In [1]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn import linear_model, preprocessing, metrics
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split, cross_val_score
from statsmodels.formula.api import logit;
from platform import python_version
```

In [ ]:
```
df = pd.read_csv('churn_clean.csv')
```

In [3]:
```
dfr = df[['Churn', 'Port_modem', 'Tablet', 'Phone', 'PaperlessBilling', 'InternetServi
```

In [4]:
```
dfr.isna().sum()
```

Out[4]:
```
Churn              0
Port_modem         0
Tablet             0
Phone              0
PaperlessBilling   0
InternetService    0
Techie             0
PaymentMethod      0
dtype: int64
```

In [5]:
```
dfr.describe()
```

Out[5]:

	Churn	Port_modem	Tablet	Phone	PaperlessBilling	InternetService	Techie	PaymentMethod
count	10000	10000	10000	10000	10000	10000	10000	10000
unique	2	2	2	2	2	3	2	4
top	No	No	No	Yes	Yes	Fiber Optic	No	Electronic Check
freq	7350	5166	7009	9067	5882	4408	8321	3398

In [6]:
```
#create dummies
dfr = pd.get_dummies(dfr, drop_first=True)
dfr.sample(3)
```

Out[6]:

	Churn_Yes	Port_modem_Yes	Tablet_Yes	Phone_Yes	PaperlessBilling_Yes	InternetService_Fiber Optic
3191	1	1	0	1	0	0
8146	0	0	1	1	0	0
6600	0	0	0	1	0	1

In [7]:
```
dfr = dfr.rename(columns={'InternetService_Fiber Optic': 'FiberOptic', 'PaymentMethod_
```

In [8]:
```
dfr.describe()
```

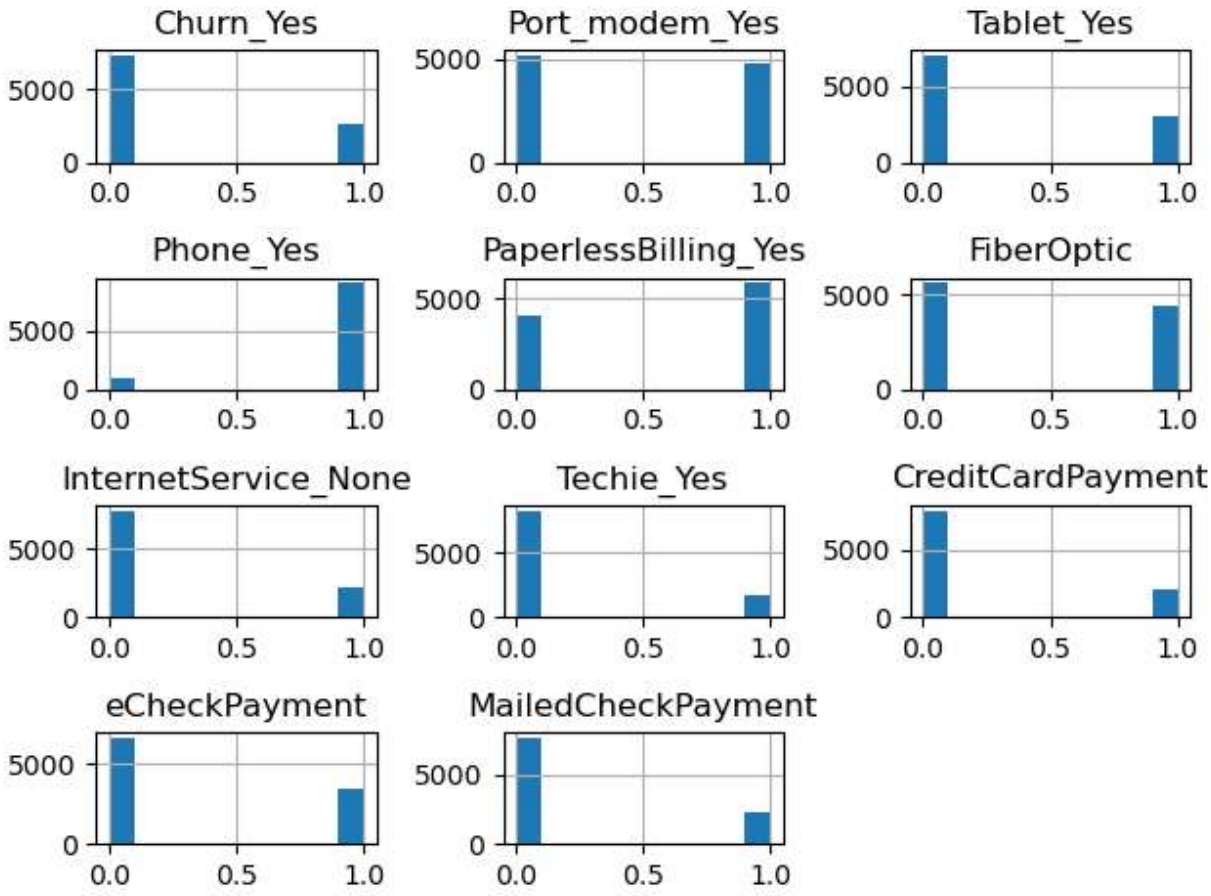
Out[8]:

	Churn_Yes	Port_modem_Yes	Tablet_Yes	Phone_Yes	PaperlessBilling_Yes	FiberOptic
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	0.265000	0.483400	0.299100	0.906700	0.588200	0.440800
std	0.441355	0.499749	0.457887	0.290867	0.492184	0.496508
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	1.000000	1.000000	0.000000
75%	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

In [9]:
```
dfr.info()
```

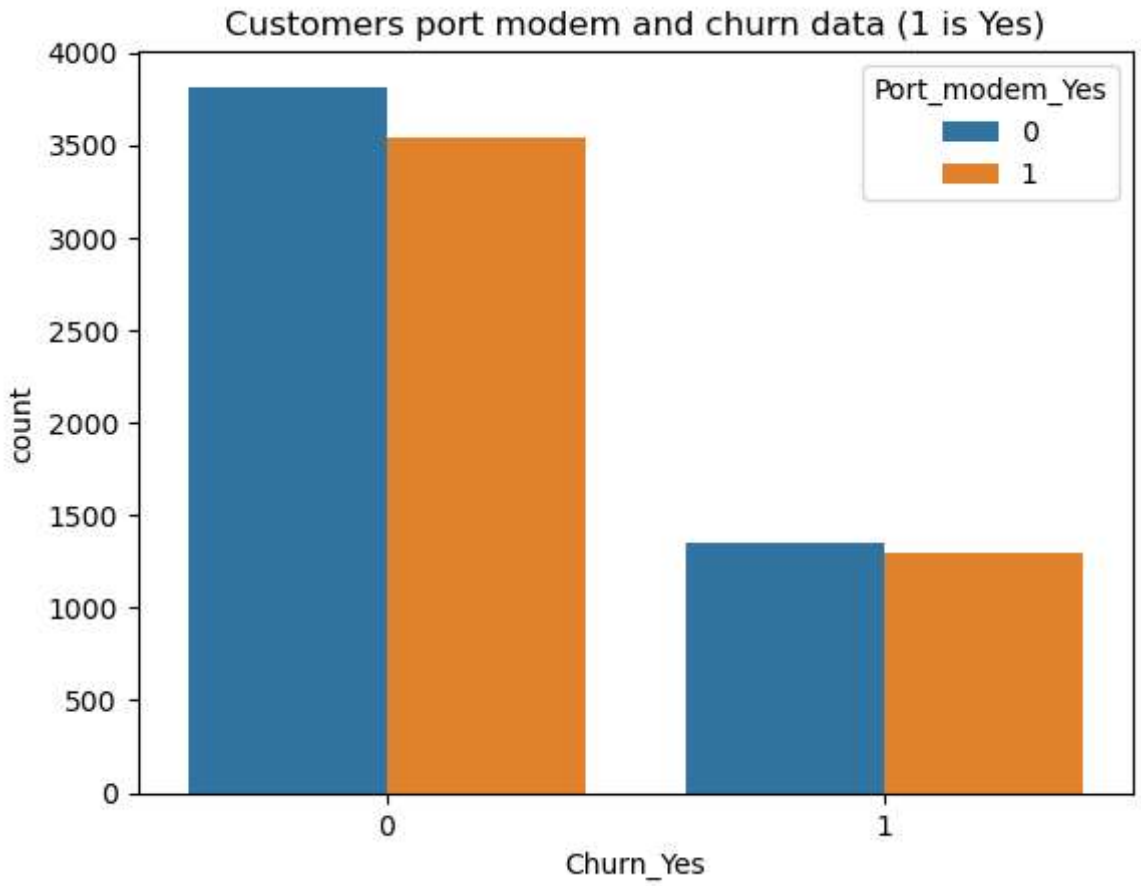
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Churn_Yes             10000 non-null  uint8
 1   Port_modem_Yes        10000 non-null  uint8
 2   Tablet_Yes            10000 non-null  uint8
 3   Phone_Yes             10000 non-null  uint8
 4   PaperlessBilling_Yes  10000 non-null  uint8
 5   FiberOptic            10000 non-null  uint8
 6   InternetService_None  10000 non-null  uint8
 7   Techie_Yes            10000 non-null  uint8
 8   CreditCardPayment     10000 non-null  uint8
 9   eCheckPayment         10000 non-null  uint8
 10  MailedCheckPayment    10000 non-null  uint8
dtypes: uint8(11)
memory usage: 107.5 KB
```

In [10]:
```
dfr.hist()
plt.tight_layout()
```



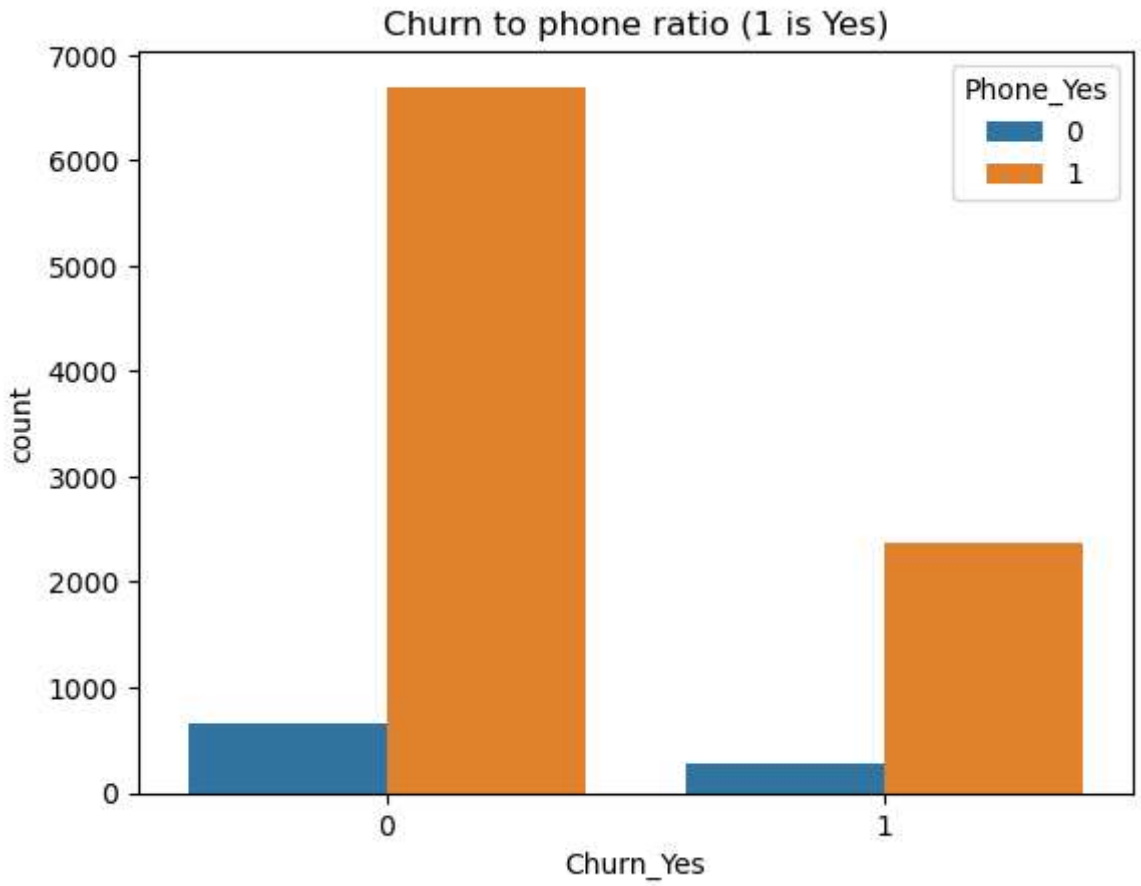
In [11]:
```
sns.countplot(x='Churn_Yes',hue='Port_modem_Yes',data=dfr).set(title='Customers port m
```

Out[11]:
```
[Text(0.5, 1.0, 'Customers port modem and churn data (1 is Yes)')]
```



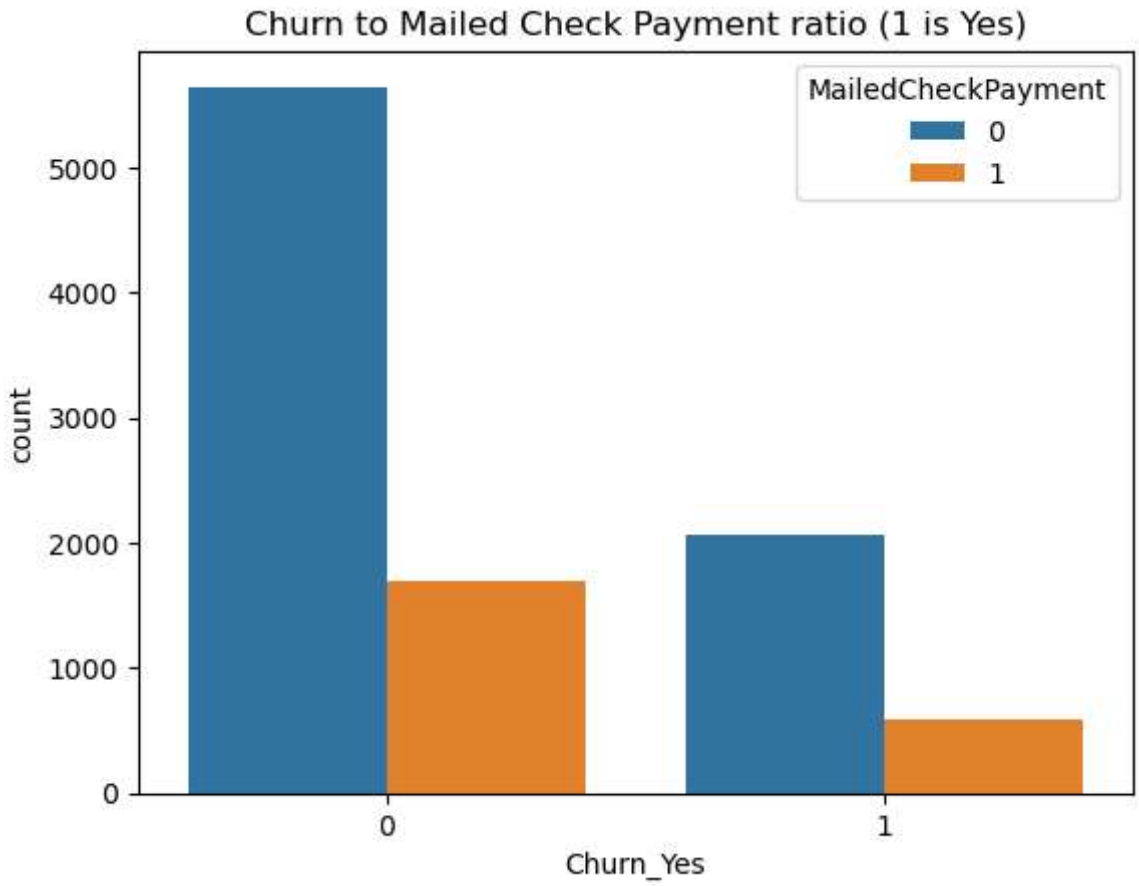
In [12]:
```
sns.countplot(x='Churn_Yes',hue='Phone_Yes',data=dfr).set(title='Churn to phone ratio
```

Out[12]:
```
[Text(0.5, 1.0, 'Churn to phone ratio (1 is Yes)')]
```



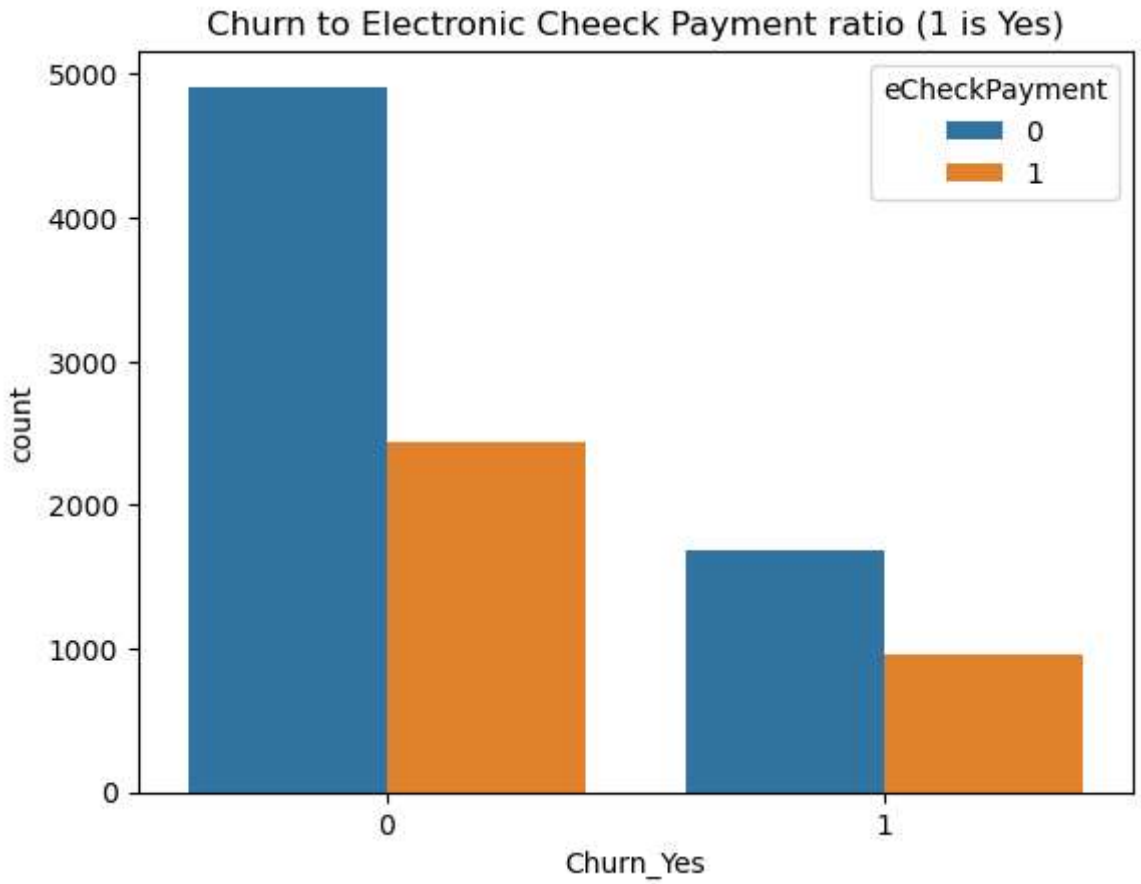
In [13]:
```
sns.countplot(x='Churn_Yes',hue='MailedCheckPayment',data=dfr).set(title='Churn to Mai
```

Out[13]:
```
[Text(0.5, 1.0, 'Churn to Mailed Check Payment ratio (1 is Yes)')]
```



In [14]:
```
sns.countplot(x='Churn_Yes',hue='eCheckPayment',data=dfr).set(title='Churn to Electror
```

Out[14]:
```
[Text(0.5, 1.0, 'Churn to Electronic Cheeck Payment ratio (1 is Yes)')]
```



In [15]:
```
dfr.to_excel('LogReg_Churn_clean.xlsx')
dfr.head(3)
```

Out[15]:

	Churn_Yes	Port_modem_Yes	Tablet_Yes	Phone_Yes	PaperlessBilling_Yes	FiberOptic	InternetServic
0	0	1	1	1	1	1	
1	1	0	1	1	1	1	
2	0	1	0	1	1	0	

In [16]:
```
modl = logit("Churn_Yes ~ Phone_Yes + Port_modem_Yes + Tablet_Yes + Phone_Yes + Paperl
```

```
Optimization terminated successfully.
         Current function value: 0.570864
         Iterations 5
```

In [17]:
```
print(modl.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:              Churn_Yes   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9989
Method:                           MLE   Df Model:                           10
Date:                Mon, 16 Jan 2023   Pseudo R-squ.:                0.01272
Time:                        12:56:28   Log-Likelihood:                -5708.6
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                 1.434e-26
==============================================================================
===
                           coef    std err          z      P>|z|      [0.025      0.9
75]
------------------------------------------------------------------------------
---
Intercept               -0.7549      0.097     -7.801      0.000      -0.945      -0.
565
Phone_Yes               -0.1999      0.076     -2.635      0.008      -0.349      -0.
051
Port_modem_Yes           0.0441      0.046      0.964      0.335      -0.045       0.
134
Tablet_Yes              -0.0191      0.050     -0.383      0.702      -0.117       0.
079
PaperlessBilling_Yes     0.0389      0.046      0.837      0.403      -0.052       0.
130
FiberOptic              -0.4327      0.051     -8.484      0.000      -0.533      -0.
333
InternetService_None    -0.4479      0.063     -7.101      0.000      -0.572      -0.
324
Techie_Yes               0.3864      0.058      6.655      0.000       0.273       0.
500
CreditCardPayment        0.0561      0.070      0.796      0.426      -0.082       0.
194
eCheckPayment            0.1777      0.062      2.846      0.004       0.055       0.
300
MailedCheckPayment       0.0438      0.069      0.635      0.525      -0.091       0.
179
==============================================================================
===
```

In [18]:
```
#reduced model
modl2 = logit("Churn_Yes ~ Phone_Yes + FiberOptic + InternetService_None + Techie_Yes
```

```
Optimization terminated successfully.
         Current function value: 0.570988
         Iterations 5
```

In [19]:
```
print(modl2.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:              Churn_Yes   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9994
Method:                           MLE   Df Model:                            5
Date:                Mon, 16 Jan 2023   Pseudo R-squ.:                 0.01251
Time:                        12:56:28   Log-Likelihood:                -5709.9
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                 1.803e-29
==============================================================================
===
                           coef    std err          z      P>|z|      [0.025      0.9
75]
------------------------------------------------------------------------------
---
Intercept               -0.6829      0.080     -8.554      0.000      -0.839      -0.
526
Phone_Yes               -0.2012      0.076     -2.652      0.008      -0.350      -0.
053
FiberOptic              -0.4308      0.051     -8.451      0.000      -0.531      -0.
331
InternetService_None    -0.4468      0.063     -7.085      0.000      -0.570      -0.
323
Techie_Yes               0.3854      0.058      6.641      0.000       0.272       0.
499
eCheckPayment            0.1443      0.048      3.024      0.002       0.051       0.
238
==============================================================================
===
```

In [20]:
```
#Confusion matrix
X = dfr.iloc[:, 1:-1].values
y = dfr.iloc[:, -1].values
```

In [21]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
```

In [22]:
```
logreg = LogisticRegression(random_state = 0)
logreg.fit(X_train, y_train)
```

Out[22]:
```
LogisticRegression(random_state=0)
```

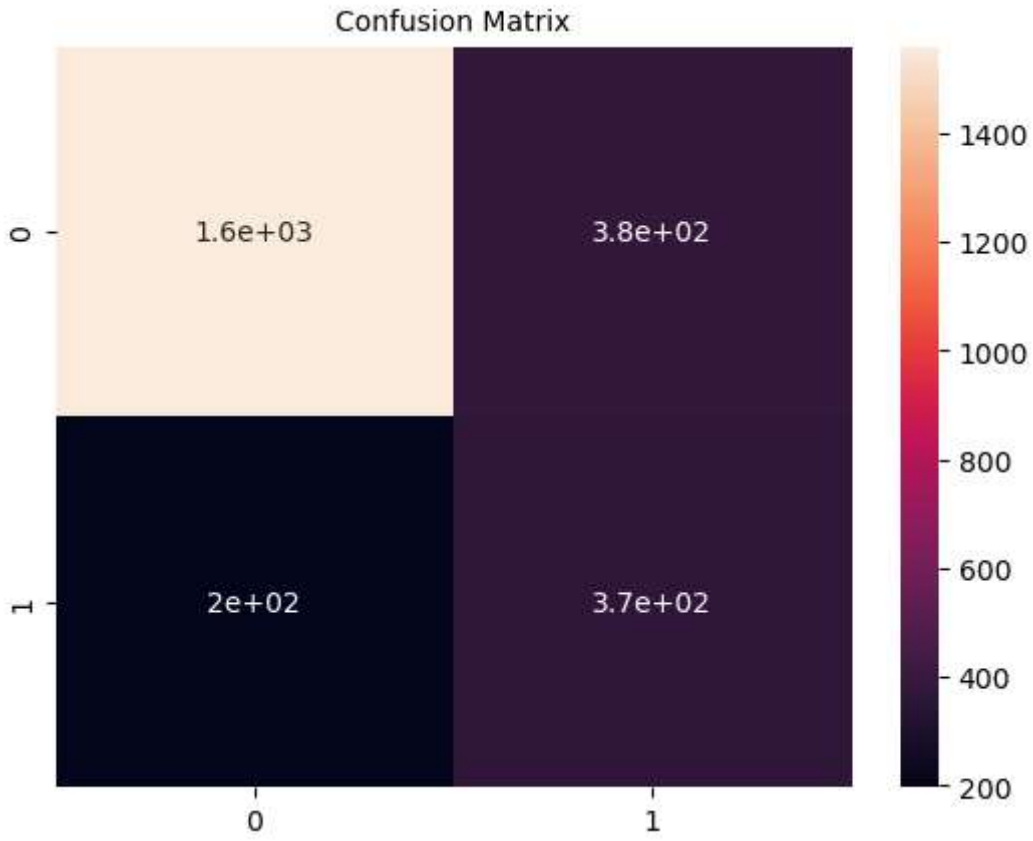
In [23]:
```
y_pred = logreg.predict(X_test)
```

In [24]:
```
cmatrix = confusion_matrix(y_test, y_pred)
print(cmatrix)
```

```
[[1559  378]
 [ 197  366]]
```

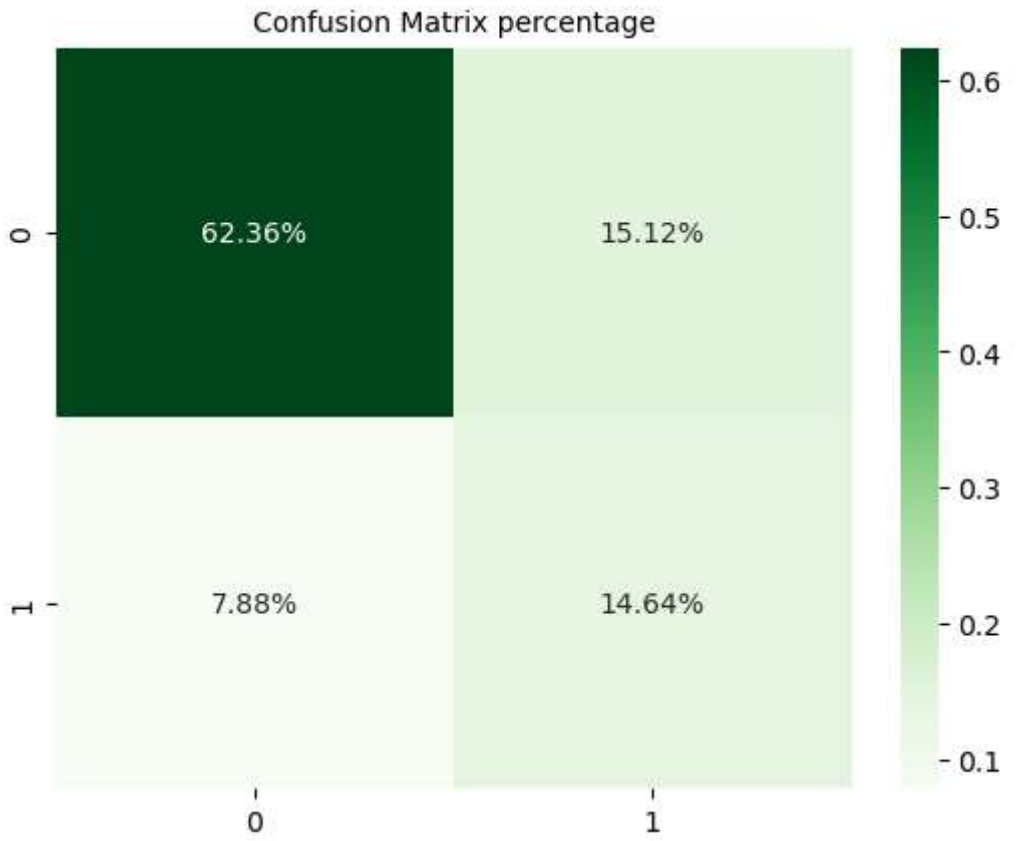
In [25]:
```
y_predict_test = logreg.predict(X_test)
cmatrix2 = confusion_matrix(y_test, y_predict_test)
sns.heatmap(cmatrix2, annot=True)
plt.title("Confusion Matrix", fontsize =10)
```

Out[25]:
```
Text(0.5, 1.0, 'Confusion Matrix')
```



In [26]:
```
sns.heatmap(cmatrix2/np.sum(cmatrix2), annot=True, fmt='.2%', cmap='Greens')
plt.title("Confusion Matrix percentage", fontsize =10)
```

Out[26]:
```
Text(0.5, 1.0, 'Confusion Matrix percentage')
```



In [27]:
```
acc = cross_val_score(estimator = logreg, X = X_train, y = y_train,cv = 10)
print("Accuracy: {:.2f} %".format(acc.mean()*100))
print("Standard Deviation: {:.2f} %".format(acc.std()*100))
```

```
Accuracy: 76.72 %
Standard Deviation: 0.88 %
```

In [28]:
```
python_version()
```

Out[28]:
```
'3.9.13'
```

In [29]:
```
!jupyter --version
```

```
Selected Jupyter core packages...
IPython          : 7.31.1
ipykernel        : 6.15.2
ipywidgets       : 7.6.5
jupyter_client   : 7.3.4
jupyter_core     : 4.11.1
jupyter_server   : 1.18.1
jupyterlab       : 3.4.4
nbclient         : 0.5.13
nbconvert        : 6.4.4
nbformat         : 5.5.0
notebook         : 6.4.12
qtconsole        : 5.2.2
traitlets        : 5.1.1
```

In [30]:
```
pd.__version__
```

Out[30]:
```
'1.4.4'
```

In [31]:
```
np.__version__
```

Out[31]:
```
'1.21.5'
```

In [32]:
```
sns.__version__
```

Out[32]:
```
'0.11.2'
```

In [33]:
```
sklearn.__version__
```

Out[33]:
```
'1.0.2'
```