```
!pip install pyspark
```

```
Collecting pyspark
  Downloading pyspark-3.5.1.tar.gz (317.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 317.0/317.0 MB 6.4 MB/s eta 0:00:0000:0100:01
  Preparing metadata (setup.py) ... done
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)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 200.5/200.5 kB 33.8 MB/s eta 0:00:00
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.1-py2.py3-none-any.whl size=317488491 sha256=c06a9f8bbd38c60f5c04e74ea864f73bacae878e0
  Stored in directory: /home/sagemaker-user/.cache/pip/wheels/80/1d/60/2c256ed38dddce2fdd93be545214a63e02fbd8d74fb0b7f3a6
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.7 pyspark-3.5.1
```

```
!sudo apt-get update
!sudo apt-get install -y openjdk-8-jdk
```

```
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1756 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2037 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [1077 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1371 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2265 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [51.1 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2339 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [81.0 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [31.9 kB]
Fetched 11.3 MB in 2s (7440 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  adwaita-icon-theme alsa-topology-conf alsa-ucm-conf at-spi2-core
  ca-certificates ca-certificates-java dbus dbus-user-session
  dconf-gsettings-backend dconf-service dmsetup fontconfig fontconfig-config
  fonts-dejavu-core fonts-dejavu-extra gir1.2-glib-2.0
  gsettings-desktop-schemas gtk-update-icon-cache hicolor-icon-theme
  humanity-icon-theme java-common libapparmor1 libargon2-1 libasound2
  libasound2-data libasyncns0 libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0
  libavahi-client3 libavahi-common-data libavahi-common3 libcairo-gobject2
  libcairo2 libcryptsetup12 libcups2 libdatrie1 libdbus-1-3 libdconf1
  libdeflate0 libdevmapper1.02.1 libdrm-amdgpu1 libdrm-common libdrm-intel1
  libdrm-nouveau2 libdrm-radeon1 libdrm2 libelf1 libflac8 libfontconfig1
  libfontenc1 libfreetype6 libfribidi0 libgail-common libgail18
  libgdk-pixbuf-2.0-0 libgdk-pixbuf2.0-bin libgdk-pixbuf2.0-common libgif7
  libgirepository-1.0-1 libgl1 libgl1-amber-dri libgl1-mesa-dri
  libgl1-mesa-glx libglapi-mesa libglib2.0-0 libglib2.0-data libglvnd0
  libglx-mesa0 libglx0 libgraphite2-3 libgtk2.0-0 libgtk2.0-bin
  libgtk2.0-common libharfbuzz0b libice-dev libice6 libip4tc2 libjbig0
  libjpeg-turbo8 libjpeg8 libkmod2 liblcms2-2 libllvm15 libmpdec3 libnspr4
  libnss-systemd libnss3 libogg0 libopus0 libpam-systemd libpango-1.0-0
  libpangocairo-1.0-0 libpangoft2-1.0-0 libpciaccess0 libpcsclite1
  libpixman-1-0 libpng16-16 libpthread-stubs0-dev libpulse0 libpython3-stdlib
  libpython3.10-minimal libpython3.10-stdlib libreadline8 librsvg2-2
  librsvg2-common libsensors-config libsensors5 libsm-dev libsm6 libsndfile1
  libsqlite3-0 libthai-data libthai0 libtiff5 libvorbis0a libvorbisenc2
  libwebp7 libx11-6 libx11-data libx11-dev libx11-xcb1 libxau-dev libxau6
  libxaw7 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-present0
  libxcb-randr0 libxcb-render0 libxcb-shape0 libxcb-shm0 libxcb-sync1
  libxcb-xfixes0 libxcb1 libxcb1-dev libxcomposite1 libxcursor1 libxdamage1
  libxdmcp-dev libxdmcp6 libxext6 libxfixes3 libxft2 libxi6 libxinerama1
  libxkbfile1 libxmu6 libxmuu1 libxpm4 libxrandr2 libxrender1 libxshmfence1
  libxt-dev libxt6 libxtst6 libxv1 libxxf86dga1 libxxf86vm1 media-types
  networkd-dispatcher openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless openssl python3 python3-dbus python3-gi
  python3-minimal python3.10 python3.10-minimal readline-common
  session-migration shared-mime-info systemd systemd-sysv systemd-timesyncd
  ubuntu-mono ucf x11-common x11-utils x11proto-dev xdg-user-dirs
  xorg-sgml-doctools xtrans-dev
```

```
import os
os.environ["JAVA_HOME"] = "/usr"
```

```
import tempfile
import boto3
from pyspark.sql import SparkSession
```

```
!pip install boto3
```

```
Requirement already satisfied: boto3 in /opt/conda/lib/python3.10/site-packages (1.34.51)
Requirement already satisfied: botocore<1.35.0,>=1.34.51 in /opt/conda/lib/python3.10/site-packages (from boto3) (1.34.51)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /opt/conda/lib/python3.10/site-packages (from boto3) (1.0.1)
Requirement already satisfied: s3transfer<0.11.0,>=0.10.0 in /opt/conda/lib/python3.10/site-packages (from boto3) (0.10.1)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/lib/python3.10/site-packages (from botocore<1.35.0,>=1.34.51->b
Requirement already satisfied: urllib3<2.1,>=1.25.4 in /opt/conda/lib/python3.10/site-packages (from botocore<1.35.0,>=1.34.51->boto3) (
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.35.0,>=
```

```
# Initialize a SparkSession
spark = SparkSession.builder \
    .appName("Fund investment Analysis") \
    .getOrCreate()

# Initialize Boto3 S3 client
s3 = boto3.client('s3')
# Define S3 bucket name and key
bucket_name = 'final-project-19'
file_key = 'inputFiles/mapping.csv'
# Download the data from S3 to memory
# Download the data from S3 to a temporary file
temp_file = tempfile.NamedTemporaryFile(delete=False)
with open(temp_file.name, 'wb') as f:
    s3.download_fileobj(bucket_name, file_key, f)

file_key2 = 'inputFiles/rounds2.csv'
# Download the data from S3 to memory
# Download the data from S3 to a temporary file
temp_file2 = tempfile.NamedTemporaryFile(delete=False)
with open(temp_file2.name, 'wb') as f:
    s3.download_fileobj(bucket_name, file_key2, f)
```

```
# Define S3 bucket and file keys
bucket_name = "final-project-19"
funding_rounds_file_key = "inputFiles/rounds2.csv"
mapping_file_key = "inputFiles/mapping.csv"

# Read data into PySpark DataFrames
#funding_rounds_df = spark.read.csv(funding_rounds_local_path, header=True, inferSchema=True)
mapping_df = spark.read.csv(temp_file.name, header=True, inferSchema=True)
mapping_df.show()
```

```
+--------------------+-------------------+------+-------------------------+-------------+------+-------------+-------------------------
|       category_list|Automotive & Sports|Blanks|Cleantech / Semiconductors|Entertainment|Health|Manufacturing|News, Search and Messagin
+--------------------+-------------------+------+-------------------------+-------------+------+-------------+-------------------------
|                NULL|                  0|     1|                        0|            0|     0|            0|
|                  3D|                  0|     0|                        0|            0|     0|            1|
|         3D Printing|                  0|     0|                        0|            0|     0|            1|
|       3D Technology|                  0|     0|                        0|            0|     0|            1|
|          Accounting|                  0|     0|                        0|            0|     0|            0|
|    Active Lifestyle|                  0|     0|                        0|            0|     1|            0|
|        Ad Targeting|                  0|     0|                        0|            0|     0|            0|
|  Advanced Materials|                  0|     0|                        0|            0|     0|            1|
|    Adventure Travel|                  1|     0|                        0|            0|     0|            0|
|         Advertising|                  0|     0|                        0|            0|     0|            0|
|Advertising Excha...|                  0|     0|                        0|            0|     0|            0|
|Advertising Networks|                  0|     0|                        0|            0|     0|            0|
|Advertising Platf...|                  0|     0|                        0|            0|     0|            0|
|              Advice|                   0|     0|                         0|           0|     0|            0|
|           Aerospace|                   1|     0|                         0|           0|     0|            0|
|         Agriculture|                   0|     0|                         0|           0|     0|            1|
|Air Pollution Con...|                   0|     0|                         1|           0|     0|            0|
|          Algorithms|                   0|     0|                         0|           0|     0|            0|
|         All Markets|                   0|     0|                         0|           0|     0|            0|
|        All Students|                   0|     0|                         0|           0|     0|            0|
+--------------------+-------------------+------+-------------------------+------------+------+------------+-------------------------
only showing top 20 rows
```

```
funding_df = spark.read.csv(temp_file2.name, header=True, inferSchema=True)
funding_df.show()
```

```
+--------------------+----------------------+-------------------+------------------+----------+-----------------+
|   company_permalink|funding_round_permalink| funding_round_type|funding_round_code| funded_at|raised_amount_usd|
+--------------------+----------------------+-------------------+------------------+----------+-----------------+
| /organization/-fame|   /funding-round/9a...|            venture|                 B|  5/1/2015|         10000000|
|/ORGANIZATION/-QO...|   /funding-round/22...|            venture|                 A|14-10-2014|             NULL|
|/organization/-qo...|   /funding-round/b4...|               seed|              NULL|  1/3/2014|           700000|
|/ORGANIZATION/-TH...|   /funding-round/65...|            venture|                 B|30-01-2014|          3406878|
|/organization/0-6...|   /funding-round/57...|            venture|                 A|19-03-2008|          2000000|
|/ORGANIZATION/004...|   /funding-round/12...|            venture|              NULL|24-07-2014|             NULL|
|/organization/01g...|   /funding-round/7d...|         undisclosed|              NULL|  1/7/2014|            41250|
|/ORGANIZATION/0ND...|   /funding-round/2b...|               seed|              NULL| 11/9/2009|            43360|
|/organization/0nd...|   /funding-round/95...|            venture|              NULL|21-12-2009|           719491|
|/ORGANIZATION/0XDATA|   /funding-round/38...|               seed|              NULL|22-05-2013|          3000000|
|/organization/0xdata|   /funding-round/3b...|            venture|                 B| 9/11/2015|         20000000|
|/ORGANIZATION/0XDATA|   /funding-round/ae...|            venture|              NULL|  3/1/2013|          1700000|
|/organization/0xdata|   /funding-round/e1...|            venture|                 A|19-07-2014|          8900000|
|     /ORGANIZATION/1|   /funding-round/03...|               seed|              NULL|  5/2/2014|           150000|
|     /organization/1|   /funding-round/5d...|         undisclosed|              NULL|  5/2/2013|             NULL|
|     /ORGANIZATION/1|   /funding-round/e8...|               seed|              NULL|20-07-2011|          1000050|
|/organization/1-2...|   /funding-round/6d...|               seed|              NULL|18-02-2013|            40000|
|/ORGANIZATION/1-4...|   /funding-round/e9...|equity_crowdfunding|              NULL|21-04-2013|             NULL|
|/organization/1-6...|   /funding-round/83...|equity_crowdfunding|              NULL|22-01-2014|             NULL|
|/ORGANIZATION/1-8...|   /funding-round/52...|         undisclosed|              NULL|19-08-2010|             NULL|
+--------------------+----------------------+-------------------+------------------+----------+-----------------+
only showing top 20 rows
```

```
# Count unique companies in funding_rounds
unique_companies = funding_df.select("company_permalink").distinct().count()
print("\nNumber of unique companies in funding_rounds:", unique_companies)
```

```

Number of unique companies in funding_rounds: 90247
```

```
from pyspark.sql.functions import udf, col
from pyspark.sql.types import IntegerType
import datetime
```

FEATURE ENGINEERING

```
from pyspark.sql.functions import col, sum as spark_sum, count
# Feature Engineering for Investment Rounds Data
funding_df = funding_df.groupBy("company_permalink") \
    .agg(spark_sum("raised_amount_usd").alias("total_raised_amount_usd"), count("*").alias("num_rounds"))

# Calculate average raised amount per funding round
funding_df = funding_df.withColumn("average_raised_amount_usd", col("total_raised_amount_usd") / col("num_rounds"))

# Feature Engineering for Mapping Data (if applicable)
mapping_df = mapping_df.withColumnRenamed("old_column_name", "new_column_name")

# Show the first few rows of each DataFrame
print("Investment Rounds Data:")
funding_df.show()

print("Mapping Data:")
mapping_df.show()
```

```
Investment Rounds Data:
+--------------------+----------------------+----------+------------------------+
|   company_permalink|total_raised_amount_usd|num_rounds|average_raised_amount_usd|
+--------------------+----------------------+----------+------------------------+
|  /organization/1lay|                  2000|         1|                  2000.0|
|/organization/24p...|                 50000|         1|                 50000.0|
|/organization/2mo...|                250000|         1|                250000.0|
|/ORGANIZATION/5-M...|               3500000|         1|               3500000.0|
|  /organization/5min|              12500000|         2|               6250000.0|
|/ORGANIZATION/7TM...|              22000000|         1|                  2.2E7|
|/ORGANIZATION/ABL...|               3000000|         1|               3000000.0|
|/organization/abp...|                560000|         1|                560000.0|
|/ORGANIZATION/ACA...|               2338650|         1|               2338650.0|
|/organization/aci...|                  NULL|         1|                    NULL|
|/organization/acl...|              21000000|         1|                  2.1E7|
|/organization/ada...|              13740640|         3|       4580213.333333333|
|/organization/aer...|              27000000|         2|                 1.35E7|
|/ORGANIZATION/AGE...|                 50311|         1|                 50311.0|
|/ORGANIZATION/AIR...|              28525779|         6|               4754296.5|
|/ORGANIZATION/ALP...|                  NULL|         1|                    NULL|
|/organization/alt...|              14000000|         2|               7000000.0|
|/ORGANIZATION/AMOBEE|              38000000|         3|    1.2666666666666666E7|
|/ORGANIZATION/AN-...|              90000000|         1|                  9.0E7|
|/ORGANIZATION/ANO...|               5600000|         1|               5600000.0|
+--------------------+----------------------+----------+------------------------+
only showing top 20 rows

Mapping Data:
+---------------------+-------------------+------+-------------------------+-------------+------+-------------+------------------------
|mapping_category_list|Automotive & Sports|Blanks|Cleantech / Semiconductors|Entertainment|Health|Manufacturing|News, Search and Messagi
+---------------------+-------------------+------+-------------------------+-------------+------+-------------+------------------------
|                 NULL|                  0|     1|                        0|            0|     0|            0|
|                   3D|                  0|     0|                        0|            0|     0|            1|
|          3D Printing|                  0|     0|                        0|            0|     0|            1|
|        3D Technology|                  0|     0|                        0|            0|     0|            1|
|           Accounting|                  0|     0|                        0|            0|     0|            0|
|     Active Lifestyle|                  0|     0|                        0|            0|     1|            0|
|         Ad Targeting|                  0|     0|                        0|            0|     0|            0|
|   Advanced Materials|                  0|     0|                        0|            0|     0|            1|
|     Adventure Travel|                  1|     0|                        0|            0|     0|            0|
|          Advertising|                  0|     0|                        0|            0|     0|            0|
| Advertising Excha...|                  0|     0|                        0|            0|     0|            0|
| Advertising Networks|                  0|     0|                        0|            0|     0|            0|
| Advertising Platf...|                  0|     0|                        0|            0|     0|            0|
|               Advice|                  0|     0|                        0|            0|     0|            0|
|            Aerospace|                  1|     0|                        0|            0|     0|            0|
|          Agriculture|                  0|     0|                        0|            0|     0|            1|
| Air Pollution Con...|                  0|     0|                        1|            0|     0|            0|
|           Algorithms|                  0|     0|                        0|            0|     0|            0|
|          All Markets|                  0|     0|                        0|            0|     0|            0|
|         All Students|                  0|     0|                        0|            0|     0|            0|
+---------------------+-------------------+------+-------------------------+-------------+------+-------------+------------------------
only showing top 20 rows
```

```
mapping_df.columns
```

```
['mapping_category_list',
 'Automotive & Sports',
 'Blanks',
 'Cleantech / Semiconductors',
 'Entertainment',
 'Health',
 'Manufacturing',
 'News, Search and Messaging',
 'Others',
 'Social, Finance, Analytics, Advertising']
```

```
funding_df.columns
```

```
['company_permalink',
 'total_raised_amount_usd',
 'num_rounds',
 'average_raised_amount_usd']
```

```
funding_df.columns
```

```
['company_permalink',
 'total_raised_amount_usd',
 'num_rounds',
 'average_raised_amount_usd']
```

RISK ASSESSMENT AND ML

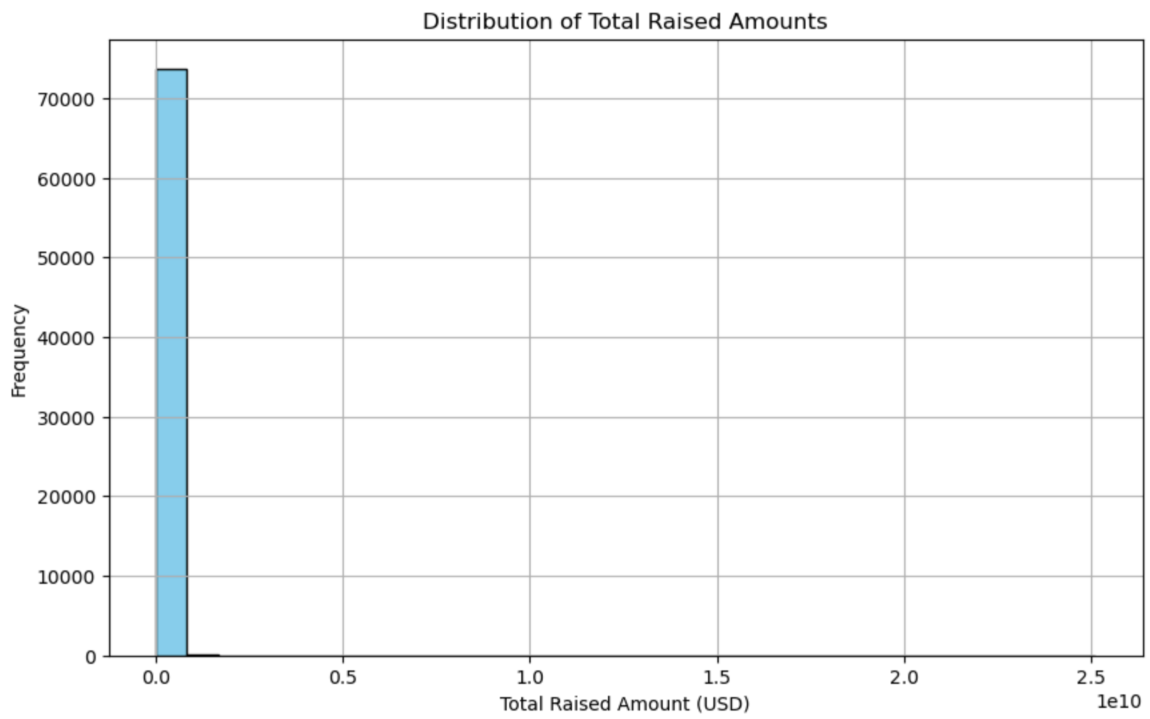
```
# Import necessary libraries
import matplotlib.pyplot as plt

# Convert Spark DataFrame to Pandas DataFrame for visualization
funding_pd_df = funding_df.toPandas()

# Plot a histogram to visualize the distribution of total raised amounts
plt.figure(figsize=(10, 6))
plt.hist(funding_pd_df['total_raised_amount_usd'], bins=30, color='skyblue', edgecolor='black')
plt.title('Distribution of Total Raised Amounts')
plt.xlabel('Total Raised Amount (USD)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()

# Calculate summary statistics
total_raised_mean = funding_pd_df['total_raised_amount_usd'].mean()
total_raised_median = funding_pd_df['total_raised_amount_usd'].median()
total_raised_max = funding_pd_df['total_raised_amount_usd'].max()
total_raised_min = funding_pd_df['total_raised_amount_usd'].min()

print("Mean Total Raised Amount (USD):", total_raised_mean)
print("Median Total Raised Amount (USD):", total_raised_median)
print("Maximum Total Raised Amount (USD):", total_raised_max)
print("Minimum Total Raised Amount (USD):", total_raised_min)
```



```
Mean Total Raised Amount (USD): 13419965.914001085
Median Total Raised Amount (USD): 1725000.0
Maximum Total Raised Amount (USD): 25106985000.0
Minimum Total Raised Amount (USD): 0.0
```

```
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import RandomForestRegressor
from pyspark.ml.evaluation import RegressionEvaluator


# Drop rows with null values in the target column
funding_df = funding_df.dropna(subset=['total_raised_amount_usd'])

# Define feature columns (excluding the target column)
feature_cols = ['num_rounds', 'average_raised_amount_usd']

# Create a VectorAssembler to combine feature columns into a single vector column
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")

# Transform the DataFrame to include the features column
funding_df = assembler.transform(funding_df)

# Split the data into training and test sets
train_data, test_data = funding_df.randomSplit([0.8, 0.2], seed=42)

# Initialize Random Forest Regressor model
rf = RandomForestRegressor(featuresCol="features", labelCol="total_raised_amount_usd")

# Train the model
rf_model = rf.fit(train_data)

# Make predictions on the test data
predictions = rf_model.transform(test_data)

# Evaluate the model using RegressionEvaluator
evaluator = RegressionEvaluator(labelCol="total_raised_amount_usd", predictionCol="prediction", metricName="rmse")
rmse = evaluator.evaluate(predictions)

# Print the Root Mean Squared Error (RMSE)
print("Root Mean Squared Error (RMSE):", rmse)
```

```
    Root Mean Squared Error (RMSE): 70269188.56535776
```

```
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import LinearRegression
from pyspark.ml.evaluation import RegressionEvaluator


# Drop rows with null values in the target column
funding_df = funding_df.dropna(subset=['total_raised_amount_usd'])

# Define feature columns (excluding the target column)
feature_cols = ['num_rounds', 'average_raised_amount_usd']

# Create a VectorAssembler to combine feature columns into a single vector column
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features2")

# Transform the DataFrame to include the features column
funding_df = assembler.transform(funding_df)

# Split the data into training and test sets
train_data, test_data = funding_df.randomSplit([0.8, 0.2], seed=42)

# Initialize Linear Regression model
lr = LinearRegression(featuresCol="features2", labelCol="total_raised_amount_usd")

# Train the model
lr_model = lr.fit(train_data)

# Make predictions on the test data
predictions = lr_model.transform(test_data)

# Evaluate the model using RegressionEvaluator
evaluator = RegressionEvaluator(labelCol="total_raised_amount_usd", predictionCol="prediction", metricName="rmse")
rmse = evaluator.evaluate(predictions)

# Print the Root Mean Squared Error (RMSE)
print("Root Mean Squared Error (RMSE):", rmse)

# Print the coefficients and intercept of the model
print("Coefficients: " + str(lr_model.coefficients))
print("Intercept: " + str(lr_model.intercept))
```

```
24/05/03 01:15:52 WARN Instrumentation: [02faf007] regParam is zero, which might cause numerical instability and overfitting.
24/05/03 01:15:52 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.JNIBLAS
24/05/03 01:15:53 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.lapack.JNILAPACK
Root Mean Squared Error (RMSE): 32933091.97079012
Coefficients: [14565984.392452283,1.2571142332052363]
Intercept: -17675564.858467475
```

```
import matplotlib.pyplot as plt
import numpy as np
import boto3

# Function to create actual vs. predicted and residuals plots
def create_plots(actual, predicted, residuals):
    # Plot actual vs. predicted values
    plt.figure(figsize=(10, 6))
    plt.scatter(actual, predicted, color='blue', alpha=0.5)
    plt.xlabel('Actual Total Raised Amount (USD)')
    plt.ylabel('Predicted Total Raised Amount (USD)')
    plt.title('Actual vs. Predicted Total Raised Amount')
    plt.grid(True)
    plt.savefig('actual_predicted_chart.png')

    # Plot residuals
    plt.figure(figsize=(10, 6))
    plt.scatter(actual, residuals, color='red', alpha=0.5)
    plt.axhline(y=0, color='black', linestyle='--')
    plt.xlabel('Actual Total Raised Amount (USD)')
    plt.ylabel('Residuals')
    plt.title('Residuals Plot')
    plt.grid(True)
    plt.savefig('residuals_chart.png')

# Function to upload files to S3
def upload_to_s3(bucket_name, folder, file_paths):
    # Initialize Boto3 S3 client
    s3 = boto3.client('s3')

    # Upload files to S3
    for file_path in file_paths:
        file_name = file_path.split('/')[-1]
        s3.upload_file(file_path, bucket_name, folder + file_name)
        print(f"Uploaded {file_name} to S3 bucket {bucket_name} at folder {folder}")

# Sample actual and predicted values (replace with your actual values)
actual = [1000000, 2000000, 3000000, 4000000, 5000000]
predicted = [950000, 2100000, 3200000, 3900000, 5100000]
residuals = np.array(actual) - np.array(predicted)

# Create plots
create_plots(actual, predicted, residuals)

# Specify S3 bucket details_
folder = 'charts/'

# Specify the paths of the local chart files
file_paths = ['actual_predicted_chart.png', 'residuals_chart.png']

# Upload the charts to S3
upload_to_s3(bucket_name, folder, file_paths)
```

```
Uploaded actual_predicted_chart.png to S3 bucket final-project-19 at folder charts/
Uploaded residuals_chart.png to S3 bucket final-project-19 at folder charts/
```