1.Problem Statement

```
The objective of this project is to predict customer churn based on
historical service and
demographic data. By identifying customers who are likely to leave, the
business can
take proactive measures to improve retention and reduce revenue loss.
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder,StandardScaler
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import CategoricalNB,BernoulliNB,GaussianNB
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_rep
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns',100)
pd.set_option('display.width',120)
```

2.Import data

In [3]:
```
df = pd.read_csv('C:/Users/raviy/OneDrive/Desktop/customer-churn-prediction/da
```

In [4]:
```
df.shape
```

Out[4]:
```
(7043, 21)
```

In [5]:
```
df.info()
```

```
<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  
 6   PhoneService      7043 non-null   object 
 7   MultipleLines     7043 non-null   object 
 8   InternetService   7043 non-null   object 
 9   OnlineSecurity    7043 non-null   object 
 10  OnlineBackup      7043 non-null   object 
 11  DeviceProtection  7043 non-null   object 
 12  TechSupport       7043 non-null   object 
 13  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
```

3. Exploratory Data Analysis

3.1 Categorical Data

3.1.1 Total customers

In [6]:
```
df['customerID'].nunique()
```

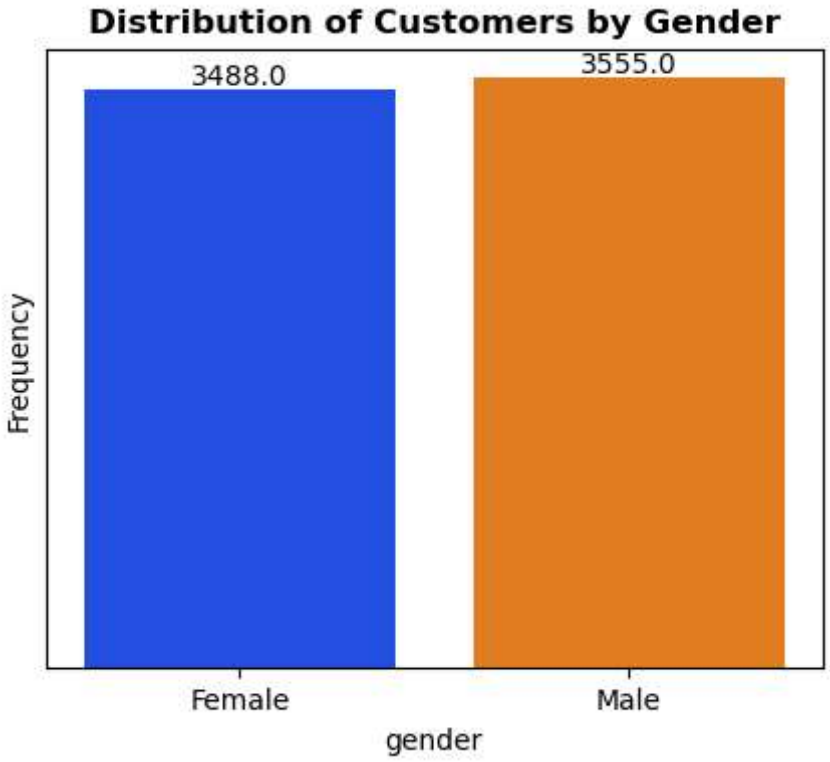
Out[6]:
```
7043
```

3.1.2 Gender

In [7]:
```
df['gender'].value_counts(dropna=False)
```

Out[7]:
```
gender
Male      3555
Female    3488
Name: count, dtype: int64
```

In [9]:
```
plt.figure(figsize=(5,4))
ax = sns.countplot(x = 'gender',data=df,palette='bright')
for p in ax.patches:
    ax.annotate(str(p.get_height()),
(p.get_x()+p.get_width()/2,p.get_height()),
    ha = 'center',va='bottom')
plt.title('Distribution of Customers by Gender',fontweight="bold")
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



The customer's gender is almost similar; there is not much difference

3.1.3 SeniorCitizen

In [10]:
```
df['SeniorCitizen'] = df['SeniorCitizen'].map({1:'Yes',0:'No'})
```

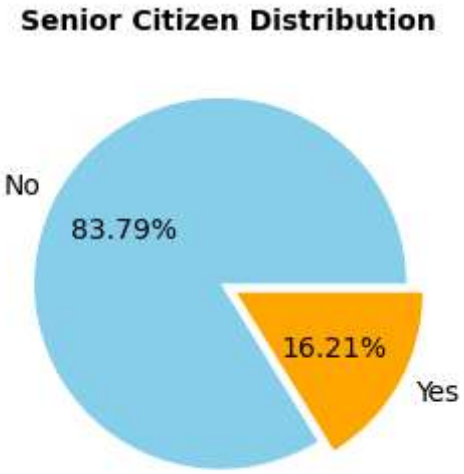
In [11]:
```
df['SeniorCitizen'].value_counts(dropna=False)
```

Out[11]:
```
SeniorCitizen
No     5901
Yes    1142
Name: count, dtype: int64
```

In [12]:
```
df['SeniorCitizen'].isnull().sum()
```

Out[12]:
```
0
```

In [14]:
```
plt.figure(figsize=(3,3))
df_s = df['SeniorCitizen'].value_counts(normalize=True,dropna=False).reset_inc
plt.pie(df_s['proportion'],labels=df_s['SeniorCitizen'],autopct='%.2f%%',
        explode=(0.05, 0.05),colors=['skyblue', 'orange'])
plt.title("Senior Citizen Distribution",size=10,pad=10,fontweight="bold")
plt.show()
```



3.1.4 Partner

In [15]:
```
df['Partner'].value_counts(dropna=False)
```

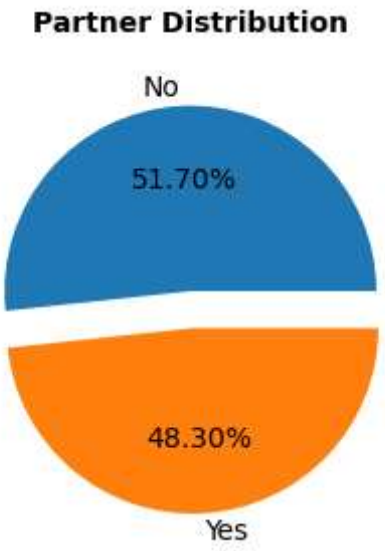
Out[15]:
```
Partner
No     3641
Yes    3402
Name: count, dtype: int64
```

In [16]:
```
df_p = df['Partner'].value_counts(normalize=True).reset_index()
df_p
```

Out[16]:

	Partner	proportion
0	No	0.516967
1	Yes	0.483033

In [23]:
```
plt.figure(figsize=(3,3))
plt.pie(df_p['proportion'],labels=df_p['Partner'],autopct='%.2f%%',explode=[0,
plt.title("Partner Distribution",size=10,pad=10,fontweight="bold")
plt.show()
```



3.1.5 Dependents

In [25]:
```
df['Dependents'].value_counts(dropna=False)
```

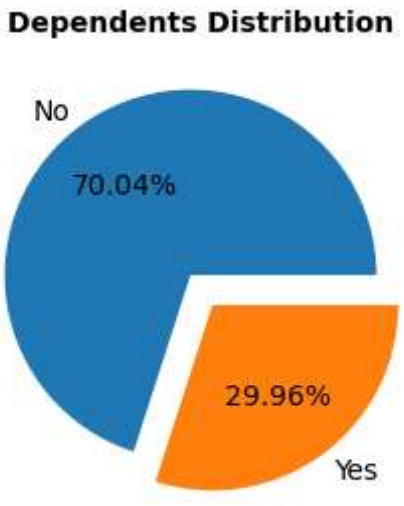
Out[25]:
```
Dependents
No     4933
Yes    2110
Name: count, dtype: int64
```

In [26]:
```
df['Dependents'].value_counts(normalize=True).reset_index()
```

Out[26]:

	Dependents	proportion
0	No	0.700412
1	Yes	0.299588

In [28]:
```
df_d = df['Dependents'].value_counts(normalize=True,dropna=False).reset_index(
plt.figure(figsize=(3,3))
plt.pie(df_d['proportion'],labels=df_d['Dependents'],autopct='%.2f%%',explode=
plt.title('Dependents Distribution',size=10,pad=10,fontweight="bold")
plt.show()
```



3.1.6 PhoneService

In [29]:
```
df['PhoneService'].value_counts()
```

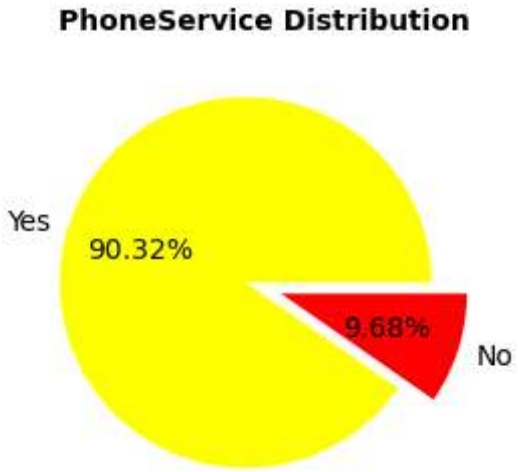
Out[29]:
```
PhoneService
Yes    6361
No      682
Name: count, dtype: int64
```

In [34]:
```
df['PhoneService'].value_counts(normalize=True).reset_index()
```

Out[34]:

	PhoneService	proportion
0	Yes	0.903166
1	No	0.096834

In [36]:
```
df_p=df['PhoneService'].value_counts(normalize=True).reset_index()
plt.figure(figsize=(3,3))
plt.pie(df_p['proportion'],labels=df_p['PhoneService'],autopct='%.2f%%',
        explode=[0.1,0.1],colors=['yellow','red'])
plt.title('PhoneService Distribution',size=10,pad=10,fontweight='bold')
plt.show()
```



3.1.7 MultipleLines

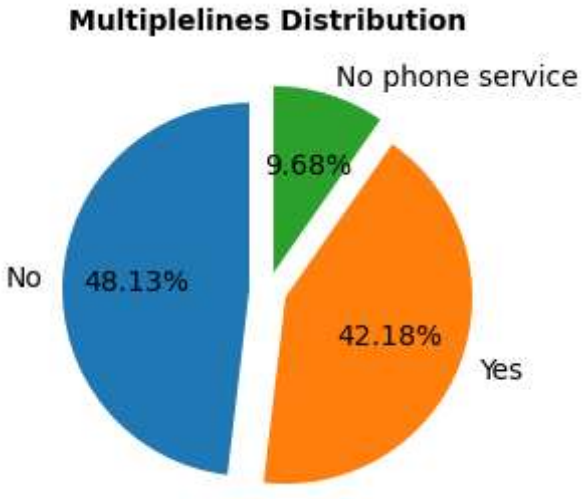
In [37]:
```
df['MultipleLines'].value_counts(dropna=False)
```

Out[37]:
```
MultipleLines
No                  3390
Yes                 2971
No phone service     682
Name: count, dtype: int64
```

In [38]:
```
df['MultipleLines'].value_counts(dropna=False,normalize=True)
```

Out[38]:
```
MultipleLines
No                  0.481329
Yes                 0.421837
No phone service    0.096834
Name: proportion, dtype: float64
```

In [46]:
```
plt.figure(figsize=(3,4))
df_m = df['MultipleLines'].value_counts(dropna=False,normalize=True).reset_inc
plt.pie(df_m['proportion'],labels=df_m['MultipleLines'],autopct='%.2f%%',explc
startangle=90)
plt.title('Multiplelines Distribution',pad=10,size=10,fontweight='bold')
plt.show()
```



3.1.8 InternetService

In [47]:
```
df['InternetService'].value_counts(dropna=False)
```

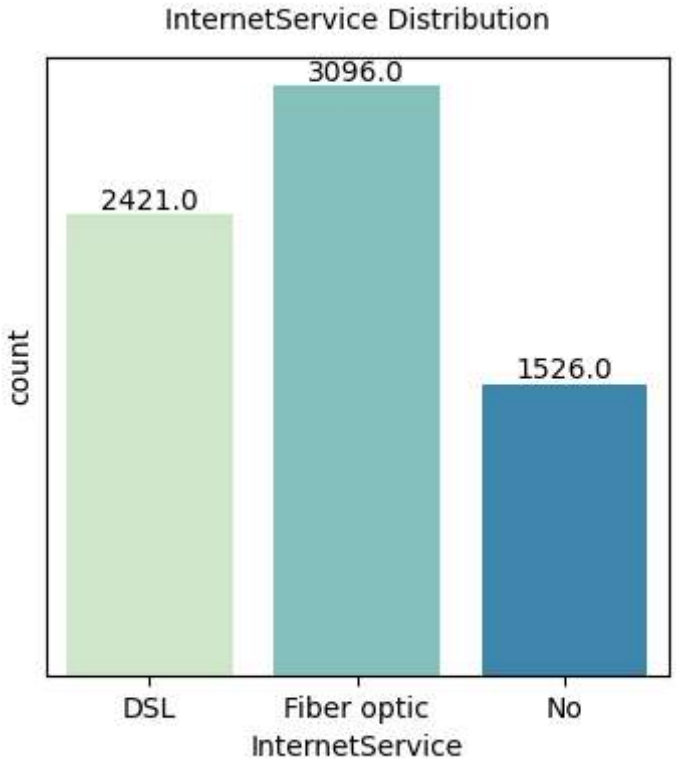
Out[47]:
```
InternetService
Fiber optic    3096
DSL            2421
No             1526
Name: count, dtype: int64
```

In [49]:
```
df_i = df['InternetService'].value_counts(dropna=False,normalize=True).reset_i
df_i
```

Out[49]:

	InternetService	proportion
0	Fiber optic	0.439585
1	DSL	0.343746
2	No	0.216669

In [50]:
```
plt.figure(figsize=(4,4))
ax = sns.countplot(x=df['InternetService'],data=df,palette='GnBu')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha = 'center',va='bottom'
)
plt.title('InternetService Distribution',pad=10,size=10)
plt.yticks([])
plt.show()
```

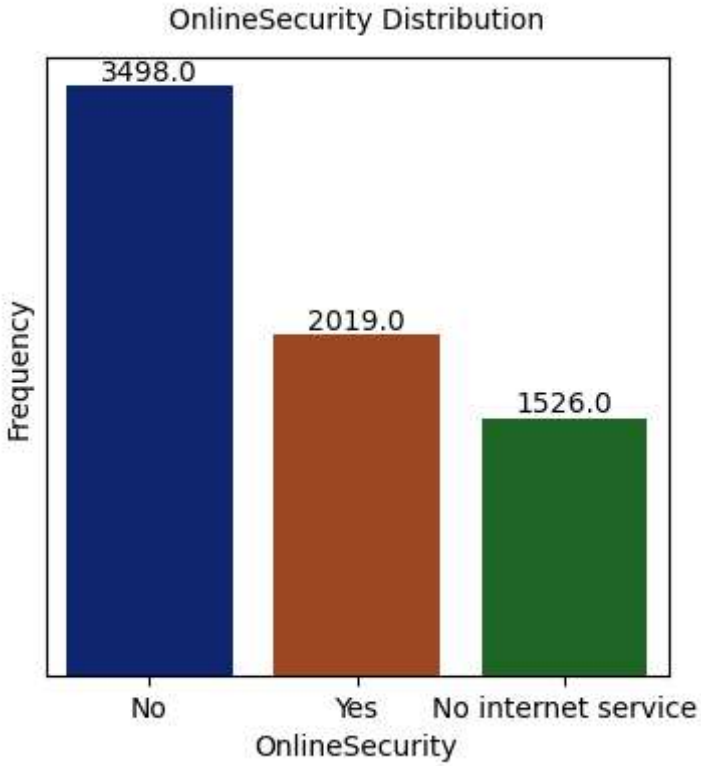


3.1.9 OnlineSecurity

In [51]:
```
df['OnlineSecurity'].value_counts(dropna=False)
```

Out[51]:
```
OnlineSecurity
No                     3498
Yes                    2019
No internet service    1526
Name: count, dtype: int64
```

In [52]:
```
plt.figure(figsize=(4,4))
ax = sns.countplot(x='OnlineSecurity',data=df,palette='dark')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.yticks([])
plt.ylabel('Frequency')
plt.title('OnlineSecurity Distribution',pad=10,size=10)
plt.show()
```

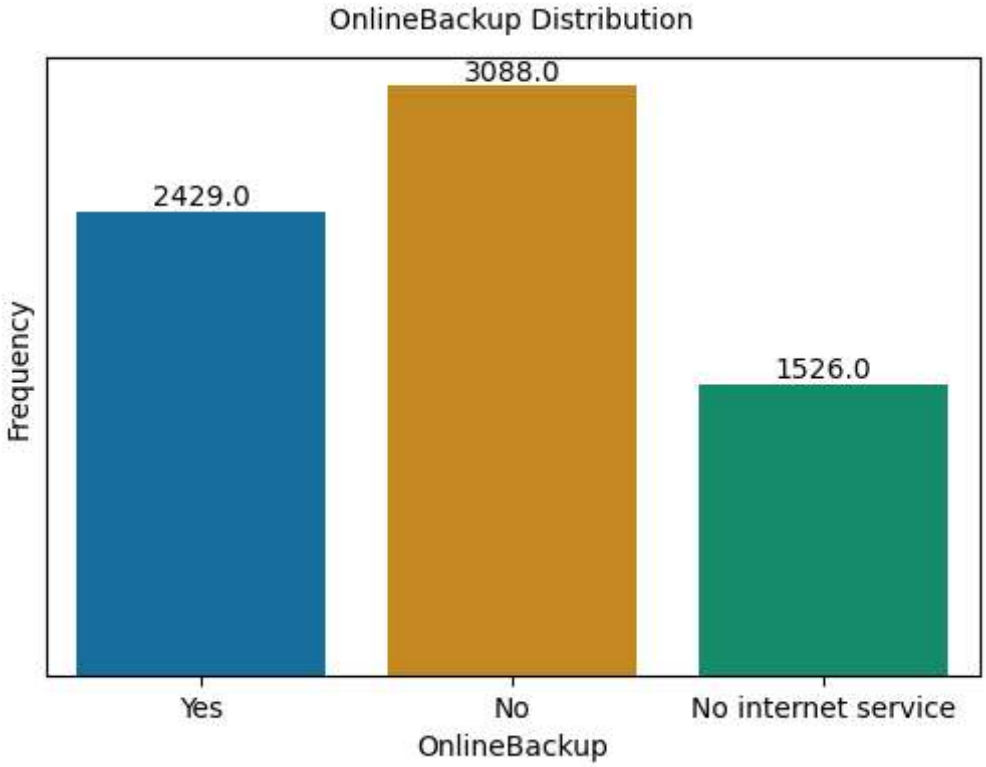


3.1.10 OnlineBackup

In [53]:
```
df['OnlineBackup'].value_counts(dropna=False)
```

Out[53]:
```
OnlineBackup
No                     3088
Yes                    2429
No internet service    1526
Name: count, dtype: int64
```

In [55]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x = 'OnlineBackup',data=df,palette='colorblind',orient='y',
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('OnlineBackup Distribution',size=10,pad=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```

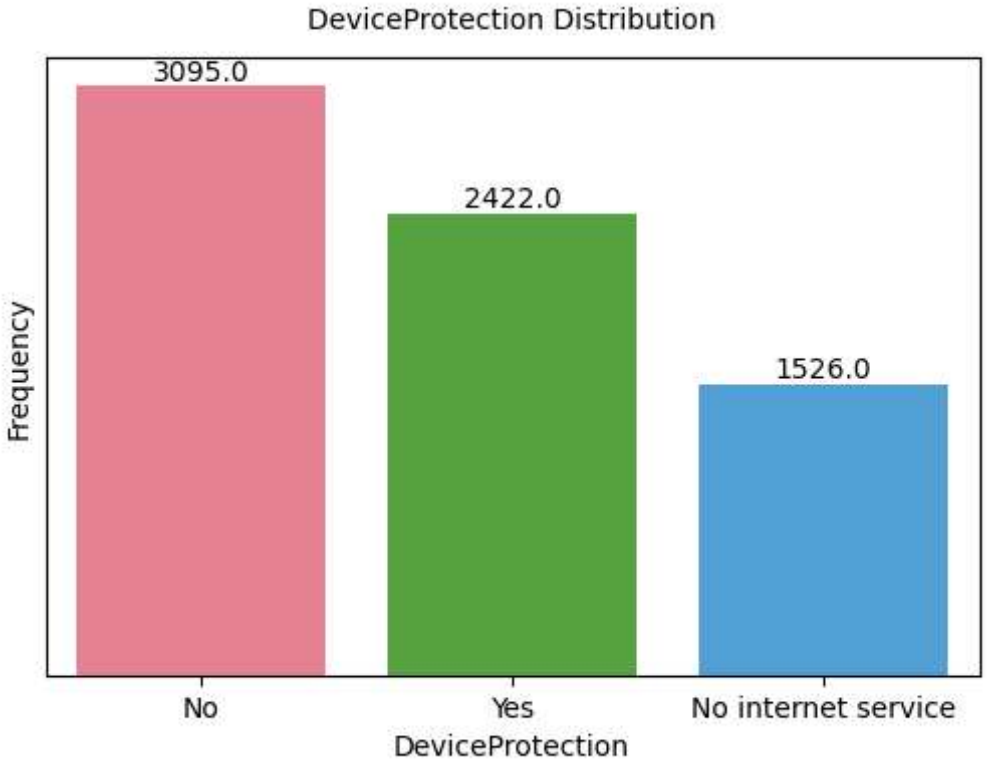


3.1.11 DeviceProtection

In [56]:
```
df['DeviceProtection'].value_counts(dropna=False)
```

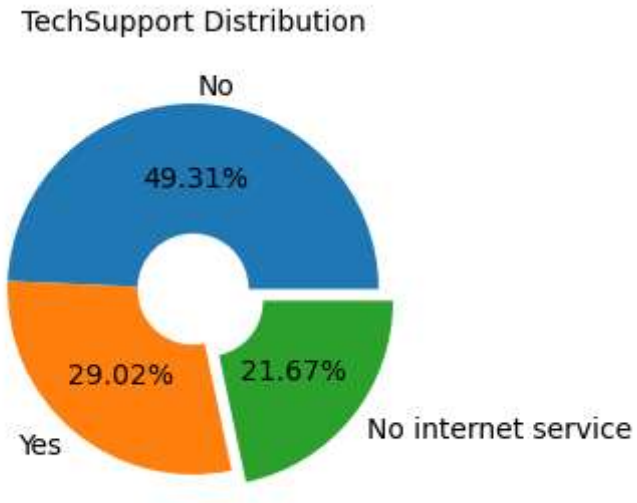
Out[56]:
```
DeviceProtection
No                     3095
Yes                    2422
No internet service    1526
Name: count, dtype: int64
```

In [58]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x ='DeviceProtection',data=df,palette='husl')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('DeviceProtection Distribution',pad=10,size=10)
plt.ylabel('Frequency')
plt.yticks([])
plt.show()
```



3.1.12 TechSupport

In [61]:
```
plt.figure(figsize=(3,3))
counts = df['TechSupport'].value_counts()
plt.pie(counts, labels=counts.index, autopct='%.2f%%', wedgeprops={'width':0.7
plt.title("TechSupport Distribution",pad=10,size=10)
plt.show()
```

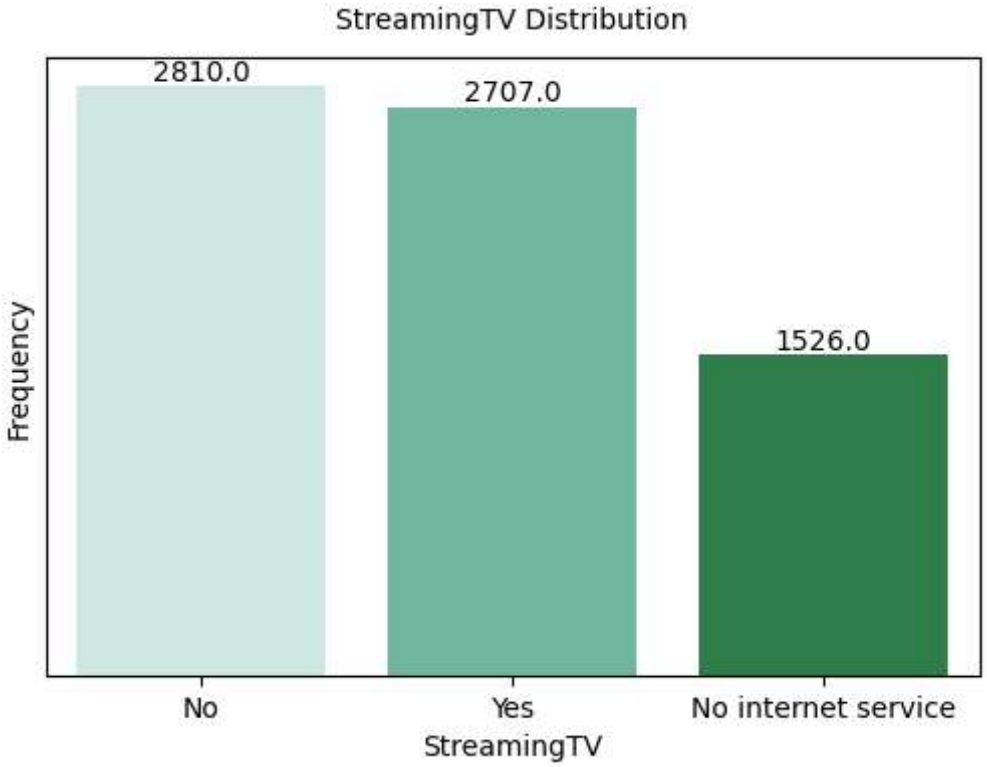


3.1.13 StreamingTV

In [62]:
```
df['StreamingTV'].value_counts()
```

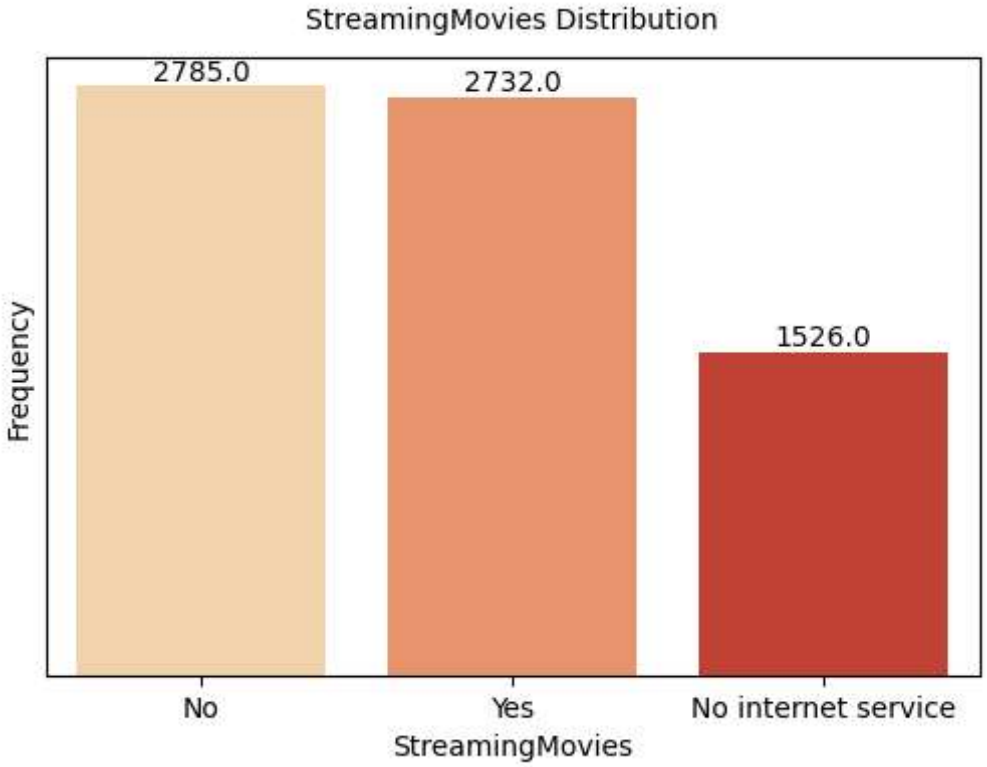
Out[62]:
```
StreamingTV
No                     2810
Yes                    2707
No internet service    1526
Name: count, dtype: int64
```

In [64]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x='StreamingTV',data=df,palette='BuGn')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('StreamingTV Distribution',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



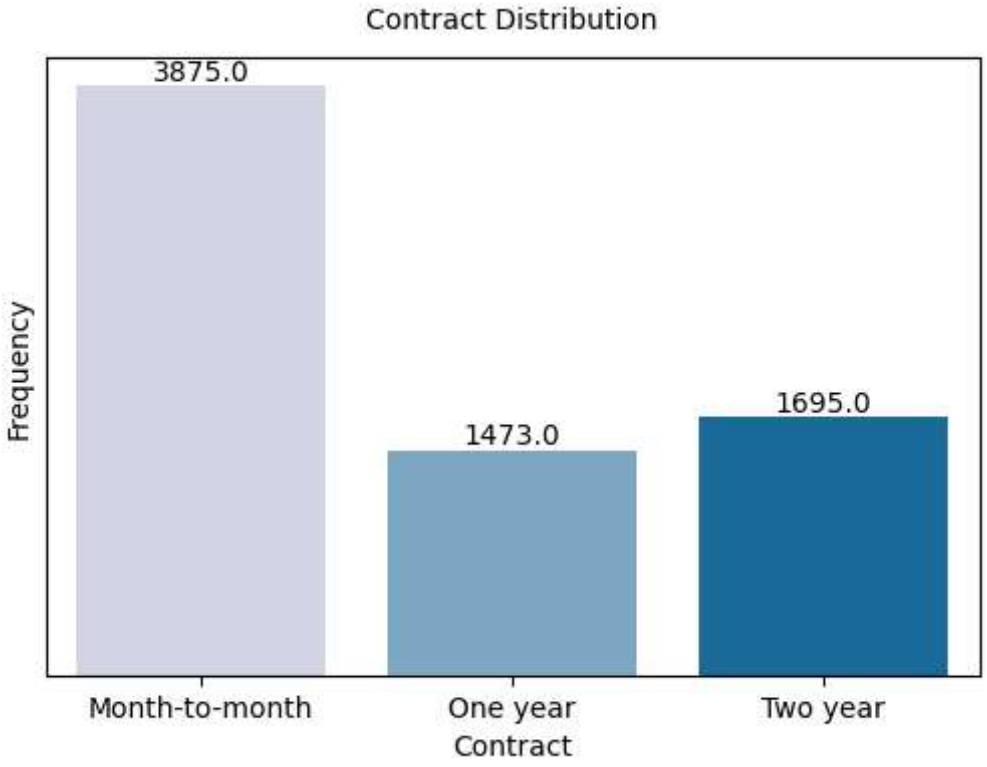
3.1.13 StreamingMovies

In [66]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x = 'StreamingMovies',data=df,palette='OrRd')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('StreamingMovies Distribution',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



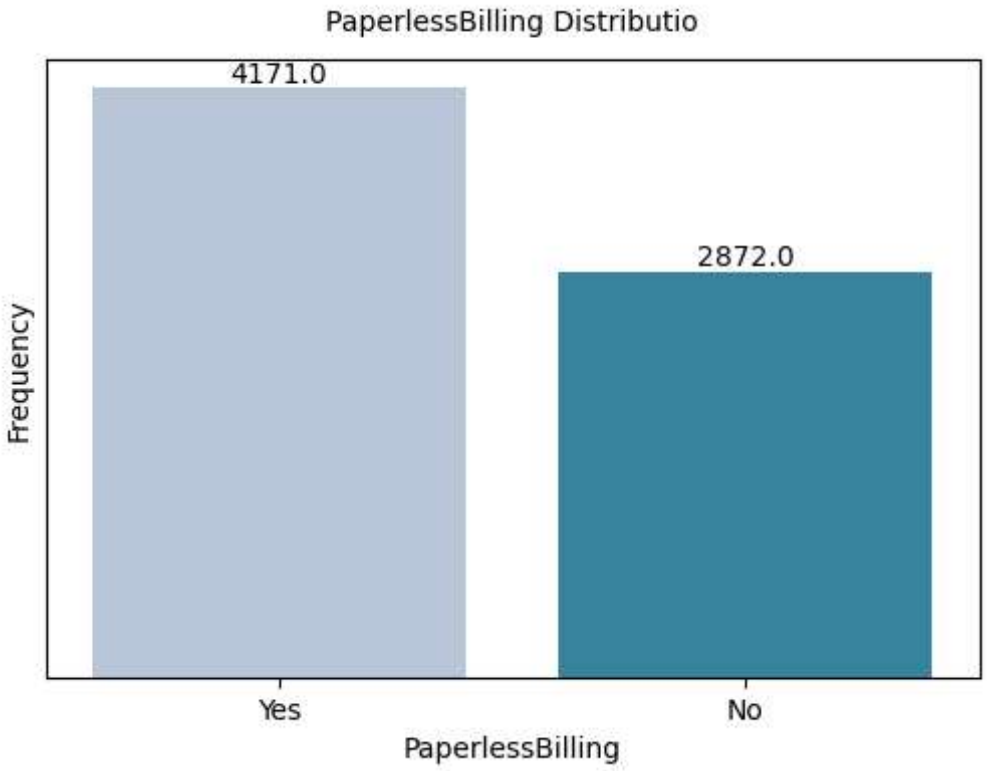
3.1.13 Contract

In [67]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x = 'Contract',data=df,palette='PuBu')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('Contract Distribution',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



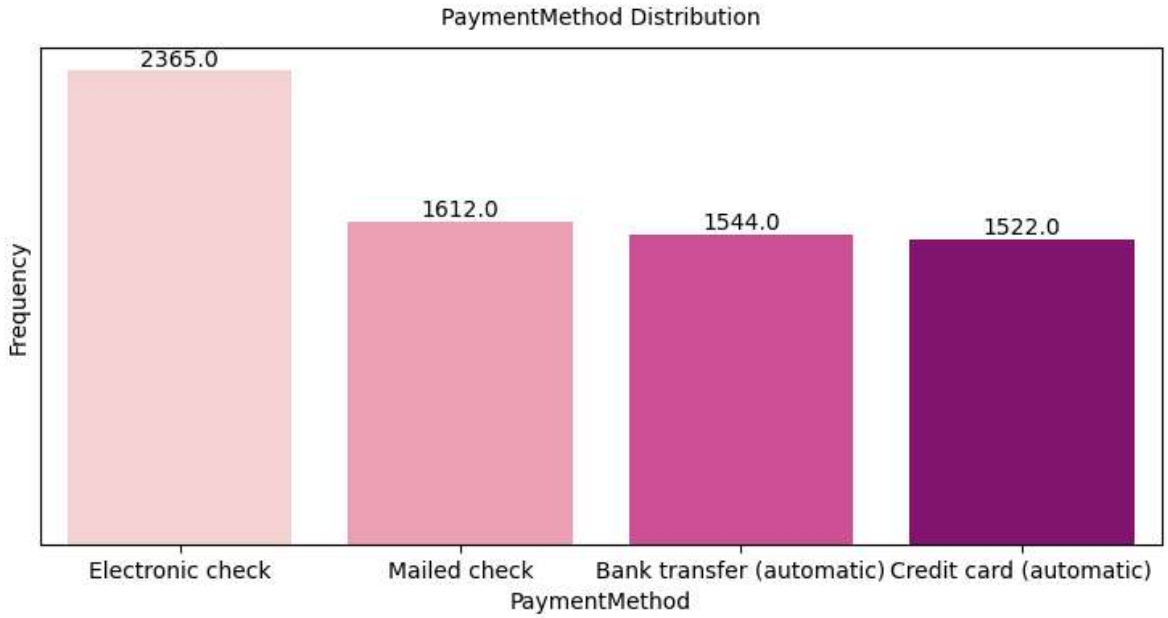
3.1.14 PaperlessBilling

In [68]:
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x = 'PaperlessBilling',data=df,palette='PuBuGn')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('PaperlessBilling Distributio',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



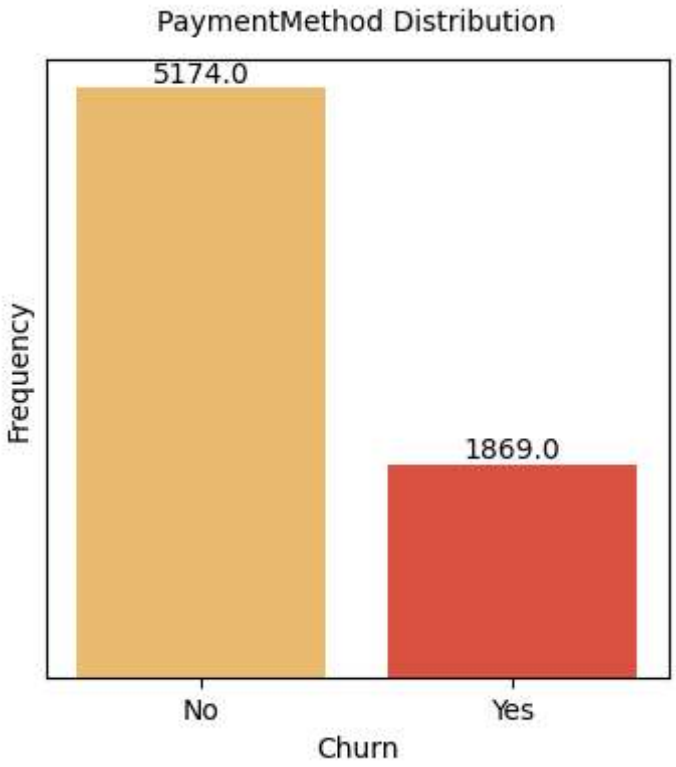
3.1.15 PaymentMethod

In [69]:
```
plt.figure(figsize=(9,4))
ax = sns.countplot(x = 'PaymentMethod',data=df,palette='RdPu')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('PaymentMethod Distribution',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



3.1.16 Churn

In [70]:
```
plt.figure(figsize=(4,4))
ax = sns.countplot(x = 'Churn',data=df,palette='YlOrRd')
for i in ax.patches:
    ax.annotate(
str(i.get_height()),
(i.get_x()+i.get_width()/2,i.get_height()),
ha='center',va='bottom')
plt.title('PaymentMethod Distribution',pad=10,size=10)
plt.yticks([])
plt.ylabel('Frequency')
plt.show()
```



3.2 Numerical values

3.2.1 tenure

In [71]:
```
print('Max Tenure in Months:',np.max(df['tenure']))
```

```
Max Tenure in Months: 72
```

In [5]:
```
print('Min Tenure in Months:',np.min(df['tenure']))
```

```
Min Tenure in Months: 0
```

In [6]:
```
df.drop(labels=df[df['tenure'] == 0].index, axis=0, inplace=True)
df[df['tenure'] == 0].index
```

Out[6]:
```
Index([], dtype='int64')
```

In [7]:
```
df.describe()
```

Out[7]:

	SeniorCitizen	tenure	MonthlyCharges
count	7032.000000	7032.000000	7032.000000
mean	0.162400	32.421786	64.798208
std	0.368844	24.545260	30.085974
min	0.000000	1.000000	18.250000
25%	0.000000	9.000000	35.587500
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.862500
max	1.000000	72.000000	118.750000

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 7032 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   customerID        7032 non-null   object 
 1   gender            7032 non-null   object 
 2   SeniorCitizen     7032 non-null   int64  
 3   Partner           7032 non-null   object 
 4   Dependents        7032 non-null   object 
 5   tenure            7032 non-null   int64  
 6   PhoneService      7032 non-null   object 
 7   MultipleLines     7032 non-null   object 
 8   InternetService   7032 non-null   object 
 9   OnlineSecurity    7032 non-null   object 
 10  OnlineBackup      7032 non-null   object 
 11  DeviceProtection  7032 non-null   object 
 12  TechSupport       7032 non-null   object 
 13  StreamingTV       7032 non-null   object 
 14  StreamingMovies   7032 non-null   object 
 15  Contract          7032 non-null   object 
 16  PaperlessBilling  7032 non-null   object 
 17  PaymentMethod     7032 non-null   object 
 18  MonthlyCharges    7032 non-null   float64
 19  TotalCharges      7032 non-null   object 
 20  Churn             7032 non-null   object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.2+ MB
```

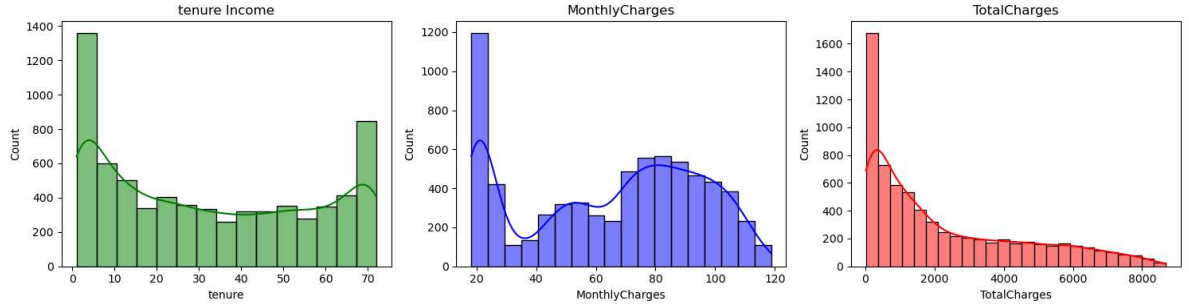
In [9]:
```
df['TotalCharges'] = df['TotalCharges'].astype('float')
```

In [10]:
```
numerical_cols = ['tenure', 'MonthlyCharges', 'TotalCharges']
df[numerical_cols].describe()
```

Out[10]:

	tenure	MonthlyCharges	TotalCharges
count	7032.000000	7032.000000	7032.000000
mean	32.421786	64.798208	2283.300441
std	24.545260	30.085974	2266.771362
min	1.000000	18.250000	18.800000
25%	9.000000	35.587500	401.450000
50%	29.000000	70.350000	1397.475000
75%	55.000000	89.862500	3794.737500
max	72.000000	118.750000	8684.800000

In [11]:
```
fig, ax = plt.subplots(1, 3, figsize=(15, 4))
# Plot histograms
sns.histplot(data=df, x="tenure", kde=True, ax=ax[0], color='green')
ax[0].set_title("tenure Income")
sns.histplot(data=df, x="MonthlyCharges", kde=True, ax=ax[1], color='blue')
ax[1].set_title("MonthlyCharges")
sns.histplot(data=df, x="TotalCharges", kde=True, ax=ax[2], color='red')
ax[2].set_title("TotalCharges")
plt.tight_layout()
plt.show()
```

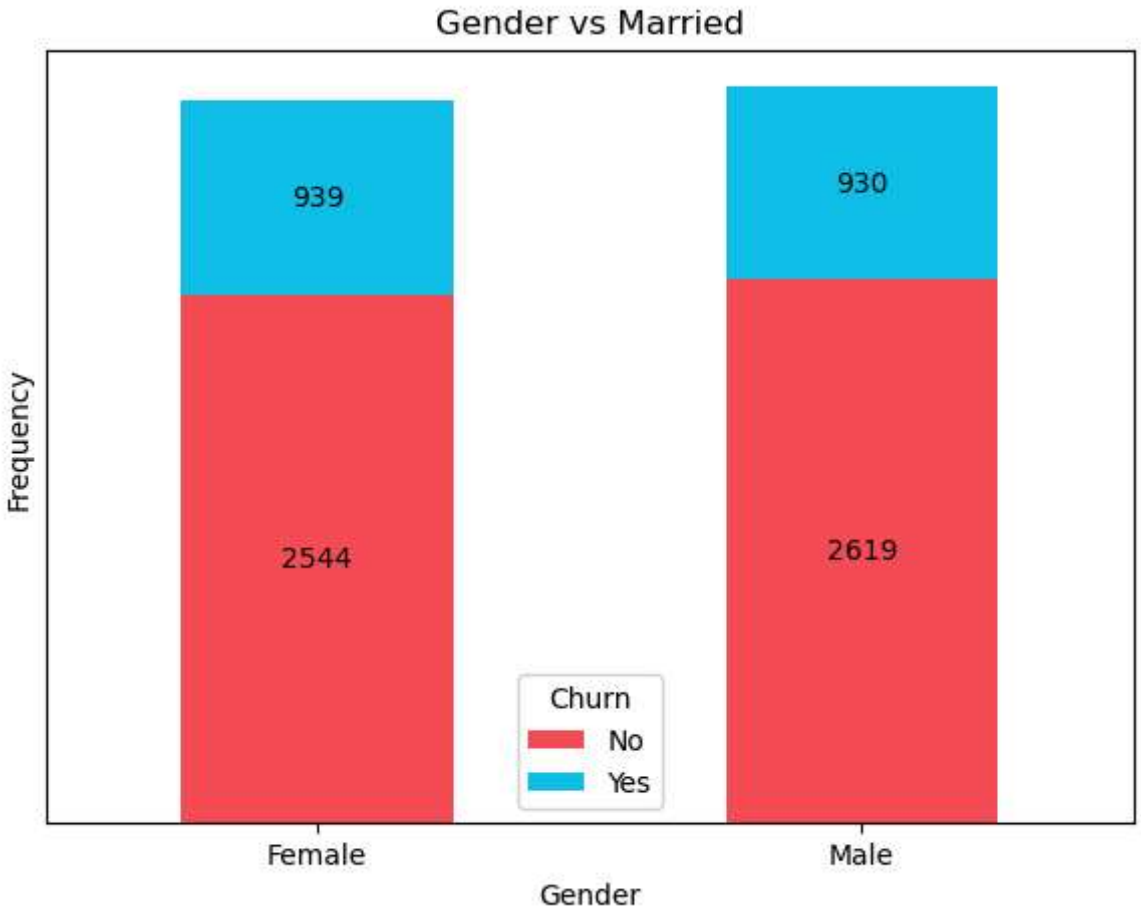


4 Categorical vs Categorical

4.1 Gender vs Married

In [13]:
```
plt.figure(figsize=(4,4))
ct = pd.crosstab(df.gender, df.Churn)
ax = ct.plot(kind='bar', stacked=True, figsize=(7,5),color=['#f64f59','#12c2e9
plt.title('Gender vs Married')
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.xticks(rotation=0)
for bar in ax.patches:
    width = bar.get_width()
    height = bar.get_height()
    x = bar.get_x()
    y = bar.get_y()
    if height > 0: # avoid labeling empty bars
        ax.text(x + width/2, y + height/2, str(int(height)),
        ha='center', va='center', color='black', fontsize=10)
plt.yticks([])
plt.show()
```

```
<Figure size 400x400 with 0 Axes>
```



In [ ]:
```
```