Importing Libraries and Dataset

```
import pandas as pd
```

```
df=pd.read_csv("/content/drive/MyDrive/Colab_ csv_files /Telco-Customer-Churn.csv")
df.head()
```

...	Dev
...	
...	
...	
...	
...	

```
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')
```

Data Preprocessing

```
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	0
Churn	0

dtype: int64

```
df.dtypes
```

	0
customerID	object
gender	object
SeniorCitizen	int64
Partner	object
Dependents	object
tenure	int64
PhoneService	object
MultipleLines	object
InternetService	object
OnlineSecurity	object
OnlineBackup	object
DeviceProtection	object
TechSupport	object
StreamingTV	object
StreamingMovies	object
Contract	object
PaperlessBilling	object
PaymentMethod	object
MonthlyCharges	float64
TotalCharges	object
Churn	object

dtype: object

```
df["TotalCharges"]=pd.to_numeric(df["TotalCharges"],errors="coerce")
```

Datatype of "TotalCharges" is changed to numeric from object since it has numeric values which might helps in future possible numeric operations.

```
df.duplicated().sum()
```

```
np.int64(0)
```

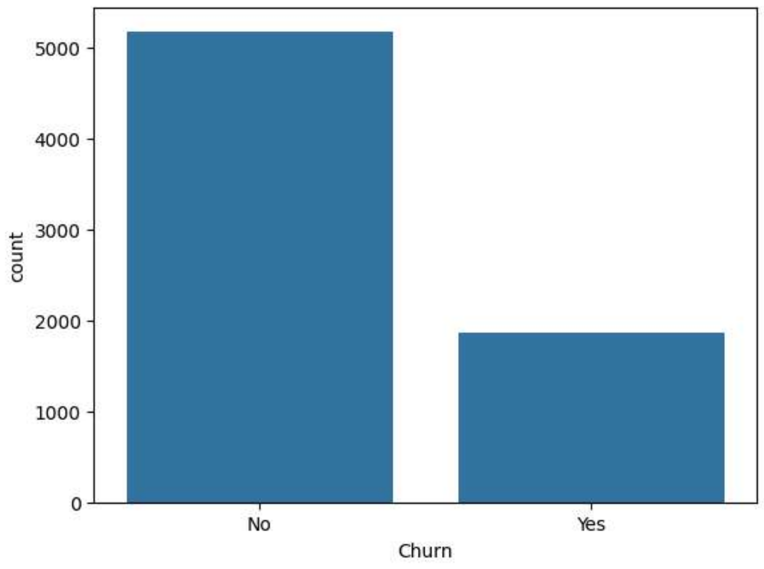
Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns
```

Now, let's analyze how other factors influence churn by comparing and plotting them against the churn variable.

```
sns.countplot(data=df, x="Churn")
```

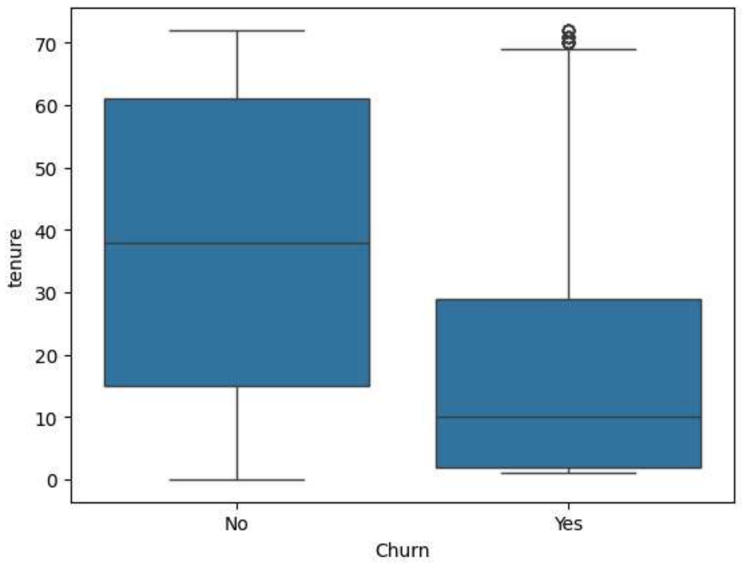
```
<Axes: xlabel='Churn', ylabel='count'>
```



Seems like,in this dataset Customers not leaving is more.

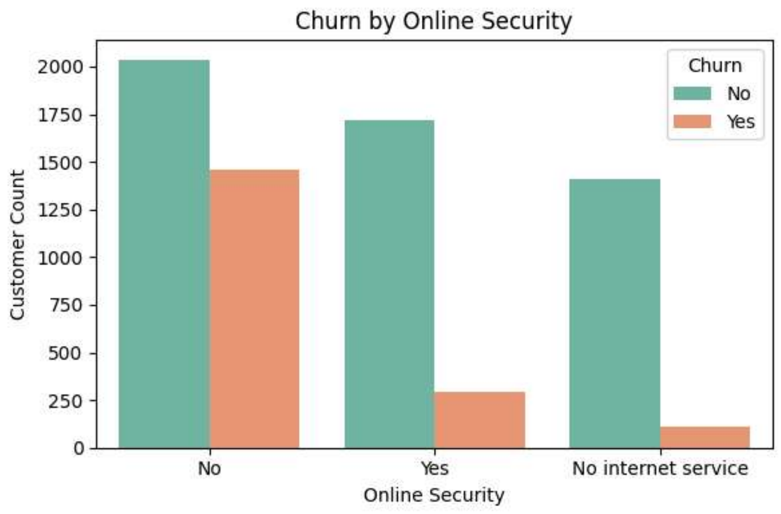
```
sns.boxplot(data=df, x="Churn", y="tenure")
```

```
<Axes: xlabel='Churn', ylabel='tenure'>
```

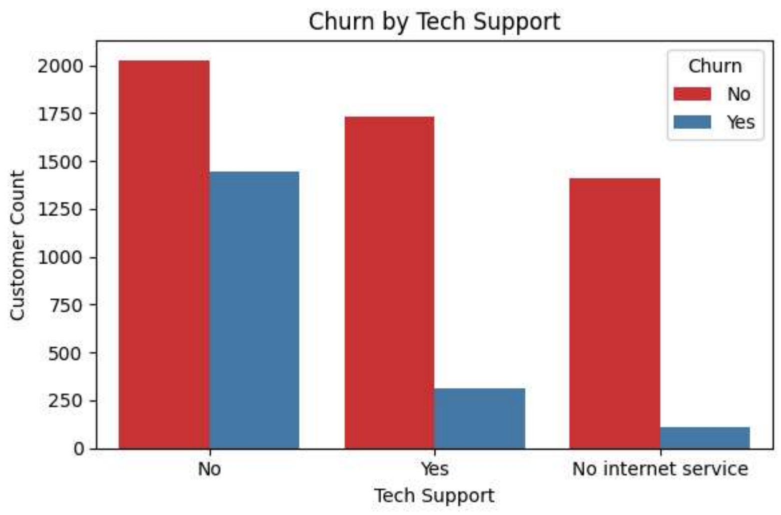


It shows churn rate is more with the customers with shorter tenure and Loyal customers stayed more than 40 moths.

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(6, 4))
sns.countplot(data=df, x='OnlineSecurity', hue='Churn', palette='Set2')
plt.title('Churn by Online Security')
plt.xlabel('Online Security')
plt.ylabel('Customer Count')
plt.legend(title='Churn')
plt.tight_layout()
plt.show()
```

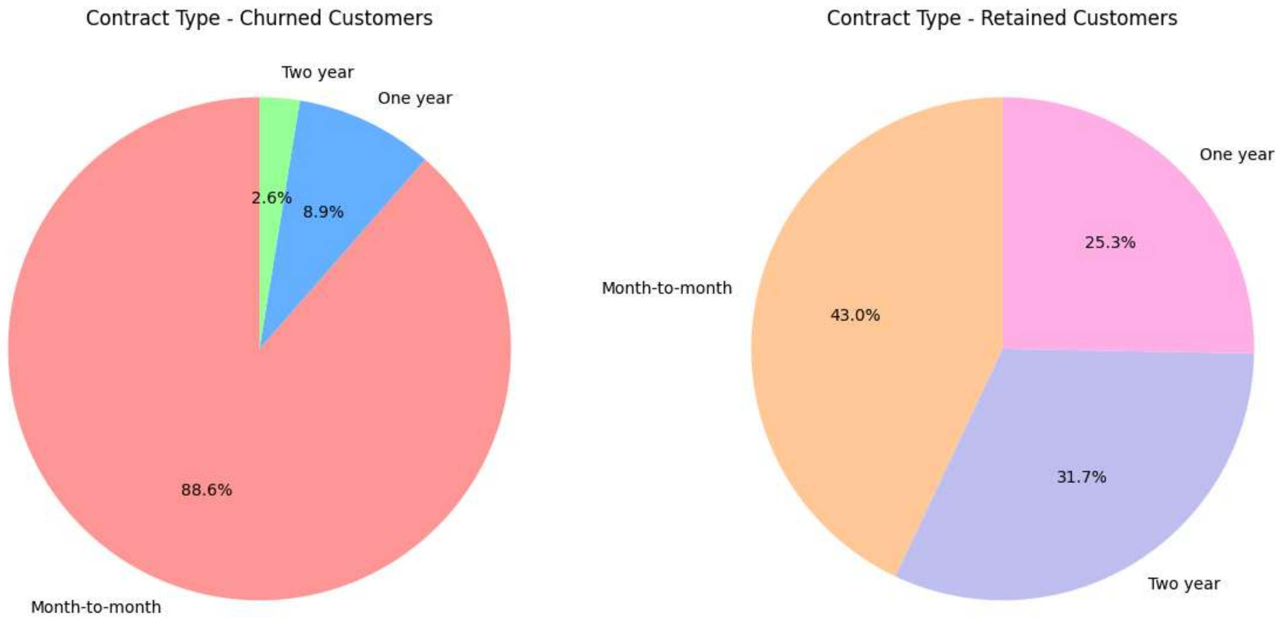


```
plt.figure(figsize=(6, 4))
sns.countplot(data=df, x='TechSupport', hue='Churn', palette='Set1')
plt.title('Churn by Tech Support')
plt.xlabel('Tech Support')
plt.ylabel('Customer Count')
plt.legend(title='Churn')
plt.tight_layout()
plt.show()
```



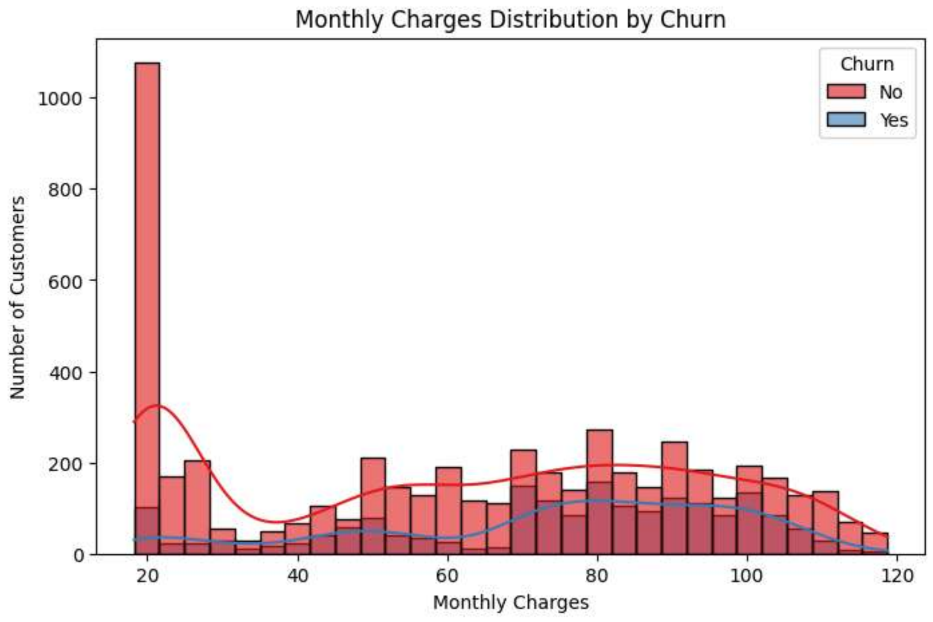
Customers leave where there is no online security and Tech support.

```
churn_yes=df[df['Churn']=='Yes']['Contract'].value_counts()
churn_no=df[df['Churn']=='No']['Contract'].value_counts()
fig, ax = plt.subplots(1, 2, figsize=(12, 6))
ax[0].pie(churn_yes, labels=churn_yes.index, autopct='%1.1f%%', startangle=90, colors=['#ff9999','#66b3ff','#99ff99'])
ax[0].set_title('Contract Type - Churned Customers')
ax[1].pie(churn_no, labels=churn_no.index, autopct='%1.1f%%', startangle=90, colors=['#ffcc99','#c2c2f0','#ffb3e6'])
ax[1].set_title('Contract Type - Retained Customers')
plt.tight_layout()
plt.show()
```



In both churned and retained customer groups, the majority of customers are on month-to-month contracts. However, it can be concluded that customers with two-year contracts are more likely to stay, as the churn rate is significantly lower among them.

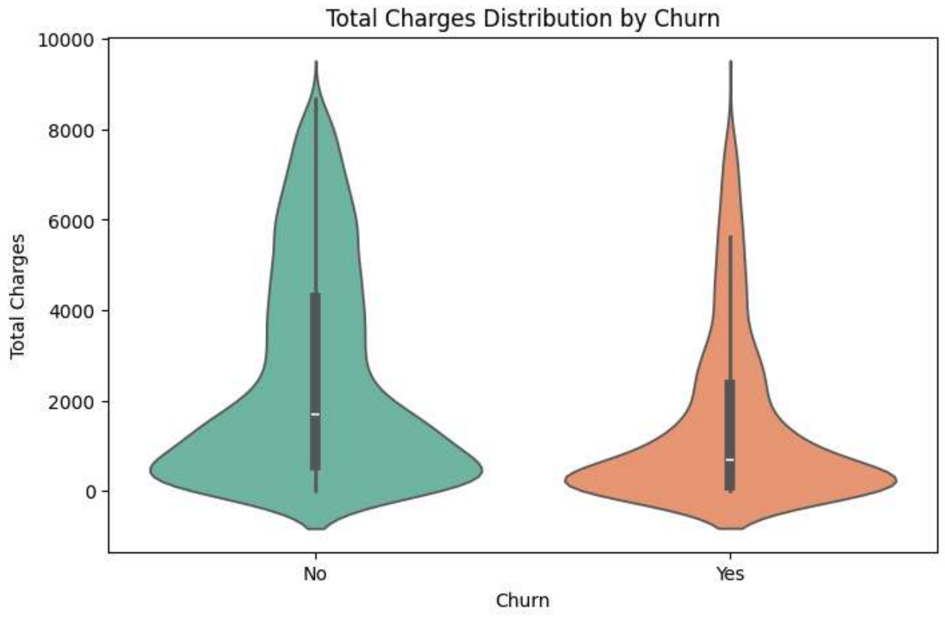
```
plt.figure(figsize=(8, 5))
sns.histplot(data=df, x='MonthlyCharges', hue='Churn', kde=True, bins=30, palette='Set1', alpha=0.6)
plt.title('Monthly Charges Distribution by Churn')
plt.xlabel('Monthly Charges')
plt.ylabel('Number of Customers')
plt.show()
```



Customers prefer services with low charges.

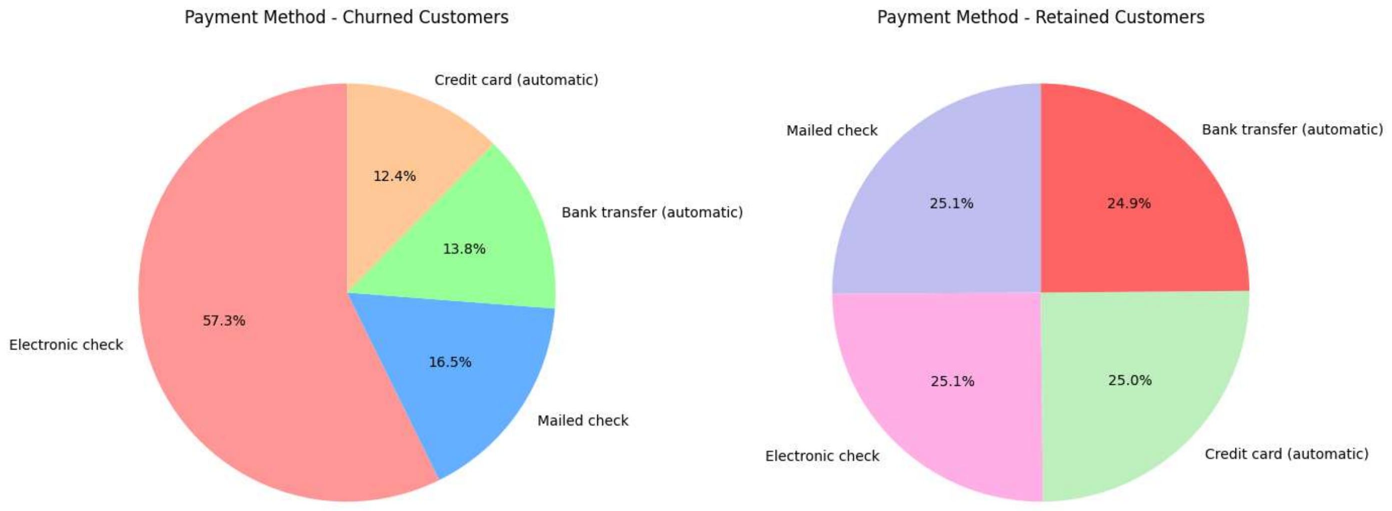
```
plt.figure(figsize=(8, 5))
sns.violinplot(data=df, x='Churn', y='TotalCharges', palette='Set2')
plt.title('Total Charges Distribution by Churn')
plt.xlabel('Churn')
plt.ylabel('Total Charges')
plt.show()
```

```
/tmp/ipython-input-51-2391231690.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.violinplot(data=df, x='Churn', y='TotalCharges', palette='Set2')
```



The violin plot indicates that churned customers typically leave early, as reflected by the narrower top of the distribution, concentrated around low TotalCharges. In contrast, retained customers show a broader distribution across higher TotalCharges, meaning they not only stay longer but continue to accumulate charges over time — highlighting stronger loyalty and longer tenure.

```
churn_yes = df[df['Churn'] == 'Yes']['PaymentMethod'].value_counts()
churn_no = df[df['Churn'] == 'No']['PaymentMethod'].value_counts()
fig, ax = plt.subplots(1, 2, figsize=(14, 7))
ax[0].pie(churn_yes, labels=churn_yes.index, autopct='%1.1f%%', startangle=90,colors=['#ff9999','#66b3ff','#99ff99','#ffcc99'])
ax[0].set_title('Payment Method - Churned Customers')
ax[1].pie(churn_no, labels=churn_no.index, autopct='%1.1f%%', startangle=90,colors=['#c2c2f0','#ffb3e6','#c2f0c2','#ff6666'])
ax[1].set_title('Payment Method - Retained Customers')
plt.tight_layout()
plt.show()
```



Customers who churned were more likely to use Electronic Check as their payment method, indicating a possible correlation between manual payment behavior and churn. In contrast, retained customers predominantly used automatic payment methods such as Bank Transfer or Credit Card, suggesting they are either more committed or less engaged with the billing process, possibly due to convenience.

Preprocessing: Encoding & Scaling

```
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Drop rows with NaN in 'TotalCharges' introduced by coercion
df.dropna(subset=['TotalCharges'], inplace=True)

# Encode categorical columns
df_encoded = pd.get_dummies(df.drop('Churn', axis=1))
df_encoded['Churn'] = df['Churn'].map({'No': 0, 'Yes': 1})

# Feature scaling
scaler = StandardScaler()
X = scaler.fit_transform(df_encoded.drop('Churn', axis=1))
y = df_encoded['Churn']
```

Model Building: Logistic Regression

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, roc_auc_score, classification_report, roc_curve

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LogisticRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
y_pred
```

```
array([0, 0, 1, ..., 0, 0, 0])
```

Model Evaluation

```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("ROC-AUC Score:", roc_auc_score(y_test, model.predict_proba(X_test)[:,1]))

# ROC Curve
fpr, tpr, _ = roc_curve(y_test, model.predict_proba(X_test)[:,1])
plt.plot(fpr, tpr)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
```

```
Accuracy: 0.7391613361762616
Confusion Matrix:
 [[750 283]
 [ 84 290]]
Classification Report:
```