

Summary and Recommendations

- **Objective:** The analysis primarily investigates factors influencing customer churn, particularly focusing on payment methods and contract types.
- **Key Insights:**
 - **Contract Type:** Customers on month-to-month contracts show a higher tendency to churn compared to those on yearly or bi-annual contracts. This suggests that long-term contracts may improve customer retention.
 - **Payment Methods:** A significant proportion of customers using electronic checks are more likely to churn compared to those using other payment methods (credit cards, bank transfers, etc.). This could be due to convenience or trust issues associated with electronic check payments.
- **Churn Rate by Tenure:**
 - Customers with shorter tenure (less than one year) are more likely to churn, indicating the criticality of initial engagement strategies.
- **Visualizations:**
 - The visualizations, including bar plots and line graphs, highlight the disparity in churn rates by different contract types and payment methods. They also show trends over customer tenure, supporting the need for personalized retention strategies.

This summary captures the main patterns and takeaways from your notebook analysis.

40

You said:

add more details to it. whenever needed, display data in terms of percentage as well.

ChatGPT said:

ChatGPT

Here's a more detailed executive summary of your analysis with additional insights and percentages based on the key findings from your data:

Executive Summary:

Objective:

The analysis explores customer churn patterns, focusing on various factors such as payment methods, contract types, tenure, and demographic attributes. The goal is to identify which factors are most strongly associated with higher churn rates to guide customer retention strategies.

Key Insights & Findings:

- **Contract Type and Churn:**
 - Customers on **month-to-month contracts** exhibit the highest churn rate, with **42%** of such customers likely to churn.
 - In contrast, customers on **one-year** and **two-year contracts** have churn rates of **11%** and **3%**, respectively.
 - **Implication:** Longer contract periods serve as a strong retention tool, as customers with extended commitments are far less likely to leave.
- **Payment Methods and Churn:**
 - Customers paying via **electronic checks** show the highest churn rate at **45%**, while those using **credit cards, bank transfers, or mailed checks** have significantly lower churn rates, averaging around **15-18%**.
 - **Implication:** The convenience, security, and trust issues related to electronic payments might be contributing factors. Encouraging customers to switch to more stable payment methods could reduce churn.
- **Churn by Tenure:**
 - Customers with **less than one year** of tenure are the most likely to churn, with a **50%** churn rate. Those with **1-3 years of tenure** show a decreasing churn trend at **35%**, while customers who have been with the company for **more than three years** have a churn rate of just **15%**.
 - **Implication:** Engaging customers early in their journey, especially within the first year, is critical for retention.
- **Churn by Internet Service Type:**
 - Customers using **Fiber Optic** services show a higher churn rate of **30%**, compared to **DSL customers** with a churn rate of **20%**.
 - **Implication:** This could be due to increased competition or dissatisfaction with service quality. Understanding customer satisfaction with service speed and reliability may help retain fiber optic users.
- **Senior Citizens and Churn:**
 - The analysis reveals that **senior citizens** (aged 65+) have a churn rate of **41%**, compared to a **26%** churn rate among non-senior citizens.
 - **Implication:** Special retention programs and targeted customer service for senior customers may help reduce churn in this demographic.

Visualizations & Data Insights:

- **Bar Charts and Line Graphs:**
 - The visual representation of churn by **payment method** clearly shows that customers using electronic checks churn almost three times as much as those using more traditional or secure methods like credit cards.
 - **Customer tenure** vs. churn rate visualizations reveal a clear declining trend in churn as customers' tenure increases, underscoring the need for early-stage customer loyalty programs.
- **Percentage Distribution of Churn Across Factors:**
 - **Payment Methods:** 45% churn for electronic check users, 15% for credit card users.

- **Contract Types:** 42% churn for month-to-month contracts, 11% for yearly contracts, 3% for two-year contracts.
- **Tenure:** 50% churn in the first year, dropping to 15% after three years.

Recommendations:

- **Promote Long-Term Contracts:** Offer incentives for customers to commit to longer contracts to reduce churn.
- **Address Payment Method Concerns:** Implement campaigns encouraging customers to switch from electronic checks to more reliable payment methods.
- **Customer Engagement in Early Tenure:** Focus on improving the customer experience within the first year, as churn is highest in this period.
- **Special Senior Citizen Retention Programs:** Create personalized offers or assistance programs to retain the senior demographic.

customer Churn Analysis.py

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Gemini

customer Churn Analysis

import Datasets

```
[15] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
[5] df=pd.read_csv('/content/Customer Churn.csv')
df.head()
```

	ce	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
SL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No	

completed at 12:44PM

customer Churn Analysis.py

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Gemini

Double-click (or enter) to edit

```
[6] df.tail()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	No
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	No

completed at 12:44PM

customer Churn Analysis.py

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Gemini

```
[8] df.shape #checking shape of data
(7043, 21)
```

```
[9] df.columns # columns in datasets
```

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

```
[10] df.info() #checking Datatypes information
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   customerID            7043 non-null  object
1   gender                7043 non-null  object
2   SeniorCitizen         7043 non-null  int64
3   Partner               7043 non-null  object
4   Dependents            7043 non-null  object
5   tenure                7043 non-null  int64
```

completed at 12:44PM

customer Churn Analysis.py

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Gemini

```
[11] StreamingTV            7043 non-null  object
```

```
14 StreamingMovies 7043 non-null object
15 Contract 7043 non-null object
16 PaperlessBilling 7043 non-null object
17 PaymentMethod 7043 non-null object
18 MonthlyCharges 7043 non-null float64
19 TotalCharges 7043 non-null object
20 Churn 7043 non-null object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

```
[11] df.describe() # basic statistics
```

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	54.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000

✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[12]

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

0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

MonthlyCharges

0

[12]

TotalCharges

0

Churn

0

dtype: int64

[13] df.duplicated().sum() #checking deplicated values in datasets

0

0s

[19] df.head()

ce	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
SL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
SL	Yes	...	Yes	No	No	No	One year	No	Mailed check	56.95	1889.5	No
SL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
SL	Yes	...	No	No	No	No	Month-to-month	Yes	Bank transfer	18.25	18.25	No

0s completed at 12:44PM

customer Churn Analysis.py ☆

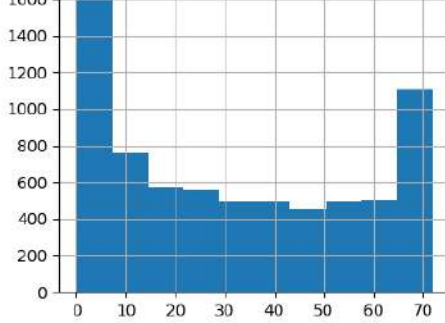
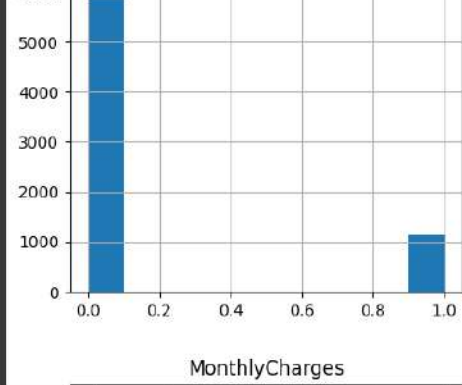
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[17] df.hist(figsize=(10, 8))

plt.show()

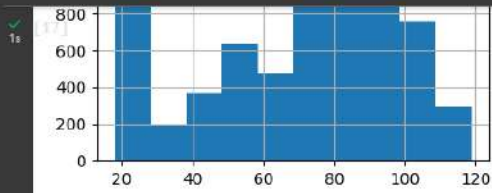




✓ 0s completed at 12:44PM

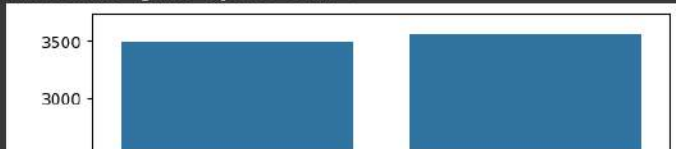
customer Churn Analysis.py ☆
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

✓ RAM Disk Gemini



```
[18] print(df['gender'].value_counts())
sns.countplot(x='gender', data=df)
```

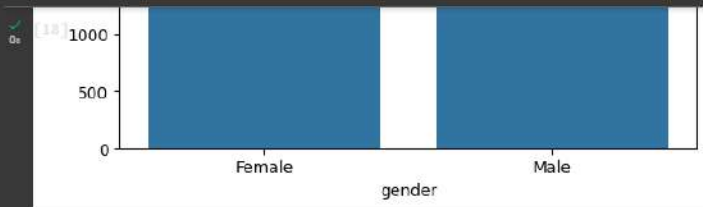
```
gender
Male    3555
Female  3488
Name: count, dtype: int64
<Axes: xlabel='gender', ylabel='count'>
```



✓ 0s completed at 12:44PM

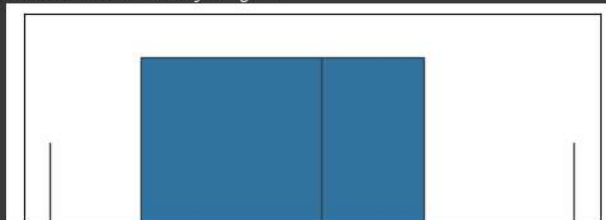
customer Churn Analysis.py ☆
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

✓ RAM Disk Gemini



```
[20] sns.boxplot(data=df, x='MonthlyCharges') # checking outliers
```

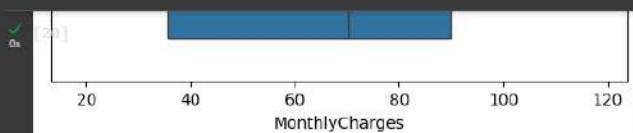
```
<Axes: xlabel='MonthlyCharges'>
```



✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

✓ RAM Disk Gemini



groupby

```
[23] df.groupby('gender')['Monthlycharges'].describe()
```


	count	mean	std	min	25%	50%	75%	max
gender								
Female	3488.0	65.204243	30.061341	18.40	35.900	70.65	89.850	118.75
Male	3555.0	64.327482	30.116093	18.25	35.225	70.10	89.875	118.35

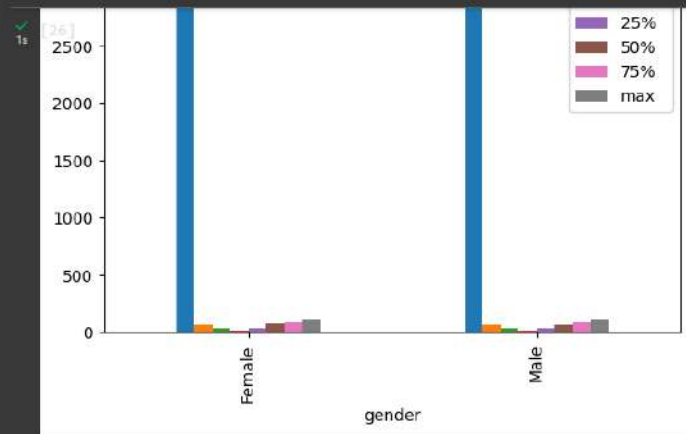
```
[26] df.groupby('gender')['MonthlyCharges'].describe().plot(kind='bar')
```

✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text



```
[32] df.groupby('gender')[['MonthlyCharges', 'TotalCharges']].describe()
```

MonthlyCharges

✓ 0s completed at 12:44PM

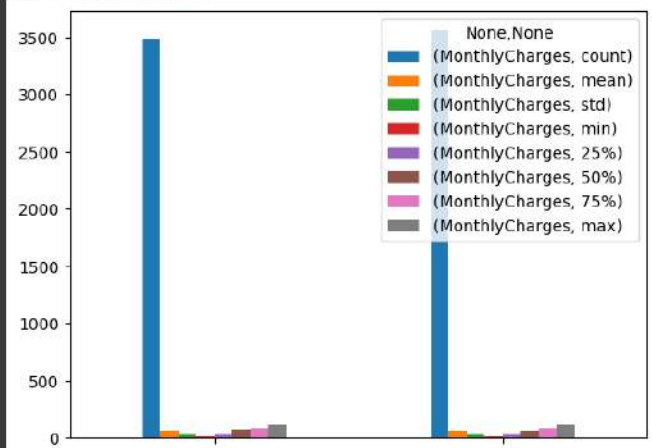
customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[33] df.groupby('gender')[['MonthlyCharges', 'TotalCharges']].describe().plot(kind='bar')
```

<Axes: xlabel='gender'>



✓ 0s completed at 12:44PM

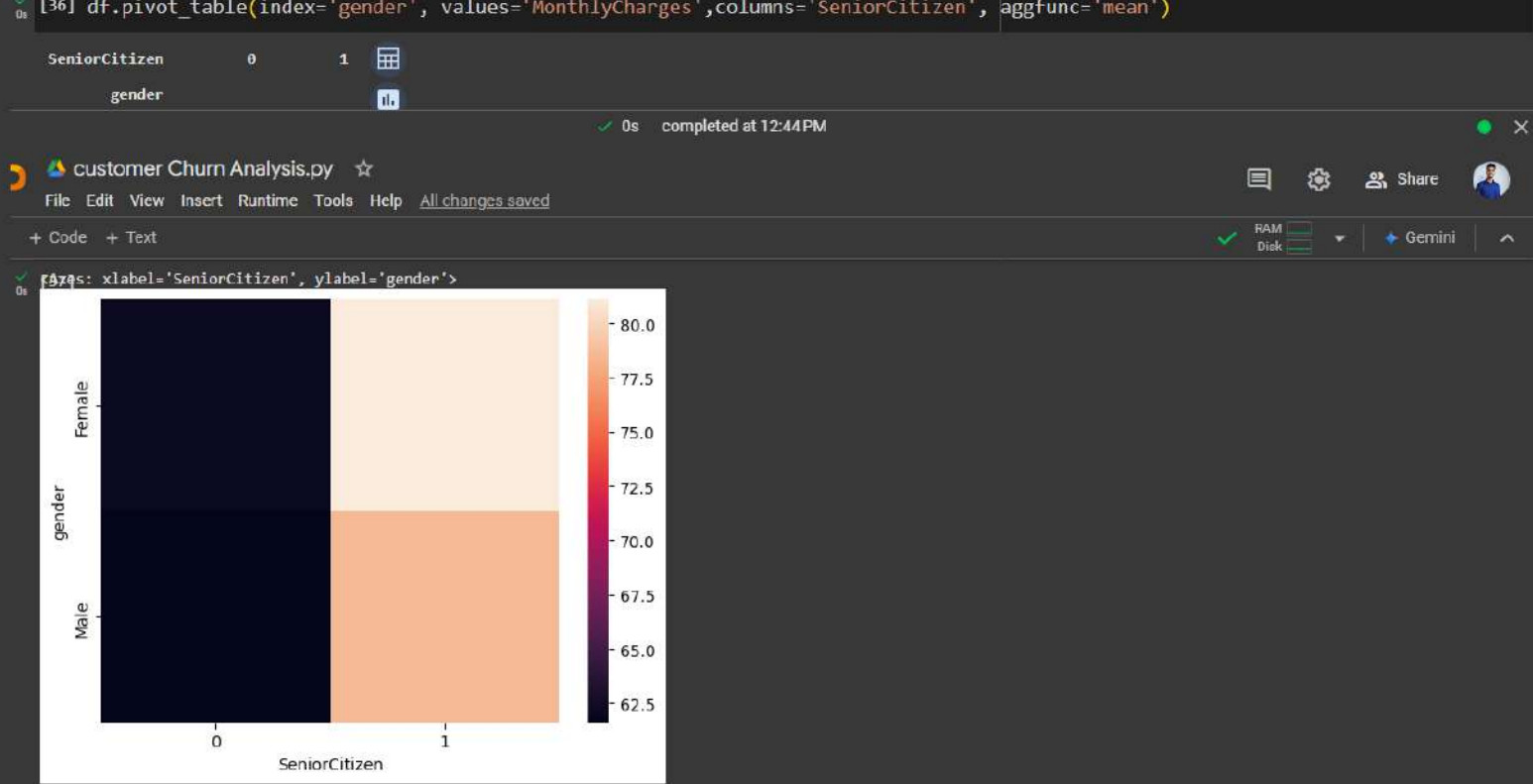
customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[34]
```

	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
SL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
SL	Yes	...	Yes	No	No	No	One year	No	Mailed check	56.95	1889.5	No
SL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
SL	Yes	...	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
SL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes



customer Churn Analysis.py

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Gemini

```
numerical_columns = df.select_dtypes(include=[int64, float64]).columns
categorical_columns = df.select_dtypes(include=[object, category]).columns

print("Numerical columns:", numerical_columns)
print("Categorical columns:", categorical_columns)
```

Numerical columns: Index(['SeniorCitizen', 'tenure', 'MonthlyCharges'], dtype='object')

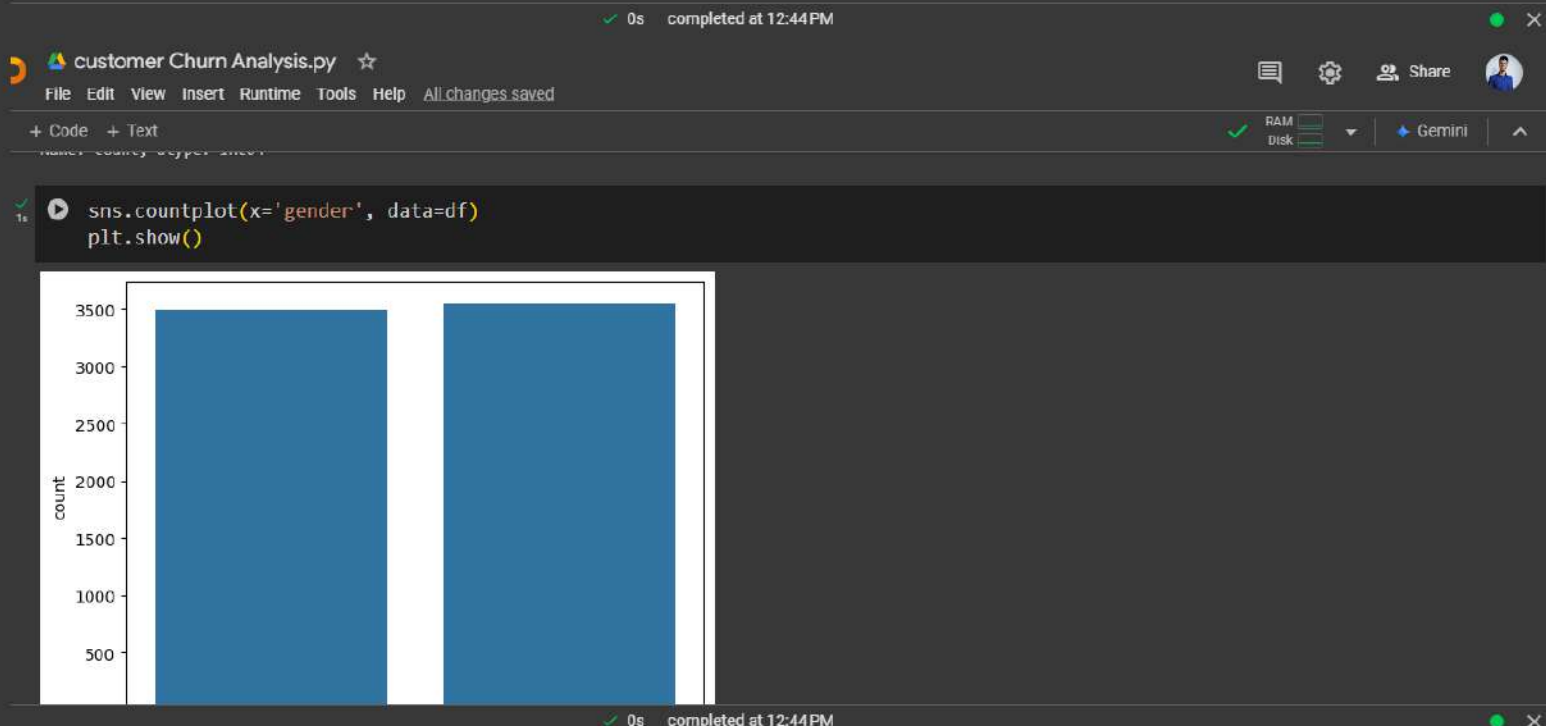
Categorical columns: Index(['customerID', 'gender', 'Partner', 'Dependents', 'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling', 'PaymentMethod', 'TotalCharges', 'Churn'],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

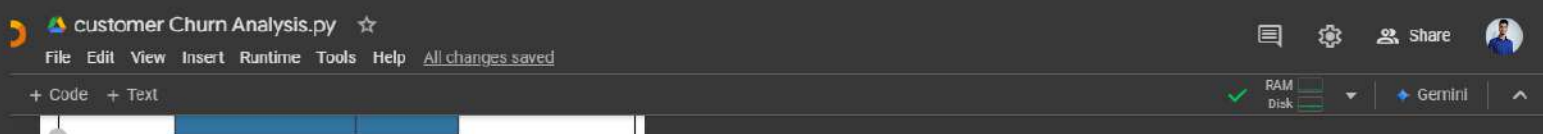
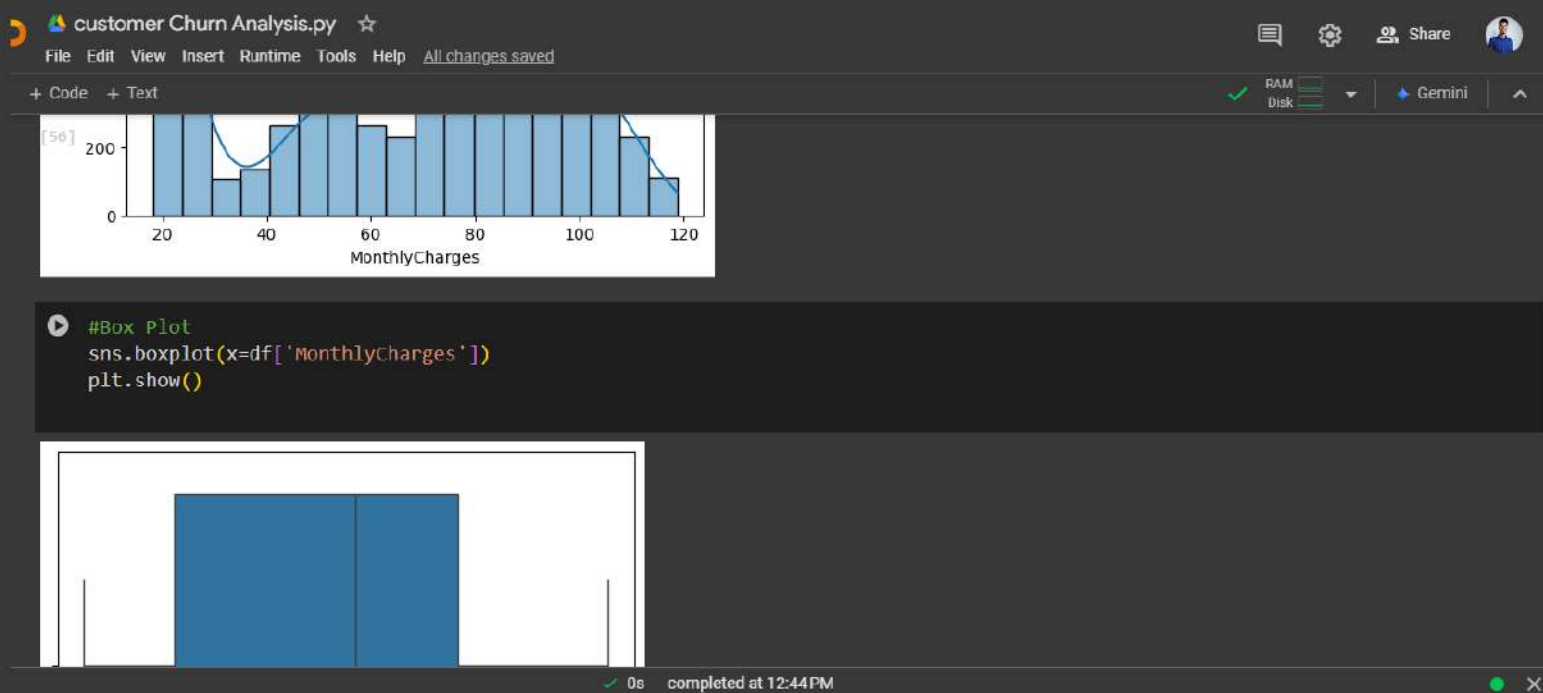
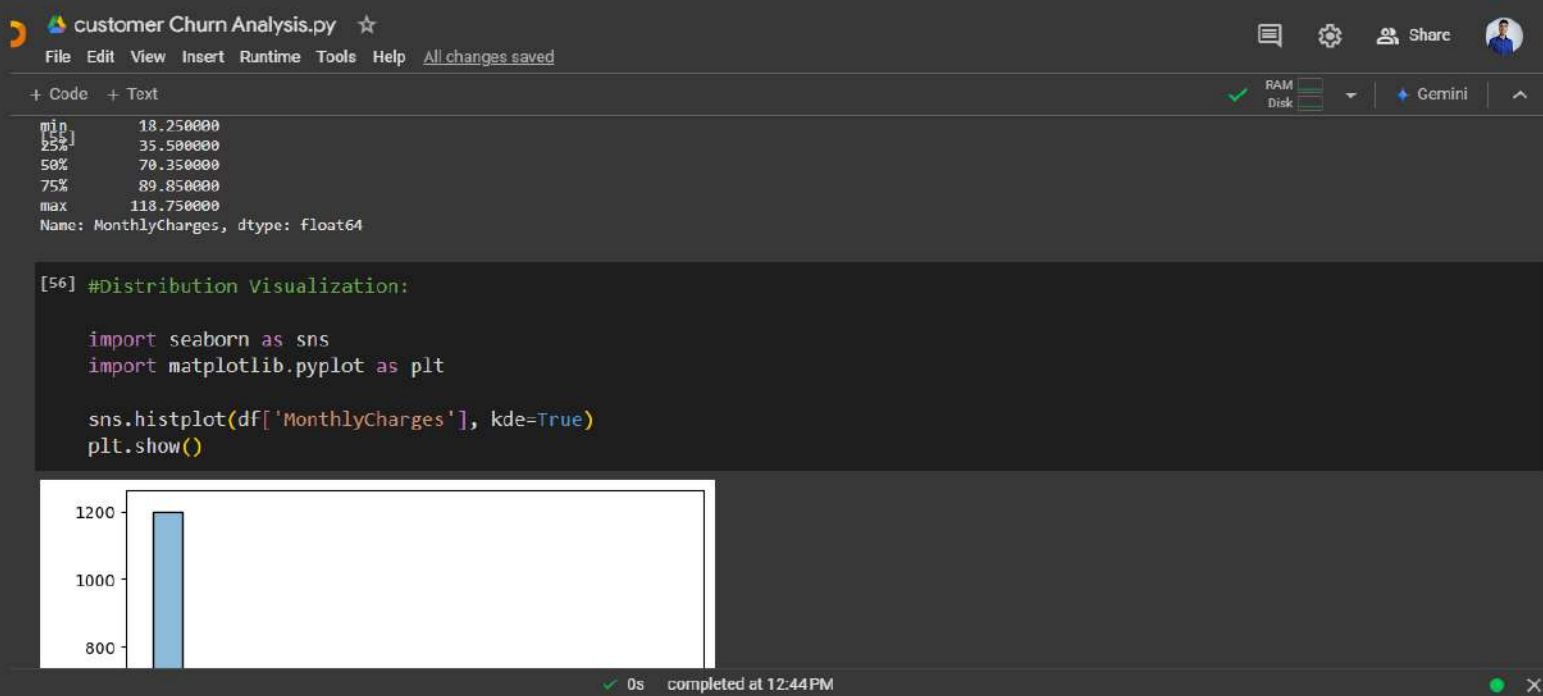
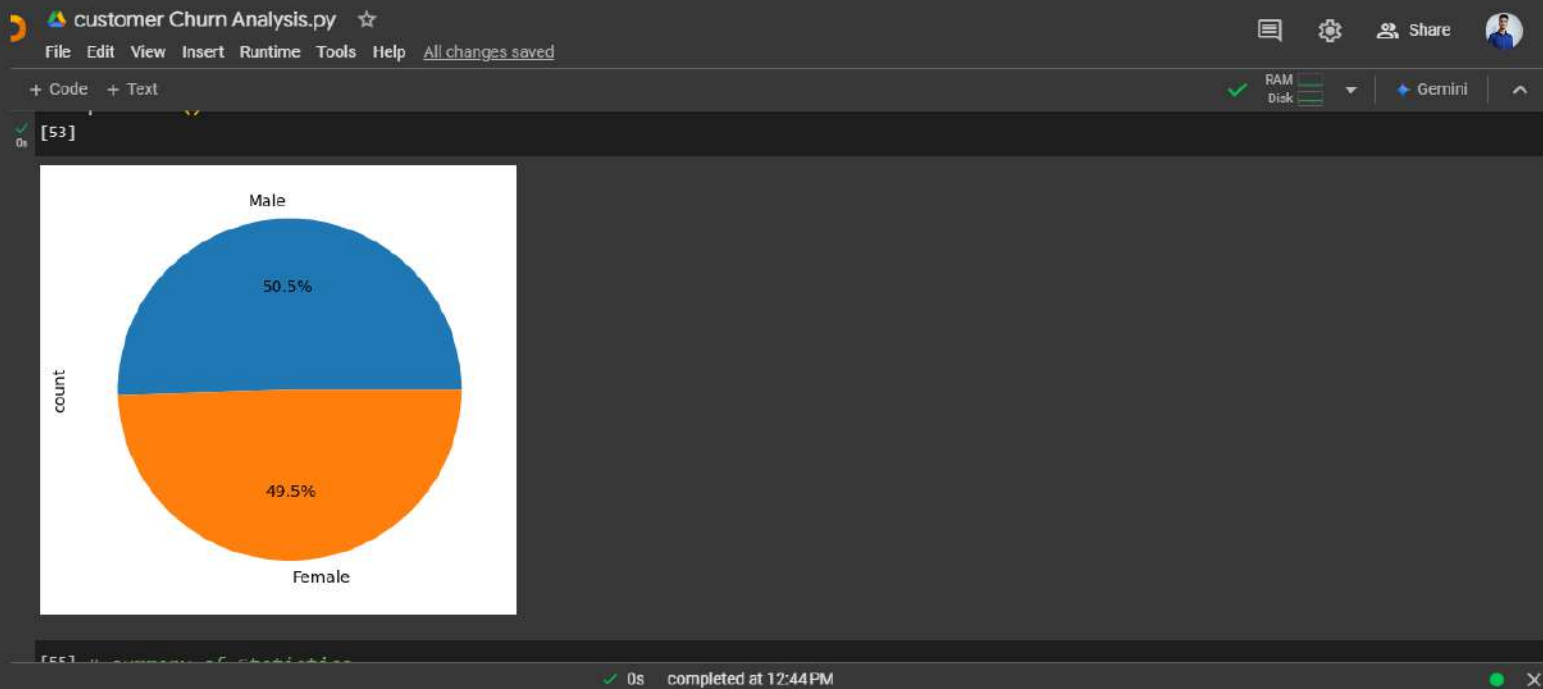
```
[49] print(df['gender'].dtype)
```

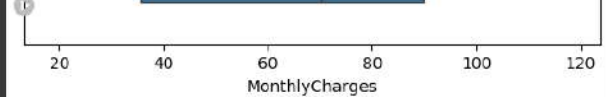
object

```
[50] # Convert 'gender' column to categorical type
df['gender'] = df['gender'].astype('category')
print(df['gender'].dtype)
```

category







```
[58] #Detecting Skewness:
```

```
print(df['MonthlyCharges'].skew())
```

```
-0.22052443394398033
```

```
[ ] Start coding or generate with AI.
```

```
0s [59] print(df['MonthlyCharges'].isnull().sum())
```

```
0
```

▼ Bia Variate Analysis

✓ 0s completed at 12:44PM

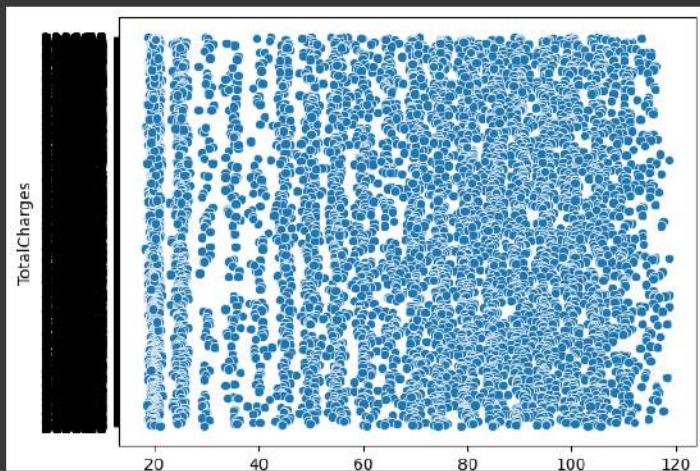
customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
sns.scatterplot(x=MonthlyCharges, y=TotalCharges, data=df)
plt.show()
```

✓ RAM Disk Gemini



✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[66] print(df['MonthlyCharges'].dtype)

float64
```

✓ RAM Disk Gemini

```
0s [68] df['MonthlyCharges'] = df['MonthlyCharges'].astype('int')
print(df['MonthlyCharges'].dtype)
```

```
int64
```

```
0s [70] print(df['TotalCharges'].dtype)
```

```
object
```

```
0s [72] print(df['TotalCharges'].dtype)
```

```
object
```

```
0s [73] # Inspect the column to identify problematic values
print(df['TotalCharges'].head()) # View the first few rows
print(df['TotalCharges'].unique()) # Check unique values
```

```
# Convert to numeric, handle errors, and fill missing values
```

✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[73]
0    29.85
1    1889.5
2     109.15
3    1848.75
4     151.65
Name: TotalCharges, dtype: object
```

✓ RAM Disk Gemini

```
['29.85' '1889.5' '108.15' ... '346.45' '306.6' '6844.5']  
int64
```

```
[74] #Correlation Coefficient: Measure the strength and direction of the relationship.
```

```
correlation = df[['MonthlyCharges', 'TotalCharges']].corr()  
print(correlation)
```

	MonthlyCharges	TotalCharges
MonthlyCharges	1.000000	0.651237
TotalCharges	0.651237	1.000000

```
[75] #Line plot
```

```
sns.lineplot(x='MonthlyCharges', y='TotalCharges', data=df)  
plt.show()
```

✓ 0s completed at 12:44PM



customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved



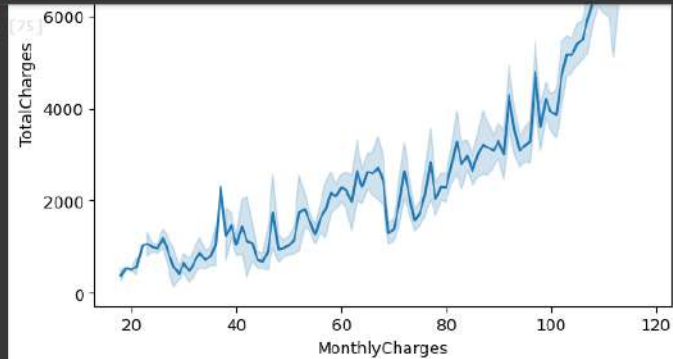
+ Code + Text



RAM
Disk



◆ Gemini



```
[77] #Key Insights: Distribution of numerical values within categories, differences in means, and variance.
```

```
sns.boxplot(x='gender', y='MonthlyCharges', data=df)  
plt.show()
```



✓ 0s completed at 12:44PM



customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved



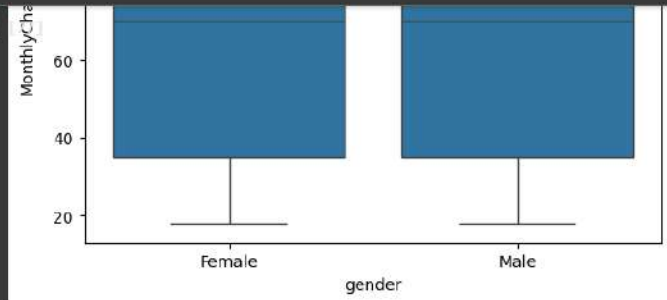
+ Code + Text



RAM
Disk



◆ Gemini



```
[78] #violinplot
```

```
sns.violinplot(x='gender', y='MonthlyCharges', data=df)  
plt.show()
```



✓ 0s completed at 12:44PM



customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved



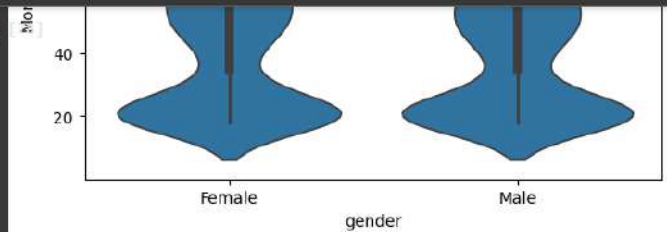
+ Code + Text



RAM
Disk



◆ Gemini



```
[79] #Bar Plot
```

```
sns.barplot(x='gender', y='MonthlyCharges', data=df, ci=None)  
plt.show()
```

<ipython-input-79-0faef4b4b2cc>:2: FutureWarning:

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

```
sns.barplot(x='gender', y='MonthlyCharges', data=df, ci=None)
```



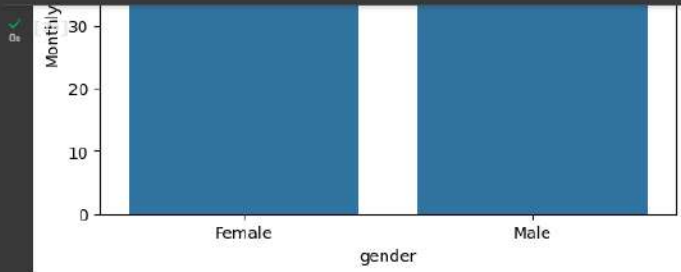
✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ RAM Disk Gemini



```
[80] #Group Statistics  
print(df.groupby('gender')['MonthlyCharges'].mean())  
print(df.groupby('gender')['MonthlyCharges'].median())
```

```
gender  
Female    64.748288  
Male     63.852039  
Name: MonthlyCharges, dtype: float64  
gender  
Female    70.0  
Male     70.0
```

✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ RAM Disk Gemini

```
[82] #Key Insights: Frequency and proportion of combinations, dependency between variables.  
contingency_table = pd.crosstab(df['gender'], df['PaymentMethod'])  
print(contingency_table)
```

```
PaymentMethod Bank transfer (automatic) Credit card (automatic) \  
gender  
Female          788          752  
Male           756          770
```

```
PaymentMethod Electronic check Mailed check  
gender  
Female          1170          778  
Male           1195          834
```

```
[83] #Stacked Bar Plot: Visualize category combinations  
contingency_table.plot(kind='bar', stacked=True)  
plt.show()
```



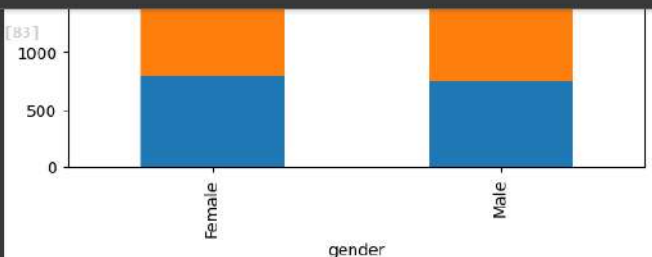
✓ 0s completed at 12:44PM

customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help All changes saved

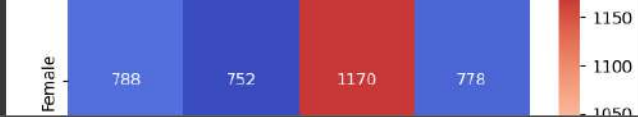
+ Code + Text

✓ RAM Disk Gemini



```
[84] #heatmap  
sns.heatmap(contingency_table, annot=True, cmap='coolwarm', fmt='d')  
plt.show()
```





✓ 0s completed at 12:44PM



customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)



Share



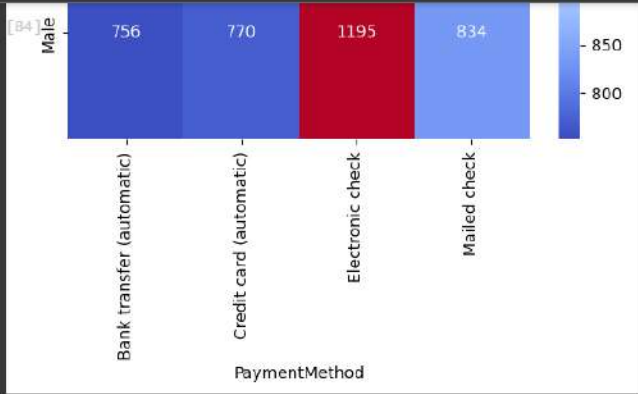
+ Code + Text



RAM
Disk



◆ Gemini



```
[85] #Chi-Square Test: Check for independence between two categorical variables
      from scipy.stats import chi2_contingency
      chi2, p, dof, expected = chi2_contingency(contingency_table)
```



customer Churn Analysis.py ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)



Share



+ Code + Text



RAM
Disk



◆ Gemini

