CS481- Homework 5AB (report only)
Name: Feven Belay Araya

Perform model evaluation of the classification model of that you developed previously in class.

Requirements:

- Use confusion matrix, F1 score, and Lift/Gain charts at a minimum
- Compare them and discuss similarities and differences as necessary

Homework 4AB- Newsgroup posts

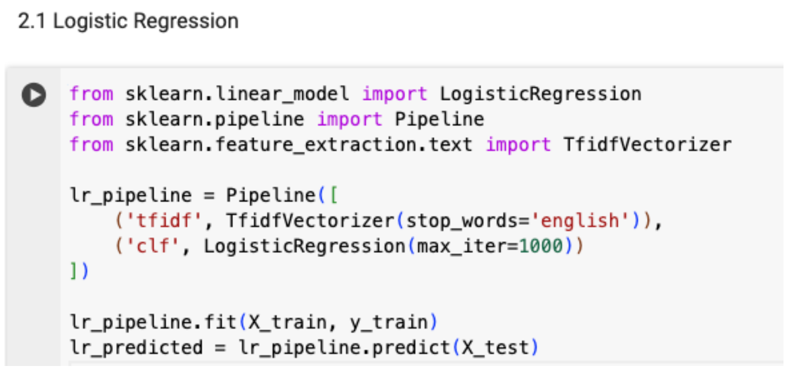
1. Short description of Data and Business Understanding

The 20 Newsgroups dataset encompasses a wide array of documents across 20 diverse topics, offering businesses significant potential for text classification applications to uncover trends in customer preferences, market developments, and competitive landscapes. Analyzing and categorizing the dataset can provide critical insights that influence strategic business decisions, such as customizing marketing efforts and refining product development. This dataset serves as a benchmark for supervised learning in text classification and supports various NLP tasks, including sentiment analysis and author identification, posing challenges due to the noise and variability in document lengths and styles.

2. Data Exploration and text processing

The given project outlines a comprehensive approach to text classification using the 20 Newsgroups dataset, with steps that include loading the dataset, analyzing its structure and distribution, and processing the text for feature extraction. Initial data exploration reveals insights such as the balanced distribution of posts across categories, the variability in post lengths, and the frequency of common vocabulary words, which guide the subsequent text preprocessing steps. For model development, the project employs techniques such as tokenization, TF-IDF vectorization, and employs a combination of machine learning models (like Logistic Regression, Multinomial Naive Bayes and RandomForest) within a voting classifier to enhance prediction accuracy, leveraging both traditional NLP techniques and modern machine learning algorithms to classify text effectively.

3. Classification model using Logistic Regression


2.1 Logistic Regression

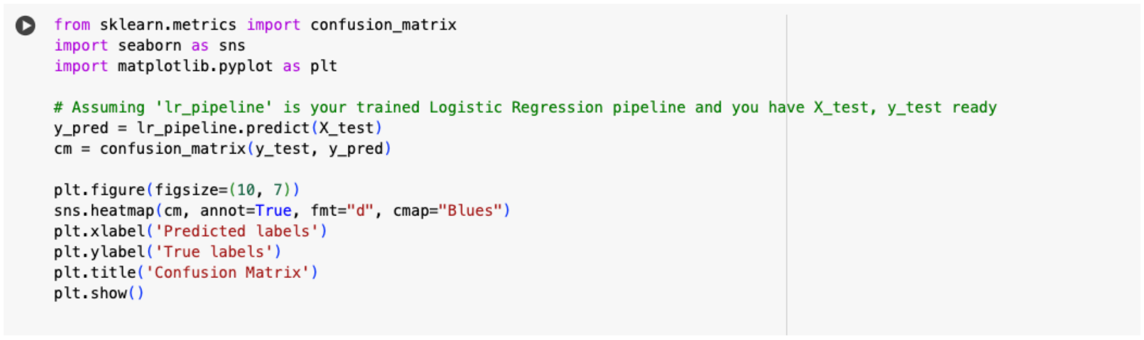
```
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer

lr_pipeline = Pipeline([
    ('tfidf', TfidfVectorizer(stop_words='english')),
    ('clf', LogisticRegression(max_iter=1000))
])

lr_pipeline.fit(X_train, y_train)
lr_predicted = lr_pipeline.predict(X_test)
```

The code snippet establishes a machine learning pipeline that employs a TfidfVectorizer to convert text data into a matrix of TF-IDF features, excluding common English stop words for more effective feature extraction. It then uses a Logistic Regression classifier with a maximum iteration parameter set to 1000 to fit the model to the training data and predict the labels for the test dataset.

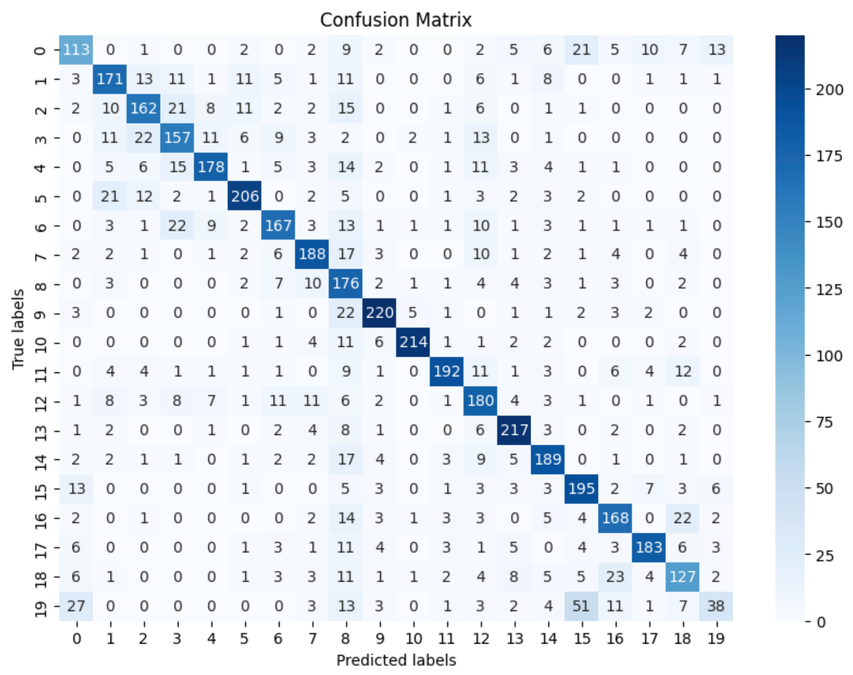
4. Model Evaluation

4.1. Confusion matrix

1. Generate and Visualize the Confusion Matrix



```python
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'lr_pipeline' is your trained Logistic Regression pipeline and you have X_test, y_test ready
y_pred = lr_pipeline.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.title('Confusion Matrix')
plt.show()
```



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	113	0	1	0	0	2	0	2	9	2	0	0	2	5	6	21	5	10	7	13
1	3	171	13	11	1	11	5	1	11	0	0	0	6	1	8	0	0	1	1	1
2	2	10	162	21	8	11	2	2	15	0	0	1	6	0	1	1	0	0	0	0
3	0	11	22	157	11	6	9	3	2	0	2	1	13	0	1	0	0	0	0	0
4	0	5	6	15	178	1	5	3	14	2	0	1	11	3	4	1	1	0	0	0
5	0	21	12	2	1	206	0	2	5	0	0	1	3	2	3	2	0	0	0	0
6	0	3	1	22	9	2	167	3	13	1	1	1	10	1	3	1	1	1	1	0
7	2	2	1	0	1	2	6	188	17	3	0	0	10	1	2	1	4	0	4	0
8	0	3	0	0	0	2	7	10	176	2	1	1	4	4	3	1	3	0	2	0
9	3	0	0	0	0	0	1	0	22	220	5	1	0	1	1	2	3	2	0	0
10	0	0	0	0	0	1	1	4	11	6	214	1	1	2	2	0	0	0	2	0
11	0	4	4	1	1	1	1	0	9	1	0	192	11	1	3	0	6	4	12	0
12	1	8	3	8	7	1	11	11	6	2	0	1	180	4	3	1	0	1	0	1
13	1	2	0	0	1	0	2	4	8	1	0	0	6	217	3	0	2	0	2	0
14	2	2	1	1	0	1	2	2	17	4	0	3	9	5	189	0	1	0	1	0
15	13	0	0	0	0	1	0	0	5	3	0	1	3	3	3	195	2	7	3	6
16	2	0	1	0	0	0	0	2	14	3	1	3	3	0	5	4	168	0	22	2
17	6	0	0	0	0	1	3	1	11	4	0	3	1	5	0	4	3	183	6	3
18	6	1	0	0	0	1	3	3	11	1	1	2	4	8	5	5	23	4	127	2
19	27	0	0	0	0	0	0	3	13	3	0	1	3	2	4	51	11	1	7	38

Observations

1. Correct Predictions (True Positives): The main diagonal represents correct predictions. Class 0 has 113 correct predictions, while class 9 has the highest number of correct predictions at 220.

2. Some of the common Misclassifications:

- Class 5 classified as class 1 with 21 instances.
- Class 6 classided as class 3 with 3 instances.

3. Some of the common classes with High Misclassification Rates:

- Class 1 was misclassified as class 2 for 13 instances and as class 3 for 11 instances.
- Class 2 was misclassified as class 1 for 10 instances and as class 3 for 21 instances,
- Class 19 classified as class 0 with 27 instances ,classified as class 15 with 51 instances.

- Class 0 classified as class 15 with 21 instances and class 19 as 13 instances indicating possible similarity or confusion between these classes.

4. High Precision Classes: Classes with a very high number of predictions concentrated on the diagonal and few off-diagonal entries indicate high precision. For example, class 9 with 220 correct predictions and only a few misclassifications show a high precision.
5. Classes with Lower Precision:

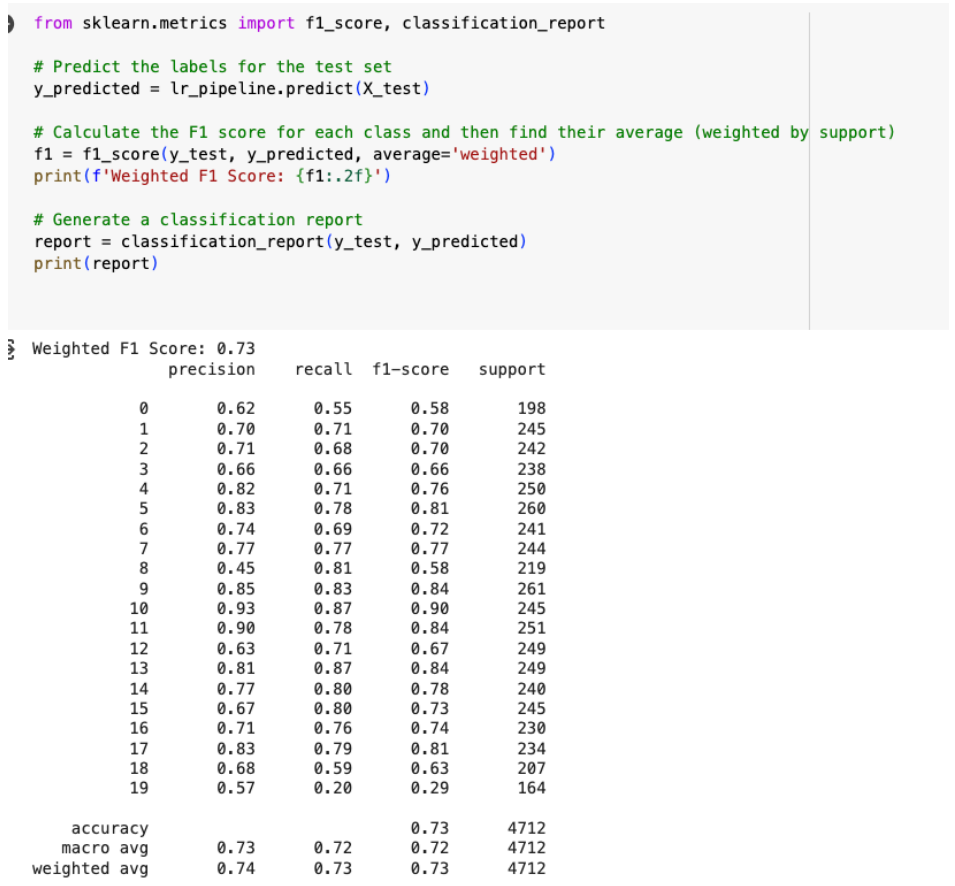
- Class 0 with more scattered misclassifications (21 misclassified as class 10 and several others).
- Class 11 has 192 correct predictions but also has notable misclassifications like 11 instances as class 10 and 12 as class 12.

6. Possible Class Imbalance:

- Some classes, like class 0, class 11, and class 19, have a relatively low number of correct predictions compared to others, which might suggest a class imbalance or that the model struggles to correctly identify these classes.

7. Least Misclassified: Class 18 seems to be the least misclassified, with many of its misclassifications being 5 as class 14 and 4 as class 19, respectively.

4.2. F1 score

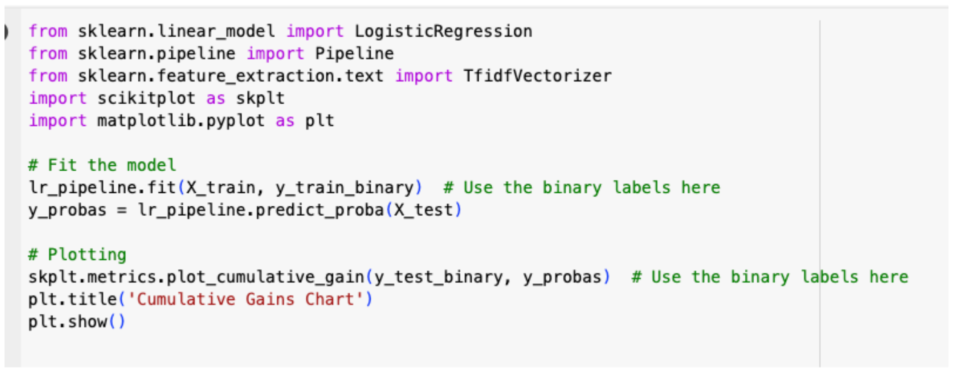


```
from sklearn.metrics import f1_score, classification_report

# Predict the labels for the test set
y_predicted = lr_pipeline.predict(X_test)

# Calculate the F1 score for each class and then find their average (weighted by support)
f1 = f1_score(y_test, y_predicted, average='weighted')
print(f'Weighted F1 Score: {f1:.2f}')

# Generate a classification report
report = classification_report(y_test, y_predicted)
print(report)
```

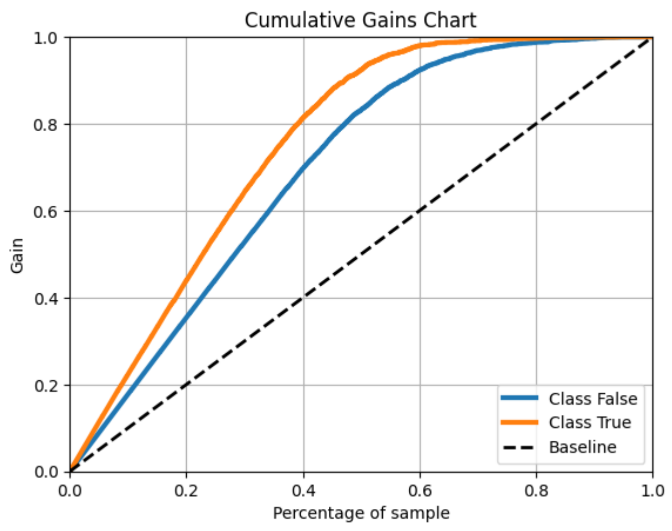
```
Weighted F1 Score: 0.73
              precision    recall  f1-score   support

           0       0.62      0.55      0.58       198
           1       0.70      0.71      0.70       245
           2       0.71      0.68      0.70       242
           3       0.66      0.66      0.66       238
           4       0.82      0.71      0.76       250
           5       0.83      0.78      0.81       260
           6       0.74      0.69      0.72       241
           7       0.77      0.77      0.77       244
           8       0.45      0.81      0.58       219
           9       0.85      0.83      0.84       261
          10       0.93      0.87      0.90       245
          11       0.90      0.78      0.84       251
          12       0.63      0.71      0.67       249
          13       0.81      0.87      0.84       249
          14       0.77      0.80      0.78       240
          15       0.67      0.80      0.73       245
          16       0.71      0.76      0.74       230
          17       0.83      0.79      0.81       234
          18       0.68      0.59      0.63       207
          19       0.57      0.20      0.29       164

    accuracy                           0.73      4712
   macro avg       0.73      0.72      0.72      4712
weighted avg       0.74      0.73      0.73      4712
```

1. Weighted F1 Score: The overall weighted F1 score for the model is 0.73. This score is a weighted average of the F1 scores of each individual class, with the weight being the support (the number of true instances for each class). A weighted F1 score close to 1 indicates a model with good performance, so 0.73 suggests a reasonably good model that could benefit from further optimization.
2. Class-by-Class Metrics:
   - Precision: The precision column indicates how many of the items predicted as belonging to a class actually belong to that class. For example, class 10 has the highest precision of 0.93, meaning it has a high number of true positives compared to false positives.
   - Recall: The recall column shows the ability of the classifier to find all the positive samples. For instance, class 9 has the highest recall of 0.81.
   - F1-Score: The F1-score combines precision and recall into a single metric by taking their harmonic mean. For example, class 10 has the highest F1-score of 0.90, indicating a good balance of precision and recall for this class.
   - Support: This column shows the number of true instances for each class in the test data. The support values vary across classes, indicating the number of samples and possibly reflecting class distribution in the test set.
3. Overall Performance:
   - Classes such as 10 and 15 show strong F1-scores (0.90 and 0.73, respectively), suggesting the model is performing well on these classes.
   - Class 19 has a notably low F1-score of 0.29, with the lowest recall of 0.20, indicating the model struggles to correctly identify this class.
   - The macro-average F1 score is 0.73, and the weighted average F1 score is 0.73, both mirroring the overall weighted F1 score.
4. Areas of Concern:
   - Some classes with low recall may need investigation to understand why the model misses these classes. For example, class 19 has a very low recall, which is a concern.
   - For some classes, there is a significant gap between precision and recall, suggesting that for certain classes, the model may be conservative in its predictions (high precision, low recall) or liberal (low precision, high recall).
5. Model Improvement:
   - To improve the model's performance, one could investigate classes with low F1 scores, especially where there is a significant imbalance between precision and recall.
   - Techniques like resampling, feature selection, or tuning hyperparameters could potentially improve model metrics, especially for classes where the model currently underperforms.

4.3.Lift/Gain charts



```
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
import scikitplot as skplt
import matplotlib.pyplot as plt

# Fit the model
lr_pipeline.fit(X_train, y_train_binary)  # Use the binary labels here
y_probas = lr_pipeline.predict_proba(X_test)

# Plotting
skplt.metrics.plot_cumulative_gain(y_test_binary, y_probas)  # Use the binary labels here
plt.title('Cumulative Gains Chart')
plt.show()
```

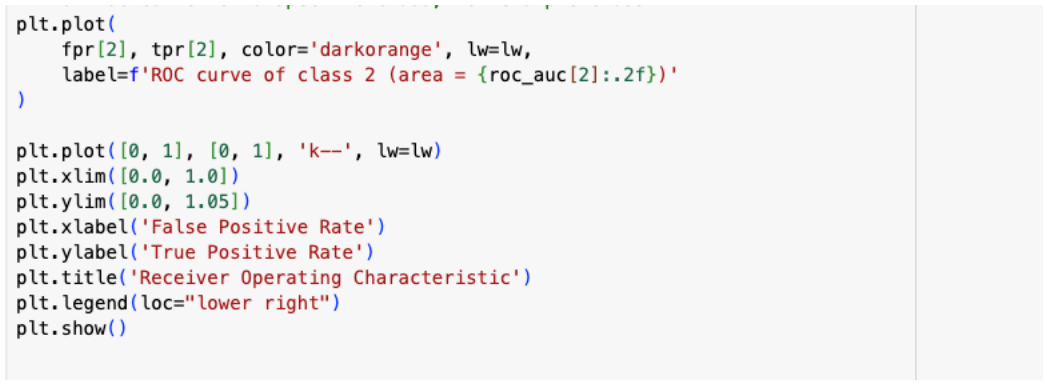


Observations

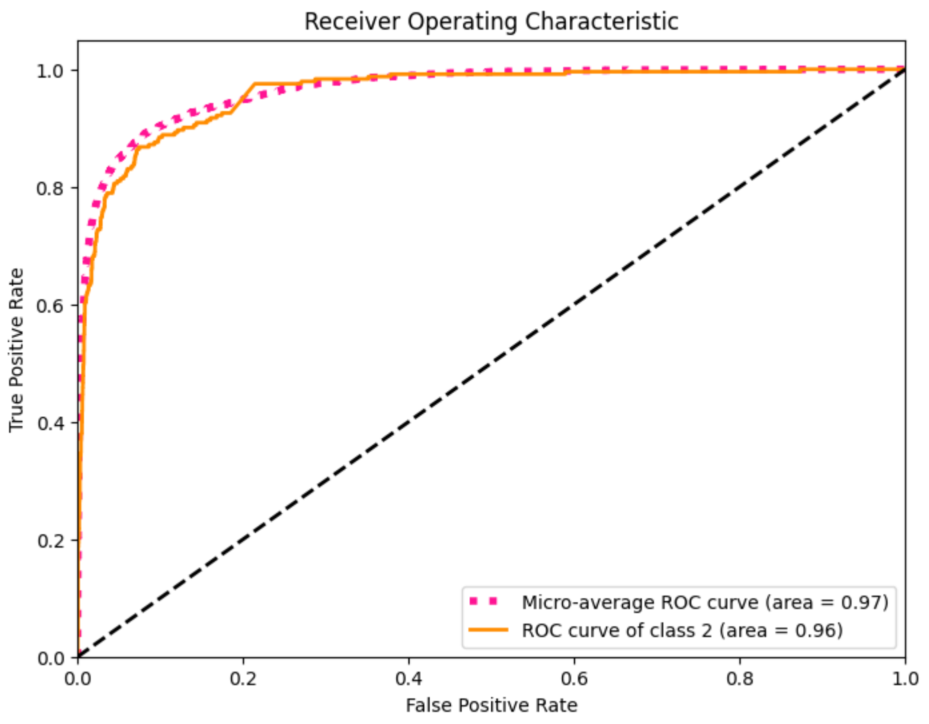
- The x-axis represents the "Percentage of sample", which means the fraction of the total sample ordered by the model from the most likely to belong to the positive class ("Class True") to the least likely.
- The y-axis represents the "Gain", which indicates the proportion of positive outcomes obtained if we contact only a certain percentage of the sample based on the model's predictions.
- The "Class True" curve (orange) represents the gain achieved by using the predictive model. This curve shows how much better one can expect to do with the predictive model compared to random chance.
- The "Class False" curve (blue) could be interpreted as the gain from a negative class, although this interpretation is less common and can depend on the specific context or the way the data is structured.
- The "Baseline" (dashed line) represents the result if we contact a random percentage of the sample. If the model has no predictive power, the gains curve would coincide with the baseline, meaning that there is no advantage over random selection.
- Ideally, the "Class True" curve should be as far away from the baseline as possible, which would indicate that the model is very effective at identifying the positive class.

- Here, the "Class True" curve is above the "Class False" and well above the baseline for all points, which indicates that the model is quite effective. The greater the area between the curve and the baseline, the more effective the model is.
- If we contact about 20% of the sample ordered by the model, we would reach about 60% of the possible total gains for the positive class, which suggests that the model is good at identifying the most promising instances first.

4.4 ROC



```
plt.plot(
    fpr[2], tpr[2], color='darkorange', lw=lw,
    label=f'ROC curve of class 2 (area = {roc_auc[2]:.2f})'
)

plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic')
plt.legend(loc="lower right")
plt.show()
```



Observations

There are two ROC curves shown:

1. The orange solid line represents the ROC curve for "class 2" with an area under the curve (AUC) of 0.96. This is a very good classifier performance, indicating that "class 2" is well discriminated by the model.

2. The pink dashed line represents the "Micro-average ROC curve" with an AUC of 0.97, which suggests an excellent aggregate performance across multiple classes or groups in the dataset.

5. Compare them and discuss similarities and differences as necessary

1. Confusion Matrix:

- A confusion matrix is a detailed table layout that allows visualization of the performance of an algorithm, typically a supervised learning one. Each row of the matrix represents the instances in an actual class, while each column represents the instances in a predicted class.
- The matrix makes it easy to see if the system is confusing two classes (i.e., commonly misclassifying one as another).
- The provided confusion matrix indicates correct and incorrect predictions across multiple classes, highlighting where the model performs well and where it does not.

2. F1 Scores and Classification Report:

- F1 score is a measure of a test's accuracy and considers both the precision and the recall of the test. The weighted F1 score of 0.73 suggests a moderately effective balance of precision and recall across all classes.
- The classification report provides detailed metrics for each class, such as precision, recall, and F1-score, as well as support (the number of true instances for each class).
- This numerical report complements the confusion matrix by providing exact performance figures and could help in pinpointing which specific classes may need further analysis or balancing.

3. Lift/Gain Charts:

- Lift and gain charts are visual aids for evaluating performance of classification models. They show how much more likely we are to receive positive responses than if we were to contact a random sample of cases.
- In the gain chart provided, the "Class True" curve shows the effectiveness of the model in identifying positive instances. A good model's gain curve will rise quickly toward the top-left, indicating that targeting a small percentage of the sample yields a large percentage of the positive cases.

4. ROC Curves:

- ROC curves plot the true positive rate against the false positive rate at various threshold settings. The AUC (area under the curve) is indicative of the model's ability to discriminate between the positive and negative classes.
- The ROC curves indicate that "class 2" and the micro-average of the model are performing very well, with AUC values close to 1.

Similarities and Differences:
Similarities:

- All these methods are aimed at quantifying the performance of classification models.
- They provide insights into aspects like how many predictions were correct, and how the model behaves in terms of false positives and false negatives.
- They can guide a data scientist in improving the model by showing which classes are hard to distinguish or have low recall/precision.

Differences:

- The confusion matrix and classification report provide class-specific information, which is very detailed for individual class analysis.
- The F1 score aggregates the model's performance into a single metric, which is useful for
- comparing models but may hide class-specific performance issues.

- Lift/Gain charts are more focused on the practical application of a model in a business context, showing how much more effective the model is compared to random guessing.
- ROC curves provide a graphical representation of a model’s diagnostic ability and are useful for comparing different models or for choosing an optimal threshold for classification.