**NAAN MUDHALVAN PHASE-4**

**DATA ANALYSTIC WITH COGNOS**

**1. Data Preparation:**

Load the cleaned dataset (cleaned\_telco\_customer\_churn.csv) into your analysis environment.

import pandas as pd

data = pd.read\_csv('cleaned\_telco\_customer\_churn.csv')

**2. Visualization using IBM Cognos:**

Connect IBM Cognos to your cleaned dataset.

Create interactive dashboards and reports to visualize churn patterns, retention rates, and key factors influencing churn.

Visualize customer segments, tenure, monthly charges, and contract details to identify patterns.

**3. Machine Learning Model:**

**3.1. Feature Selection:**

Identify relevant features for the predictive model. Features like tenure, monthly charges, contract details, and payment methods are often important.

**3.2. Data Splitting**:

Split the data into features (X) and target variable (y). Typically, 'churn' will be your target variable.

X = data.drop(columns=['churn'])

y = data['churn']

**3.3. Model Selection and Training:**

Choose an appropriate machine learning algorithm . Random Forest, Logistic Regression and train the model using the training data.

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

from sklearn.ensemble import RandomForestClassifier

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

**3.4. Model Evaluation:**

Evaluate the model's performance using the test data.

from sklearn.metrics import accuracy\_score, classification\_report

predictions = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, predictions)

report = classification\_report(y\_test, predictions)

print(f'Accuracy: {accuracy}')

print(f'Classification Report:\n{report}'

**Machine Learning algorithms to build a Predictive Model:**

import pandas as pd

import numpy as np

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.preprocessing import OneHotEncoder

data = pd.read\_csv('cleaned\_telco\_customer\_churn.csv')

data = data.drop(columns=['customerID']

X = data.drop(columns=['Churn'])

y = data['Churn'] # Target variable

non\_numeric\_cols = X.select\_dtypes(include=['object']).columns

X[non\_numeric\_cols] = X[non\_numeric\_cols].astype('category')

encoder = OneHotEncoder()

X\_encoded = encoder.fit\_transform(X).toarray()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_encoded, y, test\_size=0.2, random\_state=42)

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

predictions = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, predictions)

report = classification\_report(y\_test, predictions)

print(f'Accuracy: {accuracy}')

print(f'Classification Report:\n{report}')

**OUTPUT:**

Accuracy: 0.7934705464868701

Classification Report:

precision recall f1-score support

No 0.83 0.91 0.87 1036

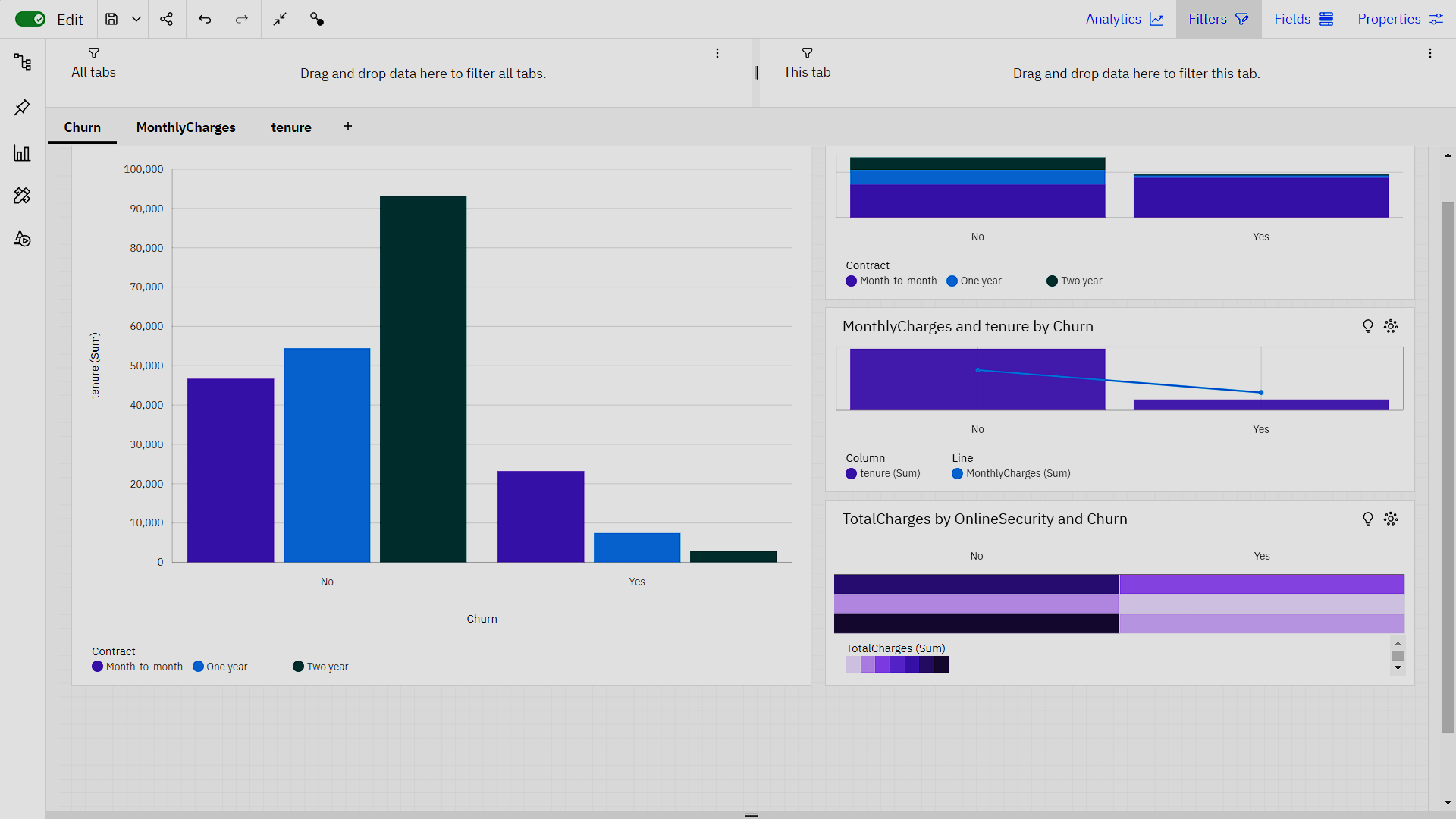
Yes 0.66 0.46 0.54 373

accuracy 0.79 1409

macro avg 0.74 0.69 0.70 1409

weighted avg 0.78 0.79 0.78 1409

**VISUALIZATION:**



**REPORT:**



* **Conclusion and Recommendations:**

Summarize the findings from your analysis.

Provide actionable recommendations based on the visualizations and the predictive model. For example, suggest personalized retention strategies for high-risk customers.