```
# Step 1: Upload Dataset (Only for Google Colab)
from google.colab import files
uploaded = files.upload()
# Step 2: Load the CSV
import pandas as pd
df = pd.read_csv("/content/drive/MyDrive/elonmusk.csv")
 # Replace with actual filename if different
df.head()
# Step 3: Basic Info
print("Shape:", df.shape)
print("Columns:", df.columns.tolist())
df.info()
print(df.describe())
# Step 4: Missing values and duplicates
print("Missing values:\n", df.isnull().sum())
print("Duplicate rows:", df.duplicated().sum())
# Step 5: Visualizations
import seaborn as sns
import matplotlib.pyplot as plt
df['tweet_length'] = df['tweet'].astype(str).apply(len)
# Tweet length distribution
sns.histplot(df['tweet_length'], kde=True, bins=30, color='teal')
plt.title('Distribution of Tweet Lengths')
plt.xlabel('Tweet Length')
plt.ylabel('Frequency')
plt.show()
# Tweets per year
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# Step 1: Convert 'date' column to datetime format
df['date'] = pd.to_datetime(df['date'], errors='coerce')
# Step 2: Extract year from 'date'
df['year'] = df['date'].dt.year
# Step 3: Plot number of tweets per year
sns.countplot(data=df, x='year', palette='coolwarm')
plt.title('Number of Tweets per Year')
plt.xlabel('Year')
plt.ylabel('Tweet Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
# Top usernames
top_users = df['username'].value_counts().head(10)
top_users.plot(kind='bar', color='orange')
plt.title('Top 10 Users by Number of Tweets')
plt.xlabel('Username')
plt.ylabel('Tweet Count')
plt.show()
# Step 6: Sentiment Analysis using TextBlob
!pip install -q textblob
from textblob import TextBlob
def get_sentiment(text):
 polarity = TextBlob(str(text)).sentiment.polarity
 if polarity > 0:
  return 'positive'
 elif polarity == 0:
  return 'neutral'
 else:
  return 'negative'
df['sentiment'] = df['tweet'].apply(get_sentiment)
print(df[['tweet', 'sentiment']].head())
# Step 7: Train-Test Split
from sklearn.model_selection import train_test_split
```

```
X = df['tweet'].astype(str)
y = df['sentiment']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
stratify=y, random_state=42)
# Step 8: TF-IDF + Logistic Regression Pipeline
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,confusion_matrix
model = Pipeline([
 ('tfidf', TfidfVectorizer(max_features=5000, stop_words='english')),
 ('clf', LogisticRegression(max_iter=1000))
])
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
# Step 9: Evaluation
print("✅ Accuracy:", accuracy_score(y_test, y_pred))
print("📊 Classification Report:\n", classification_report(y_test, y_pred))
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
# Step 10: Test on new tweets
new_tweets = [
 "Tesla's new update is amazing!",
 "I'm not happy with Twitter's new algorithm.",
 "SpaceX launch was a huge success!",
 "This is just disappointing."
]
predictions = model.predict(new_tweets)
for tweet, sentiment in zip(new_tweets, predictions):
 print(f"Tweet: {tweet}\nPredicted Sentiment: {sentiment}\n")
# Step 11: Gradio Web App
!pip install -q gradio
import gradio as gr
def predict_sentiment(tweet):
 return model.predict([tweet])[0]
iface = gr.Interface(
 fn=predict_sentiment,
 inputs=gr.Textbox(lines=3, placeholder="Enter a tweet..."),
 outputs="text",
 title="Elon Musk Tweet Sentiment Analyzer",
 description="Enter a tweet related to Elon Musk to classify sentiment as Positive, Neutral, or Negative."
)
iface.launch()
```

Choose Files elonmusk.csv (1).zip

- **elonmusk.csv (1).zip**(application/x-zip-compressed) - 977036 bytes, last modified: 5/19/2025 - 100% done

```
Saving elonmusk.csv (1).zip to elonmusk.csv (1).zip
Shape: (9286, 34)
Columns: ['id', 'conversation_id', 'created_at', 'date', 'time', 'timezone', 'user_id', 'username', 'name', 'place
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9286 entries, 0 to 9285
Data columns (total 34 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               9286 non-null   int64
 1   conversation_id  9286 non-null   int64
 2   created_at       9286 non-null   int64
 3   date             9286 non-null   object
 4   time             9286 non-null   object
 5   timezone         9286 non-null   object
 6   user_id          9286 non-null   int64
 7   username         9286 non-null   object
 8   name             9286 non-null   object
 9   place            0 non-null      float64
 10  tweet            9286 non-null   object
 11  mentions         9286 non-null   object
 12  urls             9286 non-null   object
 13  photos           9286 non-null   object
 14  replies_count    9286 non-null   int64
 15  retweets_count   9286 non-null   int64
 16  likes_count      9286 non-null   int64
 17  hashtags         9286 non-null   object
 18  cashtags         9286 non-null   object
 19  link             9286 non-null   object
 20  retweet          9286 non-null   bool
 21  quote_url        355 non-null    object
 22  video            9286 non-null   int64
 23  near             0 non-null      float64
 24  geo              0 non-null      float64
 25  source           0 non-null      float64
 26  user_rt_id       0 non-null      float64
 27  user_rt          0 non-null      float64
 28  retweet_id       0 non-null      float64
 29  reply_to         9286 non-null   object
 30  retweet_date     0 non-null      float64
 31  translate        0 non-null      float64
 32  trans_src        0 non-null      float64
 33  trans_dest       0 non-null      float64
dtypes: bool(1), float64(11), int64(8), object(14)
memory usage: 2.3+ MB
                 id  conversation_id    created_at     user_id  place  \
count  9.286000e+03     9.286000e+03  9.286000e+03      9286.0    0.0
mean   1.055061e+18     1.052389e+18  1.540381e+12  44196397.0    NaN
std    1.695110e+17     1.740050e+17  4.041457e+10         0.0    NaN
min    5.610022e+17     1.659576e+09  1.422588e+12  44196397.0    NaN
25%    9.667040e+17     9.610311e+17  1.519315e+12  44196397.0    NaN
50%    1.089662e+18     1.088672e+18  1.548631e+12  44196397.0    NaN
75%    1.187852e+18     1.187320e+18  1.572041e+12  44196397.0    NaN
max    1.282940e+18     1.282933e+18  1.594712e+12  44196397.0    NaN

       replies_count  retweets_count   likes_count        video  near  geo  \
count    9286.000000     9286.000000  9.286000e+03  9286.000000   0.0  0.0
mean      512.958432     2282.844066  1.873480e+04     0.007538   NaN  NaN
std      1720.320306    10546.775719  5.929579e+04     0.086500   NaN  NaN
min         0.000000        0.000000  2.100000e+01     0.000000   NaN  NaN
25%        43.250000       43.000000  9.430000e+02     0.000000   NaN  NaN
50%       103.000000      129.000000  2.341500e+03     0.000000   NaN  NaN
75%       371.000000      955.750000  1.159850e+04     0.000000   NaN  NaN
max     49529.000000   384289.000000  1.682551e+06     1.000000   NaN  NaN

       source  user_rt_id  user_rt  retweet_id  retweet_date  translate  \
count     0.0         0.0      0.0         0.0           0.0        0.0
mean      NaN         NaN      NaN         NaN           NaN        NaN
std       NaN         NaN      NaN         NaN           NaN        NaN
min       NaN         NaN      NaN         NaN           NaN        NaN
```