Softwarica
College of IT & E-commerce

in collaboration with

Coventry University

STW7082CEM: Big Data Management and Data Visualization

Submitted To:
Siddhartha Neupane

Submitted by:
Sandeep Shrestha

Link to github: https://github.com/sandeepstha184/Big-data-management-and-data-visualization.git

Table of Contents

Project Proposal:

Project Title: **Binary Classification of Bank Marketing Data Using Pyspark.**

Module: STW7082CEM: Big Data Management and Data Visualization

Submitted By: Sandeep Shrestha

Softwarica College ID: 230127

Introduction

The data relates to a Portuguese financial institution's direct marketing initiatives. The Bank Marketing Dataset, which is being made accessible here, covers a broad spectrum of attributes and variables meant to provide insight into customer demographics, financial practices, and the effectiveness of marketing campaigns. The primary objective of this dataset is to provide analysis aimed at enhancing operational efficiency. Predicting whether a consumer will open a term deposit is the aim of classification.

Objective

The dataset is a valuable tool for understanding customer behavior and transaction dynamics, essential for banking sector decision-making, with the classification goal of predicting term deposit openings.

Dataset Name

bank.csvs

Data Sources

The data has been obtained from Kaggle where we can find numerous data sets.

Dataset Link

https://www.kaggle.com/code/palmer0/binary-classification-with-pyspark-and-mllib/notebook#Machine-Learning-with-PySpark-and-MLlib:-Solving-a-Binary-Classification-Problem

Dataset Description

The chosen "bank.csv" file has marketing campaign data in it. The dataset has the following fields: campaign, pdays, previpous, poutcome, deposit, age, job, education, balance, housing, loan, contact, day, month, duration, and campaign. The dataset includes the following datatypes: date, float, string, and integer. There are 11163 rows and 17 columns in all.

Introduction.

Big data has revolutionized the way organizations operate, make decisions, and formulate long-term plans in the current digital transformation age. In the financial sector, big data may be applied in several significant ways. Data-driven insights can significantly enhance marketing strategies, customer relationship management, and overall business performance. Big Data management and data visualization are key components of this change because they provide the techniques and tools needed to process massive amounts of data and identify important patterns and trends.

Every organization is seeking data to analyze and forecast the trends and further decisions for an organization. A dataset from a Portuguese financial institution's direct marketing campaigns is the main subject of this study, which explores the fields of big data management and data visualization. A wealth of customer-related information is included in the Bank Marketing Dataset, which is a diverse collection of unique characteristics. Examples of this include financial practices, consumer demographics, and the results of different marketing strategies. The organization hopes to improve its comprehension of consumer behavior and optimize its marketing efforts by organizing and displaying this data in an efficient manner.

This analysis aims to predict customer subscription to a term deposit by analyzing historical data and identifying key trends. Using advanced machine learning algorithms and binary classification along with data visualization tools, the institution aims to develop a robust model that accurately predicts customer behavior, improving marketing efforts.

Hence, this study highlights the transformational potential of data-driven decision-making in improving marketing outcomes and provides insight on the complex dynamics of customer interactions through a rigorous examination of the Bank Marketing Dataset.

Tools and Technology.

This study utilizes advanced tools like Apache spark and PySpark for efficient Big Data Management, while also utilizing visualization tools like Tableau and Python's Matplotlib libraries to create insightful data representations and communicate key findings.

Apache spark:

Big data processing and machine learning are the two main applications of Apache Spark, a potent open-source unified analytics platform. When comparing data processing tasks to conventional disk-based approaches, its in-memory calculation capabilities result in considerable speed increases. Large-scale data analysis may be easily handled with Spark because of its broad support for many operations such as SQL queries, streaming data, machine learning, and graph processing. Further enhancing its usefulness in a wide range of big data applications is its smooth integration with several data sources and language compatibility with Scala, Java, and Python.

PYSPARK

PySpark is the Python programming language interface for Apache Spark, which allows Python programmers to fully utilize Spark's large data processing capabilities. Data scientists and engineers that prefer Python over other programming languages may now develop Spark applications with PySpark because to its user-friendly syntax and features. All of Spark's essential features, such as data processing, streaming, machine learning, and SQL operations, are supported. Utilizing PySpark, users may leverage Spark's in-memory computing to obtain notable speed advantages when doing distributed data processing on huge datasets. PySpark, a powerful tool for big data analytics and machine learning, is enhanced by its integration with popular Python libraries like Pandas and NumPy, enabling seamless data manipulation and analysis.

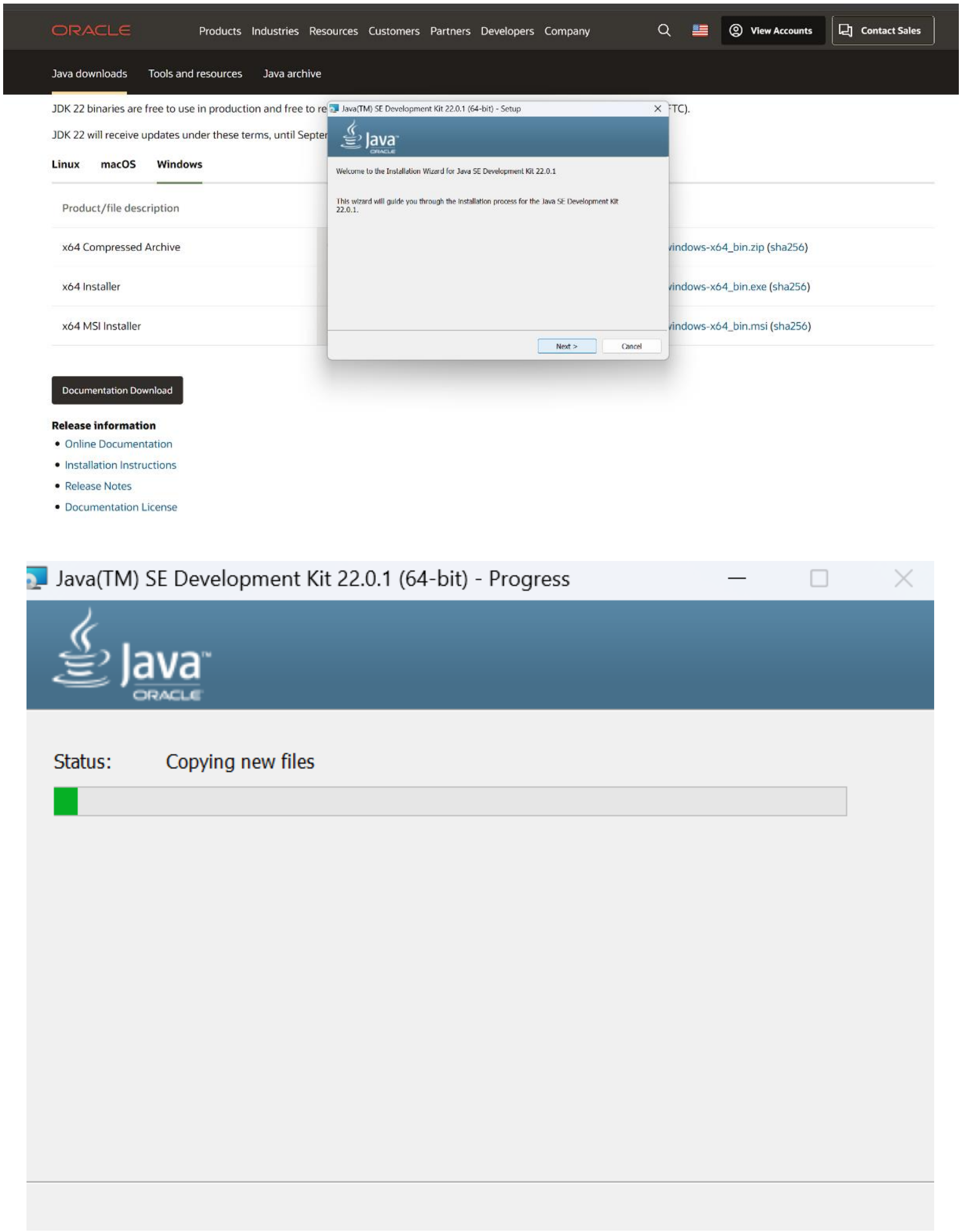
Tableau

Tableau is a powerful data visualization tool that enables users to transform unprocessed data into engaging visuals. Its user-friendly interface makes data analysis accessible to a wide range of users, including data scientists and business analysts. Tableau supports connections to various data sources, allowing users to generate dashboards and reports. Its visualization features allow for the creation of maps, graphs, and charts that reveal hidden patterns and trends. Tableau is particularly useful for organizations seeking data-driven decisions due to its robust sharing and collaboration capabilities, large data volumes, and user-friendly interface.

Installation process.

Here, I have described the method of installing the software in the windows operating system along with the screenshot.

The java file has been downloaded and installed.



JDK 22 binaries are free to use in production and free to re

JDK 22 will receive updates under these terms, until Septer

TC).

Product/file description	
x64 Compressed Archive	vindows-x64_bin.zip (sha256)
x64 Installer	vindows-x64_bin.exe (sha256)
x64 MSI Installer	vindows-x64_bin.msi (sha256)

Release information

- Online Documentation
- Installation Instructions
- Release Notes
- Documentation License

Java(TM) SE Development Kit 22.0.1 (64-bit) - Setup

Welcome to the Installation Wizard for Java SE Development Kit 22.0.1

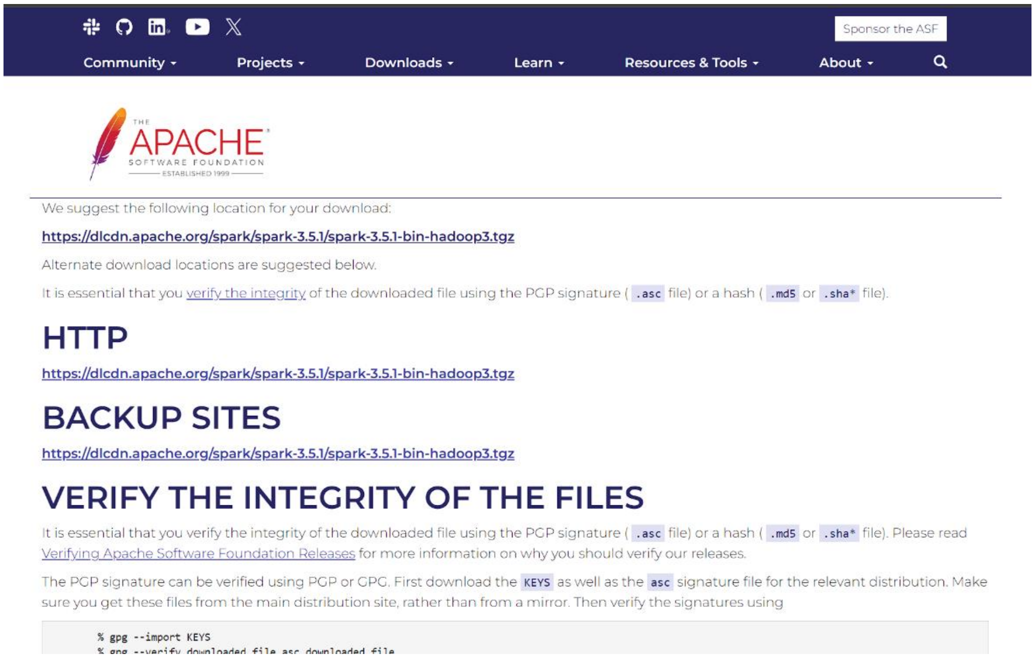
This wizard will guide you through the installation process for the Java SE Development Kit 22.0.1.

Java(TM) SE Development Kit 22.0.1 (64-bit) - Progress

Status: Copying new files

Figure 1: Java install.

The Apache spark has been downloaded from its site and the environmental setup which includes creating a new or editing environment is done and using the powershell it has been installed.



We suggest the following location for your download:

https://dlcdn.apache.org/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz

Alternate download locations are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (.asc file) or a hash (.md5 or .sha* file).

HTTP

https://dlcdn.apache.org/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz

BACKUP SITES

https://dlcdn.apache.org/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz

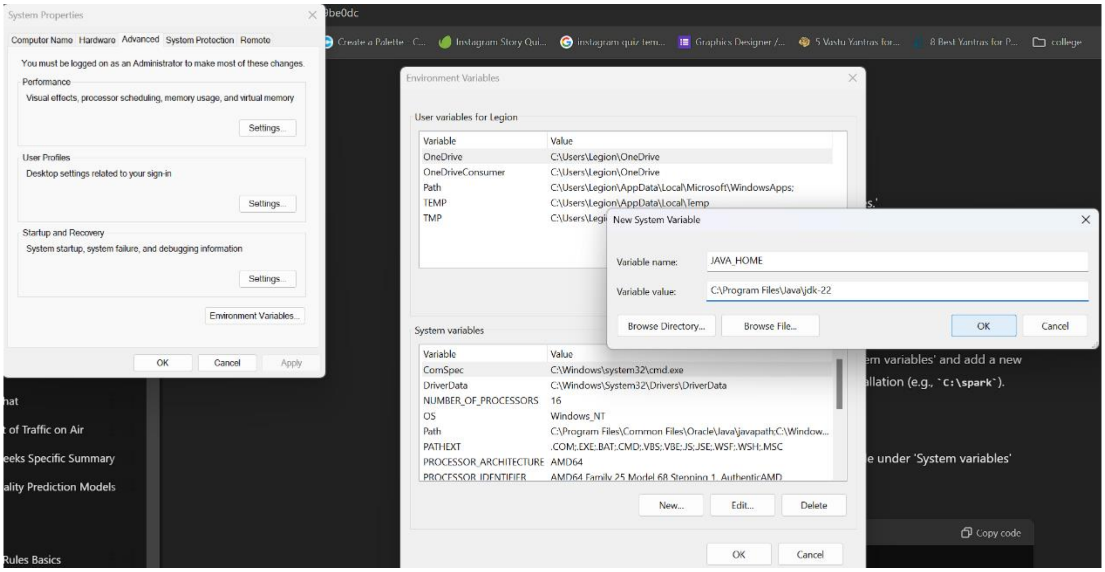
VERIFY THE INTEGRITY OF THE FILES

It is essential that you verify the integrity of the downloaded file using the PGP signature (.asc file) or a hash (.md5 or .sha* file). Please read Verifying Apache Software Foundation Releases for more information on why you should verify our releases.

The PGP signature can be verified using PGP or GPG. First download the KEYS as well as the asc signature file for the relevant distribution. Make sure you get these files from the main distribution site, rather than from a mirror. Then verify the signatures using

```
% gpg --import KEYS
% gpg --verify downloaded_file.asc downloaded_file
```

Figure 2: Downloading the Apache spark.



You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles

Desktop settings related to your sign-in

Startup and Recovery

System startup, system failure, and debugging information

User variables for Legion

Variable	Value
OneDrive	C:\Users\Legion\OneDrive
OneDriveConsumer	C:\Users\Legion\OneDrive
Path	C:\Users\Legion\AppData\Local\Microsoft\WindowsApps;
TEMP	C:\Users\Legion\AppData\Local\Temp
TMP	C:\Users\Legi

Variable name: JAVA_HOME

Variable value: C:\Program Files\Java\jdk-22

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	16
OS	Windows_NT
Path	C:\Program Files\Common Files\Oracle\Java\javapath;C:\Window...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	AMD64 Family 25 Model 68 Stepping 1, AuthenticAMD

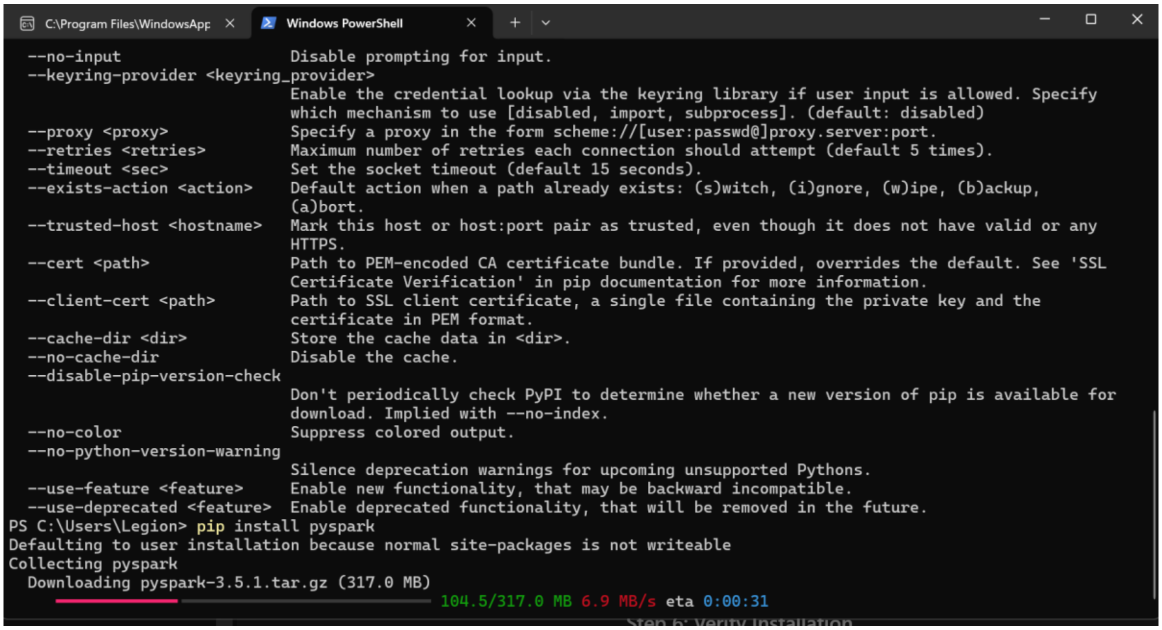
s.'

em variables' and add a new

llation (e.g., `C:\spark`).

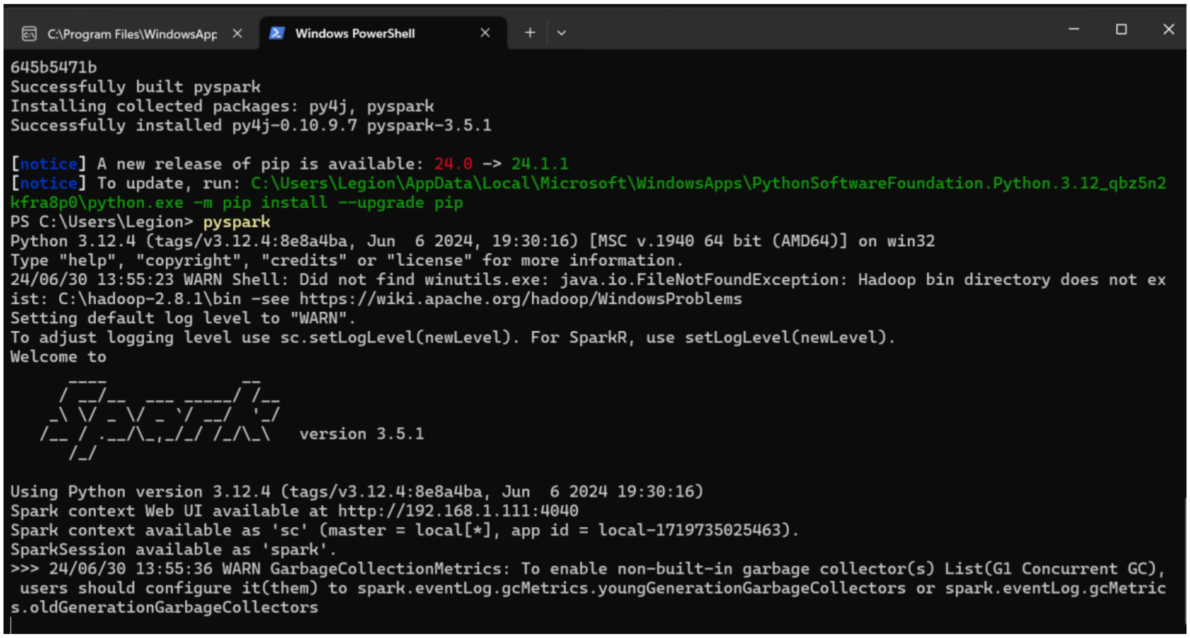
e under 'System variables'

Figure 3: environment setup.



```
  --no-input                  Disable prompting for input.
  --keyring-provider <keyring_provider>
                              Enable the credential lookup via the keyring library if user input is allowed. Specify
                              which mechanism to use [disabled, import, subprocess]. (default: disabled)
  --proxy <proxy>             Specify a proxy in the form scheme://[user:passwd@]proxy.server:port.
  --retries <retries>         Maximum number of retries each connection should attempt (default 5 times).
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup,
                              (a)bort.
  --trusted-host <hostname>   Mark this host or host:port pair as trusted, even though it does not have valid or any
                              HTTPS.
  --cert <path>               Path to PEM-encoded CA certificate bundle. If provided, overrides the default. See 'SSL
                              Certificate Verification' in pip documentation for more information.
  --client-cert <path>        Path to SSL client certificate, a single file containing the private key and the
                              certificate in PEM format.
  --cache-dir <dir>           Store the cache data in <dir>.
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI to determine whether a new version of pip is available for
                              download. Implied with --no-index.
  --no-color                  Suppress colored output.
  --no-python-version-warning
                              Silence deprecation warnings for upcoming unsupported Pythons.
  --use-feature <feature>     Enable new functionality, that may be backward incompatible.
  --use-deprecated <feature>  Enable deprecated functionality, that will be removed in the future.
PS C:\Users\Legion> pip install pyspark
Defaulting to user installation because normal site-packages is not writeable
Collecting pyspark
  Downloading pyspark-3.5.1.tar.gz (317.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 104.5/317.0 MB 6.9 MB/s eta 0:00:31
```

Figure 5: Downloading process.



```
645b5471b
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.7 pyspark-3.5.1

[notice] A new release of pip is available: 24.0 -> 24.1.1
[notice] To update, run: C:\Users\Legion\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\python.exe -m pip install --upgrade pip
PS C:\Users\Legion> pyspark
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun  6 2024, 19:30:16) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
24/06/30 13:55:23 WARN Shell: Did not find winutils.exe: java.io.FileNotFoundException: Hadoop bin directory does not ex
ist: C:\hadoop-2.8.1\bin -see https://wiki.apache.org/hadoop/WindowsProblems
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.1
      /_/

Using Python version 3.12.4 (tags/v3.12.4:8e8a4ba, Jun  6 2024 19:30:16)
Spark context Web UI available at http://192.168.1.111:4040
Spark context available as 'sc' (master = local[*], app id = local-1719735025463).
SparkSession available as 'spark'.
>>> 24/06/30 13:55:36 WARN GarbageCollectionMetrics: To enable non-built-in garbage collector(s) List(G1 Concurrent GC),
 users should configure it(them) to spark.eventLog.gcMetrics.youngGenerationGarbageCollectors or spark.eventLog.gcMetric
s.oldGenerationGarbageCollectors
```

Figure 4: Spark installation complete.

For this research, the Windows operating system is using Pyspark version 3.5.1. Pyspark generates a local host URL when it runs on a Windows operating system, which may be used to look at RDD activities, SQL operations, storage usage, and other things. It is also evident that Pyspark is operating on Python version 3.12.4.

Following by the application Tableau has also been installed. Registration is required to use tableau, after signing up with username, id, number, location the tableau desktop application has been downloaded and installed.

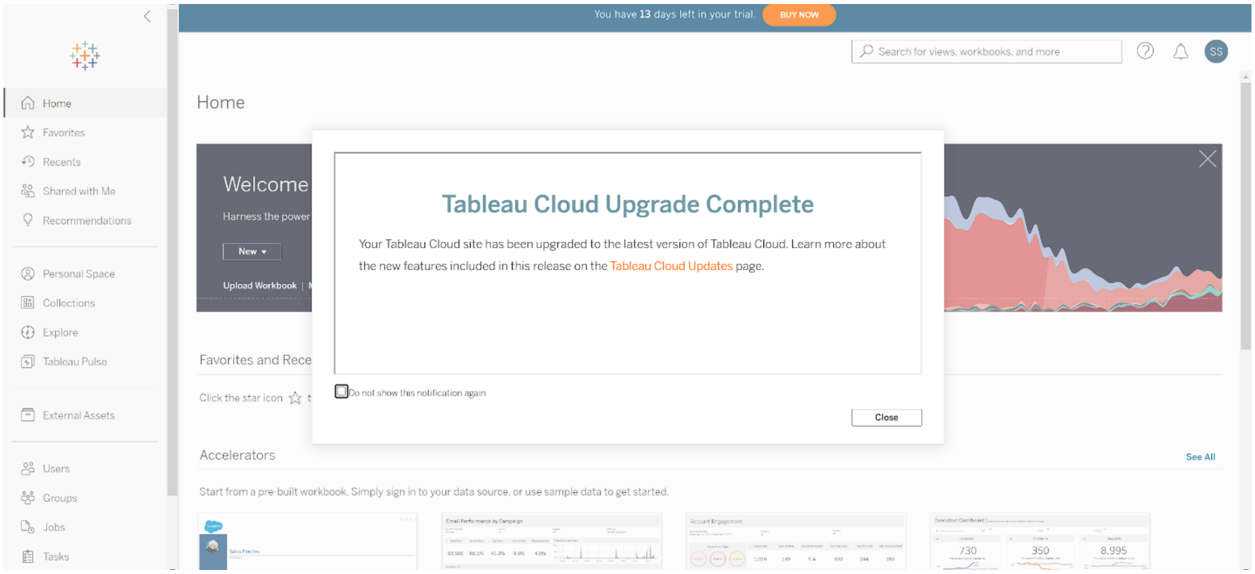


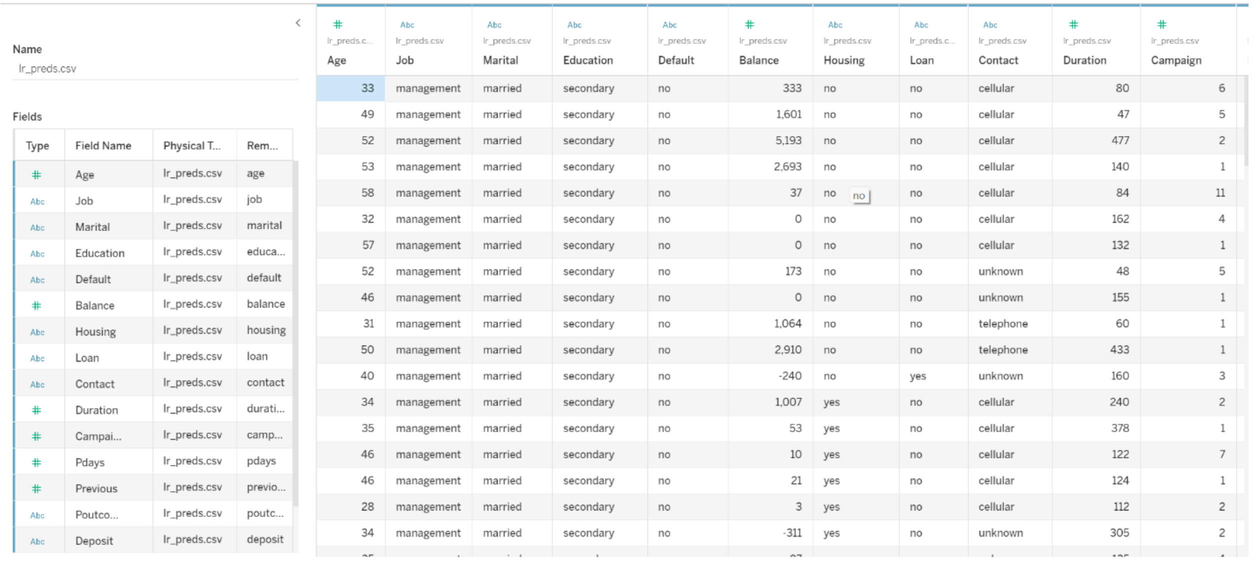
Tableau Cloud Upgrade Complete

Your Tableau Cloud site has been upgraded to the latest version of Tableau Cloud. Learn more about the new features included in this release on the Tableau Cloud Updates page.

□ Do not show this notification again

Close

Figure 7: Installation of Tableau.



Name
lr_preds.csv

Fields

Type	Field Name	Physical T...	Rem...
#	Age	lr_preds.csv	age
Abc	Job	lr_preds.csv	job
Abc	Marital	lr_preds.csv	marital
Abc	Education	lr_preds.csv	educa...
Abc	Default	lr_preds.csv	default
#	Balance	lr_preds.csv	balance
Abc	Housing	lr_preds.csv	housing
Abc	Loan	lr_preds.csv	loan
Abc	Contact	lr_preds.csv	contact
#	Duration	lr_preds.csv	durati...
#	Campai...	lr_preds.csv	camp...
#	Pdays	lr_preds.csv	pdays
#	Previous	lr_preds.csv	previo...
Abc	Poutco...	lr_preds.csv	poutc...
Abc	Deposit	lr_preds.csv	deposit

# lr_preds.c... Age	Abc lr_preds.csv Job	Abc lr_preds.csv Marital	Abc lr_preds.csv Education	Abc lr_preds.csv Default	# lr_preds.csv Balance	Abc lr_preds.csv Housing	Abc lr_preds.c... Loan	Abc lr_preds.csv Contact	# lr_preds.csv Duration	# lr_preds.csv Campaign
33	management	married	secondary	no	333	no	no	cellular	80	6
49	management	married	secondary	no	1,601	no	no	cellular	47	5
52	management	married	secondary	no	5,193	no	no	cellular	477	2
53	management	married	secondary	no	2,693	no	no	cellular	140	1
58	management	married	secondary	no	37	no no	no	cellular	84	11
32	management	married	secondary	no	0	no	no	cellular	162	4
57	management	married	secondary	no	0	no	no	cellular	132	1
52	management	married	secondary	no	173	no	no	unknown	48	5
46	management	married	secondary	no	0	no	no	unknown	155	1
31	management	married	secondary	no	1,064	no	no	telephone	60	1
50	management	married	secondary	no	2,910	no	no	telephone	433	1
40	management	married	secondary	no	-240	no	yes	unknown	160	3
34	management	married	secondary	no	1,007	yes	no	cellular	240	2
35	management	married	secondary	no	53	yes	no	cellular	378	1
46	management	married	secondary	no	10	yes	no	cellular	122	7
46	management	married	secondary	no	21	yes	no	cellular	124	1
28	management	married	secondary	no	3	yes	no	cellular	112	2
34	management	married	secondary	no	-311	yes	no	unknown	305	2

Figure 6: snapshot of dataset.

Following this, we then proceeded with the preliminary data study. The figure above provides the parts of our data set object.

DATASET, ANALYSIS AND IMPLEMENTATION

The dataset provides a useful example of using machine learning models on actual financial data. An indirect marketing effort by a Portuguese financial institution provided the dataset used in the Kaggle notebook "Binary Classification with PySpark and MLlib". It contains information about a range of consumer characteristics, including age, employment, marital status, degree of education, and the outcomes of previous marketing initiatives. Using binary classification algorithms from PySpark and MLlib, the main goal is to forecast, based on these attributes, if a client would subscribe to a term deposit. This dataset provides a realistic example of using machine learning models on actual financial data. The dataset contains integer, string, float, and date as datatypes. There are 17 total columns and 11163 rows. The dataset includes several key fields, each providing essential information:

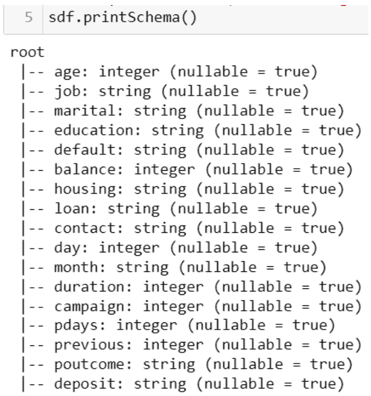
- **Age**: Represents the age of the customer. It is presented as an integer.
- **Job**: Specifies the customer's occupation, categorized into various types such as 'admin.', 'technician', 'services', etc. It is presented as a string.
- **Marital**: Indicates the marital status of the customer, such as 'married', 'single', or 'divorced'. It is presented as a string.
- **Education**: Represents the customer's level of education, categorized into 'primary', 'secondary', 'tertiary', etc. It is presented as a string.
- **Default**: Indicates whether the customer has credit in default. This field is presented as a string with possible values such as 'yes' or 'no'.
- **Balance**: Shows the average yearly balance of the customer in euros. It is presented as an integer.
- **Housing**: Specifies whether the customer has a housing loan. This field is presented as a string with possible values such as 'yes' or 'no'.
- **Loan**: Indicates whether the customer has a personal loan. This field is presented as a string with possible values such as 'yes' or 'no'.
- **Contact**: Type of communication contact used during the marketing campaign, such as 'cellular' or 'telephone'. It is presented as a string.
- **Day**: The last contact day of the month when the customer was contacted. It is presented as an integer.
- **Month**: The last contact month of the year when the customer was contacted, such as 'jan', 'feb', etc. It is presented as a string.
- **Duration**: Duration of the last contact with the customer, in seconds. It is presented as an integer.

- **Campaign**: Number of contacts performed during this marketing campaign with the customer. It is presented as an integer.
- **Pdays**: Number of days that passed since the customer was last contacted from a previous campaign. It is presented as an integer.
- **Previous**: Number of contacts performed before this campaign with the customer. It is presented as an integer.
- **Poutcome**: Outcome of the previous marketing campaign, categorized into 'success', 'failure', 'nonexistent'. It is presented as a string.
- **Deposit**: Indicates whether the customer subscribed to a term deposit. This is the target variable and is presented as a string with possible values such as 'yes' or 'no'.

To sum up, the dataset offers an extensive description of the financial behavior and demographics of the client base gathered via marketing initiatives by a Portuguese bank. For financial marketing analytics, the main objective is to forecast the likelihood that a consumer would sign up for a term deposit, allowing for exploratory data analysis and predictive modeling.

To better understand the variables affecting term deposit subscriptions, this dataset will be analyzed to identify patterns and trends in consumer behavior. This thorough research offers insightful information that will help the banking industry's future consumer targeting and marketing operations. To assure data integrity, I have used PySpark for preprocessing and data cleaning. For the best processing, PySpark's effectiveness with big datasets is utilized. Following that, Tableau does exploratory data analysis and provides a user-friendly interface for displaying trends and patterns in the dataset. To provide insights into potential future sales patterns, the last phase is applying linear regression modeling for predictive analysis using PySpark.

Investigate the data sets format and the kind of data in each field before starting the basic data analysis. Below, I have displayed the data schema.



```
sdf.printSchema()

root
 |-- age: integer (nullable = true)
 |-- job: string (nullable = true)
 |-- marital: string (nullable = true)
 |-- education: string (nullable = true)
 |-- default: string (nullable = true)
 |-- balance: integer (nullable = true)
 |-- housing: string (nullable = true)
 |-- loan: string (nullable = true)
 |-- contact: string (nullable = true)
 |-- day: integer (nullable = true)
 |-- month: string (nullable = true)
 |-- duration: integer (nullable = true)
 |-- campaign: integer (nullable = true)
 |-- pdays: integer (nullable = true)
 |-- previous: integer (nullable = true)
 |-- poutcome: string (nullable = true)
 |-- deposit: string (nullable = true)
```

Figure 8: Data schema.

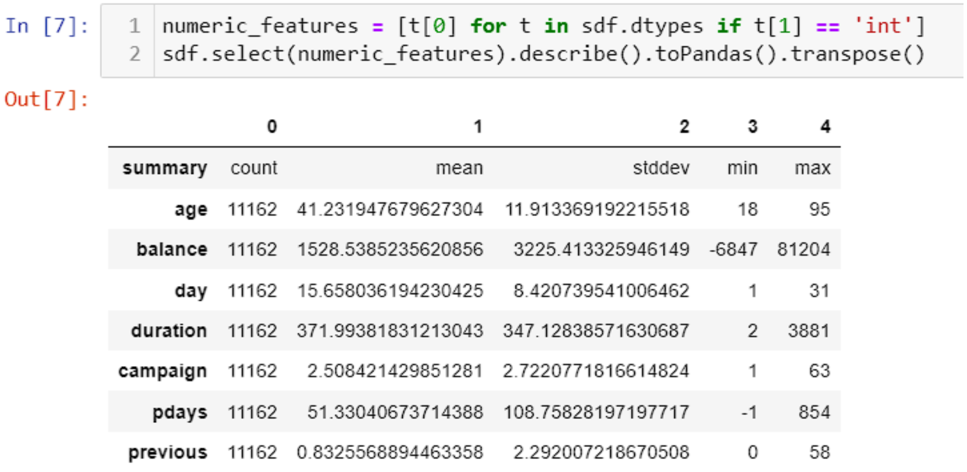
Data processing.

Data processing is the systematic approach of transforming raw data into meaningful information. It involves a series of steps that include data collection, cleaning, validation, transformation, and analysis. A crucial step in the data analysis process is data preparation, which entails cleaning and transforming raw data to improve its quality and suitability for further analysis.

This procedure deals with several issues, including inconsistent datasets, outliers, and missing values. Data cleansing is a crucial step that involves addressing missing or incorrect numbers and properly identifying and managing outliers. Standardizing or normalizing features, translating categorical variables into numerical representations, and resolving skewness are all aspects of data transformation.

Initially, raw data is gathered from various sources, which may include databases, sensors, or user inputs. This data often contains inconsistencies, duplicates, or errors, necessitating a cleaning process to ensure accuracy and quality. Validation checks are then performed to verify data integrity. The last objective is to ensure that the data complies with the specifications of the selected analytical methods and minimizes any potential biases before analyzing it. Accurate and significant discoveries in later phases of the data analysis process are based on effective data preparation.

Transforming cleansed data into a desired format or structure is known as data transformation, and it frequently entails normalization, aggregation, or other changes. After the data has been translated, it is examined using statistical techniques, machine learning algorithms, or other analytical tools to produce reports and extract insights. This processed data can assist a variety of applications in diverse sectors, corporate strategies, and decision-making.



```
In [7]: numeric_features = [t[0] for t in sdf.dtypes if t[1] == 'int']
        sdf.select(numeric_features).describe().toPandas().transpose()
```

Out[7]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
age	11162	41.231947679627304	11.913369192215518	18	95
balance	11162	1528.5385235620856	3225.413325946149	-6847	81204
day	11162	15.658036194230425	8.420739541006462	1	31
duration	11162	371.99381831213043	347.12838571630687	2	3881
campaign	11162	2.508421429851281	2.7220771816614824	1	63
pdays	11162	51.33040673714388	108.75828197197717	-1	854
previous	11162	0.8325568894463358	2.292007218670508	0	58

Figure 9: Summary statistics for numeric variables.

The figure above is a statistical summary of the numeric features in the Data Frame was generated, including age, balance, day, duration, campaign, pdays, and previous. The summary provides key statistics such as count, mean, standard deviation, minimum, and maximum values.

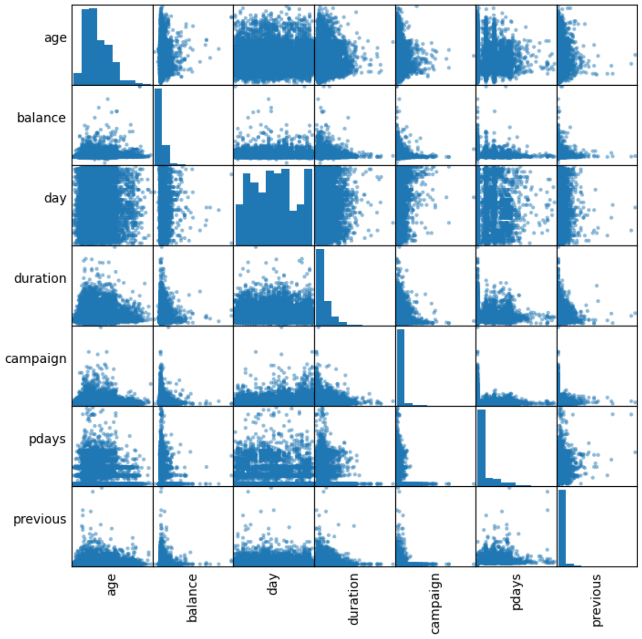
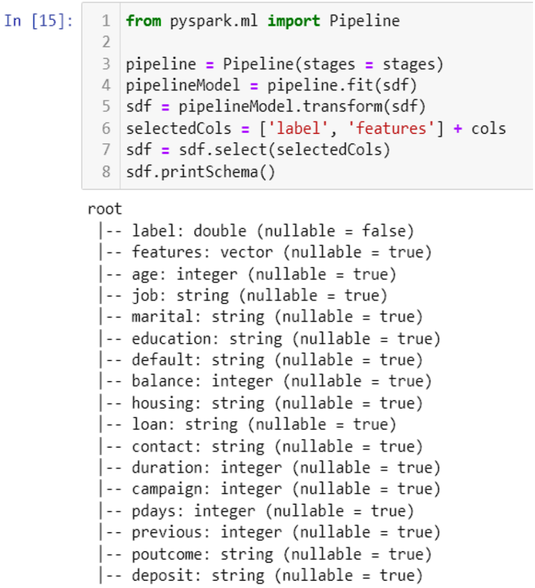


Figure 10: Correlations between independent variables.

In the figure above, scatter matrix plot has been created for the numeric features (age, balance, day, duration, campaign, pdays, and previous) of the DataFrame. This plot visualizes the pairwise relationships and distributions of these features, providing insights into potential correlations and data patterns.



```
In [15]:
from pyspark.ml import Pipeline

pipeline = Pipeline(stages = stages)
pipelineModel = pipeline.fit(sdf)
sdf = pipelineModel.transform(sdf)
selectedCols = ['label', 'features'] + cols
sdf = sdf.select(selectedCols)
sdf.printSchema()
```

```
root
 |-- label: double (nullable = false)
 |-- features: vector (nullable = true)
 |-- age: integer (nullable = true)
 |-- job: string (nullable = true)
 |-- marital: string (nullable = true)
 |-- education: string (nullable = true)
 |-- default: string (nullable = true)
 |-- balance: integer (nullable = true)
 |-- housing: string (nullable = true)
 |-- loan: string (nullable = true)
 |-- contact: string (nullable = true)
 |-- duration: integer (nullable = true)
 |-- campaign: integer (nullable = true)
 |-- pdays: integer (nullable = true)
 |-- previous: integer (nullable = true)
 |-- poutcome: string (nullable = true)
 |-- deposit: string (nullable = true)
```

Figure 11: features column and label column.

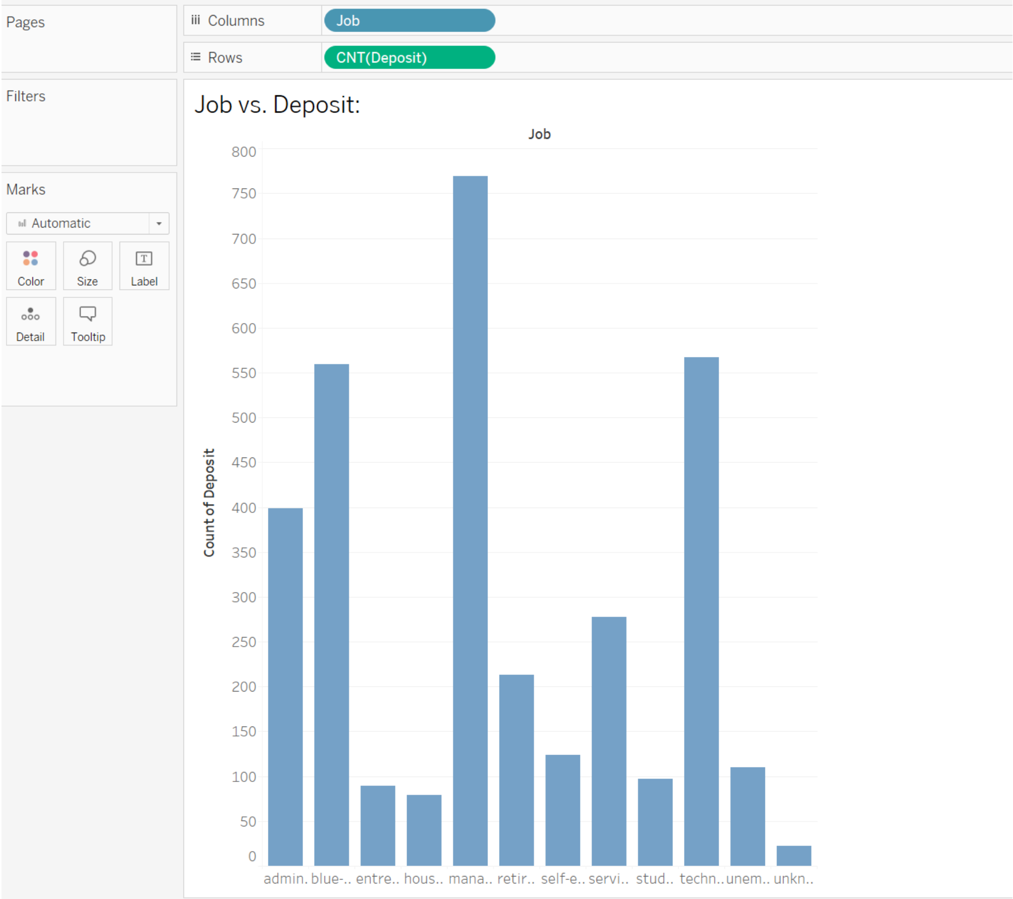
Pipeline has been used to chain multiple Transformers and Estimators together to specify the Machine Learning workflow. A Pipeline's stages are specified as an ordered array. The pipeline stages included StringIndexer and OneHotEncoder for encoding categorical columns (job, marital, education, etc.), and VectorAssembler to combine feature columns into a single vector. The label column deposit was indexed for binary classification. The pipeline was fitted and applied to the DataFrame, resulting in a transformed schema with label and features columns added alongside the original columns, ready for machine learning model training.

EXPLORATORY DATA ANALYSIS AND VISUALIZATION.

Exploratory Data Analysis (EDA) aims to identify patterns, trends, and correlations in raw data using Tableau data visualization. EDA's discovery-oriented design simplifies feature engineering by identifying essential structure, patterns, and relationships. It also helps avoid common statistical issues by examining specific assumptions required by most statistical approaches before being used. This intuitive platform allows analysts to examine data in real-time, creating engaging and intelligent representations.

Tableau converts complex datasets into visual representations, enhancing understanding of patterns. Users can dynamically engage with visual representations, allowing for deeper exploration of data points. This interactive investigation enhances EDA effectiveness, enabling quick identification of patterns, outliers, and correlations, enhancing its effectiveness.

Tableau is a powerful tool for exploratory data analysis (EDA) due to its visually appealing data representations, interactive features, and user-friendly interface. It allows analysts to quickly extract valuable insights, making complex information more accessible and intuitive. Tableau has been used for the visualization for the data set.



Columns: Job

Rows: CNT(Deposit)

Figure 12: Job vs. deposit.

The graph displayed above shows the association between employment types and deposit count is represented visually by the bar chart. According to the statistics, those who work in positions like "management" and "blue-collar" have more deposits than people who work in positions like "entrepreneur" and "self-employed." The study offers significant information for targeted marketing initiatives by identifying the job types that have a higher likelihood of depositing money.

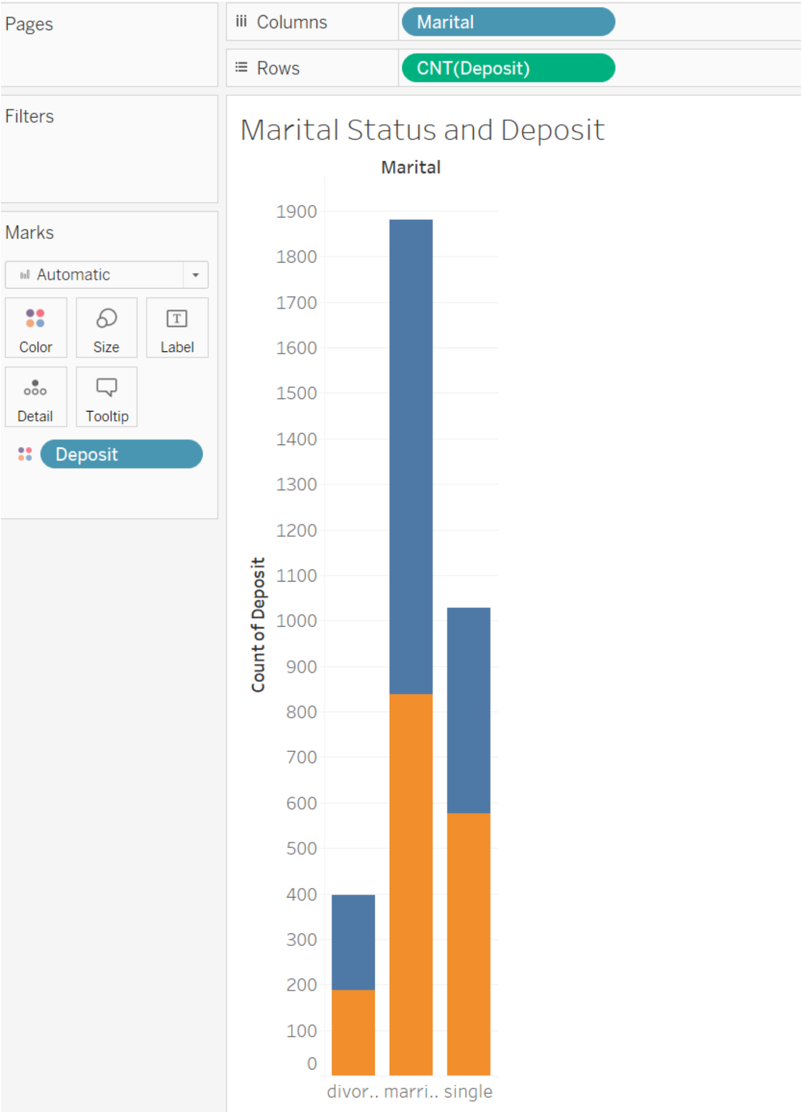
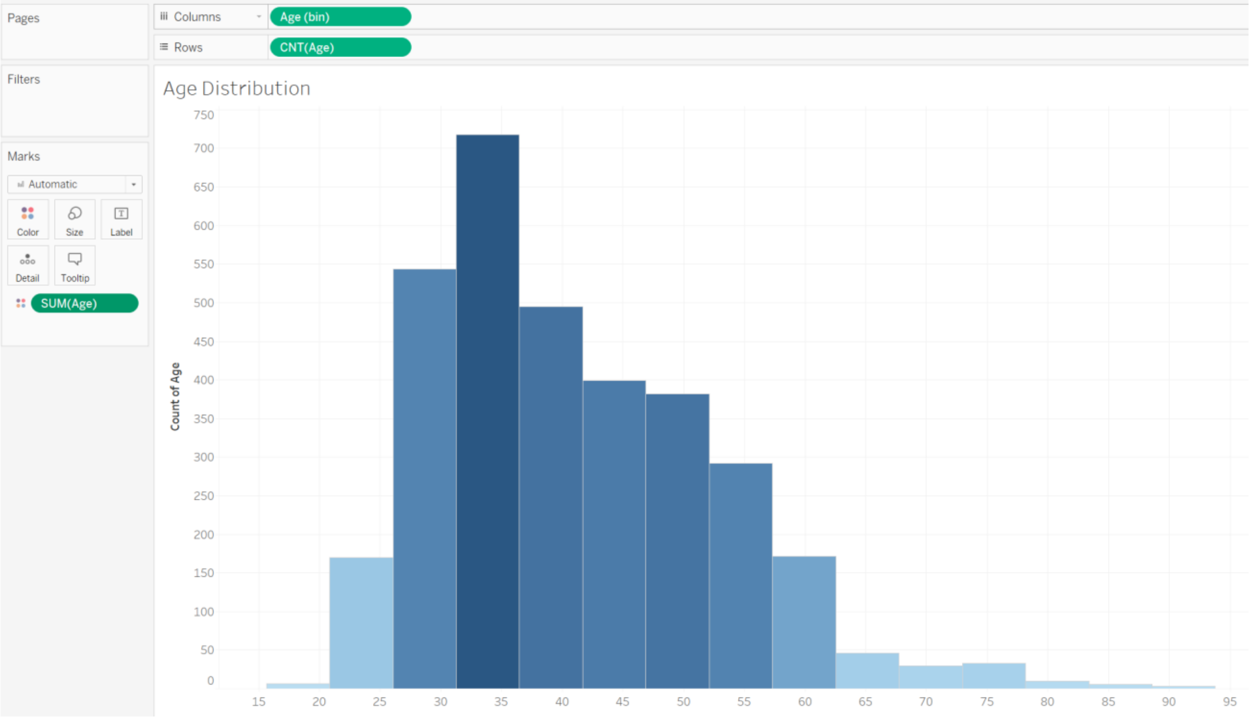


Figure 13: Marital Status and Deposit.

The bar chart shows that married individuals have the highest deposit count, followed by singles, and divorced individuals have the lowest count. This visualization helps understand the impact of marital status on the likelihood of making a deposit.



Columns Age (bin)

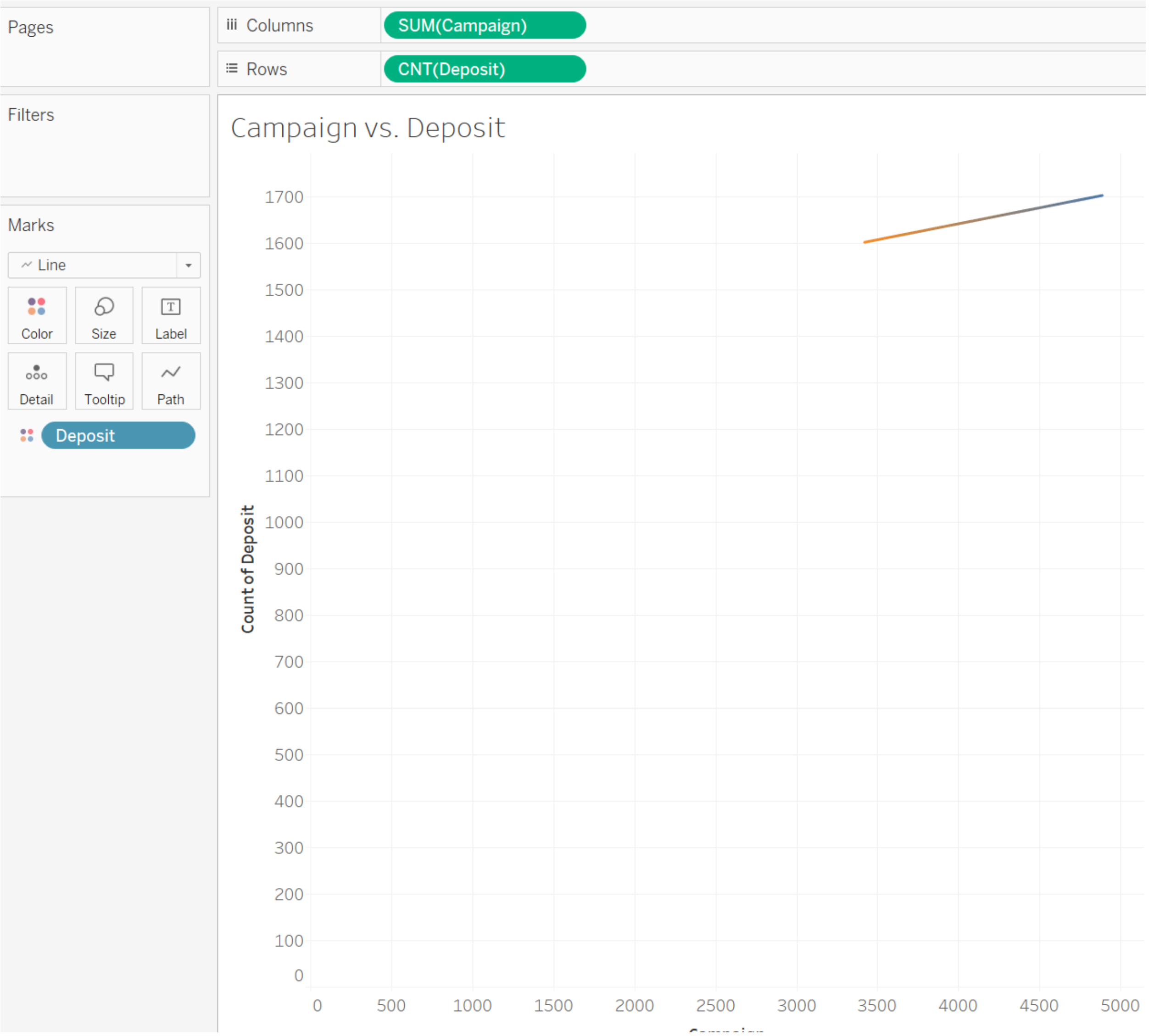
Rows CNT(Age)

Marks Automatic

SUM(Age)

Figure 14: Age distribution.

The histogram shows that the majority of individuals in the dataset are between 25-45 years old, with the highest count in the 30-35 age group, indicating a high concentration of this age group. This understanding can be useful for tailoring services or marketing efforts to these age groups.



Columns SUM(Campaign)

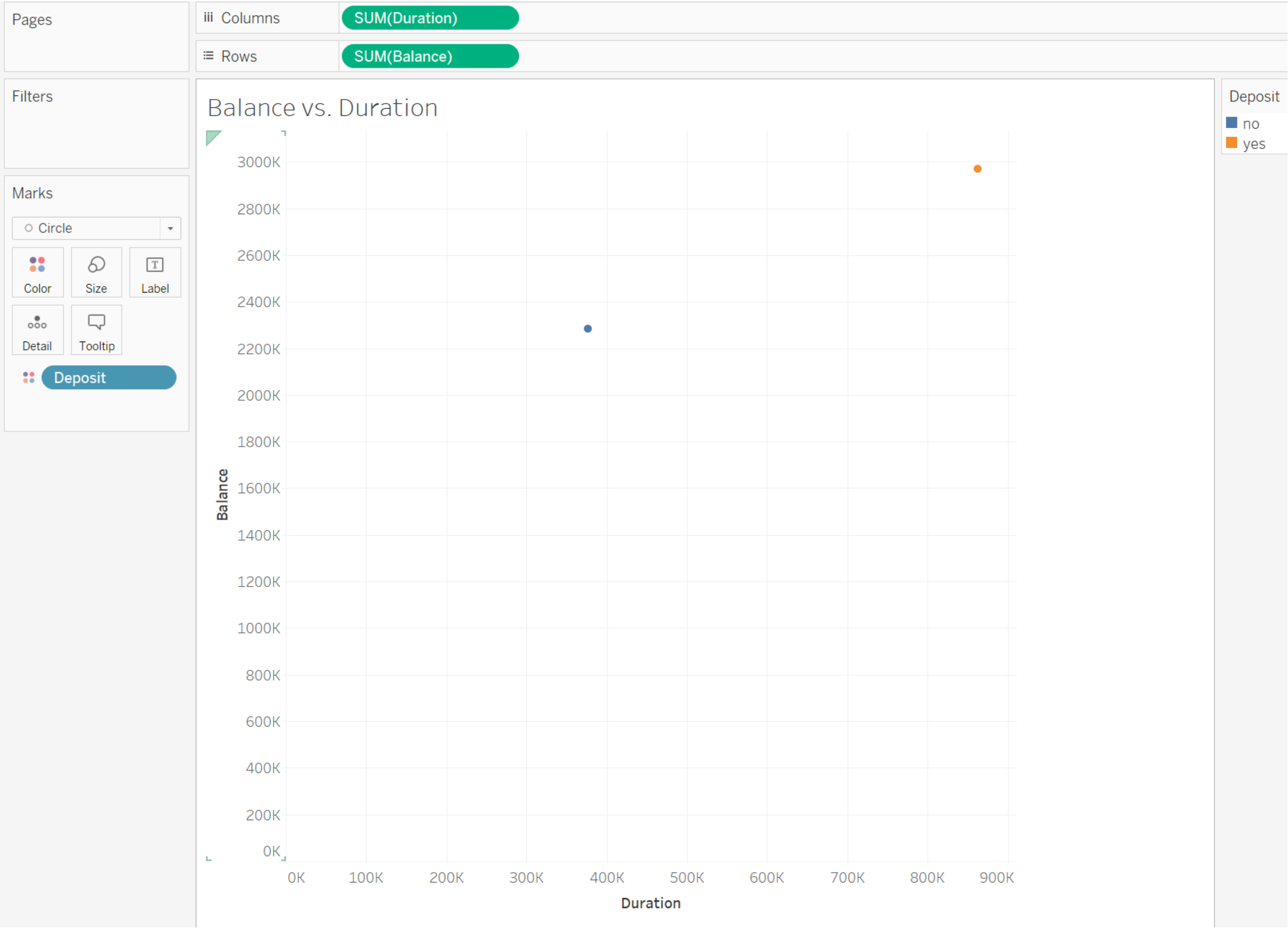
Rows CNT(Deposit)

Marks: Line

Color: Deposit

Figure 15: campaign vs deposit.

The line chart depicts the relationship between the total number of campaigns and the count of deposits. The chart indicates a positive trend, suggesting that as the number of campaigns increases, the count of deposits also rises. Therefore, this relationship highlights the effectiveness of campaign efforts in driving deposit activities.



Columns SUM(Duration)

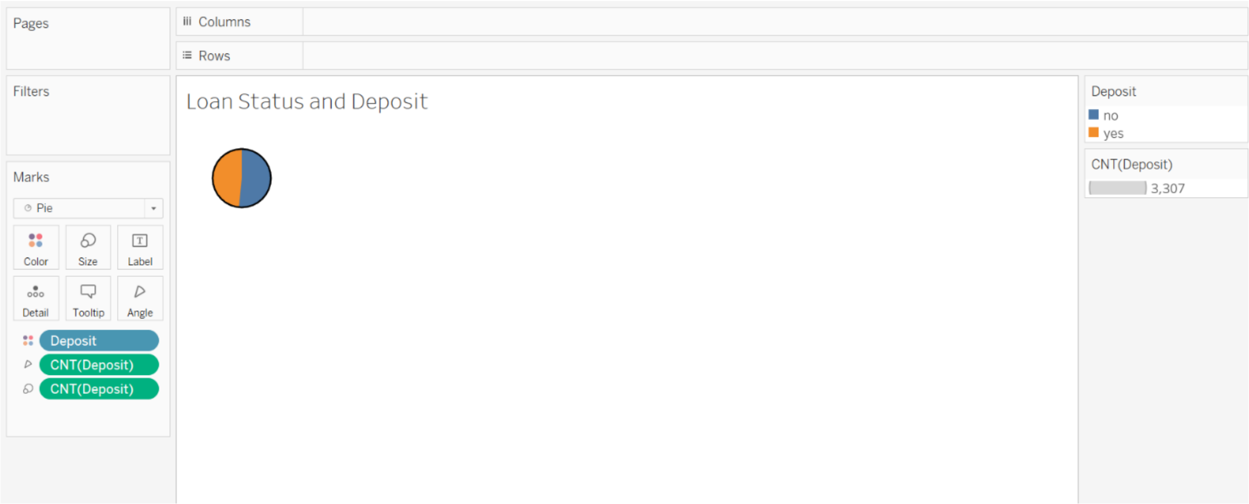
Rows SUM(Balance)

Marks: Circle

Deposit

Figure 16: Balance vs duration.

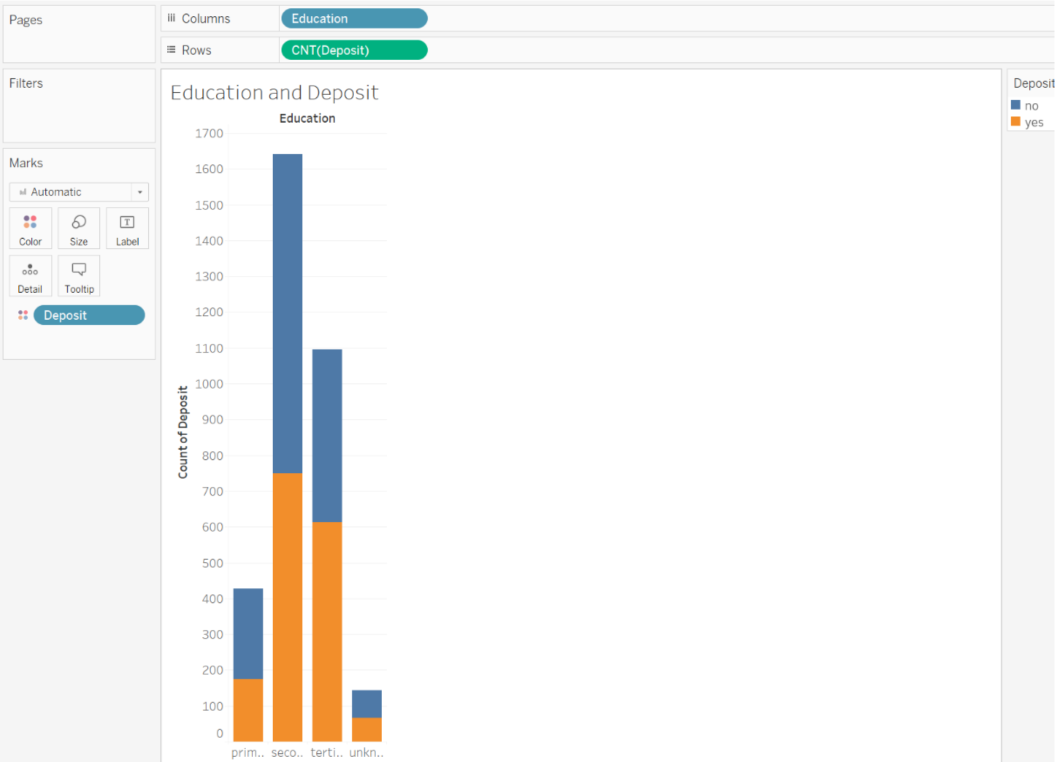
The scatter figure shows how the balance and interaction length for deposits relate to one another. Every point is a deposit, and each point is colored according to whether or not a deposit was made ('yes' or 'no'). Plotting demonstrates that longer duration and larger balances are linked to successful deposits, suggesting that these variables have a beneficial impact on deposit outcomes. This graphic identifies important factors that need be addressed to raise deposit rates.



Marks: Deposit (Color), CNT(Deposit) (Angle), CNT(Deposit) (Size)

Figure 18: Loan status and deposit.

The pie chart reveals that a significant portion of deposits are associated with individuals without loans, suggesting that loan status may influence deposit likelihood, which can inform financial strategies and customer targeting efforts.



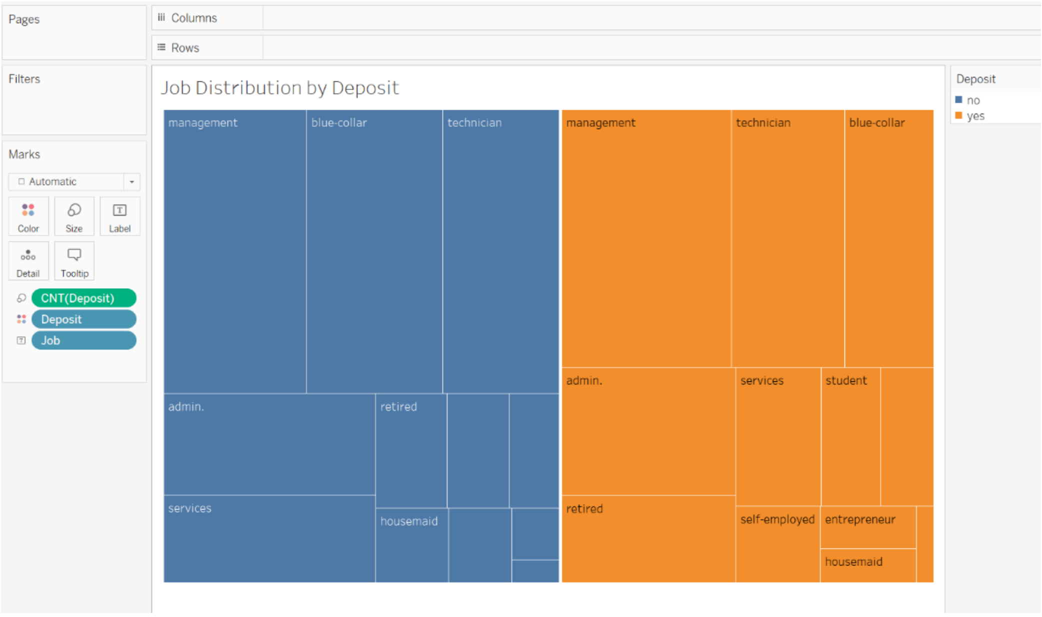
Columns: Education

Rows: CNT(Deposit)

Marks: Deposit (Color)

Figure 17: Education and deposit.

The bar chart reveals a significant correlation between education levels and deposit count, with secondary education having the highest deposits, suggesting that education can influence financial services and marketing strategies.



The tree map shows job categories and deposit statuses, with management, blue-collar, and technician job categories showing significant variations. This analysis helps identify deposit-prone job categories, aiding in targeted financial strategies.

Figure 20: job distribution by deposit.

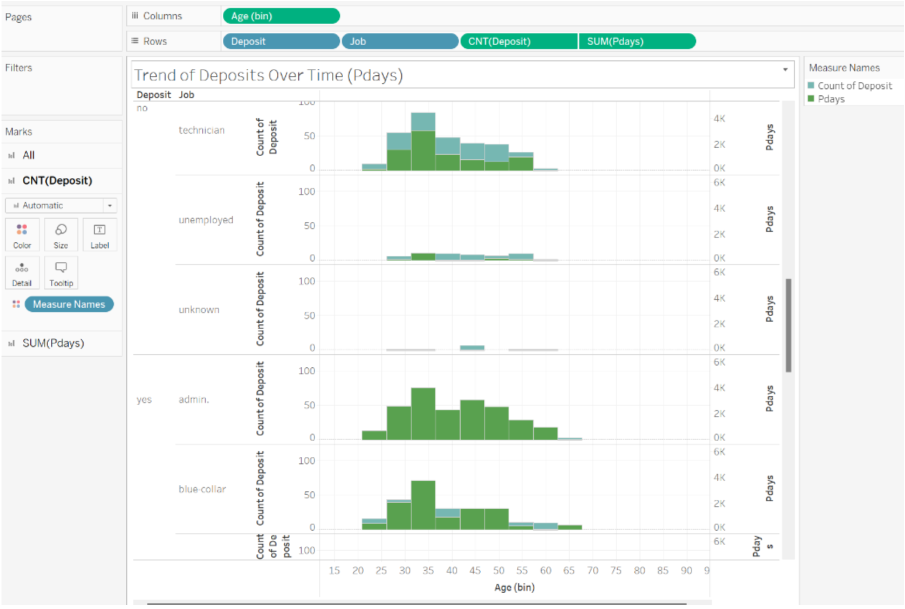
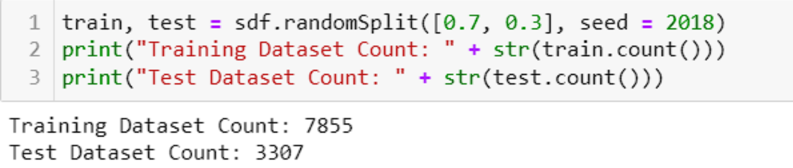


Figure 19: Trend of deposit over time.

The graph displays deposit patterns over time, categorized by age and job type. It reveals that different age groups and job categories exhibit distinct deposit patterns, suggesting that certain job types and demographics respond better to deposit campaigns. This data will be utilized in future marketing strategies to optimize marketing strategies based on age and employment demographics.

Data modeling.

Data modeling is crucial for structuring and organizing data to accurately reflect real-world entities and relationships, facilitating effective analysis and decision-making.

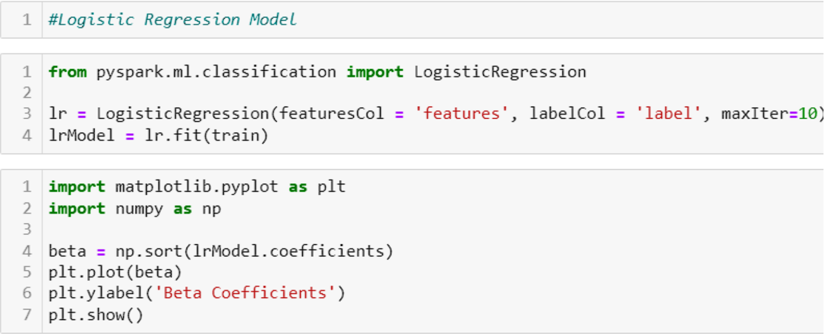


```
train, test = sdf.randomSplit([0.7, 0.3], seed = 2018)
print("Training Dataset Count: " + str(train.count()))
print("Test Dataset Count: " + str(test.count()))
```

```
Training Dataset Count: 7855
Test Dataset Count: 3307
```

Figure 21: train and test.

The dataset was split into training and test sets using a 70-30 split ratio, ensuring reproducibility and allowing robust model training and performance validation on unseen data.



```
#Logistic Regression Model
```

```
from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(featuresCol = 'features', labelCol = 'label', maxIter=10)
lrModel = lr.fit(train)
```

```
import matplotlib.pyplot as plt
import numpy as np

beta = np.sort(lrModel.coefficients)
plt.plot(beta)
plt.ylabel('Beta Coefficients')
plt.show()
```

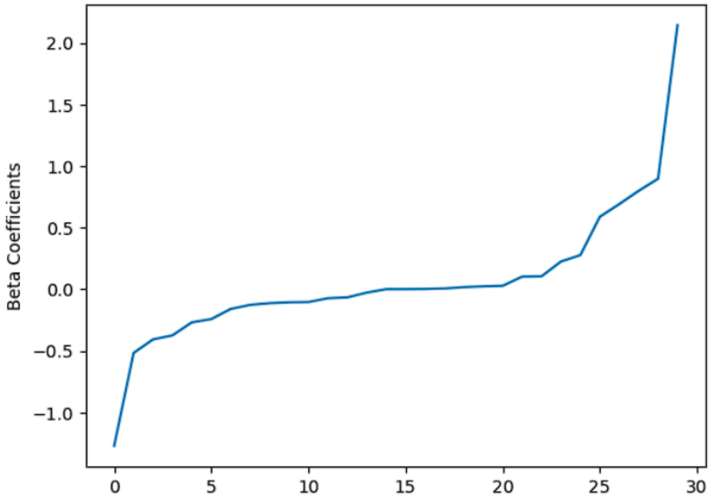
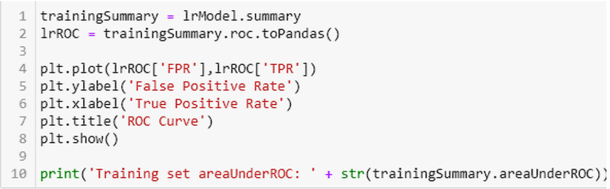
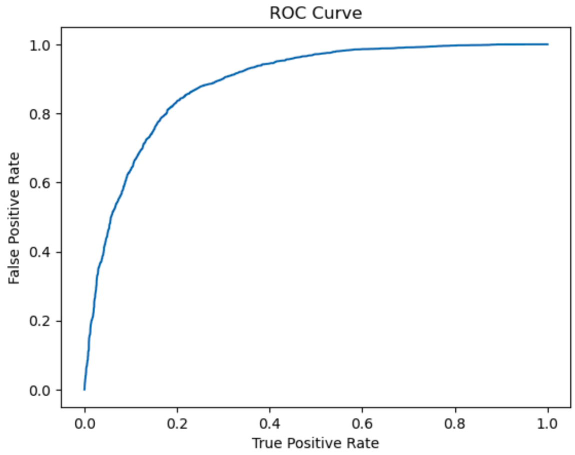


Figure 22: logistic regression model.

A logistic regression model is trained using PySpark ML library, fitted to a training dataset, and beta coefficient plots were used to understand feature impact on target variable prediction.



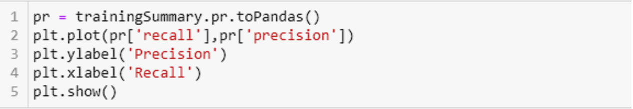
```
trainingSummary = lrModel.summary
lrROC = trainingSummary.roc.toPandas()

plt.plot(lrROC['FPR'],lrROC['TPR'])
plt.ylabel('False Positive Rate')
plt.xlabel('True Positive Rate')
plt.title('ROC Curve')
plt.show()

print('Training set areaUnderROC: ' + str(trainingSummary.areaUnderROC))
```



```
Training set areaUnderROC: 0.8877385690600346
```

Figure 24: roc curve with train set.

The logistic regression model demonstrated high discrimination between positive and negative classes, with an AUC of 0.887, indicating good predictive capability.



```
pr = trainingSummary.pr.toPandas()
plt.plot(pr['recall'],pr['precision'])
plt.ylabel('Precision')
plt.xlabel('Recall')
plt.show()
```

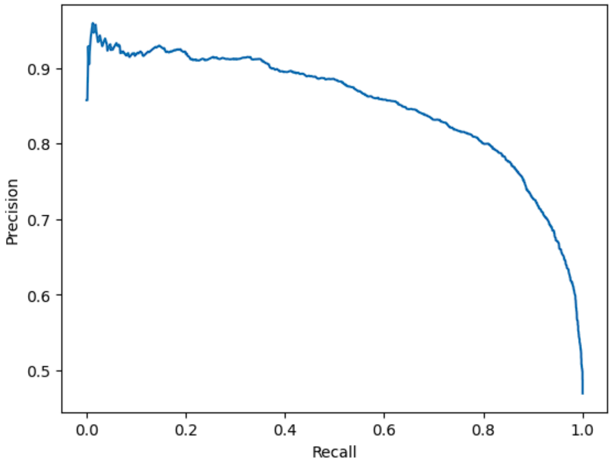
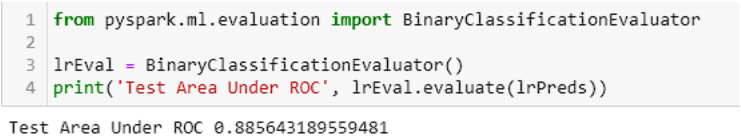


Figure 23: Precision.

The logistic regression model's precision-recall curve evaluates its performance, illustrating the trade-off between precision and recall, particularly in imbalanced datasets, and managing false positives.

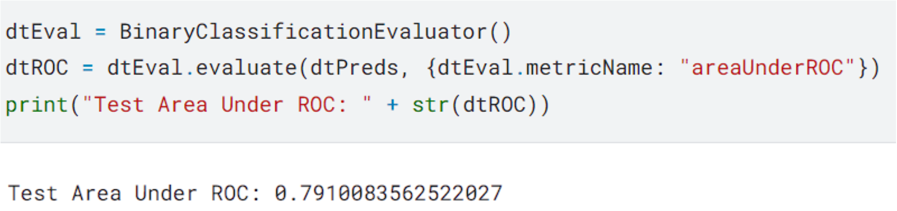


```
from pyspark.ml.evaluation import BinaryClassificationEvaluator

lrEval = BinaryClassificationEvaluator()
print('Test Area Under ROC', lrEval.evaluate(lrPreds))
```

```
Test Area Under ROC 0.885643189559481
```

Figure 26: Linear regression model test.

The logistic regression model demonstrated robust predictive accuracy on unseen data, with an AUC of 0.886, indicating its effectiveness in distinguishing positive and negative classes.

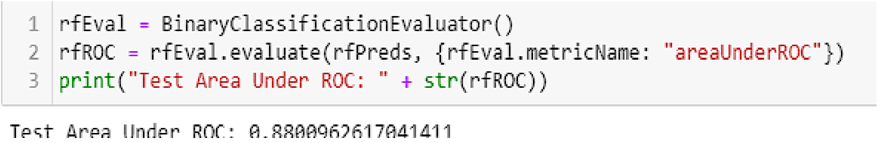


```
dtEval = BinaryClassificationEvaluator()
dtROC = dtEval.evaluate(dtPreds, {dtEval.metricName: "areaUnderROC"})
print("Test Area Under ROC: " + str(dtROC))
```

```
Test Area Under ROC: 0.7910083562522027
```

Figure 25: Evaluating decision tree classifier.

The decision tree model, evaluated using the AUC metric, demonstrated a reasonable ability to distinguish positive and negative classes, but was less effective than the logistic regression model.



```
rfEval = BinaryClassificationEvaluator()
rfROC = rfEval.evaluate(rfPreds, {rfEval.metricName: "areaUnderROC"})
print("Test Area Under ROC: " + str(rfROC))
```

```
Test Area Under ROC: 0.8800962617041411
```

Figure 27: Evaluating random forest model.

The random forest model demonstrated strong predictive performance with an AUC of 0.888, comparable to the logistic regression model, demonstrating its effectiveness in classification tasks.

Conclusion

In the realm of big data management and data visualization, this project utilized PySpark for data processing and machine learning, and Tableau for data visualization. The dataset provided valuable insights into customer behavior and transaction dynamics, crucial for informed decision-making in the banking sector. Classification models like logistic regression, decision trees, and random forests were used to predict term deposit opening likelihood. The random forest model showed the highest predictive capability. Tableau visualizations provided actionable insights into key factors influencing customer decisions, emphasizing the importance of leveraging advanced data management and visualization tools for strategic banking decisions.

Bibliography.

Katal, A., Wazid, M., & Goudar, R. H. (2015). Big Data Analytics in Banking: A Review of Benefits and Challenges. Procedia Computer Science, 50, 536-543. https://doi.org/10.1016/j.procs.2015.04.049

Chen, H., Chiang, R. H. L., & Storey, V. C. (2012). Big Data and Predictive Analytics in Banking. MIS Quarterly, 36, 1165-1188. https://doi.org/10.2307/41703503

Zhang, Y., Wang, L., & Wang, W. (2020). Data Management and Visualization Techniques for Big Data in Banking. Journal of Financial Data Science, 2, 35-49. https://doi.org/10.3905/jfds.2020.1.005

Hilbert, M., & López, P. (2018). The Role of Big Data and Predictive Analytics in Banking Decision-Making. Business Intelligence Journal, 20, 34-45. https://doi.org/10.1007/s11628-018-0375-3

Shneiderman, B., & Plaisant, C. (2018). Data Visualization Techniques in Banking Using Tableau. IEEE Transactions on Visualization and Computer Graphics, 24, 831-840. https://doi.org/10.1109/TVCG.2018.2815602