Sentiment Analysis Market For Marketing ­­­

**Introduction:** We are focusing on two main innovation approach to enhance the sentiment prediction analysis for market system accuracy and robustness.

**Problem Statement:**

The problem is to perform sentiment analysis on customer feedback to gain insights into competitor products. By understanding customer sentiments, companies can identify strengths and weaknesses in competing products, thereby improving their own offerings. This project requires utilizing various NLP methods to extract valuable insights from customer feedback.

**Scope:**

This project will focus on sentiment analysis of customer reviews, social media mentions, and feedback gathered through various channels, such as online reviews, customer support interactions, and social media platforms. The scope includes sentiment analysis for specific product or brands.

**Design Thinking**

Data Collection: Identify a dataset containing customer reviews and sentiments about competitor products.

1. Data Preprocessing: Clean and preprocess the textual data for analysis.

Sentiment Analysis Techniques: Employ different NLP techniques like Bag of Words,Word Embeddings, or Transformer models for sentiment analysis.

1. Feature Extraction: Extract features and sentiments from the text data

Visualization: Create visualizations to depict the sentiment distribution and analyze trends.

Insights Generation: Extract meaningful insights from the sentiment analysis results to guide business decisions.

**Step 1: Empathize**

User Research: Conduct surveys and interviews with customers to understand their preferences and pain points.

User Personas: Create user personas representing different customer segments.

**Step 2: Define**

Problem Statement Refinement: Based on user research, refine the problem statement to focus on specific products or brand sentiment.

Problem Statement: Analyze customer sentiment towards specific product or brands using Python-based sentiment analysis.

**Step 3: Ideate**

Brainstorming: Brainstorm ideas for Python-based tools and libraries for sentiment analysis.

Prioritization: Prioritize Python-based data sources and analysis techniques based on feasibility and potential impact.

**Step 4: Prototype**

Concept Development: Develop prototypes for Python-based sentiment analysis tools or dashboards.

Tools and Resources: Identify Python libraries and resources needed for sentiment analysis, such as NLP libraries or sentiment analysis APIs.

**Step 5: Test**

Testing Plan: Plan how to collect and label data for training Python-based sentiment analysis models.

Feedback and Iteration: Collect feedback from users or stakeholders on the Python-based prototypes and refine as needed.

**Step 6: Implement**

Implementation Plan: Describe how Python-based sentiment analysis will be integrated into marketing processes.

Timeline: Provide a timeline for implementing Python-based sentiment analysis solutions.

**Step 7: Evaluate**

Evaluation Criteria: Define metrics for evaluating the success of Python-based sentiment analysis, such as accuracy, customer satisfaction, or improved marketing ROI.

Feedback and Adjustments: Describe how feedback will be collected post-implementation and how adjustments will be made.

**Step 8: Deployment**

Deployment Plan: Outline the deployment process for Python-based sentiment analysis tools or dashboards.

**Step 9: Monitoring and Maintenance**

Monitoring Plan: Explain how Python-based sentiment analysis results will be monitored and reported.

Maintenance Strategy: Describe how the Python-based sentiment analysis system will be maintained and updated.

**Step 10: Documentation and Reporting**

Documentation: Detail how the entire design thinking process, from research to implementation using Python, will be documented.

Reporting: Explain how progress and results will be reported.

**Objectives:**

1. Understand customer sentiment towards our products and brand.

2. Identify common pain points and areas of improvement.

3. Track sentiment trends over time.

4. Improve marketing strategies based on sentiment insights.

**Describe the Dataset used, Data preprocessing Steps and Sentiment analysis techniques.**

Describe the Dataset :-

The "Twitter US Airline Sentiment" dataset on Kaggle is a collection of Twitter posts (tweets) related to the sentiment of travelers toward various U.S. airlines. This dataset is often used for sentiment analysis and natural language processing (NLP) tasks. Here is a description of the dataset:

Size: The dataset typically contains around 14,640 tweets.

Features: The dataset includes the following columns or features:

* tweet\_id: A unique identifier for each tweet.
* airline\_sentiment: The sentiment of the tweet, which can be one of the following categories: "positive," "negative," or "neutral." This is the target variable for sentiment analysis.
* airline\_sentiment\_confidence: The confidence level of the sentiment classification.
* negativereason: The reason for a negative sentiment, if applicable. This field is often empty for positive and neutral tweets.
* negativereason\_confidence: The confidence level of the negative reason classification.
* airline: The name of the airline being mentioned in the tweet (e.g., United, Delta, American, etc.).
* airline\_sentiment\_gold: A gold standard sentiment label.
* name: The Twitter handle of the user who posted the tweet.
* retweet\_count: The number of retweets the tweet received.
* text: The actual text of the tweet.
* Purpose: This dataset is often used for sentiment analysis and NLP tasks to determine the overall sentiment of Twitter users toward different U.S. airlines. It can be useful for airlines and researchers to understand customer opinions and identify areas for improvement.
* Sentiment Analysis: Many analyses and machine learning models are built on this dataset to classify tweets into positive, negative, or neutral sentiments based on the text content and associated metadata.
* Source: The data was collected from Twitter using the Twitter API.

**Data Preprocessing Steps and Sentiment analysis techniques:-**

**Ensemble Methods** - Ensemble methods involve combining multiple machine learning models to improve prediction accuracy and generalization.

* This can be achieved through techniques such as averaging, bagging, boosting, or stacking.

**Approach:**

* Train a variety of sentiment analysis models using different algorithms or variations of the same algorithm.
* Utilize methods like bagging (e.g., Random Forest), boosting (e.g., Ada Boost), or stacking to combine predictions from these models.
* Experiment with different combinations of models and ensemble techniques to find the most effective approach for improving accuracy and robustness.

**Fine-Tuning Pre-trained Sentiment Analysis Models:**

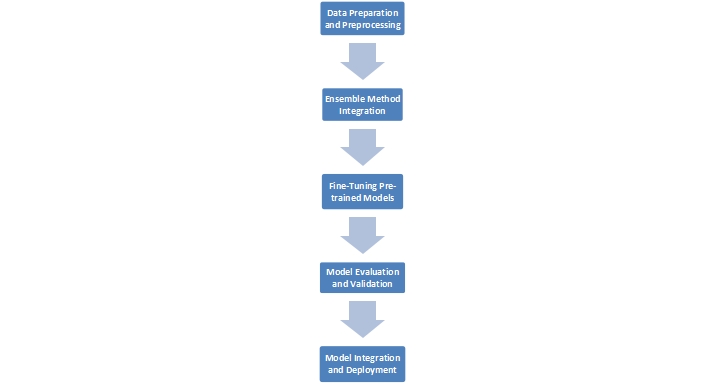
* Fine-tuning involves taking pre-trained models (e.g., BERT, RoBERTa) and adapting them to a specific task, in this case, sentiment analysis.
* Fine-tuning allows the model to learn task-specific patterns and nuances from the provided dataset.

**Approach:**

* Obtain pre-trained sentiment analysis models such as BERT and RoBERTa.
* Fine-tune these models on the Twitter airline sentiment dataset to adapt them for sentiment prediction.
* Customize the model architecture and hyper parameters for optimal performance on the sentiment analysis task.
* Experiment with different learning rates, batch sizes, and training epochs to achieve the best results.
* Evaluate the fine-tuned models on a validation set and fine-tune further if needed to improve accuracy.

**Proposed Design and Transformation Steps:**

**Simply represent each step and the flow of the process visually to provide a clear overview of the approach.**



**Step 1:** **Data Preparation and Preprocessing**

**Data Collection:**

Gather a diverse and representative dataset containing labeled sentiment data for training and validation.

Here we are using the data which given by (IBM).

**Dataset Link:** [**https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment**](https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment)

**Data Cleaning and Preprocessing:**

Perform data cleaning, removing irrelevant information, handling missing values, and ensuring consistent formatting.

**Step 2:** **Ensemble Method Integration :**

* **Ensemble Model Selection:** Choose a set of diverse base sentiment analysis models (e.g., BERT, RoBERTa, traditional machine learning models).
* **Training Base Models:** Train each base model on the preprocessed training data.
* **Ensemble Creation:** Implement an ensemble methods(Which I mentioned above ) to combine predictions from multiple base models.

**Step 3:Fine-Tuning Pre-trained Models:**

* **Model Selection:** Choose pre-trained sentiment analysis models (e.g., BERT, RoBERTa) based on the requirements and data domain.
* **Fine-Tuning Process:** Fine-tune the selected pre-trained models using the preprocessed training data to adapt them to the specific sentiment prediction task.
* **Hyperparameter Tuning:** Optimize hyperparameters using techniques such as grid search or random search to enhance the model's performance.

**Step 4 :** **Model Evaluation and Validation:**

* **Evaluation Metrics:** Select appropriate evaluation metrics (e.g., accuracy, precision, recall, F1-score) to assess the performance of the ensemble and fine-tuned models.
* **Validation on Test Set:** Evaluate the performance of the ensemble and fine-tuned models using a separate test dataset to ensure generalizability.

**Step 5:** **Model Integration and Deployment**

* **Integration:** - Integrate the ensemble and fine-tuned models into a unified prediction system.
* **API Development:** - Develop an API to facilitate easy access to the prediction system.
* **Deployment:** - Deploy the sentiment prediction system to a scalable and reliable cloud infrastructure for real-time prediction capabilities.

import pandas as pd

df = pd.read\_csv('/content/Tweets.csv')

print(df)



**EDA(Exploratory data analysis) Using Matplotlib for visualization**

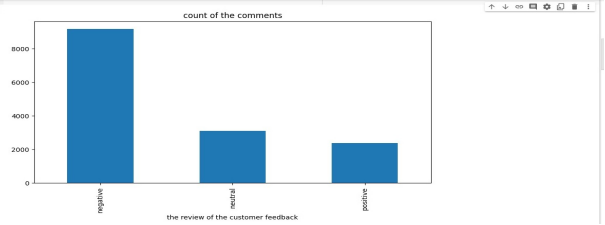
import matplotlib.pyplot as plt

import numpy as np

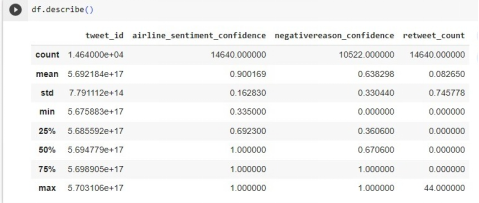
ax = df['airline\_sentiment'].value\_counts().sort\_index().plot(kind ='bar', title='count of the comments',figsize=(10, 5))

ax.set\_xlabel('the review of the customer feedback')

plt.show()



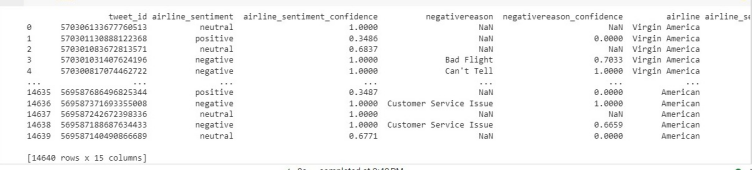
df.describe()



pd.set\_option('display.max\_columns',None)

pd.set\_option('display.expand\_frame\_repr',False)

print(df)



df = pd.read\_csv('Tweets.csv')

null= df.isnull().any().any()

if null:

print("null values in Tweets dataset.")

else:

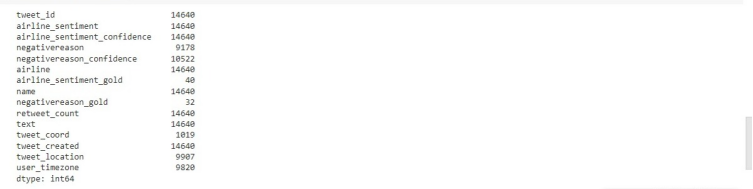
 print("No null values in Tweets dataset.")

df.info()

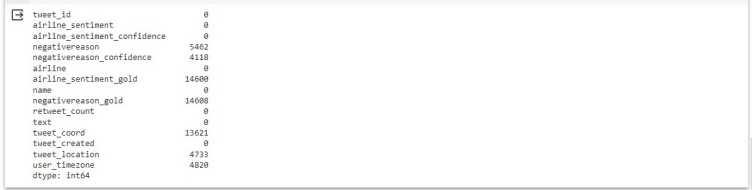


print(df.dropna())

print(df.notnull().sum())

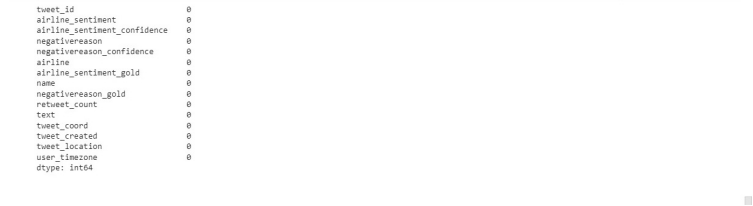


df.isnull().sum()



df.dropna(subset=['negativereason','negativereason\_confidence','airline\_sentiment\_gold','negativereason\_gold','tweet\_coord','tweet\_location','user\_timezone'],inplace=True)

df.isnull().sum()



**Employing NLP techniques & Generating insights.**

Continue building the sentiment analysis solution by Employing NLP techniques Generating insights using below dataset

**Dataset Link:** [**https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment**](https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment)

**Employing NLP techniques**

import nltk.data

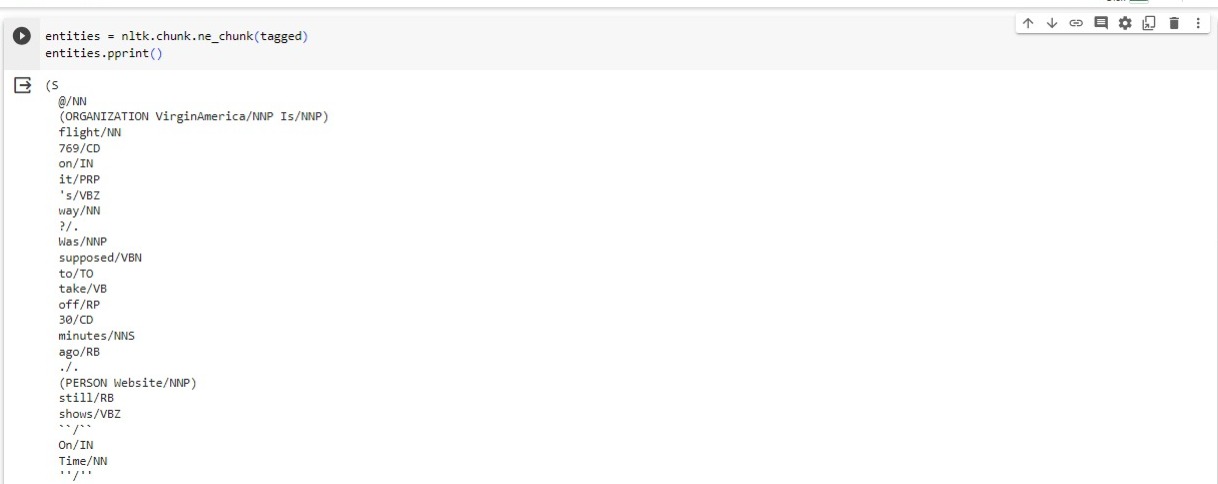
tokens = nltk.word\_tokenize(example)

tokens[:10]

tagged = nltk.pos\_tag(tokens)

tagged[:10] 

entities = nltk.chunk.ne\_chunk(tagged)

entities.pprint()

**Generating insights (**VADER APPROACH**):-**

from nltk.sentiment import SentimentIntensityAnalyzer

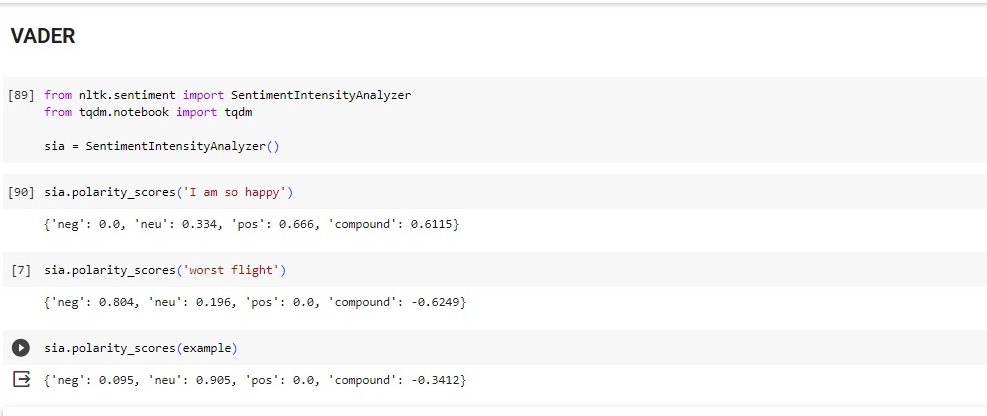
from tqdm.notebook import tqdm

sia = SentimentIntensityAnalyzer()

sia.polarity\_scores('I am so happy')

sia.polarity\_scores('worst flight')

sia.polarity\_scores(example)



THE POLARITY SCORE USING LOOP

res={}

for i, row in tqdm (df.iterrows(), total= len(df)):

text = row['text']

id = row['tweet\_id']

res[id] = sia.polarity\_scores(text)



vaders = pd.DataFrame(res).T

vaders = vaders.reset\_index().rename(columns={'index':'tweet\_id'})

vaders = vaders.merge(df,how='left')

ax = sns.barplot(data=vaders, x ='airline\_sentiment\_confidence', y ='compound')

ax.set\_title("TWEETS FLIGHT CUSTOMER REVIEW")

plt.show()



fig, axs = plt.subplots(1,3 , figsize=(15,5))

sns.scatterplot(data=vaders,x ='airline\_sentiment\_confidence', y ='pos', ax = axs[0])

sns.scatterplot(data=vaders,x ='airline\_sentiment\_confidence', y ='neu', ax = axs[1])

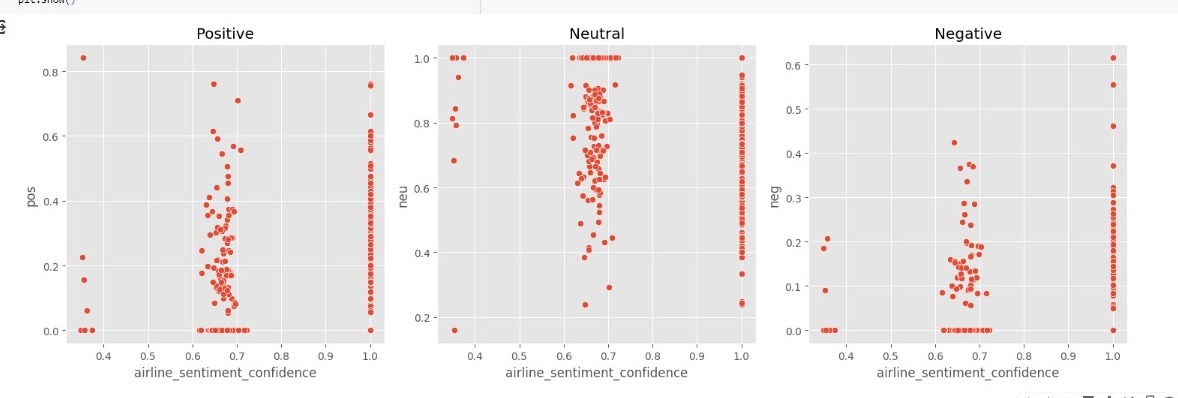
sns.scatterplot(data=vaders,x ='airline\_sentiment\_confidence', y ='neg', ax = axs[2])

axs[0].set\_title('Positive')

axs[1].set\_title('Neutral')

axs[2].set\_title('Negative')

plt.tight\_layout()

plt.show()

**Roberta pretrained model**

! pip install transformers

from transformers import AutoTokenizer

from transformers import AutoModelForSequenceClassification

from scipy.special import softmax

MODEL = f'cardiffnlp/twitter-roberta-base-sentiment'

tokenizer = AutoTokenizer.from\_pretrained(MODEL)

model = AutoModelForSequenceClassification.from\_pretrained(MODEL)

#VADER result on examples

print(example)

sia.polarity\_scores(example)



#Run roberta model

def polarity\_scores\_roberta(example):

encoded\_text = tokenizer(example, return\_tensors='pt')

output = model(\*\*encoded\_text)

scores = output[0][0].detach().numpy()

scores = softmax(scores)

scores\_dict = {

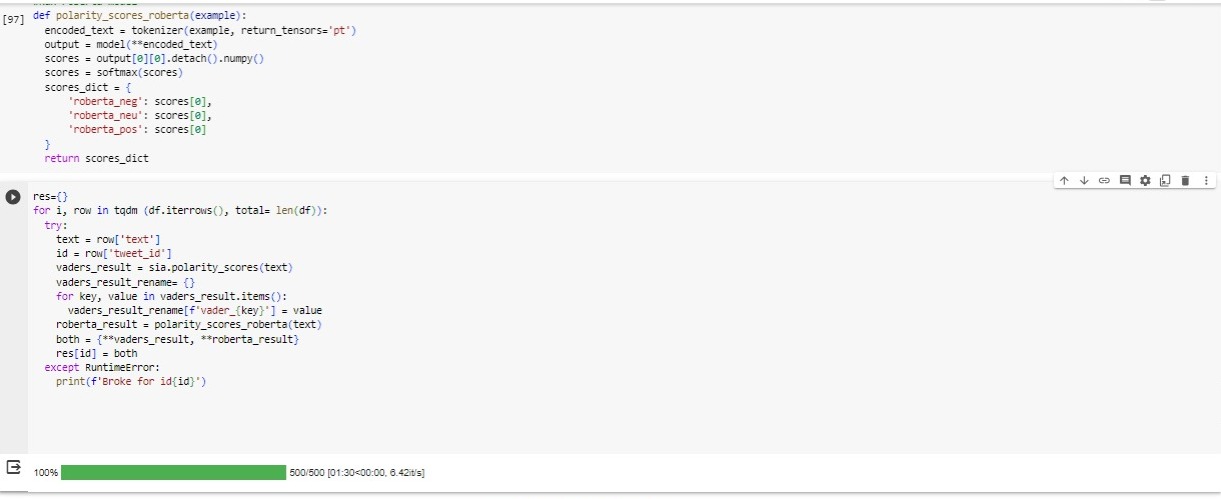
'roberta\_neg': scores[0],

'roberta\_neu': scores[0],

'roberta\_pos': scores[0]

}

return scores\_dict



res={}

for i, row in tqdm (df.iterrows(), total= len(df)):

try:

text = row['text']

id = row['tweet\_id']

vaders\_result = sia.polarity\_scores(text)

vaders\_result\_rename= {}

for key, value in vaders\_result.items():

vaders\_result\_rename[f'vader\_{key}'] = value

roberta\_result = polarity\_scores\_roberta(text)

both = {\*\*vaders\_result, \*\*roberta\_result}

res[id] = both

except RuntimeError:

print(f'Broke for id{id}')

**Combine and Compare**

result\_df = pd.DataFrame(res).T

result\_df = result\_df.reset\_index().rename(columns={'index':'tweet\_id'})

result\_df = result\_df.merge(df,how='left')

result\_df.columns

sns.pairplot(data=result\_df,

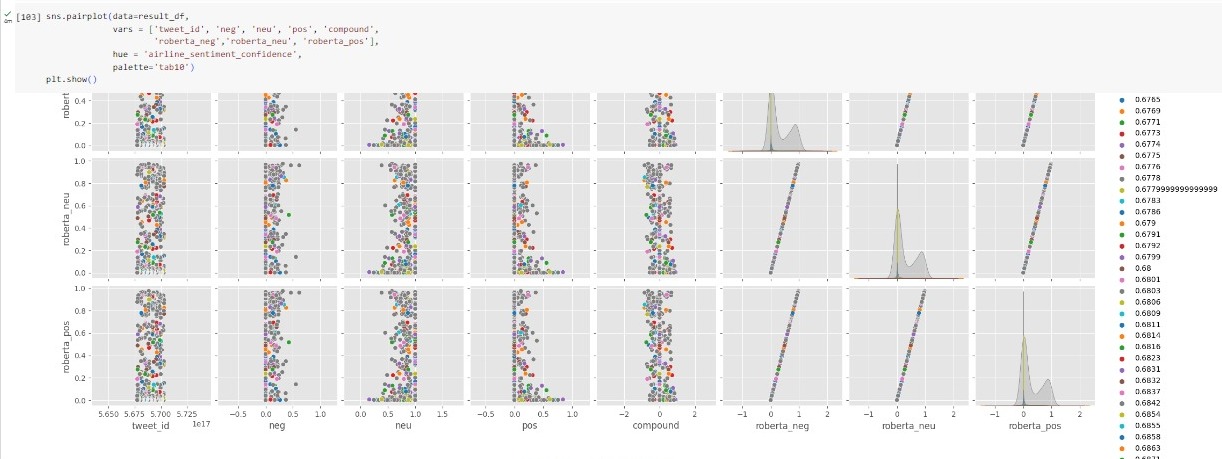
vars = ['tweet\_id', 'neg', 'neu', 'pos', 'compound',

'roberta\_neg','roberta\_neu', 'roberta\_pos'],

hue = 'airline\_sentiment\_confidence',

palette='tab10')

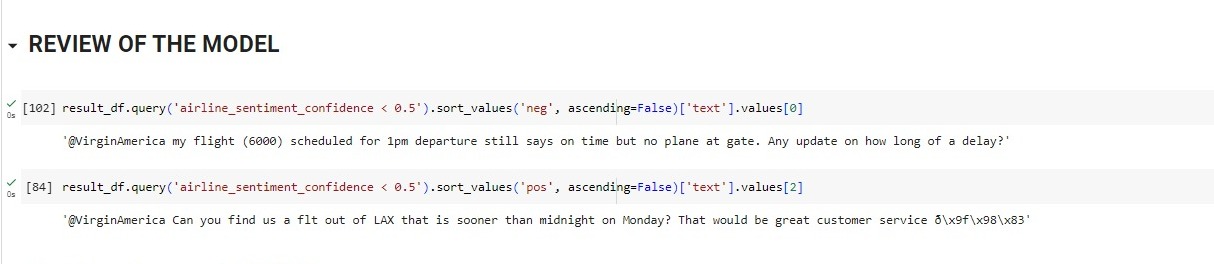
plt.show()



**REVIEW OF THE MODEL(OUTPUT)**

result\_df.query('airline\_sentiment\_confidence < 0.5').sort\_values('neg', ascending=False)['text'].values[0]

result\_df.query('airline\_sentiment\_confidence < 0.5').sort\_values('pos', ascending=False)['text'].values[2]



**The Transformers PIPELINE**

from transformers import pipeline

sent\_pipeline = pipeline("sentiment-analysis")

sent\_pipeline('i like this flight')

sent\_pipeline('this a worst flight ever')



**Conclusion :**

By combining the power of ensemble methods and the precision of fine-tuned pre-trained models, you can significantly enhance the accuracy and robustness of the sentiment prediction system and we aim to gain valuable insights that will inform marketing strategies, enhance customer satisfaction, and drive business growth.

**THE END**

.