Étude et Modélisation du Churn : Application d'Algorithmes de Machine Learning

ABDOULAYE TANGARA

Student in Msc in Quantitative and Computable Economic

Contacts

LinkedIn

Mon GitHub

Mon Portfolio

abdoulayetangara722@gmail.com

0.1 CHURN DATA MODELING

L'objectif de cette modélisation est d'identifier les clients succeptible de ce désabonner une plateforme. Ces clients sont connues sous l'etiquette de "Churn"

[2]:
```
# Module necessaire
import pandas as pd #  Manipulation des données
from tabulate import tabulate
import numpy as np # Calcule mathématique
import matplotlib.pyplot as plt # Data visualisation
import seaborn as sns # Data visualisation
import scipy.stats as stat # Test statistique

from skimpy import skim # Statistiques descriptives
import warnings
import random

from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression # Modèle de prediction
from sklearn.model_selection import train_test_split, GridSearchCV   # Division␣
↪le dataset en train et test
from sklearn.metrics import roc_curve, roc_auc_score, classification_report
from sklearn.compose import ColumnTransformer   # Transformation des colonnes␣
↪par types
from sklearn.preprocessing import StandardScaler, OneHotEncoder # Fonction de␣
↪transformation
from sklearn.decomposition import PCA   # Pour redimensionnalité
from imblearn import pipeline
import joblib

warnings.filterwarnings("ignore")
pd.options.mode.chained_assignment = None
```

[3]:
```
# Importation du dataset
churn = pd.read_csv("data_churn.csv", delimiter = ",")
churn.head(5)
```

[3]:
```
   customerID  gender  SeniorCitizen Partner Dependents  tenure PhoneService  \
0  7590-VHVEG  Female              0     Yes         No       1           No
1  5575-GNVDE    Male              0      No         No      34          Yes
2  3668-QPYBK    Male              0      No         No       2          Yes
3  7795-CFOCW    Male              0      No         No      45           No
4  9237-HQITU  Female              0      No         No       2          Yes

      MultipleLines InternetService OnlineSecurity  ... DeviceProtection  \
0  No phone service             DSL             No  ...               No
1                No             DSL            Yes  ...              Yes
2                No             DSL            Yes  ...               No
```

```
3  No phone service             DSL             Yes  ...                Yes
4                No      Fiber optic             No  ...                 No

  TechSupport StreamingTV StreamingMovies        Contract PaperlessBilling  \
0          No          No              No  Month-to-month              Yes
1          No          No              No        One year               No
2          No          No              No  Month-to-month              Yes
3         Yes          No              No        One year               No
4          No          No              No  Month-to-month              Yes

              PaymentMethod MonthlyCharges  TotalCharges Churn
0          Electronic check          29.85         29.85    No
1              Mailed check          56.95        1889.5    No
2              Mailed check          53.85        108.15   Yes
3  Bank transfer (automatic)         42.30       1840.75    No
4          Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

[4]:
```python
# Information basique sur le dataset
print("\n Description des variables : \n")
print(churn.describe().T)
```

```
 Description des variables :

                 count       mean        std    min   25%    50%    75%  \
SeniorCitizen   7043.0   0.162147   0.368612   0.00   0.0   0.00   0.00
tenure          7043.0  32.371149  24.559481   0.00   9.0  29.00  55.00
MonthlyCharges  7043.0  64.761692  30.090047  18.25  35.5  70.35  89.85

                   max
SeniorCitizen     1.00
tenure           72.00
MonthlyCharges  118.75
```

0.2 I. ANALYSE UNIVARIEE

Il s'agit ici de se faire une idée du comportement de chaque variable prise individuelle.

1. ***Analyse de la dimension du dataset utilisé***

[5]:
```
print("Le nombre de ligne du dataset est de : ", churn.shape[0])
print("Le nombre de colonne du dataset est de : ", churn.shape[1])
```

```
Le nombre de ligne du dataset est de :  7043
Le nombre de colonne du dataset est de :  21
```

2. ***Analyse du type des variables dans le dataset***

[6]:
```
print(f"Les colonnes du dataset :\n {churn.columns}")
print(" \n Les types de variables utilisées : \n", churn.dtypes.T)
```

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

 Les types de variables utilisées :
 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
```

3. ***Traitement preliminaire du dataset***
    - Recherche des valeurs manquantes

- Conversion des variables
- Imputation des variables

[7]:
```python
# Analyse des missings values
convert_var = churn["TotalCharges"].replace(" ", np.nan)

# Conversion des variables
churn["TotalCharges"] = convert_var.astype(float)
for col in ['gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'Contract', 'PaperlessBilling',␣
 ↪'PaymentMethod','Churn']:
    churn[col] = churn[col].astype("category")

# Suppression de la variables "customerID"
churn.drop(columns="customerID", inplace=True)

print(" \n Chercher de valeur manquantes : \n", churn.isnull().sum().T)
churn.dropna(inplace=True)

print(" \n Dataset après correction des valeurs manquantes : \n", churn.isnull().
 ↪sum())
```

```
 Chercher de valeur manquantes :
 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       11
Churn               0
dtype: int64
```

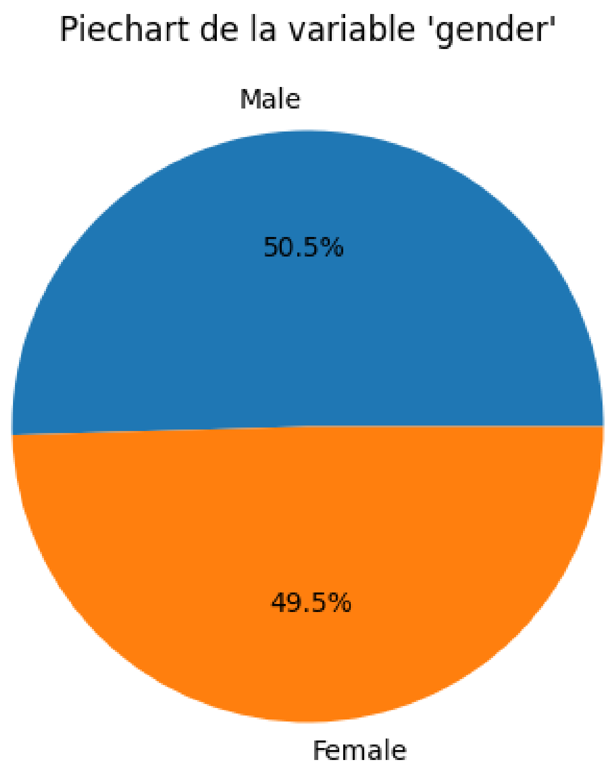
```
 Dataset après correction des valeurs manquantes :
 gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

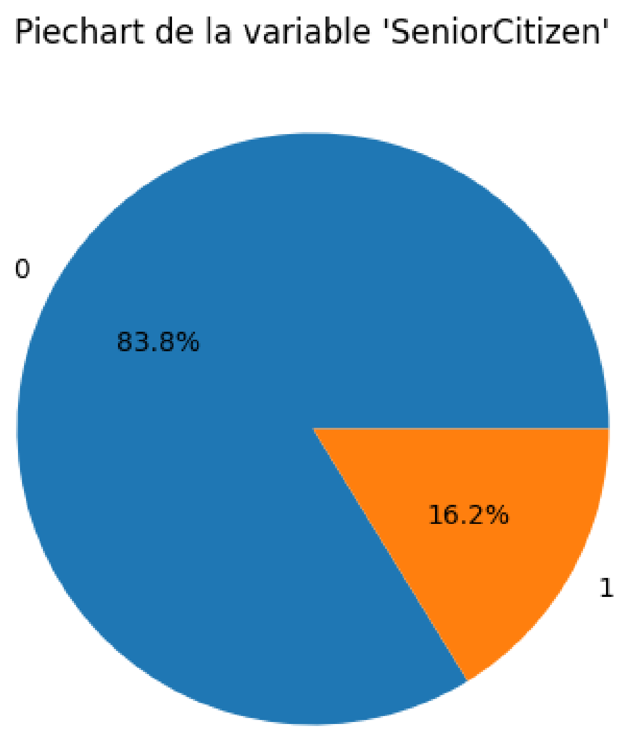
4. ***Statistique descriptive des variables***

[ ]:
```python
# Les modilitées des chaque variables
for col in churn.select_dtypes(include=['category']).columns.tolist():
    print(f'\n Les modalites de {col}', churn[col].unique())

# Analyse descriptives des valeurs Quantitatives et Qualitatives
skim(churn)
```

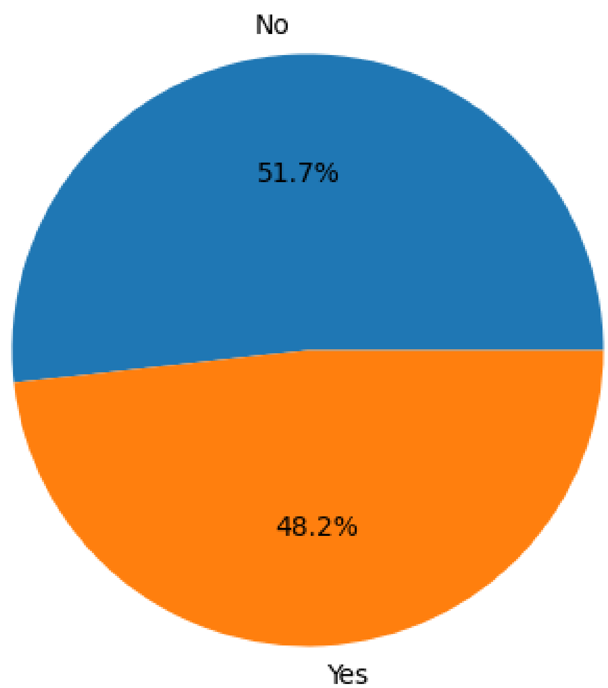
5. ***Analyse graphique des variables catégorielles à deux modalités***

[9]:
```python
var_quali_1 = ['gender', 'SeniorCitizen', 'Partner',
       'PhoneService', 'PaperlessBilling', 'PaymentMethod']
for i in var_quali_1:
    plt.figure(figsize=(10,5))
    count = np.round(churn[i].value_counts()/churn.shape[0]*100, 2)

    plt.pie(count, labels=count.index, autopct="%1.1f%%")
    plt.title(f"Piechart de la variable '{i}'")
    plt.show()
```

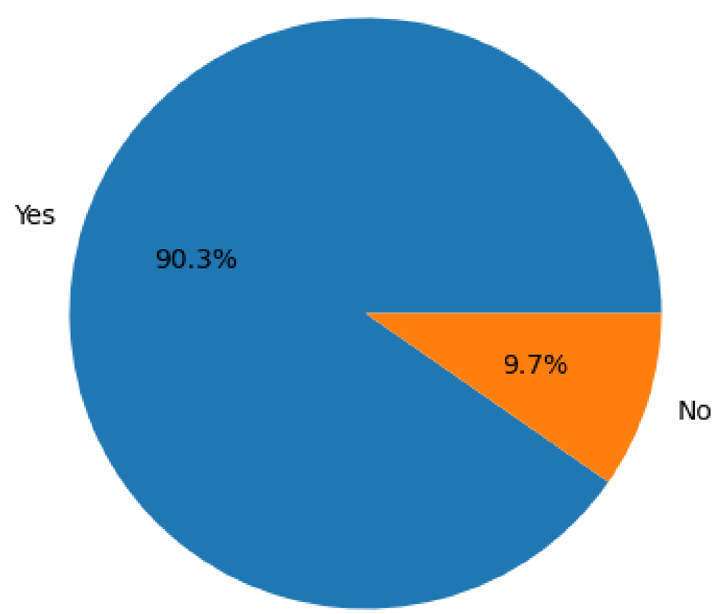

Piechart de la variable 'gender'
Male
50.5%
49.5%
Female

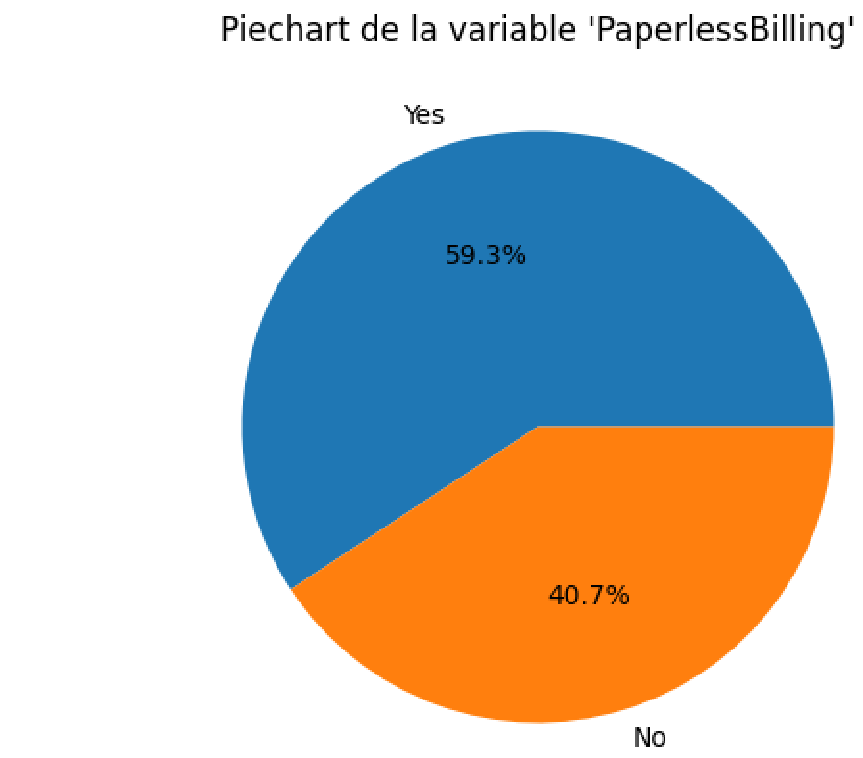

Piechart de la variable 'SeniorCitizen'
0
83.8%
16.2%
1

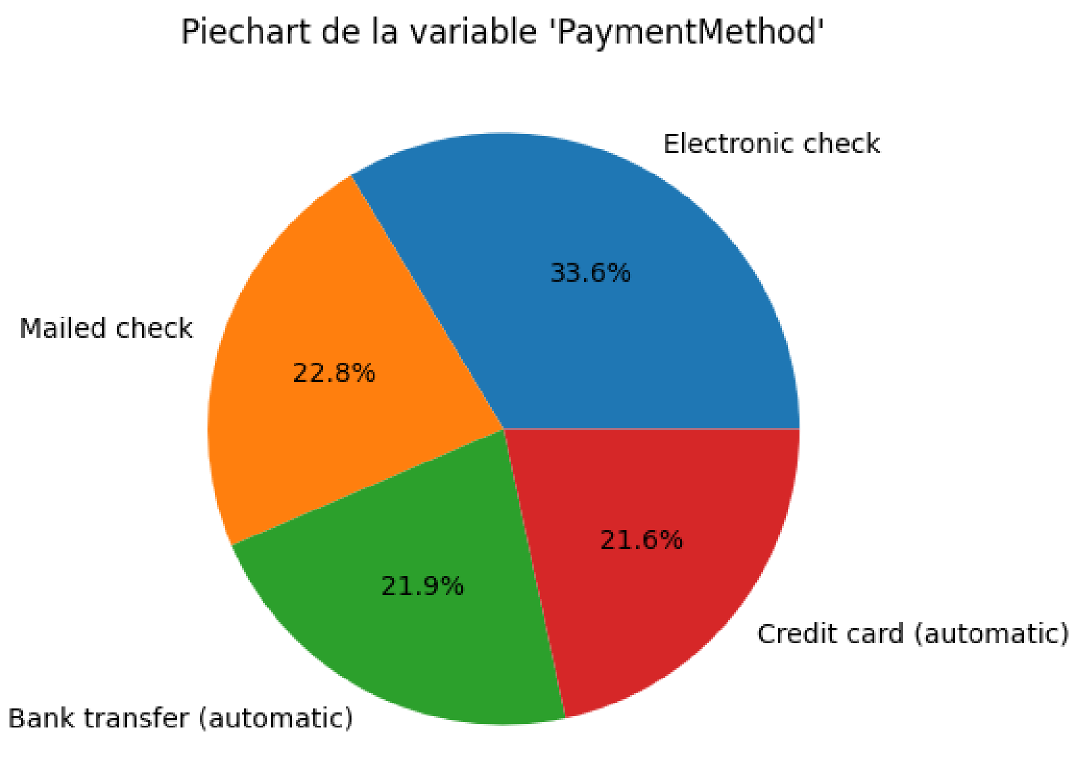
Piechart de la variable 'Partner'



Piechart de la variable 'PhoneService'

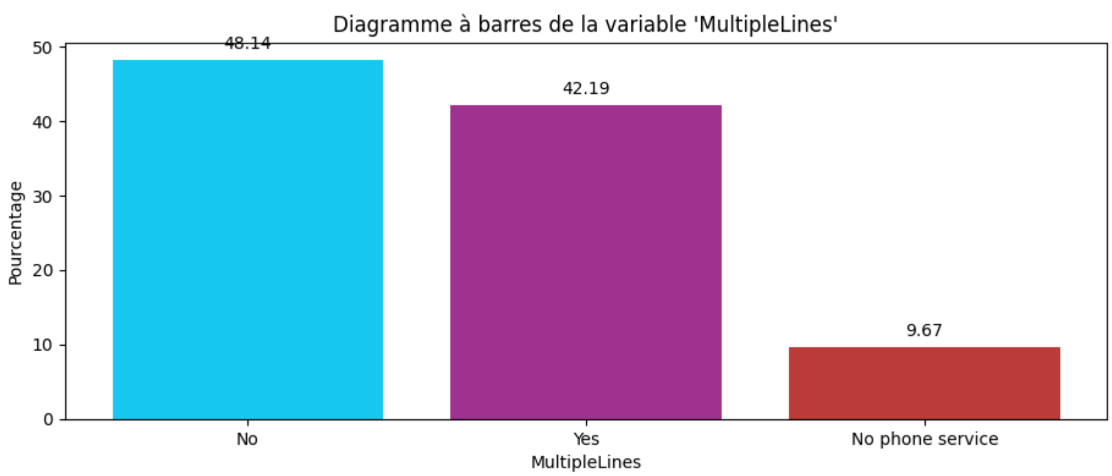



Piechart de la variable 'PaperlessBilling'
Yes
59.3%
40.7%
No


Piechart de la variable 'PaymentMethod'
Electronic check
33.6%
Mailed check
22.8%
21.9%
21.6%
Credit card (automatic)
Bank transfer (automatic)

6. ***Analyse visualisation des variables catégorielles à plus deux modalites***

[10]:
```
var_quali_2 = ['MultipleLines', 'InternetService', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'Contract']
for i in var_quali_2:
    count = np.round((churn[i].value_counts() / churn.shape[0]) * 100, 2)
    plt.figure(figsize=(11, 4))

    # Génération de couleurs aléatoires pour chaque barre
    colors = ['#%06X' % random.randint(0, 0xFFFFFF) for _ in range(len(count))]

    bars = plt.bar(count.index, count.values, color=colors)
    plt.title(f"Diagramme à barres de la variable '{i}'")

    # Ajout des valeurs au-dessus de chaque barre
    for bar in bars:
        yval = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2, yval + 1, yval, ha='center',␣
 ↪va='bottom')

    plt.xlabel(i)
    plt.ylabel('Pourcentage')
    plt.show()
```



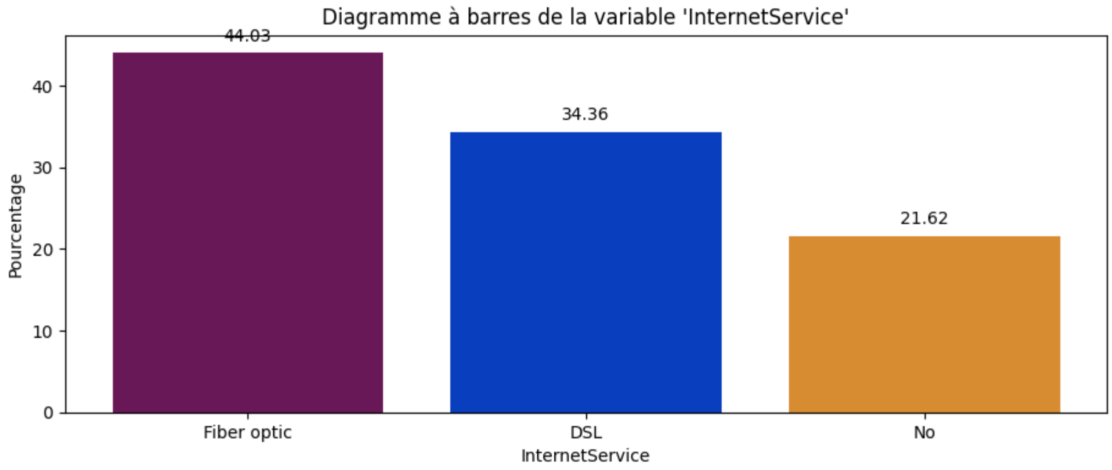


Diagramme à barres de la variable 'InternetService'
44.03
34.36
21.62
Pourcentage
0
10
20
30
40
Fiber optic
DSL
No
InternetService

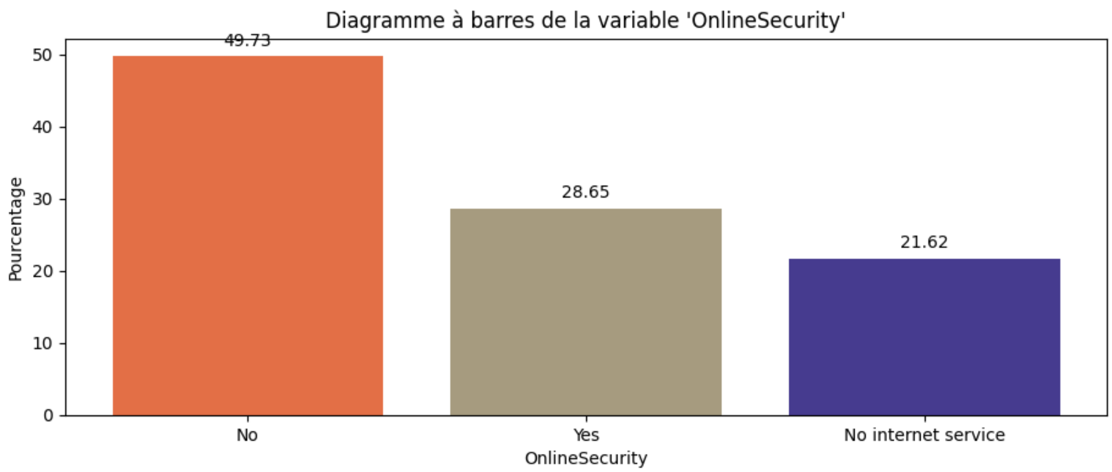


Diagramme à barres de la variable 'OnlineSecurity'
49.73
28.65
21.62
Pourcentage
0
10
20
30
40
50
No
Yes
No internet service
OnlineSecurity

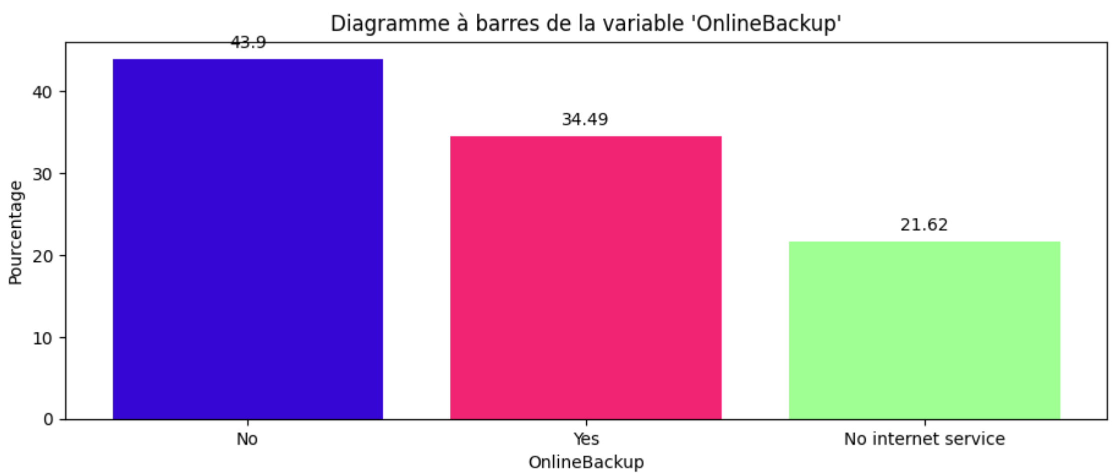


Diagramme à barres de la variable 'OnlineBackup'
43.9
34.49
21.62
Pourcentage
0
10
20
30
40
No
Yes
No internet service
OnlineBackup

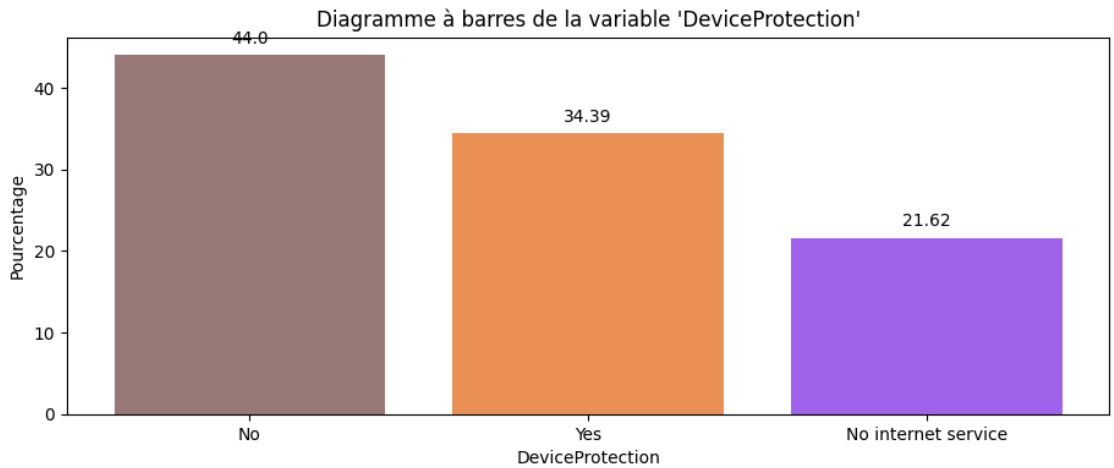


Diagramme à barres de la variable 'DeviceProtection'
44.0
34.39
21.62
Pourcentage
0
10
20
30
40
No
Yes
No internet service
DeviceProtection

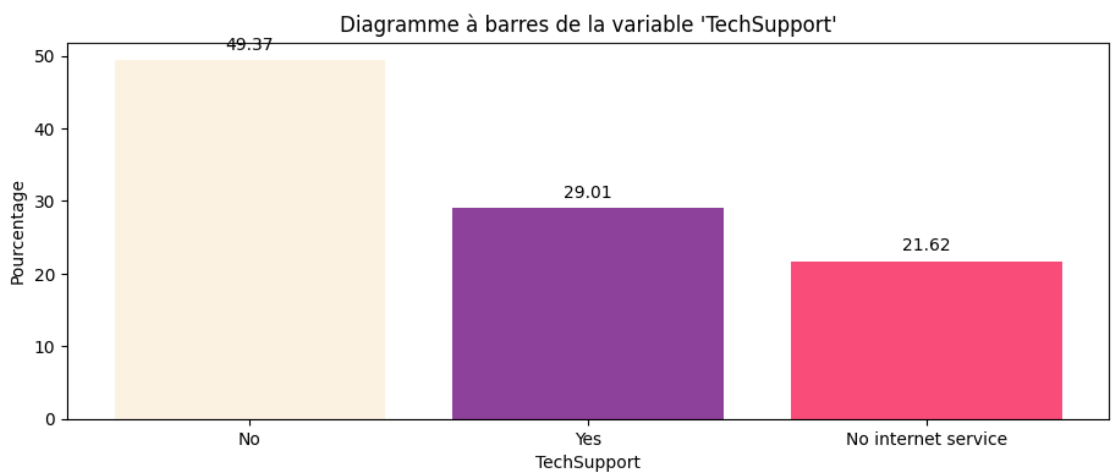


Diagramme à barres de la variable 'TechSupport'
49.37
29.01
21.62
Pourcentage
0
10
20
30
40
50
No
Yes
No internet service
TechSupport

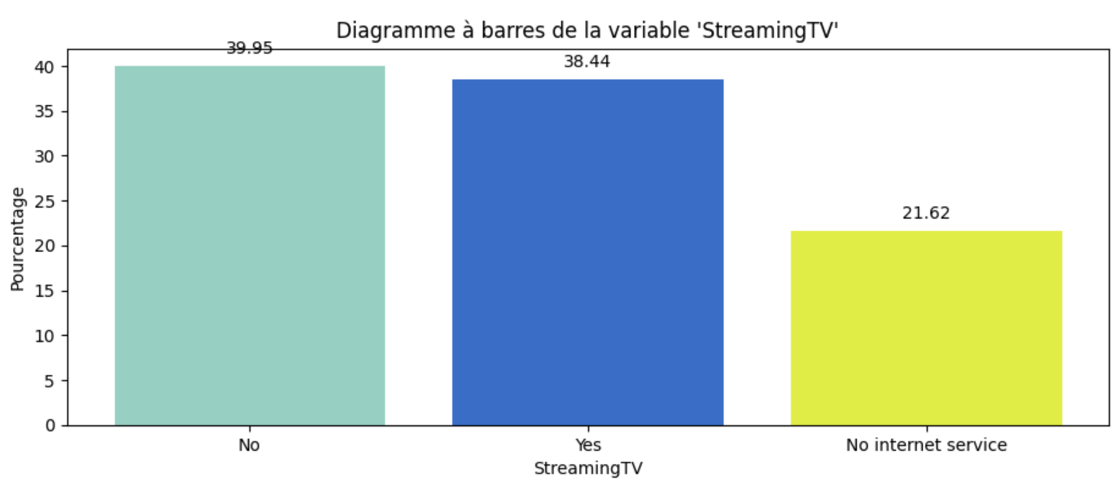
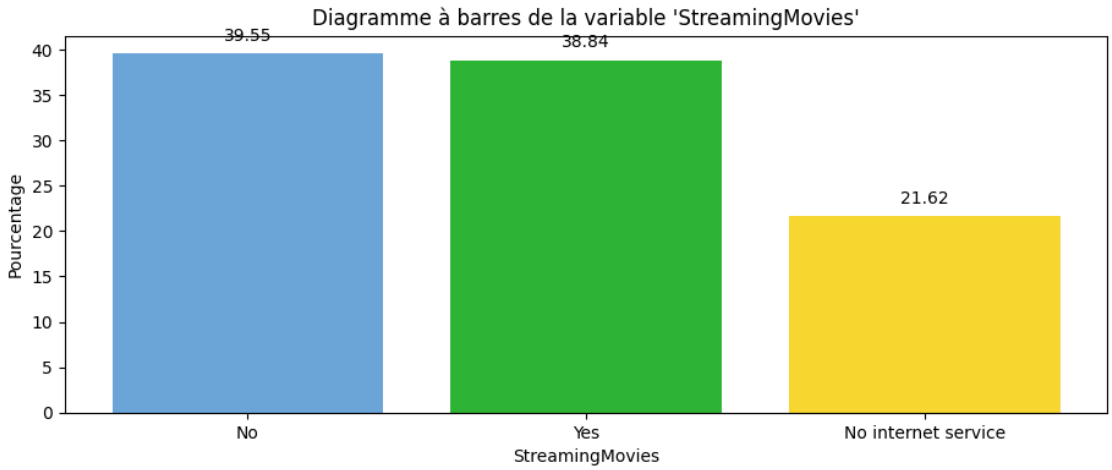
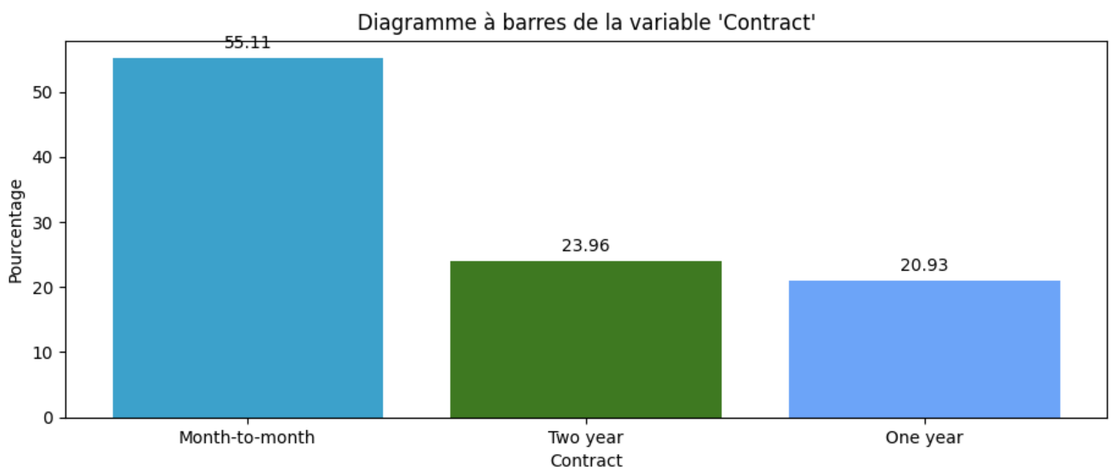


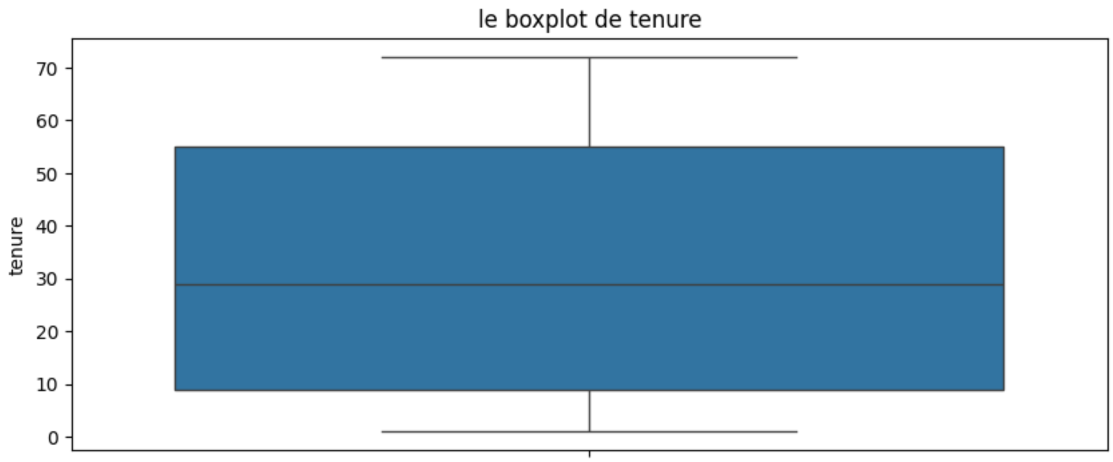
Diagramme à barres de la variable 'StreamingTV'
39.95
38.44
21.62
Pourcentage
0
5
10
15
20
25
30
35
40
No
Yes
No internet service
StreamingTV

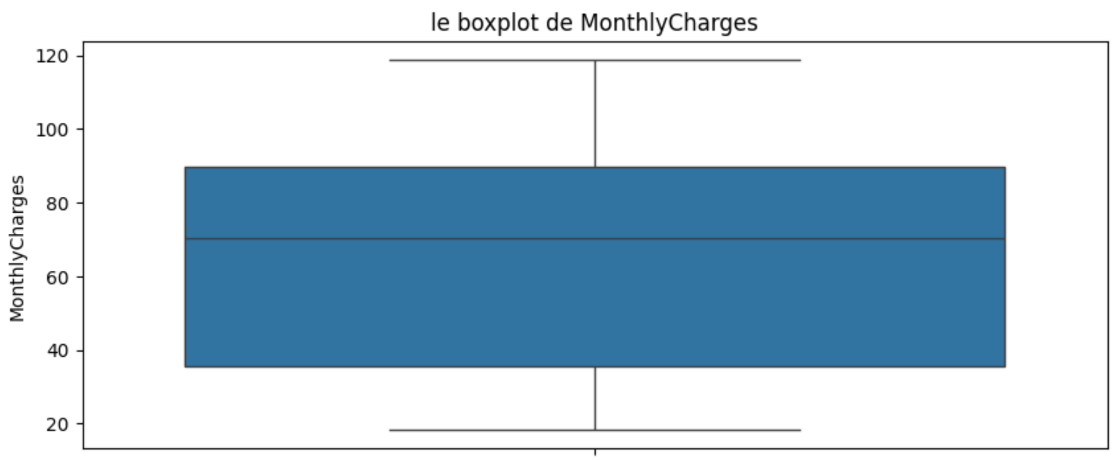


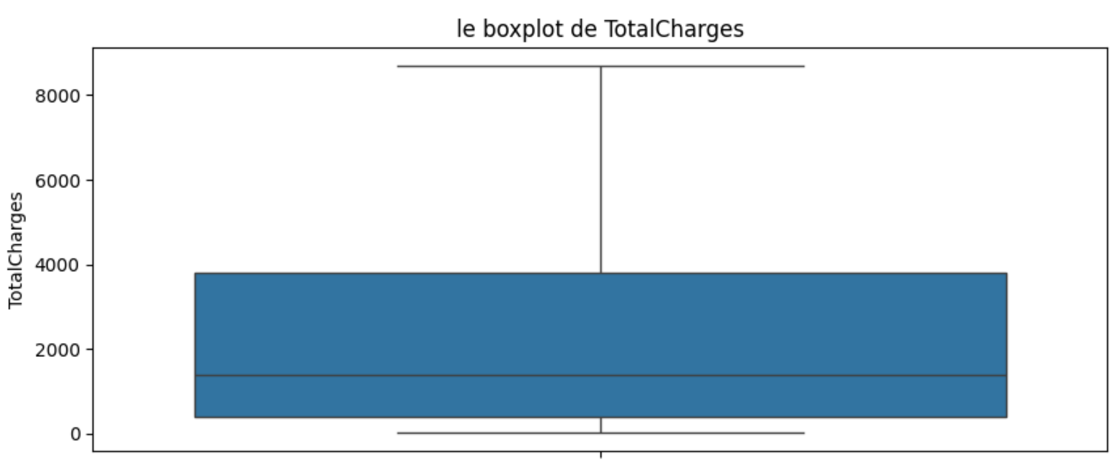


7. ***Analyse de la dispersion des variables quantitatives***

[11]:
```
var_quanti_1 = churn.select_dtypes(include=[float, int]).columns.to_list()

for i in var_quanti_1:
    plt.figure(figsize=(10,4))
    sns.boxplot(churn[i])
    plt.title(f"le boxplot de {i}")
    plt.show()
```

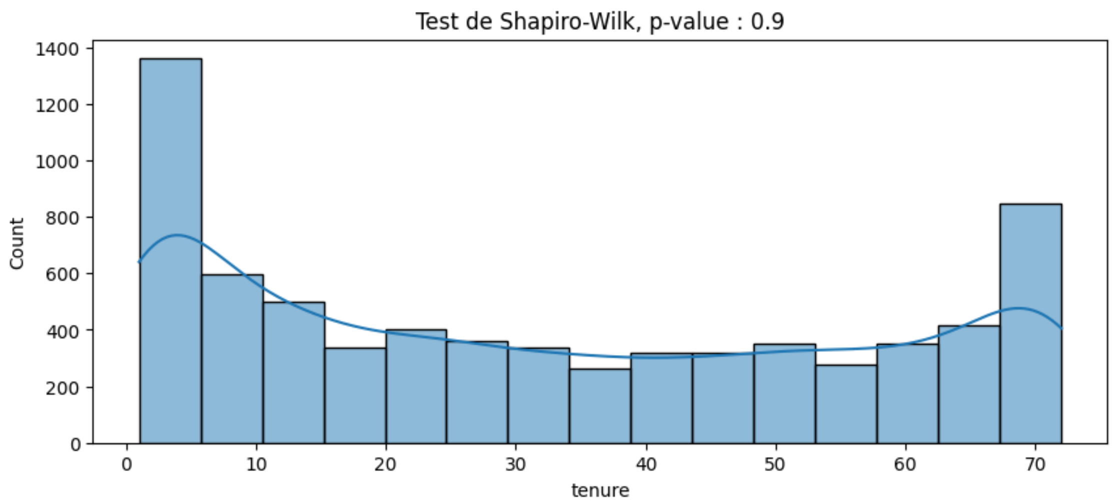

le boxplot de tenure
tenure
0
10
20
30
40
50
60
70

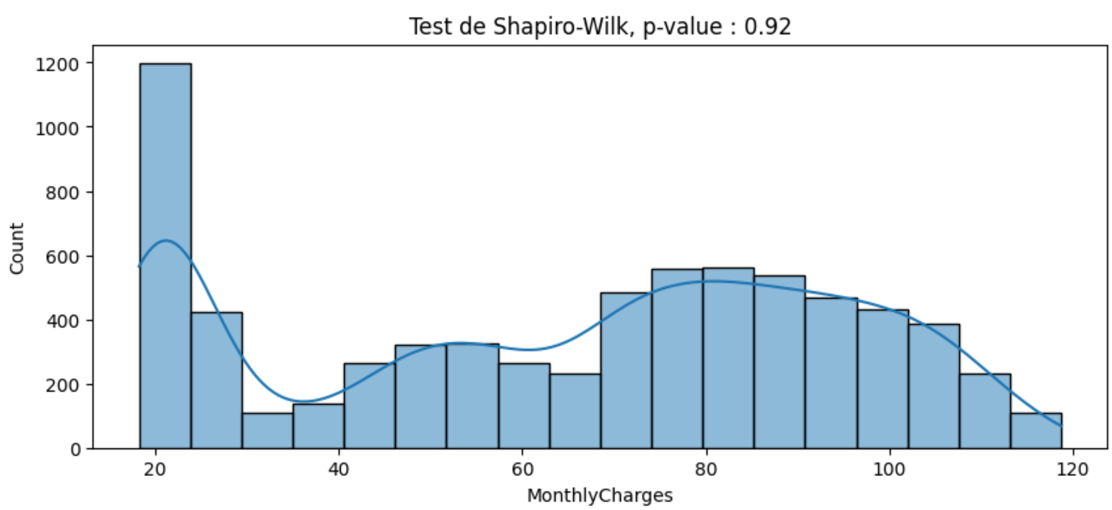

le boxplot de MonthlyCharges
MonthlyCharges
20
40
60
80
100
120

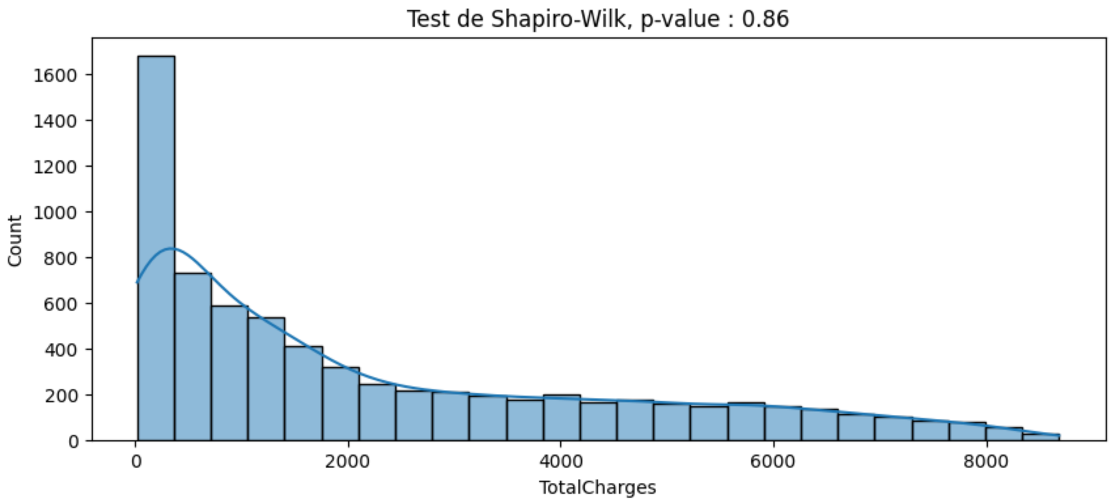

le boxplot de TotalCharges
TotalCharges
0
2000
4000
6000
8000

8. ***Test de normalité des variables quantitatives***

[12]:
```python
# Analyse de normalité des variables quantitatives
for i in var_quanti_1:
    pvalue , _ = stat.shapiro(churn[i])
    plt.figure(figsize=(10,4))
    sns.histplot(churn[i],kde=True)
    plt.title(f"Test de Shapiro-Wilk, p-value : {np.round(pvalue, 2)}")
    plt.show()
```






Test de Shapiro-Wilk, p-value : 0.86
1600
1400
1200
1000
800
600
400
200
0
Count
0
2000
4000
6000
8000
TotalCharges

0.3 II. ANALYSE BIVARIEE

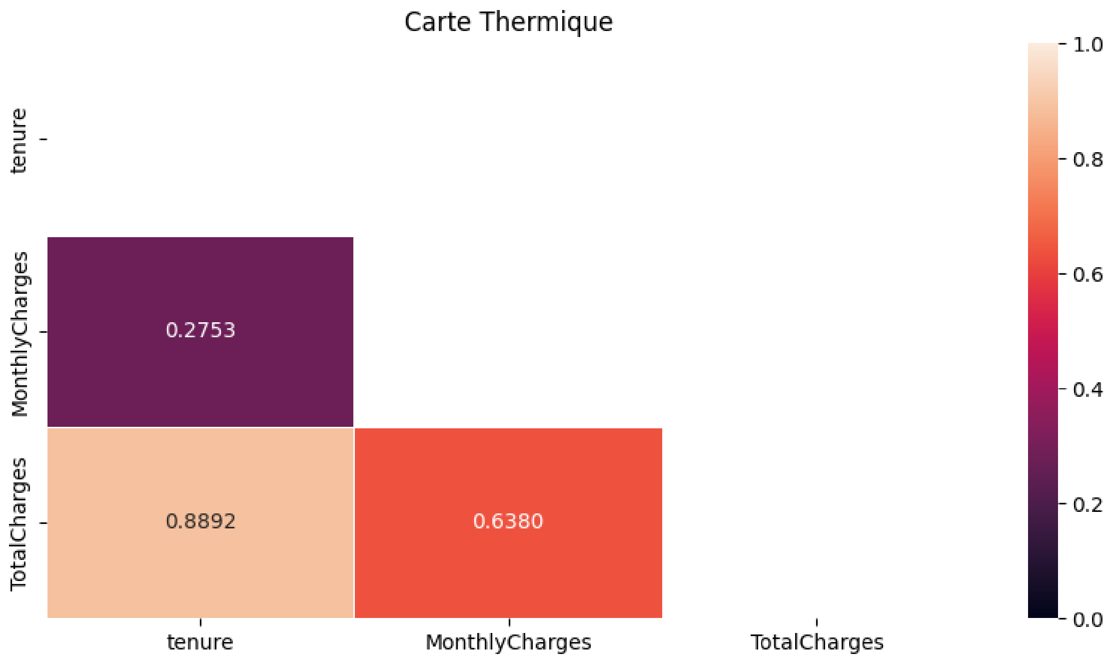
Il s'agit d'étudier la relation qui existe entre les variables deux à deux avec les tests correspondants à chaque liaison ou association.

Dans un premier temps, analyser les variables quantitatives et qualitatives entre elles. Dans un second temps, analyser la relation variable qualitative et quantitative.

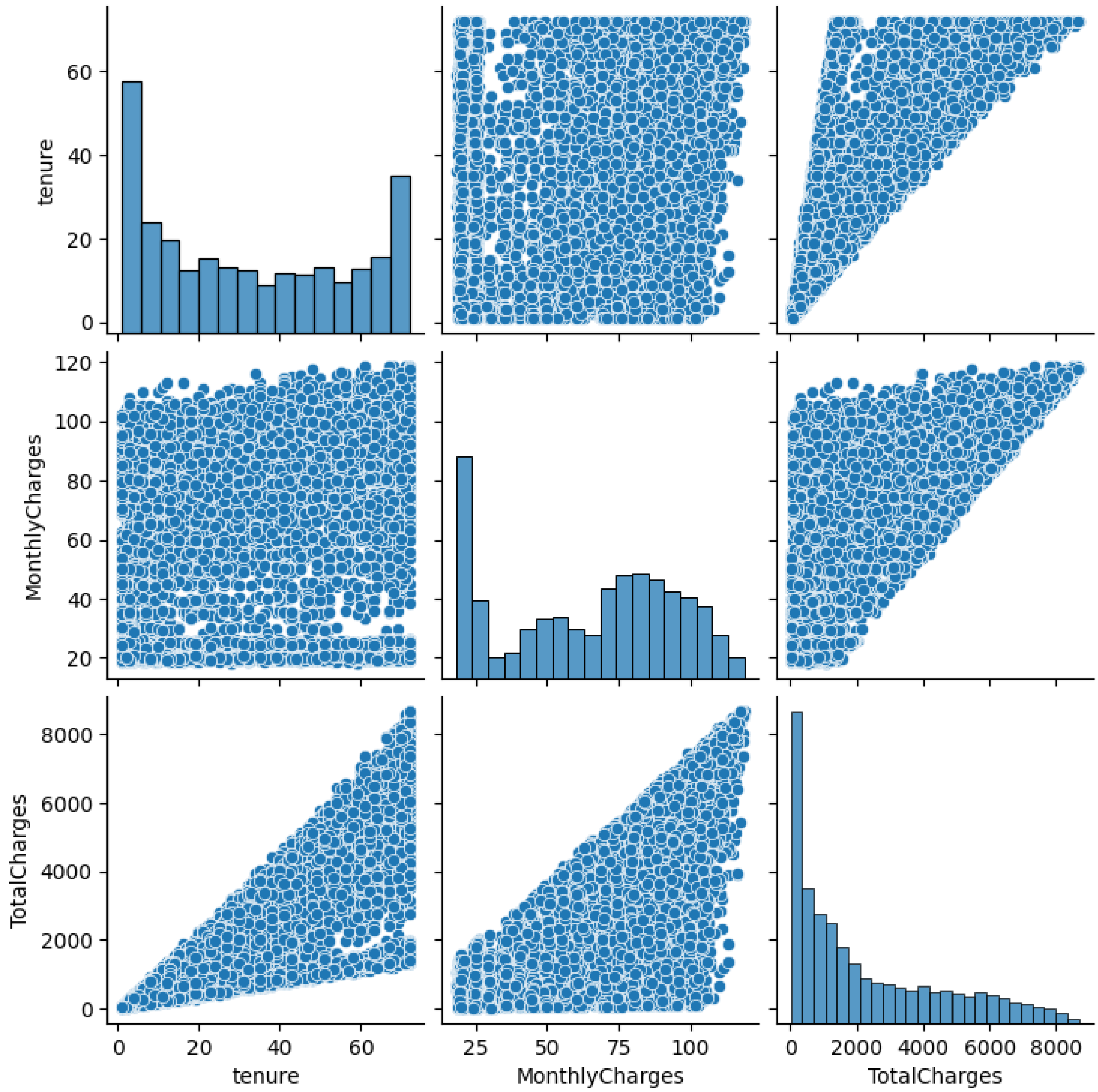
1. ***Relation des variables quantitative - quantitative***

[13]:
```
# Analyse quanti-quanti
r = churn[['tenure', 'MonthlyCharges', 'TotalCharges']].corr(method="spearman")
for i in churn[['tenure', 'MonthlyCharges', 'TotalCharges']].columns.to_list():
    for j in churn[['tenure', 'MonthlyCharges', 'TotalCharges']].columns.
 ↪to_list():
        if i != j:

            # Test de spearman car les variables ne sont pas normalement␣
 ↪distribuées
            spearman_corr, pvalue = stat.spearmanr(churn[i], churn[i])
            if pvalue<0.05 and spearman_corr > 0.5:
                text1 = f"les variables {i} et {j} sont corrélées avec pvalue =␣
 ↪{np.round(pvalue,2)}"
                print(text1)
            elif pvalue<0.05 and spearman_corr < 0.5:
                text2 = f"les variables {i} et {j} ne sont pas corrélées avec␣
 ↪pvalue = {np.round(pvalue,2)}"
                print(text2)
            else:
                print("la valeurs des statistiques calculées ne sont pas␣
 ↪significatives pour conclure")
mask = np.triu(np.ones(r.shape, dtype=int))
plt.figure(figsize=(10,5))
sns.heatmap(r, vmax = 1, vmin=0, fmt=".4f", annot=True, linewidths=0.5,␣
 ↪mask=mask)
plt.title("Carte Thermique")
plt.show()


# Pair plot
sns.pairplot(churn[['tenure', 'MonthlyCharges', 'TotalCharges']])
```

```
les variables tenure et MonthlyCharges sont corrélées avec pvalue = 0.0
les variables tenure et TotalCharges sont corrélées avec pvalue = 0.0
les variables MonthlyCharges et tenure sont corrélées avec pvalue = 0.0
les variables MonthlyCharges et TotalCharges sont corrélées avec pvalue = 0.0
les variables TotalCharges et tenure sont corrélées avec pvalue = 0.0
les variables TotalCharges et MonthlyCharges sont corrélées avec pvalue = 0.0
```



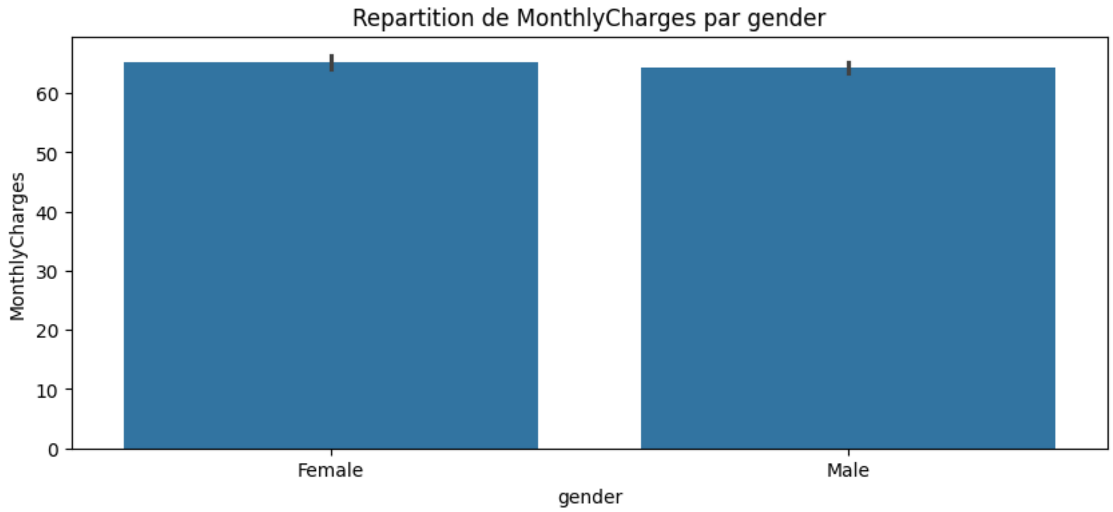
[13]: <seaborn.axisgrid.PairGrid at 0x18918028e80>

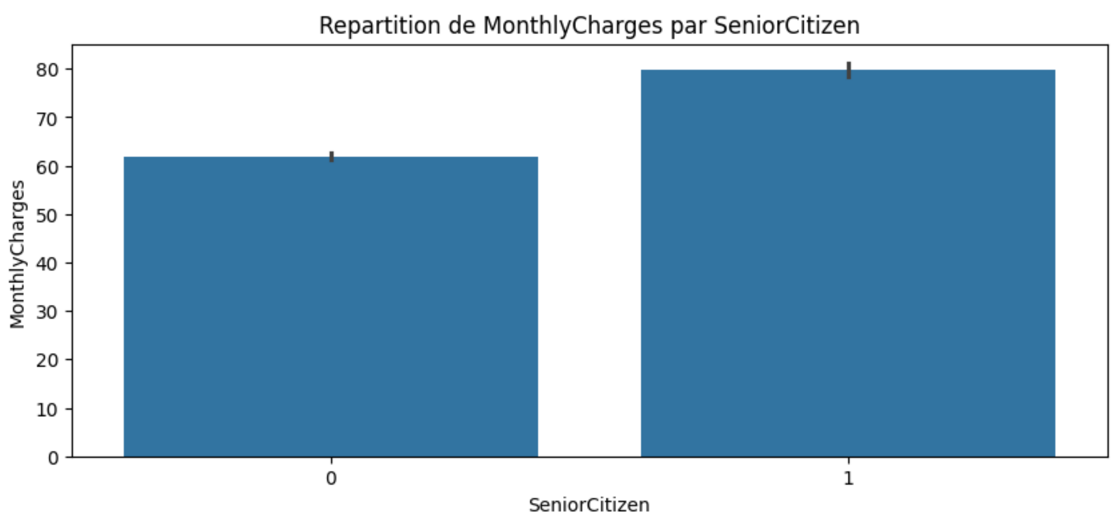


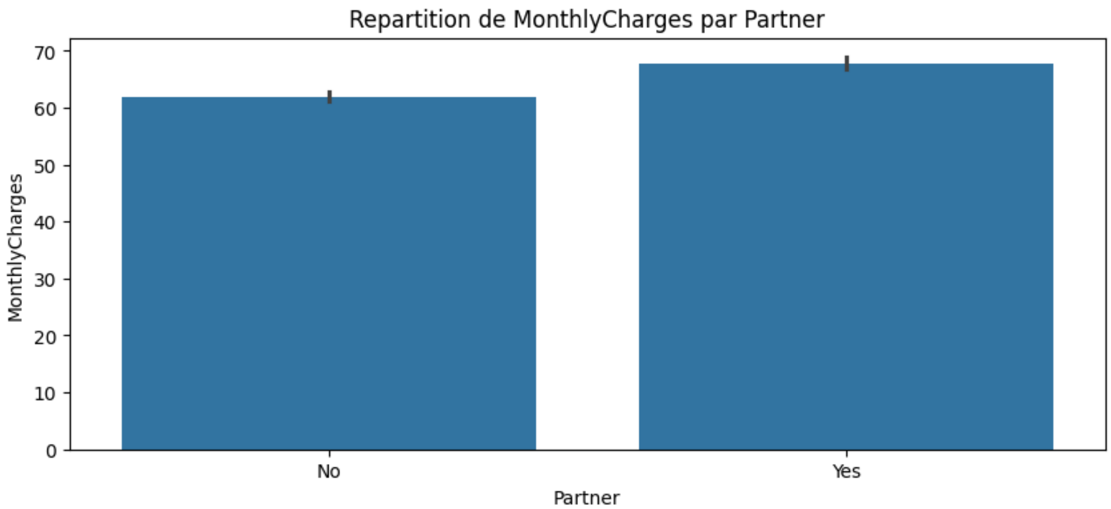
2. Relation des variables quantitative - qualitative

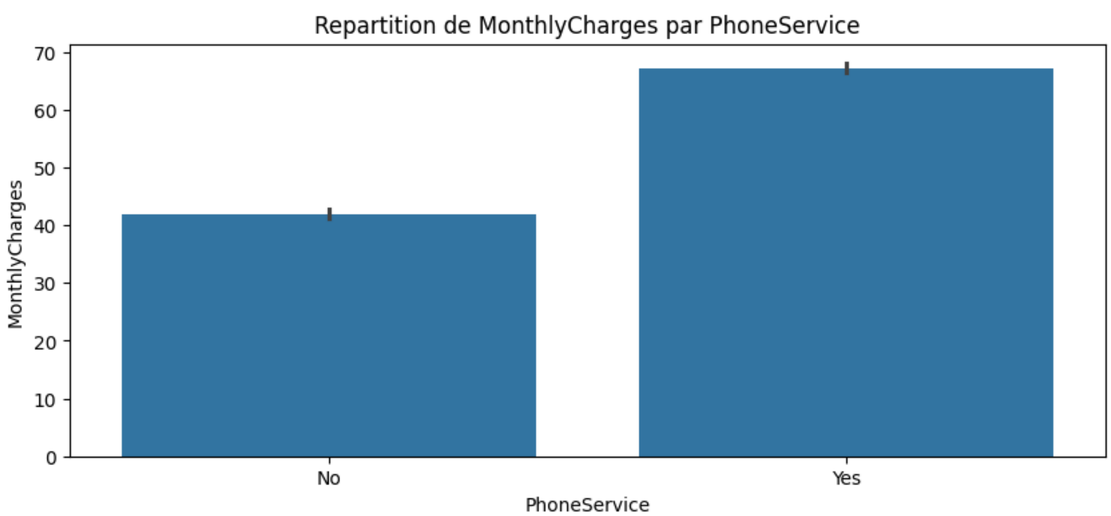
[14]:
```python
# analyse quanti-quali
var_quanti_1
for val in var_quali_1 :
    plt.figure(figsize=(10,4))
    sns.barplot(x=churn[val], y = churn[var_quanti_1[1]], estimator="mean")
    plt.title(f"Repartition de {var_quanti_1[1]} par {val}")
    plt.show()

var_quali_g = var_quali_1 + var_quali_2 # Total des variables catégorielles

for val in var_quali_2 :
    plt.figure(figsize=(10,4))
    sns.barplot(x=churn[val], y = churn[var_quanti_1[1]], estimator="mean")
    plt.title(f"Repartition de {var_quanti_1[1]} par {val}")
```

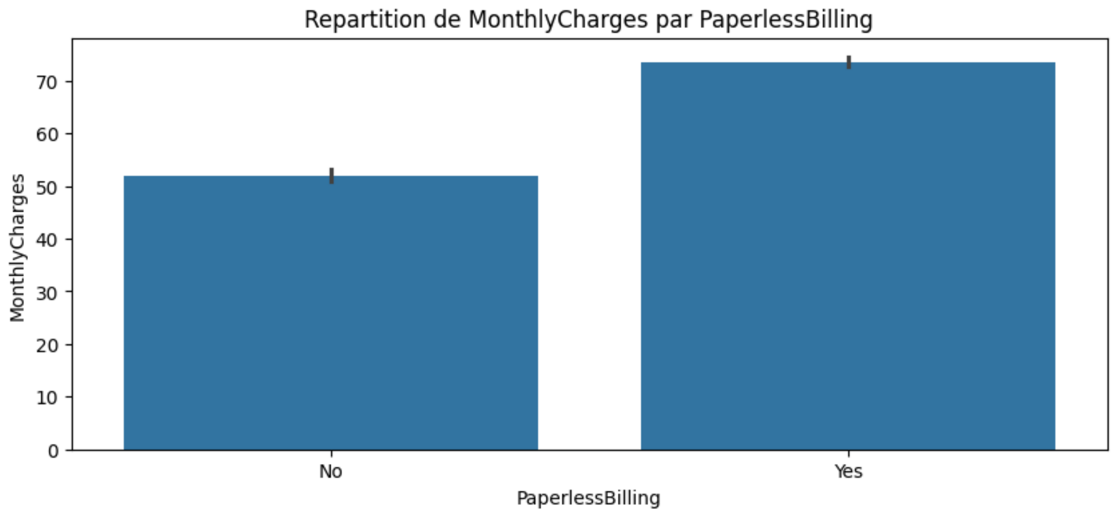
```
plt.show()
```

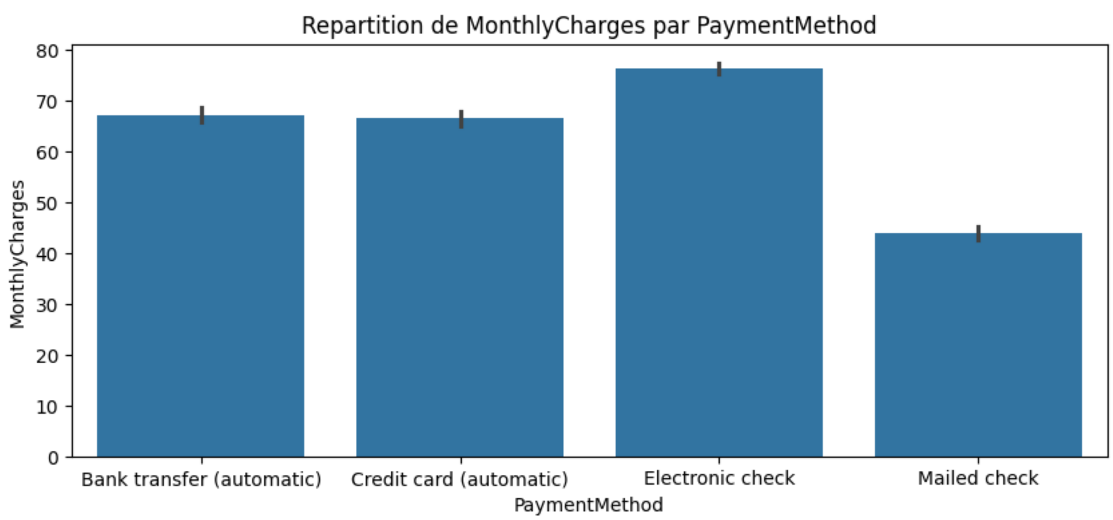

Repartition de MonthlyCharges par gender
MonthlyCharges
0
10
20
30
40
50
60
Female
Male
gender

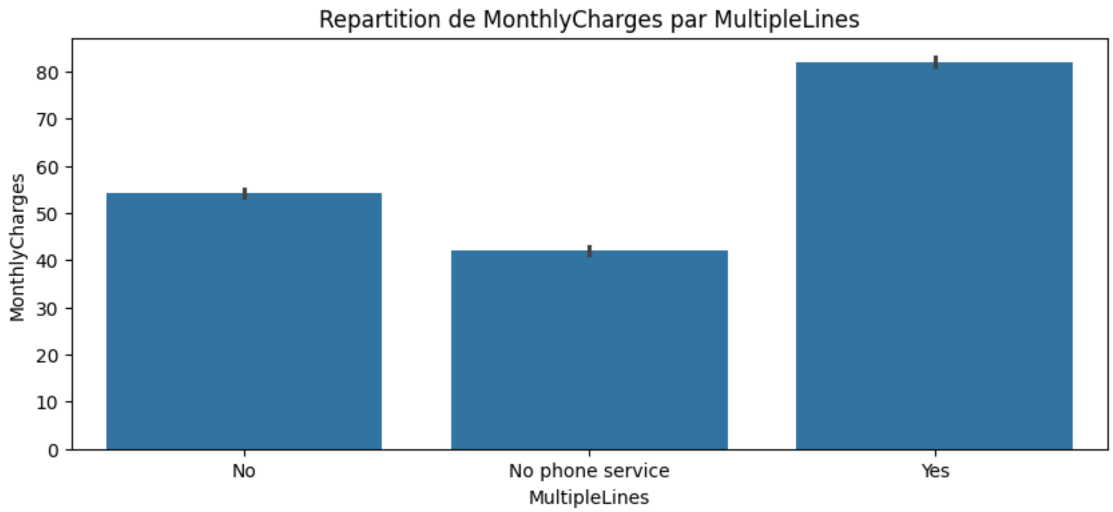

Repartition de MonthlyCharges par SeniorCitizen
MonthlyCharges
0
10
20
30
40
50
60
70
80
0
1
SeniorCitizen

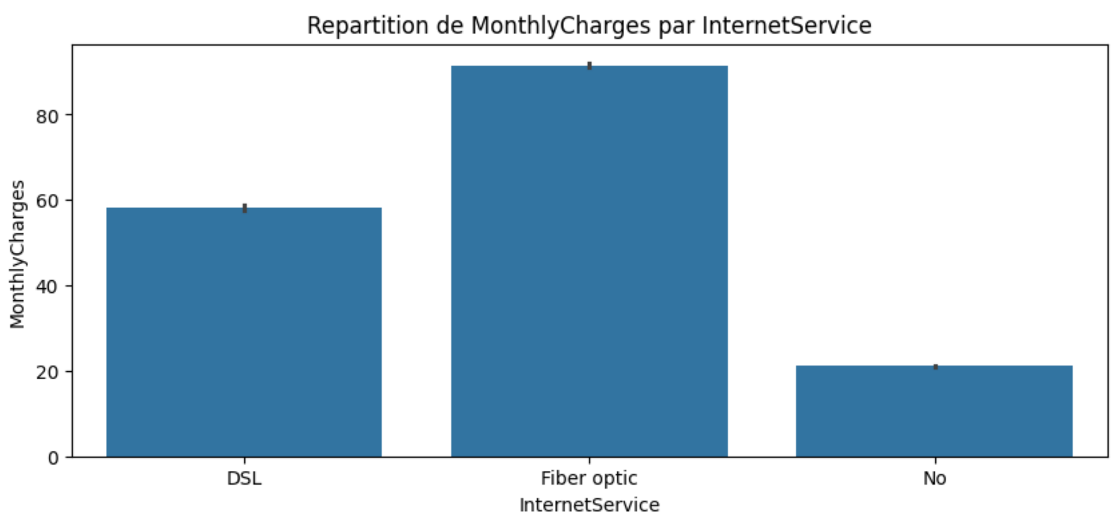

Repartition de MonthlyCharges par Partner
MonthlyCharges
0
10
20
30
40
50
60
70
No
Yes
Partner

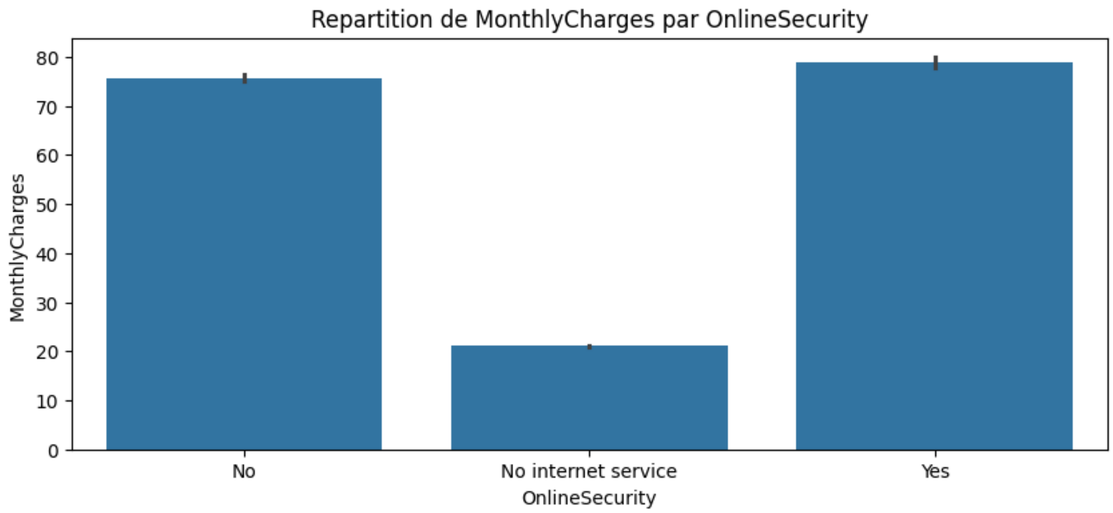

Repartition de MonthlyCharges par PhoneService
MonthlyCharges
0
10
20
30
40
50
60
70
No
Yes
PhoneService

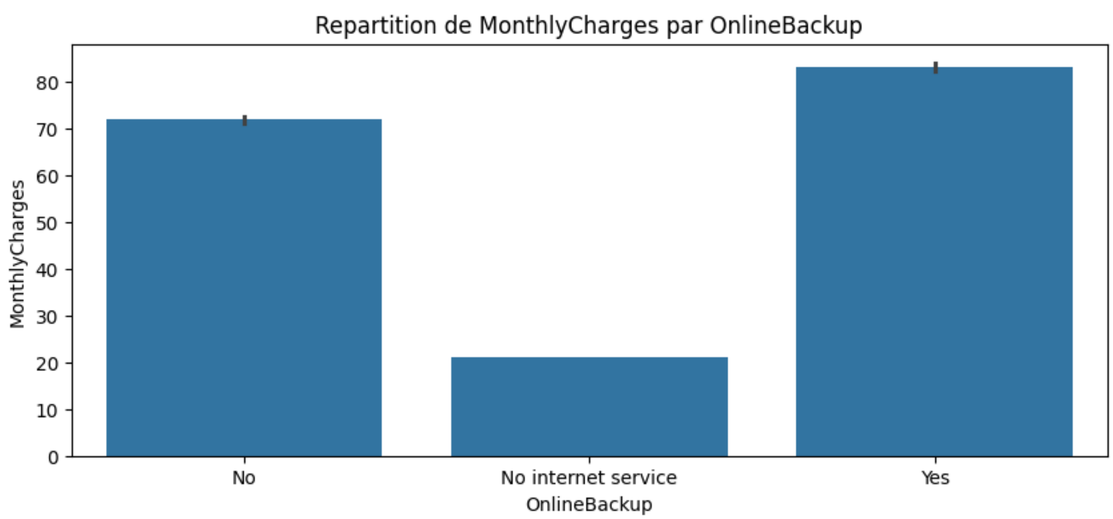

Repartition de MonthlyCharges par PaperlessBilling
MonthlyCharges
0
10
20
30
40
50
60
70
No
Yes
PaperlessBilling

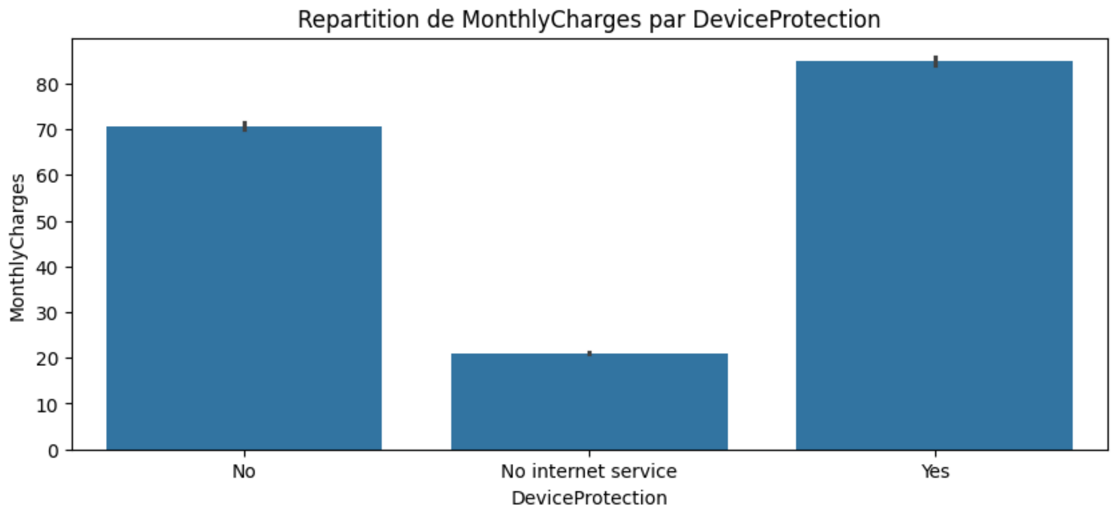

Repartition de MonthlyCharges par PaymentMethod
MonthlyCharges
0
10
20
30
40
50
60
70
80
Bank transfer (automatic)
Credit card (automatic)
Electronic check
Mailed check
PaymentMethod

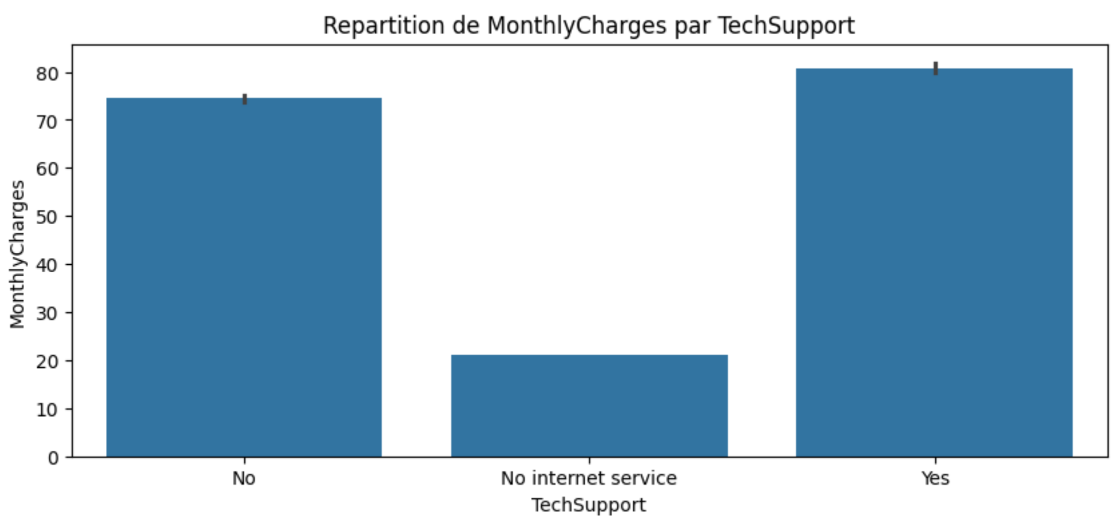

Repartition de MonthlyCharges par MultipleLines
MonthlyCharges
80
70
60
50
40
30
20
10
0
No
No phone service
Yes
MultipleLines

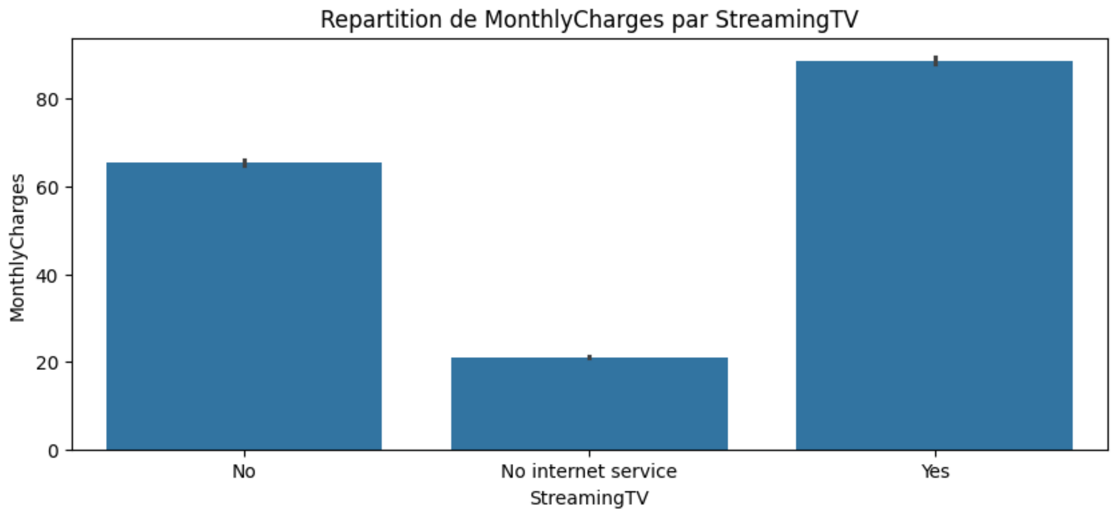

Repartition de MonthlyCharges par InternetService
MonthlyCharges
80
60
40
20
0
DSL
Fiber optic
No
InternetService

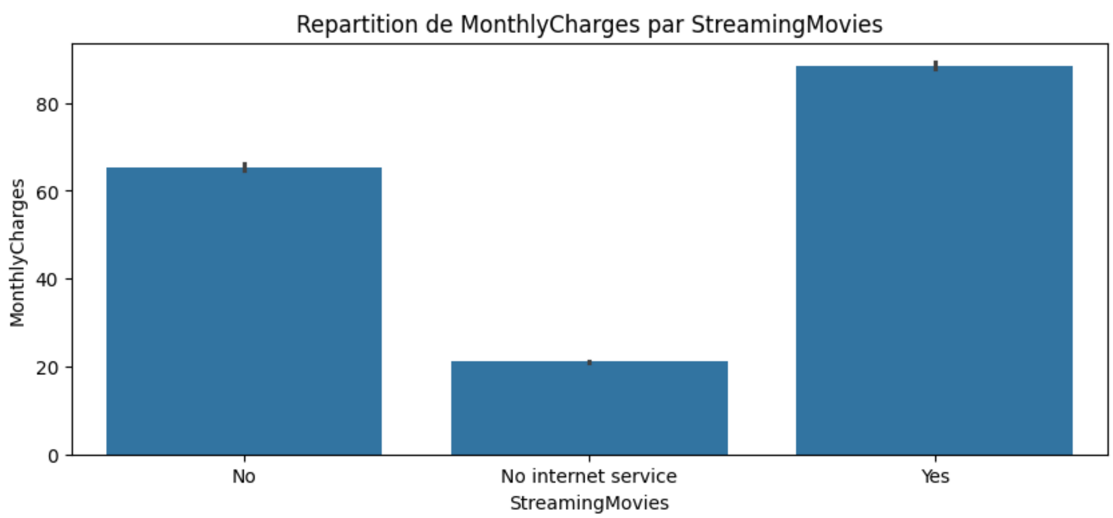

Repartition de MonthlyCharges par OnlineSecurity
MonthlyCharges
0
10
20
30
40
50
60
70
80
No
No internet service
Yes
OnlineSecurity

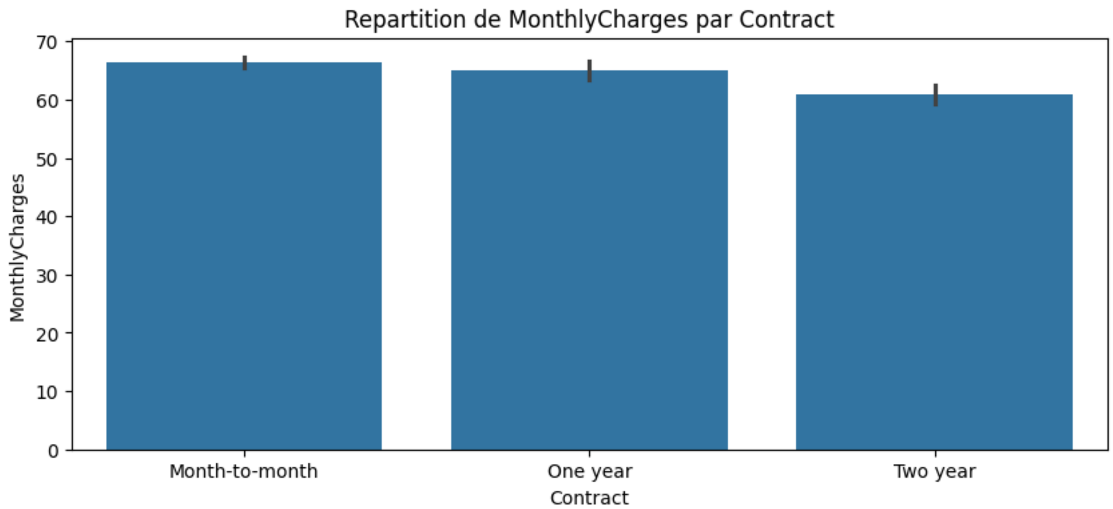

Repartition de MonthlyCharges par OnlineBackup
MonthlyCharges
0
10
20
30
40
50
60
70
80
No
No internet service
Yes
OnlineBackup


Repartition de MonthlyCharges par DeviceProtection
MonthlyCharges
0
10
20
30
40
50
60
70
80
No
No internet service
Yes
DeviceProtection


Repartition de MonthlyCharges par TechSupport
MonthlyCharges
0
10
20
30
40
50
60
70
80
No
No internet service
Yes
TechSupport


Repartition de MonthlyCharges par StreamingTV
MonthlyCharges
0
20
40
60
80
No
No internet service
Yes
StreamingTV


Repartition de MonthlyCharges par StreamingMovies
MonthlyCharges
0
20
40
60
80
No
No internet service
Yes
StreamingMovies



[ ]:
```python
# Tableaux croisés
for i in var_quali_g:
    for j in var_quali_g:
        if i != j:
            tab = pd.crosstab(churn[i], churn[j])
            chi2_stat, p_value, dof, expected = stat.chi2_contingency((tab))
            print(f" Tableau croisé entre {i} et {j} : \n {tabulate(tab,␣
↪headers='keys', tablefmt='pipe')}")
            print(f"\n Statistique de Khideux : {chi2_stat:.4f}")
            print(f"P-value : {p_value:.4f}, degré de liberté : {dof}")
            print(f"Tableau des valeurs attendues : {expected[0][0]:.4f} \n")
            print("-----------------------------------------------------------")
```

0.4 ANALYSE MULTIVARIEE

Cherche à comprendre les relations multiples qui existe entre les différentes variables.

Identifier les variables utiles et inutiles

1. Encodage et normalisation

a. Encodage des variables catégorielles

[16]:
```
binaire_list = ['Partner', 'Dependents',
       'PhoneService', 'PaperlessBilling', 'Churn']
for i in binaire_list:
    churn[i] = churn[i].replace(
        {
            'No' : 0, 'Yes' : 1
        }
    )

churn["gender"] = churn["gender"].replace(
    {
       'Female' : 0 , 'Male': 1
    }
)

# Get dummieser les variables categorielles
col_demmies = pd.get_dummies(churn[[ 'MultipleLines', 'InternetService',␣
 ↪'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport',␣
 ↪'StreamingTV','StreamingMovies', 'Contract', 'PaymentMethod'
  ]], prefix=['Mul', 'IntS', 'OnlS', 'OnlB', 'DevP', 'TechS', 'StrTV', 'StrM',␣
 ↪'Con', 'PayM'], prefix_sep='_', dtype=int)
churn_encoded = pd.concat([churn, col_demmies], axis = 1)

# Suppression des variables initiales
for col in ['MultipleLines', 'InternetService', 'OnlineSecurity','OnlineBackup',␣
 ↪'DeviceProtection', 'TechSupport', 'StreamingTV','StreamingMovies',␣
 ↪'Contract', 'PaymentMethod']:
      churn_encoded.drop(columns=col, axis=1, inplace=True)
```

b. Normalisation des variables numériques

[17]:
```
churn_encoded[['tenure', 'MonthlyCharges', 'TotalCharges']] =␣
 ↪(churn_encoded[['tenure', 'MonthlyCharges', 'TotalCharges']] -␣
 ↪churn_encoded[['tenure', 'MonthlyCharges', 'TotalCharges']].mean())/
 ↪churn_encoded[['tenure', 'MonthlyCharges', 'TotalCharges']].std()



##    Selection du set des features et du target pour l'ACP et la modélisation
features = churn_encoded.drop(columns=["Churn"], axis = 1)
```

```python
target = churn_encoded["Churn"]

churn_encoded.head(5)
```

[17]:
```
   gender  SeniorCitizen  Partner  Dependents    tenure  PhoneService  \
0       0              0        1           0 -1.280157             0
1       1              0        0           0  0.064298             1
2       1              0        0           0 -1.239416             1
3       1              0        0           0  0.512450             0
4       0              0        0           0 -1.239416             1

   PaperlessBilling  MonthlyCharges  TotalCharges  Churn  ...  StrM_No  \
0                 1       -1.161611     -0.994123      0  ...        1
1                 0       -0.260859     -0.173727      0  ...        1
2                 1       -0.363897     -0.959581      1  ...        1
3                 0       -0.747797     -0.195234      0  ...        1
4                 1        0.196164     -0.940391      1  ...        1

   StrM_No internet service  StrM_Yes  Con_Month-to-month  Con_One year  \
0                         0         0                   1             0
1                         0         0                   0             1
2                         0         0                   1             0
3                         0         0                   0             1
4                         0         0                   1             0

   Con_Two year  PayM_Bank transfer (automatic)  PayM_Credit card (automatic)  \
0             0                               0                             0
1             0                               0                             0
2             0                               0                             0
3             0                               1                             0
4             0                               0                             0

   PayM_Electronic check  PayM_Mailed check
0                      1                  0
1                      0                  1
2                      0                  1
3                      0                  0
4                      1                  0

[5 rows x 41 columns]
```

c. Mise en place de l'Analyse des composantes principales

- Valeur propre
- Coeficient des vecteurs propres
- Variance expliquée (et cumulée)

[18]:
```
# Instancier l'ACP
pca = PCA()

# Redimmensionnalisation
features_pca = pca.fit_transform(features)

composants = pd.DataFrame(
    {
        "Dimension" : ["PCA" + str(x+1) for x in range(features.shape[1])],
        "Valeur Propre" : np.round(pca.explained_variance_, 2),
        "Taux de variance_expliquee" : np.round(pca.
 ↪explained_variance_ratio_*100, 3),
        "Taux de variance_expliquee_cum": np.round(np.cumsum(pca.
 ↪explained_variance_ratio_)*100)
    }
)

print(np.sum(composants["Valeur Propre"]))
composants.head(5)
```

```
10.55
```

[18]:
```
  Dimension  Valeur Propre  Taux de variance_expliquee  \
0      PCA1           3.30                      31.285
1      PCA2           2.15                      20.323
2      PCA3           0.79                       7.469
3      PCA4           0.45                       4.255
4      PCA5           0.37                       3.473

   Taux de variance_expliquee_cum
0                            31.0
1                            52.0
2                            59.0
3                            63.0
4                            67.0
```

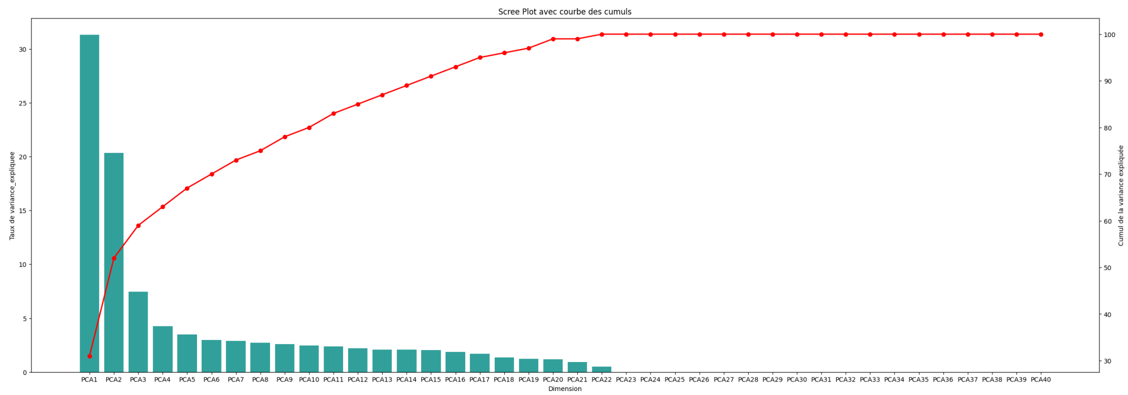
[19]:
```
# Calcul des indicateurs utiles pour l'analyse
dim = composants.loc[:, "Dimension"]
variance_rate = composants.loc[:,"Taux de variance_expliquee"]
variance_rate_cum = composants.loc[:, "Taux de variance_expliquee_cum"]

# Representation graphique
fig, ax = plt.subplots(figsize=(30,10))
sns.barplot(x = dim, y = variance_rate, palette = ["lightseagreen"])
ax.set_title('Scree Plot avec courbe des cumuls')
ax2 = ax.twinx()
```

```
ax2.plot(dim, variance_rate_cum, color='red', marker='o', linestyle='-',␣
↪linewidth=2)
ax2.set_ylabel('Cumul de la variance expliquée')


# Impact de variables initiales sur les composantes
vecteur_propre = pd.DataFrame(
    pca.components_.T,
    columns=["PCA" + str(x+1) for x in range(features.shape[1])],
    index = features.columns.to_list())

# Tri des variables par leur contribution absolue maximale à une composante␣
↪principale
vecteur_propre_sorted = vecteur_propre.abs().max(axis=1).
↪sort_values(ascending=False).index
vecteur_propre = vecteur_propre.loc[vecteur_propre_sorted]

# Affichage du tableau croisé des 5 premières variables les plus contributives
vecteur_propre.head(5)
```

[19]:

```
                        PCA1      PCA2      PCA3      PCA4      PCA5      PCA6  \
gender             -0.002201  0.001311  0.007349  0.011904 -0.030505  0.047795
PaperlessBilling    0.085362 -0.087014 -0.093774  0.044316 -0.082861 -0.072235
PayM_Mailed check -0.097597  0.023169  0.037847 -0.106937  0.114336 -0.040110
TotalCharges        0.478086  0.293637 -0.014822  0.097496  0.096553  0.021652
Con_One year        0.020571  0.065260  0.043232 -0.064796  0.043498  0.028824

                        PCA7      PCA8      PCA9     PCA10  ...         PCA31  \
gender             -0.075246 -0.070240  0.039182  0.167645  ... -0.000000e+00
PaperlessBilling   -0.099479 -0.004853 -0.107165  0.027239  ... -6.729021e-17
PayM_Mailed check  0.124201  0.039667  0.055057  0.036159  ... -8.031081e-02
TotalCharges       -0.044260 -0.107939 -0.074205 -0.021081  ...  1.526314e-15
Con_One year       -0.116846 -0.069906 -0.002615 -0.101311  ... -7.964832e-02

                           PCA32         PCA33         PCA34         PCA35  \
gender              0.000000e+00  0.000000e+00 -0.000000e+00  0.000000e+00
PaperlessBilling    1.516419e-17  5.425967e-17  3.048164e-17 -4.603101e-16
PayM_Mailed check -1.601888e-01 -6.446354e-02 -1.610111e-01 -2.933567e-01
TotalCharges        1.133208e-15  1.016395e-18  7.927889e-16 -2.961900e-16
Con_One year       -2.322479e-01  2.182405e-01  4.158779e-02 -6.083954e-02

                           PCA36         PCA37         PCA38         PCA39  \
gender              0.000000e+00  5.010264e-16 -0.000000e+00  0.000000e+00
PaperlessBilling    1.331327e-16 -5.438219e-16 -1.865972e-16  5.915279e-17
PayM_Mailed check  1.114551e-01  4.883400e-02  1.643730e-02 -1.171908e-01
TotalCharges        8.127488e-16  1.377681e-15 -2.075235e-16 -8.971792e-16
Con_One year        6.595136e-02  1.909235e-01  6.716904e-02  7.015759e-02
```

```
                            PCA40
gender               0.000000e+00
PaperlessBilling     1.195458e-16
PayM_Mailed check    2.812664e-02
TotalCharges        -9.807872e-16
Con_One year        -6.720833e-02

[5 rows x 40 columns]
```



[20]:

```python
# Contribution de chaque variable
vecteur = pca.components_.T

valeur = pca.explained_variance_
contribution = (vecteur**2)*valeur
contrib_percent = contribution / valeur * 100

column_names = [f'PC{i+1}_contrib' for i in range(features.shape[1])]
variable_contrib_df = pd.DataFrame(contrib_percent, columns=column_names,␣
 ↪index=features.columns.to_list())
variable_contrib_df
```

[20]:

```
                           PC1_contrib  PC2_contrib  PC3_contrib  \
gender                        0.000484     0.000172     0.005401
SeniorCitizen                 0.153520     0.174191     0.454570
Partner                       0.431457     1.351254     0.191480
Dependents                    0.005451     0.795321     0.529569
tenure                       10.335350    24.363572     2.832915
PhoneService                  0.007165     0.026688     3.020857
PaperlessBilling              0.728661     0.757137     0.879359
MonthlyCharges               24.595664     3.684099     8.638960
TotalCharges                 22.856642     8.622259     0.021969
Mul_No                        1.765854     0.023900     0.025864
```

Mul_No phone service	0.007165	0.026688	3.020857
Mul_Yes	1.997976	0.101100	2.487678
IntS_DSL	0.014131	0.181067	20.872783
IntS_Fiber optic	2.429978	1.571940	10.179632
IntS_No	2.814725	2.820013	1.899216
OnlS_No	0.411670	4.722400	0.687591
OnlS_No internet service	2.814725	2.820013	1.899216
OnlS_Yes	1.073502	0.243857	4.872314
OnlB_No	0.062348	4.235688	0.298866
OnlB_No internet service	2.814725	2.820013	1.899216
OnlB_Yes	2.039234	0.143481	0.691283
DevP_No	0.028958	4.620245	1.124124
DevP_No internet service	2.814725	2.820013	1.899216
DevP_Yes	2.272690	0.221075	0.101044
TechS_No	0.307521	4.990889	0.148110
TechS_No internet service	2.814725	2.820013	1.899216
TechS_Yes	1.261508	0.307737	3.108067
StrTV_No	0.000279	3.014005	7.450112
StrTV_No internet service	2.814725	2.820013	1.899216
StrTV_Yes	2.871086	0.003226	1.826197
StrM_No	0.000795	3.053100	7.003241
StrM_No internet service	2.814725	2.820013	1.899216
StrM_Yes	2.910111	0.004627	1.608441
Con_Month-to-month	0.295624	6.297274	1.277990
Con_One year	0.042317	0.425886	0.186898
Con_Two year	0.114245	3.447847	0.487434
PayM_Bank transfer (automatic)	0.093566	0.351969	0.200684
PayM_Credit card (automatic)	0.081353	0.375783	0.340331
PayM_Electronic check	0.148114	2.067753	1.987622
PayM_Mailed check	0.952508	0.053679	0.143243

	PC4_contrib	PC5_contrib	PC6_contrib	\
gender	0.014169	0.093054	0.228434	
SeniorCitizen	0.723362	0.374846	0.188119	
Partner	0.811096	0.178620	6.250789	
Dependents	0.001789	0.048860	4.628688	
tenure	10.674976	1.380016	1.514017	
PhoneService	0.325270	6.295524	1.170564	
PaperlessBilling	0.196391	0.686595	0.521795	
MonthlyCharges	1.596343	5.783736	0.205995	
TotalCharges	0.950556	0.932242	0.046879	
Mul_No	5.060840	22.374222	4.947464	
Mul_No phone service	0.325270	6.295524	1.170564	
Mul_Yes	7.952149	4.933070	1.304994	
IntS_DSL	3.512488	3.444981	1.007788	
IntS_Fiber optic	2.360018	3.363727	1.169140	
IntS_No	0.114196	0.000485	0.005988	

OnlS_No	0.462299	5.729053	0.415065
OnlS_No internet service	0.114196	0.000485	0.005988
OnlS_Yes	1.036027	5.624130	0.520759
OnlB_No	0.265098	5.231851	36.352512
OnlB_No internet service	0.114196	0.000485	0.005988
OnlB_Yes	0.031311	5.131606	35.425387
DevP_No	5.224174	0.564231	0.012409
DevP_No internet service	0.114196	0.000485	0.005988
DevP_Yes	6.883138	0.597795	0.035636
TechS_No	6.025590	1.495702	0.054088
TechS_No internet service	0.114196	0.000485	0.005988
TechS_Yes	7.798816	1.442329	0.024083
StrTV_No	7.500731	3.133729	0.149503
StrTV_No internet service	0.114196	0.000485	0.005988
StrTV_Yes	9.465926	3.212172	0.095651
StrM_No	7.840798	3.884606	0.089709
StrM_No internet service	0.114196	0.000485	0.005988
StrM_Yes	9.847488	3.971888	0.049343
Con_Month-to-month	0.248520	0.001390	1.051290
Con_One year	0.419854	0.189208	0.083080
Con_Two year	0.022334	0.158166	0.543299
PayM_Bank transfer (automatic)	0.280056	0.020858	0.290005
PayM_Credit card (automatic)	0.011091	0.024999	0.077403
PayM_Electronic check	0.189100	2.090605	0.172753
PayM_Mailed check	1.143563	1.307269	0.160880

	PC7_contrib	PC8_contrib	PC9_contrib \
gender	0.566191	0.493370	0.153525
SeniorCitizen	0.098498	0.739045	0.029636
Partner	4.821240	28.218155	9.411988
Dependents	2.202482	28.387801	7.960987
tenure	4.290224	3.158450	3.099347
PhoneService	0.310034	0.000105	0.008769
PaperlessBilling	0.989602	0.002355	1.148424
MonthlyCharges	1.312752	0.455312	0.217671
TotalCharges	0.195895	1.165081	0.550642
Mul_No	17.611058	1.904024	1.135682
Mul_No phone service	0.310034	0.000105	0.008769
Mul_Yes	22.594432	1.875831	1.344041
IntS_DSL	0.089435	0.147766	0.060795
IntS_Fiber optic	0.006161	0.133381	0.091131
IntS_No	0.048647	0.000368	0.003060
OnlS_No	9.866638	1.567390	0.010504
OnlS_No internet service	0.048647	0.000368	0.003060
OnlS_Yes	8.529667	1.615811	0.002225
OnlB_No	2.407936	1.584811	1.040235
OnlB_No internet service	0.048647	0.000368	0.003060

OnlB_Yes	3.141096	1.633499	1.156123
DevP_No	0.146582	9.439726	28.851271
DevP_No internet service	0.048647	0.000368	0.003060
DevP_Yes	0.026341	9.322169	29.448539
TechS_No	6.691021	0.270236	3.017613
TechS_No internet service	0.048647	0.000368	0.003060
TechS_Yes	5.598617	0.290557	2.828501
StrTV_No	0.611900	2.345746	1.936652
StrTV_No internet service	0.048647	0.000368	0.003060
StrTV_Yes	1.005611	2.404900	1.785761
StrM_No	0.120265	0.446981	1.883564
StrM_No internet service	0.048647	0.000368	0.003060
StrM_Yes	0.321890	0.473010	1.734797
Con_Month-to-month	0.357280	0.235680	0.222812
Con_One year	1.365293	0.488678	0.000684
Con_Two year	0.325731	0.045619	0.198804
PayM_Bank transfer (automatic)	0.003560	0.029825	0.175850
PayM_Credit card (automatic)	0.075618	0.635839	0.043944
PayM_Electronic check	2.123809	0.328816	0.116177
PayM_Mailed check	1.542577	0.157347	0.303122

	PC10_contrib	...	PC31_contrib	PC32_contrib	\
gender	2.810498	...	0.000000e+00	NaN	
SeniorCitizen	0.295065	...	2.564657e-30	NaN	
Partner	0.721640	...	2.602885e-30	NaN	
Dependents	1.266486	...	4.066236e-30	NaN	
tenure	0.035982	...	8.645880e-29	NaN	
PhoneService	0.350532	...	1.115430e+01	NaN	
PaperlessBilling	0.074198	...	4.527972e-31	NaN	
MonthlyCharges	0.083813	...	1.322515e-25	NaN	
TotalCharges	0.044440	...	2.329636e-28	NaN	
Mul_No	0.016493	...	9.951982e-03	NaN	
Mul_No phone service	0.350532	...	1.183061e+01	NaN	
Mul_Yes	0.519094	...	9.951982e-03	NaN	
IntS_DSL	0.174040	...	2.488423e+00	NaN	
IntS_Fiber optic	0.210641	...	2.488423e+00	NaN	
IntS_No	0.001745	...	5.245033e-01	NaN	
OnlS_No	22.760982	...	1.080584e+01	NaN	
OnlS_No internet service	0.001745	...	1.747783e-01	NaN	
OnlS_Yes	22.364117	...	1.080584e+01	NaN	
OnlB_No	0.002174	...	7.274116e-02	NaN	
OnlB_No internet service	0.001745	...	3.473872e-01	NaN	
OnlB_Yes	0.007816	...	7.274116e-02	NaN	
DevP_No	0.002217	...	6.477849e-01	NaN	
DevP_No internet service	0.001745	...	1.332494e-01	NaN	
DevP_Yes	0.007897	...	6.477849e-01	NaN	
TechS_No	18.981616	...	2.617166e-01	NaN	

```
TechS_No internet service         0.001745  ...  5.111076e+00          NaN
TechS_Yes                        19.347377  ...  2.617166e-01          NaN
StrTV_No                          0.325850  ...  2.152269e-01          NaN
StrTV_No internet service         0.001745  ...  2.494863e-01          NaN
StrTV_Yes                         0.279902  ...  2.152269e-01          NaN
StrM_No                           1.906580  ...  1.173683e+01          NaN
StrM_No internet service          0.001745  ...  1.351449e+01          NaN
StrM_Yes                          1.792958  ...  1.173683e+01          NaN
Con_Month-to-month                0.001822  ...  6.343855e-01          NaN
Con_One year                      1.026401  ...  6.343855e-01          NaN
Con_Two year                      1.114718  ...  6.343855e-01          NaN
PayM_Bank transfer (automatic)    0.573622  ...  6.449827e-01          NaN
PayM_Credit card (automatic)      0.147696  ...  6.449827e-01          NaN
PayM_Electronic check             2.259840  ...  6.449827e-01          NaN
PayM_Mailed check                 0.130744  ...  6.449827e-01          NaN

                               PC33_contrib  PC34_contrib  PC35_contrib  \
gender                                  NaN           NaN           NaN
SeniorCitizen                           NaN           NaN           NaN
Partner                                 NaN           NaN           NaN
Dependents                              NaN           NaN           NaN
tenure                                  NaN           NaN           NaN
PhoneService                            NaN           NaN           NaN
PaperlessBilling                        NaN           NaN           NaN
MonthlyCharges                          NaN           NaN           NaN
TotalCharges                            NaN           NaN           NaN
Mul_No                                  NaN           NaN           NaN
Mul_No phone service                    NaN           NaN           NaN
Mul_Yes                                 NaN           NaN           NaN
IntS_DSL                                NaN           NaN           NaN
IntS_Fiber optic                        NaN           NaN           NaN
IntS_No                                 NaN           NaN           NaN
OnlS_No                                 NaN           NaN           NaN
OnlS_No internet service                NaN           NaN           NaN
OnlS_Yes                                NaN           NaN           NaN
OnlB_No                                 NaN           NaN           NaN
OnlB_No internet service                NaN           NaN           NaN
OnlB_Yes                                NaN           NaN           NaN
DevP_No                                 NaN           NaN           NaN
DevP_No internet service                NaN           NaN           NaN
DevP_Yes                                NaN           NaN           NaN
TechS_No                                NaN           NaN           NaN
TechS_No internet service               NaN           NaN           NaN
TechS_Yes                               NaN           NaN           NaN
StrTV_No                                NaN           NaN           NaN
StrTV_No internet service               NaN           NaN           NaN
StrTV_Yes                               NaN           NaN           NaN
```

StrM_No	NaN	NaN	NaN
StrM_No internet service	NaN	NaN	NaN
StrM_Yes	NaN	NaN	NaN
Con_Month-to-month	NaN	NaN	NaN
Con_One year	NaN	NaN	NaN
Con_Two year	NaN	NaN	NaN
PayM_Bank transfer (automatic)	NaN	NaN	NaN
PayM_Credit card (automatic)	NaN	NaN	NaN
PayM_Electronic check	NaN	NaN	NaN
PayM_Mailed check	NaN	NaN	NaN

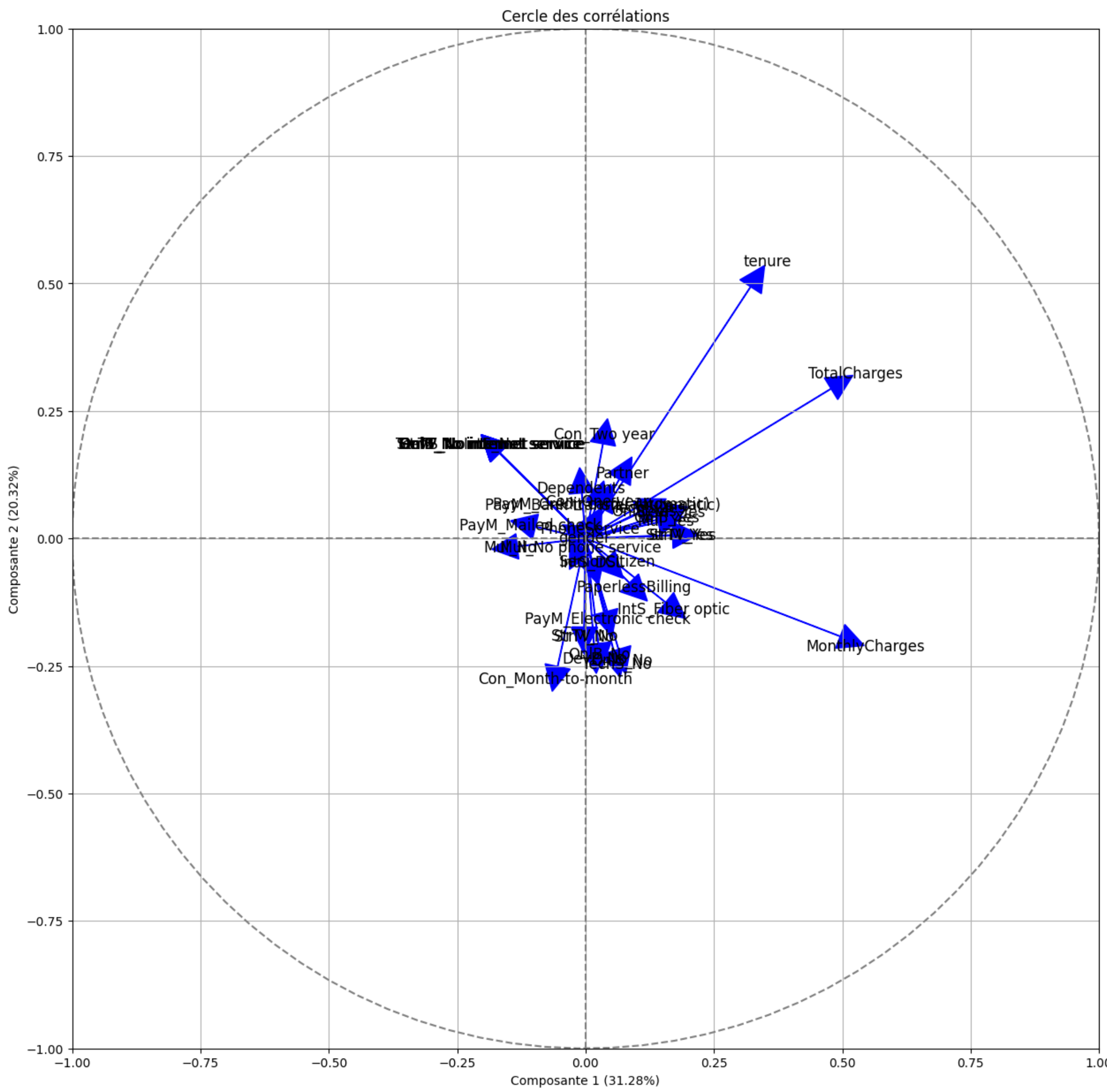
	PC36_contrib	PC37_contrib	PC38_contrib \
gender	NaN	NaN	NaN
SeniorCitizen	NaN	NaN	NaN
Partner	NaN	NaN	NaN
Dependents	NaN	NaN	NaN
tenure	NaN	NaN	NaN
PhoneService	NaN	NaN	NaN
PaperlessBilling	NaN	NaN	NaN
MonthlyCharges	NaN	NaN	NaN
TotalCharges	NaN	NaN	NaN
Mul_No	NaN	NaN	NaN
Mul_No phone service	NaN	NaN	NaN
Mul_Yes	NaN	NaN	NaN
IntS_DSL	NaN	NaN	NaN
IntS_Fiber optic	NaN	NaN	NaN
IntS_No	NaN	NaN	NaN
OnlS_No	NaN	NaN	NaN
OnlS_No internet service	NaN	NaN	NaN
OnlS_Yes	NaN	NaN	NaN
OnlB_No	NaN	NaN	NaN
OnlB_No internet service	NaN	NaN	NaN
OnlB_Yes	NaN	NaN	NaN
DevP_No	NaN	NaN	NaN
DevP_No internet service	NaN	NaN	NaN
DevP_Yes	NaN	NaN	NaN
TechS_No	NaN	NaN	NaN
TechS_No internet service	NaN	NaN	NaN
TechS_Yes	NaN	NaN	NaN
StrTV_No	NaN	NaN	NaN
StrTV_No internet service	NaN	NaN	NaN
StrTV_Yes	NaN	NaN	NaN
StrM_No	NaN	NaN	NaN
StrM_No internet service	NaN	NaN	NaN
StrM_Yes	NaN	NaN	NaN
Con_Month-to-month	NaN	NaN	NaN
Con_One year	NaN	NaN	NaN

Con_Two year	NaN	NaN	NaN
PayM_Bank transfer (automatic)	NaN	NaN	NaN
PayM_Credit card (automatic)	NaN	NaN	NaN
PayM_Electronic check	NaN	NaN	NaN
PayM_Mailed check	NaN	NaN	NaN

	PC39_contrib	PC40_contrib
gender	NaN	NaN
SeniorCitizen	NaN	NaN
Partner	NaN	NaN
Dependents	NaN	NaN
tenure	NaN	NaN
PhoneService	NaN	NaN
PaperlessBilling	NaN	NaN
MonthlyCharges	NaN	NaN
TotalCharges	NaN	NaN
Mul_No	NaN	NaN
Mul_No phone service	NaN	NaN
Mul_Yes	NaN	NaN
IntS_DSL	NaN	NaN
IntS_Fiber optic	NaN	NaN
IntS_No	NaN	NaN
OnlS_No	NaN	NaN
OnlS_No internet service	NaN	NaN
OnlS_Yes	NaN	NaN
OnlB_No	NaN	NaN
OnlB_No internet service	NaN	NaN
OnlB_Yes	NaN	NaN
DevP_No	NaN	NaN
DevP_No internet service	NaN	NaN
DevP_Yes	NaN	NaN
TechS_No	NaN	NaN
TechS_No internet service	NaN	NaN
TechS_Yes	NaN	NaN
StrTV_No	NaN	NaN
StrTV_No internet service	NaN	NaN
StrTV_Yes	NaN	NaN
StrM_No	NaN	NaN
StrM_No internet service	NaN	NaN
StrM_Yes	NaN	NaN
Con_Month-to-month	NaN	NaN
Con_One year	NaN	NaN
Con_Two year	NaN	NaN
PayM_Bank transfer (automatic)	NaN	NaN
PayM_Credit card (automatic)	NaN	NaN
PayM_Electronic check	NaN	NaN
PayM_Mailed check	NaN	NaN

[40 rows x 40 columns]

[21]:
```
# Labels des variables
variables = features.columns.to_list()

vecteur = pca.components_.T
# Cercle unité
theta = np.linspace(0, 2*np.pi, 100)
plt.figure(figsize=(15,15))
plt.plot(np.cos(theta), np.sin(theta), color='gray', linestyle="--")  # Cercle

# Ajout des vecteurs
for i, var in enumerate(variables):
    plt.arrow(0, 0, vecteur[i, 0], vecteur[i, 1],
              head_width=0.05, head_length=0.05, color='b')
     # Positionnement du texte (ajusté dynamiquement)
    text_x = vecteur[i, 0] * 1.1
    text_y = vecteur[i, 1] * 1.1
    plt.text(text_x, text_y, var, color='black', fontsize=12, ha='center',␣
 ↪va='center')

# Axes
plt.axhline(0, color='gray', linestyle='--')
plt.axvline(0, color='gray', linestyle='--')

# Configuration du graphique
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.xlabel(f"Composante 1 ({round(pca.explained_variance_ratio_[0] * 100, 2)}%)")
plt.ylabel(f"Composante 2 ({round(pca.explained_variance_ratio_[1] * 100, 2)}%)")
plt.title("Cercle des corrélations")
plt.grid()
plt.show()
```



0.5 CONTRUCTION DU MODELE DE ML

1. Elaboration du modèle avec les nouvelles variables

[22]:
```
feat = churn.drop(columns="Churn")
targ = churn["Churn"]
# Selection des variables par types
num_feature = feat.select_dtypes(include = float).columns
cat_feature = ['MultipleLines', 'InternetService', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'Contract', 'PaymentMethod']

# Preprocessing
prepros = ColumnTransformer(
```

```python
    transformers=[
        ("cat", OneHotEncoder(handle_unknown='ignore'), cat_feature),
        ("normalizer", StandardScaler(), num_feature)
])

# Pipeline

pipe = Pipeline([
    ("preprocessing", prepros),
    ("pca", PCA(n_components=5)),
    ("model_log", LogisticRegression(solver='liblinear', random_state=1))
])

# Train test split

X_train, X_test, y_train, y_test = train_test_split(feat, targ, test_size=0.2,␣
 ↪random_state=1 )

print(f"la taille du feature d'entrainement :\n {X_train.shape} \n")
print(f"la taille du target d'entrainement :\n {y_train.shape} \n")
print(f"la taille du feature de test :\n {X_test.shape} \n")
print(f"la taille du target de test :\n {y_test.shape} \n")
```

```
la taille du feature d'entrainement :
 (5625, 19)

la taille du target d'entrainement :
 (5625,)

la taille du feature de test :
 (1407, 19)

la taille du target de test :
 (1407,)
```

2. Entrainement du modèle

[23]:
```python
# Entrainement
pipe.fit(X = X_train, y = y_train)
```

[23]:
```
Pipeline(steps=[('preprocessing',
                 ColumnTransformer(transformers=[('cat',
OneHotEncoder(handle_unknown='ignore'),
                                                  ['MultipleLines',
                                                   'InternetService',
                                                   'OnlineSecurity',
                                                   'OnlineBackup',
```

```
                                                       'DeviceProtection',
                                                       'TechSupport', 'StreamingTV',
                                                       'StreamingMovies',
                                                       'Contract',
                                                       'PaymentMethod']),
                                                     ('normalizer',
                                                      StandardScaler(),
                                                      Index(['MonthlyCharges',
'TotalCharges'], dtype='object'))])),
                ('pca', PCA(n_components=5)),
                ('model_log',
                 LogisticRegression(random_state=1, solver='liblinear'))])
```

3. Prédiction

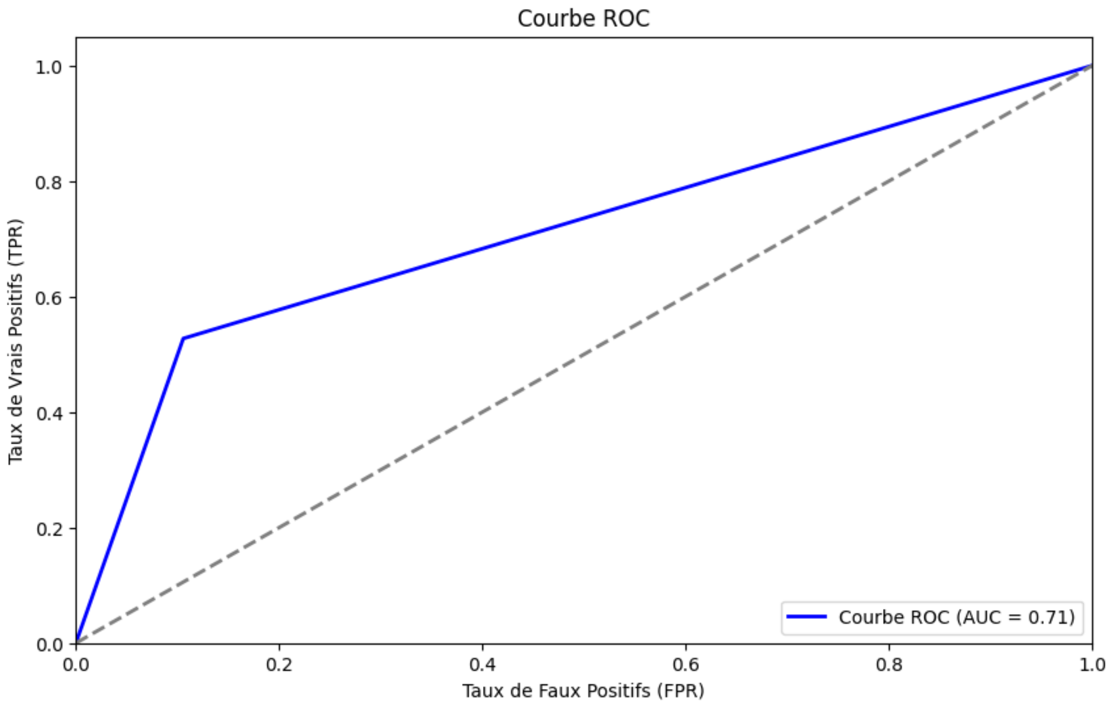
[24]:
```
y_pred = pipe.predict(X_test)
```

4. Evaluation du modèle

[25]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.84      0.89      0.87      1041
           1       0.64      0.53      0.58       366

    accuracy                           0.80      1407
   macro avg       0.74      0.71      0.72      1407
weighted avg       0.79      0.80      0.79      1407
```

[26]:
```
# Courbe ROC
# Calculer les métriques ROC
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)
# Tracer la courbe ROC
plt.figure(figsize=(10, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label=f'Courbe ROC (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='gray', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Taux de Faux Positifs (FPR)')
plt.ylabel('Taux de Vrais Positifs (TPR)')
plt.title('Courbe ROC')
plt.legend(loc="lower right")
plt.show()
```



5. Optimisation des hyperparamètre avec le GridSearch

[ ]:
```
X_train1, X_test1, y_train1, y_test1 = train_test_split(churn_encoded.
↪drop(columns="Churn"), churn_encoded["Churn"], test_size=0.2, random_state=1 )

# Définition des hyperparamètres à tester
param_grid = {
    'penalty': ['l1', 'l2'],  # Type de régularisation
    'solver': ['liblinear', 'saga']  # Solveurs compatibles
}

# Pipeline

model = LogisticRegression(max_iter=500)

# GridSearchCV avec validation croisée 5-fold
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy',␣
↪n_jobs=-1)

# Entraînement
grid_search.fit(X_train1, y_train1)

# Meilleurs hyperparamètres
print("Meilleurs paramètres :", grid_search.best_params_)
```

```python
print("Meilleure performance :", grid_search.best_score_)
```

```
Meilleurs paramètres : {'penalty': 'l1', 'solver': 'saga'}
Meilleure performance : 0.8053333333333332
```