```
import pandas as pd
import numpy as np
import nltk
import spacy
import seaborn as sns
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, accuracy_score
from textblob import TextBlob
from gensim import corpora, models

# Sample chatbot interaction data
data = {
    "user_query": [
        "Hello, how can I reset my password?",
        "I need help with my order refund",
        "Thank you, great service!",
        "Why is my payment not processing?",
        "Can you help me book a flight?",
        "This chatbot is useless!",
    ],
    "chatbot_response": [
        "Please visit the reset password page.",
        "Refunds are processed within 3-5 business days.",
        "Glad to assist you!",
        "Please check your bank details or contact support.",
        "Sure! Let me fetch flight details.",
        "I'm sorry for the inconvenience. How can I improve?",
    ]
}

# Convert to DataFrame
df = pd.DataFrame(data)

# Sentiment Analysis
def analyze_sentiment(text):
    return TextBlob(text).sentiment.polarity

df["sentiment"] = df["user_query"].apply(analyze_sentiment)

# Exploratory Data Analysis
plt.figure(figsize=(8, 4))
sns.histplot(df["sentiment"], bins=10, kde=True)
plt.title("Sentiment Distribution of User Queries")
plt.show()

# NLP Preprocessing
nlp = spacy.load("en_core_web_sm")

def preprocess_text(text):
    doc = nlp(text.lower())
    tokens = [token.lemma_ for token in doc if not token.is_stop and not token.is_punct]
    return " ".join(tokens)

df["processed_query"] = df["user_query"].apply(preprocess_text)

# Topic Modeling using LDA
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df["processed_query"])
dictionary = corpora.Dictionary([text.split() for text in df["processed_query"]])
corpus = [dictionary.doc2bow(text.split()) for text in df["processed_query"]]

lda_model = models.LdaModel(corpus, num_topics=2, id2word=dictionary, passes=10)
topics = lda_model.print_topics(num_words=3)
print("Topics Identified:", topics)

# Intent Classification (Simple Machine Learning Model)
df["intent"] = ["account_help", "order_issue", "positive_feedback", "payment_problem", "travel_booking", "negative_feedback"]
a
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(df["processed_query"])
y = df["intent"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = MultinomialNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

```
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-dad050ce26c7> in <cell line: 0>()
     11 from sklearn.metrics import classification_report, accuracy_score
     12 from textblob import TextBlob
---> 13 from gensim import corpora, models
     14 
     15 # Sample chatbot interaction data

ModuleNotFoundError: No module named 'gensim'

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

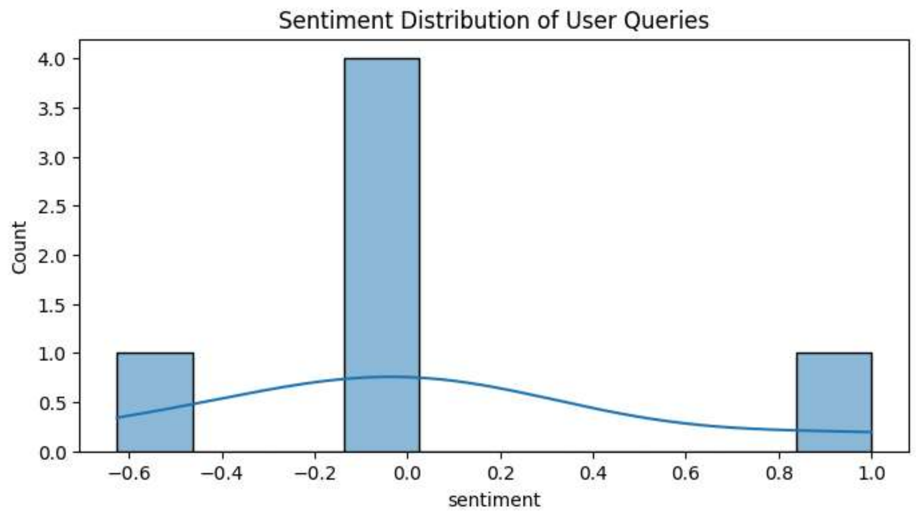
OPEN EXAMPLES

```
!pip install gensim
```

```
Collecting gensim
  Downloading gensim-4.3.3-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (8.1 kB)
Requirement already satisfied: numpy<2.0,>=1.18.5 in /usr/local/lib/python3.11/dist-packages (from gensim) (1.26.4)
Collecting scipy<1.14.0,>=1.7.0 (from gensim)
  Downloading scipy-1.13.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (60 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.6/60.6 kB 2.6 MB/s eta 0:00:00
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.11/dist-packages (from gensim) (7.1.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-packages (from smart-open>=1.8.1->gensim) (1.17.2)
Downloading gensim-4.3.3-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (26.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 26.7/26.7 MB 61.4 MB/s eta 0:00:00
Downloading scipy-1.13.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (38.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 38.6/38.6 MB 13.5 MB/s eta 0:00:00
Installing collected packages: scipy, gensim
  Attempting uninstall: scipy
    Found existing installation: scipy 1.14.1
    Uninstalling scipy-1.14.1:
      Successfully uninstalled scipy-1.14.1
Successfully installed gensim-4.3.3 scipy-1.13.1
```

```
import pandas as pd
import numpy as np
import nltk
import spacy
import seaborn as sns
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, accuracy_score
from textblob import TextBlob
from gensim import corpora, models

# Sample chatbot interaction data
data = {
    "user_query": [
        "Hello, how can I reset my password?",
        "I need help with my order refund",
        "Thank you, great service!",
        "Why is my payment not processing?",
        "Can you help me book a flight?",
        "This chatbot is useless!",
    ],
    "chatbot_response": [
        "Please visit the reset password page.",
        "Refunds are processed within 3-5 business days.",
        "Glad to assist you!",
        "Please check your bank details or contact support.",
        "Sure! Let me fetch flight details.",
        "I'm sorry for the inconvenience. How can I improve?",
    ]
}

# Convert to DataFrame
df = pd.DataFrame(data)
```

```
# Sentiment Analysis
def analyze_sentiment(text):
    return TextBlob(text).sentiment.polarity

df["sentiment"] = df["user_query"].apply(analyze_sentiment)

# Exploratory Data Analysis
plt.figure(figsize=(8, 4))
sns.histplot(df["sentiment"], bins=10, kde=True)
plt.title("Sentiment Distribution of User Queries")
plt.show()

# NLP Preprocessing
nlp = spacy.load("en_core_web_sm")

def preprocess_text(text):
    doc = nlp(text.lower())
    tokens = [token.lemma_ for token in doc if not token.is_stop and not token.is_punct]
    return " ".join(tokens)

df["processed_query"] = df["user_query"].apply(preprocess_text)

# Topic Modeling using LDA
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df["processed_query"])
dictionary = corpora.Dictionary([text.split() for text in df["processed_query"]])
corpus = [dictionary.doc2bow(text.split()) for text in df["processed_query"]]

lda_model = models.LdaModel(corpus, num_topics=2, id2word=dictionary, passes=10)
topics = lda_model.print_topics(num_words=3)
print("Topics Identified:", topics)

# Intent Classification (Simple Machine Learning Model)
df["intent"] = ["account_help", "order_issue", "positive_feedback", "payment_problem", "travel_booking", "negative_feedback"]
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(df["processed_query"])
y = df["intent"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = MultinomialNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```



```
Topics Identified: [(0, '0.125*"help" + 0.075*"order" + 0.075*"refund"'), (1, '0.114*"thank" + 0.114*"service" + 0.114*"great"')]
Classification Report:
                  precision    recall  f1-score   support

     account_help       0.00      0.00      0.00       1.0
negative_feedback       0.00      0.00      0.00       0.0
      order_issue       0.00      0.00      0.00       1.0
   travel_booking       0.00      0.00      0.00       0.0

         accuracy                           0.00       2.0
        macro avg       0.00      0.00      0.00       2.0
     weighted avg       0.00      0.00      0.00       2.0
```