📊 Analyse Prédictive du Churn Client (Secteur Télécom)

- Ce notebook présente une analyse complète pour comprendre les départs clients (churn),
- identifier les signaux prédictifs,
- et construire un modèle prédictif exploitable par les équipes métier.

📦 1. Imports & Setup

In [1]:
```
import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, roc_auc_score, confusion_matrix, RocCurveDisplay

import warnings
warnings.filterwarnings('ignore')

print("📚 Imports terminés. Prêt à plonger dans la data.")
```

```
📚 Imports terminés. Prêt à plonger dans la data.
```

📥 Chargement des données

In [2]:
```
# 📂 Chargement des données
# Dataset classique ( Telco churn)
url = 'https://raw.githubusercontent.com/IBM/telco-customer-churn-on-icp4d/master/data/Telco-Customer-Churn.cs
df = pd.read_csv(url)

print(f"✅ Données chargées. {df.shape[0]} lignes, {df.shape[1]} colonnes")
df.head()
```

```
✅ Données chargées. 7043 lignes, 21 colonnes
```

Out[2]:

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

5 rows × 21 columns

🔍 2. Quelles données avons-nous ?

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   customerID        7043 non-null   object 
 1   gender            7043 non-null   object 
 2   SeniorCitizen     7043 non-null   int64  
 3   Partner           7043 non-null   object 
 4   Dependents        7043 non-null   object 
 5   tenure            7043 non-null   int64  
 6   PhoneService      7043 non-null   object 
 7   MultipleLines     7043 non-null   object 
 8   InternetService   7043 non-null   object 
 9   OnlineSecurity    7043 non-null   object 
 10  OnlineBackup      7043 non-null   object 
 11  DeviceProtection  7043 non-null   object 
 12  TechSupport       7043 non-null   object 
 13  StreamingTV       7043 non-null   object 
 14  StreamingMovies   7043 non-null   object 
 15  Contract          7043 non-null   object 
 16  PaperlessBilling  7043 non-null   object 
 17  PaymentMethod     7043 non-null   object 
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object 
 20  Churn             7043 non-null   object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

🧹 3. Nettoyage & Prétraitement

In [4]:
```
# Supprimer les colonnes inutiles
if 'customerID' in df.columns:
    df.drop('customerID', axis=1, inplace=True)
```

In [5]:
```
# Gérer les valeurs manquantes
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = df[col].fillna(df[col].mode()[0])
    else:
        df[col] = df[col].fillna(df[col].median())

# Conversion des colonnes object en numérique si nécessaire
binary_cols = [col for col in df.columns if df[col].nunique() == 2 and df[col].dtype == 'object']
le = LabelEncoder()
for col in binary_cols:
    df[col] = le.fit_transform(df[col])

# Variables catégorielles à encoder
cat_cols = df.select_dtypes(include='object').columns
df = pd.get_dummies(df, columns=cat_cols, drop_first=True)

print("🤌 Données nettoyées et encodées. Prêtes pour l'analyse.")
```

```
🤌 Données nettoyées et encodées. Prêtes pour l'analyse.
```

📉 4. Analyse des comportements clients

Dans cette section, on explore **3 signaux faibles** détectés via la data pour anticiper le départ client dans un contexte télécom.

Chaque graphique a été réalisé avec `Plotly` pour une lecture visuelle claire et interactive.

📊 Qui part ? Qui reste

In [15]:
```
# 📊 Analyse univariée
fig1 = px.histogram(df, x='Churn', title='Répartition du churn', color='Churn')
fig1.show()
```

Répartition du churn (clients qui quittent vs. ceux qui restent)

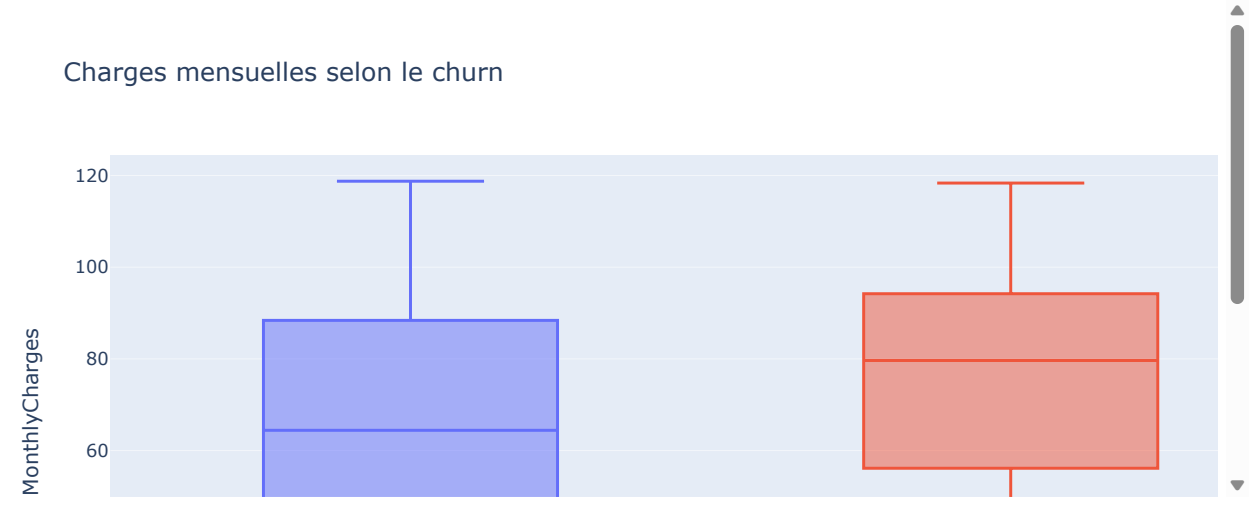
- Environ **1 client sur 4 churn** (soit 25 % des clients).
- Les **75 % restants sont fidèles** à l'opérateur.
- On observe donc une **cible minoritaire mais critique**.

> 🎯 *Objectif : identifier les 25 % avant qu'ils ne partent, pas après.*

🧠 Y a-t-il des signaux dans les données ?

💸 a- Le montant mensuel influence-t-il le départ ?

In [7]:
```
# 🔍 Analyse bivariée
fig2 = px.box(df, x='Churn', y='MonthlyCharges', color='Churn', title='Charges mensuelles selon le churn')
fig2.show()
```



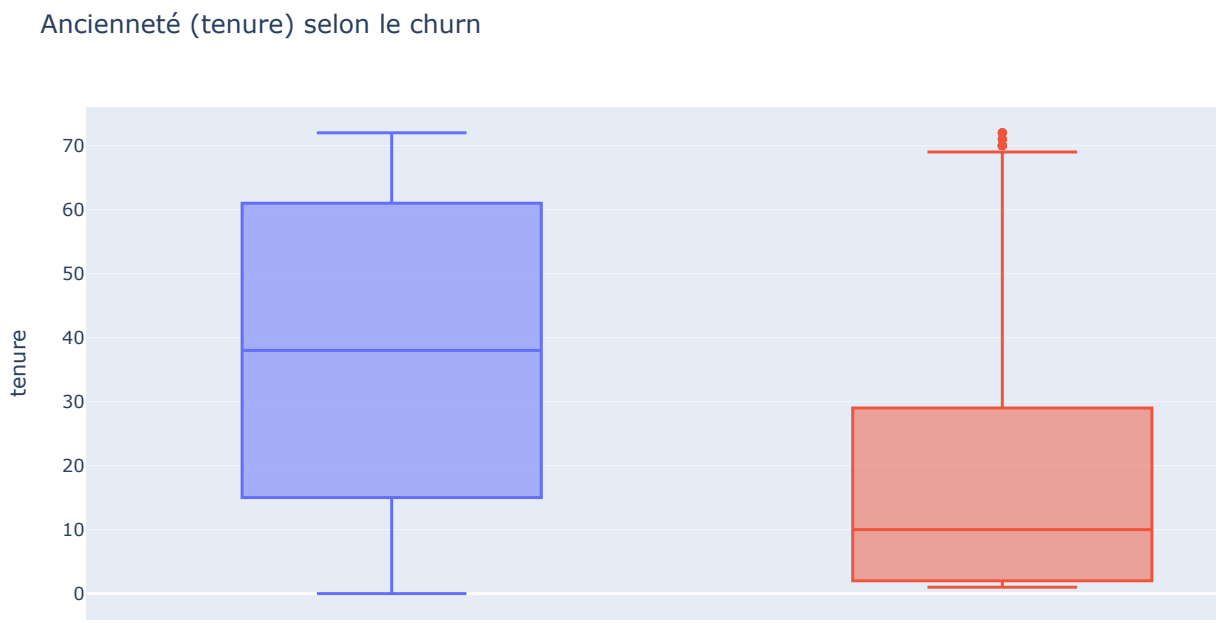
Boxplot des MonthlyCharges selon le churn

- Les clients qui partent **paient plus cher** en moyenne.
- La médiane est plus haute, avec une concentration dans les **charges mensuelles élevées**.
- Les churners se situent souvent entre **60€ et 90€**.

> 🔍 *Facture élevée = client agacé. C'est un signal faible mais clair.*

L'Ancienneté influence-t-elle le départ ?

In [8]:
```
# Tenure vs churn
fig = px.box(df, x='Churn', y='tenure', color='Churn', title="Ancienneté (tenure) selon le churn")
fig.show()
```



🧠 **Interprétation :**

- Les clients qui quittent l'opérateur ont une ancienneté bien plus faible que ceux qui restent.
- La médiane d'ancienneté des churners est largement inférieure à celle des clients fidèles.

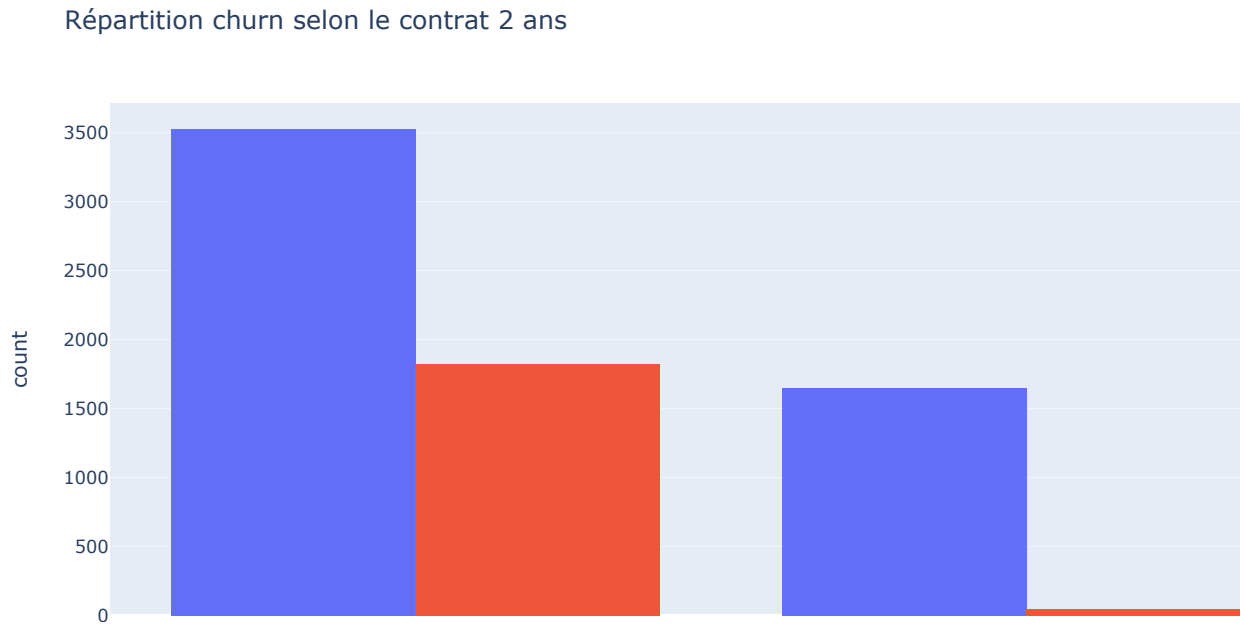
Plus un client est ancien, plus il est attaché. Moins il est enclin à partir.

En resumé :

- La fidélité se construit avec le temps. Les nouveaux clients sont les plus fragiles.

Les types de contrats influencent-t-ils le départ ?

In [9]:
```
# Churn selon type de contrat
fig = px.histogram(df, x='Contract_Two year', color='Churn', barmode='group',
                   title="Répartition churn selon le contrat 2 ans")
fig.show()
```



🧠 **Interprétation :** Les clients sous contrat long terme (2 ans) churnent beaucoup moins.

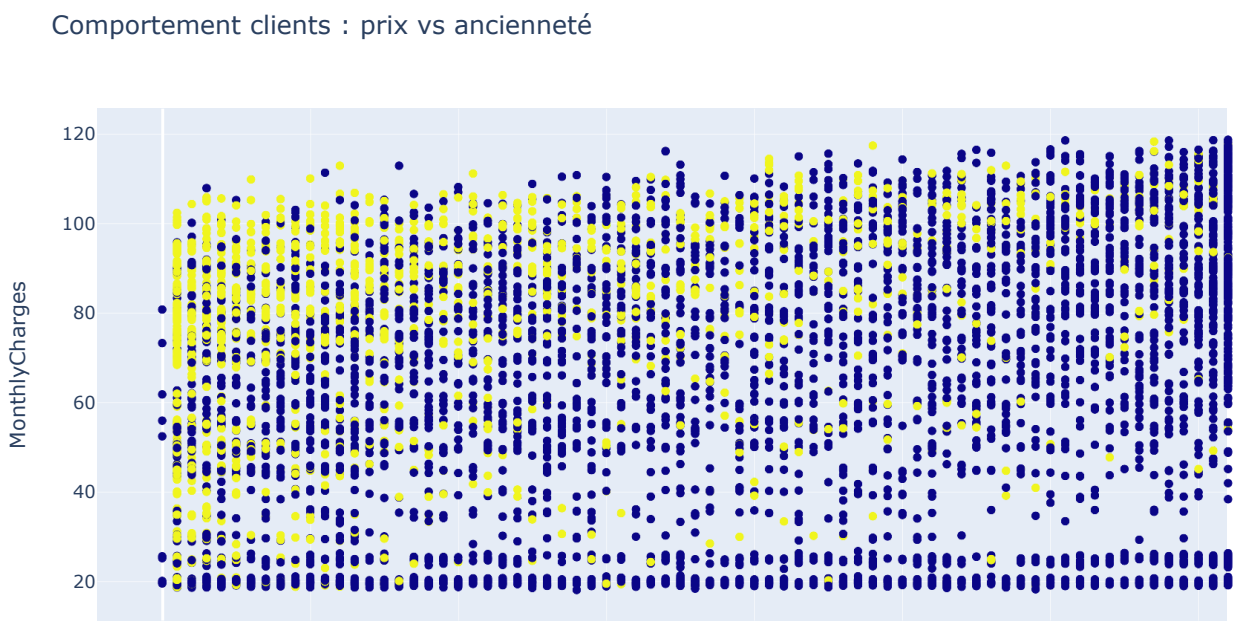
- Le churn est massivement présent chez les non-engagés.
- La fidélisation semble fortement corrélée à l'engagement contractuel.

En gros :

- Plus le contrat est long, plus le client reste. L'engagement structure la fidélité.

Les charges mensuelles & l'ancienneté influencent-t-elles le départ ?

In [10]:
```
# Scatter plot MonthlyCharges vs Tenure (couleur = churn)
fig = px.scatter(df, x='tenure', y='MonthlyCharges', color='Churn',
                 title="Comportement clients : prix vs ancienneté")
fig.show()
```



🧠 **Interprétation :**

- Le churn est particulièrement élevé chez les clients récents et à facturation élevée.
- Les clients anciens avec des prix modérés sont les plus stables.
- On observe une zone de vulnérabilité claire : prix élevé + faible ancienneté.

En clair :

- Un client qui paie cher sans avoir d'ancienneté est un client à risque.

✅ **Ce qu'on en retient**

- L'ancienneté est un facteur de stabilisation.
- Le type de contrat joue un rôle clé dans la rétention.
- Le prix devient un facteur de départ lorsqu'il est mal aligné au profil client.

🎯 Ces visualisations permettent d'identifier des profils à risque concrets. Elles offrent des leviers pour adapter les offres, les services, ou les stratégies de communication.

⚖️ 5. Split & Préparation des données

In [11]:
```
# ⚖️ Split & Préparation des données
X = df.drop('Churn', axis=1)
y = df['Churn']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y)

print("✂️ Données séparées en train/test et normalisées.")
```

```
✂️ Données séparées en train/test et normalisées.
```

🤖 6. Modélisation

In [12]:
```
# 🤖 Modélisation
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:,1]

print("🧠 Modèle entraîné. Voici le rapport de classification :")
print(classification_report(y_test, y_pred))

roc_auc = roc_auc_score(y_test, y_proba)
print(f"ROC AUC Score: {roc_auc:.2f}")
```

```
🧠 Modèle entraîné. Voici le rapport de classification :
              precision    recall  f1-score   support

           0       0.83      0.91      0.87      1035
           1       0.66      0.48      0.56       374

    accuracy                           0.80      1409
   macro avg       0.75      0.70      0.71      1409
weighted avg       0.78      0.80      0.79      1409

ROC AUC Score: 0.83
```

📈 Performance du modèle

Classification Report & AUC ROC

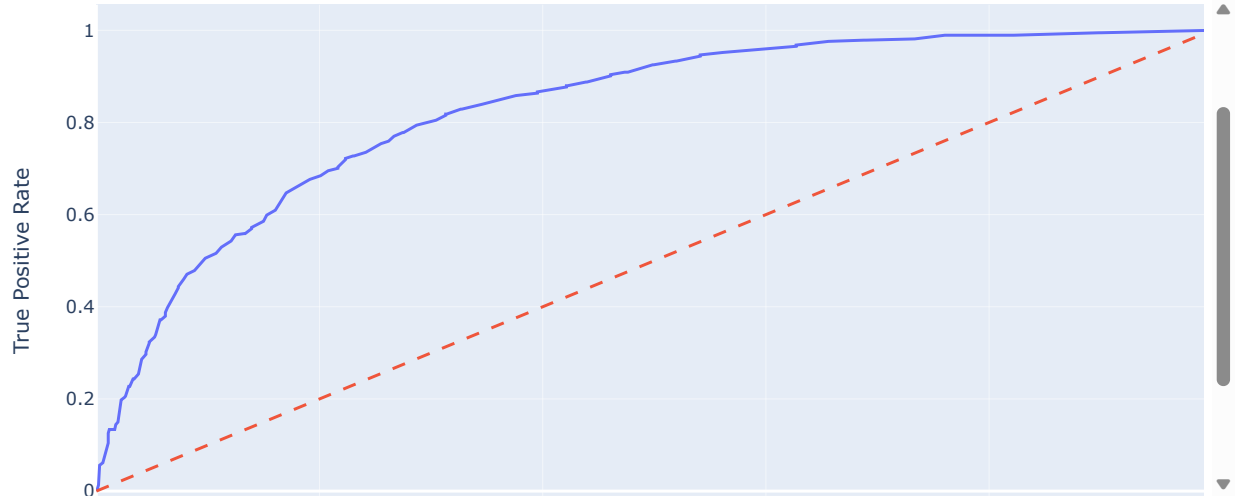
- **Accuracy globale : 80 %** → bon point de départ.
- **Précision classe 1 (churn) : 66 %**
- **Recall classe 1 (churn) : 48 %**
- **ROC AUC : 0.83** → bonne capacité de prédiction.

> 🧠 *Le modèle est efficace pour détecter les clients fidèles. Il a encore du mal à attraper tous les churners, mais il est déjà bien meilleur que le hasard.*

7.Courbe ROC

In [13]:
```
# 🎯 Courbe ROC
fpr = []
tpr = []
from sklearn.metrics import roc_curve
fpr, tpr, _ = roc_curve(y_test, y_proba)

fig3 = go.Figure()
fig3.add_trace(go.Scatter(x=fpr, y=tpr, mode='lines', name='ROC Curve'))
fig3.add_trace(go.Scatter(x=[0,1], y=[0,1], mode='lines', name='Random', line=dict(dash='dash')))
fig3.update_layout(title='Courbe ROC', xaxis_title='False Positive Rate', yaxis_title='True Positive Rate')
fig3.show()
```



Évaluation du modèle

- La courbe est bien au-dessus de la ligne aléatoire.

- **AUC de 0.83 = bonne performance** pour un modèle de base.
- Bonne séparation entre les classes 0 et 1.

🔍 Quelles variables influencent le churn ?

In [14]:
```
# 🔎 9. Variables importantes
importances = pd.Series(model.feature_importances_, index=X.columns)
importances_sorted = importances.sort_values(ascending=False).head(10)

fig4 = px.bar(importances_sorted[::-1], orientation='h', title='Top 10 des variables les plus importantes', la
fig4.show()
```

Features les plus influentes dans la prédiction

- `tenure` → plus le client est ancien, **moins il part**.
- `MonthlyCharges` → plus il paie, **plus il est à risque**.
- `PaymentMethod_Electronic check` → souvent lié à un churn élevé.
- `Contract_Two year`, `Contract_One year` → les contrats longs **fidélisent**.
- `OnlineSecurity_Yes`, `TechSupport_Yes` → **les services activés rassurent**.
- `InternetService_Fiber optic` → peut être un facteur de churn si instable.
- `PaperlessBilling` → comportement digital à surveiller.
- `gender` → peu d'impact, mais présent dans le top 10.

> 🛠️ *Ce sont ces leviers que tu peux **activer ou surveiller** pour anticiper le départ d'un client.*

✅ Ce qu'on retient

- ✅ On **peut détecter** des clients à risque de départ.
- ✅ Des variables comme le **type de contrat, les services, les charges mensuelles** sont déterminantes.
- ✅ Même un modèle simple peut **produire des signaux d'alerte utiles** pour les actions marketing.

> 🚀 *La data science, ce n'est pas prédire l'avenir. C'est **agir maintenant** avec ce qu'on sait déjà.*