import numpy as np import pandas as pd import
matplotlib.pyplot as plt import seaborn as sns from
sklearn.datasets import fetch_20newsgroups from
sklearn.model_selection import train_test_split from
sklearn.feature_extraction.text import TfidfVectorizer from
sklearn.metrics import accuracy_score,
classification_report from sklearn.pipeline import
```

```
make_pipeline from sklearn.naive_bayes import
MultinomialNB from sklearn.ensemble import
RandomForestClassifier from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression from
sklearn.preprocessing import LabelEncoder import gradio
as gr
```

Displaying the dataset and splitting it into training and testing sets of data

```
data = fetch_20newsgroups(subset='all',remove=('headers', 'footers', 'quotes'))
print("First few rows of the dataset:") print(data.data[:2]) print("Number of
samples:", len(data.data)) print("\nTarget names:", data.target_names)

X_train, X_test, y_train, y_test = train_test_split(data.data, data.target,
test_size=0.1, random_state=1) categories = ['alt.atheism',
'comp.graphics',
'comp.os.ms-windows.misc','comp.sys.ibm.pc.hardware','comp.sys.mac.hardware',
'comp.windows.x','misc.forsale', 'rec.autos', 'rec.motorcycles','rec.sport.baseball',
'rec.sport.hockey', 'sci.crypt' ,'sci.electronics', 'sci.med', 'sci.space',
'soc.religion.christian', 'talk.politics.guns','talk.politics.mideast',
'talk.politics.misc','talk.religion.misc']
# Training the data on these categories
train = fetch_20newsgroups (subset='train', categories=categories)
```

Multinomial function without API

```
class MultinomialNaiveBayes:
    def _init_(self, alpha=0.01):
        self.alpha = alpha
        self.class_probs = None
        self.feature_probs = None
    def fit(self, X, y):
        num_classes = len(np.unique(y))
        num_features = X.shape[1]

        # Calculate class probabilities
        self.class_probs = np.zeros(num_classes)
        for i in range(num_classes):
            self.class_probs[i] = np.sum(y == i) / len(y)

        # Calculate feature probabilities self.feature_probs =
        np.zeros((num_classes, num_features)) for i in range(num_classes):
        class_count = np.sum(y == i) self.feature_probs[i, :] = (np.sum(X[y
        == i], axis=0) + self.alpha) /
(class_count + self.alpha * num_features)

    def predict(self, X):
        num_samples = X.shape[0] num_classes =
        len(self.class_probs) predictions =
        np.zeros(num_samples, dtype=int)

        for i in range(num_samples):
            # Ensure X[i] is a 2D array with a single row sample_probs =
            np.sum(np.log(self.feature_probs) * X[i, :].toarray(),
axis=1) + np.log(self.class_probs) predictions[i]

        = np.argmax(sample_probs) return

        predictions
```

Define a list of classifiers to try

```
classifiers = [
    MultinomialNB(),
    RandomForestClassifier(),
    SVC(),
    LogisticRegression()
]

ma=0
bar_values=[]
bar_class=["MultinomialNB","RandomForestClassifier","SVC","LogisticRegressi
on",] classifi=None for classifier in classifiers:
    Create a pipeline with TF-IDF vectorizer and the current classifier
    model = make_pipeline(TfidfVectorizer(), classifier)

    Training the model
    model.fit(X_train, y_train)

    Make predictions on the test set
    predictions = model.predict(X_test)



    Evaluate the performance of the model


    accuracy = accuracy_score(y_test, predictions)
    print(f"\nClassifier:
    {classifier._class.name_}")
    maxx=round(accuracy, 2)
    bar_values.append(maxx) print(f"Accuracy:
    {accuracy:.2f}")

    Display classification report
    print("Classification Report:\n", classification_report(y_test, predictions))
    conf_matrix = confusion_matrix(y_test, predictions)

    Plot confusion matrix as a heatmap

   plt.figure(figsize=(8, 6))
    sns.heatmap(conf_matrix, annot=True, fmt='d', cbar=False,
xticklabels=data.target_names, yticklabels=data.target_names)
    plt.xlabel('Predicted') plt.ylabel('Actual')
    plt.title(f'Confusion Matrix - {classifier._class.name_}')
    plt.show()
    #getting best model
    train if(maxx>ma):
    ma=maxx
        classifi=classifier
    print("\n\n\n")
plt.xlabel('Model', fontweight ='bold', fontsize = 15)
plt.ylabel('Accuracy', fontweight ='bold', fontsize = 15)
plt.bar(bar_class,bar_values, width = 0.4)
```

Annotating each bar with its value for i, value in enumerate(bar_values):
plt.text(i, value, f'{value:.2f}', ha='center', va='bottom', fontweight='bold')

best algo model is trained again print(f"Best
accuracy model is {classifi}") model =
make_pipeline(TfidfVectorizer(), classifi)

Train the model
model.fit(X_train, y_train)

Make predictions on the test set predictions
= model.predict(X_test) *Evaluate the
performance of the model* accuracy =
accuracy_score(y_test, predictions)
print(f"\nClassifier: {classifi}")
maxx=round(accuracy, 2)
print(f"Accuracy: {accuracy:.2f}")

Display classification report

```
print("Classification Report:\n", classification_report(y_test, predictions))
conf_matrix = confusion_matrix(y_test, predictions)

def predict_category(Enter_article, train=train, model=model):
    pred=model.predict([Enter_article])
    return train.target_names[pred[0]]
iface=gr.Interface(fn=predict_category,inputs=gr.Textbox(lines=10,
placeholder="Enter text here"),outputs="text", title="Text
Classification",description="getting... the categories of Artical/news")
iface.launch(inline=False,share=True)
```

Algorithm:

Data Collection and Preprocessing:

The 20 Newsgroups dataset was retrieved to develop a robust text classification system. Preprocessing steps, including the removal of headers, footers, and quotes, were executed to refine the dataset before model training.

Model Training and Selection:

Various machine learning algorithms were implemented for text classification, encompassing Logistic Regression, Random Forest, Support Vector Classification (SVC), and Naive Bayes. These models were trained using the dataset, each utilizing a distinct approach to learn from the text data's patterns.

Evaluation and Validation:

The trained models underwent thorough evaluation using multiple metrics such as accuracy, classification reports, and confusion matrices. These metrics facilitated a comprehensive assessment of each model's performance in categorizing text inputs into predefined categories.

Model Deployment:

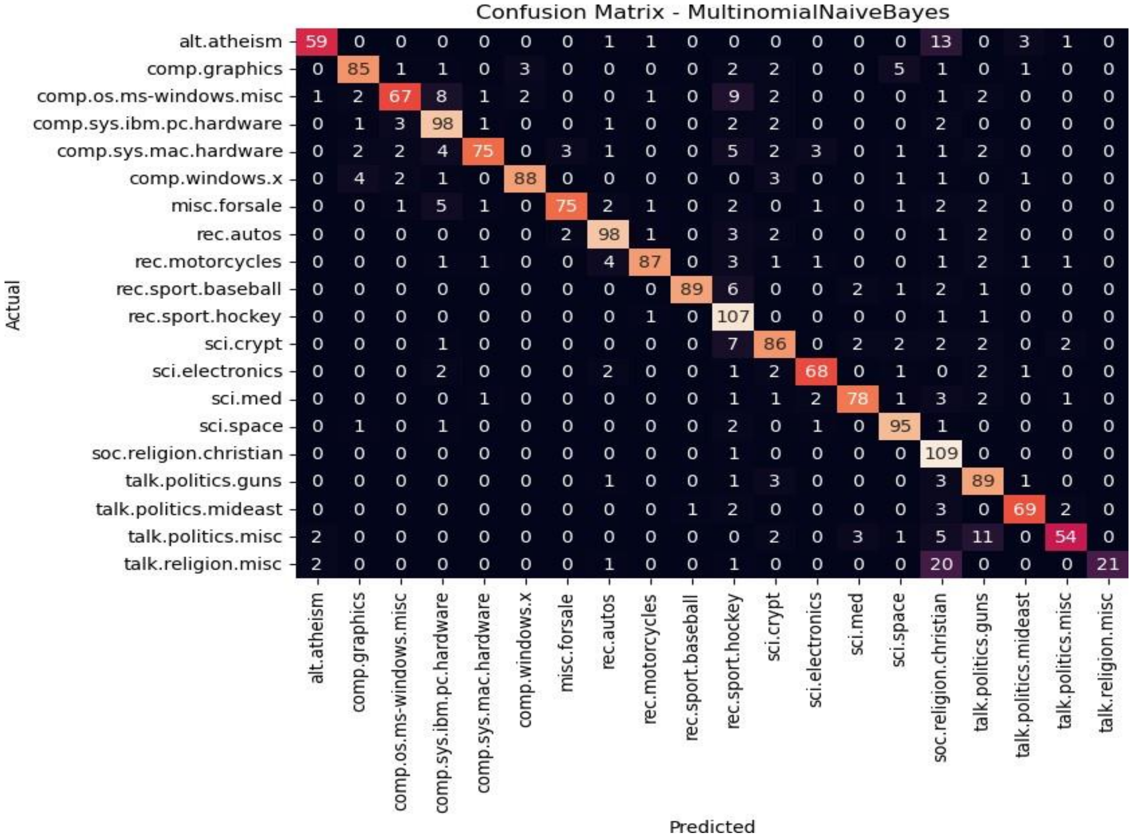
Methods for model deployment were explored to apply the best-performing classification model effectively for categorizing text inputs.

Output:

```
Classifier: MultinomialNaiveBayes
Accuracy: 0.85
 Classification         Report            recall     f1-score         support
                        Precision:
                 0            0.92          0.76          0.83             78

                 1            0.89          0.84          0.87            101
                 2            0.88          0.70          0.78             96
                 3            0.80          0.89          0.84            110
                 4            0.94          0.74          0.83            101
                 5            0.95          0.87          0.91            101
                 6            0.94          0.81          0.87             93
                 7            0.88          0.90          0.89            109
                 8            0.95          0.84          0.89            103
                 9            0.99          0.88          0.93            101
                10            0.69          0.97          0.81            110
                11            0.80          0.83          0.81            104
                12            0.89          0.86          0.88             79
                13            0.92          0.87          0.89             90
                14            0.87          0.94          0.90            101
                15            0.63          0.99          0.77            110
                16            0.75          0.91          0.82             98
                17            0.90          0.90          0.90             77
                18            0.89          0.69          0.78             78
                19            1.00          0.47          0.64             45
          accuracy                                      0.85           1885


          macro avg           0.87          0.83          0.84           1885
 weighted avg                 0.87          0.85          0.85           1885
```



Confusion Matrix - MultinomialNaiveBayes

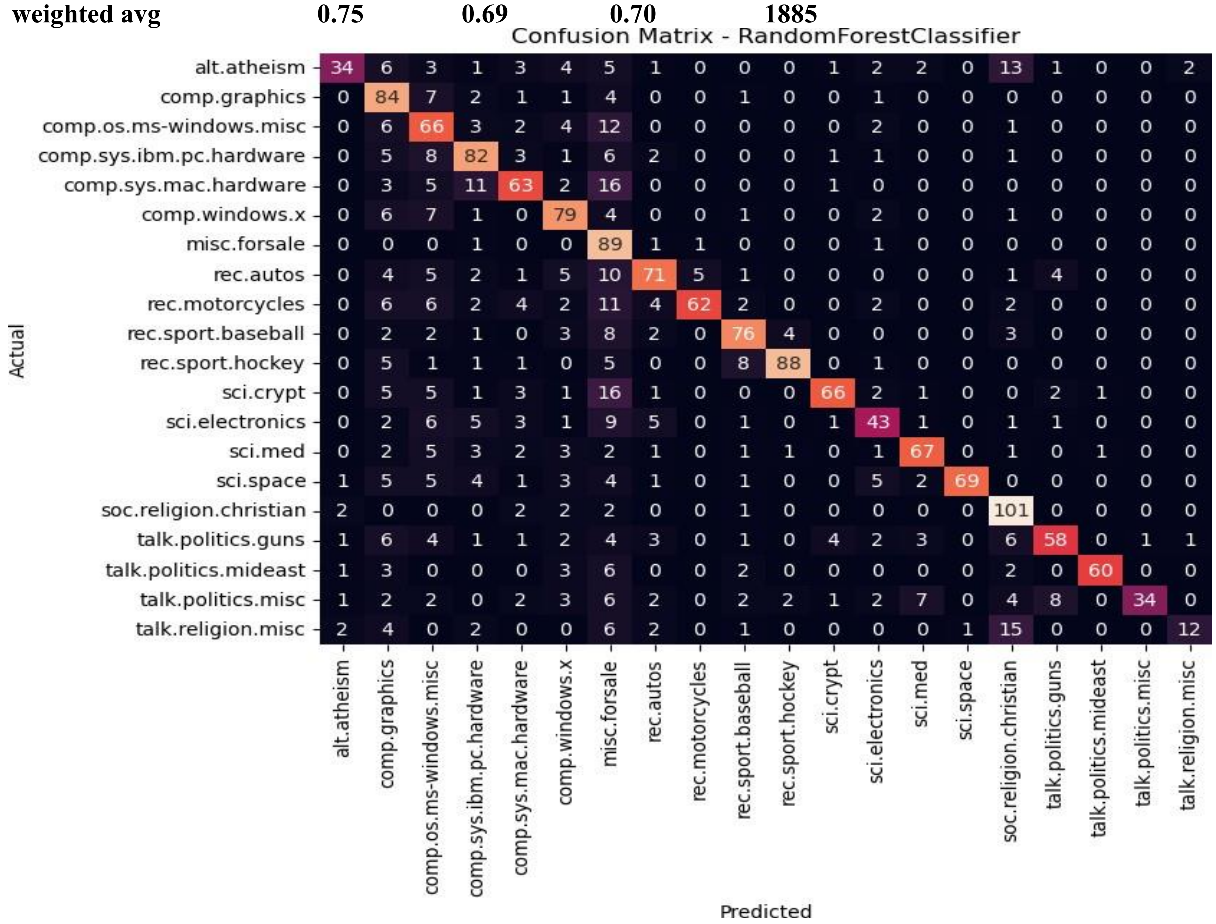
	alt.atheism	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x	misc.forsale	rec.autos	rec.motorcycles	rec.sport.baseball	rec.sport.hockey	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion.christian	talk.politics.guns	talk.politics.mideast	talk.politics.misc	talk.religion.misc
alt.atheism	59	0	0	0	0	0	0	1	1	0	0	0	0	0	0	13	0	3	1	0
comp.graphics	0	85	1	1	0	3	0	0	0	0	2	2	0	0	5	1	0	1	0	0
comp.os.ms-windows.misc	1	2	67	8	1	2	0	0	1	0	9	2	0	0	0	1	2	0	0	0
comp.sys.ibm.pc.hardware	0	1	3	98	1	0	0	1	0	0	2	2	0	0	0	2	0	0	0	0
comp.sys.mac.hardware	0	2	2	4	75	0	3	1	0	0	5	2	3	0	1	1	2	0	0	0
comp.windows.x	0	4	2	1	0	88	0	0	0	0	0	3	0	0	1	1	0	1	0	0
misc.forsale	0	0	1	5	1	0	75	2	1	0	2	0	1	0	1	2	2	0	0	0
rec.autos	0	0	0	0	0	0	2	98	1	0	3	2	0	0	0	1	2	0	0	0
rec.motorcycles	0	0	0	1	1	0	0	4	87	0	3	1	1	0	0	1	2	1	1	0
rec.sport.baseball	0	0	0	0	0	0	0	0	0	89	6	0	0	2	1	2	1	0	0	0
rec.sport.hockey	0	0	0	0	0	0	0	0	1	0	107	0	0	0	0	1	1	0	0	0
sci.crypt	0	0	0	1	0	0	0	0	0	0	7	86	0	2	2	2	2	0	2	0
sci.electronics	0	0	0	2	0	0	0	2	0	0	1	2	68	0	1	0	2	1	0	0
sci.med	0	0	0	0	1	0	0	0	0	0	1	1	2	78	1	3	2	0	1	0
sci.space	0	1	0	1	0	0	0	0	0	0	2	0	1	0	95	1	0	0	0	0
soc.religion.christian	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	109	0	0	0	0
talk.politics.guns	0	0	0	0	0	0	0	1	0	0	1	3	0	0	0	3	89	1	0	0
talk.politics.mideast	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	3	0	69	2	0
talk.politics.misc	2	0	0	0	0	0	0	0	0	0	0	2	0	3	1	5	11	0	54	0
talk.religion.misc	2	0	0	0	0	0	0	1	0	0	1	0	0	0	0	20	0	0	0	21

Classifier: RandomForestClassifier

Accuracy: 0.69

Classification	**Report Precision:**	**recall**	**f1-score**	**support**
0	**0.81**	**0.44**	**0.57**	**78**
1	**0.54**	**0.83**	**0.65**	**101**
2	**0.48**	**0.69**	**0.57**	**96**
3	**0.67**	**0.75**	**0.70**	**110**
4	**0.68**	**0.62**	**0.65**	**101**
5	**0.66**	**0.78**	**0.72**	**101**
6	**0.40**	**0.96**	**0.56**	**93**
7	**0.74**	**0.65**	**0.69**	**109**
8	**0.91**	**0.60**	**0.73**	**103**
9	**0.77**	**0.75**	**0.76**	**101**
10	**0.93**	**0.80**	**0.86**	**110**
11	**0.88**	**0.63**	**0.74**	**104**
12	**0.64**	**0.54**	**0.59**	**79**
13	**0.81**	**0.74**	**0.77**	**90**
14	**0.99**	**0.68**	**0.81**	**101**
15	**0.66**	**0.92**	**0.77**	**110**
16	**0.78**	**0.59**	**0.67**	**98**
17	**0.97**	**0.78**	**0.86**	**77**

18	0.97	0.44	0.60	78
19	0.80	0.27	0.40	45
accuracy			0.69	1885
macro avg	0.75	0.67	0.68	1885
weighted avg	0.75	0.69	0.70	1885



Confusion Matrix - RandomForestClassifier

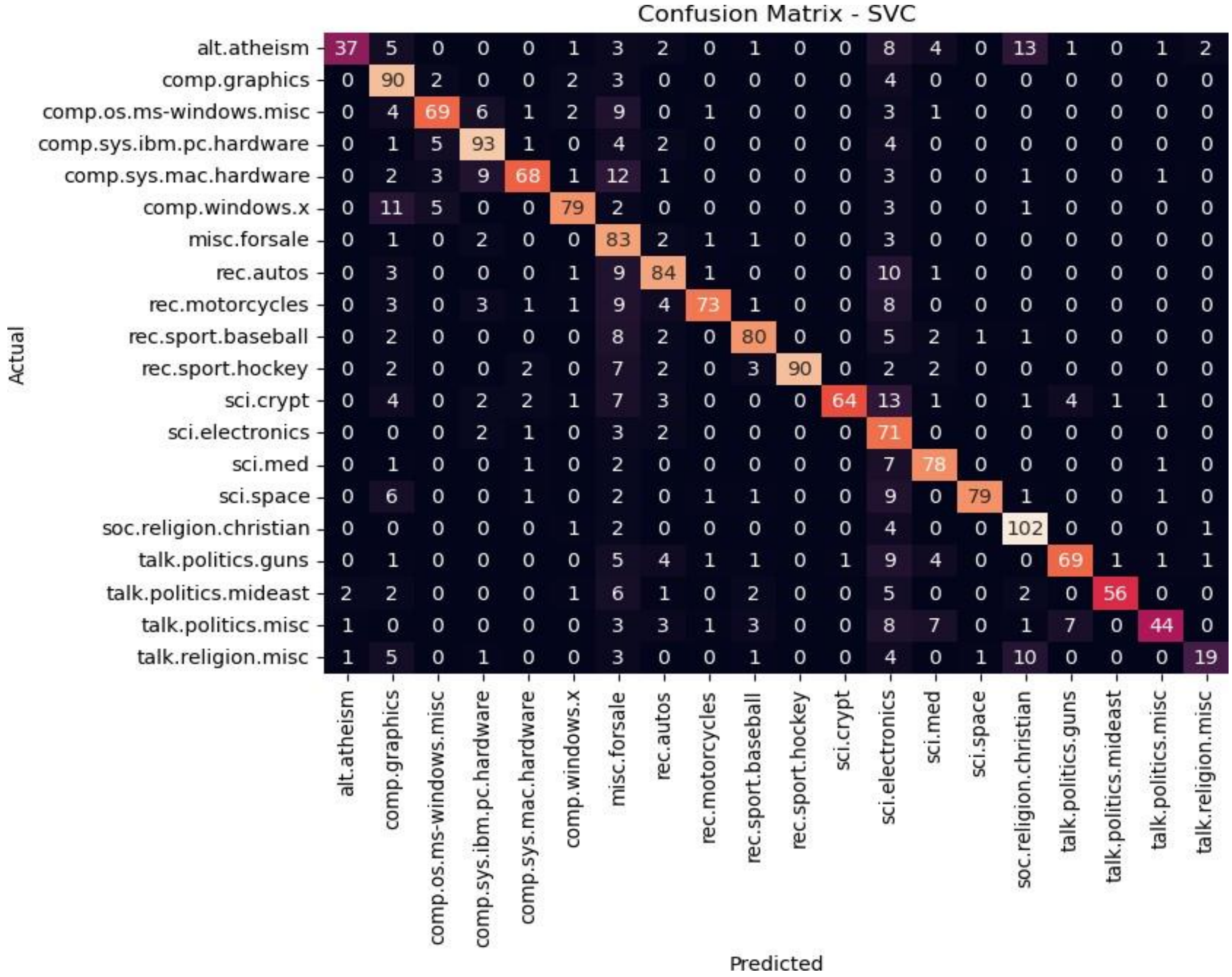
Actual \ Predicted	alt.atheism	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x	misc.forsale	rec.autos	rec.motorcycles	rec.sport.baseball	rec.sport.hockey	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion.christian	talk.politics.guns	talk.politics.mideast	talk.politics.misc	talk.religion.misc
alt.atheism	34	6	3	1	3	4	5	1	0	0	0	1	2	2	0	13	1	0	0	2
comp.graphics	0	84	7	2	1	1	4	0	0	1	0	0	1	0	0	0	0	0	0	0
comp.os.ms-windows.misc	0	6	66	3	2	4	12	0	0	0	0	0	2	0	0	1	0	0	0	0
comp.sys.ibm.pc.hardware	0	5	8	82	3	1	6	2	0	0	0	1	1	0	0	1	0	0	0	0
comp.sys.mac.hardware	0	3	5	11	63	2	16	0	0	0	0	1	0	0	0	0	0	0	0	0
comp.windows.x	0	6	7	1	0	79	4	0	0	1	0	0	2	0	0	1	0	0	0	0
misc.forsale	0	0	0	1	0	0	89	1	1	0	0	0	1	0	0	0	0	0	0	0
rec.autos	0	4	5	2	1	5	10	71	5	1	0	0	0	0	0	1	4	0	0	0
rec.motorcycles	0	6	6	2	4	2	11	4	62	2	0	0	2	0	0	2	0	0	0	0
rec.sport.baseball	0	2	2	1	0	3	8	2	0	76	4	0	0	0	0	3	0	0	0	0
rec.sport.hockey	0	5	1	1	1	0	5	0	0	8	88	0	1	0	0	0	0	0	0	0
sci.crypt	0	5	5	1	3	1	16	1	0	0	0	66	2	1	0	0	2	1	0	0
sci.electronics	0	2	6	5	3	1	9	5	0	1	0	1	43	1	0	1	1	0	0	0
sci.med	0	2	5	3	2	3	2	1	0	1	1	0	1	67	0	1	0	1	0	0
sci.space	1	5	5	4	1	3	4	1	0	1	0	0	5	2	69	0	0	0	0	0
soc.religion.christian	2	0	0	0	2	2	2	0	0	1	0	0	0	0	0	101	0	0	0	0
talk.politics.guns	1	6	4	1	1	2	4	3	0	1	0	4	2	3	0	6	58	0	1	1
talk.politics.mideast	1	3	0	0	0	3	6	0	0	2	0	0	0	0	0	2	0	60	0	0
talk.politics.misc	1	2	2	0	2	3	6	2	0	2	2	1	2	7	0	4	8	0	34	0
talk.religion.misc	2	4	0	2	0	0	6	2	0	1	0	0	0	0	1	15	0	0	0	12

Classifier: SVC Accuracy: 0.76

Classifier: SVC Accuracy: 0.76 Classification	**Report: precision**	**recall**	**f1-score**	**support**
0	**0.90**	**0.47**	**0.62**	**78**
1	**0.63**	**0.89**	**0.74**	**101**
2	**0.82**	**0.72**	**0.77**	**96**
3	**0.79**	**0.85**	**0.82**	**110**
4	**0.87**	**0.67**	**0.76**	**101**
5	**0.88**	**0.78**	**0.83**	**101**
6	**0.46**	**0.89**	**0.60**	**93**

7 0.75 0.77 0.76 109 8 0.92 0.71
0.80 103
9 0.85 0.79 0.82 101
10 1.00 0.82 0.90 110 11 0.98 0.62
0.76 104
12 0.39 0.90 0.54 79
13 0.78 0.87 0.82 90 14 0.98 0.78 0.87 101
15 0.77 0.93 0.84 110
16 0.85 0.70 0.77 98
17 0.97 0.73 0.83 77
18 0.88 0.56 0.69 78
19 0.83 0.42 0.56 45 accuracy 0.76 1885

macro avg 0.81 0.74 0.75 1885 weighted avg 0.82
0.76 0.77 1885



Confusion Matrix - SVC

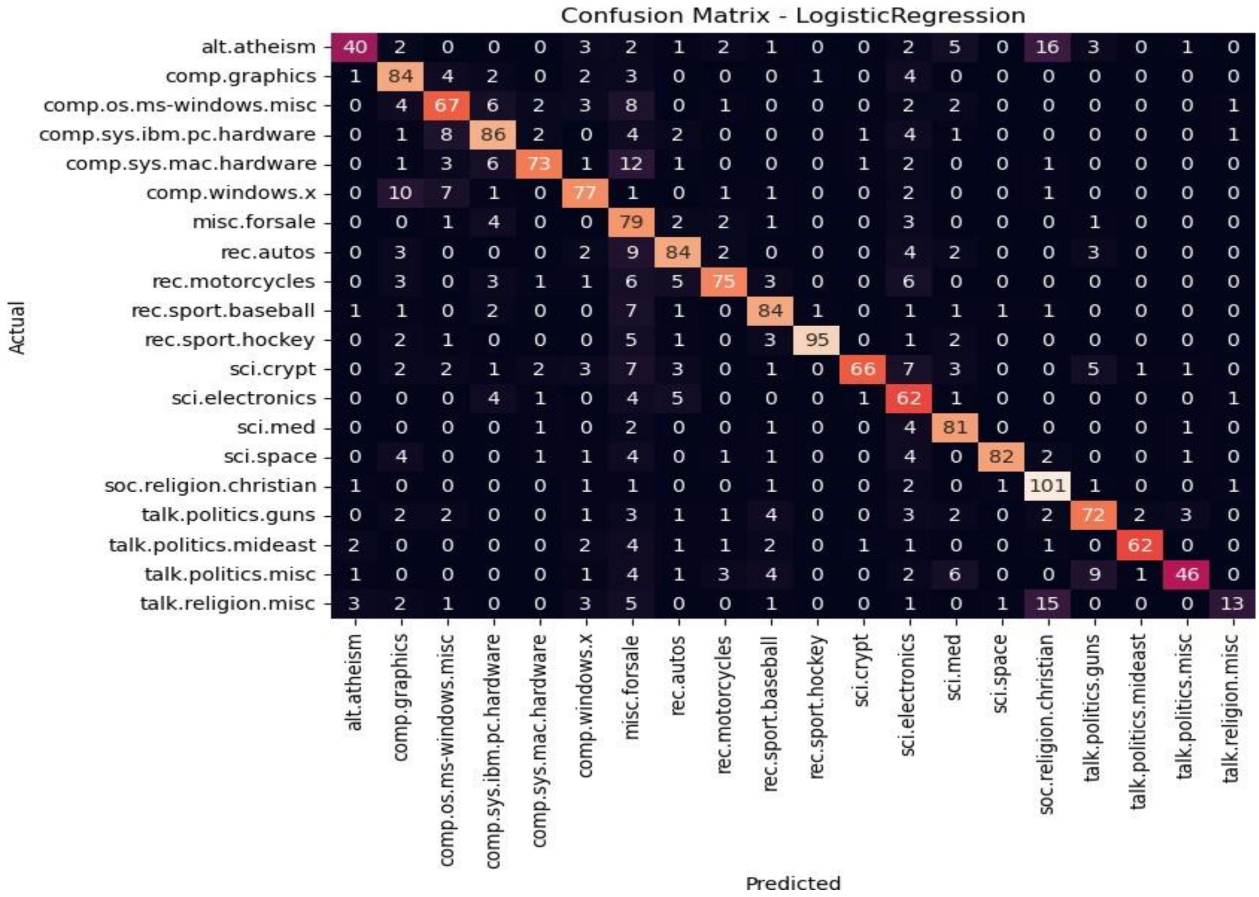
	alt.atheism	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x	misc.forsale	rec.autos	rec.motorcycles	rec.sport.baseball	rec.sport.hockey	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion.christian	talk.politics.guns	talk.politics.mideast	talk.politics.misc	talk.religion.misc
alt.atheism	37	5	0	0	0	1	3	2	0	1	0	0	8	4	0	13	1	0	1	2
comp.graphics	0	90	2	0	0	2	3	0	0	0	0	0	4	0	0	0	0	0	0	0
comp.os.ms-windows.misc	0	4	69	6	1	2	9	0	1	0	0	0	3	1	0	0	0	0	0	0
comp.sys.ibm.pc.hardware	0	1	5	93	1	0	4	2	0	0	0	0	4	0	0	0	0	0	0	0
comp.sys.mac.hardware	0	2	3	9	68	1	12	1	0	0	0	0	3	0	0	1	0	0	1	0
comp.windows.x	0	11	5	0	0	79	2	0	0	0	0	0	3	0	0	1	0	0	0	0
misc.forsale	0	1	0	2	0	0	83	2	1	1	0	0	3	0	0	0	0	0	0	0
rec.autos	0	3	0	0	0	1	9	84	1	0	0	0	10	1	0	0	0	0	0	0
rec.motorcycles	0	3	0	3	1	1	9	4	73	1	0	0	8	0	0	0	0	0	0	0
rec.sport.baseball	0	2	0	0	0	0	8	2	0	80	0	0	5	2	1	1	0	0	0	0
rec.sport.hockey	0	2	0	0	2	0	7	2	0	3	90	0	2	2	0	0	0	0	0	0
sci.crypt	0	4	0	2	2	1	7	3	0	0	0	64	13	1	0	1	4	1	1	0
sci.electronics	0	0	0	2	1	0	3	2	0	0	0	0	71	0	0	0	0	0	0	0
sci.med	0	1	0	0	1	0	2	0	0	0	0	0	7	78	0	0	0	0	1	0
sci.space	0	6	0	0	1	0	2	0	1	1	0	0	9	0	79	1	0	0	1	0
soc.religion.christian	0	0	0	0	0	1	2	0	0	0	0	0	4	0	0	102	0	0	0	1
talk.politics.guns	0	1	0	0	0	0	5	4	1	1	0	1	9	4	0	0	69	1	1	1
talk.politics.mideast	2	2	0	0	0	1	6	1	0	2	0	0	5	0	0	2	0	56	0	0
talk.politics.misc	1	0	0	0	0	0	3	3	1	3	0	0	8	7	0	1	7	0	44	0
talk.religion.misc	1	5	0	1	0	0	3	0	0	1	0	0	4	0	1	10	0	0	0	19

Classifier: LogisticRegression

Accuracy: 0.76

Classification Report: recall f1-score support
Precision

0	0.82	0.51	0.63	78
1	0.69	0.83	0.76	101
2	0.70	0.70	0.70	96
3	0.75	0.78	0.76	110
4	0.88	0.72	0.79	101
5	0.76	0.76	0.76	101
6	0.46	0.85	0.60	93
7	0.78	0.77	0.77	109
8	0.84	0.73	0.78	103
9	0.78	0.83	0.80	101
10	0.98	0.86	0.92	110
11	0.94	0.63	0.76	104
12	0.53	0.78	0.63	79
13	0.76	0.90	0.83	90
14	0.96	0.81	0.88	101
15	0.72	0.92	0.81	110
16	0.77	0.73	0.75	98
17	0.94	0.81	0.87	77
18	0.87	0.59	0.70	78
19	0.76	0.29	0.42	45
accuracy			0.76	1885
macro avg	0.79	0.74	0.75	1885
weighted avg	0.79	0.76	0.76	1885



Confusion Matrix - LogisticRegression

	alt.atheism	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x	misc.forsale	rec.autos	rec.motorcycles	rec.sport.baseball	rec.sport.hockey	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion.christian	talk.politics.guns	talk.politics.mideast	talk.politics.misc	talk.religion.misc
alt.atheism	40	2	0	0	0	3	2	1	2	1	0	0	2	5	0	16	3	0	1	0
comp.graphics	1	84	4	2	0	2	3	0	0	0	1	0	4	0	0	0	0	0	0	0
comp.os.ms-windows.misc	0	4	67	6	2	3	8	0	1	0	0	0	2	2	0	0	0	0	0	1
comp.sys.ibm.pc.hardware	0	1	8	86	2	0	4	2	0	0	0	1	4	1	0	0	0	0	0	1
comp.sys.mac.hardware	0	1	3	6	73	1	12	1	0	0	0	1	2	0	0	1	0	0	0	0
comp.windows.x	0	10	7	1	0	77	1	0	1	1	0	0	2	0	0	1	0	0	0	0
misc.forsale	0	0	1	4	0	0	79	2	2	1	0	0	3	0	0	0	1	0	0	0
rec.autos	0	3	0	0	0	2	9	84	2	0	0	0	4	2	0	0	3	0	0	0
rec.motorcycles	0	3	0	3	1	1	6	5	75	3	0	0	6	0	0	0	0	0	0	0
rec.sport.baseball	1	1	0	2	0	0	7	1	0	84	1	0	1	1	1	1	0	0	0	0
rec.sport.hockey	0	2	1	0	0	0	5	1	0	3	95	0	1	2	0	0	0	0	0	0
sci.crypt	0	2	2	1	2	3	7	3	0	1	0	66	7	3	0	0	5	1	1	0
sci.electronics	0	0	0	4	1	0	4	5	0	0	0	1	62	1	0	0	0	0	0	1
sci.med	0	0	0	0	1	0	2	0	0	1	0	0	4	81	0	0	0	0	1	0
sci.space	0	4	0	0	1	1	4	0	1	1	0	0	4	0	82	2	0	0	1	0
soc.religion.christian	1	0	0	0	0	1	1	0	0	1	0	0	2	0	1	101	1	0	0	1
talk.politics.guns	0	2	2	0	0	1	3	1	1	4	0	0	3	2	0	2	72	2	3	0
talk.politics.mideast	2	0	0	0	0	2	4	1	1	2	0	1	1	0	0	1	0	62	0	0
talk.politics.misc	1	0	0	0	0	1	4	1	3	4	0	0	2	6	0	0	9	1	46	0
talk.religion.misc	3	2	1	0	0	3	5	0	0	1	0	0	1	0	1	15	0	0	0	13

```
Best accuracy model is < main .MultinomialNaiveBayes object at 0x00 000204D5B31810>

Classifier: < main .MultinomialNaiveBayes object at 0x00000204D5B318 10>
Accuracy: 0.85
```

Conclusion: Our analysis reveals that among the algorithms investigated—Naive Bayes, Logistic Regression, Support Vector Classifier (SVC), and Random Forest—Naive Bayes demonstrated the highest efficacy in text categorization. Following Naive Bayes, both Logistic Regression and SVC exhibited comparable performance, while Random Forest yielded the least accurate results among the tested models.

This empirical evidence underscores Naive Bayes as the most proficient algorithm for text categorization within this study, closely trailed by Logistic

Regression and SVC. The comparatively lower performance of Random Forest implies its limitations in effectively addressing the specific nuances of this text classification task.

These findings advocate for prioritizing Naive Bayes as the primary choice for text categorization based on our thorough evaluation of these algorithms.

Bibliography:

- Research Paper on 'Text Classification with Naive Bayes ' by Journal of Applied Technology and Innovation.

  Reference:https://dif7uuh3zqcps.cloudfront.net/wp-content/uploads/sites/11/2021/03/15095538/Volume5_Issue2_Paper1_2021.pdf

- Text Classification using Naive Bayes by simplilearn.

  Reference:-
  google.com/url?q=https://m.youtube.com/watch%3Fv%3D60pqgfT5tZM&sa=U&ved=2ahUKEwjou6CE6KmDAxUkwzgGHc_1CyIQwqsBegQIDRAF&usg=AOvVaw2JaZ0QOgEHBKAwZdzOAq4B

- Classification of Text Documents using the approach of Naïve Bayes by GeeksforGeeks.

  Reference:https://www.geeksforgeeks.org/classification-of-text-documents-using-t he-approach-of-naive-bayes/