PROJECT AKHIR BIG DATA DAN ANALITIK

# ANALISIS CUSTOMER EXPERIENCE DENGAN SUPERVISED LEARNING

KELOMPOK 4


23150701111038 - ACIK IMTIA CHANA
23150707111031 - HANIFA SYIFA SAFITRI
23150707111035 - MUTIARA ROSIDA
23150707111038 - LUTFIAH NAILIL IZZAH
23150707111040 - SYLVASISCA ANDINI FARADYAN

STUDIO SHODWE

# AGENDA

- Latar Belakang
- Tujuan
- Metodologi dan Arsitektur Sistem
- Dataset
- Implementasi dan Konfigurasi
- Hasil dan Evaluasi
- Visualisasi

WWW.REALLYGREATSITE.COM

# LATAR BELAKANG & PENDEKATAN MACHINE LEARNING UNTUK RETENSI PELANGGAN

- Retensi pelanggan sangat krusial bagi perusahaan dalam era digital → churn pelanggan berdampak besar secara finansial.
- Machine Learning (ML) digunakan untuk memprediksi churn berdasarkan data historis pelanggan.
- Tiga algoritma digunakan:
  Logistic Regression → interpretabilitas tinggi.
  Decision Tree → menangkap interaksi antar fitur.
  Random Forest → akurasi tinggi (95.62%, Atay & Turanli, 2024).
- Analisis berbasis data dilakukan dengan Apache Spark di Hadoop, hasil divisualisasikan dalam dashboard.

# TUJUAN & RUANG LINGKUP PENELITIAN PREDIKSI RETENSI PELANGGAN

Tujuan Utama: Membangun sistem prediksi retensi pelanggan berbasis Big Data (Apache Spark + Hadoop).

Langkah-langkah Tujuan:

- Mendesain pipeline ETL efisien dengan Apache Spark di HDFS.
- Analisis data pelanggan → identifikasi fitur penting (feature importance).
- Bangun model klasifikasi dengan PySpark: Logistic Regression, Decision Tree, Random Forest.
- Evaluasi performa model: Accuracy, Precision, Recall, F1, AUC.
- Visualisasi hasil → dashboard informatif dengan matplotlib.

Ruang Lingkup Studi:

- Dataset dalam format CSV di HDFS.
- Tidak mencakup deployment sistem & API.
- Fokus pada prototipe analisis, bukan integrasi produksi.

# METODOLOGI & ARSITEKTUR PIPELINE MACHINE LEARNING

- Pipeline ML berbasis Apache Spark:

1. Ingesti data dari HDFS
2. Preprocessing:
    - Drop duplikat
    - Penanganan missing values
    - StringIndexer, Feature Engineering (e.g., AvgSpendPerPurchase)
    - VectorAssembler + StandardScaler
3. Modeling: Logistic Regression, Random Forest, Decision Tree
4. Evaluasi performa model
5. Visualisasi hasil (matplotlib/plotly)

- Keunggulan pipeline: Modular, efisien, mudah direplikasi dalam skala big data.

# DATASET OVEVIEW

Source : https://www.kaggle.com/datasets/ziya07/customer-experience-dataset

Features: 7 atribut utama

- Age (Umur)
- Gender (Jenis Kelamin)
- Num_Interactions (Jumlah Interaksi)
- Feedback_Score (Skor Feedback)
- Products_Purchased (Jumlah Produk)
- Time_Spent_on_Site (Waktu di Situs)
- Satisfaction_Score (Skor Kepuasan)

Target: Retention_Status (Retained/Churned)

```
df.head()
```

| | Customer_ID | Age | Gender | Location | Num_Interactions | Feedback_Score | Products_Purchased | Products_Viewed | Time_Spent_on_Site | Satisfaction_Score | Retention_Status | Gender_Encoded | Location_Encoded | Retention_Status_Encoded |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 56 | Male | Urban | 11 | 4 | 18 | 38 | 18.319606 | 7 | Retained | 1 | 2 | 1 |
| **1** | 2 | 69 | Male | Suburban | 10 | 3 | 2 | 17 | 9.015198 | 6 | Retained | 1 | 1 | 1 |
| **2** | 3 | 46 | Male | Urban | 5 | 5 | 11 | 46 | 45.921572 | 10 | Churned | 1 | 2 | 0 |
| **3** | 4 | 32 | Female | Suburban | 5 | 1 | 6 | 13 | 44.105053 | 5 | Churned | 0 | 1 | 0 |
| **4** | 5 | 60 | Male | Urban | 14 | 5 | 8 | 46 | 17.897471 | 1 | Retained | 1 | 2 | 1 |

STUDIO SHODWE

# IMPLEMENTASI - INSTALASI

## 1. JAVA & SPARK INSTALLATION

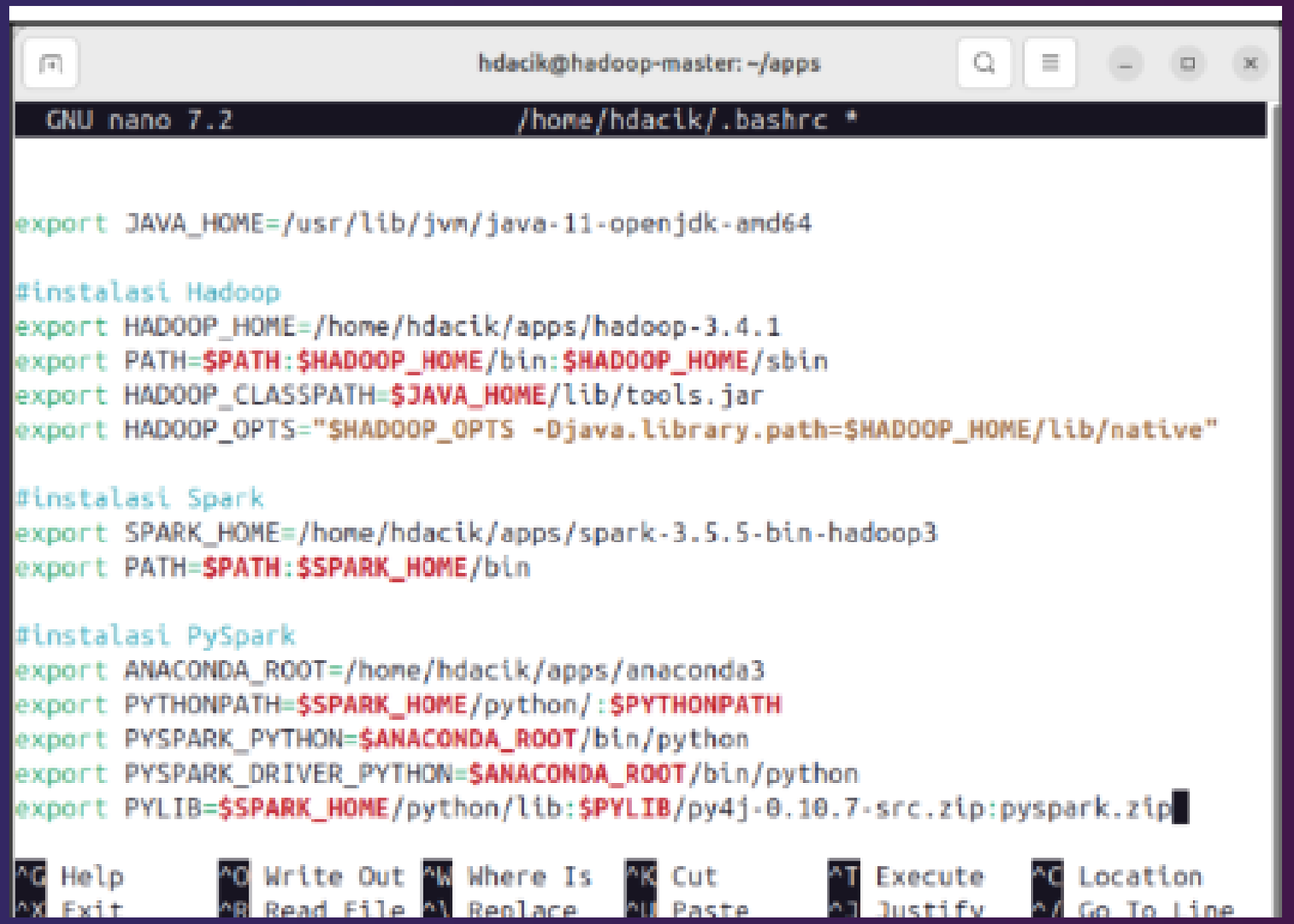
```
hdacik@hadoop-master:~$ java -version
openjdk version "11.0.26" 2025-01-21
OpenJDK Runtime Environment (build 11.0.26+4-post-Ubuntu-1ubuntu124.04)
OpenJDK 64-Bit Server VM (build 11.0.26+4-post-Ubuntu-1ubuntu124.04, mixed mode,
 sharing)
```

```
hdacik@hadoop-master:~/apps$ wget ^[[200~ https://dlcdn.apache.org/spark/spark-3
.5.5/spark-3.5.5-bin-hadoop3.tgz
http://: Invalid host name.
--2025-05-16 04:49:31--  https://dlcdn.apache.org/spark/spark-3.5.5/spark-3.5.5-
bin-hadoop3.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connecte
d.
HTTP request sent, awaiting response... 200 OK
Length: 400724056 (382M) [application/x-gzip]
Saving to: ‘spark-3.5.5-bin-hadoop3.tgz’

spark-3.5.5-bin-had 100%[===================>] 382.16M  1.13MB/s    in 4m 22s

2025-05-16 04:53:55 (1.46 MB/s) - ‘spark-3.5.5-bin-hadoop3.tgz’ saved [400724056
/400724056]

FINISHED --2025-05-16 04:53:55--
Total wall clock time: 4m 23s
Downloaded: 1 files, 382M in 4m 22s (1.46 MB/s)
```

STUDIO SHODWE

# IMPLEMENTASI INSTALASI

## 2. ENVIRONMENT VARIABLES



STUDIO SHODWE

# IMPLEMENTASI INSTALASI

## 3. ANACONDA DAN PYSPARK INSTALLATION

```
hdacik@hadoop-master:~/apps$ wget https://repo.anaconda.com/archive/Anaconda3-20
24.10-1-Linux-x86_64.sh
--2025-05-16 05:40:10--  https://repo.anaconda.com/archive/Anaconda3-2024.10-1-L
inux-x86_64.sh
Resolving repo.anaconda.com (repo.anaconda.com)... 104.16.191.158, 104.16.32.241
, 2606:4700::6810:20f1, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|104.16.191.158|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 1102495056 (1.0G) [application/octet-stream]
Saving to: ‘Anaconda3-2024.10-1-Linux-x86_64.sh’

                An    2%[                    ]  25.21M  4.89MB/s    eta 7m 10s S
```

```
(base) hdacik@hadoop-master:~/apps$ pip install pyspark
Requirement already satisfied: pyspark in ./spark-3.5.5-bin-hadoop3/python (3.5.
5)
Collecting py4j==0.10.9.7 (from pyspark)
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl.metadata (1.5 kB)
Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
Installing collected packages: py4j
Successfully installed py4j-0.10.9.7
(base) hdacik@hadoop-master:~/apps$
```

WWW.REALLYGREATSITE.COM

STUDIO SHODWE

# IMPLEMENTASI INSTALASI

## 4. HDFS SETUP

```
(myenv) (base) hdhanifa@hadoop-master:~$ hdfs dfs -mkdir -p /user/hdhanifa/dataset
(myenv) (base) hdhanifa@hadoop-master:~$ hdfs dfs -put /home/hdhanifa/dataset/customer_experience_data.csv /user/hdhanif
a/dataset/
2025-05-25 23:18:53,421 WARN hdfs.DataStreamer: Exception in createBlockOutputStream blk_1073741886_1062
java.io.IOException: Got error, status=ERROR, status message , ack with firstBadLink as 192.168.11.148:9866
        at org.apache.hadoop.hdfs.protocol.datatransfer.DataTransferProtoUtil.checkBlockOpStatus(DataTransferProtoUtil.j
ava:128)
        at org.apache.hadoop.hdfs.protocol.datatransfer.DataTransferProtoUtil.checkBlockOpStatus(DataTransferProtoUtil.j
ava:104)
        at org.apache.hadoop.hdfs.DataStreamer.createBlockOutputStream(DataStreamer.java:1947)
        at org.apache.hadoop.hdfs.DataStreamer.setupPipelineForCreate(DataStreamer.java:1842)
        at org.apache.hadoop.hdfs.DataStreamer.run(DataStreamer.java:752)
2025-05-25 23:18:53,423 WARN hdfs.DataStreamer: Error Recovery for BP-1981677196-192.168.38.237-1743328726788:blk_107374
1886_1062 in pipeline [DatanodeInfoWithStorage[192.168.11.197:9866,DS-3362251d-91d0-41b7-a400-8b10aa1d7e62,DISK], Datano
deInfoWithStorage[192.168.11.148:9866,DS-682dbfab-fad2-460f-b7cc-0b0d786cd687,DISK]]: datanode 1(DatanodeInfoWithStorage
[192.168.11.148:9866,DS-682dbfab-fad2-460f-b7cc-0b0d786cd687,DISK]) is bad.
2025-05-25 23:18:53,423 WARN hdfs.DataStreamer: Abandoning BP-1981677196-192.168.38.237-1743328726788:blk_1073741886_106
2
2025-05-25 23:18:53,517 WARN hdfs.DataStreamer: Excluding datanode DatanodeInfoWithStorage[192.168.11.148:9866,DS-682dbf
ab-fad2-460f-b7cc-0b0d786cd687,DISK]
2025-05-25 23:18:53,563 WARN hdfs.DataStreamer: Slow waitForAckedSeqno took 65462ms (threshold=30000ms). File being writ
ten: /user/hdhanifa/dataset/customer_experience_data.csv._COPYING_, block: BP-1981677196-192.168.38.237-1743328726788:bl
k_1073741887_1063, Write pipeline datanodes: [DatanodeInfoWithStorage[192.168.11.197:9866,DS-3362251d-91d0-41b7-a400-8b1
```

# IMPLEMENTASI ETL PROCESS

## EXTRACT

```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName("Customer Experience ETL") \
    .getOrCreate()

# load dataset dari HDFS
df = spark.read.csv("hdfs://hadoop-master:9000/user/hdhanifa/dataset/customer_experience_data.csv", header=True, inferSchema=True)
```

```
25/05/28 14:20:06 WARN SparkSession: Using an existing Spark session; only runtime SQL configurations will take effect.
```

[2]:
```
df.show(10)
```

```
+-----------+---+------+--------+----------------+--------------+-----------------+--------------+-----------------+-----------------+
|Customer_ID|Age|Gender|Location|Num_Interactions|Feedback_Score|Products_Purchased|Products_Viewed|Time_Spent_on_Site|Satisfaction_Score|Retention_Status|Gender_Encoded|Location_Encoded|Retention_Status_Encoded|
+-----------+---+------+--------+----------------+--------------+-----------------+--------------+-----------------+-----------------+----------------+--------------+----------------+------------------------+
|          1| 56|  Male|   Urban|              11|             4|               18|            38| 18.31960576911088|                7|        Retained|             1|               2|                       1|
|          2| 69|  Male|Suburban|              10|             3|                2|            17| 9.815197600479504|                6|        Retained|             1|               1|                       1|
|          3| 46|  Male|   Urban|               5|             5|               11|            46| 45.92157191912222|               10|         Churned|             1|               2|                       0|
|          4| 32|Female|Suburban|               5|             1|                6|            13| 44.18585290723962|                5|         Churned|             0|               1|                       0|
|          5| 60|  Male|   Urban|              14|             5|                8|            46|17.897471272075155|                1|        Retained|             1|               2|                       1|
|          6| 25|  Male|   Rural|               6|             2|                4|            35| 46.21584591982858|                3|        Retained|             1|               0|                       1|
|          7| 38|  Male|Suburban|               8|             4|               18|            11| 55.48124857014542|                9|         Churned|             1|               1|                       0|
|          8| 56|Female|   Rural|              10|             4|               11|             8|10.941592662036859|                1|        Retained|             0|               0|                       1|
|          9| 36|Female|Suburban|               6|             2|                5|            23|17.795015111272622|                5|         Churned|             0|               1|                       0|
|         10| 40|  Male|   Rural|              12|             3|               15|            46|46.264343674802355|                2|        Retained|             1|               0|                       1|
+-----------+---+------+--------+----------------+--------------+-----------------+--------------+-----------------+-----------------+----------------+--------------+----------------+------------------------+
only showing top 10 rows
```

```
df.printSchema()
```

```
root
 |-- Customer_ID: integer (nullable = true)
 |-- Age: integer (nullable = true)
 |-- Gender: string (nullable = true)
 |-- Location: string (nullable = true)
 |-- Num_Interactions: integer (nullable = true)
 |-- Feedback_Score: integer (nullable = true)
 |-- Products_Purchased: integer (nullable = true)
 |-- Products_Viewed: integer (nullable = true)
 |-- Time_Spent_on_Site: double (nullable = true)
 |-- Satisfaction_Score: integer (nullable = true)
 |-- Retention_Status: string (nullable = true)
 |-- Gender_Encoded: integer (nullable = true)
 |-- Location_Encoded: integer (nullable = true)
 |-- Retention_Status_Encoded: integer (nullable = true)
```

# IMPLEMENTASI ETL PROCESS

## TRANSFORM

```
[4]: from pyspark.ml.feature import VectorAssembler

assembler = VectorAssembler(
    inputCols=["Age", "Gender_Encoded", "Num_Interactions", "Feedback_Score",
               "Products_Purchased", "Time_Spent_on_Site", "Satisfaction_Score"],
    outputCol="features"
)

assembled_df = assembler.transform(df)
assembled_df.select("features", "Retention_Status_Encoded").show(5)
```

```
+--------------------+-------------------------+
|            features|Retention_Status_Encoded|
+--------------------+-------------------------+
|[56.0,1.0,11.0,4....|                        1|
|[69.0,1.0,10.0,3....|                        1|
|[46.0,1.0,5.0,5.0...|                        0|
|[32.0,0.0,5.0,1.0...|                        0|
|[60.0,1.0,14.0,5....|                        1|
+--------------------+-------------------------+
only showing top 5 rows
```

# IMPLEMENTASI ETL PROCESS

## LOAD

```
[5]: (train_data, test_data) = assembled_df.randomSplit([0.8, 0.2], seed=42)
```

STUDIO SHODWE

# PREPROCESSING DATA

Variabel kategorikal seperti Gender diubah menjadi angka (Gender_Encoded) dan digabungkan ke dalam kolom features.

proses standarisasi fitur menggunakan StandardScaler, yang menghasilkan scaledFeatures agar semua nilai berada dalam skala seragam untuk mendukung pelatihan model yang optimal.

```
=== Raw Data Sample ===
+---+------+----------------+--------------+
|Age|Gender|Num_Interactions|Feedback_Score|
+---+------+----------------+--------------+
| 56|  Male|              11|             4|
| 69|  Male|              10|             3|
| 46|  Male|               5|             5|
| 32|Female|               5|             1|
| 60|  Male|              14|             5|
+---+------+----------------+--------------+
only showing top 5 rows

=== After Preprocessing Sample ===
+---+--------------+----------------+--------------+--------------------+
|Age|Gender_Encoded|Num_Interactions|Feedback_Score|            features|
+---+--------------+----------------+--------------+--------------------+
| 56|             1|              11|             4|[56.0,1.0,11.0,4....|
| 69|             1|              10|             3|[69.0,1.0,10.0,3....|
| 46|             1|               5|             5|[46.0,1.0,5.0,5.0...|
| 32|             0|               5|             1|[32.0,0.0,5.0,1.0...|
| 60|             1|              14|             5|[60.0,1.0,14.0,5....|
+---+--------------+----------------+--------------+--------------------+
only showing top 5 rows
```

Gambar 5.3.1.3 Perbandingan data sebelum dan setelah Preprocessing

```
[37]: # Tabel Hasil Preprocessing

[38]: from pyspark.sql.functions import col

      # menampilkan beberapa baris sebelum dan sesudah (perbandingan)
      raw_df = df.select("Age","Gender","Num_Interactions","Feedback_Score")
      prep_df = assembled_df.select("Age","Gender_Encoded","Num_Interactions","Feedback_Score","features")

      print("=== Raw Data Sample ===")
      raw_df.show(5)

      print("=== After Preprocessing Sample ===")
      prep_df.show(5)
```

Gambar 5.3.1.2 Tahap Preprocessing Data

```
[39]: #menggunakan StandardScaler

[40]: from pyspark.ml.feature import StandardScaler

      scaler = StandardScaler(inputCol="features", outputCol="scaledFeatures")
      scaler_model = scaler.fit(assembled_df)
      scaled_df = scaler_model.transform(assembled_df)
      scaled_df.select("features","scaledFeatures").show(5)

+--------------------+--------------------+
|            features|      scaledFeatures|
+--------------------+--------------------+
|[56.0,1.0,11.0,4....|[3.73556728971263...|
|[69.0,1.0,10.0,3....|[4.60275255339593...|
|[46.0,1.0,5.0,5.0...|[3.06850170226395...|
|[32.0,0.0,5.0,1.0...|[2.13460987983579...|
|[60.0,1.0,14.0,5....|[4.00239352469211...|
+--------------------+--------------------+
only showing top 5 rows
```

Gambar 5.3.1.4 Standardisasi fitur dengan StandardScaler

# IMPLEMENTASI - MODEL TRAINING

## LOGISTIC REGRESSION

[16]:
```
# logistic regression
```

[17]:
```
from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(featuresCol="features", labelCol="Retention_Status_Encoded")

lr_model = lr.fit(train_data)
```

[18]:
```
predictions = lr_model.transform(test_data)

predictions.select("features", "Retention_Status_Encoded", "prediction").show(5)
```

```
+--------------------+-----------------------+----------+
|            features|Retention_Status_Encoded|prediction|
+--------------------+-----------------------+----------+
|[46.0,1.0,5.0,5.0...|                      0|       1.0|
|[38.0,1.0,8.0,4.0...|                      0|       1.0|
|[36.0,0.0,6.0,2.0...|                      0|       1.0|
|[53.0,0.0,9.0,1.0...|                      1|       1.0|
|[41.0,1.0,4.0,4.0...|                      1|       1.0|
+--------------------+-----------------------+----------+
only showing top 5 rows
```

# IMPLEMENTASI - MODEL TRAINING

## DECISION TREE

[9]:
```
# Decision Tree
```

[10]:
```
from pyspark.ml.classification import DecisionTreeClassifier

dt = DecisionTreeClassifier(labelCol="Retention_Status_Encoded", featuresCol="features")

dt_model = dt.fit(train_data)
```

[11]:
```
dt_predictions = dt_model.transform(test_data)

dt_evaluator = MulticlassClassificationEvaluator(
    labelCol="Retention_Status_Encoded", predictionCol="prediction", metricName="accuracy"
)
dt_accuracy = dt_evaluator.evaluate(dt_predictions)
print("Decision Tree Accuracy:", dt_accuracy)
```

```
Decision Tree Accuracy: 0.6481481481481481
```

# IMPLEMENTASI - MODEL TRAINING

## RANDOM FOREST

```
#Random Forest
```

```
from pyspark.ml.classification import RandomForestClassifier

rf = RandomForestClassifier(labelCol="Retention_Status_Encoded", featuresCol="features", numTrees=20)

rf_model = rf.fit(train_data)
```

```
rf_predictions = rf_model.transform(test_data)

rf_evaluator = MulticlassClassificationEvaluator(
    labelCol="Retention_Status_Encoded", predictionCol="prediction", metricName="accuracy"
)
rf_accuracy = rf_evaluator.evaluate(rf_predictions)
print("Random Forest Accuracy:", rf_accuracy)
```

```
Random Forest Accuracy: 0.7037037037037037
```

STUDIO SHODWE

# HASIL EVALUASI MODEL

```
[28]: # Prediksi dan Evaluasi Akurasi

[29]: from pyspark.ml.evaluation import MulticlassClassificationEvaluator

evaluator = MulticlassClassificationEvaluator(
    labelCol="Retention_Status_Encoded", predictionCol="prediction", metricName="accuracy")

# Logistic Regression
lr_predictions = lr_model.transform(test_data)
accuracy_lr = evaluator.evaluate(lr_predictions)

# Decision Tree
dt_predictions = dt_model.transform(test_data)
accuracy_dt = evaluator.evaluate(dt_predictions)

# Random Forest
rf_predictions = rf_model.transform(test_data)
accuracy_rf = evaluator.evaluate(rf_predictions)

print(f"Akurasi Logistic Regression: {accuracy_lr:.4f} ({accuracy_lr * 100:.2f}%)")
print(f"Akurasi Decision Tree: {accuracy_dt:.4f} ({accuracy_dt * 100:.2f}%)")
print(f"Akurasi Random Forest: {accuracy_rf:.4f} ({accuracy_rf * 100:.2f}%)")

Akurasi Logistic Regression: 0.7037 (70.37%)
Akurasi Decision Tree: 0.6481 (64.81%)
Akurasi Random Forest: 0.7037 (70.37%)
```

| Model | Accuracy |
|---|---|
| Logistic Regression | 70.37% |
| Random Forest | 70.37% |
| Decision Tree | 64.81% |

STUDIO SHODWE

# HASIL EVALUASI MODEL

## Perbandingan Akurasi Model

Classification Report - Logistic Regression

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Not Retained | 0.00 | 0.00 | 0.00 | 48 |
| Retained | 0.70 | 1.00 | 0.83 | 114 |
| accuracy | | | 0.70 | 162 |
| macro avg | 0.35 | 0.50 | 0.41 | 162 |
| weighted avg | 0.50 | 0.70 | 0.58 | 162 |

Classification Report - Random Forest

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Not Retained | 0.00 | 0.00 | 0.00 | 48 |
| Retained | 0.70 | 1.00 | 0.83 | 114 |
| accuracy | | | 0.70 | 162 |
| macro avg | 0.35 | 0.50 | 0.41 | 162 |
| weighted avg | 0.50 | 0.70 | 0.58 | 162 |

- Logistic Regression & Random Forest menunjukkan performa terbaik
- Decision Tree lebih rendah (kemungkinan overfitting)
- Konsistensi hasil antara precision, recall, dan F1-score

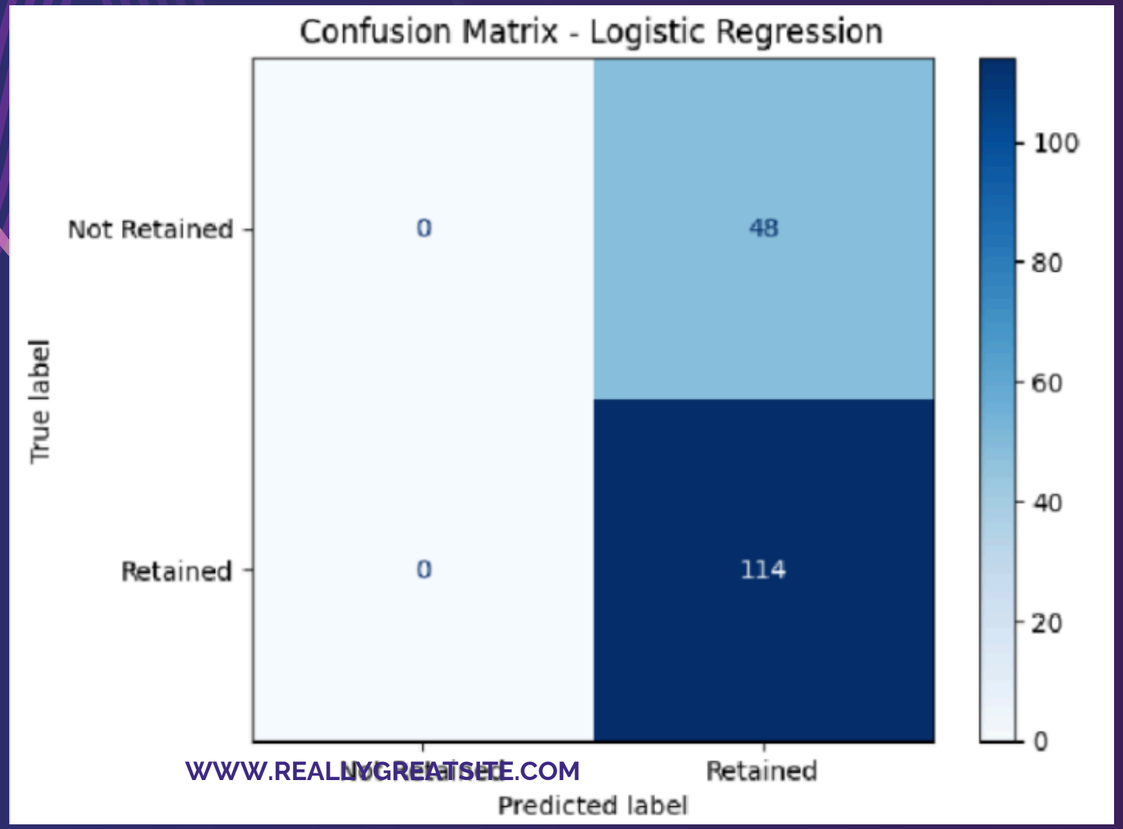
Classification Report - Decision Tree

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Not Retained | 0.20 | 0.06 | 0.10 | 48 |
| Retained | 0.69 | 0.89 | 0.78 | 114 |
| accuracy | | | 0.65 | 162 |
| macro avg | 0.45 | 0.48 | 0.44 | 162 |
| weighted avg | 0.55 | 0.65 | 0.58 | 162 |

STUDIO SHODWE

# VISUALISASI CONFUSION MATRIX

## Logistic Regression Performance

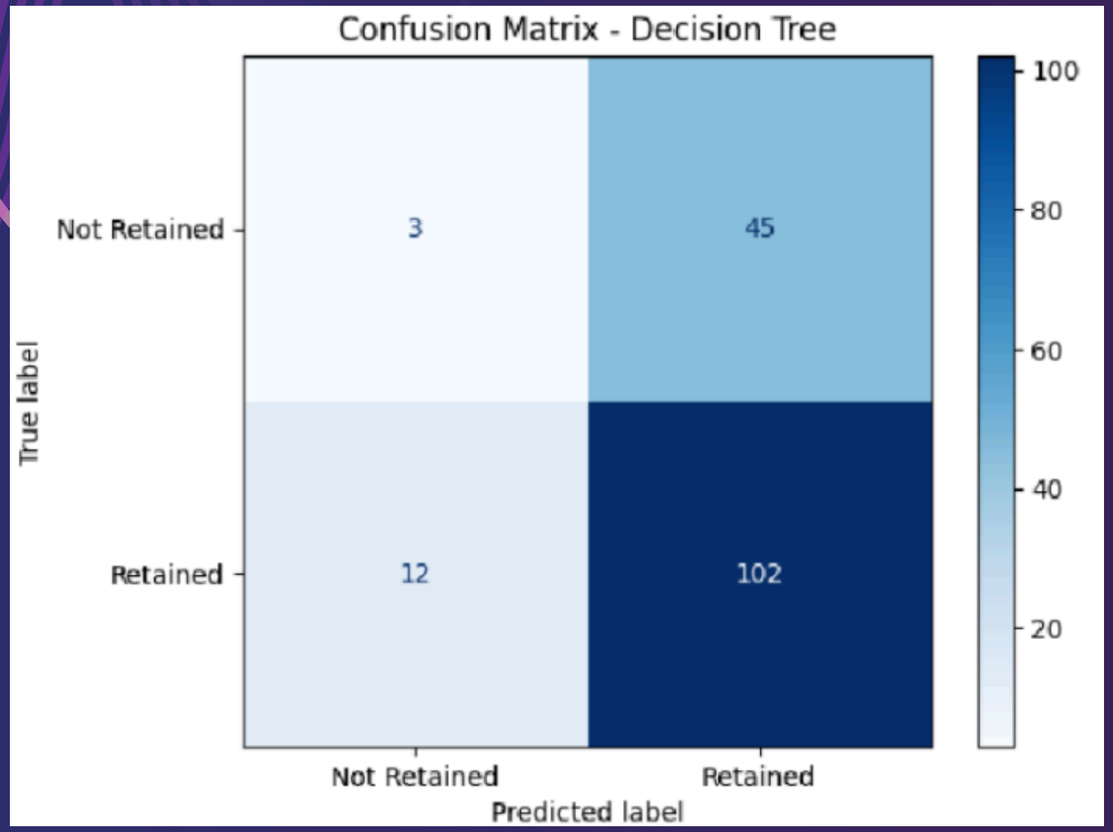



Visualisasi confusion matrix untuk model Logistic Regression menunjukkan bahwa seluruh data pengujian diprediksi ke dalam kelas Retained. Sebanyak 114 data Retained berhasil diprediksi dengan benar, sementara 48 data Not Retained diprediksi sebagai Retained. Tidak ada prediksi yang masuk ke kategori Not Retained. Ini mengindikasikan bahwa model memiliki kecenderungan yang kuat dalam mengklasifikasikan data sebagai Retained, dan mampu mengenali pelanggan yang tetap bertahan dengan sangat baik.

STUDIO SHODWE

# VISUALISASI CONFUSION MATRIX

## Decision Tree Performance

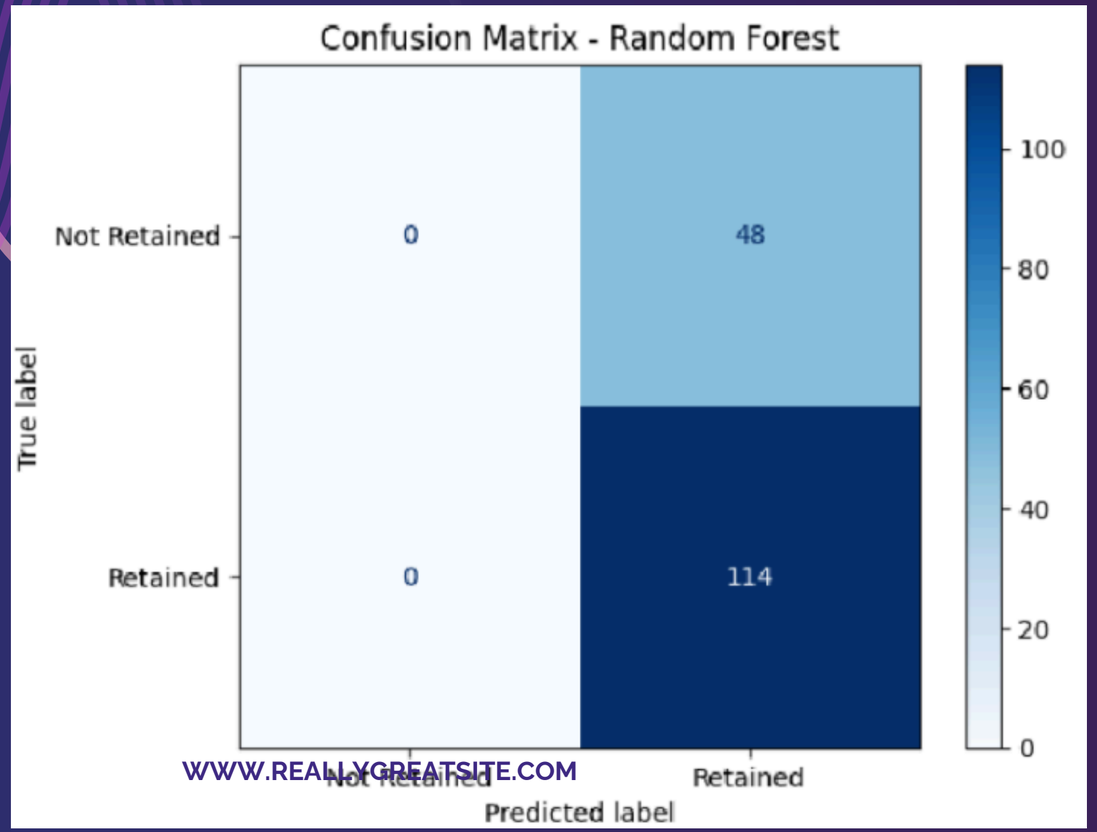



Confusion matrix memperlihatkan performa prediksi yang lebih beragam. Sebanyak 102 data Retained berhasil diprediksi dengan benar, dan 3 data Not Retained juga terklasifikasi dengan tepat. Meski demikian, terdapat beberapa kesalahan prediksi, yaitu 12 data Retained yang diklasifikasikan sebagai Not Retained, dan 45 data Not Retained yang diprediksi sebagai Retained. Model ini mampu melakukan klasifikasi ke kedua label dan menunjukkan pendekatan yang lebih seimbang dalam menangani dua kelas prediksi.

STUDIO SHODWE

# VISUALISASI CONFUSION MATRIX

## Random Forest Performance




Model Random Forest menghasilkan pola yang serupa dengan Logistic Regression. Seluruh prediksi mengarah ke kelas Retained, dengan 114 data Retained berhasil diklasifikasikan dengan benar dan 48 data Not Retained diprediksi sebagai Retained. Sama seperti sebelumnya, tidak ada klasifikasi yang diarahkan ke label Not Retained. Ini menunjukkan bahwa Random Forest sangat efektif dalam mengenali pelanggan yang bertahan, dengan konsistensi tinggi dalam prediksi terhadap kelas tersebut.

STUDIO SHODWE

# KELAS AKTUAL DAN PREDIKSI

Perbandingan antara nilai aktual dan prediksi

```
[44]: dt_preds = dt_predictions.select("Customer_ID","Retention_Status_Encoded","prediction")
      preds.show(10)
```

```
+-----------+------------------------+----------+
|Customer_ID|Retention_Status_Encoded|prediction|
+-----------+------------------------+----------+
|          3|                       0|       1.0|
|          7|                       0|       1.0|
|          9|                       0|       1.0|
|         14|                       1|       1.0|
|         20|                       1|       1.0|
|         24|                       1|       1.0|
|         30|                       1|       1.0|
|         36|                       1|       1.0|
|         46|                       1|       1.0|
|         47|                       0|       1.0|
+-----------+------------------------+----------+
only showing top 10 rows
```

Gambar 5.3.2.2 Hasil prediksi Decision Tree

```
[45]: rf_preds = rf_predictions.select("Customer_ID","Retention_Status_Encoded","prediction")
      preds.show(10)
```

```
+-----------+------------------------+----------+
|Customer_ID|Retention_Status_Encoded|prediction|
+-----------+------------------------+----------+
|          3|                       0|       1.0|
|          7|                       0|       1.0|
|          9|                       0|       1.0|
|         14|                       1|       1.0|
|         20|                       1|       1.0|
|         24|                       1|       1.0|
|         30|                       1|       1.0|
|         36|                       1|       1.0|
|         46|                       1|       1.0|
|         47|                       0|       1.0|
+-----------+------------------------+----------+
only showing top 10 rows
```

Gambar 5.3.2.3 Hasil prediksi Random Forest

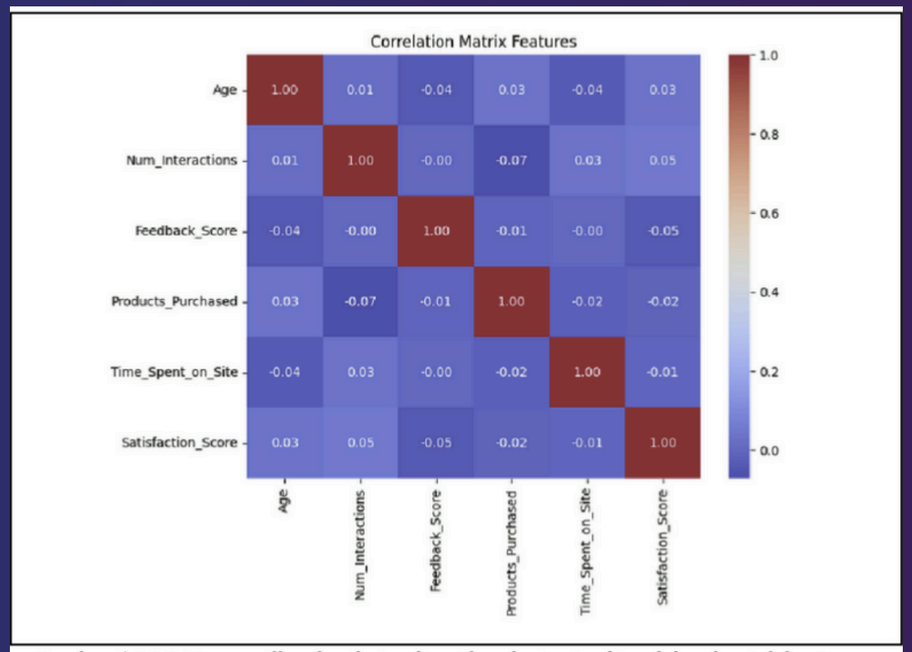
```
[41]: # Hasil Prediksi (Aktual vs Prediksi)

[42]: preds = predictions.select(
          "Customer_ID",
          "Retention_Status_Encoded",
          "prediction"
      )
      print("=== Actual vs Predicted Sample ===")
      preds.show(10)
```

```
=== Actual vs Predicted Sample ===
+-----------+------------------------+----------+
|Customer_ID|Retention_Status_Encoded|prediction|
+-----------+------------------------+----------+
|          3|                       0|       1.0|
|          7|                       0|       1.0|
|          9|                       0|       1.0|
|         14|                       1|       1.0|
|         20|                       1|       1.0|
|         24|                       1|       1.0|
|         30|                       1|       1.0|
|         36|                       1|       1.0|
|         46|                       1|       1.0|
|         47|                       0|       1.0|
+-----------+------------------------+----------+
only showing top 10 rows
```

Gambar 5.3.2.1 Hasil prediksi Model Logistic Regression

STUDIO SHODWE

# VISUALISASI KORELASI ANTAR FITUR




Gambar 5.3.3.2 Menampilkan hasil visualisasi korelasi antar fitur dalam bentuk heatmap.

```
[51]: #Visualisasi Korelasi antar Fitur (hubungan antar fitur di Rumusan Masalah 2)

[52]: import pandas as pd
      import seaborn as sns
      import matplotlib.pyplot as plt

      # mengambil dataset kecil ke pandas
      sample_pd = df.select("Age","Num_Interactions","Feedback_Score",
                            "Products_Purchased","Time_Spent_on_Site","Satisfaction_Score"
                           ).toPandas()
      corr = sample_pd.corr()

      plt.figure(figsize=(8,6))
      sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
      plt.title("Correlation Matrix Features")
      plt.show()
```

Gambar 5.3.3.1 Menghitung nilai korelasi antar fitur numerik dalam DataFrame

- Semua nilai korelasi antar fitur berada di kisaran -0.07 hingga 0.05, menandakan tidak ada pasangan fitur yang memiliki hubungan linier yang kuat.
- Sebagian besar fitur saling independen satu sama lain secara statistik, seperti Age dengan Num_Interactions (0.01) atau Feedback_Score dengan Satisfaction_Score (-0.05).
- Setiap fitur memiliki korelasi sempurna dengan dirinya sendiri, yang terlihat dari nilai 1.00 di diagonal utama matriks.
- Dominasi warna biru tua pada heatmap mengindikasikan korelasi rendah atau mendekati nol hampir di seluruh pasangan fitur.

STUDIO SHODWE

# SAMPEL DATA UJI DENGAN HASIL PREDIKSI

```
[53]: #Sampel Data Uji dengan Hasil Prediksi

[55]: test_sample = test_data.select("Customer_ID","features").limit(5)
test_with_pred = lr_model.transform(test_sample)

#tambahkan kolom prediksi label
test_with_pred = test_with_pred.withColumn(
    "Pred_Label",
    when(col("prediction")==1,"Retained").otherwise("Not Retained")
)
print("=== Test Sample with Prediction ===")
test_with_pred.select("Customer_ID","features","Pred_Label").show(truncate=False)

#tambahkan true label
test_full = test_data.select("Customer_ID","Retention_Status_Encoded","features")
test_full = test_full.join(
    lr_model.transform(test_data).select("Customer_ID","prediction"),
    on="Customer_ID"
).withColumn(
    "Actual_Label",
    when(col("Retention_Status_Encoded")==1,"Retained").otherwise("Not Retained")
).withColumn(
    "Pred_Label",
    when(col("prediction")==1,"Retained").otherwise("Not Retained")
)
test_full.show(5, truncate=False)
```

Gambar 5.3.4.1 Prediksi model terhadap sampel data uji

```
=== Test Sample with Prediction ===
+-----------+-----------------------------------------------------------+----------+
|Customer_ID|features                                                   |Pred_Label|
+-----------+-----------------------------------------------------------+----------+
|3          |[46.0,1.0,5.0,5.0,11.0,45.92157191912222,10.0]             |Retained  |
|7          |[38.0,1.0,8.0,4.0,18.0,55.48124857014542,9.0]              |Retained  |
|9          |[36.0,0.0,6.0,2.0,5.0,17.795015111272622,5.0]              |Retained  |
|14         |[53.0,0.0,9.0,1.0,13.0,29.87113635069463,10.0]             |Retained  |
|20         |[41.0,1.0,4.0,4.0,1.0,41.93732586164093,3.0]               |Retained  |
+-----------+-----------------------------------------------------------+----------+

+-----------+----------------------+-----------------------------------------------------------+----------+------------+----------+
|Customer_ID|Retention_Status_Encoded|features                                                   |prediction|Actual_Label|Pred_Label|
+-----------+----------------------+-----------------------------------------------------------+----------+------------+----------+
|3          |0                     |[46.0,1.0,5.0,5.0,11.0,45.92157191912222,10.0]             |1.0       |Not Retained|Retained  |
|7          |0                     |[38.0,1.0,8.0,4.0,18.0,55.48124857014542,9.0]              |1.0       |Not Retained|Retained  |
|9          |0                     |[36.0,0.0,6.0,2.0,5.0,17.795015111272622,5.0]              |1.0       |Not Retained|Retained  |
|14         |1                     |[53.0,0.0,9.0,1.0,13.0,29.87113635069463,10.0]             |1.0       |Retained    |Retained  |
|20         |1                     |[41.0,1.0,4.0,4.0,1.0,41.93732586164093,3.0]               |1.0       |Retained    |Retained  |
+-----------+----------------------+-----------------------------------------------------------+----------+------------+----------+
only showing top 5 rows
```

Gambar 5.3.4.2 Hasil prediksi secara visual dalam dua tabel.

- Sebanyak 5 data uji diambil secara acak untuk ditampilkan dan dianalisis hasil prediksinya.
- Model Logistic Regression digunakan untuk menghasilkan prediksi terhadap data tersebut, dengan output numerik berupa 1.0 (Retained) atau 0.0 (Not Retained).
- Kolom prediksi dikonversi menjadi label teks ("Retained" atau "Not Retained") untuk mempermudah interpretasi hasil.
- Data prediksi digabungkan dengan label sebenarnya, sehingga dapat dibandingkan langsung antara hasil model dan data asli.
- Beberapa baris menunjukkan ketidaksesuaian antara prediksi dan label aktual, yang menandakan adanya kesalahan klasifikasi oleh model.

STUDIO SHODWE

# THANK YOU!

DATA ANALYSIS IS KEY TO BUSINESS GROWTH AND SUCCESS!

KELOMPOK 4