#Spark

1. Install Java Development Kit (JDK)

Apache Spark requires Java. We'll install `openjdk-8-jdk`.

```
!apt-get update
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
```

```
 0% [Working]              Hit:1 https://cli.github.com/packages
stable InRelease
 0% [Connecting to archive.ubuntu.com (185.125.190.83)] [Connecting to
security.
Hit:2 https://cloud.r-project.org/bin/linux/ubuntu jammy-cran40/
InRelease
 0% [Connecting to archive.ubuntu.com (185.125.190.83)] [Connecting to
security. 0% [Waiting for headers] [Waiting for headers] [Waiting for
headers] [Connected
Hit:3 https://r2u.stat.illinois.edu/ubuntu jammy InRelease
Hit:4 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:6 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:8 https://ppa.launchpadcontent.net/deadsnakes/ppa/ubuntu jammy
InRelease
Hit:9 https://ppa.launchpadcontent.net/ubuntugis/ppa/ubuntu jammy
InRelease
Reading package lists... Done
W: Skipping acquire of configured file 'main/source/Sources' as
repository 'https://r2u.stat.illinois.edu/ubuntu jammy InRelease' does
not seem to provide it (sources.list entry misspelt?)
```

2. Download and Extract Apache Spark

We'll download a pre-built version of Spark and extract it to a convenient location.

```
!wget -q https://archive.apache.org/dist/spark/spark-3.5.1/spark-
3.5.1-bin-hadoop3.tgz
!tar xf spark-3.5.1-bin-hadoop3.tgz
```

3. Set Environment Variables

We need to tell our system where Java and Spark are located.

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.5.1-bin-hadoop3"
```

4. Install `findspark` and `pyspark`

`findspark` helps PySpark locate Spark, and `pyspark` is the Python API for Spark.

```
!pip install -q findspark pyspark
```

5. Initialize SparkSession

Now you can initialize Spark and start using PySpark.

```
import findspark
findspark.init()
from pyspark.sql import SparkSession

spark =
SparkSession.builder.master("local[*]").appName("ColabSpark").getOrCre
ate()
print("SparkSession created successfully!")
spark
```

```
SparkSession created successfully!

<pyspark.sql.session.SparkSession at 0x7fe8f457f230>
```

#Adjusting Loan Approval.

```
import pyspark
from pyspark.sql import SparkSession
import pyspark.sql.functions as F

import pandas as pd
import warnings
warnings.filterwarnings('ignore')

spark = SparkSession.builder.appName('loan_prediction').getOrCreate()

df = spark.read.csv('Loan_default.csv', header=True, sep=',',
inferSchema=True)
df.show(5)
```

```
+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
```

```
|    LoanID|Age|Income|LoanAmount|CreditScore|MonthsEmployed|
NumCreditLines|InterestRate|LoanTerm|DTIRatio|  Education|
EmploymentType|MaritalStatus|HasMortgage|HasDependents|LoanPurpose|
HasCoSigner|Default|
+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
|I38PQUQS96| 56| 85994|     50587|        520|            80|
4|       15.23|      36|    0.44| Bachelor's|     Full-time|
Divorced|        Yes|          Yes|      Other|        Yes|      0|
|HPSK72WA7R| 69| 50432|    124440|        458|            15|
1|        4.81|      60|    0.68|   Master's|     Full-time|
Married|         No|           No|      Other|        Yes|      0|
|C1OZ6DPJ8Y| 46| 84208|    129188|        451|            26|
3|       21.17|      24|    0.31|   Master's|    Unemployed|
Divorced|        Yes|          Yes|       Auto|         No|      1|
|V2KKSFM3UN| 32| 31713|     44799|        743|             0|
3|        7.07|      24|    0.23|High School|     Full-time|
Married|         No|           No|   Business|         No|      0|
|EY08JDHTZP| 60| 20437|      9139|        633|             8|
4|        6.51|      48|    0.73| Bachelor's|    Unemployed|
Divorced|         No|          Yes|       Auto|         No|      0|
+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
only showing top 5 rows


df.printSchema()

root
 |-- LoanID: string (nullable = true)
 |-- Age: integer (nullable = true)
 |-- Income: integer (nullable = true)
 |-- LoanAmount: integer (nullable = true)
 |-- CreditScore: integer (nullable = true)
 |-- MonthsEmployed: integer (nullable = true)
 |-- NumCreditLines: integer (nullable = true)
 |-- InterestRate: double (nullable = true)
 |-- LoanTerm: integer (nullable = true)
 |-- DTIRatio: double (nullable = true)
 |-- Education: string (nullable = true)
 |-- EmploymentType: string (nullable = true)
 |-- MaritalStatus: string (nullable = true)
 |-- HasMortgage: string (nullable = true)
 |-- HasDependents: string (nullable = true)
 |-- LoanPurpose: string (nullable = true)
 |-- HasCoSigner: string (nullable = true)
```

```
 |-- Default: integer (nullable = true)


df.dtypes

[('LoanID', 'string'),
 ('Age', 'int'),
 ('Income', 'int'),
 ('LoanAmount', 'int'),
 ('CreditScore', 'int'),
 ('MonthsEmployed', 'int'),
 ('NumCreditLines', 'int'),
 ('InterestRate', 'double'),
 ('LoanTerm', 'int'),
 ('DTIRatio', 'double'),
 ('Education', 'string'),
 ('EmploymentType', 'string'),
 ('MaritalStatus', 'string'),
 ('HasMortgage', 'string'),
 ('HasDependents', 'string'),
 ('LoanPurpose', 'string'),
 ('HasCoSigner', 'string'),
 ('Default', 'int')]
```

##Data Analysis

```
df.groupby('Default').count().show()

+-------+------+
|Default| count|
+-------+------+
|      1| 29653|
|      0|225694|
+-------+------+


from pyspark.sql import window

df_raw = (
    spark.read
    .option("header", True)
    .option("inferSchema", True)
    .csv("Loan_default.csv")
)

df_raw.printSchema()
df_raw.show(5, truncate=False)

root
 |-- LoanID: string (nullable = true)
 |-- Age: integer (nullable = true)
```

```
 |-- Income: integer (nullable = true)
 |-- LoanAmount: integer (nullable = true)
 |-- CreditScore: integer (nullable = true)
 |-- MonthsEmployed: integer (nullable = true)
 |-- NumCreditLines: integer (nullable = true)
 |-- InterestRate: double (nullable = true)
 |-- LoanTerm: integer (nullable = true)
 |-- DTIRatio: double (nullable = true)
 |-- Education: string (nullable = true)
 |-- EmploymentType: string (nullable = true)
 |-- MaritalStatus: string (nullable = true)
 |-- HasMortgage: string (nullable = true)
 |-- HasDependents: string (nullable = true)
 |-- LoanPurpose: string (nullable = true)
 |-- HasCoSigner: string (nullable = true)
 |-- Default: integer (nullable = true)

+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
|LoanID    |Age|Income|LoanAmount|CreditScore|MonthsEmployed|
NumCreditLines|InterestRate|LoanTerm|DTIRatio|Education  |
EmploymentType|MaritalStatus|HasMortgage|HasDependents|LoanPurpose|
HasCoSigner|Default|
+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
|I38PQUQS96|56 |85994 |50587     |520        |80            |4
|15.23       |36      |0.44    |Bachelor's |Full-time     |Divorced
|Yes         |Yes          |Other      |Yes        |0      |
|HPSK72WA7R|69 |50432 |124440    |458        |15            |1
|4.81        |60      |0.68    |Master's   |Full-time     |Married
|No          |No           |Other      |Yes        |0      |
|C1OZ6DPJ8Y|46 |84208 |129188    |451        |26            |3
|21.17       |24      |0.31    |Master's   |Unemployed    |Divorced
|Yes         |Yes          |Auto       |No         |1      |
|V2KKSFM3UN|32 |31713 |44799     |743        |0             |3
|7.07        |24      |0.23    |High School|Full-time     |Married
|No          |No           |Business   |No         |0      |
|EY08JDHTZP|60 |20437 |9139      |633        |8             |4
|6.51        |48      |0.73    |Bachelor's |Unemployed    |Divorced
|No          |Yes          |Auto       |No         |0      |
+----------+---+------+----------+-----------+--------------
+--------------+------------+--------+--------+-----------
+--------------+-------------+-----------+-------------+-----------
+-----------+-------+
```

```
only showing top 5 rows
```

```
df = (
    df_raw
    .withColumn("Age", F.col("Age").cast("int"))
    .withColumn("Income", F.col("Income").cast("int"))
    .withColumn("LoanAmount", F.col("LoanAmount").cast("int"))
    .withColumn("CreditScore", F.col("CreditScore").cast("int"))
    .withColumn("MonthsEmployed", F.col("MonthsEmployed").cast("int"))
    .withColumn("NumCreditLines", F.col("NumCreditLines").cast("int"))
    .withColumn("InterestRate", F.col("InterestRate").cast("double"))
    .withColumn("LoanTerm", F.col("LoanTerm").cast("int"))
    .withColumn("DTIRatio", F.col("DTIRatio").cast("double"))
    .withColumn("Default", F.col("Default").cast("int"))
)

# Basic sanity filters (tweak thresholds if needed)
df = df.filter(
    (F.col("LoanAmount") > 0) &
    (F.col("Income") > 0) &
    (F.col("CreditScore").between(300, 850)) &
    (F.col("DTIRatio").between(0.0, 1.0)) &
    (F.col("InterestRate") >= 0.0) &
    (F.col("LoanTerm") > 0) &
    (F.col("Default").isin([0, 1]))
)

# Optional: drop rows missing core fields
core = ["LoanAmount", "InterestRate", "DTIRatio", "CreditScore",
"Income", "Default"]
df = df.dropna(subset=core)

df.count()
```

```
255347
```

```
df = (
    df
    .withColumn(
        "CreditScoreBand",
        F.when(F.col("CreditScore") < 600, "<600")
         .when((F.col("CreditScore") >= 600) & (F.col("CreditScore")
<= 649), "600-649")
         .when((F.col("CreditScore") >= 650) & (F.col("CreditScore")
<= 699), "650-699")
         .when((F.col("CreditScore") >= 700) & (F.col("CreditScore")
<= 749), "700-749")
         .otherwise("750+")
```

```
    )
    .withColumn(
        "DTIBand",
        F.when(F.col("DTIRatio") < 0.20, "<0.20")
         .when((F.col("DTIRatio") >= 0.20) & (F.col("DTIRatio") <
0.35), "0.20-0.34")
         .when((F.col("DTIRatio") >= 0.35) & (F.col("DTIRatio") <
0.50), "0.35-0.49")
         .otherwise("0.50+")
    )
    .withColumn(
        "IncomeBand",
        F.when(F.col("Income") < 50000, "<50k")
         .when((F.col("Income") >= 50000) & (F.col("Income") <
100000), "50k-99k")
         .when((F.col("Income") >= 100000) & (F.col("Income") <
150000), "100k-149k")
         .otherwise("150k+")
    )
)

df.select("CreditScore", "CreditScoreBand", "DTIRatio", "DTIBand",
"Income", "IncomeBand").show(10, truncate=False)
```

```
+-----------+---------------+--------+---------+------+----------+
|CreditScore|CreditScoreBand|DTIRatio|DTIBand  |Income|IncomeBand|
+-----------+---------------+--------+---------+------+----------+
|520        |<600           |0.44    |0.35-0.49|85994 |50k-99k   |
|458        |<600           |0.68    |0.50+    |50432 |50k-99k   |
|451        |<600           |0.31    |0.20-0.34|84208 |50k-99k   |
|743        |700-749        |0.23    |0.20-0.34|31713 |<50k      |
|633        |600-649        |0.73    |0.50+    |20437 |<50k      |
|720        |700-749        |0.1     |<0.20    |90298 |50k-99k   |
|429        |<600           |0.16    |<0.20    |111188|100k-149k |
|531        |<600           |0.43    |0.35-0.49|126802|100k-149k |
|827        |750+           |0.2     |0.20-0.34|42053 |<50k      |
|480        |<600           |0.33    |0.20-0.34|132784|100k-149k |
+-----------+---------------+--------+---------+------+----------+
only showing top 10 rows
```

```
portfolio_overview = df.agg(
    F.count("*").alias("total_loans"),
    F.sum("LoanAmount").alias("total_exposure"),
    F.avg("Default").alias("default_rate"),
    F.avg("InterestRate").alias("avg_interest_rate"),
    F.avg("DTIRatio").alias("avg_dti"),
    F.avg("CreditScore").alias("avg_credit_score"),
    F.avg("Income").alias("avg_income")
).withColumn("default_rate", F.round("default_rate", 4)) \
```

```
 .withColumn("avg_interest_rate", F.round("avg_interest_rate", 4)) \
 .withColumn("avg_dti", F.round("avg_dti", 4)) \
 .withColumn("avg_credit_score", F.round("avg_credit_score", 2)) \
 .withColumn("avg_income", F.round("avg_income", 2))

portfolio_overview.show(truncate=False)
```

```
+-----------+--------------+------------+-----------------+-------
+----------------+----------+
|total_loans|total_exposure|default_rate|avg_interest_rate|avg_dti|
avg_credit_score|avg_income|
+-----------+--------------+------------+-----------------+-------
+----------------+----------+
|255347     |32576880572   |0.1161      |13.4928          |0.5002 |
574.26          |82499.3   |
+-----------+--------------+------------+-----------------+-------
+----------------+----------+
```

```
def segment_table(df, segment_col, min_count=100):
    out = (
        df.groupBy(segment_col)
          .agg(
              F.count("*").alias("n_loans"),
              F.sum("LoanAmount").alias("exposure"),
              F.avg("Default").alias("default_rate"),
              F.avg("InterestRate").alias("avg_interest_rate"),
              F.avg("DTIRatio").alias("avg_dti"),
              F.avg("CreditScore").alias("avg_credit_score"),
          )
          .withColumn("default_rate", F.round("default_rate", 4))
          .withColumn("avg_interest_rate",
F.round("avg_interest_rate", 4))
          .withColumn("expected_loss_proxy",
F.round(F.col("default_rate") * F.col("exposure"), 2))
          .filter(F.col("n_loans") >= min_count)
          .orderBy(F.col("expected_loss_proxy").desc())
    )
    return out

seg_purpose = segment_table(df, "LoanPurpose", min_count=50)
seg_employment = segment_table(df, "EmploymentType", min_count=50)
seg_cosigner = segment_table(df, "HasCoSigner", min_count=50)

seg_purpose.show(20, truncate=False)
seg_employment.show(20, truncate=False)
seg_cosigner.show(20, truncate=False)
```

```
+-----------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+
|LoanPurpose|n_loans|exposure  |default_rate|avg_interest_rate|avg_dti
|avg_credit_score |expected_loss_proxy|
+-----------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+
|Business   |51298  |6522120439|0.1233      |13.4791          |
0.5004335451674542|574.0621076845101|8.0417745013E8     |
|Auto       |50844  |6500807511|0.1188      |13.4672          |
0.5012202029738031|574.575918495791 |7.7229593231E8     |
|Education  |51005  |6510575194|0.1184      |13.5126          |
0.5013735908244298|573.2128418782472|7.7085210297E8     |
|Other      |50914  |6498135901|0.1179      |13.4778          |
0.4985940998546577|574.4078642416624|7.6613022273E8     |
|Home       |51286  |6545241527|0.1023      |13.5269          |
0.4994421479546087|575.0610108021682|6.6957820821E8     |
+-----------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+

+--------------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+
|EmploymentType|n_loans|exposure  |default_rate|avg_interest_rate|
avg_dti           |avg_credit_score |expected_loss_proxy|
+--------------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+
|Unemployed    |63824  |8174575251|0.1355      |13.5047          |
0.4999833918275285|574.873824893457 |1.10765494651E9    |
|Part-time     |64161  |8169601068|0.1197      |13.496           |
0.5005684138339472|574.0955253191191|9.7790124784E8     |
|Self-employed |63706  |8118482425|0.1146      |13.481           |
0.5000773867453601|574.680045835557 |9.3037808591E8     |
|Full-time     |63656  |8114221828|0.0946      |13.4893          |
0.5002169473419669|573.4073928616313|7.6760538493E8     |
+--------------+-------+----------+------------+-----------------
+------------------+-----------------+-------------------+

+-----------+-------+-----------+------------+-----------------
+------------------+-----------------+-------------------+
|HasCoSigner|n_loans|exposure   |default_rate|avg_interest_rate|
avg_dti           |avg_credit_score |expected_loss_proxy|
+-----------+-------+-----------+------------+-----------------
+------------------+-----------------+-------------------+
|No         |127646 |16301645747|0.1287      |13.5193          |
0.5001258167118413|574.702207668082 |2.09802180764E9    |
|Yes        |127701 |16275234825|0.1036      |13.4663          |
0.50029827487647  |573.8266732445321|1.68611432787E9    |
+-----------+-------+-----------+------------+-----------------
+------------------+-----------------+-------------------+
```

```
pricing_matrix = (
    df.groupBy("CreditScoreBand", "DTIBand")
      .agg(
          F.count("*").alias("n_loans"),
          F.sum("LoanAmount").alias("exposure"),
          F.avg("InterestRate").alias("avg_interest_rate"),
          F.avg("Default").alias("default_rate"),
      )
      .withColumn("avg_interest_rate", F.round("avg_interest_rate",
4))
      .withColumn("default_rate", F.round("default_rate", 4))
)

# Simple "mispricing flag": high default but not high interest
(relative to overall avg)
overall = df.agg(
    F.avg("InterestRate").alias("overall_avg_ir"),
    F.avg("Default").alias("overall_default_rate")
).collect()[0]

overall_avg_ir = float(overall["overall_avg_ir"])
overall_default_rate = float(overall["overall_default_rate"])

pricing_matrix_flagged = (
    pricing_matrix
    .withColumn("high_default", F.col("default_rate") >
F.lit(overall_default_rate))
    .withColumn("low_rate", F.col("avg_interest_rate") <
F.lit(overall_avg_ir))
    .withColumn("mispricing_flag", F.col("high_default") &
F.col("low_rate"))
    .orderBy(F.col("default_rate").desc(),
F.col("avg_interest_rate").asc())
)

pricing_matrix_flagged.show(50, truncate=False)
```

```
+--------------+---------+-------+----------+-----------------
+------------+------------+--------+---------------+
|CreditScoreBand|DTIBand  |n_loans|exposure  |avg_interest_rate|
default_rate|high_default|low_rate|mispricing_flag|
+--------------+---------+-------+----------+-----------------
+------------+------------+--------+---------------+
|<600           |0.50+    |70848  |9026344994|13.4686          |0.1289
|true        |true    |true           |
|<600           |0.35-0.49|26160  |3347078288|13.4525          |0.1244
|true        |true    |true           |
|<600           |0.20-0.34|26082  |3327874967|13.539           |0.1207
|true        |false   |false          |
|650-699        |0.50+    |11683  |1501376391|13.6239          |0.1191
```

```
|true         |false   |false          |
|600-649      |0.50+   |11673  |1504056035|13.462          |0.1163
|true         |true    |true           |
|600-649      |0.35-0.49|4439  |568832030 |13.5686         |0.1158
|false        |false   |false          |
|<600         |<0.20   |16393  |2081547447|13.5121         |0.1126
|false        |false   |false          |
|700-749      |0.50+   |11570  |1474174678|13.4652         |0.1119
|false        |true    |false          |
|650-699      |<0.20   |2946   |379478310 |13.6425         |0.1103
|false        |false   |false          |
|750+         |0.50+   |23523  |3002359015|13.497          |0.1071
|false        |false   |false          |
|600-649      |0.20-0.34|4436  |564178856 |13.4448         |0.1066
|false        |true    |false          |
|650-699      |0.35-0.49|4248  |536237527 |13.4004         |0.1057
|false        |true    |false          |
|700-749      |0.20-0.34|4227  |541096671 |13.5586         |0.0998
|false        |false   |false          |
|700-749      |0.35-0.49|4300  |544128960 |13.5278         |0.0993
|false        |false   |false          |
|750+         |0.35-0.49|8814  |1128012375|13.4967         |0.0988
|false        |false   |false          |
|650-699      |0.20-0.34|4274  |546434318 |13.4681         |0.0976
|false        |true    |false          |
|750+         |0.20-0.34|8746  |1110010238|13.6178         |0.0964
|false        |false   |false          |
|600-649      |<0.20   |2735   |349829830 |13.3441         |0.0962
|false        |true    |false          |
|700-749      |<0.20   |2759   |344521184 |13.5421         |0.0892
|false        |false   |false          |
|750+         |<0.20   |5491   |699308458 |13.3001         |0.0843
|false        |true    |false          |
+-------------+--------+-------+----------+----------------
+-----------+-----------+--------+--------------+
```

```
k = 0.20  # tweak; it's a proxy used consistently for
ranking/comparison

pricing_scored = (
    pricing_matrix
    .withColumn("risk_spread_proxy",
F.round(F.col("avg_interest_rate") - (F.col("default_rate") *
F.lit(k)), 4))
    .orderBy(F.col("risk_spread_proxy").asc())  # low score =
potentially underpriced risk
)

pricing_scored.show(50, truncate=False)
```

```
+---------------+---------+-------+----------+-----------------
+------------+-----------------+
|CreditScoreBand|DTIBand  |n_loans|exposure  |avg_interest_rate|
default_rate|risk_spread_proxy|
+---------------+---------+-------+----------+-----------------
+------------+-----------------+
|750+           |<0.20    |5491   |699308458 |13.3001          |0.0843
|13.2832        |
|600-649        |<0.20    |2735   |349829830 |13.3441          |0.0962
|13.3249        |
|650-699        |0.35-0.49|4248   |536237527 |13.4004          |0.1057
|13.3793        |
|600-649        |0.20-0.34|4436   |564178856 |13.4448          |0.1066
|13.4235        |
|<600           |0.35-0.49|26160  |3347078288|13.4525          |0.1244
|13.4276        |
|600-649        |0.50+    |11673  |1504056035|13.462           |0.1163
|13.4387        |
|<600           |0.50+    |70848  |9026344994|13.4686          |0.1289
|13.4428        |
|700-749        |0.50+    |11570  |1474174678|13.4652          |0.1119
|13.4428        |
|650-699        |0.20-0.34|4274   |546434318 |13.4681          |0.0976
|13.4486        |
|750+           |0.50+    |23523  |3002359015|13.497           |0.1071
|13.4756        |
|750+           |0.35-0.49|8814   |1128012375|13.4967          |0.0988
|13.4769        |
|<600           |<0.20    |16393  |2081547447|13.5121          |0.1126
|13.4896        |
|700-749        |0.35-0.49|4300   |544128960 |13.5278          |0.0993
|13.5079        |
|<600           |0.20-0.34|26082  |3327874967|13.539           |0.1207
|13.5149        |
|700-749        |<0.20    |2759   |344521184 |13.5421          |0.0892
|13.5243        |
|700-749        |0.20-0.34|4227   |541096671 |13.5586          |0.0998
|13.5386        |
|600-649        |0.35-0.49|4439   |568832030 |13.5686          |0.1158
|13.5454        |
|750+           |0.20-0.34|8746   |1110010238|13.6178          |0.0964
|13.5985        |
|650-699        |0.50+    |11683  |1501376391|13.6239          |0.1191
|13.6001        |
|650-699        |<0.20    |2946   |379478310 |13.6425          |0.1103
|13.6204        |
+---------------+---------+-------+----------+-----------------
+------------+-----------------+
```

```
from pyspark.sql import functions as F

# 1) Build the matrix
pricing_matrix = (
    df.groupBy("CreditScoreBand", "DTIBand")
      .agg(
          F.count("*").alias("n_loans"),
          F.sum("LoanAmount").alias("exposure"),
          F.avg("InterestRate").alias("avg_interest_rate"),
          F.avg("Default").alias("default_rate"),
      )
      .withColumn("avg_interest_rate", F.round("avg_interest_rate",
4))
      .withColumn("default_rate", F.round("default_rate", 4))
      .withColumn("exposure", F.round("exposure", 0))
)

# 2) Portfolio averages (benchmarks)
overall = df.agg(
    F.avg("InterestRate").alias("overall_avg_ir"),
    F.avg("Default").alias("overall_default_rate")
).collect()[0]

overall_avg_ir = float(overall["overall_avg_ir"])
overall_default_rate = float(overall["overall_default_rate"])

# 3) Add readable labels + BOTH underpriced and overpriced
pricing_matrix_labeled = (
    pricing_matrix
    # Plain English comparisons
    .withColumn(
        "risk_vs_portfolio",
        F.when(F.col("default_rate") > F.lit(overall_default_rate),
"Higher")
         .when(F.col("default_rate") < F.lit(overall_default_rate),
"Lower")
         .otherwise("Same as avg")
    )
    .withColumn(
        "rate_vs_portfolio",
        F.when(F.col("avg_interest_rate") > F.lit(overall_avg_ir),
"Higher")
         .when(F.col("avg_interest_rate") < F.lit(overall_avg_ir),
"Lower")
         .otherwise("Same as avg")
    )
    # Pricing status: underpriced / overpriced / fair
    .withColumn(
        "pricing_status",
        F.when(
```

```
            (F.col("default_rate") > F.lit(overall_default_rate)) &
            (F.col("avg_interest_rate") < F.lit(overall_avg_ir)),
            "UNDERPRICED"
        ).when(
            (F.col("default_rate") < F.lit(overall_default_rate)) &
            (F.col("avg_interest_rate") > F.lit(overall_avg_ir)),
            "OVERPRICED"
        ).otherwise("FAIR")
    )
    # Optional: a simple “severity” number (bigger = more extreme
mispricing)
    .withColumn(
        "mispricing_severity",
        F.round(
            (F.col("default_rate") - F.lit(overall_default_rate)) -
            (F.col("avg_interest_rate") - F.lit(overall_avg_ir)) /
F.lit(100.0),
            4
        )
    )
    .select(
        "CreditScoreBand", "DTIBand", "n_loans", "exposure",
        "avg_interest_rate", "default_rate",
        "risk_vs_portfolio", "rate_vs_portfolio", "pricing_status",
"mispricing_severity"
    )
)

# 4) Sort so it’s easy to read:
# Show UNDERPRICED first (most urgent), then OVERPRICED, then FAIR.
pricing_matrix_final = (
    pricing_matrix_labeled
    .withColumn(
        "sort_key",
        F.when(F.col("pricing_status").startswith("UNDERPRICED"),
F.lit(1))
         .when(F.col("pricing_status").startswith("OVERPRICED"),
F.lit(2))
         .otherwise(F.lit(3))
    )
    .orderBy(
        F.col("sort_key").asc(),
        F.col("default_rate").desc(),
        F.col("avg_interest_rate").asc()
    )
    .drop("sort_key")
)

pricing_matrix_final.show(50, truncate=False)
```

```
+--------------+---------+-------+----------+-----------------
+------------+-----------------+-----------------+-------------
+-------------------+
|CreditScoreBand|DTIBand  |n_loans|exposure  |avg_interest_rate|
default_rate|risk_vs_portfolio|rate_vs_portfolio|pricing_status|
mispricing_severity|
+--------------+---------+-------+----------+-----------------
+------------+-----------------+-----------------+-------------
+-------------------+
|<600          |0.50+    |70848  |9026344994|13.4686          |0.1289
|Higher          |Lower            |UNDERPRICED   |0.013
|
|<600          |0.35-0.49|26160  |3347078288|13.4525          |0.1244
|Higher          |Lower            |UNDERPRICED   |0.0087
|
|600-649       |0.50+    |11673  |1504056035|13.462           |0.1163
|Higher          |Lower            |UNDERPRICED   |5.0E-4
|
|600-649       |0.35-0.49|4439   |568832030 |13.5686          |0.1158
|Lower           |Higher           |OVERPRICED    |-0.0011
|
|<600          |<0.20    |16393  |2081547447|13.5121          |0.1126
|Lower           |Higher           |OVERPRICED    |-0.0037
|
|650-699       |<0.20    |2946   |379478310 |13.6425          |0.1103
|Lower           |Higher           |OVERPRICED    |-0.0073
|
|750+          |0.50+    |23523  |3002359015|13.497           |0.1071
|Lower           |Higher           |OVERPRICED    |-0.0091
|
|700-749       |0.20-0.34|4227   |541096671 |13.5586          |0.0998
|Lower           |Higher           |OVERPRICED    |-0.017
|
|700-749       |0.35-0.49|4300   |544128960 |13.5278          |0.0993
|Lower           |Higher           |OVERPRICED    |-0.0172
|
|750+          |0.35-0.49|8814   |1128012375|13.4967          |0.0988
|Lower           |Higher           |OVERPRICED    |-0.0174
|
|750+          |0.20-0.34|8746   |1110010238|13.6178          |0.0964
|Lower           |Higher           |OVERPRICED    |-0.021
|
|700-749       |<0.20    |2759   |344521184 |13.5421          |0.0892
|Lower           |Higher           |OVERPRICED    |-0.0274
|
|<600          |0.20-0.34|26082  |3327874967|13.539           |0.1207
|Higher          |Higher           |FAIR          |0.0041
|
|650-699       |0.50+    |11683  |1501376391|13.6239          |0.1191
|Higher          |Higher           |FAIR          |0.0017
```

```
|
|700-749        |0.50+    |11570  |1474174678|13.4652          |0.1119
|Lower            |Lower            |FAIR          |-0.004
|
|600-649        |0.20-0.34|4436   |564178856 |13.4448          |0.1066
|Lower            |Lower            |FAIR          |-0.009
|
|650-699        |0.35-0.49|4248   |536237527 |13.4004          |0.1057
|Lower            |Lower            |FAIR          |-0.0095
|
|650-699        |0.20-0.34|4274   |546434318 |13.4681          |0.0976
|Lower            |Lower            |FAIR          |-0.0183
|
|600-649        |<0.20    |2735   |349829830 |13.3441          |0.0962
|Lower            |Lower            |FAIR          |-0.0184
|
|750+           |<0.20    |5491   |699308458 |13.3001          |0.0843
|Lower            |Lower            |FAIR          |-0.0299
|
+---------------+---------+-------+----------+-----------------
+------------+-----------------+-----------------+--------------
+-------------------+
```

```
def scenario_metrics(df, scenario_name, approve_condition):
    approved = df.withColumn("Approved", F.when(approve_condition,
F.lit(1)).otherwise(F.lit(0)))

    # Metrics on approved loans
    metrics = approved.agg(
        F.avg("Approved").alias("approval_rate"),
        F.sum(F.when(F.col("Approved") == 1,
F.col("LoanAmount")).otherwise(F.lit(0))).alias("approved_exposure"),
        F.avg(F.when(F.col("Approved") == 1,
F.col("Default"))).alias("approved_default_rate")
    )

    # Expected loss proxy on approved = approved_default_rate *
approved_exposure
    metrics = (
        metrics
        .withColumn("scenario", F.lit(scenario_name))
        .withColumn("approved_default_rate",
F.round("approved_default_rate", 4))
        .withColumn("approval_rate", F.round("approval_rate", 4))
        .withColumn("approved_exposure", F.round("approved_exposure",
2))
        .withColumn("expected_loss_proxy",
F.round(F.col("approved_default_rate") * F.col("approved_exposure"),
2))
```

```
        .select("scenario", "approval_rate", "approved_exposure",
"approved_default_rate", "expected_loss_proxy")
    )
    return metrics

# Baseline: everyone approved
baseline = scenario_metrics(df, "Baseline_AllApproved", F.lit(True))

# Scenario A: CreditScore >= 680
scA = scenario_metrics(df, "A_CreditScore>=680", F.col("CreditScore")
>= 680)

# Scenario B: DTIRatio <= 0.35
scB = scenario_metrics(df, "B_DTI<=0.35", F.col("DTIRatio") <= 0.35)

# Scenario C: CreditScore>=660 AND DTIRatio<=0.40
scC = scenario_metrics(df, "C_Score>=660_AND_DTI<=0.40",
                       (F.col("CreditScore") >= 660) &
(F.col("DTIRatio") <= 0.40))

# Scenario D: If score < 640 require co-signer, else approve
# (Assumes HasCoSigner is "Yes"/"No"; adjust if it's 1/0)
scD_condition = (
    (F.col("CreditScore") >= 640) |
    ((F.col("CreditScore") < 640) & (F.col("HasCoSigner") == "Yes"))
)
scD = scenario_metrics(df, "D_LowScoreRequiresCosigner",
scD_condition)

scenario_compare =
baseline.unionByName(scA).unionByName(scB).unionByName(scC).unionByNam
e(scD)

scenario_compare.show(truncate=False)
print('What if Policy Results')
```

```
+-------------------------+-------------+-----------------
+--------------------+-------------------+
|scenario                 |approval_rate|approved_exposure|
approved_default_rate|expected_loss_proxy|
+-------------------------+-------------+-----------------
+--------------------+-------------------+
|Baseline_AllApproved     |1.0          |32576880572      |0.1161
|3.78217583441E9    |
|A_CreditScore>=680       |0.3078       |10013932616      |0.1032
|1.03343784597E9    |
|B_DTI<=0.35              |0.3178       |10342481236      |0.1083
|1.12009071786E9    |
|C_Score>=660_AND_DTI<=0.40|0.1313       |4267830775       |0.0946
|4.0373679132E8     |
```

```
|D_LowScoreRequiresCosigner|0.6912       |22507893488      |0.1068
|2.40384302452E9    |
+--------------------------+-------------+-----------------
+---------------------+------------------+

What if Policy Results
```

##What if compared to today

```
from pyspark.sql.window import Window

w = Window.orderBy(F.lit(1))

baseline_vals = scenario_compare.filter(F.col("scenario") ==
"Baseline_AllApproved").collect()[0]
base_exposure = float(baseline_vals["approved_exposure"])
base_loss = float(baseline_vals["expected_loss_proxy"])
base_approval = float(baseline_vals["approval_rate"])

scenario_compare_delta = (
    scenario_compare
    .withColumn("delta_exposure", F.round(F.col("approved_exposure") -
F.lit(base_exposure), 2))
    .withColumn("delta_expected_loss",
F.round(F.col("expected_loss_proxy") - F.lit(base_loss), 2))
    .withColumn("delta_approval_rate", F.round(F.col("approval_rate")
- F.lit(base_approval), 4))
    .orderBy(F.col("expected_loss_proxy").asc())
)

scenario_compare_delta.show(truncate=False)

+--------------------------+-------------+-----------------
+---------------------+------------------+---------------
+-------------------+-------------------+
|scenario                  |approval_rate|approved_exposure|
approved_default_rate|expected_loss_proxy|delta_exposure  |
delta_expected_loss|delta_approval_rate|
+--------------------------+-------------+-----------------
+---------------------+------------------+---------------
+-------------------+-------------------+
|C_Score>=660_AND_DTI<=0.40|0.1313       |4267830775       |0.0946
|4.0373679132E8     |-2.8309049797E10|-3.37843904309E9   |-0.8687
|
|A_CreditScore>=680        |0.3078       |10013932616      |0.1032
|1.03343784597E9    |-2.2562947956E10|-2.74873798844E9   |-0.6922
|
|B_DTI<=0.35               |0.3178       |10342481236      |0.1083
|1.12009071786E9    |-2.2234399336E10|-2.66208511655E9   |-0.6822
|
```

```
|D_LowScoreRequiresCosigner|0.6912       |22507893488      |0.1068
|2.40384302452E9    |-1.0068987084E10|-1.37833280989E9   |-0.3088
|
|Baseline_AllApproved      |1.0          |32576880572      |0.1161
|3.78217583441E9    |0.0              |0.0                |0.0
|
+--------------------------+-------------+-----------------
+---------------------+-------------------+----------------
+--------------------+--------------------+
```

“If we raised interest rates only for the borrowers we already know are underpriced, how much risk would that reduce?”

#Code

```
underpriced_segments = (
    pricing_matrix_labeled
    .filter(F.col("pricing_status").startswith("UNDERPRICED"))
    .select("CreditScoreBand", "DTIBand")
)

underpriced_segments.show(truncate=False)
```

```
+---------------+---------+
|CreditScoreBand|DTIBand  |
+---------------+---------+
|<600           |0.50+    |
|600-649        |0.50+    |
|<600           |0.35-0.49|
+---------------+---------+
```

```
from pyspark.sql import functions as F

underpriced_segments = underpriced_segments.select(
    "CreditScoreBand", "DTIBand"
).distinct().withColumn("underpriced_marker", F.lit(1))

df_with_pricing_flag = (
    df.join(underpriced_segments, on=["CreditScoreBand", "DTIBand"],
how="left")
      .withColumn(
          "is_underpriced",
          F.when(F.col("underpriced_marker").isNotNull(),
F.lit(1)).otherwise(F.lit(0))
      )
      .drop("underpriced_marker")
```

```
)

# sanity check
df_with_pricing_flag.groupBy("is_underpriced").count().show()


+--------------+------+
|is_underpriced| count|
+--------------+------+
|             1|108681|
|             0|146666|
+--------------+------+


df_price_plus_1 = (
    df_with_pricing_flag
    .withColumn(
        "AdjustedInterestRate",
        F.when(
            F.col("is_underpriced") == 1,
            F.col("InterestRate") + F.lit(1.0)
        ).otherwise(F.col("InterestRate"))
    )
)

pricing_plus_1 = (
    df_price_plus_1
    .groupBy("CreditScoreBand", "DTIBand")
    .agg(
        F.avg("AdjustedInterestRate").alias("avg_adj_interest_rate"),
        F.avg("Default").alias("default_rate")
    )
    .withColumn("avg_adj_interest_rate",
F.round("avg_adj_interest_rate", 4))
)

pricing_plus_1 = (
    pricing_plus_1
    .withColumn(
        "new_risk_spread_proxy",
        F.round(
            F.col("avg_adj_interest_rate") - (F.col("default_rate") *
F.lit(k)),
            4
        )
    )
)

pricing_comparison = (
    pricing_scored
    .join(
```

```
        pricing_plus_1.select("CreditScoreBand", "DTIBand",
"new_risk_spread_proxy"),
        on=["CreditScoreBand", "DTIBand"],
        how="left"
    )
    .withColumn(
        "delta_risk_spread",
        F.round(
            F.col("new_risk_spread_proxy") -
F.col("risk_spread_proxy"),
            4
        )
    )
    .orderBy(F.col("delta_risk_spread").desc())
)

pricing_comparison.show(50, truncate=False)

print('Counterfactual Pricing Test')
```

```
+---------------+---------+-------+----------+-----------------
+------------+-----------------+--------------------
+-----------------+
|CreditScoreBand|DTIBand  |n_loans|exposure  |avg_interest_rate|
default_rate|risk_spread_proxy|new_risk_spread_proxy|
delta_risk_spread|
+---------------+---------+-------+----------+-----------------
+------------+-----------------+--------------------
+-----------------+
|<600           |0.50+    |70848  |9026344994|13.4686          |0.1289
|13.4428          |14.4428              |1.0                 |
|600-649        |0.50+    |11673  |1504056035|13.462           |0.1163
|13.4387          |14.4387              |1.0                 |
|<600           |0.35-0.49|26160  |3347078288|13.4525          |0.1244
|13.4276          |14.4276              |1.0                 |
|700-749        |<0.20    |2759   |344521184 |13.5421          |0.0892
|13.5243          |13.5243              |0.0                 |
|650-699        |0.50+    |11683  |1501376391|13.6239          |0.1191
|13.6001          |13.6001              |0.0                 |
|650-699        |0.35-0.49|4248   |536237527 |13.4004          |0.1057
|13.3793          |13.3793              |0.0                 |
|<600           |0.20-0.34|26082  |3327874967|13.539           |0.1207
|13.5149          |13.5149              |0.0                 |
|750+           |0.20-0.34|8746   |1110010238|13.6178          |0.0964
|13.5985          |13.5985              |0.0                 |
|750+           |0.35-0.49|8814   |1128012375|13.4967          |0.0988
|13.4769          |13.4769              |0.0                 |
|700-749        |0.20-0.34|4227   |541096671 |13.5586          |0.0998
|13.5386          |13.5386              |0.0                 |
|600-649        |0.35-0.49|4439   |568832030 |13.5686          |0.1158
```

```
|13.5454          |13.5454              |0.0               |
|600-649        |0.20-0.34|4436   |564178856 |13.4448          |0.1066
|13.4235          |13.4235              |0.0               |
|750+           |0.50+    |23523  |3002359015|13.497           |0.1071
|13.4756          |13.4756              |0.0               |
|700-749        |0.35-0.49|4300   |544128960 |13.5278          |0.0993
|13.5079          |13.5079              |0.0               |
|700-749        |0.50+    |11570  |1474174678|13.4652          |0.1119
|13.4428          |13.4428              |0.0               |
|650-699        |0.20-0.34|4274   |546434318 |13.4681          |0.0976
|13.4486          |13.4486              |0.0               |
|750+           |<0.20    |5491   |699308458 |13.3001          |0.0843
|13.2832          |13.2832              |0.0               |
|650-699        |<0.20    |2946   |379478310 |13.6425          |0.1103
|13.6204          |13.6204              |0.0               |
|<600           |<0.20    |16393  |2081547447|13.5121          |0.1126
|13.4896          |13.4896              |0.0               |
|600-649        |<0.20    |2735   |349829830 |13.3441          |0.0962
|13.3249          |13.3249              |0.0               |
+---------------+---------+-------+----------+-----------------
+------------+-----------------+--------------------
+-----------------+

Counterfactual Pricing Test
```

A targeted 1% interest rate increase applied only to underpriced borrower segments uniformly improved pricing alignment across those segments without impacting the rest of the portfolio.