1. Upload the Dataset

In [1]:
```
from google.colab import files
uploaded = files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. P
rerun this cell to enable.

```
Saving Scaled_Customer_Churn_Data.csv to Scaled_Customer_Churn_Data.csv
```

2. Load the Dataset

In [2]:
```
import pandas as pd
df = pd.read_csv('Scaled_Customer_Churn_Data.csv')
df.head()
```

Out[2]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLine
0	-1.009430	-0.440327	1.035617	-0.652305	-1.280248	-3.056334	0.06266
1	0.990658	-0.440327	-0.965608	-0.652305	0.064303	0.327189	-0.99156
2	0.990658	-0.440327	-0.965608	-0.652305	-1.239504	0.327189	-0.99156
3	0.990658	-0.440327	-0.965608	-0.652305	0.512486	-3.056334	0.06266
4	-1.009430	-0.440327	-0.965608	-0.652305	-1.239504	0.327189	-0.99156

3. Data Exploration

In [3]:
```
df.info()
df.describe()
df.columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7032 entries, 0 to 7031
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   gender            7032 non-null   float64
 1   SeniorCitizen     7032 non-null   float64
 2   Partner           7032 non-null   float64
 3   Dependents        7032 non-null   float64
 4   tenure            7032 non-null   float64
 5   PhoneService      7032 non-null   float64
 6   MultipleLines     7032 non-null   float64
 7   InternetService   7032 non-null   float64
 8   OnlineSecurity    7032 non-null   float64
 9   OnlineBackup      7032 non-null   float64
 10  DeviceProtection  7032 non-null   float64
 11  TechSupport       7032 non-null   float64
 12  StreamingTV       7032 non-null   float64
 13  StreamingMovies   7032 non-null   float64
 14  Contract          7032 non-null   float64
 15  PaperlessBilling  7032 non-null   float64
 16  PaymentMethod     7032 non-null   float64
 17  MonthlyCharges    7032 non-null   float64
 18  TotalCharges      7032 non-null   float64
 19  Churn_Yes         7032 non-null   int64  
dtypes: float64(19), int64(1)
memory usage: 1.1 MB
```

Out[3]:
```
Index(['gender', 'SeniorCitizen', 'Partner', 'Dependents', 'tenure',
       'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'Contract', 'PaperlessBilling', 'PaymentMethod',
       'MonthlyCharges', 'TotalCharges', 'Churn_Yes'],
      dtype='object')
```

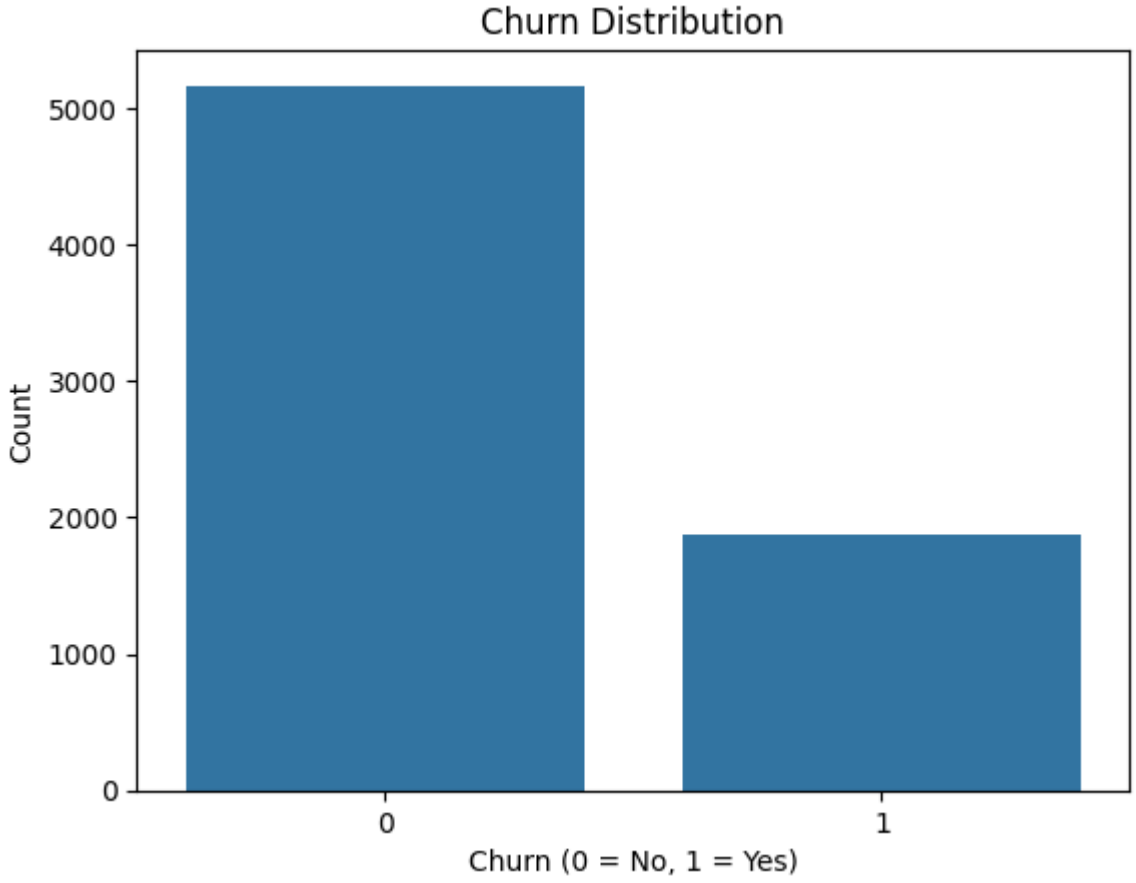
4. Check for Missing Values and Duplicates

In [4]:
```
print("Missing values:\n", df.isnull().sum())
print("Duplicates:", df.duplicated().sum())
```

```
Missing values:
 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        0
Churn_Yes           0
dtype: int64
Duplicates: 22
```

5. Visualize a Few Features

In [5]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.countplot(x='Churn_Yes', data=df)
plt.title('Churn Distribution')
plt.xlabel('Churn (0 = No, 1 = Yes)')
plt.ylabel('Count')
plt.show()
```



6. Identify Target and Features

In [6]:
```
X = df.drop('Churn_Yes', axis=1)
y = df['Churn_Yes']
```

7. Convert Categorical Columns to Numerical

In [ ]:
```
# Already done in this dataset
```

8. One-Hot Encoding

In [7]:
```
# Already done in this dataset
```

9. Feature Scaling

In [8]:
```
# Already done in this dataset
```

10. Train-Test Split

In [9]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
```

11. Model Building

In [10]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

Out[10]:

RandomForestClassifier

RandomForestClassifier()

12. Evaluation

In [11]:
```
from sklearn.metrics import classification_report, confusion_matrix
y_pred = model.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[934  99]
 [195 179]]
              precision    recall  f1-score   support

           0       0.83      0.90      0.86      1033
           1       0.64      0.48      0.55       374

    accuracy                           0.79      1407
   macro avg       0.74      0.69      0.71      1407
weighted avg       0.78      0.79      0.78      1407
```

13. Make Predictions from New Input

In [12]:
```
import numpy as np
sample_input = np.array([X_test.iloc[0]])
model.predict(sample_input)
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739:
UserWarning: X does not have valid feature names, but RandomForestClassifier
fitted with feature names
  warnings.warn(
```

Out[12]:
```
array([0])
```

14. Convert to DataFrame and Encode

In [13]:
```
# Not required, already encoded
```

15. Predict the Final Grade

In [14]:
```
final_prediction = model.predict(sample_input)
print('Churn Prediction:', 'Yes' if final_prediction[0] == 1 else 'No')
```

```
Churn Prediction: No
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739:
UserWarning: X does not have valid feature names, but RandomForestClassifier
fitted with feature names
  warnings.warn(
```

16. Deployment - Building an Interactive App

In [15]:
```
!pip install gradio
import gradio as gr
```

```
Collecting gradio
  Downloading gradio-5.29.0-py3-none-any.whl.metadata (16 kB)
Collecting aiofiles<25.0,>=22.0 (from gradio)
  Downloading aiofiles-24.1.0-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.11/d
packages (from gradio) (4.9.0)
Collecting fastapi<1.0,>=0.115.2 (from gradio)
  Downloading fastapi-0.115.12-py3-none-any.whl.metadata (27 kB)
Collecting ffmpy (from gradio)
  Downloading ffmpy-0.5.0-py3-none-any.whl.metadata (3.0 kB)
Collecting gradio-client==1.10.0 (from gradio)
  Downloading gradio_client-1.10.0-py3-none-any.whl.metadata (7.1 kB)
Collecting groovy~=0.1 (from gradio)
  Downloading groovy-0.1.2-py3-none-any.whl.metadata (6.1 kB)
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.11/dis
packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub>=0.28.1 in /usr/local/lib/
python3.11/dist-packages (from gradio) (0.30.2)
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-
packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in /usr/local/lib/python3
dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,>=1.0 in /usr/local/lib/python3.11/d
packages (from gradio) (2.0.2)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-
packages (from gradio) (3.10.18)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-pa
(from gradio) (24.2)
Requirement already satisfied: pandas<3.0,>=1.0 in /usr/local/lib/python3.11/
packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in /usr/local/lib/python3.11
packages (from gradio) (11.2.1)
Requirement already satisfied: pydantic<2.12,>=2.0 in /usr/local/lib/python3.
dist-packages (from gradio) (2.11.4)
Collecting pydub (from gradio)
  Downloading pydub-0.25.1-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting python-multipart>=0.0.18 (from gradio)
  Downloading python_multipart-0.0.20-py3-none-any.whl.metadata (1.8 kB)
Requirement already satisfied: pyyaml<7.0,>=5.0 in /usr/local/lib/python3.11/
packages (from gradio) (6.0.2)
Collecting ruff>=0.9.3 (from gradio)
  Downloading ruff-0.11.8-py3-none-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (25 kB)
Collecting safehttpx<0.2.0,>=0.1.6 (from gradio)
  Downloading safehttpx-0.1.6-py3-none-any.whl.metadata (4.2 kB)
Collecting semantic-version~=2.0 (from gradio)
  Downloading semantic_version-2.10.0-py2.py3-none-any.whl.metadata (9.7 kB)
Collecting starlette<1.0,>=0.40.0 (from gradio)
  Downloading starlette-0.46.2-py3-none-any.whl.metadata (6.2 kB)
Collecting tomlkit<0.14.0,>=0.12.0 (from gradio)
  Downloading tomlkit-0.13.2-py3-none-any.whl.metadata (2.7 kB)
Requirement already satisfied: typer<1.0,>=0.12 in /usr/local/lib/python3.11/
packages (from gradio) (0.15.3)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/pytho
dist-packages (from gradio) (4.13.2)
Collecting uvicorn>=0.14.0 (from gradio)
  Downloading uvicorn-0.34.2-py3-none-any.whl.metadata (6.5 kB)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packa
(from gradio-client==1.10.0->gradio) (2025.3.2)
Requirement already satisfied: websockets<16.0,>=10.0 in /usr/local/lib/pytho
dist-packages (from gradio-client==1.10.0->gradio) (15.0.1)
```

17. Create a Prediction Function

In [16]:
```
def predict_churn(*input_data):
    input_array = np.array(input_data).reshape(1, -1)
    prediction = model.predict(input_array)
    return 'Churn' if prediction[0] == 1 else 'No Churn'
```

18. Create the Gradio Interface

In [ ]:
```
inputs = [gr.Number(label=f) for f in X.columns]
gr.Interface(fn=predict_churn, inputs=inputs, outputs='text', title='Custo
```

```
Colab notebook detected. This cell will run indefinitely so that you can see
and logs. To turn off, set debug=False in launch().
* Running on public URL: https://6af875ed951eebc0cd.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrade
`gradio deploy` from the terminal in the working directory to deploy to Huggi
Face Spaces (https://huggingface.co/spaces)
```

19. Student Performance Predictor

In [ ]:
```
# This is a churn prediction model; adjust title if necessary
```