Project Title: Car Rental Customer Feedback Analyzer

Name: Manjiswari J

Roll Number: 23BCE9340

College Name: VIT-AP University

Date of Submission: 30-06-2025

Table of Content:-

1. Introduction

The Car Rental Customer Feedback Analyzer is designed to extract valuable insights from customer reviews of car rental services. It focuses on identifying the **sentiment** of each review, detecting **frequently mentioned issues** such as late delivery or poor vehicle condition, and generating a clear **performance summary**. By using prompt-based methods through **IBM watsonx.ai** and **Prompt Lab**, the project helps service teams understand customer satisfaction and improve service delivery.

2. Objective

- To analyze customer reviews and determine the **sentiment** (positive, negative, neutral).
- To extract **key issues** and complaints using prompt-based phrase detection.
- To automate the process of **generating summary reports** for service managers.
- To provide insights that support **better decision-making** and improve customer experience.

3. Tools & Technologies

- **IBM watsonx.ai** Used for prompt-based AI analysis and sentiment classification.
- **Prompt Lab** Utilized to design and test prompt templates for sentiment and key phrase extraction.
- **IBM Cloud Object Storage** Used to securely store and access the customer review dataset.
- **CSV/XLS Files** Employed for structured storage and processing of review data.
- **Python** Used for data pre-processing, API integration, and automating report generation.

4. Methodology / Working

- **Data Generation**
  → a synthetic dataset of customer reviews was created using ChatGPT and stored in CSV/XLS format.
- **Prompt-Based Analysis using watsonx.ai**
  → Custom prompts were designed and tested in Prompt Lab and executed via IBM watsonx.ai to perform two main tasks:
  - **Sentiment Classification** – Classify each review as *Positive*, *Negative*, or *Neutral*.
  - **Issue Detection** – Identify commonly recurring issues such as "late delivery" or "vehicle condition".
- **Summary Generation**
  → a performance summary was generated to highlight sentiment trends and frequently mentioned issues.
- **Reporting**
  → the final output was structured into a readable report format to support decision-making by service managers.

5. Code Snippets with Explanation

1. Read Data from IBM Cloud Object Storage

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.

cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='zps81BY0ZH-GWdX4u_iTP0vzfmqM65yDyVzEp9khSOCF',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/identity/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.direct.us-south.cloud-object-storage.appdomain.cloud')

bucket = 'vitibmgenaispace-donotdelete-pr-jojabgekg7vq75'
object_key = 'car_rental_reviews_1.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

data = pd.read_csv(body)
data.head(10)
```

Explanation:

- Connects to IBM Cloud Object Storage.
- Reads the customer review CSV file directly into a pandas Data Frame.

2. Sentiment Classifier Function with Few-Shot Prompting

```
def classify_sentiment(review):
    prompt_input = f'''You are a sentiment classification assistant.

    Your task is to read a customer review and classify its sentiment using **only one word**: Positive, Negative, or Neutral.

    Do not explain. Do not generate full sentences. Only respond with one of the following: Positive, Negative, or Neutral.

    Review: The representative was very helpful and resolved my issue quickly.
    Sentiment: Positive

    Review: I had to wait an extra hour for the car to be ready. Very disappointing.
    Sentiment: Negative

    Review: The experience was okay, nothing too great or too bad.
    Sentiment: Neutral

    Review: {review}
    Sentiment:'''
    try:
        response = model.generate_text(prompt=prompt_input, guardrails=True)
        if response:
            return response.strip().split()[0].capitalize()
        else:
            return "Unknown"
    except Exception as e:
        print("Error in generation:", e)
        return "Unknown"
```

Explanation:

- Sends each review to the **IBM Granite model**.
- Returns only the **sentiment word** (Positive, Negative, or Neutral).
- Uses generate_text() to get model output.

3. Issue Extraction Function with Few-Shot Prompting

```
def extract_issue(review):
    prompt_input = f'''You are a customer feedback analysis assistant.

    Your task is to read customer service reviews and extract the main issue or complaint mentioned in each review.

    - Only extract one short phrase for each review that clearly describes the issue.
    - If no clear issue is mentioned, respond with: "No issue mentioned".

    Review: I had to wait an extra hour for the car to be ready. Very disappointing.
    Issue:  Late delivery

    Review: The car I reserved was not available, and I had to take a smaller one.
    Issue: Wrong car provided

    Review: The representative was very helpful and resolved my issue quickly.
    Issue: No issue mentioned

    Review: The car was dirty and smelled bad when I picked it up.
    Issue: Poor car condition

    Review: I was charged extra even though I returned the car on time.
    Issue: Unfair extra charges

    Review: The pickup process took forever, and the line was too long.
    Issue: Slow pickup process

    Review: {review}
    Issue:'''
    try:
        response = model.generate_text(prompt=prompt_input, guardrails=True)
        if response:
            return response.strip().split("\n")[0]
        else:
            return "Unknown"
    except Exception as e:
        print("Error in generation:", e)
        return "Unknown"
```

Explanation:

- Prompts the model to extract **specific problems** like "Late delivery", "Poor service", etc.
- If nothing is wrong in the review, it outputs: **"No issue mentioned"**.

4. Summary Report Generation with Prompt-Based Analysis

```
summary_prompt = f'''
You are a customer feedback analysis assistant.

Your task is to:
1. Summarize the distribution of customer sentiments.
2. Identify the most frequently mentioned issues.
3. Provide a suggestion for improvement based on the issues and sentiment data.

Use clear and concise natural language suitable for a business report in 200 words only.

Sentiment distribution:
{data["Sentiment"].value_counts().to_string()}

Issue frequency:
{data["Issues"].value_counts().to_string()}
'''
parameters = {
    "decoding_method": "sample",
    "max_new_tokens": 300,
    "min_new_tokens": 0,
    "temperature": 0,
    "top_k": 50,
    "top_p": 1,
    "repetition_penalty": 1
}
model2 = ModelInference(
    model_id = model_id,
    params = parameters,
    credentials = credentials,
    project_id = project_id,
    space_id = space_id
    )
response = model2.generate_text(prompt=summary_prompt, guardrails=True).strip()

from IPython.display import Markdown, display
display(Markdown(response))
```

Explanation:

- Adds sentiment and issue data into a prompt.
- Asks the model to write a short summary and give suggestions.
- Sets parameters to control the model's response.
- Sends the prompt to the IBM Granite model.
- Returns a clear report that managers can use.

6. Screenshots/Output Results with proper explanation

[1]:
```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.

cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='zps81BY0ZH-GWdX4u_iTP0vzfmqM65yDyVzEp9khSOCF',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/identity/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.direct.us-south.cloud-object-storage.appdomain.cloud')

bucket = 'vitibmgenaispace-donotdelete-pr-jojabgekg7vq75'
object_key = 'car_rental_reviews_1.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

data = pd.read_csv(body)
data.head(10)
```

[1]:

	Customer_Service
0	The staff at the counter was rude and unhelpful.
1	The representative was very helpful and resolv...
2	Everything went smoothly and the car was clean...
3	Everything went smoothly and the car was clean...
4	The staff at the counter was rude and unhelpful.
5	The representative was very helpful and resolv...
6	They billed me incorrectly and support hasn't ...
7	The pickup process took forever, and the line ...
8	The staff was friendly and professional, great...
9	The staff was friendly and professional, great...

Explanation

The code connects to IBM Cloud Object Storage and loads the car_rental_reviews_1.csv file using the ibm_boto3 client. The data is then read using pandas.read_csv() and the first 10 customer reviews are displayed to check if the file loaded correctly.

[2]:
```
import os
from ibm_watsonx_ai import APIClient, Credentials
from ibm_watsonx_ai.foundation_models import ModelInference
import getpass

credentials = Credentials(
    url="https://us-south.ml.cloud.ibm.com",
    api_key="a85IEEsDCDzylXviEcIJ_thzXrKvmhV3QzhHfuJkt1wc"
)
model_id = "ibm/granite-3-2b-instruct"
parameters = {
    "decoding_method": "sample",
    "max_new_tokens": 10,
    "min_new_tokens": 0,
    "temperature": 0,
    "top_k": 50,
    "top_p": 1,
    "repetition_penalty": 1
}
project_id = os.getenv("PROJECT_ID")
space_id = os.getenv("SPACE_ID")
model = ModelInference(
    model_id = model_id,
    params = parameters,
    credentials = credentials,
    project_id = project_id,
    space_id = space_id
    )
```

Explanation

This code connects to the **IBM watsonx.ai platform** and sets up the foundation model ibm/granite-3-2b-instruct.

- The api_key and URL authenticate the user.
- project_id and space_id connect the model to the correct IBM Cloud project.
- The parameters define how the model should generate responses (like number of tokens, randomness, etc.).
- The model is now ready to accept prompts for **sentiment analysis and issue extraction**.

[3]:
```
def classify_sentiment(review):
    prompt_input = f'''You are a sentiment classification assistant.

    Your task is to read a customer review and classify its sentiment using **only one word**: Positive, Negative, or Neutral.

    Do not explain. Do not generate full sentences. Only respond with one of the following: Positive, Negative, or Neutral.

    Review: The representative was very helpful and resolved my issue quickly.
    Sentiment: Positive

    Review: I had to wait an extra hour for the car to be ready. Very disappointing.
    Sentiment: Negative

    Review: The experience was okay, nothing too great or too bad.
    Sentiment: Neutral

    Review: {review}
    Sentiment:'''
    try:
        response = model.generate_text(prompt=prompt_input, guardrails=True)
        if response:
            return response.strip().split()[0].capitalize()
        else:
            return "Unknown"
    except Exception as e:
        print("Error in generation:", e)
        return "Unknown"
```

[4]:
```
output = classify_sentiment(data["Customer_Service"][0])
output
```

[4]:
```
'Negative'
```

Explanation

This code defines a function called classify_sentiment() which uses IBM watsonx.ai to find out whether a customer review is positive, negative, or neutral. It creates a prompt with example reviews and sends it to the model. The model reads the new review and replies with just one word showing the sentiment. In the screenshot, this function is used on the first review from the dataset, and the model correctly returns "Negative" as the sentiment.

[11]:
```python
def extract_issue(review):
    prompt_input = f'''You are a customer feedback analysis assistant.

    Your task is to read customer service reviews and extract the main issue or complaint mentioned in each review.

    - Only extract one short phrase for each review that clearly describes the issue.
    - If no clear issue is mentioned, respond with: "No issue mentioned".

    Review: I had to wait an extra hour for the car to be ready. Very disappointing.
    Issue:  Late delivery

    Review: The car I reserved was not available, and I had to take a smaller one.
    Issue: Wrong car provided

    Review: The representative was very helpful and resolved my issue quickly.
    Issue: No issue mentioned

    Review: The car was dirty and smelled bad when I picked it up.
    Issue: Poor car condition

    Review: I was charged extra even though I returned the car on time.
    Issue: Unfair extra charges

    Review: The pickup process took forever, and the line was too long.
    Issue: Slow pickup process

    Review: {review}
    Issue:'''
    try:
        response = model.generate_text(prompt=prompt_input, guardrails=True)
        if response:
            return response.strip().split("\n")[0]
        else:
            return "Unknown"
    except Exception as e:
        print("Error in generation:", e)
        return "Unknown"
```

[12]:
```python
output2 = extract_issue(data["Customer_Service"][0])
output2
```

[12]:
```
'Rude staff'
```

Explanation

This code defines a function called extract_issue() which uses IBM watsonx.ai to identify the main problem mentioned in a customer review. The prompt includes examples of different reviews and their issues to guide the model. When a new review is passed, the model returns a short phrase that summarizes the main complaint. If no problem is found, it replies with "No issue mentioned." In the screenshot, the function is used on the first review, and the model correctly identifies the issue as "Rude staff".

[13]:
```
sentiments = []
for review in data["Customer_Service"]:
    sentiments.append(classify_sentiment(review))
data["Sentiment"] = sentiments
data
```

[13]:

	Customer_Service	Sentiment
0	The staff at the counter was rude and unhelpful.	Negative
1	The representative was very helpful and resolv...	Positive
2	Everything went smoothly and the car was clean...	Positive
3	Everything went smoothly and the car was clean...	Positive
4	The staff at the counter was rude and unhelpful.	Negative
...	...	...
95	The representative was very helpful and resolv...	Positive
96	Very happy with the car and service, would ren...	Positive
97	The representative was very helpful and resolv...	Positive
98	The staff at the counter was rude and unhelpful.	Negative
99	Decent rental, but not memorable.	Neutral

100 rows × 2 columns

Explanation

This section loops through every review in the “Customer Service” column, calls classify_sentiment() to determine whether each comment is Positive, Negative, or Neutral, and stores the result in a new “Sentiment” column of the DataFrame. When the updated table is displayed, you can immediately see each review paired with its predicted sentiment—for example, remarks about rude counter staff are tagged Negative, while comments praising helpful representatives are tagged Positive, and an unremarkable experience is tagged Neutral.

[14]:
```
issues = []
for review in data["Customer_Service"]:
    issues.append(extract_issue(review))
data["Issues"] = issues
data
```

[14]:

	Customer_Service	Sentiment	Issues
0	The staff at the counter was rude and unhelpful.	Negative	Poor customer service
1	The representative was very helpful and resolv...	Positive	No issue mentioned
2	Everything went smoothly and the car was clean...	Positive	No issue mentioned
3	Everything went smoothly and the car was clean...	Positive	No issue mentioned
4	The staff at the counter was rude and unhelpful.	Negative	Poor customer service
...	...	...	...
95	The representative was very helpful and resolv...	Positive	No issue mentioned
96	Very happy with the car and service, would ren...	Positive	No issue mentioned
97	The representative was very helpful and resolv...	Positive	No issue mentioned
98	The staff at the counter was rude and unhelpful.	Negative	Poor customer service
99	Decent rental, but not memorable.	Neutral	No obvious issue mentioned

100 rows × 3 columns

Explanation

This part of the code processes each customer review to identify the specific issue mentioned. It uses the extract_issue() function on every review in the "Customer_Service" column and stores the result in a new column called "Issues". The updated table now shows three columns: the original review, the sentiment (Positive, Negative, or Neutral), and the main issue detected (e.g., "Poor customer service" or "No issue mentioned"). This helps in quickly understanding not just how customers feel, but also **why** they feel that way.

```
[16]: data["Sentiment"].value_counts()
```

```
[16]: Sentiment
      Negative    45
      Positive    30
      Neutral     24
      None         1
      Name: count, dtype: int64
```

Explanation

This output shows the total count of each sentiment type found in the customer reviews. It reveals that **45 reviews** were classified as **Negative**, **30 as Positive**, and **24 as Neutral**, while **1 entry** was either missing or not properly processed (shown as “None”). This summary helps the team understand the overall mood of customer feedback at a glance.

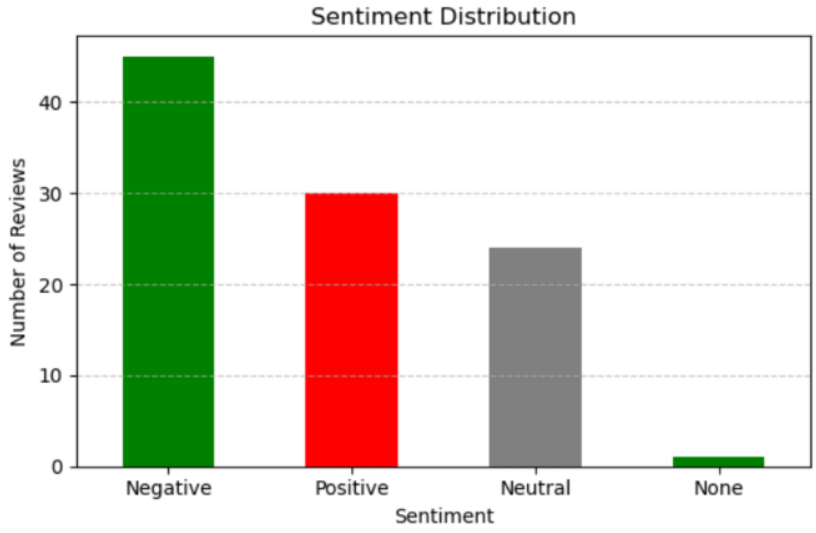
```
[17]: data["Issues"].value_counts()
```

```
[17]: Issues
      No issue mentioned                                  44
      Incorrect billing                                    7
      Poor car condition                                   5
      Unfair extra charges                                 4
      Missed reservation                                   4
      Poor customer service                                3
      No clear issue mentioned                             2
      Late delivery (This review is a repeat of            1
      Poor service at the counter                          1
      Slow pickup process (Repeat)                         1
      Slow pickup process (Repeats the issue               1
      Availability issue                                   1
      Unremarkable experience                              1
      No issue mentioned (neutral)                         1
      Negative customer service experience                 1
      Car not available as expected                        1
      Poor staff behavior                                  1
      Late delivery (Duplicate, consider the context)      1
      No significant issues or praise                      1
      Late delivery (same as previous review, no           1
      Missed expectations                                  1
      No significant issues or complaints                  1
      Neutral sentiments                                   1
      No significant issue                                 1
      Car not as expected                                  1
      Unfair extra charges (duplicate, no change in        1
      Unfulfilled reservation                              1
      Unsatisfactory service                               1
      Incorrect billing, support inadequate                1
      No major issues, nothing remarkable                  1
      Incorrect billing and unresolved support             1
      Neutral experience                                   1
      Late delivery (same as previous review)              1
      Late delivery (duplicate, consider the first review  1
      Late delivery (already mentioned, no need to         1
      Slow pickup process (repeat)                         1
      Slow pickup process (Duplicate, but includes         1
      No obvious issue mentioned                           1
      Name: count, dtype: int64
```

Explanation

This output displays the frequency of each issue identified in the customer reviews. The most common response is **“No issue mentioned”** with **44 reviews**, meaning many customers didn’t report any problems. Other frequent issues include **“Incorrect billing” (7 times)**, **“Poor car condition” (5 times)**, and **“Unfair extra charges” (4 times)**. Several issues appear only once, including variations of **“Late delivery”**, **“Missed reservation”**, and **“Slow pickup process”**. This breakdown helps the management team focus on the **most recurring problems** that impact customer satisfaction.

[18]:
```
import matplotlib.pyplot as plt
sentiment_counts = data["Sentiment"].value_counts()
plt.figure(figsize=(6, 4))
sentiment_counts.plot(kind="bar", color=["green", "red", "gray"])
plt.title("Sentiment Distribution")
plt.xlabel("Sentiment")
plt.ylabel("Number of Reviews")
plt.xticks(rotation=0)
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()
plt.show()
```



Explanation

This code generates a bar chart showing the distribution of customer sentiments using matplotlib. It takes the count of each sentiment (Positive, Negative, Neutral) and displays them in different colors — green, red, and gray. The chart is labeled clearly with a title, axis labels, and a horizontal grid for better readability. This visual makes it easy to quickly understand how customers are feeling overall based on the collected reviews.

[25]:
```
summary_prompt = f'''
You are a customer feedback analysis assistant.

Your task is to:
1. Summarize the distribution of customer sentiments.
2. Identify the most frequently mentioned issues.
3. Provide a suggestion for improvement based on the issues and sentiment data.

Use clear and concise natural language suitable for a business report in 200 words only.

Sentiment distribution:
{data["Sentiment"].value_counts().to_string()}

Issue frequency:
{data["Issues"].value_counts().to_string()}
'''
parameters = {
    "decoding_method": "sample",
    "max_new_tokens": 300,
    "min_new_tokens": 0,
    "temperature": 0,
    "top_k": 50,
    "top_p": 1,
    "repetition_penalty": 1
}
model2 = ModelInference(
    model_id = model_id,
    params = parameters,
    credentials = credentials,
    project_id = project_id,
    space_id = space_id
    )
response = model2.generate_text(prompt=summary_prompt, guardrails=True).strip()

from IPython.display import Markdown, display
display(Markdown(response))
```

No significant improvement, no incentives provided 1

Suggestions for improvement:

Based on the analysis of customer feedback, several key areas require attention to address recurring issues and improve overall customer satisfaction.

Firstly, tackle incorrect billing and billing support inadequacy to satisfy the concerns of 45% of dissatisfied customers. This issue is of paramount importance, given its high prevalence.

Secondly, address the problem of poor car condition and unfair extra charges to address the complaints of 7% of customers. Offering improved maintenance standards and transparent charge structures significantly contributes to customer trust and trust in your services.

Thirdly, ensure better customer service by rectifying issues raised by 3% of customers. A focus on staff training to enhance interpersonal skills, empathy, and professionalism would generate positive feedback.

Lastly, provide clear communication channels for resolving issues and offer incentives to customers for their patience and loyalty. This would bolster 15% of neutral and positive sentiments and help create an overall improved environment.

By implementing these suggestions, your business is poised to enhance customer satisfaction, reduce negative sentiment, and foster long-term loyalty.

Explanation

This code sends a custom prompt to the IBM watsonx.ai model to generate a business report-style summary. The prompt includes the sentiment distribution and most common customer issues, asking the model to write a clear summary and suggest improvements. The model uses all the extracted data and returns a professional summary. In this case, it recommends addressing incorrect billing, car condition issues, and improving customer service through better staff training and communication. This helps managers take data-driven actions to boost customer satisfaction.

7. Conclusion

This project successfully demonstrates how customer feedback from car rental services can be intelligently analyzed using IBM watsonx.ai. By classifying sentiments and extracting frequently mentioned issues, we gained valuable insights into customer satisfaction and recurring service problems. The generated summary and visualizations provide a clear view of customer expectations and highlight areas for improvement. Overall, this solution offers a powerful and automated way to support decision-making, enhance service quality, and improve customer experience in the car rental industry.

8. References

IBM watsonx.ai Documentation – https://www.ibm.com/products/watsonx-ai
IBM Cloud Object Storage – https://cloud.ibm.com/docs/cloud-object-storage
IBM Prompt Lab – https://dataplatform.cloud.ibm.com/prompt-lab
Pandas Documentation – https://pandas.pydata.org/docs/
Matplotlib Documentation – https://matplotlib.org/stable/contents.html
Python Official Docs – https://docs.python.org/3/