Sentiment Analysis Notebook

What is Sentiment Analysis?

Sentiment analysis, also known as opinion mining, is a Natural Language Processing (NLP) technique used to determine whether a piece of text expresses a positive, negative, or neutral sentiment. It is widely used in areas like customer feedback analysis, brand monitoring, and social media analysis. In this notebook, sentiment analysis is performed on tweets directed at various airlines to classify public opinion into sentiments.

Cell 1: Load and Display Data

The pandas library is used to read the dataset 'Tweets.csv' containing tweets about various airlines. The dataset is stored in a DataFrame called `data`, which is then displayed. This provides an overview of the data including columns like 'text' (the tweet), 'airline_sentiment' (the label), and other metadata.

Code:

```
import pandas
data = pandas.read_csv("datasets\\Tweets.csv")
data
```

Cell 2: Check for Missing Values

This cell checks for missing (null) values in each column using `data.isnull().sum()`. It helps identify which fields are incomplete. Columns like 'text' and 'airline_sentiment' are complete, which is crucial for training a sentiment model.

Code:

```
data.isnull().sum()
# Text and airline sentiments, no empties
```

Cell 3: Check Data Types

This cell checks the data types of all columns using `data.dtypes`. It ensures that each field has the correct data type for analysis and model training. Most text-based fields are of type `object`, while numeric fields like 'tweet_id' and 'retweet_count' are `int64` or `float64`.

Code:

```
data.dtypes
# the data types are correct
```

Cell 4: Code Explanation

This cell runs the following code:

```
# split to 30% for testing 70% for training
```

```
from sklearn import model_selection
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(data['text'],
data['airline_sentiment'],
                                                                    test_size = 0.3,
                                                                    random_state=42)
```

Explanation:
- This cell imports necessary modules for feature extraction and modeling.

Cell 5: Code Explanation

This cell runs the following code:

```
# remove stop words  from X_train or X_test, and into a numerical representation suitable
for machine learning tasks using scikit-learn's CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer #importing the
CountVectorizer class, which helps convert text documents into a numerical representation
based on word frequencies.
vectorizer = CountVectorizer(lowercase = True, stop_words = 'english') # lowercase = True:
Converts all text to lowercase before processing (recommended for most NLP tasks).
# stop_words = 'english': Instructs the vectorizer to remove stop words from the analysis
(using the built-in English stop words list)

X_train_new = vectorizer.fit_transform(X_train)
# .fit(X_train): Fits the vectorizer to the training data (X_train). This analyzes the vocabulary
and creates a document-term matrix
# .transform(X_train): Transforms the training data (X_train) using the fitted vectorizer.
This converts each text document into a numerical feature vector representing word
frequencies.
X_train_new
```

Explanation:
- This cell imports necessary modules for feature extraction and modeling.

Cell 6: Code Explanation

This cell runs the following code:

```
# creates and trains a Logistic Regression model for classification
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import LinearSVC

model = LogisticRegression()
model.fit(X_train_new, Y_train)
```

Explanation:
- This cell imports necessary modules for feature extraction and modeling.

Cell 7: Code Explanation

This cell runs the following code:

```
# test the model, first vectorize
X_test_new = vectorizer.transform(X_test)
predictions = model.predict(X_test_new)
print(data.groupby('airline_sentiment').size())
```

Explanation:
- Uses the trained model to predict sentiment labels for the test data.

Cell 8: Code Explanation

This cell runs the following code:

```
#Test accuracy
from sklearn.metrics import accuracy_score
print('Accuracy ', accuracy_score(Y_test, predictions))
```

Explanation:
- This cell imports necessary modules for feature extraction and modeling.

Cell 9: Code Explanation

This cell runs the following code:

```
from sklearn.metrics import classification_report # it show the classification of the metrics on per-class basis
print(classification_report(Y_test, predictions))
```

Explanation:
- This cell imports necessary modules for feature extraction and modeling.

Cell 10: Code Explanation

This cell runs the following code:

```
# pull a database, csv, from api
newdata = pandas.read_csv("datasets\sample.csv")
newdata
```

Explanation:

- This code performs a specific operation required in the pipeline (data inspection, cleaning, transformation, modeling, or evaluation).

Cell 11: Code Explanation

This cell runs the following code:

```
# Use the model to predict above clients sentiments
newdata_vectorized = vectorizer.transform(newdata['Text'])
outcome = model.predict(newdata_vectorized)
print('airline_sentiment',  outcome)
#Improve the model to atleast 85%
outcomedf = pandas.DataFrame(outcome,  columns=['Sentiment'])
outcomedf
```

Explanation:
- Uses the trained model to predict sentiment labels for the test data.

Cell 12: Code Explanation

This cell runs the following code:

```
results = newdata.merge(outcomedf, left_index=True,right_index=True)
results
```

Explanation:
- This code performs a specific operation required in the pipeline (data inspection, cleaning, transformation, modeling, or evaluation).

Cell 13: Code Explanation

This cell runs the following code:

```
import matplotlib.pyplot as plt

results.groupby('Sentiment').size().plot(kind='pie', autopct = '%1.1f%%')
```

Explanation:
- This code performs a specific operation required in the pipeline (data inspection, cleaning, transformation, modeling, or evaluation).

Cell 14: Code Explanation

This cell runs the following code:

```
negatives = results[results['Sentiment'] == 'negative']
negatives
```

```
positives = results[results['Sentiment'] == 'positive']
positives

neutral = results[results['Sentiment'] == 'neutral']
neutral
```

Explanation:
- This code performs a specific operation required in the pipeline (data inspection, cleaning, transformation, modeling, or evaluation).

Cell 15: Code Explanation

This cell runs the following code:

```
# create a CSV with the negatives only
# negatives.to_excel("sample_data/negatives.xls")
# positives.to_excel("sample_data/positives.xls")
# neutral.to_excel("sample_data/neutral.xls")
```

Explanation:
- This code performs a specific operation required in the pipeline (data inspection, cleaning, transformation, modeling, or evaluation).