Institut Supérieur de Sciences Appliquées et de Technologies
De Mateur

Tunisie Telecom

Rapport de Stage Ouvrier

Réalisé par

Argoubi Ahmed

Encadrée par

mr.Ali Laabidi

Période de stage: 01/08/2023 a 30/08/2023

Remerciement

Au terme de ce stage , Je tiens vraiment à remercier Mr.Ali de m'avoir appris tellement de choses sur le machine learning ,grâce à ses explications claires, j'ai pu comprendre des concepts compliqués et découvrir un nouvel univers.

Je suis reconnaissant pour cette opportunité d'apprendre et de grandir sous sa guidance experte.

Table des Matières

Introduction générale

Ce rapport présente mon stage au sein de Tunisie Telecom sur le domaine du machine learning.

Au cours de cette période, j'ai eu l'occasion d'explorer le monde professionnel du Machine Learning et d'acquérir une compréhension des activités et des processus qui y sont liés.

Encadré par mr Ali , j'ai eu l'opportunité de mettre en pratique les compétences techniques et les connaissances acquises lors de ma formation académique.

Chapter 1

Présentation de l'entreprise

1.1 Introduction :

Dans ce chapitre , nous allons présenter en premier lieu le société dans laquelle nous avons effectue notre sujet de travail propose tout en expliquant ses objectifs ainsi les fonctionnalités nécessaires pour le réaliser

1.2 Présentation de Tunisie Telecom :

Tunisie Telecom est l'opérateur national de télécommunications de la Tunisie, établi en 1995 et détenu majoritairement par l'État tunisien. Il exerce son activité dans le domaine des télécommunications, fournissant une gamme étendue de services de communication aux particuliers, aux entreprises et aux institutions gouvernementales Ces services englobent la téléphonie fixe et mobile, l'accès à Internet, la transmission de données, les services à valeur ajoutée et d'autres solutions de communication avancées.

1.3 Contexte de stage :

Le contexte de stage consiste en la prédiction des taux de désabonnement d'un opérateur de télécommunication. L'objectif de ce projet est de développer un modèle de prédiction efficace qui permettra à l'opérateur de mieux comprendre les tendances de désabonnement parmi ses abonnés. En utilisant des techniques de machine learning et d'analyse de données, l'objectif est d'identifier les facteurs clés qui influencent les clients à se désabonner et de créer un modèle capable de prédire ces désabonnements avec une précision significative.

1.4 Conclusion:

Dans ce chapitre , nous avons présente le cadre du projet , identifiée a atteindre et le travail a faire Compte tenu de cela , nous proposant les détails du travail dans le chapitre suivant

Chapter 2

Les fondations du Machine Learning

2.1 pourquoi le Machine Learning est utilisé :

Pour comprendre au mieux ce qu'est le Machine Learning et comment cela fonctionne, il faut commencer par comprendre pourquoi il est utilisé. Le Machine Learning est utilisé pour automatiser des tâches, faire des prédictions précises et extraire des informations utiles à partir de données complexes, ce qui trouve des applications dans des domaines tels que la santé, la finance, la personnalisation des services et l'analyse de texte/vidéo.

2.2 Comment fonctionne le Machine Learning :

Le Machine Learning est une technologie permettant aux ordinateurs d'apprendre sans avoir été programmés à cet effet. Alors , Au lieu d'être explicitement programmés, les algorithmes apprennent des modèles, des relations et des informations à partir des données et utilisent ces connaissances pour faire des prédictions éclairées ou prendre des mesures.

2.3 Les méthodes d'apprentissage :

Pour donner à un ordinateur la capacité d'apprendre, on utilise des méthodes d'apprentissage qui sont fortement inspirées de la façon dont nous, les êtres humains, apprenons à faire des choses. Parmi ces méthodes, on compte :
• L'apprentissage supervisé (Supervised Learning)
• L'apprentissage non supervisé (Unsupervised Learning)
• L'apprentissage par renforcement (Reinforcement Learning)
L'apprentissage supervisé est la méthode la plus utilisée en machine learning, et c'est la méthode que j'ai utilisée dans mon projet, nous en parlerons donc dans le chapitre suivant

2.4 Qu'est-ce que le Machine Learning Supervisé:

Le machine learning supervisé est une technologie élémentaire mais stricte. Les opérateurs présentent à l'ordinateur des exemples d'entrées et les sorties souhaitées, et l'ordinateur recherche des solutions pour obtenir ces sorties en fonction de ces entrées.
Le but est que l'ordinateur apprenne la règle générale qui mappe les entrées et les sorties.

Le machine learning supervisé peut être utilisé pour faire des prédictions sur des données indisponibles ou futures (on parle alors de "modélisation prédictive").
L'algorithme essaie de développer une fonction qui prédit avec précision la sortie à partir des variables d'entrée – par exemple, prédire la valeur d'un bien immobilier (sortie) à partir d'entrées telles que le nombre de pièces, l'année de construction, la surface du terrain, l'emplacement, etc.
On parle ainsi d'apprentissage supervisé lorsque l'on fournit à une machine beaucoup d'exemples qu'elle doit étudier.

Pour maîtriser l'apprentissage supervisé, il faut absolument comprendre et connaitre les 4 notions suivantes :
• Le Dataset
• Le Modèle et ses paramètres
• La Fonction Coût
• L'Algorithme d'apprentissage

i. **Le Dataset** :
Le dataset (ensemble de données) est la collection de toutes les données sur lesquelles vous allez entraîner et tester votre modèle. Il est constitué d'exemples d'entrée et de leurs étiquettes (valeurs de sortie attendues).
En Machine Learning, on compile ces exemples (x, y) dans un tableau que l'on appelle Dataset :
• La variable y porte le nom de target (la cible). C'est la valeur que l'on cherche à prédire.
• La variable x porte le nom de feature (facteur). Un facteur influence la valeur de y, et on a en général beaucoup de features (x1, x2, ...) dans notre Dataset que l'on regroupe dans une matrice X.

ii. **Le Modèle :**
Le modèle est la structure mathématique ou algorithmique qui tente d'apprendre la relation entre les entrées et les sorties. Les paramètres du modèle sont les valeurs ajustables qui sont optimisées pendant l'apprentissage pour minimiser la différence entre les prédictions du modèle et les étiquettes réelles dans le dataset.

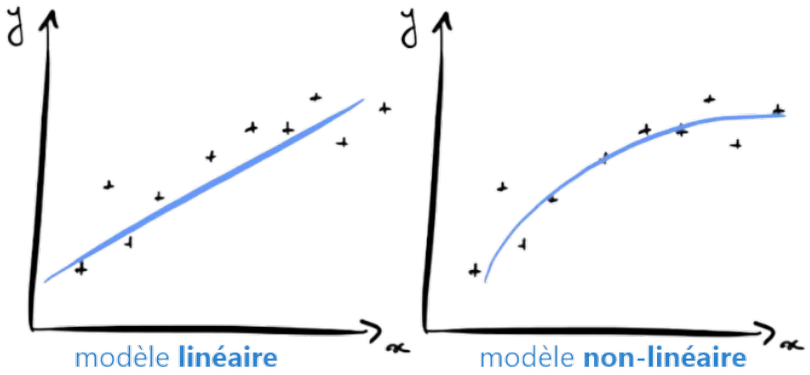


Figure 2.1: Les types de Modèle

iii. **La Fonction Coût :**
Aussi appelée fonction d'erreur ou fonction de perte, elle mesure à quel point les prédictions du modèle diffèrent des étiquettes réelles dans le dataset. L'objectif de l'apprentissage supervisé est de minimiser cette fonction coût en ajustant les paramètres du modèle.

iv. **L'Algorithme d'Apprentissage :**
C'est la méthode utilisée pour ajuster les paramètres du modèle en fonction des données d'entraînement. L'algorithme d'apprentissage recherche généralement les paramètres qui minimisent la fonction coût.

2.4.1 Les applications de l'apprentissage supervisé:

Avec l'apprentissage supervisé on peut développer des modèles pour résoudre 2 types de problèmes :
• Les problèmes de Régression
• Les problèmes de Classification

Dans les problèmes de régression, on cherche à prédire la valeur d'une variable continue, c'est-à-dire une variable qui peut prendre une infinité de valeurs.

Par exemple :
• Prédire le prix d'un appartement (y) selon sa surface habitable (x)

Dans un problème de classification, on cherche à classer un objet dans différentes classes, c'est-à-dire que l'on cherche à prédire la valeur d'une variable discrète (qui ne prend qu'un nombre fini de valeurs).

Par exemple :
• Prédire si un email est un spam (classe y = 1) ou non (classe y = 0) selon le nombre de liens présent dansl'email(x)

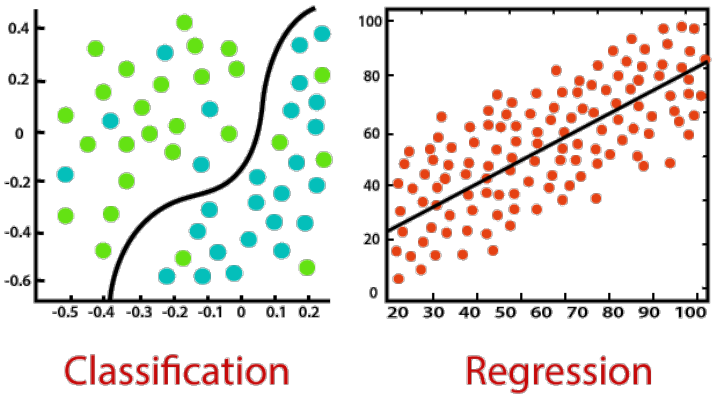


Figure 2.2: Les applications de l'apprentissage supervisé

2.5 Qu'est-ce que le Machine Learning non Supervisé :

L'apprentissage non supervisé est un type d'apprentissage automatique dans lequel l'algorithme apprend des modèles et des structures dans les données sans conseils explicites ni exemples étiquetés. Contrairement à l'apprentissage supervisé, où l'algorithme est fourni avec des paires entrée-sortie étiquetées à partir desquelles apprendre, l'apprentissage non supervisé fonctionne sur des données non étiquetées, cherchant à identifier les relations et les modèles inhérents dans les données elles-mêmes.

Dans l'apprentissage non-supervisé, on dispose ainsi d'un Dataset (x) sans valeur (y), et la machine apprend à reconnaitre des structures dans les données (x) qu'on lui montre. On peut ainsi regrouper des donnés dans des clusters (c'est le Clustering), détecter des anomalies, ou encore réduire la dimension de données très riches en compilant les dimensions ensembles.

2.6 Les 4 librairies à maîtriser pour le Machine Learning:

Il existe de nombreuses bibliothèques utiles pour le machine learning en Python, mais voici quatre des plus importantes et des plus couramment utilisées que vous devriez envisager de maîtriser :

i. **NumPy:** NumPy est une bibliothèque fondamentale pour le calcul numérique en Python. Elle fournit des tableaux multidimensionnels efficaces (appelés ndarray) qui sont essentiels pour la manipulation des données

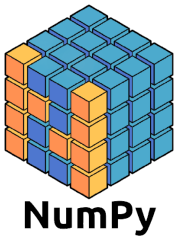


Figure 2.3: NumPy logo

ii. **Matplotlib:** Matplotlib est la librairie qui permet de visualiser nos Datasets, nos fonctions, nos résultats sous forme de graphes, courbes et nuages de points.

Figure 2.4: Matplotlib logo

iii. **Sklearn:** Sklearn est la librairie qui contient toutes les fonctions de l'état de l'art du Machine Learning. On y trouve les algorithmes les plus importants ainsi que diverses fonctions de pre-processing.

Figure 2.5: skLearn logo

iv. **Pandas:** Pandas est une excellente librairie pour importer vos tableaux Excel (et autres formats) dans Python dans le but de tirer des statistiques et de charger votre Dataset dans Sklearn.

Figure 2.6: Pandas logo

Chapter 3

Telecom Churn Predection

3.1 Contexte du projet :

Prédiction de taux de désabonnement d'un opérateur de télécommunication .

3.2 Le travail demandé :

a. Chargement des bibliothèques et des données
b. Comprendre les données
c. Manipulation des données
d. Visualisation des données
e. Prétraitement des données
f. Évaluations et prédictions du modèle d'apprentissage automatique

3.3 L'outil de travail:

Anaconda : Anaconda contient tous les outils et librairies dont vous avez besoin pour faire du Machine Learning : Numpy, Matplotlib, Sklearn, etc.

Figure 3.1: Anaconda logo

3.4 Chargement des bibliothèques et des données:

Nous allons commencer par importer les bibliothèques nécessaires, comme Pandas, NumPy, Matplotlib et Scikit-learn, en utilisant la commande "import ".

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O
import seaborn as sns # For creating plots
import matplotlib.ticker as mtick
import matplotlib.pyplot as plt
```

Ensuite, nous allons charger les données sur lesquelles nous allons travailler. Ces données est stockées dans un fichier CSV .
Nous utiliserons la fonction appropriée de Pandas pour charger les données dans un DataFrame.

```
telecom_cust = pd.read_csv('DATA.csv')
```

3.5 Comprendre les données:

Nous allons explorer le DataFrame pour comprendre la structure des données, les différentes colonnes et leurs types. Nous utiliserons les méthodes et attributs de Pandas.

```
telecom_cust.head()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupp
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

5 rows × 21 columns

Figure 3.2: The DATA

```
telecom_cust.columns.values
OUTPUT ; array(['customerID', 'gender', 'SeniorCitizen',
'Partner','Dependents','tenure', 'PhoneService',
'MultipleLines', 'InternetService', 'OnlineSecurity',
'OnlineBackup', 'DeviceProtection', 'TechSupport'
'StreamingTV', 'StreamingMovies', 'Contract',
'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
'TotalCharges', 'Churn'], dtype=object)
```

```
telecom_cust.dtypes
OUTPUT:
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
```

ensuite nous Explorons les données pour voir s'il y a des valeurs manquantes.

```
telecom_cust.TotalCharges = pd.to_numeric(telecom_cust.Total
Charges, errors='coerce')
telecom_cust.isnull().sum()
```

3.6 Manipulation des données :

Nous allons nettoyer les données en traitant les valeurs manquantes et les identifiants clients .

```
#Removing missing values
telecom_cust.dropna(inplace = True)
#Remove customer IDs from the data set
df2 = telecom_cust.iloc[:,1:]
```

3.7 Prétraitement et Visualisation des données :

Nous pouvons ensuite utiliser certaines fonctions pour nettoyer notre Dataset et convertir les catégories en valeurs numériques binaire.

```
#Convertin the predictor variable in a binary numeric variable
df2['Churn'].replace(to_replace='Yes', value=1, inplace=True)
df2['Churn'].replace(to_replace='No',  value=0, inplace=True)

#Let's convert all the categorical variables into dummy variables
df_dummies = pd.get_dummies(df2)
df_dummies.head()
```

ensuite nous Obtenions la corrélation de Churn avec d'autres variables

```
plt.figure(figsize=(15,8))
df_dummies.corr()['Churn'].sort_values(ascending = False)
.plot(kind='bar')
```

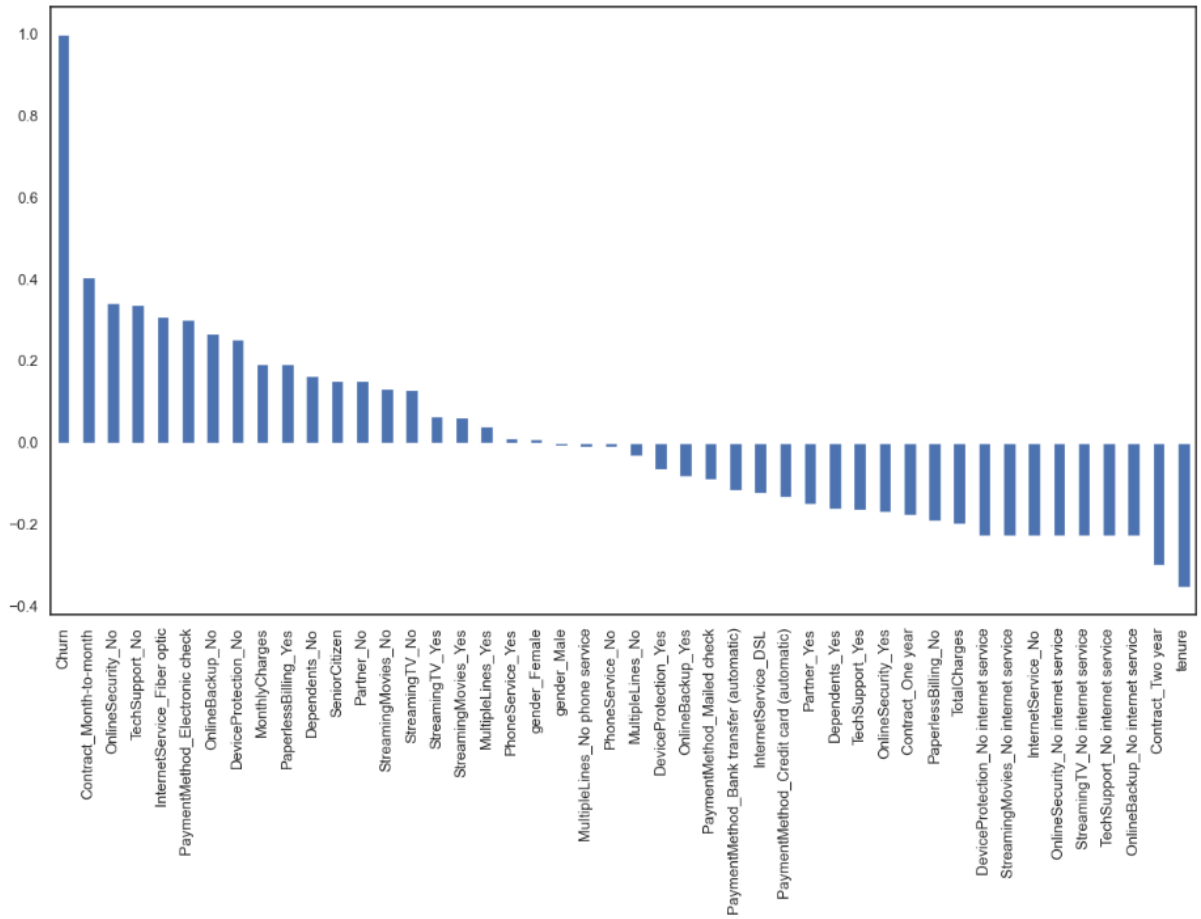


Figure 3.3: Corrélation de Churn

'Month to month contracts', 'absence of online 'security' et 'tech support' semblent être positivement corrélés avec le taux de désabonnement. Tandis que 'tenure', 'two year contracts' semblent être corrélés négativement avec le taux de désabonnement.

Fait intéressant, des services tels que 'Online security', 'streaming TV', 'online backup',' tech support', etc. sans connexion Internet semblent être négativement liés au désabonnement.

3.7.1 Exploration de Données:

Nous Commençons par explorer notre ensemble de données, afin de mieux comprendre les modèles dans les données et éventuellement de formuler des hypothèses.

i. **Répartition par sexe :**

```
colors = ['#4D3425','#E4512B']
ax = (telecom_cust['gender'].value_counts()*100.0
/len(telecom_cust)).plot(kind='bar', stacked = True,
rot = 0, color = colors)
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers')
ax.set_xlabel('Gender')
ax.set_ylabel('% Customers')
ax.set_title('Gender Distribution')

# create a list to collect the plt.patches data
totals = []

# find the values and append to list
for i in ax.patches:
    totals.append(i.get_width())

# set individual bar lables using above list
total = sum(totals)

for i in ax.patches:
    # get_width pulls left or right; get_y pushes up or down
    ax.text(i.get_x()+.15, i.get_height()-3.5, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight = 'bold')
```

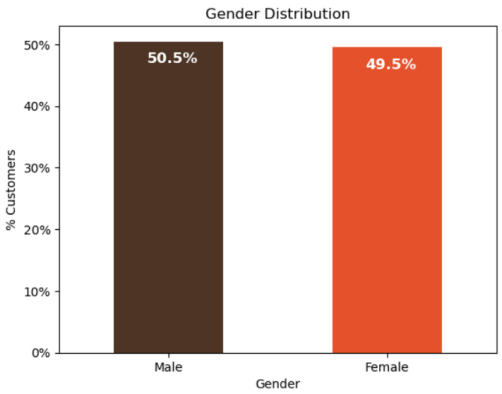


Figure 3.4: Gender Distribution

Environ la moitié des clients de notre ensemble de données sont des hommes tandis que l'autre moitié sont des femmes

ii. **Personnes agées :**

```
ax = (telecom_cust['SeniorCitizen'].value_counts()*100.0
/len(telecom_cust))\
.plot.pie(autopct='%.1f%%',
labels = ['No', 'Yes'],figsize =(5,5), fontsize = 12 )
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('Senior Citizens',fontsize = 12)
ax.set_title('% of Senior Citizens', fontsize = 12)
```

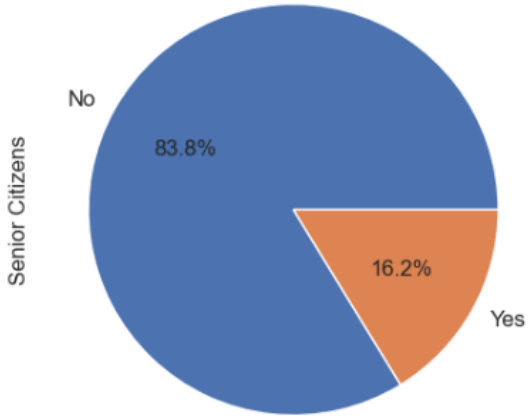
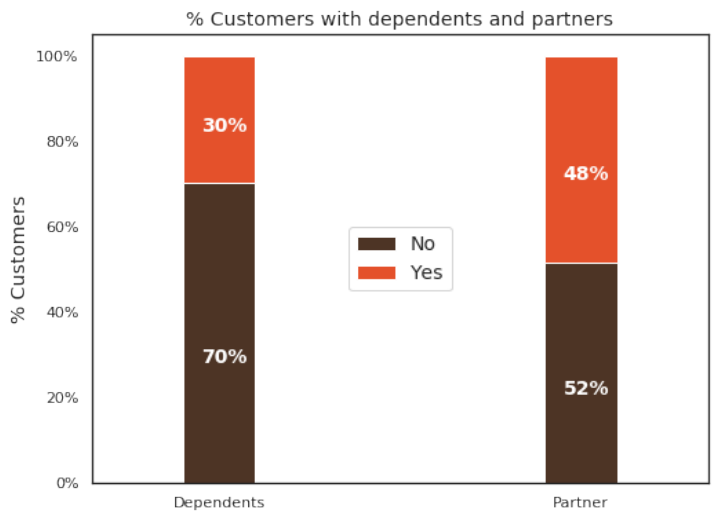


Figure 3.5: Personnes agées

Seuls 16 /° des clients sont des seniors. Ainsi, la plupart de nos clients dans les données sont des jeunes.

iii. **Statut de partenaire et de personne à charge :**

```
df2 = pd.melt(telecom_cust, id_vars=['customerID'],
value_vars=['Dependents','Partner'])
df3 = df2.groupby(['variable','value']).count().unstack()
df3 = df3*100/len(telecom_cust)
colors = ['#4D3425','#E4512B']
ax = df3.loc[:,'customerID']
.plot.bar(stacked=True, color=colors,figsize=(8,6),rot = 0,
                                        width = 0.2)

ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 14)
ax.set_xlabel('')
ax.set_title('% Customers with dependents and partners'
,size = 14)
ax.legend(loc = 'center',prop={'size':14})
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.annotate('{:.0f}%'.format(height)
    ,(p.get_x()+.25*width, p.get_y()+.4*height),
     color = 'white', weight = 'bold',  size = 14)
```



Environ 50 /° des clients ont un partenaire, tandis que seulement 30 /° du total des clients ont des personnes à charge.

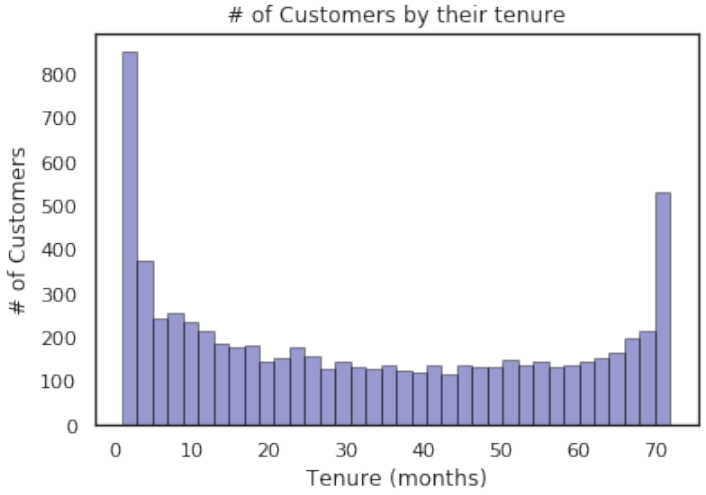
Fait intéressant, parmi les clients qui ont un partenaire, seulement environ la moitié d'entre eux ont également une personne à charge, tandis que l'autre moitié n'a pas d'indépendants. De plus, comme prévu, parmi les clients qui n'ont pas de partenaire, une majorité (80/°) d'entre eux n'ont pas de personnes à charge .

```
colors = ['#4D3425','#E4512B']
partner_dependents = telecom_cust.groupby(['Partner',
'Dependents']).size().unstack()
ax = (partner_dependents.T*100.0 /
partner_dependents.T.sum()).T.plot(kind='bar',
width = 0.2,stacked = True,rot = 0, figsize = (8,6),
 color = colors)
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.legend(loc='center',
prop={'size':14},title = 'Dependents',fontsize =14)
ax.set_ylabel('% Customers',size = 14)
ax.set_title('% Customers with/without dependents
based on whether they have a partner',size = 14)
ax.xaxis.label.set_size(14)
# Code to add the data labels on the stacked bar chart
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.annotate('{:.0f}%'.format(height),
    (p.get_x()+.25*width, p.get_y()+.4*height),
 color = 'white', weight = 'bold',    size = 14)
```

3.7.2 Informations sur le compte client :

i. **Tenure :** Après avoir examiné l'histogramme ci-dessous, nous pouvons voir que de nombreux clients sont avec la société de télécommunications depuis un mois seulement, alors qu'un bon nombre sont là depuis environ 72 mois. Cela pourrait être dû au fait que différents clients ont des contrats différents. Ainsi, en fonction du contrat qu'ils ont conclu, il pourrait être plus/moins facile pour les clients de rester/quitter l'entreprise de télécommunications

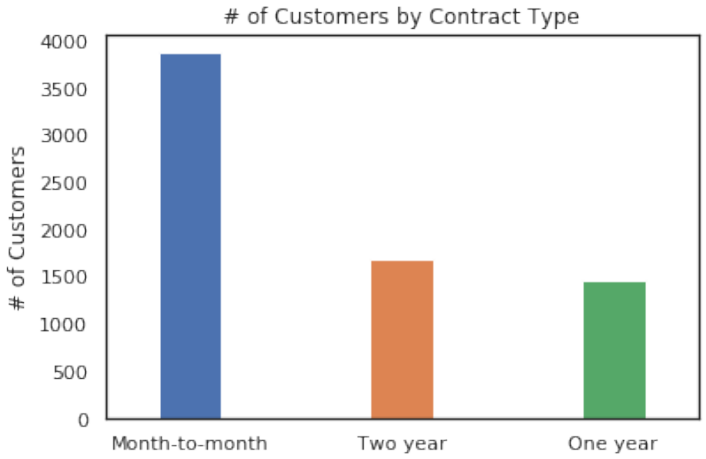
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(style='white')
# Assuming telecom_cust is your DataFrame
# Create a displot figure
g = sns.displot(data=telecom_cust, x='tenure',
bins=int(180/5), color='darkblue', edgecolor='black')
# Set axis labels and title
g.set_axis_labels(y_var='# of Customers',
x_var='Tenure (months)')
g.fig.suptitle('# of Customers by their tenure', y=1.02)
# Show the plot
plt.show()
```



ii. **Contrats :** pour comprendre le graphique ci-dessus,nous examinons alors le nombre de clients par différents contrats.

```
ax = telecom_cust['Contract'].value_counts()
.plot(kind = 'bar',rot = 0, width = 0.3)
ax.set_ylabel('# of Customers')
ax.set_title('# of Customers by Contract Type')
```

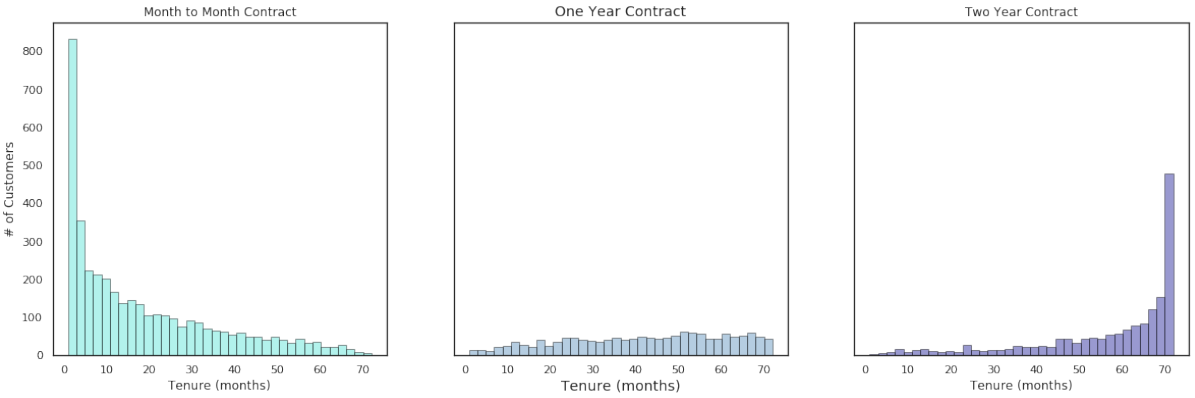
la plupart des clients sont sous contrat mensuel. Alors qu'il y a un nombre égal de clients dans les contrats de 1 an et de 2 ans.



Ci-dessous, nous comprendrons le mandat des clients en fonction de leur type de contrat.

```
fig , (ax1,ax2,ax3) = plt.subplots(nrows=1, ncols=3,
sharey = True, figsize = (20,6))
ax = sns.distplot(telecom_cust[telecom_cust['Contract']
=='Month-to-month']['tenure'],hist=True, kde=False,
bins=int(180/5), color = 'turquoise',
hist_kws={'edgecolor': 'black'},
kde_kws={'linewidth': 4},ax=ax1)
ax.set_ylabel('# of Customers')
ax.set_xlabel('Tenure (months)')
ax.set_title('Month to Month Contract')
ax = sns.distplot(telecom_cust[telecom_cust['Contract']
=='One year']['tenure'],
 hist=True, kde=False, bins=int(180/5), color = 'steelblue',
hist_kws={'edgecolor': 'black'},kde_kws={'linewidth': 4},
ax=ax2)
ax.set_xlabel('Tenure (months)',size = 14)
ax.set_title('One Year Contract',size = 14)
ax = sns.distplot(telecom_cust[telecom_cust['Contract']==
'Two year']['tenure'],hist=True, kde=False,
 bins=int(180/5), color = 'darkblue',
 hist_kws={'edgecolor': 'black'},
 kde_kws={'linewidth': 4}, ax=ax3)
ax.set_xlabel('Tenure (months)')
ax.set_title('Two Year Contract')
```

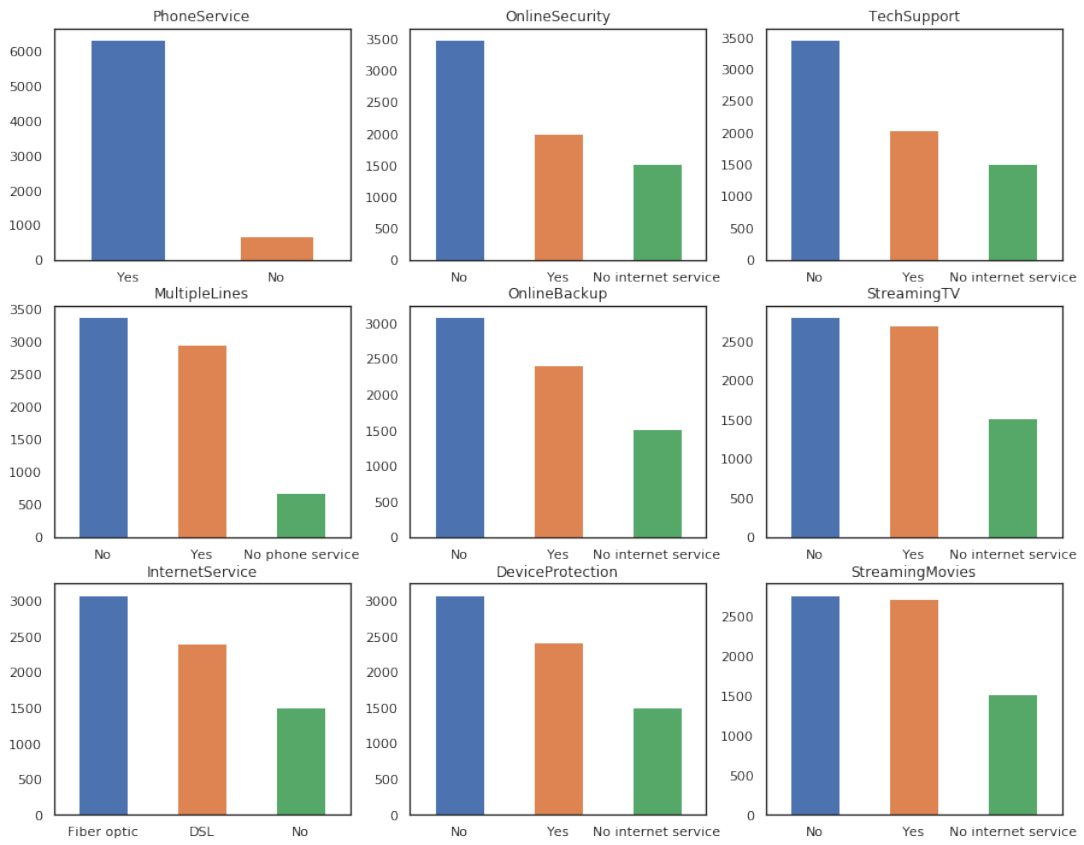
Fait intéressant, la plupart des contrats mensuels durent 1 à 2 mois, tandis que les contrats de 2 ans ont tendance à durer environ 70 mois. Cela montre que les clients qui prennent un contrat plus long sont plus fidèles à l'entreprise et ont tendance à rester avec elle plus longtemps.



Voyons maintenant la répartition des différents services utilisés par les clients...

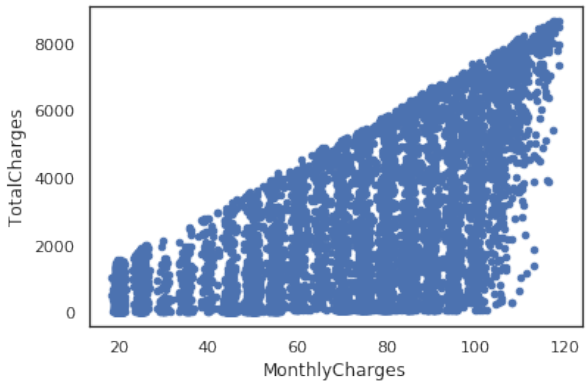
```
telecom_cust.columns.values

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

```
services = ['PhoneService','MultipleLines','InternetService',
'OnlineSecurity', 'OnlineBackup','DeviceProtection',
'TechSupport','StreamingTV','StreamingMovies']
fig, axes = plt.subplots(nrows = 3,ncols
= 3,figsize = (15,12))
for i, item in enumerate(services):

 if i < 3:
   ax = telecom_cust[item].value_counts()
   .plot(kind = 'bar',ax=axes[i,0],rot = 0)
 elif i >=3 and i < 6:
  =  ax = telecom_cust[item].value_counts()
  .plot(kind = 'bar',ax=axes[i-3,1],rot = 0)

elif i < 9:
  ax = telecom_cust[item].value_counts()
  .plot(kind = 'bar',ax=axes[i-6,2],rot = 0)
    ax.set_title(item)
```



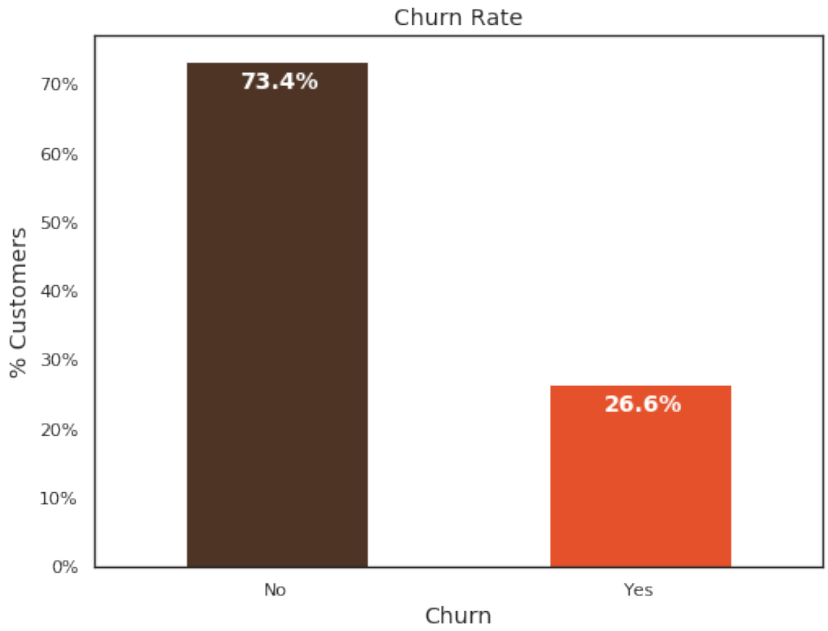
Maintenant, nous examinons rapidement la relation entre les frais mensuels et les charges totale ...

```
telecom_cust[['MonthlyCharges', 'TotalCharges']]
.plot.scatter(x = 'MonthlyCharges', y='TotalCharges')
```



Enfin, examinons notre variable prédictive (Churn) et comprenons son interaction avec d'autres variables importantes, comme cela a été découvert dans le diagramme de corrélation...

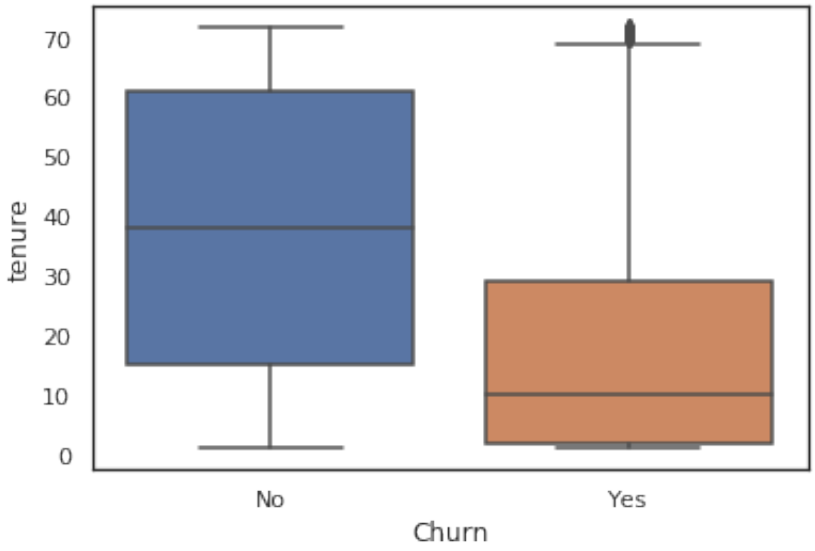
```
colors = ['#4D3425','#E4512B']
ax = (telecom_cust['Churn'].value_counts()*100.0
/len(telecom_cust))
.plot(kind='bar',    stacked = True, rot = 0 color = colors,
 figsize = (8,6))
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 14)
ax.set_xlabel('Churn',size = 14)
ax.set_title('Churn Rate', size = 14)
# create a list to collect the plt.patches data
totals = []
# find the values and append to list
for i in ax.patches:
    totals.append(i.get_width())
# set individual bar lables using above list
total = sum(totals)
for i in ax.patches:
    # get_width pulls left or right; get_y pushes up or down
    ax.text(i.get_x()+.15, i.get_height()-4.0, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight = 'bold',
           size = 14)
```



Dans nos données, 74 /° des clients ne se désabonnent pas. De toute évidence, les données sont faussées car nous nous attendrions à ce qu'une grande majorité des clients ne se désabonnent pas. Il est important de garder cela à l'esprit pour notre modélisation, car l'asymétrie peut entraîner de nombreux faux négatifs.

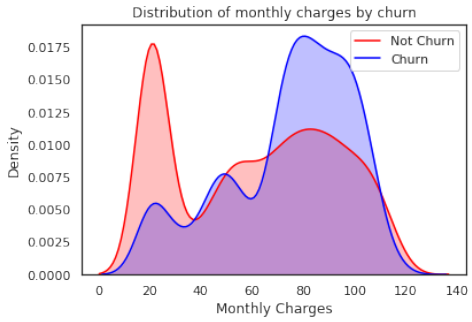
Nous explorons maintenant le taux de désabonnement par 'tenure, seniority, contract type, monthly charges et total charges' pour voir comment il varie en fonction de ces variables

```
sns.boxplot(x = telecom_cust.Churn, y = telecom_cust.tenure)
```



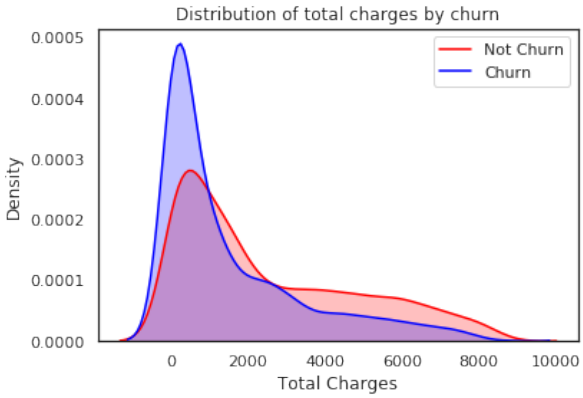
Comme nous pouvons voire dans le graphique ci-dessus, les clients qui ne se désintéressent pas ont tendance à rester plus longtemps avec la société de télécommunications

```
ax = sns.kdeplot(telecom_cust.MonthlyCharges
[(telecom_cust["Churn"] == 'No') ]
,color="Red", shade = True)
ax = sns.kdeplot(telecom_cust.MonthlyCharges
[(telecom_cust["Churn"] == 'Yes') ],
ax =ax, color="Blue", shade= True)
ax.legend(["Not Churn","Churn"],loc='upper right')
ax.set_ylabel('Density')
ax.set_xlabel('Monthly Charges')
ax.set_title('Distribution of monthly charges by churn')
```



Un pourcentage plus élevé de clients abandonnent lorsque les frais mensuels sont élevés.

```
ax = sns.kdeplot(telecom_cust.TotalCharges
[(telecom_cust["Churn"] == 'No') ],
 color="Red", shade = True)
ax = sns.kdeplot(telecom_cust.TotalCharges
[(telecom_cust["Churn"] == 'Yes') ],
 ax =ax, color="Blue", shade= True)
ax.legend(["Not Churn","Churn"],loc='upper right')
ax.set_ylabel('Density')
ax.set_xlabel('Total Charges')
ax.set_title('Distribution of total charges by churn')
```



Il semble que le taux de désabonnement soit plus élevé lorsque les charges totales sont inférieures.

3.8 Évaluations et prédictions du modèle d'apprentissage automatique :

maintenant, nous allons développer la régression logistique, Random Forest, ADA Boost ...

3.8.1 Régression logistique :

```
# We will use the data frame where we had created dummy variables
y = df_dummies['Churn'].values
X = df_dummies.drop(columns = ['Churn'])
# Scaling all the variables to a range of 0 to 1
from sklearn.preprocessing import MinMaxScaler
features = X.columns.values
scaler = MinMaxScaler(feature_range = (0,1))
scaler.fit(X)
X = pd.DataFrame(scaler.transform(X))
X.columns = features
```

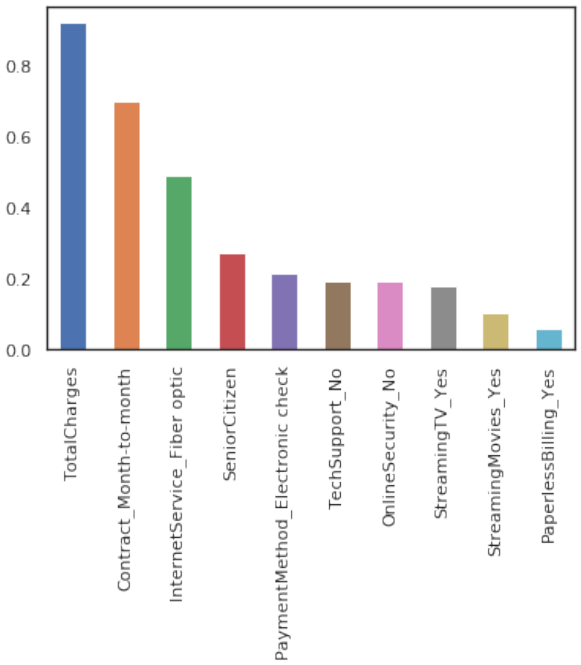
```
# Create Train & Test Data

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test
= train_test_split(X, y, test_size=0.3, random_state=101)
```

```
# Running logistic regression model

from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
result = model.fit(X_train, y_train)
```

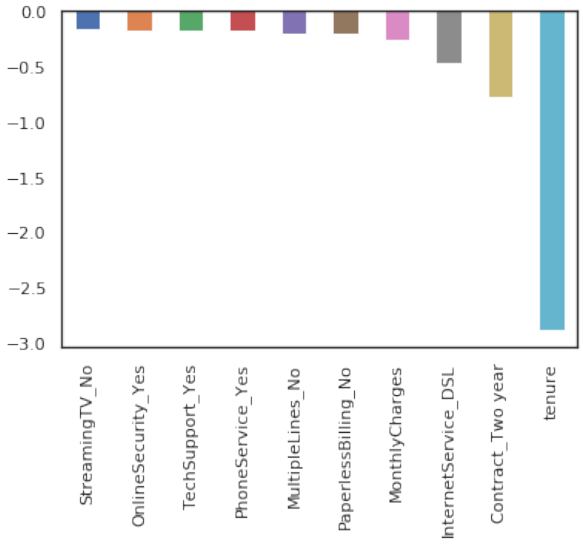
```
from sklearn import metrics
prediction_test = model.predict(X_test)

# Print the prediction accuracy
print (metrics.accuracy_score(y_test, prediction_test))
```

Resultat: 0.8075829383886256

```
# To get the weights of all the variables

weights = pd.Series(model.coef_[0],index=X.columns.values)

print (weights.sort_values(ascending = False)
[:10].plot(kind='bar'))
```



```
print(weights.sort_values(ascending = False)[-10:]
.plot(kind='bar'))
```



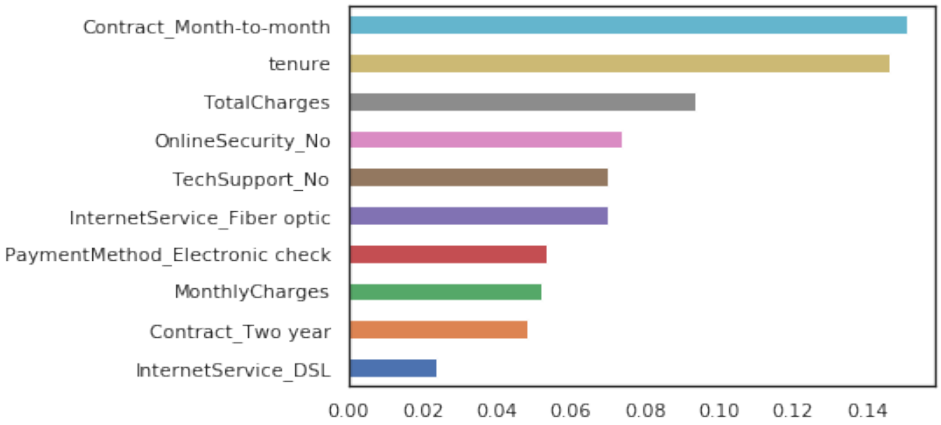
Nous pouvons voir que certaines variables ont une relation négative avec notre variable prédite (Churn), tandis que d'autres ont une relation positive. Une relation négative signifie que la probabilité de désabonnement diminue avec cette variable. Résumons ci-dessous certaines des fonctionnalités intéressantes :
-Comme nous l'avons vu dans notre EDA, avoir un contrat de 2 mois réduit les risques de désabonnement. Le contrat de 2 mois avec l'ancienneté a la relation la plus négative avec le taux de désabonnement, comme prédit par les régressions logistiques
-Avoir un service Internet DSL réduit également le risque de désabonnement
-Enfin, les charges totales, les contrats mensuels, les services Internet par fibre optique et l'ancienneté peuvent entraîner des taux de désabonnement plus élevés. Ceci est intéressant car bien que les services de fibre optique soient plus rapides, les clients sont susceptibles de se désabonner à cause de cela.

3.8.2 Random Forest

```
from sklearn.ensemble import RandomForestClassifier
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=101)
# Update max_features parameter to 'sqrt'
model_rf = RandomForestClassifier
(n_estimators=1000, oob_score=True, n_jobs=-1, random_state=50,
max_features='sqrt',
# Update this linemax_leaf_nodes=30)
model_rf.fit(X_train, y_train)
# Make predictions
prediction_test = model_rf.predict(X_test)
print(metrics.accuracy_score(y_test, prediction_test))
```

Resultat: 0.8088130774697939

```
importances = model_rf.feature_importances_
weights = pd.Series(importances,
                    index=X.columns.values)
weights.sort_values()[-10:].plot(kind = 'barh')
```



D'après l'algorithme de Random Forest, le contrat mensuel, la tenure et les charges totales sont les variables prédictives les plus importantes pour prédire le taux de désabonnement. **Les résultats de Random Forest sont très similaires à ceux de la régression.**

3.8.3 ADA Boost

```
# AdaBoost Algorithm
from sklearn.ensemble import AdaBoostClassifier
model = AdaBoostClassifier()
# n_estimators = 50 (default value)
# base_estimator = DecisionTreeClassifier (default value)
model.fit(X_train,y_train)
preds = model.predict(X_test)
metrics.accuracy_score(y_test, preds)
```

Resultat: 0.8159203980099502

3.8.4 XG Boost

```
from xgboost import XGBClassifier
# Assuming X_train, X_test, y_train, y_test are defined
model = XGBClassifier()
model.fit(X_train, y_train)
preds = model.predict(X_test)
accuracy = metrics.accuracy_score(y_test, preds)
print(accuracy)
```

Resultat: 0.8294243070362474

De toute évidence, XG Boost est le gagnant parmi toutes les autres techniques

3.9 conclusion:

En conclusion de ce chapitre dédié à la prédiction de la désertion des clients dans le secteur des télécommunications, nous avons parcouru un voyage essentiel à travers les différentes étapes du processus.

Nous avons commencé par le chargement des bibliothèques et des données, assurant ainsi une base solide pour nos analyses. Ensuite, nous nous sommes plongés dans la compréhension approfondie des données, ce qui nous a permis de saisir les tenants et aboutissants de notre jeu de données.

La manipulation des données a été une étape cruciale pour préparer nos informations, nous permettant de les adapter aux besoins de notre modèle. Le prétraitement des données a également joué un rôle vital, incluant la gestion des valeurs manquantes, la normalisation et la transformation des caractéristiques.

La visualisation des données nous a offert des perspectives visuelles riches, nous aidant à identifier des tendances et des corrélations potentielles. Enfin, nous avons évalué plusieurs modèles d'apprentissage automatique pour prédire la désertion des clients, en utilisant des métriques appropriées pour sélectionner le modèle le plus performant.

Conclusion génerale

Ce rapport offre un aperçu complet des fondations du Machine Learning ainsi qu'une application concrète dans un domaine spécifique. Il vise à armer le lecteur de connaissances et de compétences essentielles pour naviguer dans le paysage complexe de l'apprentissage automatique, tout en soulignant son impact positif sur des secteurs variés, comme celui des télécommunications, où la prédiction de la désertion des clients peut contribuer à une gestion plus efficace et à une meilleure rétention de la clientèle.