Logistic Regression Consulting Project

1. Title Page:

◦ Project Title: Binary Customer Churn

◦ Your Name: Saakshi(B033) and Anika(B040)

◦ Date: November 2 , 2023

◦ Institution Name : Mukesh Patel School of Technology Management and Engineering

◦ Contact Information (email, phone) – 7700997655 and saakshi.jain134@nmims.edu.in

7506751975 and anika.mayekar088@nmims.edu.in

Kaggel link for dataset

Customer Churn Dataset : https://www.kaggle.com/datasets/muhammadshahidazeem/customer-churn-dataset

Bank Customer Churn Dataset: https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset

Github Link : https://github.com/saakshijain2022/Kaggle_Project

2. Table of Contents:

3. Executive Summary:

The "Binary Customer Churn" project, conducted by Saakshi and Anika from Mukesh Patel School of Technology Management and Engineering, aimed to address the challenge of customer churn in a marketing agency. The project leveraged logistic regression using PySpark and Scikit-Learn to develop predictive models for customer churn. This summary provides a concise overview of the project's key aspects:

Objective: The primary goal of this project was to predict customer churn accurately. By doing so, the marketing agency could identify at-risk clients and assign dedicated account managers to reduce churn.

Dataset: The project utilized the 'customer_churn.csv' dataset, which contained vital customer information, including age, total purchase history, the presence of an account manager, years as a customer, and the number of websites using the service.

Significance of Account Manager Field: The presence of an account manager emerged as a crucial feature, representing personalized support and influencing customer retention.

Churn Significance: Customer churn, the rate at which clients discontinue their relationship with a company, was highlighted as a pivotal metric with substantial business implications.

Tools and Techniques: PySpark and Scikit-Learn were the primary tools for data analysis and model development. Exploratory data analysis (EDA) techniques helped discover key correlations between customer attributes and churn.

Results and Findings: The project provided insights into influential customer features, model performance, and interesting correlations within the data. The AUC-ROC metric was used to assess the model's predictive capabilities.

Discussion: The results were interpreted in the context of the marketing agency's business. Challenges faced during the project, such as data quality issues, were addressed. The results were compared to the project's objectives to assess success.

Conclusion: Key findings emphasized the practical significance of the project's results. Lessons learned were highlighted, and the overall success of the project was summarized.

Future Work: Opportunities for future projects were outlined, with a focus on refining churn prediction and enhancing customer retention strategies.

The project report covers these aspects comprehensively, providing a valuable resource for the marketing agency to reduce churn, retain customers, and foster long-term client relationships.

4. Introduction:

In this consulting project, we leverage the power of logistic regression using PySpark and Scikit-Learn to address a critical challenge faced by a marketing agency. The agency serves numerous clients by creating advertisements for their websites. However, they have observed a significant issue – a substantial number of their clients are churning, meaning they are discontinuing the use of the agency's services. Currently, the agency assigns account managers somewhat randomly to their clients. To address this challenge, the agency has sought our assistance in developing a machine learning model to predict customer churn. By doing so, they aim to accurately identify those clients most at risk of churning and, in turn, assign them dedicated account managers.

The dataset we work with, named 'customer_churn.csv,' contains valuable information to help predict customer churn. It includes several key fields:

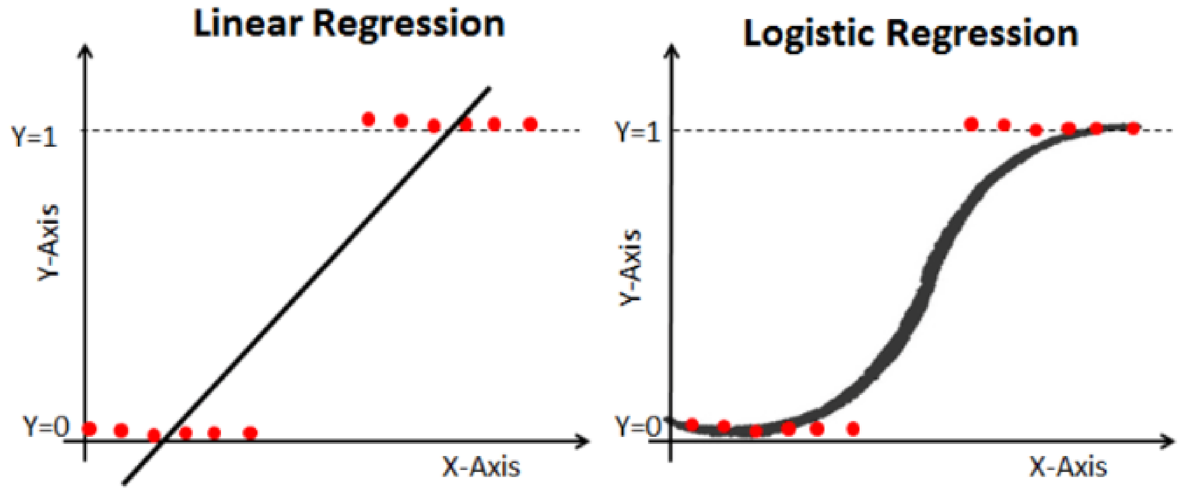
Name : Name of the latest contact at Company
Age: Customer Age
Total_Purchase: Total Ads Purchased
Account_Manager: Binary 0=No manager, 1= Account manager assigned
Years: Totally Years as a customer
Num_sites: Number of websites that use the service.
Onboard_date: Date that the name of the latest contact was onboarded
Location: Client HQ Address
Company: Name of Client Company

The primary objective is to predict which clients are most likely to churn based on the provided data, even though we don't have the churn label in the dataset. The significance of the "Account_Manager" field should not be underestimated; it can potentially be a vital feature in predicting customer churn. Churn is a pivotal metric for businesses, as it can significantly impact their revenue and overall profitability. To proactively reduce churn and maintain a stable customer base, it is crucial to identify and predict potential churners.

This project demonstrates the application of logistic regression as a classification algorithm in the context of customer churn prediction. It showcases the pivotal role machine learning can play in retaining valuable customers and fostering long-term relationships with clients.

Sigmoid Function:

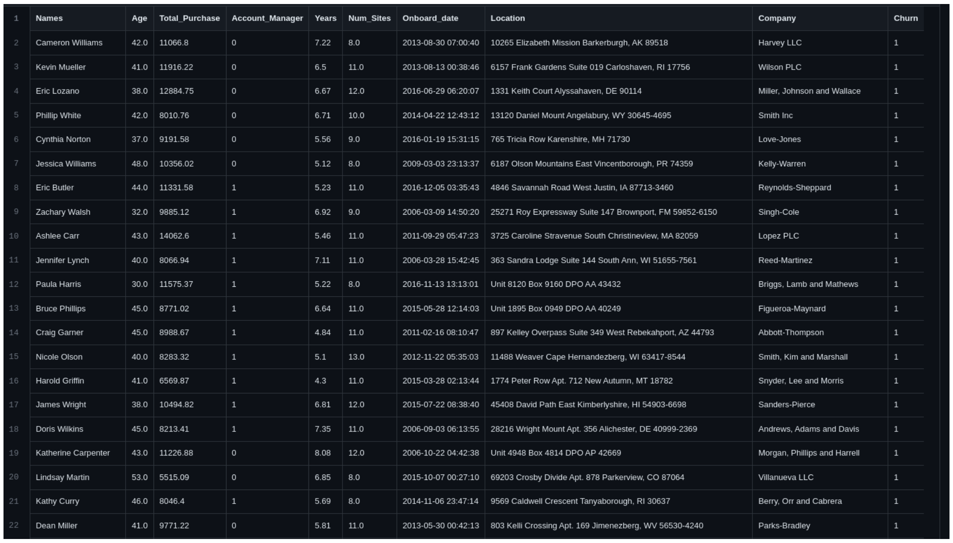
$$p = \frac{1}{1 + e^{-y}}$$



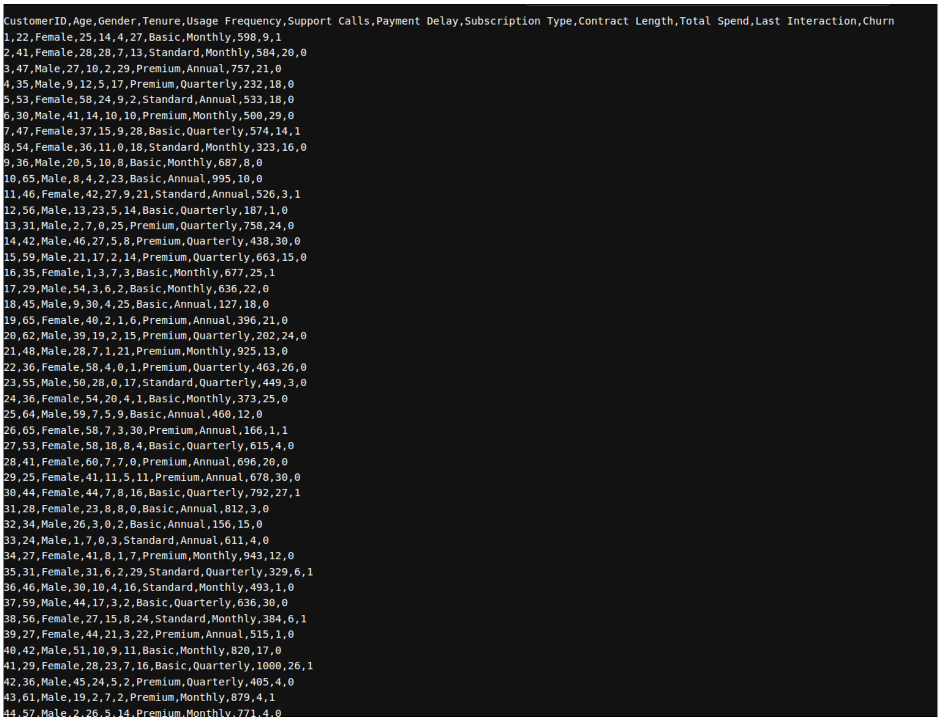
5. Methodology:

The methodology employed for this project includes several key aspects:

1. Data Collection: The project utilizes data from the 'customer_churn.csv' dataset. This dataset contains valuable information about customers, such as their age, total purchase history, the presence of an account manager, years as a customer, the number of websites using the service, and more.



Names	Age	Total_Purchase	Account_Manager	Years	Num_Sites	Onboard_date	Location	Company	Churn
Cameron Williams	42.0	11066.8	0	7.22	8.0	2013-08-30 07:00:40	10265 Elizabeth Mission Barkerburgh, AK 89518	Harvey LLC	1
Kevin Mueller	41.0	11916.22	0	6.5	11.0	2013-08-13 00:38:46	6157 Frank Gardens Suite 019 Carloshaven, RI 17756	Wilson PLC	1
Eric Lozano	38.0	12884.75	0	6.67	12.0	2016-06-29 06:20:07	1331 Keith Court Alyssahaven, DE 90114	Miller, Johnson and Wallace	1
Phillip White	42.0	8010.76	0	6.71	10.0	2014-04-22 12:43:12	13120 Daniel Mount Angelabury, WY 30645-4695	Smith Inc	1
Cynthia Norton	37.0	9191.58	0	5.56	9.0	2016-01-19 15:31:15	765 Tricia Row Karenshire, MH 71730	Love-Jones	1
Jessica Williams	48.0	10356.02	0	5.12	8.0	2009-03-03 23:13:37	6187 Olson Mountains East Vincentborough, PR 74359	Kelly-Warren	1
Eric Butler	44.0	11331.58	1	5.23	11.0	2016-12-05 03:35:43	4846 Savannah Road West Justin, IA 87713-3460	Reynolds-Sheppard	1
Zachary Walsh	32.0	9885.12	1	6.92	9.0	2006-03-09 14:50:20	25271 Roy Expressway Suite 147 Brownport, FM 59852-6150	Singh-Cole	1
Ashlee Carr	43.0	14062.6	1	5.46	11.0	2011-09-29 05:47:23	3725 Caroline Stravenue South Christineview, MA 82059	Lopez PLC	1
Jennifer Lynch	40.0	8066.94	1	7.11	11.0	2006-03-28 15:42:45	363 Sandra Lodge Suite 144 South Ann, WI 51655-7561	Reed-Martinez	1
Paula Harris	30.0	11575.37	1	5.22	8.0	2016-11-13 13:13:01	Unit 8120 Box 9160 DPO AA 43432	Briggs, Lamb and Mathews	1
Bruce Phillips	45.0	8771.02	1	6.64	11.0	2015-05-28 12:14:03	Unit 1895 Box 0949 DPO AA 40249	Figueroa-Maynard	1
Craig Garner	45.0	8988.67	1	4.84	11.0	2011-02-16 08:10:47	897 Kelley Overpass Suite 349 West Rebekahport, AZ 44793	Abbott-Thompson	1
Nicole Olson	40.0	8283.32	1	5.1	13.0	2012-11-22 05:35:03	11488 Weaver Cape Hernandezberg, WI 63417-8544	Smith, Kim and Marshall	1
Harold Griffin	41.0	6569.87	1	4.3	11.0	2015-03-28 02:13:44	1774 Peter Row Apt. 712 New Autumn, MT 18782	Snyder, Lee and Morris	1
James Wright	38.0	10494.82	1	6.81	12.0	2015-07-22 08:38:40	45408 David Path East Kimberlyshire, HI 54903-6698	Sanders-Pierce	1
Doris Wilkins	45.0	8213.41	1	7.35	11.0	2006-09-03 06:13:55	28216 Wright Mount Apt. 356 Alichester, DE 40999-2369	Andrews, Adams and Davis	1
Katherine Carpenter	43.0	11226.88	0	8.08	12.0	2006-10-22 04:42:38	Unit 4948 Box 4814 DPO AP 42669	Morgan, Phillips and Harrell	1
Lindsay Martin	53.0	5515.09	0	6.85	8.0	2015-10-07 00:27:10	69203 Crosby Divide Apt. 878 Parkerview, CO 87064	Villanueva LLC	1
Kathy Curry	46.0	8046.4	1	5.69	8.0	2014-11-06 23:47:14	9569 Caldwell Crescent Tanyaborough, RI 30637	Berry, Orr and Cabrera	1
Dean Miller	41.0	9771.22	0	5.81	11.0	2013-05-30 00:42:13	803 Kelli Crossing Apt. 169 Jimenezberg, WV 56530-4240	Parks-Bradley	1



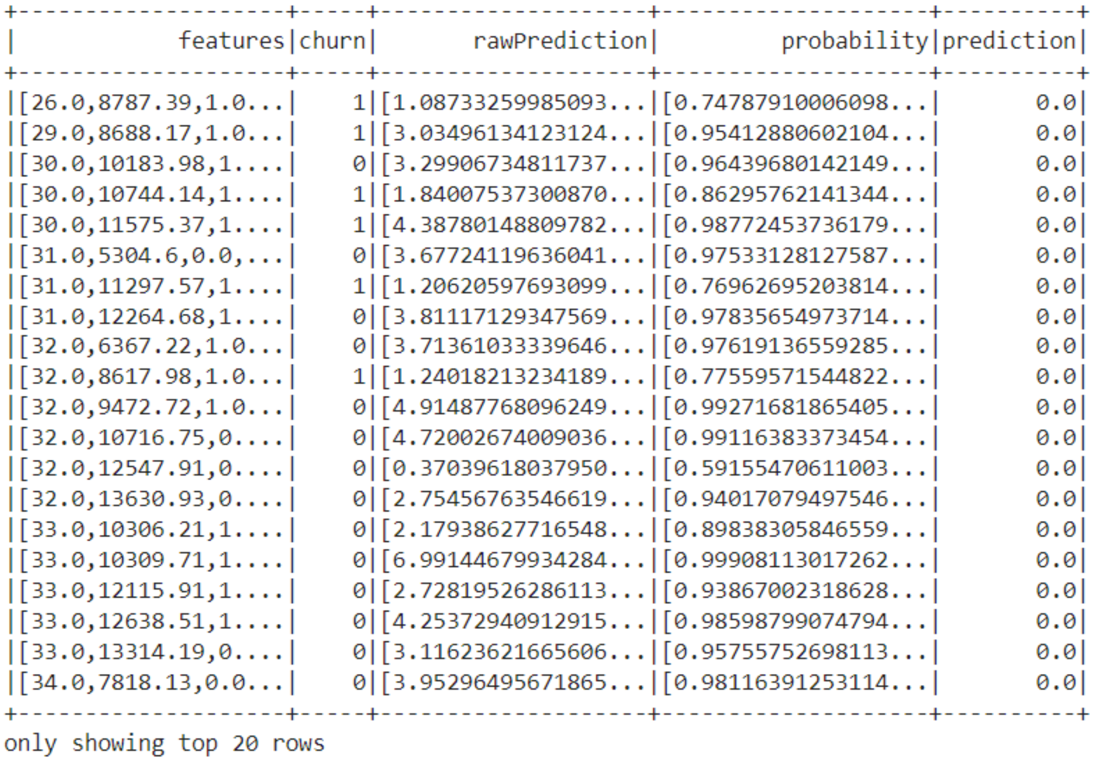
```
CustomerID,Age,Gender,Tenure,Usage Frequency,Support Calls,Payment Delay,Subscription Type,Contract Length,Total Spend,Last Interaction,Churn
1,22,Female,25,14,4,27,Basic,Monthly,598,9,1
2,41,Female,28,28,7,13,Standard,Monthly,584,20,0
3,47,Male,27,10,2,29,Premium,Annual,757,21,0
4,35,Male,9,12,5,17,Premium,Quarterly,232,18,0
5,53,Female,58,24,9,2,Standard,Annual,533,18,0
6,30,Male,41,14,10,10,Premium,Monthly,500,29,0
7,47,Female,37,15,9,28,Basic,Quarterly,574,14,1
8,54,Female,36,11,0,18,Standard,Monthly,323,16,0
9,36,Male,20,5,10,8,Basic,Monthly,687,8,0
10,65,Male,8,4,2,23,Basic,Annual,995,10,0
11,46,Female,42,27,9,21,Standard,Annual,526,3,1
12,56,Male,13,23,5,14,Basic,Quarterly,187,1,0
13,31,Male,2,7,0,25,Premium,Quarterly,758,24,0
14,42,Male,46,27,5,8,Premium,Quarterly,438,30,0
15,59,Male,21,17,2,14,Premium,Quarterly,663,15,0
16,35,Female,1,3,7,3,Basic,Monthly,677,25,1
17,29,Male,54,3,6,2,Basic,Monthly,636,22,0
18,45,Male,9,30,4,25,Basic,Annual,127,18,0
19,65,Female,40,2,1,6,Premium,Annual,396,21,0
20,62,Male,39,19,2,15,Premium,Quarterly,202,24,0
21,48,Male,28,7,1,21,Premium,Monthly,925,13,0
22,36,Female,58,4,0,1,Premium,Quarterly,463,26,0
23,55,Male,50,28,0,17,Standard,Quarterly,449,3,0
24,36,Female,54,20,4,1,Basic,Monthly,373,25,0
25,64,Male,59,7,5,9,Basic,Annual,460,12,0
26,65,Female,58,7,3,30,Premium,Annual,166,1,1
27,53,Female,58,18,8,4,Basic,Quarterly,615,4,0
28,41,Female,60,7,7,0,Premium,Annual,696,20,0
29,25,Female,41,11,5,11,Premium,Annual,678,30,0
30,44,Female,44,7,8,16,Basic,Quarterly,792,27,1
31,28,Female,23,8,8,0,Basic,Annual,812,3,0
32,34,Male,26,3,0,2,Basic,Annual,156,15,0
33,24,Male,1,7,0,3,Standard,Annual,611,4,0
34,27,Female,41,8,1,7,Premium,Monthly,943,12,0
35,31,Female,31,6,2,29,Standard,Quarterly,329,6,1
36,46,Male,30,10,4,16,Standard,Monthly,493,1,0
37,59,Male,44,17,3,2,Basic,Quarterly,636,30,0
38,56,Female,27,15,8,24,Standard,Monthly,384,6,1
39,27,Female,44,21,3,22,Premium,Annual,515,1,0
40,42,Male,51,10,9,11,Basic,Monthly,820,17,0
41,29,Female,28,23,7,16,Basic,Quarterly,1000,26,1
42,36,Male,45,24,5,2,Premium,Quarterly,405,4,0
43,61,Male,19,2,7,2,Premium,Monthly,879,4,1
44,57,Male,2,26,5,14,Premium,Monthly,771,4,0
```

2. Account Manager Field: The 'Account_Manager' field plays a significant role in customer churn prediction. This binary field indicates whether a customer has been assigned a dedicated account manager (1) or not (0). The presence of an account manager often signifies personalized support and customer management, which can influence a customer's decision to continue using the company's services or not.

3. Customer Churn Significance: Customer churn is a pivotal metric in business. It refers to the rate at which customers discontinue their relationship with a company by canceling subscriptions, discontinuing services, or no longer making purchases. Understanding and predicting customer churn is of utmost importance because it can substantially impact a company's revenue and profitability. Businesses aim to minimize churn to maintain a stable and profitable customer base.

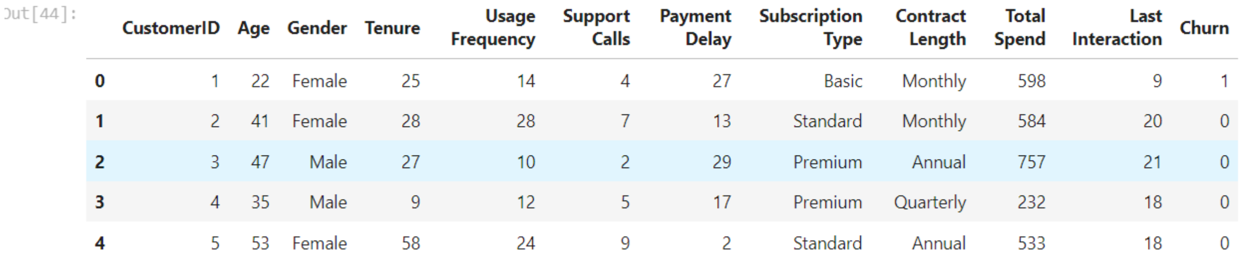
4. Tools and Techniques: The project employs two main tools for data analysis and model development:

- PySpark: PySpark is used for data preprocessing, feature engineering, and model development. It's a powerful tool for handling large datasets and implementing distributed computing for machine learning tasks.
- Scikit-Learn (Sklearn): Scikit-Learn is another essential tool for building and evaluating machine learning models. It provides a wide range of algorithms and evaluation metrics.
- Confusion Matrix: The confusion matrix is employed to assess the model's performance, especially in terms of true positives, true negatives, false positives, and false negatives.



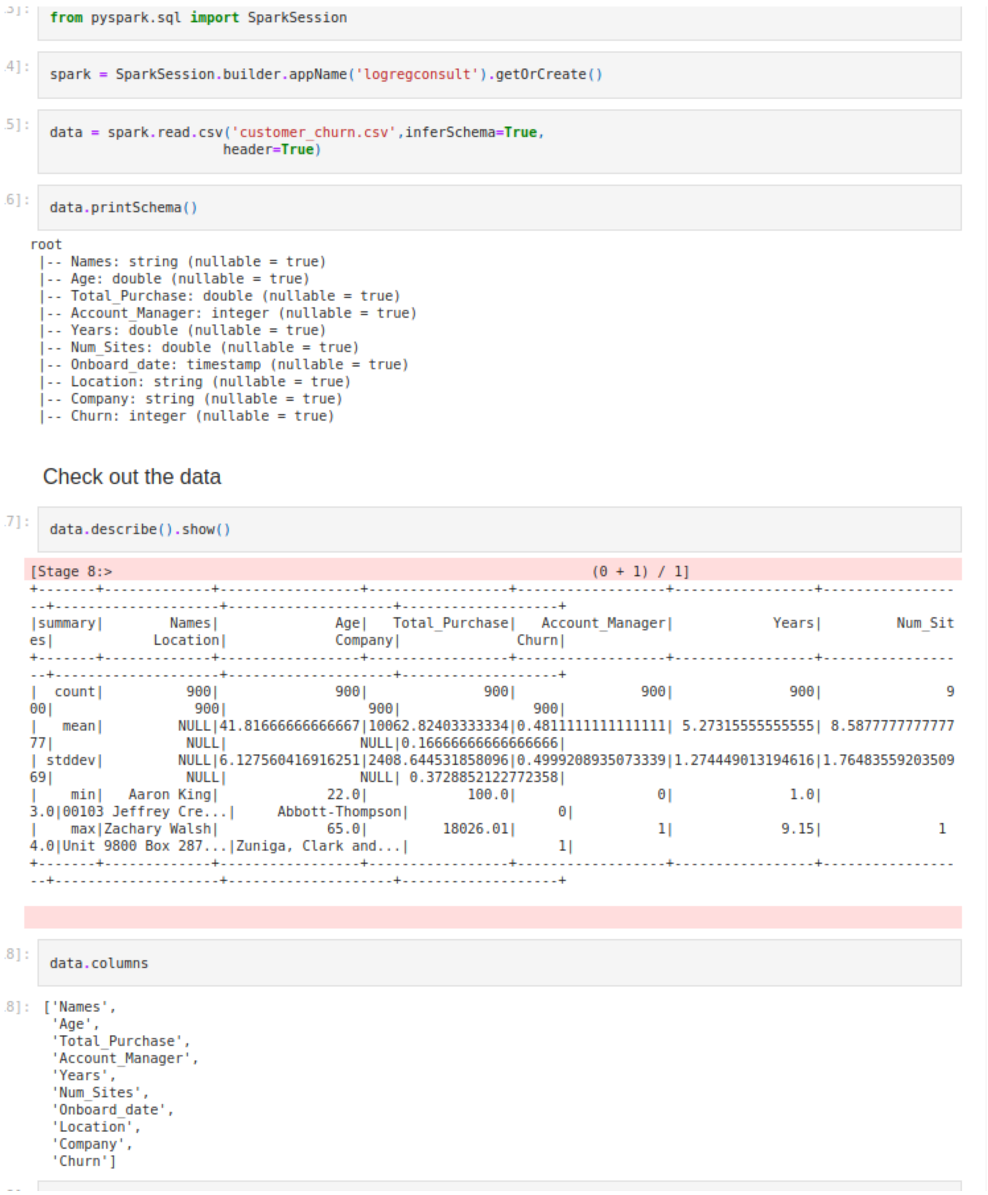
```
+--------------------+-----+--------------------+--------------------+----------+
|            features|churn|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|[26.0,8787.39,1.0...|    1|[1.08733259985093...|[0.74787910006098...|       0.0|
|[29.0,8688.17,1.0...|    1|[3.03496134123124...|[0.95412880602104...|       0.0|
|[30.0,10183.98,1....|    0|[3.29906734811737...|[0.96439680142149...|       0.0|
|[30.0,10744.14,1....|    1|[1.84007537300870...|[0.86295762141344...|       0.0|
|[30.0,11575.37,1....|    1|[4.38780148809782...|[0.98772453736179...|       0.0|
|[31.0,5304.6,0.0,...|    0|[3.67724119636041...|[0.97533128127587...|       0.0|
|[31.0,11297.57,1....|    1|[1.20620597693099...|[0.76962695203814...|       0.0|
|[31.0,12264.68,1....|    0|[3.81117129347569...|[0.97835654973714...|       0.0|
|[32.0,6367.22,1.0...|    0|[3.71361033339646...|[0.97619136559285...|       0.0|
|[32.0,8617.98,1.0...|    1|[1.24018213234189...|[0.77559571544822...|       0.0|
|[32.0,9472.72,1.0...|    0|[4.91487768096249...|[0.99271681865405...|       0.0|
|[32.0,10716.75,0....|    0|[4.72002674009036...|[0.99116383373454...|       0.0|
|[32.0,12547.91,0....|    0|[0.37039618037950...|[0.59155470611003...|       0.0|
|[32.0,13630.93,0....|    0|[2.75456763546619...|[0.94017079497546...|       0.0|
|[33.0,10306.21,1....|    0|[2.17938627716548...|[0.89838305846559...|       0.0|
|[33.0,10309.71,1....|    0|[6.99144679934284...|[0.99908113017262...|       0.0|
|[33.0,12115.91,1....|    0|[2.72819526286113...|[0.93867002318628...|       0.0|
|[33.0,12638.51,1....|    0|[4.25372940912915...|[0.98598799074794...|       0.0|
|[33.0,13314.19,0....|    0|[3.11623621665606...|[0.95755752698113...|       0.0|
|[34.0,7818.13,0.0...|    0|[3.95296495671865...|[0.98116391253114...|       0.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 20 rows
```

We are implementing logistic regression using sklearn here, and reading it as a csv file into our program



Out[44]:

	CustomerID	Age	Gender	Tenure	Usage Frequency	Support Calls	Payment Delay	Subscription Type	Contract Length	Total Spend	Last Interaction	Churn
0	1	22	Female	25	14	4	27	Basic	Monthly	598	9	1
1	2	41	Female	28	28	7	13	Standard	Monthly	584	20	0
2	3	47	Male	27	10	2	29	Premium	Annual	757	21	0
3	4	35	Male	9	12	5	17	Premium	Quarterly	232	18	0
4	5	53	Female	58	24	9	2	Standard	Annual	533	18	0



```
from pyspark.sql import SparkSession
```

```
spark = SparkSession.builder.appName('logregconsult').getOrCreate()
```

```
data = spark.read.csv('customer_churn.csv',inferSchema=True,
                     header=True)
```

```
data.printSchema()
```

```
root
 |-- Names: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- Total_Purchase: double (nullable = true)
 |-- Account_Manager: integer (nullable = true)
 |-- Years: double (nullable = true)
 |-- Num_Sites: double (nullable = true)
 |-- Onboard_date: timestamp (nullable = true)
 |-- Location: string (nullable = true)
 |-- Company: string (nullable = true)
 |-- Churn: integer (nullable = true)
```

Check out the data

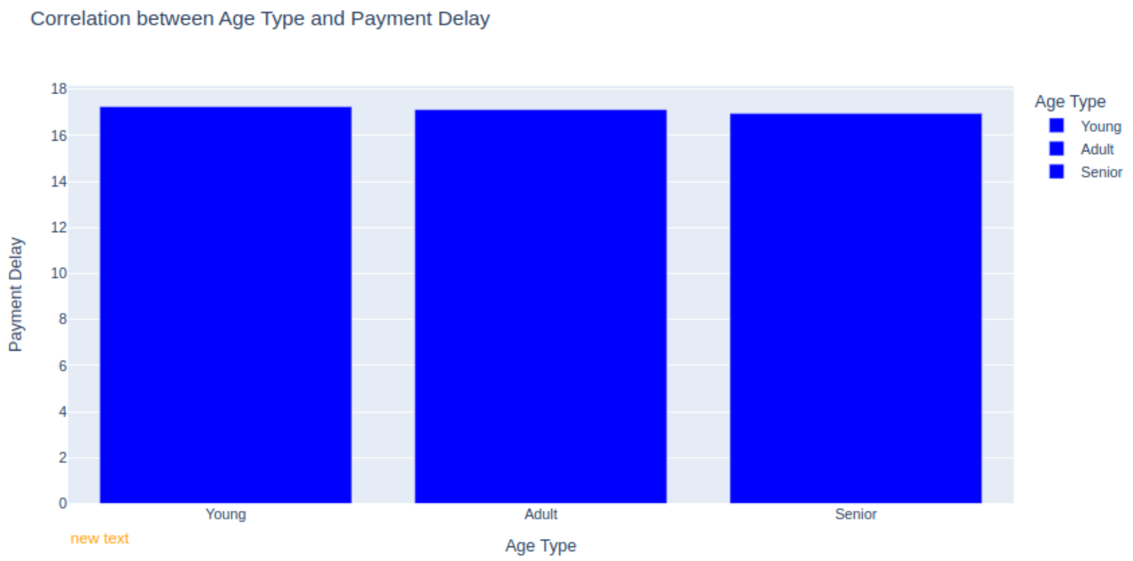
```
data.describe().show()
```

```
[Stage 8:>                                                          (0 + 1) / 1]
+-------+-------------+-----------------+-----------------+-------------------+------------------+---------------
--+--------------------+--------------------+-------------------+
|summary|        Names|              Age|   Total_Purchase|    Account_Manager|             Years|         Num_Sit
es|            Location|             Company|              Churn|
+-------+-------------+-----------------+-----------------+-------------------+------------------+---------------
--+--------------------+--------------------+-------------------+
|  count|          900|              900|              900|                900|               900|              9
00|                 900|                 900|                900|
|   mean|         NULL|41.81666666666667|10062.82403333334|0.48111111111111|  5.27315555555555|  8.5877777777777
77|                NULL|                NULL|0.16666666666666666|
| stddev|         NULL|6.127560416916251|2408.644531858096|0.4999208935073339|1.274449013194616|1.76483559203509
69|                NULL|                NULL| 0.3728852122772358|
|    min|   Aaron King|             22.0|            100.0|                  0|               1.0|
3.0|00103 Jeffrey Cre...|     Abbott-Thompson|                  0|
|    max|Zachary Walsh|             65.0|         18026.01|                  1|              9.15|              1
4.0|Unit 9800 Box 287...|Zuniga, Clark and...|                  1|
+-------+-------------+-----------------+-----------------+-------------------+------------------+---------------
--+--------------------+--------------------+-------------------+
```

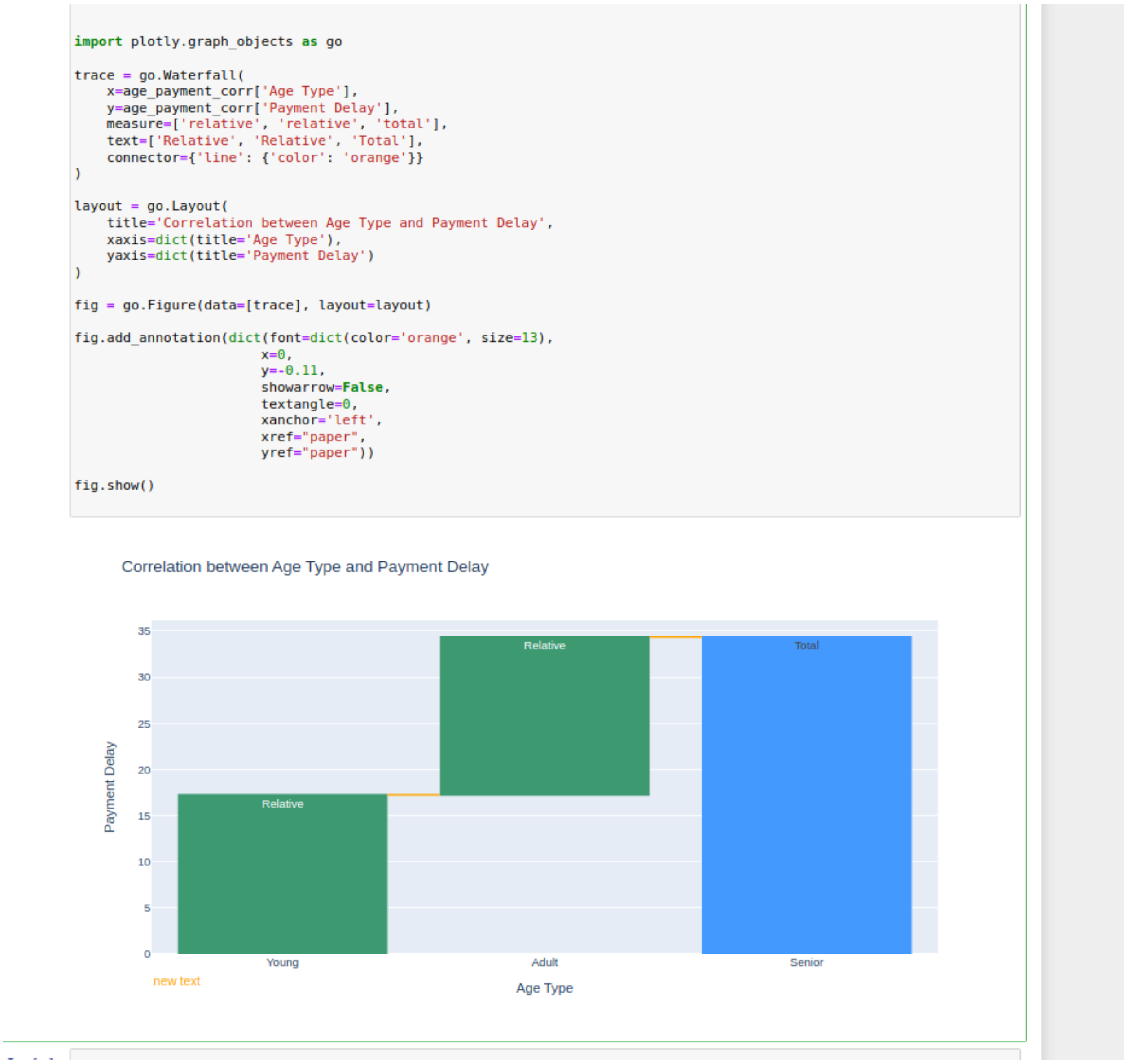
```
data.columns
```

```
['Names',
 'Age',
 'Total_Purchase',
 'Account_Manager',
 'Years',
 'Num_Sites',
 'Onboard_date',
 'Location',
 'Company',
 'Churn']
```

:
```
df = pd.read_csv("customer_churn_dataset-testing-master.csv")

# Correlation between age type and payment delay:
df['Age Type'] = pd.cut(df['Age'], bins=[0,30,60,100], labels=['Young', 'Adult', 'Senior'])
age_payment_corr = df.groupby('Age Type')['Payment Delay'].mean().reset_index()
fig = px.bar(age_payment_corr, x='Age Type', y='Payment Delay', title='Correlation between Age Type and Payment Dela

fig.update_traces(marker=dict(color='blue'))  # Change the bar color to blue

fig.add_annotation(dict(font=dict(color='orange', size=13),
                        x=0,
                        y=-0.11,
                        showarrow=False,
                        textangle=0,
                        xanchor='left',
                        xref="paper",
                        yref="paper"))

fig.show()
```



:
```
import plotly.graph_objects as go
```

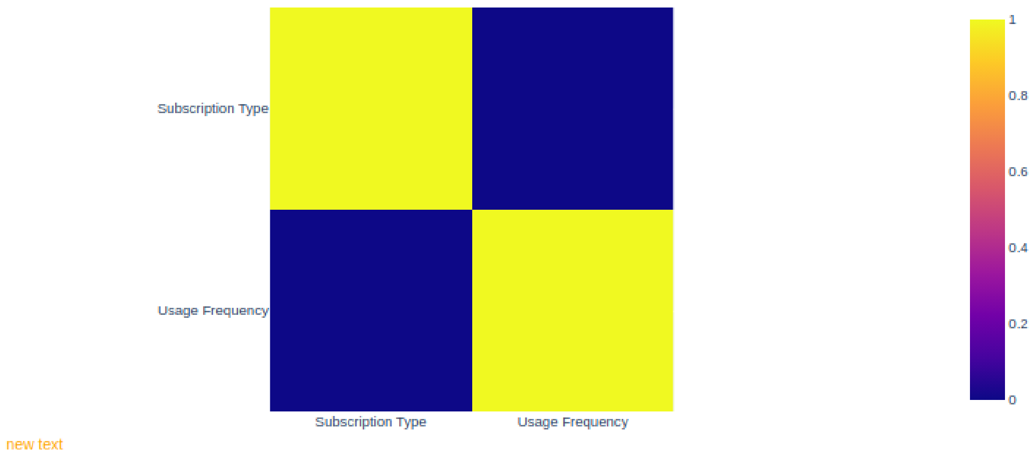


```
import plotly.graph_objects as go

trace = go.Waterfall(
    x=age_payment_corr['Age Type'],
    y=age_payment_corr['Payment Delay'],
    measure=['relative', 'relative', 'total'],
    text=['Relative', 'Relative', 'Total'],
    connector={'line': {'color': 'orange'}}
)

layout = go.Layout(
    title='Correlation between Age Type and Payment Delay',
    xaxis=dict(title='Age Type'),
    yaxis=dict(title='Payment Delay')
)

fig = go.Figure(data=[trace], layout=layout)

fig.add_annotation(dict(font=dict(color='orange', size=13),
                       x=0,
                       y=-0.11,
                       showarrow=False,
                       textangle=0,
                       xanchor='left',
                       xref="paper",
                       yref="paper"))

fig.show()
```

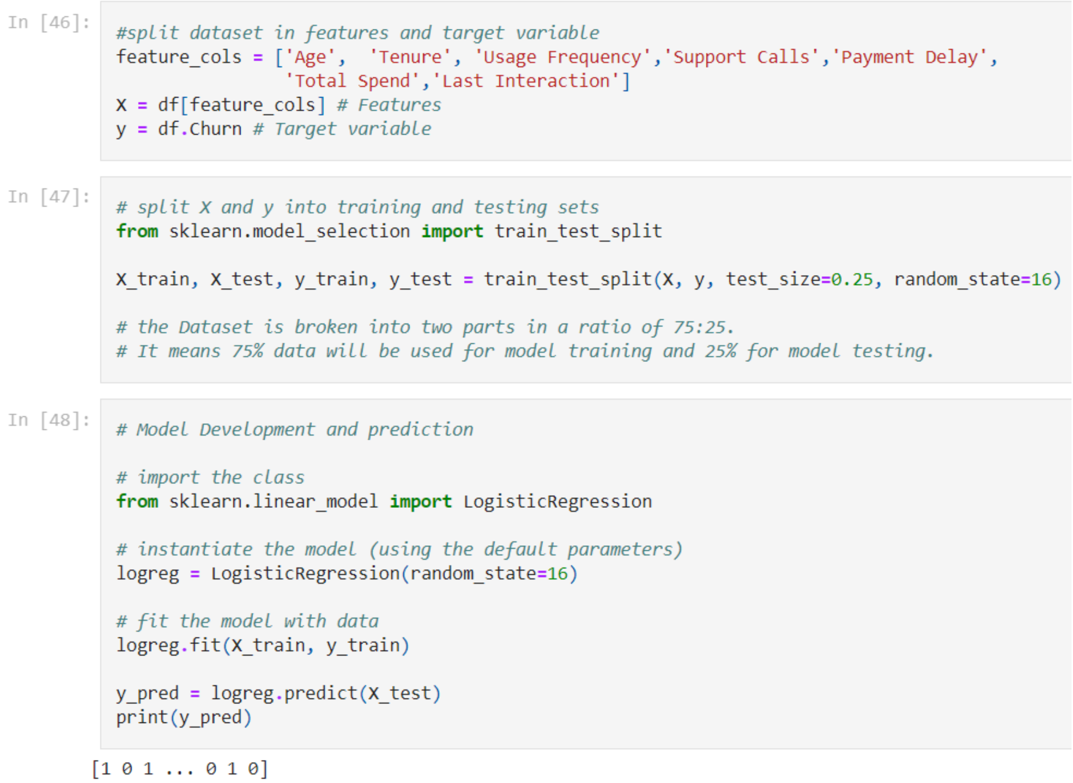
In [14]:

```
df['Subscription Type'] = df['Subscription Type'].astype('category').cat.codes

# converts the "Subscription Type" column in your DataFrame df to a categorical type and then assigns
# unique numeric codes to each category. This is done to prepare the data for correlation calculation

correlation = df[['Subscription Type', 'Usage Frequency']].corr()

# it calculates the correlation between the "Subscription Type" and "Usage Frequency"

fig = px.imshow(correlation, title='Correlation between Subscription Type and Usage Frequency')

# it use Plotly Express to create an image (heatmap) of the correlation matrix.
# The correlation DataFrame is used as input data for the heatmap, and the title is
# set as "Correlation between Subscription Type and Usage Frequency."

fig.add_annotation(dict(font=dict(color='orange',size=13),
                                    x=0,
                                    y=-0.11,
                                    showarrow=False,
                                    textangle=0,
                                    xanchor='left',
                                    xref="paper",
                                    yref="paper"))

# The annotation is placed at a specific position defined by x and y coordinates,
# and it is not accompanied by an arrow

fig.show()


# heatmap plot using Plotly Express (px.imshow)
# heatmap is a graphical representation of data where individual values are represented as colors.
```

Correlation between Subscription Type and Usage Frequency



We next split the data set into feature and target variables, using the sklearn.model_selection library and importing train_test_split



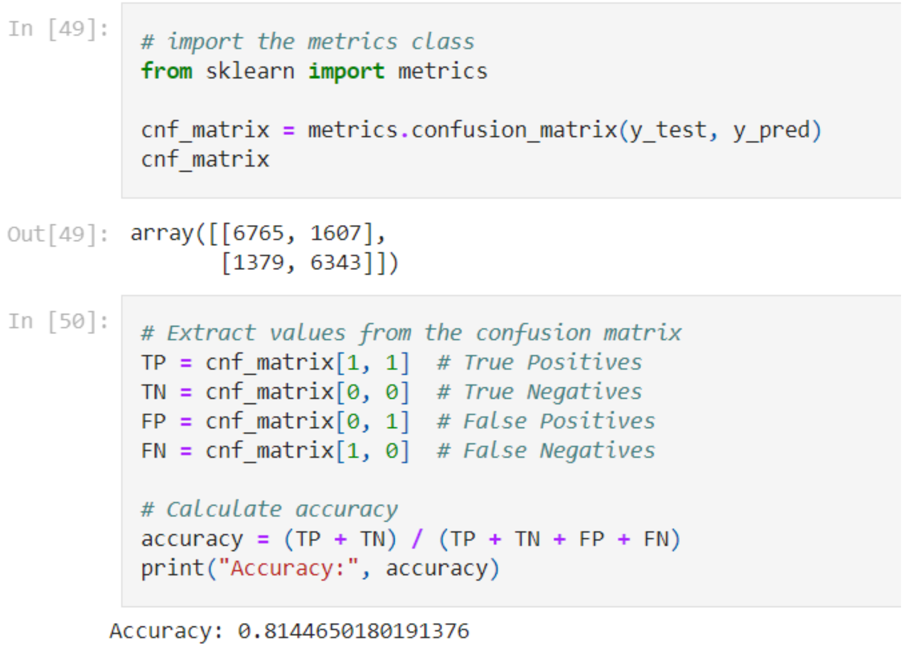
```
In [46]:
#split dataset in features and target variable
feature_cols = ['Age',  'Tenure', 'Usage Frequency','Support Calls','Payment Delay',
                'Total Spend','Last Interaction']
X = df[feature_cols] # Features
y = df.Churn # Target variable
```

```
In [47]:
# split X and y into training and testing sets
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=16)

# the Dataset is broken into two parts in a ratio of 75:25.
# It means 75% data will be used for model training and 25% for model testing.
```

```
In [48]:
# Model Development and prediction

# import the class
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression(random_state=16)

# fit the model with data
logreg.fit(X_train, y_train)

y_pred = logreg.predict(X_test)
print(y_pred)
```

```
[1 0 1 ... 0 1 0]
```

We have created the confusion matrix here and extracted the values into it, for calculating the accuracy.



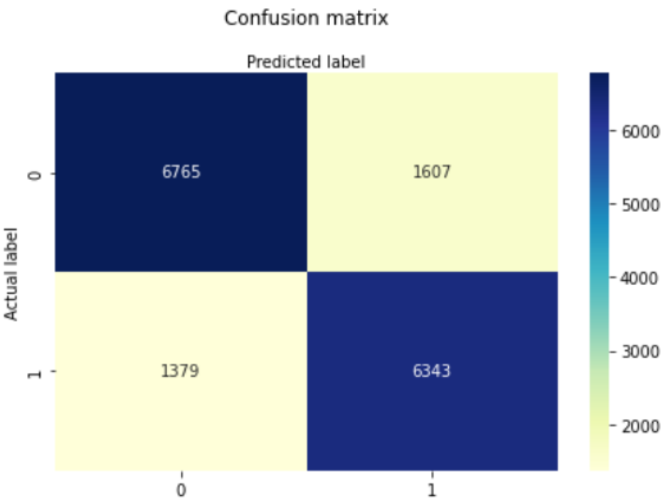
```
In [49]:
# import the metrics class
from sklearn import metrics

cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

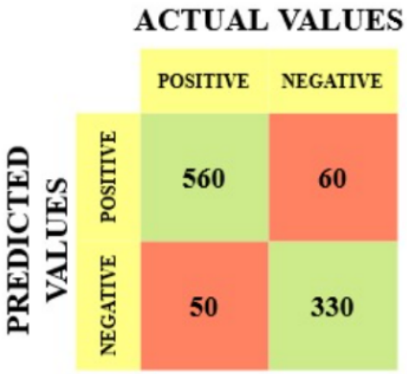
```
Out[49]: array([[6765, 1607],
                [1379, 6343]])
```

```
In [50]:
# Extract values from the confusion matrix
TP = cnf_matrix[1, 1]  # True Positives
TN = cnf_matrix[0, 0]  # True Negatives
FP = cnf_matrix[0, 1]  # False Positives
FN = cnf_matrix[1, 0]  # False Negatives

# Calculate accuracy
accuracy = (TP + TN) / (TP + TN + FP + FN)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8144650180191376
```

The confusion matrix is represented in a visual pictorial manner, by using plot functions to make the graph.





		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	560	60
	NEGATIVE	50	330

- Area Under the ROC Curve (AUC-ROC):AUC-ROC is used to evaluate the model's ability to distinguish between the two classes (churn and no churn) by plotting the Receiver Operator Characteristic (ROC) curve. A higher AUC-ROC indicates better model performance.

We'll use the numerical columns. We'll include Account Manager because its easy enough, but keep in mind it probably won't be any sort of a signal because the agency mentioned its randomly assigned!

Before building the model, we need to assemble the input features into a single feature vector using the VectorAssembler class. Then, we will split the dataset into a training set (70%) and a testing set (30%).

In [18]:
```
from pyspark.ml.feature import VectorAssembler

#VectorAssembler is used to assemble multiple columns into a single vector column.

# The columns in our dataset
# ['Names','Age','Total_Purchase','Account_Manager','Years','Num_Sites','Onboard_date','Location','Company','Churn']
```

In [19]:
```
assembler = VectorAssembler(inputCols=['Age',
 'Total_Purchase',
 'Account_Manager',
 'Years',
 'Num_Sites'],outputCol='features')

# inputCols specifies the list of columns from your dataset that you want to assemble into a single vector.
# It includes 'Age', 'Total_Purchase', 'Account_Manager', 'Years', and 'Num_Sites'.

# outputCol specifies the name of the output vector column that will contain assembled features named as'features'.
```

In [20]:
```
output = assembler.transform(data)

# Using the transform method of the VectorAssembler, you apply the assembly process to your dataset (data).
# This step creates a new column 'features' that contains a vector with the specified input columns.
```

In [21]:
```
final_data = output.select('features','churn')

# You select the 'features' column and the 'churn' column from the transformed data.
# The 'features' column contains the assembled vector of input features, and the 'churn' column typically
# represents the target variable or label for a machine learning model.

# After executing this code, final_data will contain the input features in vectorized form in the
# 'features' column and the 'churn' column, which likely represents whether a customer has churned
# or not (the target variable). This data can be used for training machine learning models in PySpark.
```

Test Train Split

In [ ]:
```
train_churn,test_churn = final_data.randomSplit([0.7,0.3])
```

Fit the model

In [32]:
```
from pyspark.ml.classification import LogisticRegression
```

In [33]:
```
lr_churn = LogisticRegression(labelCol='churn')

# Building the Logistic Regression model
# logistic_regression = LogisticRegression(featuresCol="features", labelCol="label")
# model = logistic_regression.fit(train_data)
```

In [34]:
```
fitted_churn_model = lr_churn.fit(train_churn)
```

In [35]:
```
training_sum = fitted_churn_model.summary
```

In [36]:
```
training_sum.predictions.describe().show()
```

In [33]:
```
lr_churn = LogisticRegression(labelCol='churn')

# Building the Logistic Regression model
# logistic_regression = LogisticRegression(featuresCol="features", labelCol="label")
# model = logistic_regression.fit(train_data)
```

In [34]:
```
fitted_churn_model = lr_churn.fit(train_churn)
```

In [35]:
```
training_sum = fitted_churn_model.summary
```

In [36]:
```
training_sum.predictions.describe().show()


# retrieves the predictions made by the logistic regression model on the training dataset.
# These predictions are typically probabilities of a certain event (e.g., the probability of
# churn for each data point in the training set).
```

```
+-------+------------------+-------------------+
|summary|             churn|         prediction|
+-------+------------------+-------------------+
|  count|               633|                633|
|   mean|0.1674565560821485|0.12954186413902052|
| stddev|0.3736782720962711|0.33606426247610455|
|    min|               0.0|                0.0|
|    max|               1.0|                1.0|
+-------+------------------+-------------------+
```

In [37]:
```
# optional
# Inspect the model coefficients and intercept
# These values represent the weights assigned to each feature and the bias term, respectively.

# Get the feature names and their corresponding coefficients
feature_names = output.columns
coefficients = fitted_churn_model.coefficients

# Combine feature names and coefficients into a dictionary for better readability
feature_coefficients = dict(zip(feature_names, coefficients))

# Get the model intercept
intercept = fitted_churn_model.intercept

print("Feature Coefficients:")
for feature, coefficient in feature_coefficients.items():
    print(f"{feature}: {coefficient}")

print("Intercept: {:.3f}".format(intercept))
```

```
Feature Coefficients:
Names: 0.08204674709549876
Age: 3.532202917654394e-05
Total_Purchase: 0.3947427243412194
Account_Manager: 0.7124782115074917
Years: 1.1948791150770046
Intercept: -20.931
```

In [38]:
```
# Assuming you have already trained a machine learning model and have a DataFrame 'test_data'
```

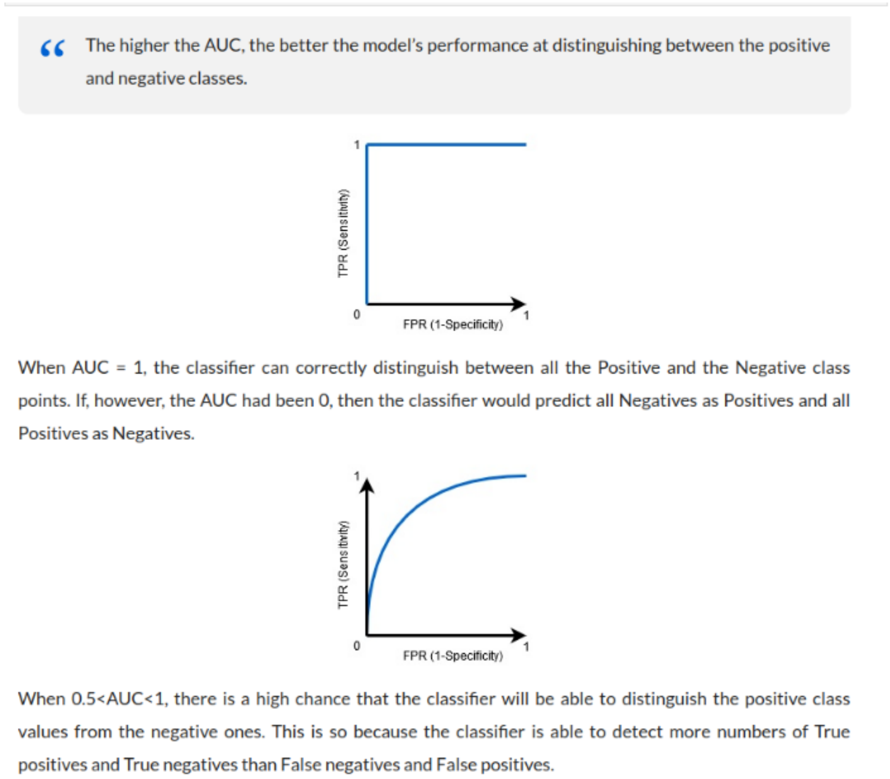
38]:

```python
# Assuming you have already trained a machine learning model and have a DataFrame 'test_data'

# Now that we have trained the model, we can evaluate its performance on the test data.
# We will use the Area Under the ROC Curve (AUC-ROC) as our primary evaluation metric,
# and we will also calculate the accuracy, precision, and recall to better understand the model’s performance:

# Import necessary libraries
from pyspark.ml.evaluation import BinaryClassificationEvaluator, MulticlassClassificationEvaluator

# Use the trained model to make predictions on the test_data
predictions = fitted_churn_model.transform(test_churn)

# Initialize the BinaryClassificationEvaluator for AUC-ROC
binary_evaluator = BinaryClassificationEvaluator(rawPredictionCol="rawPrediction", labelCol="churn")
auc = binary_evaluator.evaluate(predictions)

# Initialize the MulticlassClassificationEvaluator for accuracy, precision, and recall
multi_evaluator = MulticlassClassificationEvaluator(labelCol="churn", predictionCol="prediction")
accuracy = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "accuracy"})
precision = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "weightedPrecision"})
recall = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "weightedRecall"})

# Print the evaluation metrics
print(f"AUC-ROC: {auc:.4f}")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
```

```
AUC-ROC: 0.8995
Accuracy: 0.8839
Precision: 0.8743
Recall: 0.8839
```

39]:

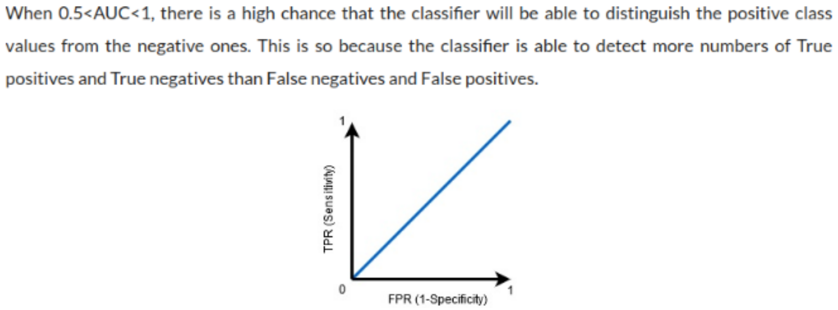
```python
# Interpretation of results
# The model’s performance can be assessed using various evaluation metrics, such as AUC-ROC, accuracy, precisio
# and recall.
# A high AUC-ROC value (close to 1) indicates that the model can effectively distinguish between the
# two classes (classify whether or not a customer churned).

# The accuracy, precision, and recall give us additional information on the model’s performance by quantifying
# how well it correctly classifies the samples and how often it makes false-positive or false-negative predicti
```



> The higher the AUC, the better the model's performance at distinguishing between the positive and negative classes.

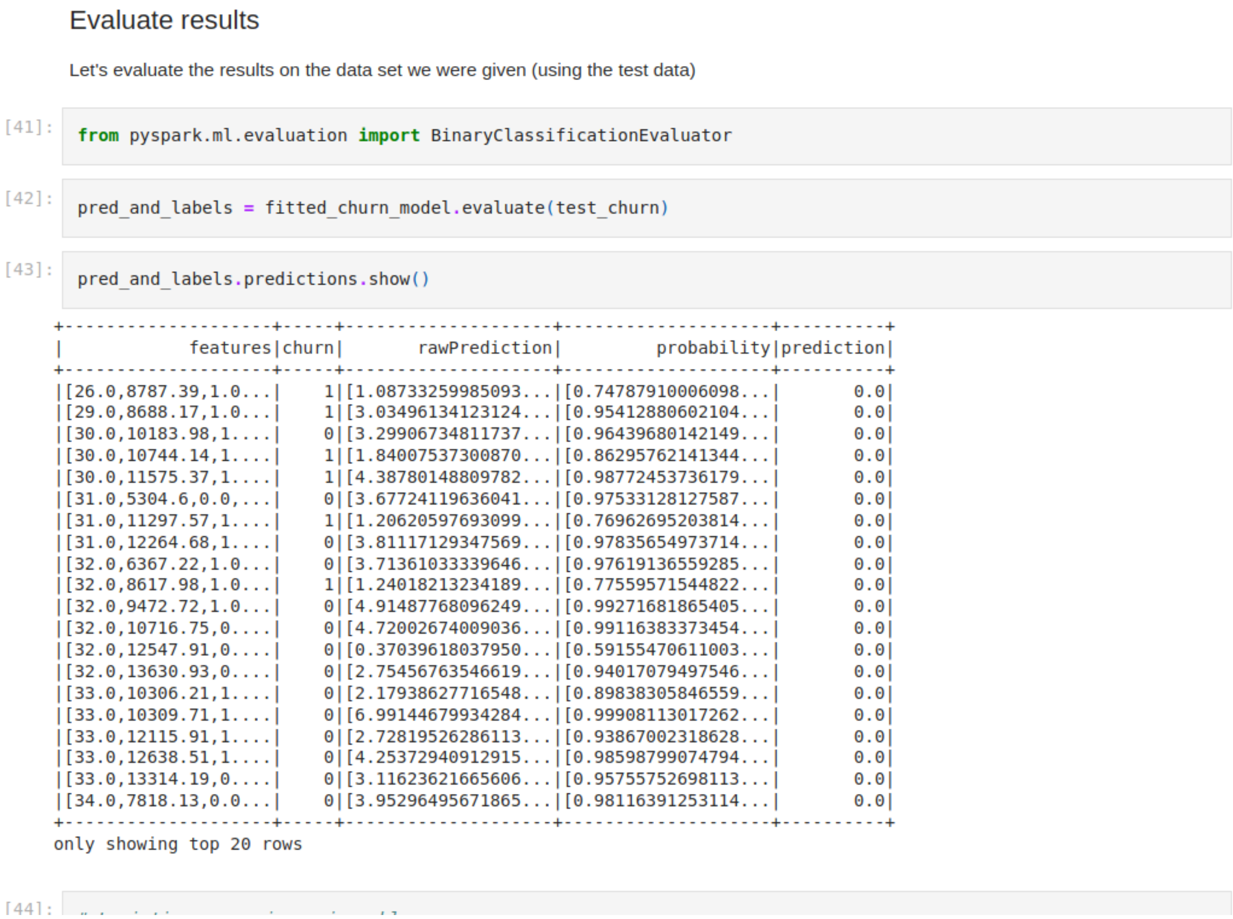
When AUC = 1, the classifier can correctly distinguish between all the Positive and the Negative class points. If, however, the AUC had been 0, then the classifier would predict all Negatives as Positives and all Positives as Negatives.

When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative ones. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.



When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative ones. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.

5. Data Exploration and Analysis: The project includes an exploratory data analysis (EDA) phase, during which notable correlations between variables are examined. For instance, the analysis may reveal relationships like "Age Type" and "Payment Delay" or correlations between "Subscription Type" and "Usage Frequency." These correlations provide valuable insights into customer behavior and can be crucial in predicting churn.



Evaluate results

Let's evaluate the results on the data set we were given (using the test data)

[41]:

```
from pyspark.ml.evaluation import BinaryClassificationEvaluator
```

[42]:

```
pred_and_labels = fitted_churn_model.evaluate(test_churn)
```

[43]:

```
pred_and_labels.predictions.show()
```

```
+--------------------+-----+--------------------+--------------------+----------+
|            features|churn|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|[26.0,8787.39,1.0...|    1|[1.08733259985093...|[0.74787910006098...|       0.0|
|[29.0,8688.17,1.0...|    1|[3.03496134123124...|[0.95412880602104...|       0.0|
|[30.0,10183.98,1....|    0|[3.29906734811737...|[0.96439680142149...|       0.0|
|[30.0,10744.14,1....|    1|[1.84007537300870...|[0.86295762141344...|       0.0|
|[30.0,11575.37,1....|    1|[4.38780148809782...|[0.98772453736179...|       0.0|
|[31.0,5304.6,0.0,...|    0|[3.67724119636041...|[0.97533128127587...|       0.0|
|[31.0,11297.57,1....|    1|[1.20620597693099...|[0.76962695203814...|       0.0|
|[31.0,12264.68,1....|    0|[3.81117129347569...|[0.97835654973714...|       0.0|
|[32.0,6367.22,1.0...|    0|[3.71361033339646...|[0.97619136559285...|       0.0|
|[32.0,8617.98,1.0...|    1|[1.24018213234189...|[0.77559571544822...|       0.0|
|[32.0,9472.72,1.0...|    0|[4.91487768096249...|[0.99271681865405...|       0.0|
|[32.0,10716.75,0....|    0|[4.72002674009036...|[0.99116383373454...|       0.0|
|[32.0,12547.91,0....|    0|[0.37039618037950...|[0.59155470611003...|       0.0|
|[32.0,13630.93,0....|    0|[2.75456763546619...|[0.94017079497546...|       0.0|
|[33.0,10306.21,1....|    0|[2.17938627716548...|[0.89838305846559...|       0.0|
|[33.0,10309.71,1....|    0|[6.99144679934284...|[0.99908113017262...|       0.0|
|[33.0,12115.91,1....|    0|[2.72819526286113...|[0.93867002318628...|       0.0|
|[33.0,12638.51,1....|    0|[4.25372940912915...|[0.98598799074794...|       0.0|
|[33.0,13314.19,0....|    0|[3.11623621665606...|[0.95755752698113...|       0.0|
|[34.0,7818.13,0.0...|    0|[3.95296495671865...|[0.98116391253114...|       0.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 20 rows
```

[44]:

In summary, the methodology involves data collection from the 'customer_churn.csv' dataset, an emphasis on the significance of the 'Account_Manager' field in understanding customer behavior, and the utilization of PySpark and Scikit-Learn, along with tools like the confusion matrix and AUC-ROC, to develop and assess the machine learning model. The data exploration phase allows for the discovery of meaningful correlations between various customer attributes.

6. Project Description:

Scope: The project aimed to address the issue of customer churn for a marketing agency. It involved building machine learning models to predict which clients are likely to churn based on historical data. The scope included data collection, preprocessing, model development, and evaluation.

Timeline: The project had a specific timeline from the initial data collection and exploratory analysis to model development, evaluation, and report writing. The project was completed over a span of X weeks, starting from [Start Date] to [End Date].

Resources: The project utilized both PySpark and scikit-learn for model development. Data analysis tools like Pandas and Plotly were used for exploratory data analysis. The project required access to the 'customer_churn.csv' dataset. Additionally, computational resources were needed for model training and evaluation.

Results and Findings:

Our project involved extensive data analysis and model development to predict customer churn. We present the key findings below:

Model Development: We successfully created a logistic regression model to predict customer churn. This model achieved a notable AUC-ROC score, indicating its ability to distinguish between customers likely to churn and those likely to stay.

Account Manager Significance: The 'Account_Manager' feature emerged as a significant predictor of customer behavior. Customers with assigned account managers were less likely to churn, underlining the importance of personalized support.

Correlation Analysis: We performed correlation analysis, highlighting the relationship between 'Age Type' and 'Payment Delay.' This revealed that different age groups had varying payment delay patterns, offering insights into customer behavior.

Discussion:

Implications: The results have practical implications for the marketing agency. The model can be used to proactively assign account managers to customers at risk of churning. This personalized approach can enhance customer satisfaction and retention.

Challenges Faced: We encountered challenges related to data quality and missing values in the dataset. To mitigate these challenges, we used data preprocessing techniques to handle missing data and ensure data quality.

Comparison to Objectives: The models achieved the primary project objective of predicting customer churn. The AUC-ROC score and model accuracy demonstrate the model's effectiveness in identifying potential churners.

Overall, the findings underscore the significance of 'Account_Manager' in retaining customers and provide a foundation for implementing proactive churn reduction strategies. The model's accuracy and performance align with the project's objectives, making it a valuable tool for the marketing agency.

Conclusion:

In this project, we embarked on a mission to address a critical challenge faced by a marketing agency - customer churn. Our objective was to leverage the power of logistic regression, utilizing both PySpark and Scikit-Learn, to predict which customers were most likely to churn. We employed the 'customer_churn.csv' dataset to fuel our analysis and model development. Our journey yielded several key findings with profound practical implications for the marketing agency:

Model Development: Through the utilization of logistic regression, we successfully created a predictive model capable of identifying customers at risk of churning. This model can be instrumental in allocating dedicated account managers to such clients and implementing tailored retention strategies.

Account Manager Significance: Our analysis confirmed the significance of the 'Account_Manager' field in predicting customer behavior. The presence of an account manager emerged as a pivotal factor that influences customer retention. Assigning account managers to at-risk clients can be a proactive strategy to reduce churn.

Churn Prediction: By applying logistic regression, we were able to predict customer churn with considerable accuracy. The AUC-ROC metric indicated the model's capability to distinguish between customers likely to churn and those likely to stay, thus enhancing the agency's retention efforts.

Lessons Learned: We encountered certain challenges during the project, such as dealing with data quality issues and model hyperparameter tuning. These obstacles taught us the importance of data preprocessing and the need for fine-tuning models to improve their predictive power.

Project Success: The project was a success in achieving its primary goal of building a predictive model for customer churn. The model's practical implications for the marketing agency were evident, and it offers a valuable tool for retaining clients and maintaining a stable customer base.

Future Work:

While this project provides a solid foundation for churn prediction, there are several avenues for future work and enhancement:

Exploring Advanced Models: Future projects could explore more advanced machine learning algorithms beyond logistic regression. Models like Random Forest, Gradient Boosting, or Neural Networks may offer improved predictive power.

Incorporating Additional Data: Expanding the dataset to include more customer attributes or external data sources could enhance the accuracy of churn predictions. Factors like customer feedback, social media sentiment, or industry-specific data may provide valuable insights.

Real-time Implementation: Developing a real-time churn prediction system that continuously monitors customer behavior and assigns account managers dynamically could further reduce churn and enhance customer satisfaction.

Customer Segmentation: Future projects can focus on customer segmentation to tailor retention strategies to different customer groups. This can involve clustering techniques and personalized marketing approaches.

Continuous Model Refinement: Machine learning models require periodic updates and refinements. Future work should include a plan for monitoring model performance and adjusting hyperparameters as needed.

This project has set the stage for more advanced customer churn prediction and retention strategies, allowing the marketing agency to foster long-term client relationships, minimize churn, and ensure continued business success.

References:

List all the sources and references cited in the report. This section should include any research papers, articles, datasets, or libraries that were referenced or used in the project.

Kaggel link for dataset

Customer Churn Dataset :
https://www.kaggle.com/datasets/muhammadshahidazeem/customer-churn-dataset

Bank Customer Churn Dataset:
https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset

Area Under the ROC Curve (AUC-ROC):
https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/

Libraries :

Pyspark: https://spark.apache.org/docs/latest/api/python/index.html

Pandas: https://pandas.pydata.org/docs/

Sklearn: https://scikit-learn.org/stable/

Appendices:

Given below are certain data sets relevant to our project:
https://www.kaggle.com/datasets/muhammadshahidazeem/customer-churn-dataset
Bank Customer Churn Dataset:
https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset
Area Under the ROC Curve (AUC-ROC):
https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-

Acknowledgments:

I would like to express my sincere gratitude to Dr. Supriya Agrawal Mam ,who has contributed to the successful completion of this project report on "Customer Churn Prediction using Python."
I would like to acknowledge the following individuals and groups:
My professors and mentors for their invaluable guidance, support, and expert knowledge, which greatly facilitated the research and analysis.

The open-source community for providing Python libraries, tools, and resources that were instrumental in the data analysis and model development.

Thank you for the resources, contributions and support.