```
# STEP 1: SETUP & DATA LOADING

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Load the dataset
# Ensure the filename matches exactly what you uploaded
df = pd.read_csv('WA_Fn-UseC_-Telco-Customer-Churn.csv')

# Display first 5 rows
df.head()
```

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

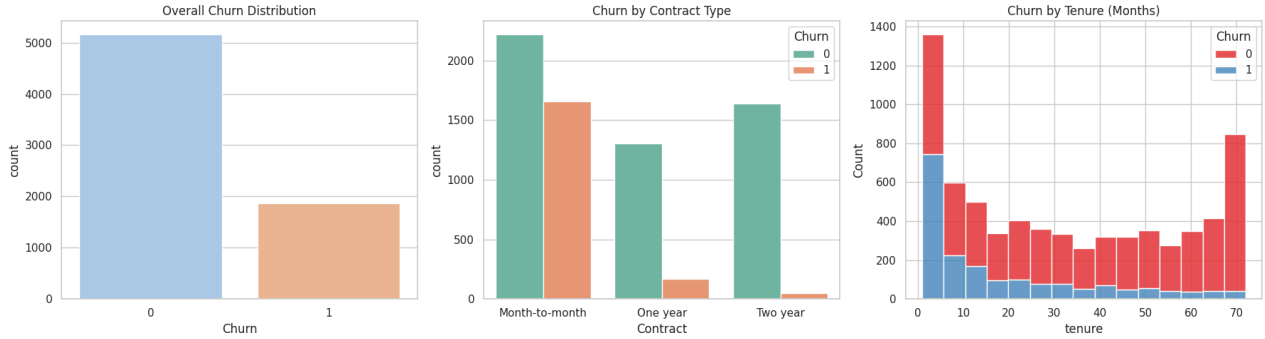
5 rows × 21 columns

```
# STEP 2: DATA CLEANING

# 1. Convert 'TotalCharges' to numeric (forcing errors to NaN)
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')

# 2. Drop rows with missing values (11 rows found)
df.dropna(inplace=True)

# 3. Drop 'customerID' (It is unique ID and has no predictive power)
df.drop(columns=['customerID'], inplace=True)

# 4. Convert Target 'Churn' to binary (1 for Yes, 0 for No)
df['Churn'] = df['Churn'].map({'Yes': 1, 'No': 0})

# Verify the changes
print("Data Cleaned Successfully!")
df.info()
```

```
Data Cleaned Successfully!
<class 'pandas.core.frame.DataFrame'>
Index: 7032 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   gender            7032 non-null   object
 1   SeniorCitizen     7032 non-null   int64
 2   Partner           7032 non-null   object
 3   Dependents        7032 non-null   object
 4   tenure            7032 non-null   int64
 5   PhoneService      7032 non-null   object
 6   MultipleLines     7032 non-null   object
 7   InternetService   7032 non-null   object
 8   OnlineSecurity    7032 non-null   object
 9   OnlineBackup      7032 non-null   object
 10  DeviceProtection  7032 non-null   object
 11  TechSupport       7032 non-null   object
 12  StreamingTV       7032 non-null   object
 13  StreamingMovies   7032 non-null   object
 14  Contract          7032 non-null   object
 15  PaperlessBilling  7032 non-null   object
 16  PaymentMethod     7032 non-null   object
 17  MonthlyCharges    7032 non-null   float64
 18  TotalCharges      7032 non-null   float64
 19  Churn             7032 non-null   int64
dtypes: float64(2), int64(3), object(15)
memory usage: 1.1+ MB
```

```
# STEP 3: VISUALIZATION

# Set the style
sns.set(style="whitegrid")

# Create a figure with subplots for a clean layout
fig, axes = plt.subplots(1, 3, figsize=(18, 5))

# Plot 1: Churn Distribution
sns.countplot(x='Churn', data=df, ax=axes[0], palette='pastel')
axes[0].set_title('Overall Churn Distribution')

# Plot 2: Contract Type vs Churn
sns.countplot(x='Contract', hue='Churn', data=df, ax=axes[1], palette='Set2')
axes[1].set_title('Churn by Contract Type')

# Plot 3: Tenure Distribution
sns.histplot(x='tenure', hue='Churn', data=df, multiple='stack', ax=axes[2], palette='Set1')
axes[2].set_title('Churn by Tenure (Months)')

plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2455415828.py:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

  sns.countplot(x='Churn', data=df, ax=axes[0], palette='pastel')
```



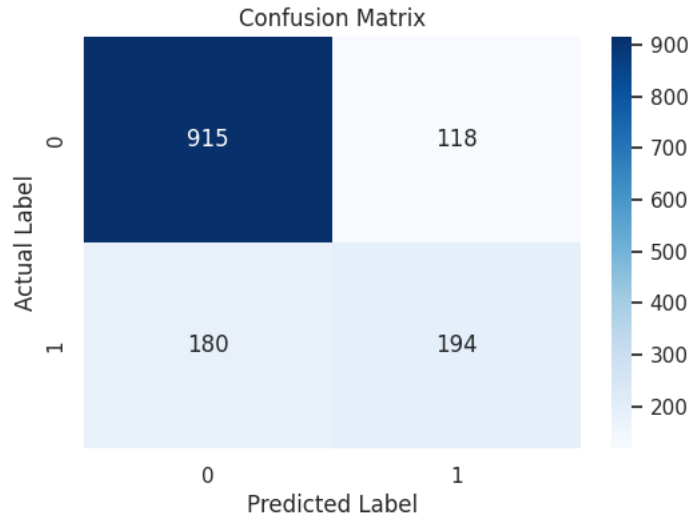

```
# STEP 4: PREPROCESSING

# 1. One-Hot Encoding (Convert text columns to numbers)
df_dummy = pd.get_dummies(df, drop_first=True)

# 2. Define Features (X) and Target (y)
X = df_dummy.drop('Churn', axis=1)
y = df_dummy['Churn']

# 3. Split the data (80% Train, 20% Test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Check the results
print(f"Original Features: {df.shape[1]}")
print(f"Encoded Features:  {df_dummy.shape[1]}")
print(f"Training Samples:  {X_train.shape[0]}")
print(f"Testing Samples:   {X_test.shape[0]}")
```

```
Original Features: 20
Encoded Features:  31
Training Samples:  5625
Testing Samples:   1407
```

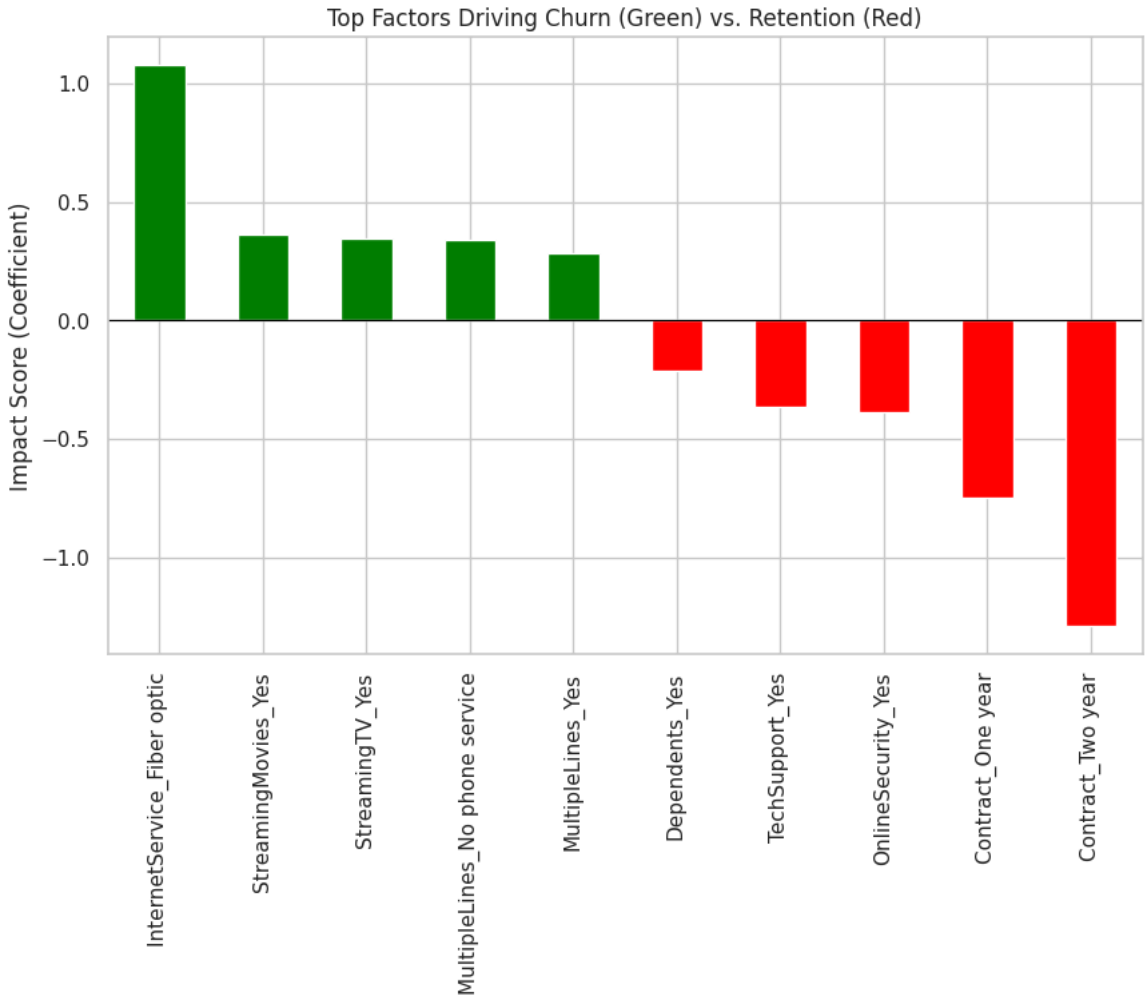
```
# STEP 5: MODEL TRAINING

# 1. Train the Model
# We increase max_iter to 5000 to ensure the math converges
model = LogisticRegression(max_iter=5000)
model.fit(X_train, y_train)

# 2. Make Predictions
y_pred = model.predict(X_test)
```

```
# 3. Evaluate
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy:.2f}")
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# 4. Confusion Matrix Visualization
plt.figure(figsize=(6, 4))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.show()
```

```
Model Accuracy: 0.79

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.89      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
```




```
# STEP 6: FEATURE IMPORTANCE

# 1. Get Model Coefficients
weights = pd.Series(model.coef_[0], index=X.columns.values)
weights = weights.sort_values(ascending=False)

# 2. Visualize Top Positive and Negative Factors
plt.figure(figsize=(10, 6))

# Combine top 5 churn drivers (positive) and top 5 retention drivers (negative)
top_weights = pd.concat([weights.head(5), weights.tail(5)])

# Create bar plot
colors = ['red' if x < 0 else 'green' for x in top_weights]
top_weights.plot(kind='bar', color=colors)

plt.title('Top Factors Driving Churn (Green) vs. Retention (Red)')
plt.ylabel('Impact Score (Coefficient)')
plt.axhline(0, color='black', linewidth=0.8)
plt.show()

# Print specific insights
print("Top Churn Driver:", weights.idxmax())
print("Top Retention Driver:", weights.idxmin())
```




```
Top Churn Driver: InternetService_Fiber optic
Top Retention Driver: Contract_Two year
```