Implementation & Deviations from Initial Plan

Big Data Management and Processing

Maria Christofi
University of Nicosia
christofi.m9@live.unic.ac.cy

Project Overview

This project performs a Big Data analysis of Bitcoin network activity using historical daily transaction data. The goal is to demonstrate a complete Big Data workflow, including data ingestion into a NoSQL database, data cleaning and transformation, batch analytics, trend and anomaly detection, and visualization of insights.

The project applies core Big Data concepts such as:

- NoSQL data modeling with MongoDB
- Large-scale data ingestion
- Indexing and aggregation for efficient queries
- Batch analytics concepts inspired by MapReduce and Apache Spark
- Analytical interpretation of trends and abnormal market activity

All analysis is implemented in a well-documented Jupyter Notebook to ensure clarity and reproducibility.

Dataset

- **Dataset:** Bitcoin daily network statistics
- **Source:** Google Cloud (BigData Query)
- **Time Range:** 2009 – 2026
- **Key Attributes:**
    1. day
    2. tx_count
    3. total_output_satoshis
    4. total_fee_satoshis
    5. avg_fee_satoshis

The dataset is provided as a CSV file *(btc_daily.csv)*.

Project Structure

This project is organized into two Jupyter Notebooks, each serving a distinct role within the overall Big Data workflow. The primary notebook, MongoDB-DatasetInsights.ipynb, contains the complete end-to-end analysis, including data ingestion into MongoDB Atlas, data cleaning and transformation, indexing strategies, aggregation queries, trend analysis, spike detection, and visualization of results. A secondary notebook, Spark-BatchAnalysis.ipynb, is included to demonstrate batch analytics using Apache Spark and to illustrate distributed data processing concepts taught in the course.

Installation & Environment Setup

1. Python Environment

Ensure Python 3.9 or newer is installed.

2. Required Python Packages

Install the required dependencies:

- pip install pandas
- pip install pymongo
- pip install matplotlib

For Apache Spark functionality:

- pip install pyspark

Note: Spark is optional for reviewing the project results. The core analytics and insights can be reproduced without Spark.

MongoDB Atlas Setup:

To reproduce the full pipeline, the reader must provide MongoDB Atlas credentials.

Steps:

1. Create a free MongoDB Atlas account: https://www.mongodb.com/cloud/atlas
2. Create a **free tier cluster**.
3. Create a **database user** with **read/write** permissions.
4. Add your IP address to the **Network Access** whitelist.
5. Copy the MongoDB connection URI.

Replace the URI in the notebook configuration section with your own credentials.

Apache Spark Usage

Apache Spark is used in this project to demonstrate batch-oriented Big Data processing concepts. While the core analytical results are produced using MongoDB aggregation pipelines and Python-based analysis, the Spark notebook illustrates how similar transformations and aggregations can be executed using a distributed processing framework. This separation highlights the distinction between data storage (MongoDB) and data processing (Spark), aligning with standard Big Data architectural principles. Spark execution may depend on local Java and Spark configuration and therefore serves as a supplementary demonstration rather than a dependency for reproducing the main results.

Running the Notebook

To reproduce the core results of this project, users should first configure their MongoDB Atlas credentials and execute the MongoDB-DatasetInsights.ipynb notebook from top to bottom. This notebook generates all primary analyses, aggregations, and visualizations discussed in the report. The Spark-BatchAnalysis.ipynb notebook can then be run optionally to observe batch analytics implemented using Apache Spark. Running the Spark notebook is not required to reproduce the main findings but provides additional insight into distributed data processing techniques.

The MongoDB-DatasetInsights notebook performs the following automatically:

I. Loads the dataset

II. Cleans and transforms the data

III. Ingests the data into MongoDB

IV. Creates indexes for time-series queries

V. Executes aggregation pipelines for trends and spikes

VI. Generates all visualizations used in the analysis

The Spark-BatchAnalysis notebook performs the following automatically:

VII. Loads the cleaned daily Bitcoin dataset into an Apache Spark DataFrame

VIII. Applies schema inference and column validation for distributed computation

IX. Executes batch aggregations using Spark SQL and DataFrame APIs

X. Produces yearly and multi-period summaries for trend analysis at scale

XI. Identifies anomalous activity patterns through aggregation-based analytics

XII. Demonstrates Big Data batch processing concepts using Apache Spark

Reproducibility Notes

- All preprocessing, aggregation, and analysis steps are explicitly documented in the notebook.
- Indexes and queries are created programmatically.
- Visualizations are generated directly from processed data.
- The separation between storage (MongoDB) and processing (Python/Spark) follows Big Data best practices.

Key Results

- Identification of long-term growth trends in Bitcoin network usage.
- Detection of transaction volume spikes linked to periods of high market activity.
- Analysis of abnormal fee spikes indicating network congestion events.
- Clear demonstration of Big Data ingestion, processing, and analytics concepts.