Milestone 4 - Westyn Hilliard and Matthew Garlock

November 7, 2024

[2]:

# 

***Predictive Analytics***

***DSC630***

***WESTYN HILLIARD AND MATTHEW GARLOCK***

cd /Users/westynhilliard/Desktop/S C H O O L/Masters/DSC630 - Predicitve␣

𝗌Analytics/Group Project

[3]:

/Users/westynhilliard/Desktop/S C H O O L/Masters/DSC630 - Predicitve Analytics/GROUP Project

*# Imports -*

**import pandas as pd import warnings**

**import matplotlib.pyplot as plt import seaborn as sns**

**import numpy as np**

**from sklearn.model\_selection import** train\_test\_split

**from sklearn.preprocessing import** LabelEncoder

**from sklearn.linear\_model import** LogisticRegression

**from sklearn.ensemble import** RandomForestClassifier, GradientBoostingClassifier

**from sklearn.metrics import** classification\_report, roc\_auc\_score,␣

𝗌confusion\_matrix

[4]:

# Data Loading

[4]: customerID gender SeniorCitizen Partner Dependents tenure PhoneService \

*# Load the dataset*

file\_path = 'Telco-Customer-Churn.csv' data = pd.read\_csv(file\_path)

*# Display the first few rows of the dataset to understand its structure*

data.head()

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]

The dataset consists of customer information from a telecommunications company, containing 21 columns with various features. Here’s a breakdown of some key columns:

## Customer Demographics:

**customerID:** Unique identifier for each customer.

**gender:** Indicates the gender of the customer (e.g., Male, Female).

**SeniorCitizen:** Binary indicator if the customer is a senior citizen (0 = No, 1 = Yes).

**Partner and Dependents:** Indicate if the customer has a partner or dependents (Yes/No).

## Account Information:

**tenure:** Number of months the customer has been with the company. **Contract:** Type of contract the customer has (e.g., Month-to-month, One year). **PaperlessBilling:** Indicates if the customer uses paperless billing (Yes/No).

**PaymentMethod:** Method of payment used by the customer (e.g., Electronic check, Mailed check).

[5]:

## Services and Usage:

**PhoneService:** Indicates if the customer has phone service (Yes/No).

**MultipleLines:** Indicates if the customer has multiple phone lines.

**InternetService:** Type of internet service (e.g., DSL, Fiber optic, No).

**OnlineSecurity, DeviceProtection, TechSupport, StreamingTV, StreamingMovies:** In- dicate if the customer subscribes to these additional services (Yes/No).

## Billing Information:

**MonthlyCharges:** The monthly charge for the customer.

**TotalCharges:** Total charges incurred by the customer.

## Target Variable:

**Churn:** Indicates if the customer has churned (Yes/No).

This dataset is well-suited for analyzing factors that contribute to customer churn, allowing for predictive modeling and customer retention strategies.

*# Checking for missing values in the dataset*

missing\_values = data.isnull().sum()

*# Summary statistics of the dataset*

summary\_stats = data.describe(include='all')

missing\_values, summary\_stats

[5]: (customerID 0

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  dtype: int64,  customerID | | 0  gender | SeniorCitizen | | Partner | Dependents | tenure \ | |
| count 7043 | | 7043 | 7043.000000 | | 7043 | 7043 | 7043.000000 | |
| unique 7043 | | 2 | NaN | | 2 | 2 | NaN | |
| top 7590-VHVEG | | Male | NaN | | No | No | NaN | |
| freq 1 | | 3555 | NaN | | 3641 | 4933 | NaN | |
| mean NaN | | NaN | 0.162147 | | NaN | NaN | 32.371149 | |
| std NaN | | NaN | 0.368612 | | NaN | NaN | 24.559481 | |
| min NaN | | NaN | 0.000000 | | NaN | NaN | 0.000000 | |
| 25% NaN | | NaN | 0.000000 | | NaN | NaN | 9.000000 | |
| 50% NaN | | NaN | 0.000000 | | NaN | NaN | 29.000000 | |
| 75% NaN | | NaN | 0.000000 | | NaN | NaN | 55.000000 | |
| max NaN | | NaN | 1.000000 | | NaN | NaN | 72.000000 | |
|  | PhoneService | MultipleLines | | InternetService | | OnlineSecurity | | … \ |
| count | 7043 | 7043 | | 7043 | | 7043 | | … |
| unique | 2 | 3 | | 3 | | 3 | | … |
| top | Yes | No | | Fiber optic | | No | | … |
| freq | 6361 | 3390 | | 3096 | | 3498 | | … |
| mean | NaN | NaN | | NaN | | NaN | | … |
| std | NaN | NaN | | NaN | | NaN | | … |
| min | NaN | NaN | | NaN | | NaN | | … |
| 25% | NaN | NaN | | NaN | | NaN | | … |
| 50% | NaN | NaN | | NaN | | NaN | | … |
| 75% | NaN | NaN | | NaN | | NaN | | … |
| max | NaN | NaN | | NaN | | NaN | | … |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | DeviceProtection | TechSupport | StreamingTV | | StreamingMovies \ | | |
| count | 7043 | 7043 | 7043 | | 7043 | | |
| unique | 3 | 3 | 3 | | 3 | | |
| top | No | No | No | | No | | |
| freq | 3095 | 3473 | 2810 | | 2785 | | |
| mean | NaN | NaN | NaN | | NaN | | |
| std | NaN | NaN | NaN | | NaN | | |
| min | NaN | NaN | NaN | | NaN | | |
| 25% | NaN | NaN | NaN | | NaN | | |
| 50% | NaN | NaN | NaN | | NaN | | |
| 75% | NaN | NaN | NaN | | NaN | | |
| max | NaN | NaN | NaN | | NaN | | |
|  | Contract | PaperlessBilling | | PaymentMethod | | MonthlyCharges | \ |
| count | 7043 | 7043 | | 7043 | | 7043.000000 |  |
| unique top  freq | 3  Month-to-month  3875 | 2  Yes 4171 | | 4  Electronic check  2365 | | NaN NaN  NaN |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| mean | NaN | | NaN | NaN | 64.761692 |
| std | NaN | | NaN | NaN | 30.090047 |
| min | NaN | | NaN | NaN | 18.250000 |
| 25% | NaN | | NaN | NaN | 35.500000 |
| 50% | NaN | | NaN | NaN | 70.350000 |
| 75% | NaN | | NaN | NaN | 89.850000 |
| max | NaN | | NaN | NaN | 118.750000 |
|  | TotalCharges | Churn | | | |
| count | 7043 | 7043 | | | |
| unique top  freq | 6531  11 | 2  No 5174 | | | |
| mean | NaN | NaN | | | |
| std | NaN | NaN | | | |
| min | NaN | NaN | | | |
| 25% | NaN | NaN | | | |
| 50% | NaN | NaN | | | |
| 75% | NaN | NaN | | | |
| max | NaN | NaN | | | |

[11 rows x 21 columns])

## Missing Values:

There are no missing values in any column of the dataset.

## Categorical Columns:

Many columns are categorical, such as gender, Partner, Dependents, PhoneService, InternetService, and Churn.

* + 1. **Key points for some categorical columns: Gender:** 2 unique values (Male, Female)

**Partner and Dependents:** Majority have ‘No’ values. **InternetService:** 3 unique types (DSL, Fiber optic, No) **Contract:** Most customers are on a month-to-month contract.

**Churn:** Indicates a class imbalance with more “No” (5,174) than “Yes”.

## Numerical Columns: tenure:

* + - * Range: 0 to 72 months
      * Average tenure: ~32.4 months

[6]:

*# Attempting to convert 'TotalCharges' to numeric, coercing errors to NaN*

data['TotalCharges'] = pd.to\_numeric(data['TotalCharges'], errors='coerce')

*# Checking if there are any NaN values introduced in 'TotalCharges'*

total\_charges\_missing = data['TotalCharges'].isna().sum()

print(f"Number of missing values in 'TotalCharges': **{**total\_charges\_missing**}**")

## MonthlyCharges:

* Range: 18.25 dollars to 118.75 dollars
* Average monthly charge: ~$64.76

## TotalCharges:

* Converted to numeric, initially had some non-numeric values.
* 11 entries have NaN values.

## Key Observations:

* Class Imbalance: There is a significant imbalance in the Churn column, with a majority of customers not churning.
* Customer Contracts: Most customers are on month-to-month contracts.
* Billing: Most customers use electronic checks as their payment method.

# Data Cleaning

Number of missing values in 'TotalCharges': 11

* + The TotalCharges column was converted to a numeric type.
  + There are 11 entries in TotalCharges that were non-numeric and have been converted to NaN. This requires further handling to either impute these values or remove the affected rows.

[7]:

*# Dropping rows with missing values in 'TotalCharges'*

data\_cleaned = data.dropna(subset=['TotalCharges'])

*# Checking the shape of the dataset after dropping rows*

data\_cleaned.shape

[7]: (7032, 21)

* + Dropped the 11 rows with missing values in the TotalCharges column.

Result: The dataset now has fewer rows, removing entries with non-numeric TotalCharges values to ensure the integrity of the numerical analysis.

[8]:

*# Encode categorical variables* data\_encoded = data\_cleaned.copy() label\_encoders = {}

*# Encoding all categorical columns*

categorical\_columns = data\_encoded.select\_dtypes(include=['object']).columns

**for** col **in** categorical\_columns:

**if** col != 'customerID': label\_encoders[col] = LabelEncoder()

data\_encoded[col] = label\_encoders[col].fit\_transform(data\_encoded[col])

*# Splitting the data into features and target*

X = data\_encoded.drop(columns=['Churn', 'customerID']) y = data\_encoded['Churn']

*# Split the dataset into training and test sets*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2,␣

𝗌random\_state=42)

# Data Processing

[9]:

*# Initialize models*

logistic\_model = LogisticRegression(max\_iter=1000)

random\_forest\_model = RandomForestClassifier(n\_estimators=100, random\_state=42) gbm\_model = GradientBoostingClassifier(n\_estimators=100, learning\_rate=0.1,␣

𝗌random\_state=42)

*# Train Logistic Regression* logistic\_model.fit(X\_train, y\_train) logistic\_pred = logistic\_model.predict(X\_test)

logistic\_prob = logistic\_model.predict\_proba(X\_test)[:, 1]

*# Train Random Forest* random\_forest\_model.fit(X\_train, y\_train) rf\_pred = random\_forest\_model.predict(X\_test)

rf\_prob = random\_forest\_model.predict\_proba(X\_test)[:, 1]

*# Train Gradient Boosting Machine*

## Encoding:

Categorical variables are encoded using LabelEncoder to convert them into a numerical format.

## Splitting:

The dataset is split into training and test sets (80% training, 20% testing).

# Model Training and Evaluation

gbm\_model.fit(X\_train, y\_train) gbm\_pred = gbm\_model.predict(X\_test)

gbm\_prob = gbm\_model.predict\_proba(X\_test)[:, 1]

*# Evaluation Function*

**def** evaluate\_model(y\_true, y\_pred, y\_prob, model\_name): print(f"Model: **{**model\_name**}**") print(classification\_report(y\_true, y\_pred)) print(f"AUC-ROC: **{**roc\_auc\_score(y\_true, y\_prob)**}**") print("Confusion Matrix:") print(confusion\_matrix(y\_true, y\_pred))

print("**\n**")

*# Evaluate Logistic Regression*

evaluate\_model(y\_test, logistic\_pred, logistic\_prob, "Logistic Regression")

*# Evaluate Random Forest*

evaluate\_model(y\_test, rf\_pred, rf\_prob, "Random Forest")

*# Evaluate Gradient Boosting Machine*

evaluate\_model(y\_test, gbm\_pred, gbm\_prob, "Gradient Boosting Machine")

Model: Logistic Regression

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | f1-score | support |
| 0 | 0.84 | 0.88 | 0.86 | 1033 |
| 1 | 0.62 | 0.52 | 0.57 | 374 |
| accuracy |  |  | 0.79 | 1407 |
| macro avg | 0.73 | 0.70 | 0.71 | 1407 |
| weighted avg | 0.78 | 0.79 | 0.78 | 1407 |

AUC-ROC: 0.8295939866750184

Confusion Matrix: [[912 121]

[178 196]]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model: Random | Forest precision | recall | f1-score | support |
| 0 | 0.83 | 0.90 | 0.86 | 1033 |
| 1 | 0.63 | 0.48 | 0.54 | 374 |
| accuracy |  |  | 0.79 | 1407 |
| macro avg | 0.73 | 0.69 | 0.70 | 1407 |
| weighted avg | 0.77 | 0.79 | 0.78 | 1407 |

AUC-ROC: 0.8123993767180375

Confusion Matrix: [[928 105]

[195 179]]

|  |  |  |  |
| --- | --- | --- | --- |
| Model: Gradient Boosting  precision | Machine recall | f1-score | support |
| 0 0.83 | 0.91 | 0.87 | 1033 |
| 1 0.65 | 0.49 | 0.56 | 374 |
| accuracy |  | 0.80 | 1407 |
| macro avg 0.74 | 0.70 | 0.71 | 1407 |
| weighted avg 0.78 | 0.80 | 0.79 | 1407 |

AUC-ROC: 0.835711623380321

Confusion Matrix:

[[935 98]

[190 184]]

## Model Training:

Trained Logistic Regression, Random Forest, and GBM models on the training data.

## Model Evaluation:

*Evaluated each model using:*

* Classification Report: Provides precision, recall, and F1-score for each class.
* AUC-ROC: Measures the model’s ability to distinguish between churn and non-churn cus- tomers.
* Confusion Matrix: Shows the true positives, true negatives, false positives, and false negatives.

## 1. Logistic Regression:

**Precision:**

* + - * For class 0 (No Churn): 84% of predicted non-churn customers were correct.
      * For class 1 (Churn): 62% of predicted churn customers were correct.

## Recall:

* + - * For class 0: 88% of actual non-churn customers were correctly identified.
      * For class 1: 52% of actual churn customers were correctly identified.

## F1-Score: A balance between precision and recall:

* + - * For class 0: 0.86
      * For class 1: 0.57

## AUC-ROC:

0.83, indicating good discrimination between churn and non-churn customers.

## Confusion Matrix:

Shows that it correctly classified a majority of non-churn customers but struggled with some false negatives (178 actual churns predicted as non-churn).

## 2. Random Forest:

**Precision:**

* + - * For class 0: 83%
      * For class 1: 63%

## Recall:

* + - * For class 0: 90% (better than Logistic Regression)
      * For class 1: 48% (lower than Logistic Regression)
      * F1-Score: Similar to Logistic Regression with a slightly lower recall for the churn class.

## AUC-ROC:

0.81, slightly lower than Logistic Regression, indicating slightly less effective discrimination between classes.

## Confusion Matrix:

Correctly identified a majority of non-churn customers but had some false negatives (195).

## 3. Gradient Boosting Machine (GBM):

**Precision:**

* + - * For class 0: 83%
      * For class 1: 65% (higher than both Logistic Regression and Random Forest)

## Recall:

* + - * For class 0: 91% (highest among all models)
      * For class 1: 49% (similar to Random Forest, slightly better)

## F1-Score:

* + - * For class 0: 0.87 (highest among the models)
      * For class 1: 0.56

## AUC-ROC:

0.84, the highest among the three models, indicating the best discrimination capability.

## Confusion Matrix:

[10]:

*# Set up the style for the plots*

sns.set(style='whitegrid')

*# Churn Distribution*

plt.figure(figsize=(8, 6)) sns.countplot(data=data\_cleaned, x='Churn') plt.title('Distribution of Churn') plt.xlabel('Churn')

plt.ylabel('Count') plt.show()

Better performance overall with the highest correctly predicted churners (184).

## Summary of Model Performance:

**Logistic Regression:**

Performed well with a high AUC-ROC of 0.83, indicating good overall discrimination. However, it struggled with recall for the churn class, indicating it missed a significant number of actual churners.

## Random Forest:

Offered a similar performance to Logistic Regression with a slight drop in AUC-ROC to 0.81. It had a better recall for non-churners but a lower recall for churners, resulting in more false negatives.

## Gradient Boosting Machine (GBM):

Outperformed the other models with the highest AUC-ROC of 0.84. It had the best balance between precision and recall, especially for the churn class, making it the most effective model for this problem.

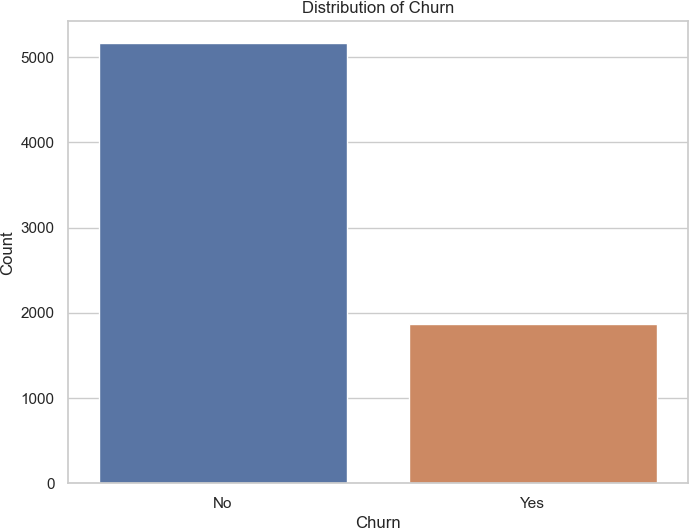
## Conclusion:

**Best Model:**

The Gradient Boosting Machine (GBM) has the highest AUC-ROC and a better balance in per- formance metrics, making it the best model among the three for predicting customer churn.

**Recommendations:** GBM should be used for predicting churn as it captures complex relation- ships in the data better than the others. Further feature analysis can be done to understand what influences churn the most, using feature importance from the Random Forest and GBM models.

# Visualizations



[11]:

## Interpretation of Churn Distribution:

* + **Class Imbalance:** The plot shows a significant imbalance between customers who did not churn (majority) and those who did churn (minority). This imbalance is important to consider when building predictive models, as it can affect the model’s ability to accurately predict the minority class (churn).
  + **Implications for Modeling:** Since most customers did not churn, models trained on this data might be biased towards predicting the majority class (no churn). Special techniques, such as resampling or adjusting class weights, might be needed to ensure the model adequately captures patterns leading to churn.

*# Suppress specific warnings*

warnings.filterwarnings("ignore", category=**FutureWarning**)

*# Plotting the distribution of tenure again without showing the warning*

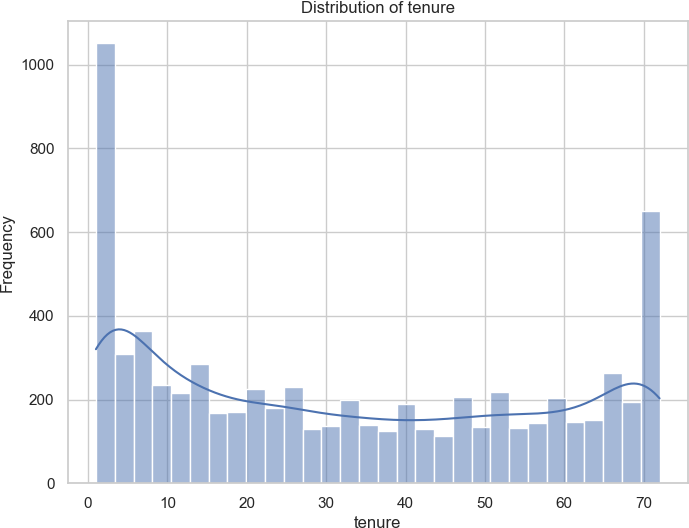
plt.figure(figsize=(8, 6)) sns.histplot(data\_cleaned['tenure'], kde=**True**, bins=30) plt.title('Distribution of tenure') plt.xlabel('tenure')

plt.ylabel('Frequency')

plt.show()

*# Resetting the warning filter back to default*

warnings.filterwarnings("default", category=**FutureWarning**)



## Interpretation of Tenure Distribution:

* + - **Short Tenure:** There is a high frequency of customers with a very short tenure (close to 0 months), indicating that many customers leave shortly after joining.
    - **Long Tenure:** There’s also a spike at the maximum tenure, suggesting a group of loyal customers who have been with the company for an extended period (around 70 months).
    - **Overall Spread:** The distribution has a somewhat bimodal shape with a higher concen- tration of customers at both very short and very long tenures, with fewer customers in the middle range.

## Insights for Churn:

* + - Customers with shorter tenure might be more likely to churn, potentially due to dissatisfaction early in their relationship with the company.
    - Understanding why customers with short tenure churn can help the company implement

strategies to improve customer retention early on.

[12]:

*# Categorical features to visualize with Churn*

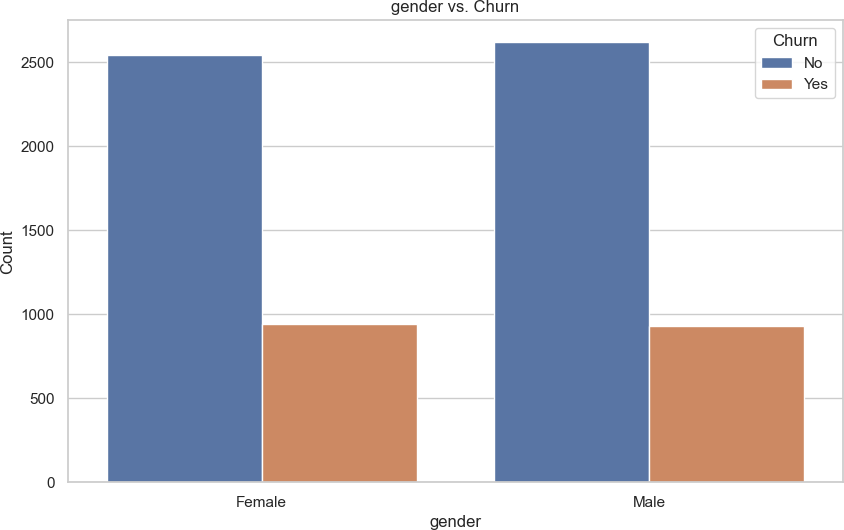
categorical\_features = ['gender', 'InternetService', 'Contract',␣

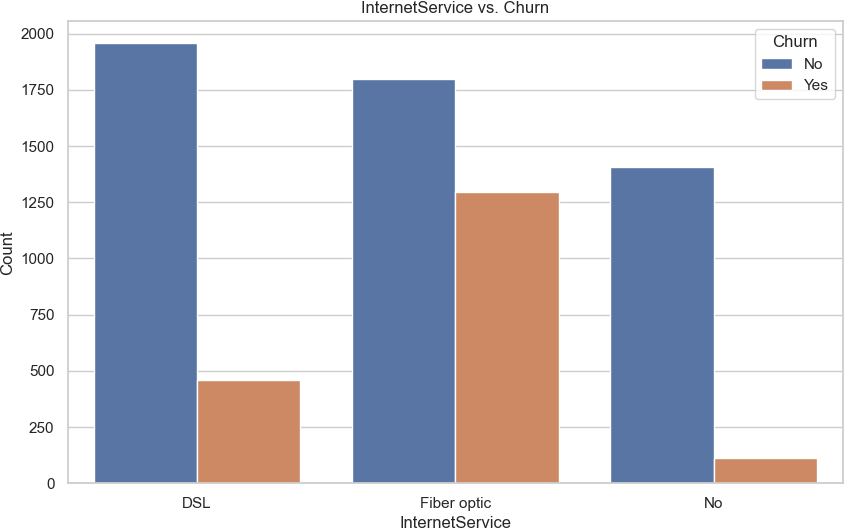
𝗌'PaymentMethod']

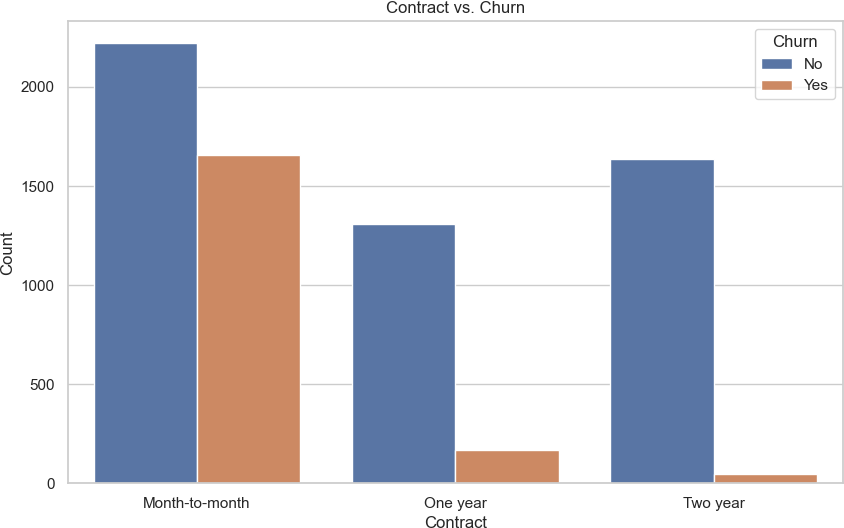
**for** feature **in** categorical\_features: plt.figure(figsize=(10, 6))

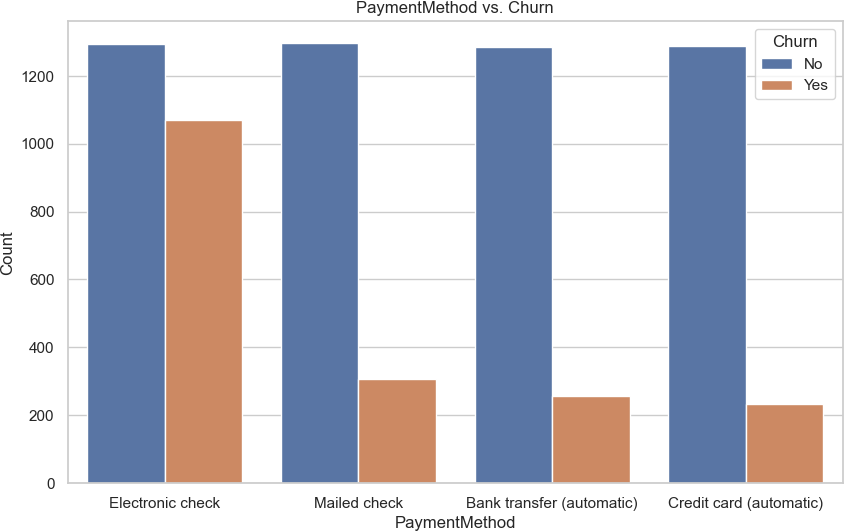
sns.countplot(data=data\_cleaned, x=feature, hue='Churn') plt.title(f'**{**feature**}** vs. Churn')

plt.xlabel(feature) plt.ylabel('Count') plt.legend(title='Churn') plt.show()









## Interpretation of Gender vs. Churn:

* + - * **Observation:** Similar churn rates for both males and females.

**Insight:** Gender does not appear to be a significant predictor of churn, suggesting that other factors are more influential.

## Interpretation of InternetService vs. Churn:

* + - * **Fiber Optic:** Higher churn rate compared to DSL and those with no internet service.
      * **DSL and No Internet Service:** Lower churn rates, with most customers staying.

**Insight:** Internet service type impacts churn, with Fiber optic users more likely to churn. This suggests a need to investigate and improve Fiber optic customer satisfaction.

## Interpretation of Contract vs. Churn:

* + - * **Month-to-Month Contracts:** Higher churn rate, as nearly equal numbers of customers churn and stay.
      * **One-Year and Two-Year Contracts:** Lower churn rates, with most customers staying.

**Insight:** Customers on longer-term contracts are less likely to churn, suggesting a strategy to encourage longer commitments could help reduce churn.

[13]:

## Interpretation of Payment Method Vs. Churn

* + - * **Electronic Check:** Higher churn rate compared to other payment methods.
      * **Mailed Check, Bank Transfer, Credit Card:** Lower churn rates, with most customers staying.

**Insight:** Customers using electronic checks are more likely to churn, indicating potential dissat- isfaction with this payment method or its associated convenience/fees. This suggests focusing on understanding and improving the experience for these customers.

*# Convert 'Churn' column to numeric: 1 for 'Yes', 0 for 'No'*

data\_cleaned['Churn'] = data\_cleaned['Churn'].apply(**lambda** x: 1 **if** x == 'Yes'␣

𝗌**else** 0)

*# Correlation heatmap for numerical features*

plt.figure(figsize=(12, 8))

correlation = data\_cleaned[['tenure', 'MonthlyCharges', 'TotalCharges',␣

𝗌'Churn']].corr()

sns.heatmap(correlation, annot=**True**, cmap='coolwarm', fmt='.2f') plt.title('Correlation Heatmap')

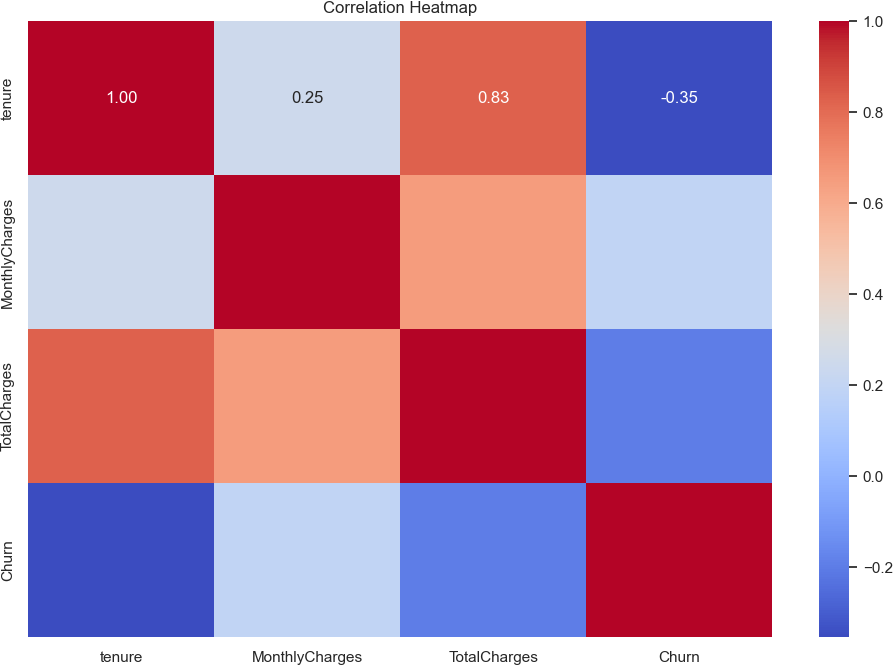
plt.show()

/var/folders/67/hl77bzs97pggp3g9r\_hnl9kw0000gn/T/ipykernel\_1668/1760577846.py:2: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame. Try using .loc[row\_indexer,col\_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas- docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy

data\_cleaned['Churn'] = data\_cleaned['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)



[ ]:

## Interpretation of Correlation Heatmap

* + - * **Tenure and Churn:** Moderate negative correlation (-0.35) — longer tenure means lower churn.
      * **TotalCharges and Tenure:** Strong positive correlation (0.83) — higher total charges usu- ally mean longer tenure.

## Insight:

* + - * **Retention Focus:** Shorter-tenure customers are more prone to churn, indicating a need for early engagement.