Applications in Business and Industrial problem

Customer Churn Prediction in Telecommunication Industry


01 Introduction
02 Modeling
03 Results
04 conclusions
Agenda

Introduction

What are applications in Business and Industrial problem ??

1.Customer Lifetime Value Modeling

2.Churn Modeling

3. Customer Segmentation

4.Recommendation Engines

5. Dynamic Pricing

Introduction

Churn Modeling

Customer Churn Prediction
in Telecommunication Industry

Telecommunication

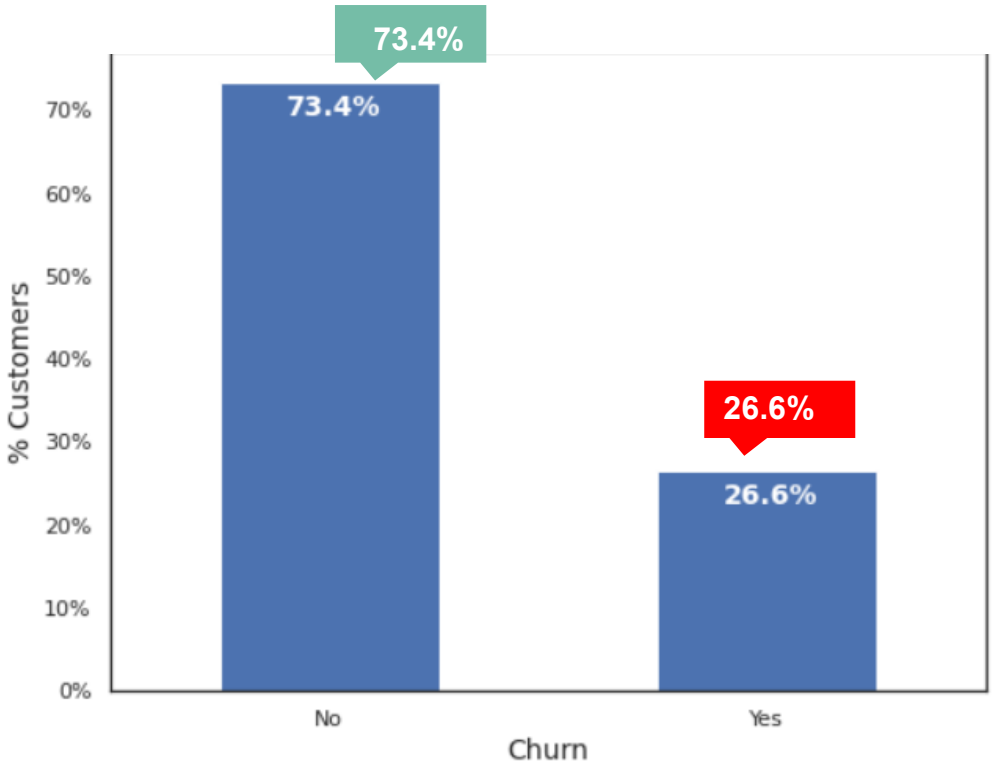
What is Telecommunication ?

Why is it important ?

How to use telecom ?

Introduction

Churn Rate

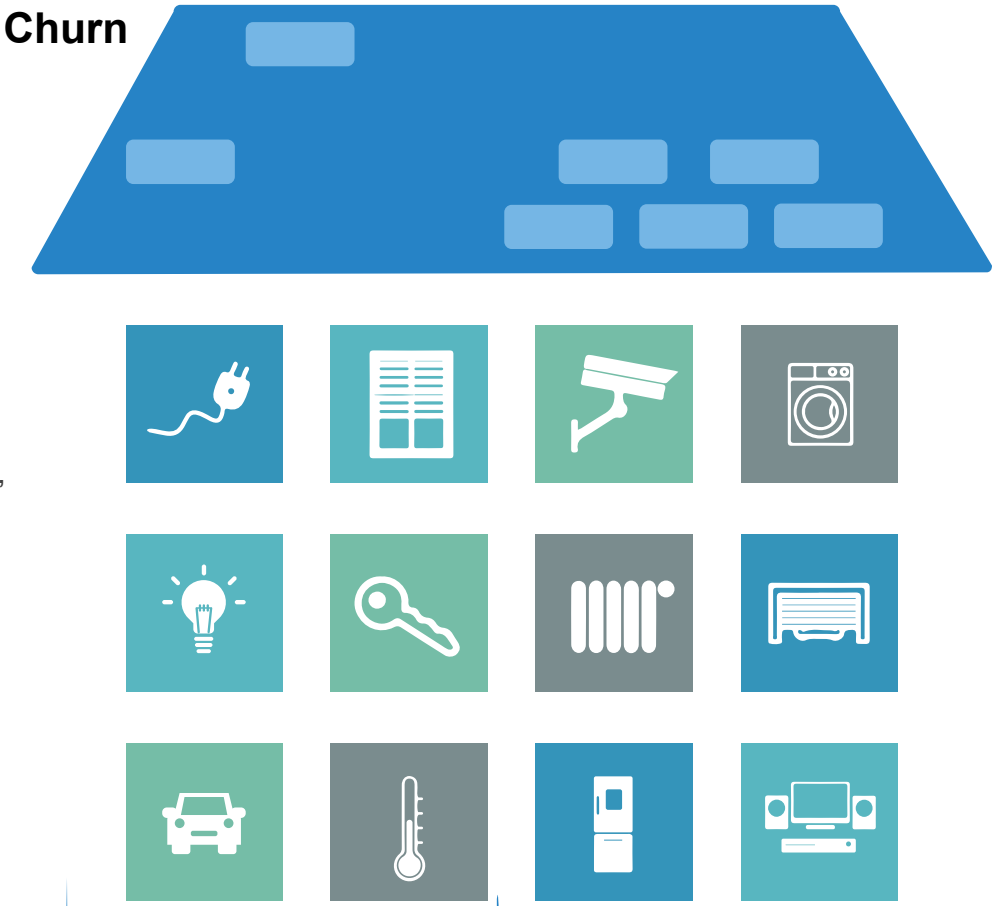


How to know
Who will Churn ?

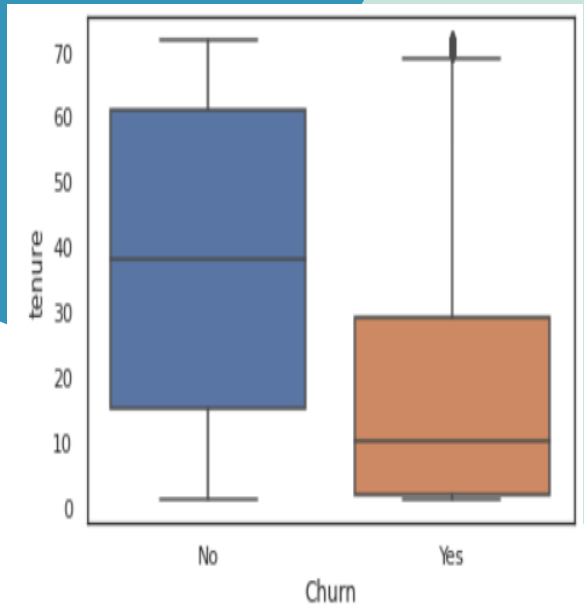
Introduction

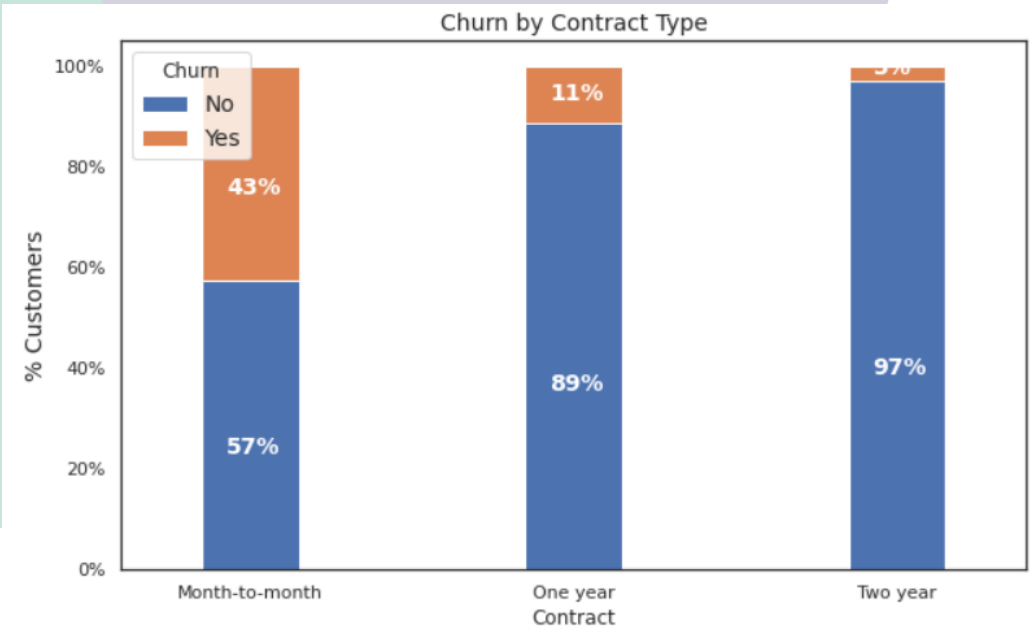
Dataset in use case is Telecom Customer Churn

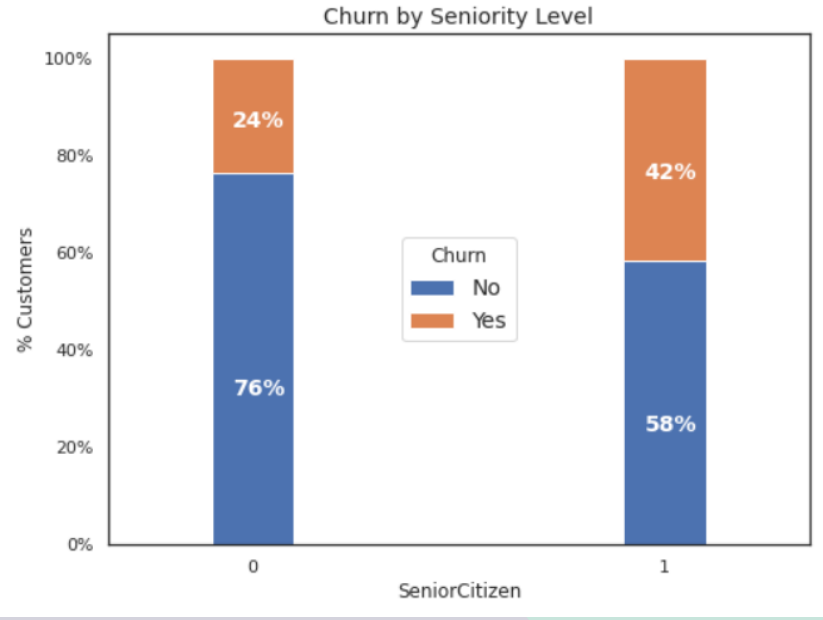
Factor	
Services	phone, multiple lines, internet, online security, online backup etc.
Customer account information	contract, payment method, paperless billing, monthly charges etc.
Demographic	gender, age range etc.



VARIABLE







Boxplot

CHURN
COMPARE
WITH TENURE

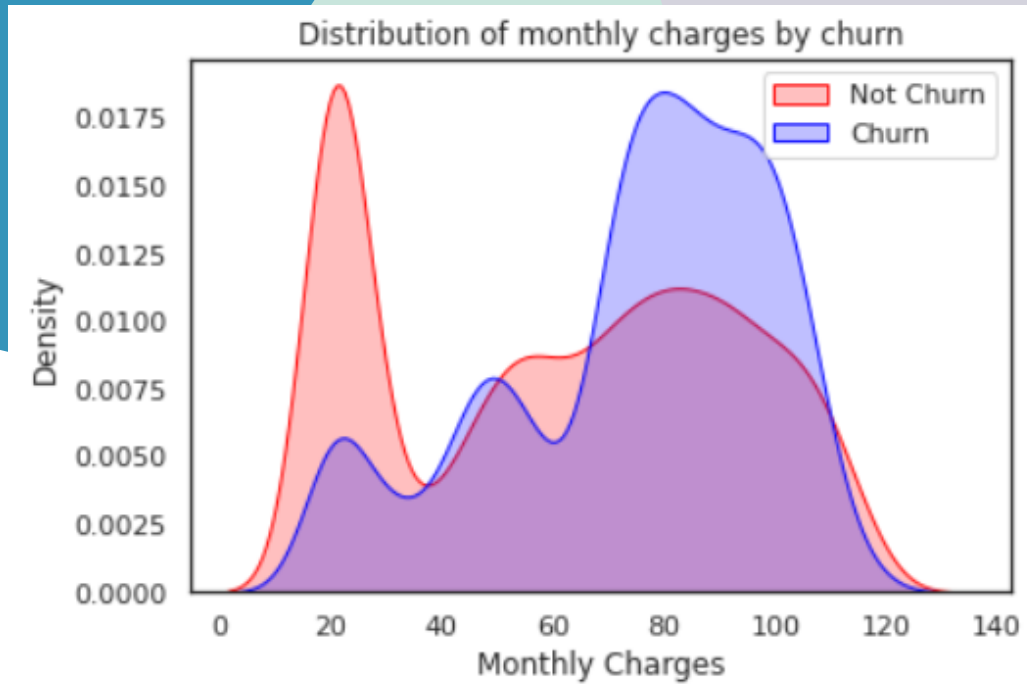
Stacked bar

Churn by
Contract Type

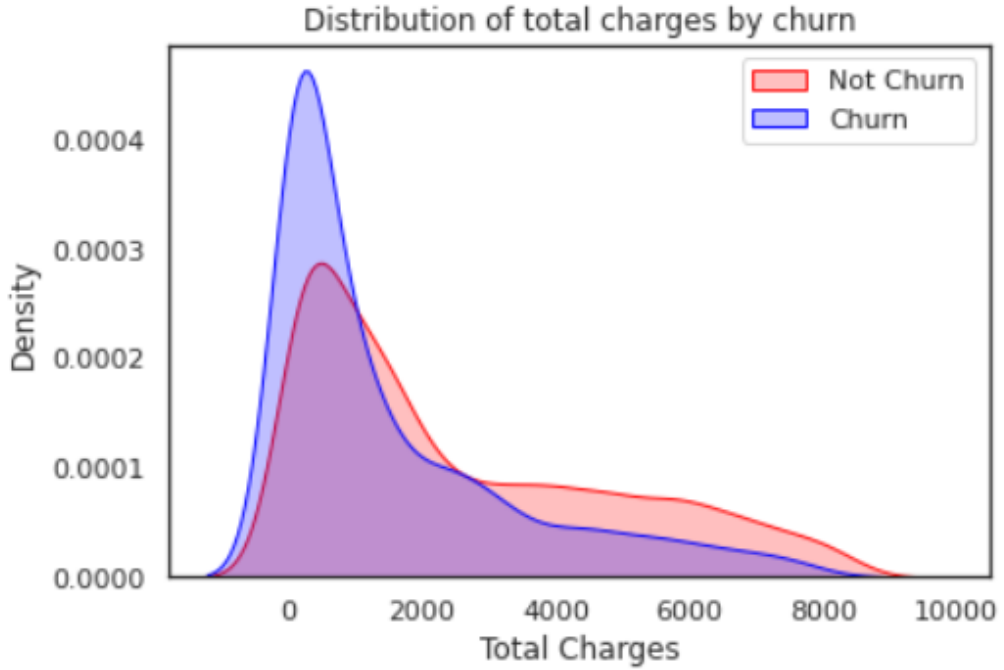
Stacked bar

Churn by
Seniority Level

VARIABLE



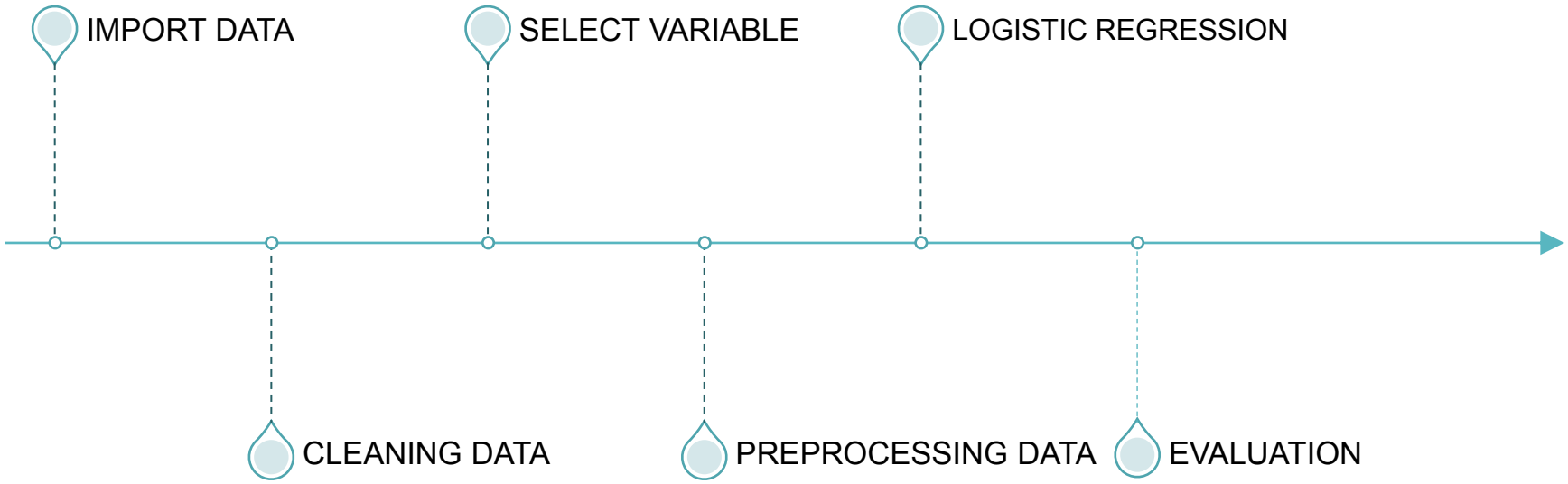
Distribution of monthly charges by churn



Distribution of total charges by churn

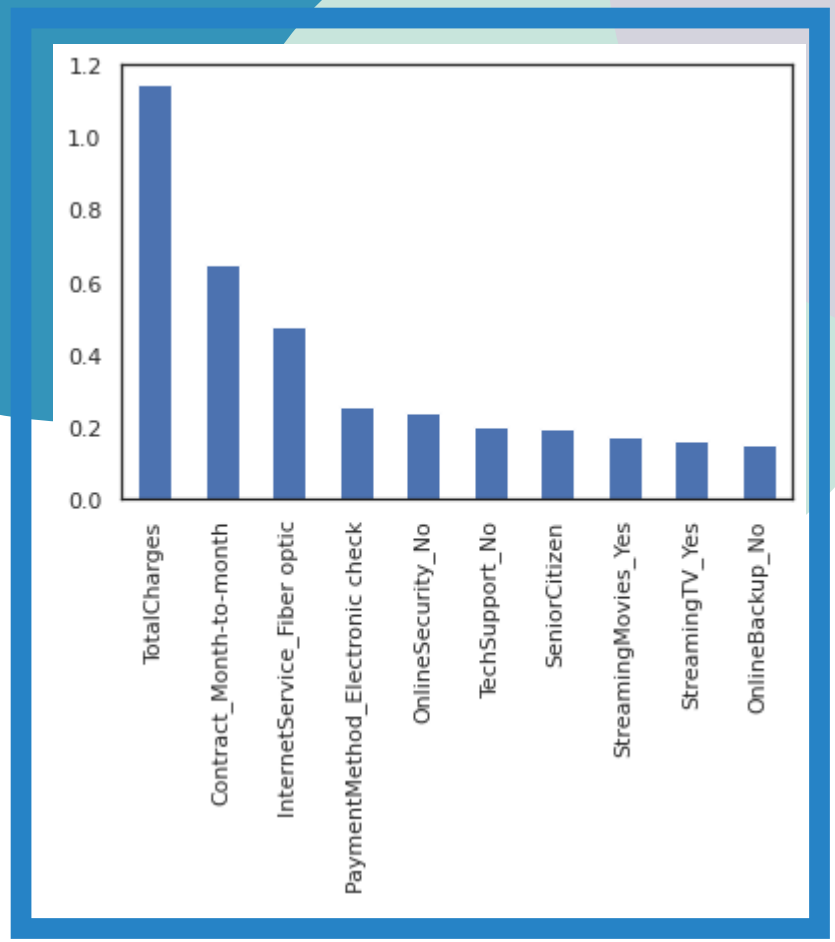
2.Modeling

Modeling


IMPORT DATA
SELECT VARIABLE
LOGISTIC REGRESSION
CLEANING DATA
PREPROCESSING DATA
EVALUATION


3.Results

RESULT LOGISTIC REGRESSION



Modeling

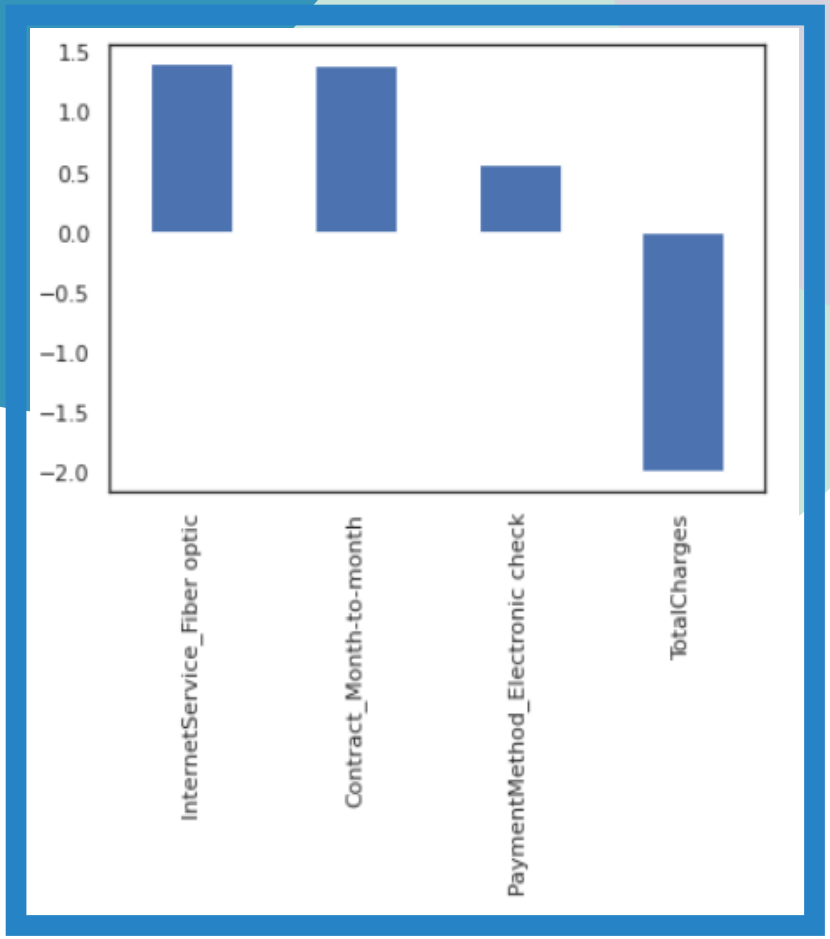
Beta 0 :
[-1.06351583]

Coefficient :
[[1.93539116e-01 -2.98146524e+00 -1.36432342e-01 1.14694845e+00 1.51836880e-02 -1.49579528e-02 3.81211240e-02 -3.78953888e-02 1.25994493e-02 -1.23737141e-02 7.73479455e-02 -7.71222102e-02 -1.57712442e-01 7.73479455e-02 8.05902313e-02 -3.59136790e-01 4.78030485e-01 -1.18667959e-01 2.37173810e-01 -1.18667959e-01 -1.18280115e-01 1.50954645e-01 -1.18667959e-01 -3.20609501e-02 3.80355661e-02 -1.18667959e-01 8.08581285e-02 1.97465812e-01 -1.18667959e-01 -7.85721178e-02 -4.43637263e-02 -1.18667959e-01 1.63257421e-01 -5.53843321e-02 -1.18667959e-01 1.74278027e-01 6.45790317e-01 -8.91169605e-02 -5.56447621e-01 -1.32223750e-01 1.32449485e-01 -3.26690710e-02 -2.22089219e-01 2.56487631e-01 -1.50360638e-03]]

EVALUATION

Accuracy	: 0.80758
Precisions	: 0.78226
Recall	: 0.73205
F1 Score	: 0.74890

RESULT LOGISTIC REGRESSION



Modeling

Beta 0 :
[-2.53019529]

Coefficient :
[[-1.99163571 1.39213777 1.40070402 0.57168699]]

EVALUATION

Accuracy	: 0.78483
Precisions	: 0.74998
Recall	: 0.70295
F1 Score	: 0.71786

4.Conclusion

CONCLUSION

- Compare Evolution between of All factors and Top 4 factors

Evolution	All Factors	Top 4 Factors
Accuracy	0.80758	0.78483
Precisions	0.78226	0.74998
Recall	0.78226	0.70295
F1 Score	0.74890	0.71786

Highest Evolution

- Compare between Model of All factors and Top 4 factors

```
[-1.06351583]

[[ 1.93539116e-01 -2.98146524e+00 -1.36432342e-01 1.14694845e+00 1.51836880e-02 -
1.49579528e-02 3.81211240e-02 -3.78953888e-02 1.25994493e-02 -1.23737141e-02
7.73479455e-02 -7.71222102e-02 -1.57712442e-01 7.73479455e-02 8.05902313e-02 -
3.59136790e-01 4.78030485e-01                  3810e-01 -1.18667959e-01 -
1.18280115e-01 1.50954645e-01                  09501e-02 3.80355661e-02 -
1.18667959e-01 8.08581285e-02                  7959e-01 -7.85721178e-02 -
4.43637263e-02 -1.18667959e-01 1.63257421e-01 -5.53843321e-02 -1.18667959e-01
1.74278027e-01 6.45790317e-01 -8.91169605e-02 -5.56447621e-01 -1.32223750e-01
1.32449485e-01 -3.26690710e-02 -2.22089219e-01 2.56487631e-01 -1.50360638e-03]]
```

Complex

Top 4 Factors

```
[-2.53019529]

[-1.99163571 1.39213777 1.40070402 0.57168699]]
```

Non-complex and simplify

CONCLUSION

- Model of top 4 factors

$$Y = -2.53019529 + (-1.99163571)\mathbf{A} + (1.39213777)\mathbf{B} + (1.40070402)\mathbf{C} + (0.57168699)\mathbf{D}$$

Where

Beta 0 is -2.53019529

A is TotalCharges

B is Contract_Month-to-month

C is InternetService_Fiber optic

D is PaymentMethod_Electronic check

Coding

```
## IMPORT LIBRARY
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns # For creating plots
import matplotlib.ticker as mtick # For specifying the axes tick format
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix

sns.set(style = 'white')

#Upload file
from google.colab import files
uploaded = files.upload()

# READ FILE
telecom_cust = pd.read_csv('WA_Fn-UseC_-Telco-Customer-Churn.csv')
```

Coding

```
# Check column
telecom_cust.columns.values

# Check Data Type
telecom_cust.dtypes

# Converting Total Charges to a numerical data type.
telecom_cust.TotalCharges = pd.to_numeric(telecom_cust.TotalCharges, errors
='coerce')
telecom_cust.isnull().sum()

#Removing missing values BY DELETE MISSING DATA
telecom_cust.dropna(inplace = True)
#Remove customer IDs (ไม่ใช้)
df2 = telecom_cust.iloc[:,1:]
```

Coding

```
#Converting the predictor variable in a binary numeric variable เปลี่ยนตัว
แปล Y  เป็น 0 1 (1ย้าย 0 ไม่ย้าย)
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 สร้าง dummy ของ
งพวก Category
df_dummies = pd.get_dummies(df2)
df_dummies.head()
```

Coding

```
#### COMPARE RARIO YES NO
ax = (telecom_cust['Churn'].value_counts()*100.0 /len(telecom_cust)).plot(kind='bar',
 stacked = True,rot = 0,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)
```

Coding

```
# 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)
```

Coding

```
Logistic Regression
# 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 70 : 30
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, ra
ndom_state=123)
```

Coding

```
# 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)) #ความแม่นยำ
# 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 (model.intercept_) ## ค่าคงที่ (beta0)
print (model.coef_) ## ค่าสัมประสิทธิ์
```

Coding

```
# 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 (model.intercept_) ## ค่าคงที่ (beta0)
print (model.coef_) ## ค่าสัมประสิทธิ์

from sklearn.metrics import precision_recall_fscore_support
precision_recall_fscore_support(y_test, prediction_test, average='macro')
##precisions, recalls, F1-scores , supports(None)
```

Coding

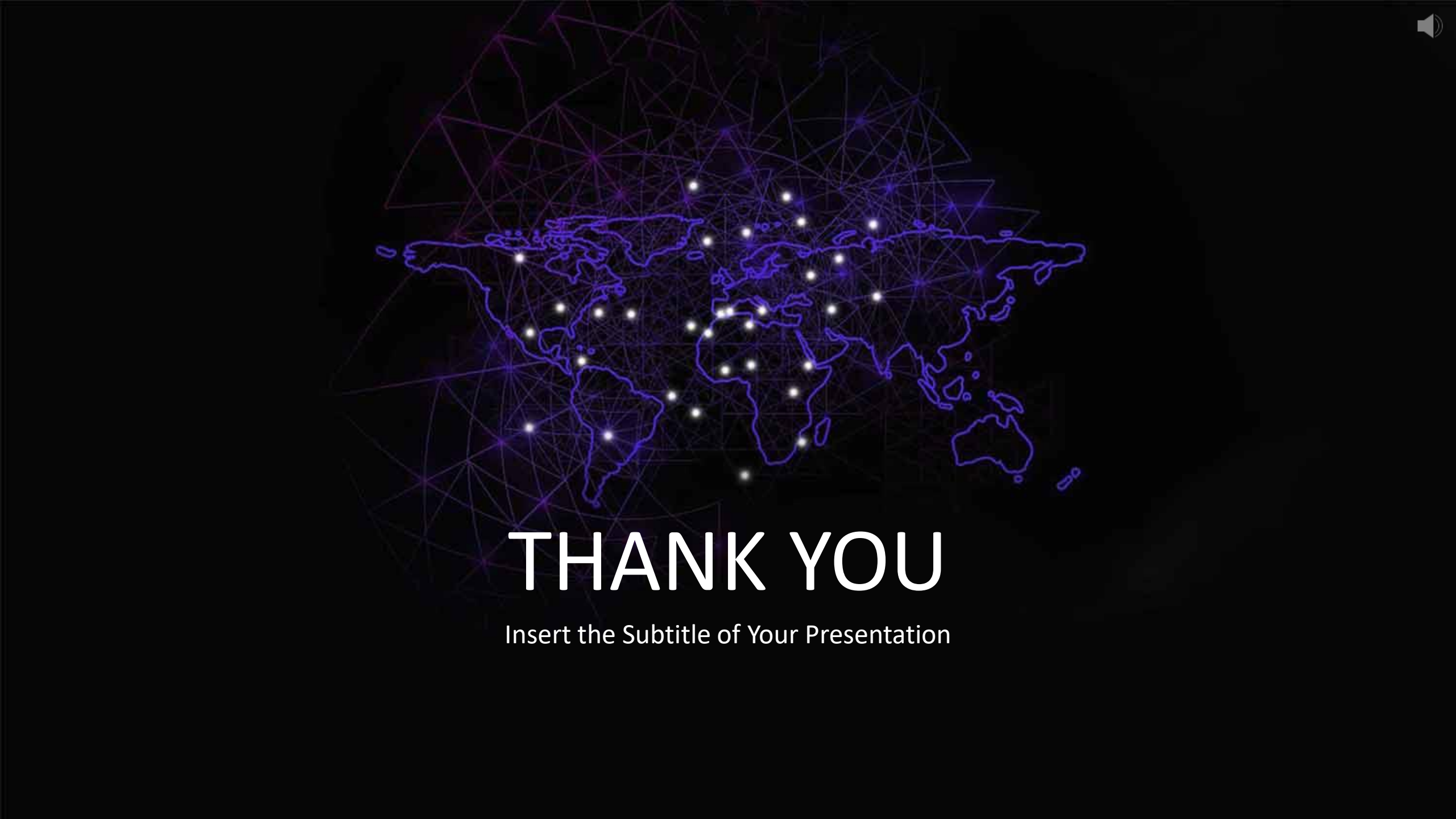
```
  #### NEW MODEL WITH MOST 5 WEIGTH VARIABLE
'TotalCharges','Contract_Month-to-
month','InternetService_Fiber optic','PaymentMethod_Electronic check'

# We will use the data frame where we had created dummy variables
y = df_dummies['Churn'].values
X = df_dummies[['TotalCharges','Contract_Month-to-
month','InternetService_Fiber optic','PaymentMethod_Electronic check']]

# 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
```

Coding

```
# Create Train & Test Data 70 : 30
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, ra
ndom_state=123)

# 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)) #ความแม่นยำ
```

Coding

```
# 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 (model.intercept_) ## ค่าคงที่ (beta0)
print (model.coef_) ## ค่าสัมประสิทธิ์

from sklearn.metrics import precision_recall_fscore_support
precision_recall_fscore_support(y_test, prediction_test, average='macro')
##precisions, recalls, F1-scores , supports(None)
```



THANK YOU

Insert the Subtitle of Your Presentation