RANCANGAN UJIAN TENGAH SEMESTER (UTS)

BIG DATA & DATA MINING KELAS C

TUGAS KELOMPOK

Dosen Pengampu:

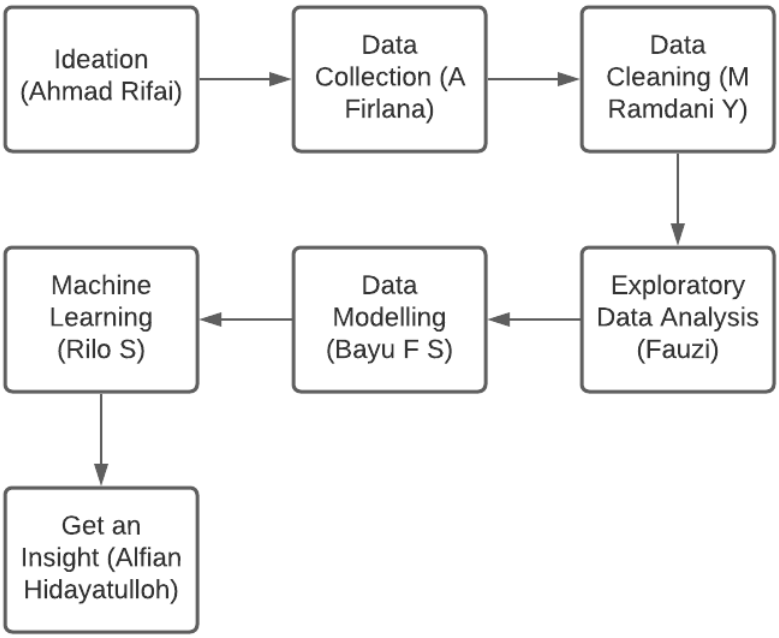
Dr. Enny Itje Sela, S.Si., M.Kom.

Disusun oleh:

(1) 5180411022 Ahmad Rifai
(2) 5180411040 Anandika Firlana
(3) 5180411059 M Milandika Ramdani Yunas
(4) 5180411137 Fauzi
(5) 5180411168 Bayu Fauzi Saputra
(6) 5180411351 Rilo Supriyatno
(7) 5180411382 Alfian Hidayatulloh

PROGRAM STUDI S1 INFORMATIKA
FAKULTAS SAINS & TEKNOLOGI
UNIVERSITAS TEKNOLOGI YOGYAKARTA
2021

A. Job Description of Each Students



B. Source of the Dataset

Dataset ini bersumber dari situs repositori UCI Machine Learning (Donald Bren School of Information and Computer Sciences, University of California) dengan alamat URL sebagai berikut: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

C. Screenshot of the Displayed Dataset

Berikut cuplikan lima baris pertama dari dataset kami.

```
df.show(5)
```

```
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|deposit|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
| 59|    admin.|married|secondary|     no|   2343|    yes|  no|unknown|  5|  may|    1042|       1|   -1|       0| unknown|    yes|
| 56|    admin.|married|secondary|     no|     45|     no|  no|unknown|  5|  may|    1467|       1|   -1|       0| unknown|    yes|
| 41|technician|married|secondary|     no|   1270|    yes|  no|unknown|  5|  may|    1389|       1|   -1|       0| unknown|    yes|
| 55|  services|married|secondary|     no|   2476|    yes|  no|unknown|  5|  may|     579|       1|   -1|       0| unknown|    yes|
| 54|    admin.|married| tertiary|     no|    184|     no|  no|unknown|  5|  may|     673|       2|   -1|       0| unknown|    yes|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
only showing top 5 rows
```

D. Description of the Dataset

Mengacu situs repositori UCI Machine Learning, dataset ini terdiri dari empat jenis. Kelompok kami menggunakan dataset dengan 17 atribut karena lebih mudah untuk diolah. Dataset ini merupakan kumpulan data nasabah yang dihimpun oleh sebuah perusahaan bank dari Portugis untuk keperluan strategi marketing.

Dataset ini pertama kali digunakan oleh sebuah penelitian berjudul “A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems” pada tahun 2014 oleh Sérgio Moro dkk. Penelitian tersebut membahas mengenai sebuah Sistem Pendukung Keputusan (SPK) yang dapat memprediksi apakah seorang nasabah akan terus menggunakan layanan perbankan atau tidak.

Adapun kumpulan data tersebut dihimpun oleh bank melalui proses pendaftaran (pengisian awal data) dan dilengkapi dengan proses marketing (yaitu menelpon secara manual kepada seluruh nasabah bank untuk menanyakan apakah akan terus menggunakan layanan atau tidak).

Untuk 17 atribut dari dataset akan dijelaskan oleh kelompok kami sebagi berikut.

(1) **age**: merupakan usia dari nasabah bank (usia dari 18 s/d 95 tahun).
(2) **job**: merupakan profesi atau pekerjaan dari nasabah bank (terdiri dari 12 jenis profesi meliputi 'admin.', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', dan 'unknown').
(3) **marital**: merupakan status perkawinan dari nasabah bank (terdiri.dari 4 jenis status meliputi 'divorced', 'married', 'single', dan 'unknown').
(4) **education**: merupakan tingkat pendidikan dari nasabah bank (terdiri dari 4 jenis tingkat meliputi 'primary', 'secondary', 'teritary', dan 'unknown').
(5) **default**: merupakan apakah nasabah bank memiliki kartu kredit (terdiri dari 3 status meliputi 'yes', 'no', dan 'unknown').
(6) **balance**: merupakan saldo tersedia dalam rekening dari nasabah bank (dalam satuan United State Dollar/USD).
(7) **housing**: merupakan apakah nasabah bank memiliki utang KPR (terdiri dari 3 status meliputi 'yes', 'no', dan 'unknown').
(8) **loan**: merupakan apakah nasabah bank memiliki utang pribadi (terdiri dari 3 status meliputi 'yes', 'no', dan 'unknown').
(9) **contact**: merupakan metode telepon dengan nasabah bank (terdiri dari 3 status meliputi 'cellular', 'telephone', dan 'unknown').
(10) **day**: merupakan hari terakhir dilakukannya telepon dengan nasabah bank (terdiri dari 5 hari meliputi 'mon', 'tue', 'wed', 'thu', dan 'fri').

(11) **month**: merupakan bulan terakhir dilakukannya telepon dengan nasabah bank (terdiri dari 12 bulan meliputi 'jan', 'feb', 'mar', 'apr', 'may', 'june', 'july', 'agt', 'okt', 'nov', dan 'dec').

(12) **duration**: merupakan durasi lamanya telepon dengan nasabah bank (dalam satuan detik).

(13) **campaign**: merupakan jumlah telepon yang dilakukan dengan nasabah bank.

(14) **pdays**: merupakan jumlah hari sejak telepon terakhir sampai dengan telepon terbaru (contoh: hari selasa ke hari senin, maka pdays = 6. Jika pdays = -1 artinya tidak pernah ada telepon terbaru dengan nasabah bank).

(15) **previous**: merupakan jumlah telepon sejak telepon terakhir sampai dengan telepon terbaru (dalam satu hari dimungkinkan untuk telepon lebih dari satu kali. Jika previous = 0 artinya tidak pernah ada telepon terbaru dengan nasabah bank).

(16) **poutcome**: merupakan hasil dari telepon dengan nasabah bank (terdiri dari 4 status meliputi 'failure', 'nonexistent', 'success', dan 'unknown').

(17) **deposit**: merupakan target prediksi apakah nasabah akan terus menggunakan layanan atau tidak (terdiri dari 2 status meliputi 'yes' dan 'no').

E. Result of the Data Processing (with screenshots)

i. Install JDK terlebih dulu sebagai syarat meng-install PySpark

```
!apt-get install openjdk-8-jdk # JDK 8 biar lebih stabil
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fonts-dejavu-core fonts-dejavu-extra libatk-wrapper-java
  libatk-wrapper-java-jni libgail-common libgail18 libgtk2.0-0 libgtk2.0-bin
  libgtk2.0-common libxxf86dga1 openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11-utils
Suggested packages:
  gvfs openjdk-8-demo openjdk-8-source visualvm icedtea-8-plugin libnss-mdns
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
  fonts-wqy-zenhei fonts-indic mesa-utils
The following NEW packages will be installed:
  fonts-dejavu-core fonts-dejavu-extra libatk-wrapper-java
  libatk-wrapper-java-jni libgail-common libgail18 libgtk2.0-0 libgtk2.0-bin
  libgtk2.0-common libxxf86dga1 openjdk-8-jdk openjdk-8-jdk-headless
  openjdk-8-jre openjdk-8-jre-headless x11-utils
0 upgraded, 15 newly installed, 0 to remove and 11 not upgraded.
Need to get 43.5 MB of archives.
After this operation, 163 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 libxxf86dga1 amd64 2:1.1.4-1 [13.7 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-dejavu-core all 2.37-1 [1,041 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-dejavu-extra all 2.37-1 [1,953 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic/main amd64 x11-utils amd64 7.7+3build1 [196 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic/main amd64 libatk-wrapper-java all 0.33.3-20ubuntu0.1 [34.7 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic/main amd64 libatk-wrapper-java-jni amd64 0.33.3-20ubuntu0.1 [28.3 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/main amd64 libgtk2.0-common all 2.24.32-1ubuntu1 [125 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic/main amd64 libgtk2.0-0 amd64 2.24.32-1ubuntu1 [1,769 kB]
Get:9 http://archive.ubuntu.com/ubuntu bionic/main amd64 libgail18 amd64 2.24.32-1ubuntu1 [14.2 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic/main amd64 libgail-common amd64 2.24.32-1ubuntu1 [112 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/main amd64 libgtk2.0-bin amd64 2.24.32-1ubuntu1 [7,536 B]
Get:12 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 openjdk-8-jre-headless amd64 8u282-b08-0ubuntu1~18.04 [28.2 MB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 openjdk-8-jre amd64 8u282-b08-0ubuntu1~18.04 [69.7 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 openjdk-8-jdk-headless amd64 8u282-b08-0ubuntu1~18.04 [8,267 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 openjdk-8-jdk amd64 8u282-b08-0ubuntu1~18.04 [1,630 kB]
Fetched 43.5 MB in 2s (19.4 MB/s)
Selecting previously unselected package libxxf86dga1:amd64.
(Reading database ... 149414 files and directories currently installed.)
Preparing to unpack .../00-libxxf86dga1_2%3a1.1.4-1_amd64.deb ...
Unpacking libxxf86dga1:amd64 (2:1.1.4-1) ...
```

ii. **Atur environment JDK agar PySpark bisa berjalan lancar**

```
import os # library OS berfungsi menjembatansi proses/tugas antara kodingan dengan sistem operasi
os.environ["JAVA_HOME"]="/usr/lib/jvm/java-8-openjdk-amd64" # path folder JDK bisa kita dapatkan dari output saat install JDK
!echo $JAVA_HOME # cek kembali apakah sudah di-set dengan benar
```

```
/usr/lib/jvm/java-8-openjdk-amd64
```

iii. **Saatnya meng-install PySpark**

```
!pip install pyspark
```

```
Collecting pyspark
  Downloading https://files.pythonhosted.org/packages/27/67/5158f846202d7f012d1c9ca21c3549a58fd3c6707ae8ee823adcaca6473c/pyspark-3.0.2.tar.gz (204.8MB)
     |████████████████████████████████| 204.8MB 69kB/s
Collecting py4j==0.10.9
  Downloading https://files.pythonhosted.org/packages/9e/b6/6a4fb90cd235dc8e265a6a2067f2a2c99f0d91787f06aca4bcf7c23f3f80/py4j-0.10.9-py2.py3-none-any.whl (198kB)
     |████████████████████████████████| 204kB 16.7MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.0.2-py2.py3-none-any.whl size=205186687 sha256=9284d8eb67b4e9bb4ceb561ede5646d9549157ac520bc19e6c52a0a69c258f7c
  Stored in directory: /root/.cache/pip/wheels/8b/09/da/c1f2859bcc86375dc972c5b6af4881b3603269bcc4c9be5d16
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9 pyspark-3.0.2
```

iv. **Selanjutnya, integrasikan Google Colab dengan Google Drive**

```
from google.colab import drive # tempat menyimpan dataset
drive.mount('/content/drive') # path folder default Google Drive di Google Colab
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

v. **Lalu kita buat SparkSession untuk memulai menjalankan PySpark**

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('Analisis Data Nasabah Bank').getOrCreate()
```

vi. **Setelah itu, kita lakukan read pada dataset dengan inferSchema dan header**

```
# "inferSchema=True" berfungsi agar kita bisa menampilkan informasi dari dataframe kita nanti
# "header=True" berfungsi agar baris pertama pada dataset diubah menjadi header dalam dataframe
df = spark.read.csv('/content/drive/MyDrive/Colab Notebooks/Datasets/dataset-nasabah-bank.csv',inferSchema=True,header=True)
```

vii. **Sebelum melakukan analisis, kita lihat dimensi dataset kita**

```
print((df.count(),len(df.columns))) # baris x kolom
```

```
(11162, 17)
```

viii. **Dan juga kita lihat Schema (informasi atribut) pada dataset kita**

```
df.printSchema() # nullable=true artinya nilai/value pada variabel
bisa kosong atau tanpa nilai
```

```
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)
```

ix. **Serta kita tampilkan cuplikan dari dataset yang telah kita read di awal**

```
df.show(5) # menampilkan 5 baris pertama
```

```
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
|age|       job|marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|deposit|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
| 59|    admin.|married|secondary|     no|   2343|    yes|  no|unknown|  5|  may|    1042|       1|   -1|       0| unknown|    yes|
| 56|    admin.|married|secondary|     no|     45|     no|  no|unknown|  5|  may|    1467|       1|   -1|       0| unknown|    yes|
| 41|technician|married|secondary|     no|   1270|    yes|  no|unknown|  5|  may|    1389|       1|   -1|       0| unknown|    yes|
| 55|  services|married|secondary|     no|   2476|    yes|  no|unknown|  5|  may|     579|       1|   -1|       0| unknown|    yes|
| 54|    admin.|married| tertiary|     no|    184|     no|  no|unknown|  5|  may|     673|       2|   -1|       0| unknown|    yes|
+---+----------+-------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+-------+
only showing top 5 rows
```

x. **Lalu kita lakukan data cleaning terhadap atribut yang kurang penting**

```
my_data = df.drop(*['default', 'contact', 'day','month']) # menghapus
kolom default, contact, day, dan month
my_data.columns # menampilkan sisa kolom setelah proses drop kolom
dijalankan
```

```
['age',
 'job',
 'marital',
 'education',
 'balance',
 'housing',
 'loan',
 'duration',
 'campaign',
 'pdays',
 'previous',
 'poutcome',
 'deposit']
```

xi. **Selanjutnya kita ulik lagi informasi lain dari dataset kita**

```
my_data.describe().show()
```

summary	age	job	marital	education	balance	housing	loan	duration	campaign	pdays	previous	poutcome	deposit
count	11162	11162	11162	11162	11162	11162	11162	11162	11162	11162	11162	11162	11162
mean	41.231947679627304	null	null	null	1528.5385235620856	null	null	371.99381831213043	2.508421429851281	51.33040673714388	0.8325568894463358	null	null
stddev	11.913369192215518	null	null	null	3225.413325946149	null	null	347.12838571630687	2.722077181661482 4	108.75828197197717	2.292007218670508	null	null
min	18	admin.	divorced	primary	-6847	no	no	2	1	-1	0	failure	no
max	95	unknown	single	unknown	81204	yes	yes	3881	63	854	58	unknown	yes

xii. **Dari informasi tersebut, kita ketahui bahwa ada NULL pada value tiap atribut, maka kita harus cleaning lagi melalui cara menghapus row data itu**

df.na.drop(subset=["job","marital","education","housing","loan","poutcome","deposit"]).show(truncate=False) # menghapus baris data null hanya pada kolom terpilih

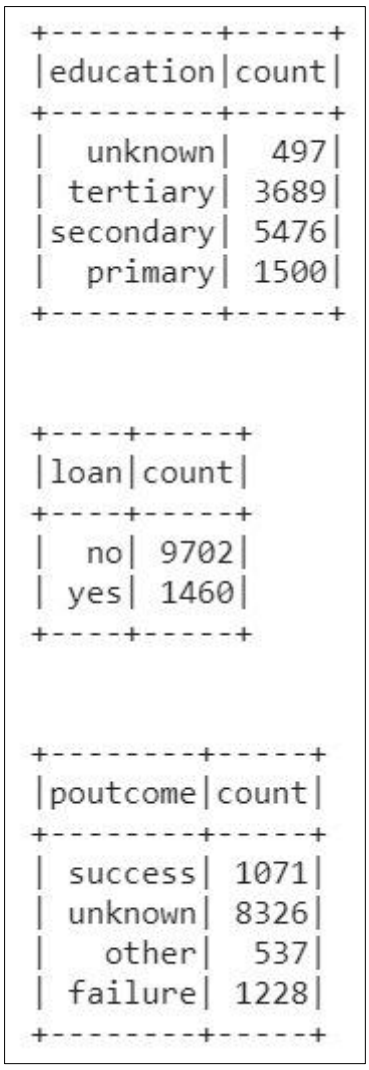
age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	deposit
59	admin.	married	secondary	no	2343	yes	no	unknown	5	may	1042	1	-1	0	unknown	yes
56	admin.	married	secondary	no	45	no	no	unknown	5	may	1467	1	-1	0	unknown	yes
41	technician	married	secondary	no	1270	yes	no	unknown	5	may	1389	1	-1	0	unknown	yes
55	services	married	secondary	no	2476	yes	no	unknown	5	may	579	1	-1	0	unknown	yes
54	admin.	married	tertiary	no	184	no	no	unknown	5	may	673	2	-1	0	unknown	yes
42	management	single	tertiary	no	0	yes	yes	unknown	5	may	562	2	-1	0	unknown	yes
56	management	married	tertiary	no	830	yes	yes	unknown	6	may	1201	1	-1	0	unknown	yes
60	retired	divorced	secondary	no	545	yes	no	unknown	6	may	1030	1	-1	0	unknown	yes
37	technician	married	secondary	no	1	yes	no	unknown	6	may	608	1	-1	0	unknown	yes
28	services	single	secondary	no	5090	yes	no	unknown	6	may	1297	3	-1	0	unknown	yes
38	admin.	single	secondary	no	100	yes	no	unknown	7	may	786	1	-1	0	unknown	yes
30	blue-collar	married	secondary	no	309	yes	no	unknown	7	may	1574	2	-1	0	unknown	yes
29	management	married	tertiary	no	199	yes	yes	unknown	7	may	1689	4	-1	0	unknown	yes
46	blue-collar	single	tertiary	no	460	yes	no	unknown	7	may	1102	2	-1	0	unknown	yes
31	technician	single	tertiary	no	703	yes	no	unknown	8	may	943	2	-1	0	unknown	yes
35	management	divorced	tertiary	no	3837	yes	no	unknown	8	may	1084	1	-1	0	unknown	yes
32	blue-collar	single	primary	no	611	yes	no	unknown	8	may	541	3	-1	0	unknown	yes
49	services	married	secondary	no	-8	yes	no	unknown	8	may	1119	1	-1	0	unknown	yes
41	admin.	married	secondary	no	55	yes	no	unknown	8	may	1120	2	-1	0	unknown	yes
49	admin.	divorced	secondary	no	168	yes	yes	unknown	8	may	513	1	-1	0	unknown	yes

only showing top 20 rows

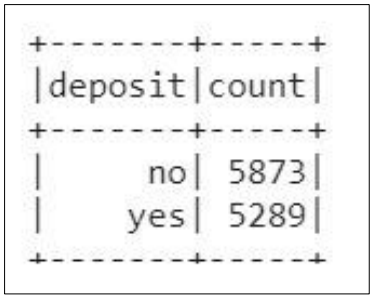
xiii. **Berikutnya kita ulik lebih dalam lagi dengan melihat informasi tiap atribut**

```
# Di sini, kita akan menghitung jumlah data (baris) pada setiap
kategori dari sebuah variabel data
my_data.groupBy('job').count().show()
print()
my_data.groupBy('marital').count().show()
print()
my_data.groupBy('education').count().show()
print()
my_data.groupBy('loan').count().show()
print()
my_data.groupBy('poutcome').count().show()
print()
my_data.groupBy('deposit').count().show()
```

```
+-------------+-----+
|          job|count|
+-------------+-----+
|   management| 2566|
|      retired|  778|
|      unknown|   70|
|self-employed|  405|
|      student|  360|
|  blue-collar| 1944|
| entrepreneur|  328|
|       admin.| 1334|
|   technician| 1823|
|     services|  923|
|    housemaid|  274|
|   unemployed|  357|
+-------------+-----+


+--------+-----+
| marital|count|
+--------+-----+
|divorced| 1293|
| married| 6351|
|  single| 3518|
+--------+-----+
```



```
+---------+-----+
|education|count|
+---------+-----+
|  unknown|  497|
| tertiary| 3689|
|secondary| 5476|
|  primary| 1500|
+---------+-----+


+----+-----+
|loan|count|
+----+-----+
|  no| 9702|
| yes| 1460|
+----+-----+


+--------+-----+
|poutcome|count|
+--------+-----+
| success| 1071|
| unknown| 8326|
|   other|  537|
| failure| 1228|
+--------+-----+
```



```
+-------+-----+
|deposit|count|
+-------+-----+
|     no| 5873|
|    yes| 5289|
+-------+-----+
```

xiv. **Akhirnya kita selesai melakukan Data Cleaning. Mulai dari sini, kita akan melakukan Exploratory Data Analysis**

```
from pyspark.ml.feature import StringIndexer, OneHotEncoder
# Membuat objek dari StringIndexer class dan menge-set kolom input & output
SI_job = StringIndexer(inputCol='job',outputCol='job_Index')
SI_marital = StringIndexer(inputCol='marital',outputCol='marital_Index')
```

```
SI_education =
StringIndexer(inputCol='education',outputCol='education_Index')
SI_housing =
StringIndexer(inputCol='housing',outputCol='housing_Index')
SI_loan = StringIndexer(inputCol='loan',outputCol='loan_Index')
SI_poutcome =
StringIndexer(inputCol='poutcome',outputCol='poutcome_Index')
SI_deposit =
StringIndexer(inputCol='deposit',outputCol='deposit_Index')
# Mentransformasi / mengubah data ke dalam bentuk yang baru untuk
mempermudah proses prediksi
my_data = SI_job.fit(my_data).transform(my_data)
my_data = SI_marital.fit(my_data).transform(my_data)
my_data = SI_education.fit(my_data).transform(my_data)
my_data = SI_housing.fit(my_data).transform(my_data)
my_data = SI_loan.fit(my_data).transform(my_data)
my_data = SI_poutcome.fit(my_data).transform(my_data)
my_data = SI_deposit.fit(my_data).transform(my_data)
```

xv. **Kode di atas digunakan untuk mentransformasi data ke dalam data numerik agar nantinya lebih mudah dalam proses modelling, dan inilah hasilnya**

```
my_data.select('job', 'job_Index', 'marital',
'marital_Index','housing','housing_Index','poutcome','poutcome_Index',
'deposit','deposit_Index').show(10)
```

job	job_Index	marital	marital_Index	housing	housing_Index	poutcome	poutcome_Index	deposit	deposit_Index
admin.	3.0	married	0.0	yes	1.0	unknown	0.0	yes	1.0
admin.	3.0	married	0.0	no	0.0	unknown	0.0	yes	1.0
technician	2.0	married	0.0	yes	1.0	unknown	0.0	yes	1.0
services	4.0	married	0.0	yes	1.0	unknown	0.0	yes	1.0
admin.	3.0	married	0.0	no	0.0	unknown	0.0	yes	1.0
management	0.0	single	1.0	yes	1.0	unknown	0.0	yes	1.0
management	0.0	married	0.0	yes	1.0	unknown	0.0	yes	1.0
retired	5.0	divorced	2.0	yes	1.0	unknown	0.0	yes	1.0
technician	2.0	married	0.0	yes	1.0	unknown	0.0	yes	1.0
services	4.0	single	1.0	yes	1.0	unknown	0.0	yes	1.0

only showing top 10 rows

xvi. **Dilanjutkan dengan proses transformasi ke dalam bentuk List sebelum masuk ke dalam proses Feature Selection**

```
# Buat objek dan set kolom input & output
OHE = OneHotEncoder(inputCols=['job_Index',
'marital_Index','education_Index','housing_Index','loan_Index','poutco
me_Index','deposit_Index'],outputCols=['job_OHE',
```

```
'marital_OHE','education_OHE','housing_OHE','loan_OHE','poutcome_OHE',
'deposit_OHE'])
# Proses transformasi data
my_data = OHE.fit(my_data).transform(my_data)
# Lihat hasil transformasi data
my_data.select('job', 'job_Index',
'job_OHE','education','education_Index','education_OHE').show(10)
```

```
+----------+---------+-------------+---------+---------------+-------------+
|       job|job_Index|      job_OHE|education|education_Index|education_OHE|
+----------+---------+-------------+---------+---------------+-------------+
|    admin.|      3.0|(11,[3],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|    admin.|      3.0|(11,[3],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|technician|      2.0|(11,[2],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|  services|      4.0|(11,[4],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|    admin.|      3.0|(11,[3],[1.0])| tertiary|            1.0|(3,[1],[1.0])|
|management|      0.0|(11,[0],[1.0])| tertiary|            1.0|(3,[1],[1.0])|
|management|      0.0|(11,[0],[1.0])| tertiary|            1.0|(3,[1],[1.0])|
|   retired|      5.0|(11,[5],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|technician|      2.0|(11,[2],[1.0])|secondary|            0.0|(3,[0],[1.0])|
|  services|      4.0|(11,[4],[1.0])|secondary|            0.0|(3,[0],[1.0])|
+----------+---------+-------------+---------+---------------+-------------+
only showing top 10 rows
```

xvii. **Nah, di sini kita transformasi lagi menjadi vektor sebelum masuk ke dalam proses Feature Selection**

```
from pyspark.ml.feature import VectorAssembler # men-transformasi list
menjadi vektor biner
# menge-set kolom input & output dari transformasi data
assembler = VectorAssembler(inputCols=['age',
                                       'job_Index',
                                       'marital_Index',
                                       'education_Index',
                                       'balance',
                                       'housing_Index',
                                       'loan_Index',
                                       'duration',
                                       'campaign',
                                       'pdays',
                                       'previous',
                                       'poutcome_Index',
                                       'job_OHE',
                                       'marital_OHE',
                                       'housing_OHE',
```

```
                                        'education_OHE',
                                        'loan_OHE',
                                        'poutcome_OHE'],
                              outputCol='features')
# Proses transformasi data
final_data = assembler.transform(my_data)
```

xviii. **Lalu seperti biasa, kita lihat cuplikan hasil transformasi data tadi**

final_data.select('features','deposit_Index').show(10)

```
+--------------------+-------------+
|            features|deposit_Index|
+--------------------+-------------+
|(33,[0,1,4,5,7,8,...|          1.0|
|(33,[0,1,4,7,8,9,...|          1.0|
|(33,[0,1,4,5,7,8,...|          1.0|
|(33,[0,1,4,5,7,8,...|          1.0|
|(33,[0,1,3,4,7,8,...|          1.0|
|(33,[0,2,3,5,6,7,...|          1.0|
|(33,[0,3,4,5,6,7,...|          1.0|
|(33,[0,1,2,4,5,7,...|          1.0|
|(33,[0,1,4,5,7,8,...|          1.0|
|(33,[0,1,2,4,5,7,...|          1.0|
+--------------------+-------------+
only showing top 10 rows
```

xix. **Kemudian hasi Feature Selection di atas, akan kita kemas menjadi Data Model sebelum masuk proses Training & Testing**

```
model_df = final_data.select(['features','deposit_Index'])
model_df = model_df.withColumnRenamed("deposit_Index","label") #
mengubah nama kolom supaya lebih mudah dimengerti
model_df.printSchema()
```

```
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = false)
```

xx. **Selanjutnya kita lakukan Split untuk membagi menjadi Data Training dan Data Testing sebelum masuk ke dalam proses Machine Learning**

```
training_df,test_df = model_df.randomSplit([0.75,0.25]) # 75% dari
data untuk training dan 25% dari data untuk testing
```

xxi. **Adapun algoritma yang kita gunakan adalah Logistic Regression karena dirasa lebih cocok pada dataset ini**

```
from pyspark.ml.classification import LogisticRegression # Prediksi menggunakan Metode Logistic Regression
log_reg = LogisticRegression().fit(training_df)
```

xxii. **Setelah proses Data Training & Data Training selesai, kita lihat akurasi hasil dari kedua proses tersebut**

```
lr_summary = log_reg.summary
lr_summary.accuracy # Akurasi secara keseluruhan = 79%. Hasil akurasi biasanya berkorelasi dengan konsep underfitting/overfitting.
# Referensi bacaan mengenai konsep underfitting/overfitting: https://s.id/yhXPu
```

```
0.7973215353342102
```

xxiii. **Terakhir, kita tampilkan hasil proses Data Testing untuk memprediksi Churn Rate pada Nasabah Bank**

```
predictions = log_reg.transform(test_df)
predictions.select('label','prediction').show(10) # menampilkan 10 baris data teratas
```

```
+-----+----------+
|label|prediction|
+-----+----------+
|  1.0|       1.0|
|  1.0|       0.0|
|  0.0|       0.0|
|  0.0|       0.0|
|  0.0|       0.0|
|  1.0|       0.0|
|  1.0|       1.0|
|  1.0|       1.0|
|  1.0|       0.0|
|  0.0|       0.0|
+-----+----------+
only showing top 10 rows
```

Nah, insight yang diperoleh dari prediksi data di atas ialah bahwa hasil prediksi masih belum akurat. Ini mungkin disebabkan oleh proses pelabelan data yang kurang maksimal atau mungkin terjadi underfitting saat melakukan proses Data Training & Data Testing.

F. Advantages of Our Result

Adapun beberapa kelebihan yang dimiliki oleh kelompok kami antara lain:

- Topik pengolahan data yang cukup bagus karena sesuai dengan kebutuhan industri bisnis, sehingga bisa menjadi bekal untuk meraih karir yang baik di masa depan
- Menggunakan algoritma Machine Learning bernama Logistic Regression yang mana sangat efektif dan efisien untuk prediksi Data Training & Data Testing.
- Akurasi hasil prediksi yang lumayan. Meskipun masih di bawah 90%, namun ini masih dapat ditingkatkan dengan melakukan pelabelan data secara lebih hati-hati.