```
!pip install -r Full_test_requirements.txt
```

```
Requirement already satisfied: watchdog<7,>=2.1.5 in /usr/local/lib/python3.11/dist-packages (from streamlit->-r Full_test_requiremen
Requirement already satisfied: gitpython!=3.1.19,<4,>=3.0.7 in /usr/local/lib/python3.11/dist-packages (from streamlit->-r Full_test_
Requirement already satisfied: pydeck<1,>=0.8.0b4 in /usr/local/lib/python3.11/dist-packages (from streamlit->-r Full_test_requiremen
Requirement already satisfied: tornado<7,>=6.0.3 in /usr/local/lib/python3.11/dist-packages (from streamlit->-r Full_test_requirement
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.11/dist-packages (from Wikipedia->-r Full_test_requirements.t
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk->-r Full_test_requirements.txt (line 18))
Requirement already satisfied: torchvision>=0.5 in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.t
Requirement already satisfied: opencv-python-headless in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirem
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (line 19
Requirement already satisfied: scikit-image in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (
Requirement already satisfied: python-bidi in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (l
Requirement already satisfied: Shapely in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (line
Requirement already satisfied: pyclipper in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (lin
Requirement already satisfied: ninja in /usr/local/lib/python3.11/dist-packages (from easyocr->-r Full_test_requirements.txt (line 19
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langch
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-com
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-commun
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-co
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-com
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-co
Requirement already satisfied: jsonschema>=3.0 in /usr/local/lib/python3.11/dist-packages (from altair<6,>=4.0->streamlit->-r Full_te
Requirement already satisfied: narwhals>=1.14.2 in /usr/local/lib/python3.11/dist-packages (from altair<6,>=4.0->streamlit->-r Full_t
Requirement already satisfied: marshmallow<4.0.0,>=3.18.0 in /usr/local/lib/python3.11/dist-packages (from dataclasses-json<0.7,>=0.5
Requirement already satisfied: typing-inspect<1,>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from dataclasses-json<0.7,>=0.5.7
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.11/dist-packages (from gitpython!=3.1.19,<4,>=3.0.7->streaml
Requirement already satisfied: jsonpatch<2.0,>=1.33 in /usr/local/lib/python3.11/dist-packages (from langchain-core<1.0.0,>=0.3.59->l
Requirement already satisfied: httpx<1,>=0.23.0 in /usr/local/lib/python3.11/dist-packages (from langsmith<0.4,>=0.1.17->langchain->-
Requirement already satisfied: orjson<4.0.0,>=3.9.14 in /usr/local/lib/python3.11/dist-packages (from langsmith<0.4,>=0.1.17->langcha
Requirement already satisfied: requests-toolbelt<2.0.0,>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from langsmith<0.4,>=0.1.1
Requirement already satisfied: zstandard<0.24.0,>=0.23.0 in /usr/local/lib/python3.11/dist-packages (from langsmith<0.4,>=0.1.17->lan
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.7.4->langch
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.7.4->langcha
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<3.0.0,>=2.7.4->lang
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from jinja2->torch==2.6.0->-r Full_test_re
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib->-r Full_te
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface_hub->-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface_hub->-r Full_test_
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface_hub->-r Full_
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface_hub->-r Full_
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (from sentence-transformers>=2.6.0->langchain-
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.11/dist-packages (from SQLAlchemy<3,>=1.4->langchain->-r Full_te
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4->Wikipedia->-r Full_test_
Requirement already satisfied: imageio!=2.35.0,>=2.33 in /usr/local/lib/python3.11/dist-packages (from scikit-image->easyocr->-r Full_
Requirement already satisfied: tifffile>=2022.8.12 in /usr/local/lib/python3.11/dist-packages (from scikit-image->easyocr->-r Full_te
Requirement already satisfied: lazy-loader>=0.4 in /usr/local/lib/python3.11/dist-packages (from scikit-image->easyocr->-r Full_test_
Requirement already satisfied: smmap<6,>=3.0.1 in /usr/local/lib/python3.11/dist-packages (from gitdb<5,>=4.0.1->gitpython!=3.1.19,<4
Requirement already satisfied: anyio in /usr/local/lib/python3.11/dist-packages (from httpx<1,>=0.23.0->langsmith<0.4,>=0.1.17->langc
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-packages (from httpx<1,>=0.23.0->langsmith<0.4,>=0.1.1
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.11/dist-packages (from httpcore==1.*->httpx<1,>=0.23.0->langsmith<
Requirement already satisfied: jsonpointer>=1.9 in /usr/local/lib/python3.11/dist-packages (from jsonpatch<2.0,>=1.33->langchain-core
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=3.0-
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=3.0->altair<6,>=4.0->
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=3.0->altair<6,>=4.0->strea
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from typing-inspect<1,>=0.4.0->data
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn->sentence-transform
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-packages (from anyio->httpx<1,>=0.23.0->langsmith<0.4,>
```

Importing Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
import faiss
import os
import wikipedia
import fitz
import nltk
import shutil
import re


from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig
from nltk.translate.bleu_score import sentence_bleu, SmoothingFunction
from sentence_transformers import SentenceTransformer, util
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from langchain_community.llms import HuggingFaceEndpoint
from sklearn.metrics.pairwise import cosine_similarity
from langchain_huggingface import HuggingFaceEndpoint
from google.colab import userdata, files
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords


import os
import cv2
import easyocr
```

Loading pdf Data

```
folder_name = 'pdfs'
if not os.path.exists(folder_name):
    os.makedirs(folder_name)

uploaded = files.upload()
for filename in uploaded.keys():
    shutil.move(filename, os.path.join(folder_name, filename))
```

Choose Files accountingi…bility (1).pdf

- **accountinginsights.orgwhat-does-churn-in-business-mean-for-your-revenue-and-profitability (1).pdf**(application/pdf) - 454680 bytes, last modified: 4/28/2025 - 100% done

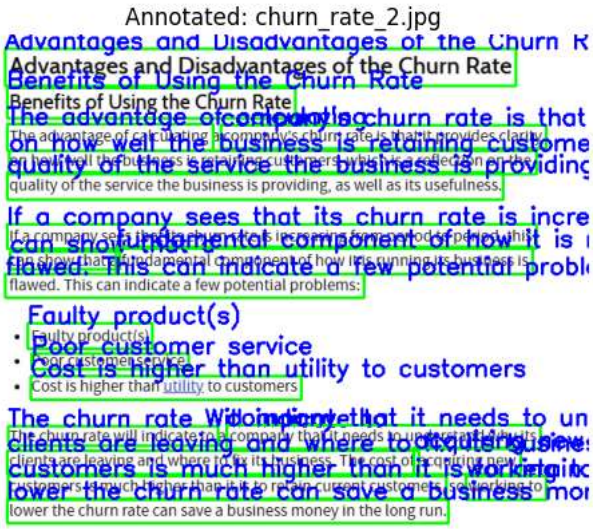
```
Saving accountinginsights.orgwhat-does-churn-in-business-mean-for-your-revenue-and-profitability (1).pdf to accountinginsights.orgwhat-d
```

Loading Image data and retriving Text using OCR

```
image_folder = '/content/image_folder'  # Update this path to your folder containing images
threshold = 0.25
max_words_per_chunk = 100  # Adjust as needed

reader = easyocr.Reader(['en'], gpu=False)

# === Function to Chunk Text ===
def chunk_text(text, max_words=100):
    words = text.split()
    return [' '.join(words[i:i+max_words]) for i in range(0, len(words), max_words)]

all_text_chunks_img = []

for image_file in os.listdir(image_folder):
    if not image_file.lower().endswith(('.jpg', '.jpeg', '.png', '.bmp')):
        continue

    image_path = os.path.join(image_folder, image_file)
    img = cv2.imread(image_path)

    text_detect = reader.readtext(img)
    detected_text = []

    print(f"\nProcessing image: {image_file}")
    print("Detected text blocks:")

    for t in text_detect:
        bbox, text, score = t
        if score > threshold:
            detected_text.append(text)
            print(f" - {text} (score: {score:.2f})")
            # Optional visualization
            bbox = [tuple(map(int, point)) for point in bbox]
            cv2.rectangle(img, bbox[0], bbox[2], (0, 255, 0), 2)
            cv2.putText(img, text, bbox[0], cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255, 0, 0), 2)

    full_text = ' '.join(detected_text)
    chunks_img = chunk_text(full_text, max_words=max_words_per_chunk)

    for chunk in chunks_img:
        all_text_chunks_img.append({
            'image_file': image_file,
```

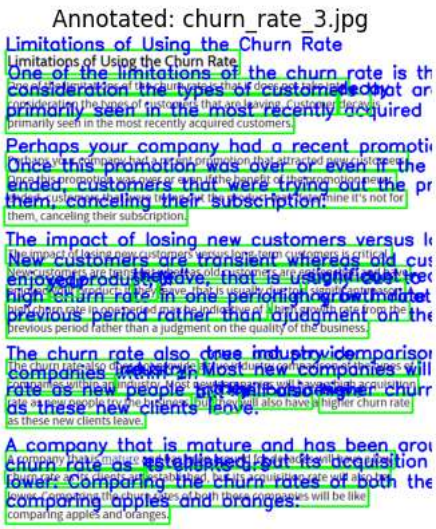
```
                'chunk_text': chunk
            })

        plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
        plt.title(f"Annotated: {image_file}")
        plt.axis('off')
        plt.show()

    # === Final Output: List of Chunks ===
    print("\nAll OCR Chunks (ready for RAG):\n")
    for item in all_text_chunks_img:
        print(f"[{item['image_file']}] {item['chunk_text']}\n")
```

```
WARNING:easyocr.easyocr:Using CPU. Note: This module is much faster with a GPU.

Processing image: churn_rate_2.jpg
Detected text blocks:
 - Advantages and Disadvantages of the Churn Rate (score: 0.97)
 - Benefits of Using the Churn Rate (score: 0.68)
 - The advantage of calculating (score: 0.98)
 - company's churn rate is that it provides clarity (score: 0.76)
 - on how well the business is retaining customers, which is a reflection on the (score: 0.73)
 - quality of the service the business is providing; as well as its usefulness (score: 0.70)
 - If a company sees that its churn rate is increasing from period to period, this (score: 0.64)
 - can show that a (score: 0.97)
 - fundamental component of how it is running its business is (score: 0.81)
 - flawed: This can indicate a few potential problems: (score: 0.66)
 - Faulty product(s) (score: 0.99)
 - Poor customer service (score: 0.81)
 - Cost is higher than utility to customers (score: 0.65)
 - The churn rate Will indicate to (score: 0.86)
 - company that it needs to understand why its (score: 0.89)
 - clients are leaving and where to fix its business The cost of (score: 0.85)
 - acquiring new (score: 0.81)
 - customers is much higher than it is to retain current customers, (score: 0.53)
 - working to (score: 1.00)
 - lower the churn rate can save a business money in the long run: (score: 0.71)
```

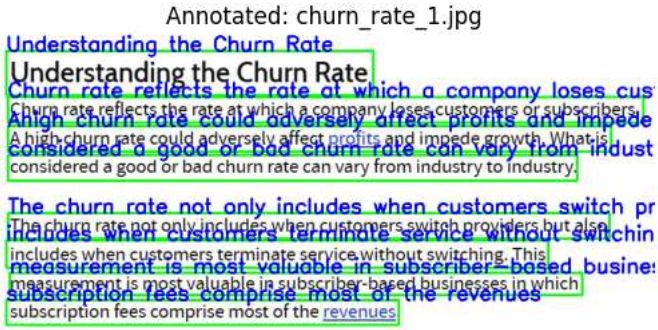



```
Processing image: churn_rate_3.jpg
Detected text blocks:
 - Limitations of Using the Churn Rate (score: 0.85)
 - One of the limitations of the churn rate is that it doesnot take into (score: 0.69)
 - consideration the types of customers that are leaving: Customer (score: 0.82)
 - primarily seen in the most recently acquired customers. (score: 0.74)
 - Perhaps your company had a recent promotion that attracted new customers (score: 0.77)
 - Once this promotion was over or even if the benefit of the promotion never (score: 0.63)
 - ended, customers that were trying out the product may determine it's not for (score: 0.74)
 - them, canceling their subscription: (score: 0.87)
 - The impact of losing new customers versus long-term customers is critical: (score: 0.84)
 - New customers are transient whereas old customers are entrenched and have (score: 0.88)
 - enjoyed (score: 1.00)
 - product; if (score: 0.78)
 - leave, that is usually due to (score: 0.75)
 - significant reason: (score: 0.74)
 - high churn rate in one period may be indicative of . (score: 0.57)
 - high growth rate from the (score: 0.97)
 - previous period rather than ajudgment on the quality of the business: (score: 0.63)
 - The churn rate also does not provide (score: 0.81)
 - true industry comparison of the types of (score: 0.86)
 - companies within an (score: 0.92)
 - industry: Most new companies will have a high acquisition (score: 0.80)
 - rate as new people try the business_ (score: 0.72)
 - but- (score: 0.99)
 - will also have (score: 0.79)
 - higher churn rate (score: 1.00)
 - as these new clients leave. (score: 0.77)
 - A company that is mature and has been around for decades will have a low (score: 0.63)
 - churn rate as its clients are (score: 0.64)
 - established, but its acquisition rate will also be (score: 0.80)
 - lower: Comparing the churn rates of both these companies will be like (score: 0.73)
 - comparing apples and oranges: (score: 0.62)
 - decay (score: 1.00)
 - they " (score: 0.60)
 - your (score: 1.00)
```

```
your (score: 1.00)
- they - (score: 0.47)
```




```
Processing image: churn_rate_1.jpg
Detected text blocks:
 - Understanding the Churn Rate (score: 0.77)
 - Churn rate reflects the rate at which a company loses customers or subscribers: (score: 0.74)
 - Ahigh churn rate could adversely affect profits and impede growth: What is (score: 0.80)
 - considered a good or bad churn rate can vary from industry to industry: (score: 0.85)
 - The churn rate not only includes when customers switch providers but also (score: 0.82)
 - includes when customers terminate service without switching: This (score: 0.70)
 - measurement is most valuable in subscriber-based businesses in which (score: 0.83)
 - subscription fees comprise most of the revenues (score: 0.70)
```




```
All OCR Chunks (ready for RAG):

[churn_rate_2.jpg] Advantages and Disadvantages of the Churn Rate Benefits of Using the Churn Rate The advantage of calculating compa

[churn_rate_2.jpg] customers The churn rate Will indicate to company that it needs to understand why its clients are leaving and wher

[churn_rate_3.jpg] Limitations of Using the Churn Rate One of the limitations of the churn rate is that it doesnot take into consider

[churn_rate_3.jpg] have enjoyed product; if leave, that is usually due to significant reason: high churn rate in one period may be in

[churn_rate_3.jpg] rate as its clients are established, but its acquisition rate will also be lower: Comparing the churn rates of bot

[churn_rate_1.jpg] Understanding the Churn Rate Churn rate reflects the rate at which a company loses customers or subscribers: Ahigh
```

Loading CSV data

```
from google.colab import drive
drive.mount('/content/drive')
file_path = '/content/drive/MyDrive/customer_churn.csv'
df = pd.read_csv(file_path)
df.head()
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	Dev
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	

5 rows × 21 columns

Data Cleaning

```
df.replace(r'^\s*$', np.nan, regex=True, inplace=True)
df.columns
```

```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

```
df.isnull().sum()
```

	0
customerID	0
gender	0
SeniorCitizen	0
Partner	0
Dependents	0
tenure	0
PhoneService	0
MultipleLines	0
InternetService	0
OnlineSecurity	0
OnlineBackup	0
DeviceProtection	0
TechSupport	0
StreamingTV	0
StreamingMovies	0
Contract	0
PaperlessBilling	0
PaymentMethod	0
MonthlyCharges	0
TotalCharges	11
Churn	0

dtype: int64

```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df.fillna(df['TotalCharges'].mean(), inplace=True)
```

```
df=df[['customerID','gender','SeniorCitizen','Partner','tenure','InternetService','OnlineSecurity','MonthlyCharges', 'TotalCharges','Contrac
df.head(2)
```

	customerID	gender	SeniorCitizen	Partner	tenure	InternetService	OnlineSecurity	MonthlyCharges	TotalCharges	Contract	Churn
0	7590-VHVEG	Female	0	Yes	1	DSL	No	29.85	29.85	Month-to-month	No
1	5575-GNVDE	Male	0	No	34	DSL	Yes	56.95	1889.50	One year	No

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.isnull().sum()
```

	0
customerID	0
gender	0
SeniorCitizen	0
Partner	0
tenure	0
InternetService	0
OnlineSecurity	0
MonthlyCharges	0
TotalCharges	0
Contract	0
Churn	0

dtype: int64

```
df = df[(df['tenure'] > 0)]
df.shape
```

```
(7032, 11)
```

```
df.groupby(['Churn', 'gender','Contract']).size()
```

			0
Churn	gender	Contract	
No	Female	Month-to-month	1083
		One year	643
		Two year	818
	Male	Month-to-month	1137
		One year	663
		Two year	819
Yes	Female	Month-to-month	842
		One year	75
		Two year	22
	Male	Month-to-month	813
		One year	91
		Two year	26

dtype: int64

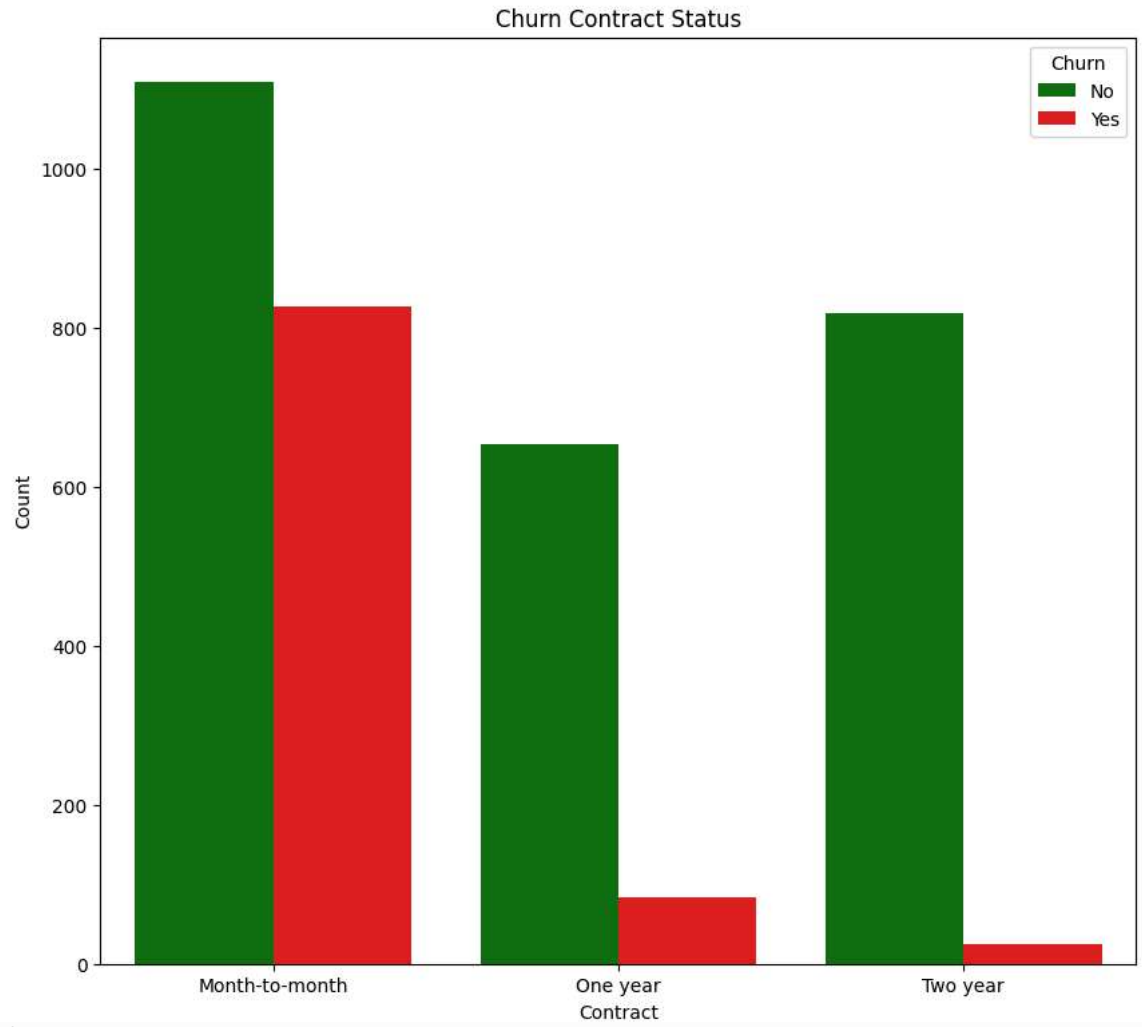
```
df5 = df[['gender','Contract','Churn']]
df_churn_count = df5.groupby(['Churn', 'gender','Contract']).size().reset_index(name='Count')
df_churn_count
```

	Churn	gender	Contract	Count
0	No	Female	Month-to-month	1083
1	No	Female	One year	643
2	No	Female	Two year	818
3	No	Male	Month-to-month	1137
4	No	Male	One year	663
5	No	Male	Two year	819
6	Yes	Female	Month-to-month	842
7	Yes	Female	One year	75
8	Yes	Female	Two year	22
9	Yes	Male	Month-to-month	813
10	Yes	Male	One year	91
11	Yes	Male	Two year	26

Next steps: Generate code with df_churn_count | View recommended plots | New interactive sheet

```
plt.figure(figsize=(10,9))
sb.barplot(data=df_churn_count, x='Contract', y='Count', hue='Churn', ci=None,
           palette={'Yes': 'red', 'No': 'green'}, dodge=True)

plt.xlabel("Contract")
plt.ylabel("Count")
plt.title("Churn Contract Status")
plt.legend(title="Churn")

plt.show()
```

```
<ipython-input-16-799aa5d9abd7>:2: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sb.barplot(data=df_churn_count, x='Contract', y='Count', hue='Churn', ci=None,
```




```
df.TotalCharges.describe()
```

	TotalCharges
count	7032.000000
mean	2283.300441
std	2266.771362
min	18.800000
25%	401.450000
50%	1397.475000
75%	3794.737500
max	8684.800000

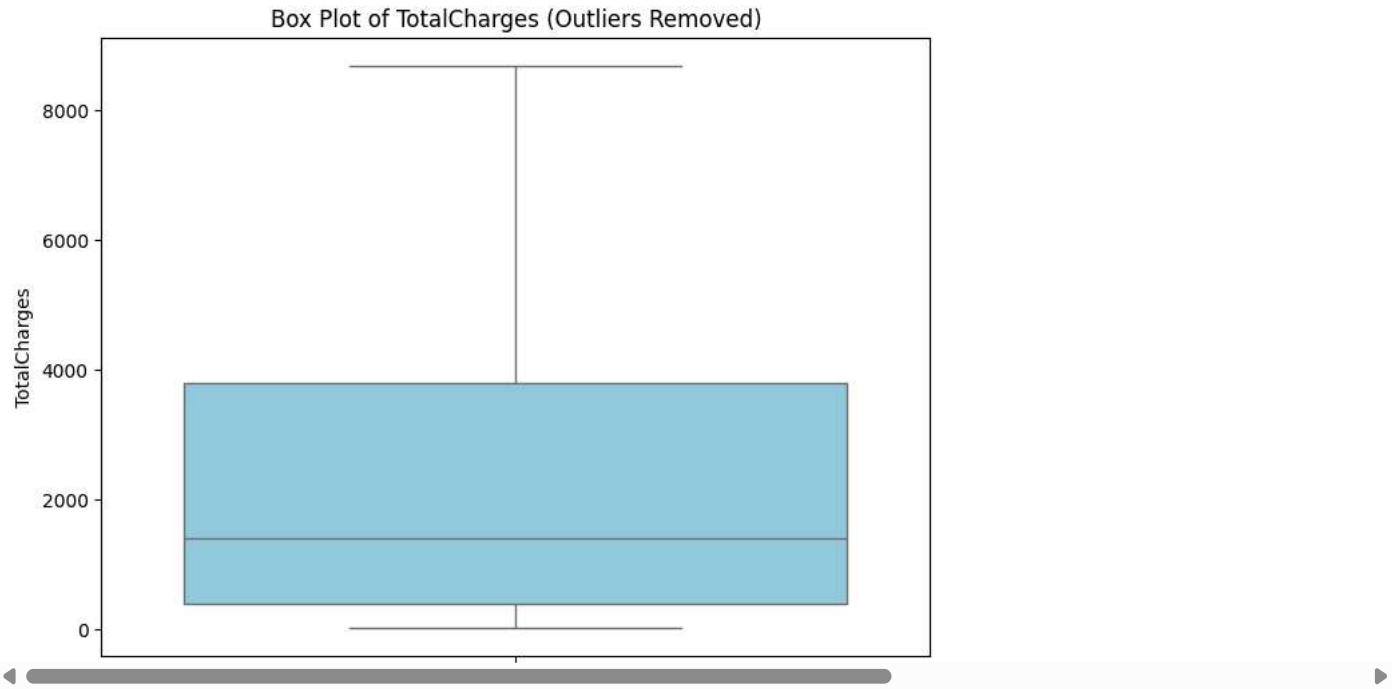
dtype: float64

```
Q1 = df['TotalCharges'].quantile(0.25)
Q3 = df['TotalCharges'].quantile(0.75)
IQR = Q3 - Q1  #Interquartile Range

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

#Remove_outliers
df_cleaned = df[(df['TotalCharges'] >= lower_bound) & (df['TotalCharges'] <= upper_bound)]
```

```
print(f"Original size: {df.shape[0]}, After outlier removal: {df_cleaned.shape[0]}")
```

```
Original size: 7032, After outlier removal: 7032
```

```
plt.figure(figsize=(8, 6))
sb.boxplot(data=df_cleaned, y='TotalCharges', color='skyblue')
plt.title("Box Plot of TotalCharges (Outliers Removed)")
plt.ylabel("TotalCharges")
plt.show()
```




```
df_MonthlyCharges=df[['Contract', 'InternetService' , 'MonthlyCharges']]
mean_monthly_charges = df_MonthlyCharges.groupby('Contract')['MonthlyCharges'].mean()
print(mean_monthly_charges)
```

```
Contract
Month-to-month    66.398490
One year          65.079416
Two year          60.872374
Name: MonthlyCharges, dtype: float64
```

```
ISP_mean_monthly_charges = df_MonthlyCharges.groupby('InternetService')['MonthlyCharges'].mean()
print(ISP_mean_monthly_charges)
```

```
InternetService
DSL            58.088017
Fiber optic    91.500129
No             21.076283
Name: MonthlyCharges, dtype: float64
```

```
columns_df = ", ".join(df.columns)
columns_df
```

```
'customerID, gender, SeniorCitizen, Partner, tenure, InternetService, OnlineSecurity, MonthlyCharges, TotalCharges, Contract, Churn'
```

```
df["SeniorCitizen"]= df["SeniorCitizen"].map({0: "No", 1: "Yes"})
df.head()
```

	customerID	gender	SeniorCitizen	Partner	tenure	InternetService	OnlineSecurity	MonthlyCharges	TotalCharges	Contract	Churn
0	7590-VHVEG	Female	No	Yes	1	DSL	No	29.85	29.85	Month-to-month	No
1	5575-GNVDE	Male	No	No	34	DSL	Yes	56.95	1889.50	One year	No
2	3668-QPYBK	Male	No	No	2	DSL	Yes	53.85	108.15	Month-to-month	Yes
3	7795-CFOCW	Male	No	No	45	DSL	Yes	42.30	1840.75	One year	No
4	9237-	Female	No	No	2	Fiber optic	No	70.70	151.65	Month-to-	Yes

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape
```

```
(7032, 11)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 7032 entries, 0 to 7042
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   customerID       7032 non-null   object 
 1   gender           7032 non-null   object 
 2   SeniorCitizen    7032 non-null   object 
 3   Partner          7032 non-null   object 
 4   tenure           7032 non-null   int64  
 5   InternetService  7032 non-null   object 
 6   OnlineSecurity   7032 non-null   object 
 7   MonthlyCharges   7032 non-null   float64
 8   TotalCharges     7032 non-null   float64
 9   Contract         7032 non-null   object 
 10  Churn            7032 non-null   object 
dtypes: float64(2), int64(1), object(8)
memory usage: 659.2+ KB
```

```
df['tenure'] = df['tenure'].astype('int32')
df['MonthlyCharges'] = df['MonthlyCharges'].astype('float32')
df['TotalCharges'] = df['TotalCharges'].astype('float32')
```

Creating Text csv chunks

```
grouped = df.groupby("customerID")
chunks_csv = []
metadata = []

for name, group in grouped:

    text_chunk = f"customerID: {name}\n"
    for _, row in group.iterrows():
        entry = f"  - customerID: {row['customerID']}, gender: {row['gender']}, SeniorCitizen: {row['SeniorCitizen']}, Partner: {row['Partne
        text_chunk += entry + "\n"

    chunks_csv.append(text_chunk)
    metadata.append({"group": name})
```

Loading Wikepedia and creating chunks

```
wiki_topics = ["Churn rate"]
wiki_chunks = []
wiki_metadata = []

for topic in wiki_topics:
    try:
        content = wikipedia.page(topic).content
        chunks = [content[i:i+512] for i in range(0, len(content), 512)]
        wiki_chunks.extend(chunks)
```

```
        wiki_metadata.extend([{"source": "wikipedia", "topic": topic}] * len(chunks))
    except wikipedia.exceptions.DisambiguationError as e:
        print(f"Disambiguation required for: {topic}, options: {e.options}")
    except wikipedia.exceptions.PageError:
        print(f"Page not found: {topic}")
```

Creating PDF chunks

```
pdf_folder = "pdfs"
pdf_chunks = []
pdf_metadata = []

for file_name in os.listdir(pdf_folder):
    if file_name.endswith(".pdf"):
        file_path = os.path.join(pdf_folder, file_name)
        doc = fitz.open(file_path)
        for page in doc:
            text = page.get_text()
            chunks = [text[i:i+512] for i in range(0, len(text), 512)]
            pdf_chunks.extend(chunks)
            pdf_metadata.extend([{"source": "pdf", "file": file_name}] * len(chunks))
```

Standardizing all the chunks together

```
standardized_chunks = []


for i, text in enumerate(pdf_chunks):
    standardized_chunks.append({
        'chunk_text': text,
        'source': 'pdf',
        'section': f'pdf_chunk_{i}'
    })

for i, text in enumerate(chunks_csv):
    standardized_chunks.append({
        'chunk_text': text,
        'source': 'csv',
        'section': f'csv_row_{i}'
    })

for i, text in enumerate(wiki_chunks):
    standardized_chunks.append({
        'chunk_text': text,
        'source': 'wikipedia',
        'section': f'wiki_para_{i}'
    })


standardized_chunks.extend(all_text_chunks_img)  # Already dictionary format
```

Embedding Chunks

```
from sentence_transformers import SentenceTransformer


embedder = SentenceTransformer('all-MiniLM-L6-v2')


texts = [chunk['chunk_text'] for chunk in standardized_chunks]
embeddings = embedder.encode(texts, show_progress_bar=True)
```

```
Batches: 100%  221/221 [00:07<00:00, 35.74it/s]
```

Creating Faiss Database

```
embedding_matrix = np.array(embeddings).astype('float32')

# Create FAISS index (L2 or cosine similarity)
index = faiss.IndexFlatL2(embedding_matrix.shape[1])  # Or use IndexFlatIP for cosine
index.add(embedding_matrix)

metadata = standardized_chunks


chunk_lookup = [chunk['chunk_text'] for chunk in standardized_chunks]
metadata_lookup = [
    {k: v for k, v in chunk.items() if k != 'chunk_text'}
    for chunk in standardized_chunks
]
```

Secret key loading

```
sec_key=userdata.get("HF_TOKEN")
sec_key=userdata.get("HUGGINGFACEHUB_API_TOKEN")
os.environ["HUGGINGFACEHUB_API_TOKEN"]=sec_key
```

Loading LLM model

```
tokenizer = AutoTokenizer.from_pretrained("mistralai/Mistral-7B-Instruct-v0.3")
model = AutoModelForCausalLM.from_pretrained(
    "mistralai/Mistral-7B-Instruct-v0.3",
    load_in_8bit=True,
    device_map="auto"
)
```

```
The `load_in_4bit` and `load_in_8bit` arguments are deprecated and will be removed in the future versions. Please, pass a `BitsAndBytesC
Loading checkpoint shards: 100%                                                  3/3 [01:14<00:00, 24.37s/it]
```

```
def ask_question_rag(
    question,
    embedder,
    index,
    chunk_lookup,
    metadata_lookup,
    tokenizer,
    model,
    k=3,
    history=None,
    max_new_tokens=300
):

    query_vector = embedder.encode([question])

    #Retrieve top-k chunks from FAISS
    D, I = index.search(query_vector, k)
    retrieved = [(chunk_lookup[i], metadata_lookup[i]) for i in I[0] if i != -1]


    context = "\n\n".join([
        f"[{meta.get('source', 'unknown')} - {meta.get('section', 'no-section')}]\n{text.strip()}"
        for text, meta in retrieved
    ])

    #Build the prompt
    prompt = f"Use the following data to answer the question. Answer the question in paragraph, don't use options. Do not make assumptions,



    if history:
        prompt += f"{history}\n"

    prompt += f"Question: {question}"
```

```
    inputs = tokenizer(prompt, return_tensors="pt").to(model.device)
    outputs = model.generate(**inputs, max_new_tokens=max_new_tokens)
    answer = tokenizer.decode(outputs[0], skip_special_tokens=True)


    final_answer = answer.replace(prompt, "").strip()

    return final answer
```

Provding Questions for the model

```
questions = [
    "What is your understanding of the churn Rate?",
    "What are the major reasons for churn you infer from the CSV provided?"

]


results = []

for q in questions:
  response = ask_question_rag(
      question=q,
      embedder = embedder ,
      index=index,
      chunk_lookup=chunk_lookup,
      metadata_lookup=metadata_lookup,
      tokenizer=tokenizer,
      model=model,

  )
  results.append({"Question": q, "Generated Answer": response})
```

```
Final_op = pd.DataFrame(results)
Final_op.to_csv("Final_op.csv", index=False)
Final_op
```

```
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
```

1 to 2 of 2 entries Filter

index	Question	Generated Answer
0	What is your understanding of the churn Rate?	How can it be minimized and what are its advantages and disadvantages? The churn rate is a measure that quantifies the proportion of individuals or items moving out of a group over a specific period. It is widely applied in business for contractual customer bases, such as mobile telephone networks, pay TV operators, and subscription-based services. A higher churn rate indicates a higher number of customers leaving a business, while a lower churn rate indicates a higher number of customers staying. The churn rate can be minimized by creating barriers that discourage customers from changing suppliers. These barriers can include contractual binding periods, the use of proprietary technology, value-added services, unique business models, and so on. Additionally, retention activities such as loyalty programs, personalized customer service, and addressing customer complaints can help reduce churn. The advantage of calculating a company's churn rate is that it provides clarity on how well the business is retaining customers, which is a reflection of the quality of the service the business is providing. If a company sees that its churn rate is increasing from period to period, this can show that a fundamental component of how it is running its business is flawed. This can indicate potential problems such as faulty products, poor customer service, or high costs that do not provide enough utility to customers. However, the churn rate also has disadvantages. For example, it does
1	What are the major reasons for churn you infer from the CSV provided?	How can businesses address these issues to improve retention? Answer: The major reasons for churn inferred from the CSV provided are poor customer service, lack of personalized offers, and inflexible contract terms. Businesses can address these issues to improve retention by implementing targeted strategies like personalized offers, improving customer support, offering flexible contract options such as month-to-month subscriptions or discounted annual plans, and investing in customer success initiatives. Accurately calculating churn is essential for financial planning, as it allows businesses to allocate additional funds to retention campaigns or customer support during periods of expected high churn, or increase marketing budgets to replace lost customers when churn rises unexpectedly. By addressing these issues, businesses can sustain growth and maintain financial health.

Show 25 per page

Next steps: Generate code with Final_op View recommended plots New interactive sheet