Table of Contents

Calculations

All calculations for speed-up, efficiency, and overheads are performed using Spark SQL functions with a test CSV file. The following sections describe the specific Spark operations used in different analyses.

Average Price Calculation

This function calculates the average close price per year from a CSV file. It uses the Spark DataFrame API to read, process, and save the results.

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, avg, year, to_date

def average_close_per_year(input_path, output_path):
    spark = SparkSession.builder.appName("AverageClosePerYear").getOrCreate()

    # Read the CSV file as DataFrame
    df = spark.read.csv(input_path, header=True, inferSchema=True)

    # Convert 'Date' column to date type with the format 'DD-MM-YYYY'
    df = df.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))

    # Filter rows where 'Date' is not null
    df = df.filter(df["Date"].isNotNull())
```

```
# Calculate the average close price per year
average_close_per_year = (
    df.withColumn("Year", year(col("Date")))
      .groupBy("Year")
      .agg(avg("Close").alias("AverageClose"))
      .orderBy("Year")
)

# Save the result in the output directory
average_close_per_year.write.mode("overwrite").csv(output_path, header=True)

spark.stop()
```

Explanation

- **Reads a CSV file** and converts the `Date` column to a proper date format (`DD-MM-YYYY`).
- **Filters out rows** where the `Date` is null.
- **Groups data by `Year`** and calculates the **average `Close` price**.
- **Saves the output** to a CSV file in the specified output directory.

Análisis de la Gráfica

La siguiente gráfica muestra la tendencia del precio de cierre promedio por año:

General Trend

The graph shows the average annual close prices from **1999 to 2022**. There is a steady upward trend, represented by the **regression line** (blue).

Regression Line

The regression equation is **y = 3.68x - 7348.54**, indicating an annual increase of approximately **3.68 units**. The positive slope suggests a gradual rise in average close prices over time.

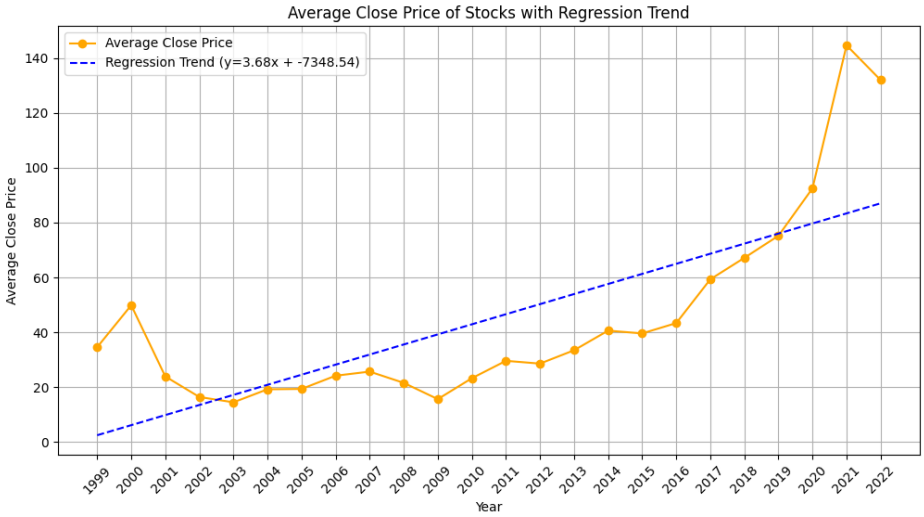

Figure 1: Average Close Price Trend with Regression (1999-2022)

Performance Evaluation

The performance evaluation focuses on analyzing the processing speed and efficiency when handling stock market data using different configurations on cloud infrastructure.

The analysis uses a **test CSV file** with sample stock market data for generating plots and conducting calculations.

Speed-Up with Different Number of vCPUs and Nodes

This section examines how the execution time improves when increasing the number of virtual CPUs (vCPUs) and nodes.

A regression analysis is plotted using the data from the test file to visualize the speed-up trend.

Identified Overheads

Identifies and analyzes the overheads encountered during data processing, such as:

- **Data Shuffling Overheads**
- **Network Latency**
- **I/O Operations Delays**

Overheads are calculated based on execution times derived from the test file.

Optimizations Done

This section outlines the optimizations implemented to improve performance, including:

- **Parallel Processing**
- **Efficient Data Partitioning**
- **Caching Strategies**

The optimizations are tested and validated using the sample data file.
