Cloud Computing Project proposal

Big Data Analytics (Type 2)

# Introduction

In this project, I used advanced big data technologies to analyze the “MovieLens dataset”, a large-scale dataset containing over 32 million ratings and 2 million tags across 87,585 movies. By leveraging cloud computing and machine learning skills, the project discloses patterns and insights that inform movie-related decision-making and user engagement strategies.

Data-driven visions have become progressively vital in the entertainment industry, where understanding audience preferences is the key to success. These streaming services and movie studios rely on gathered information to assemble personalized content, identify trends, and increase user satisfaction. By exploring the given dataset, we can benefit content creators, producers, and even market planners by retrieving valuable information from the resource.

The primary objective is to conduct a comprehensive analysis of the “MovieLens dataset” by using cloud-based big data analytics techniques. This includes implementing classification, clustering, regression, and association rule mining methods to identify and understand decisive patterns within the data. By applying these techniques, we can provide a deeper understanding of the factors influencing movie ratings and user interactions.

Traditional computing methods often face visible limitations when dealing with large datasets like the “MovieLens dataset”. When handling thousands, and millions of records, it requires considerable computational resources and storage capacity to execute complex analytics tasks, which is simply not possible on conventional local systems. What’s more, traditional devices cannot scale efficiently, making it difficult to perform in-depth analysis and process data in real time.

Cloud computing allows scalability and cost-effectiveness to large-scale data analysis. By utilizing cloud-based tools such as Apache Spark for distributed data processing and HDFS for scalable storage, this project can efficiently manage the “MovieLens dataset” and execute complex analysis.

# Technical Solutions

**Docker**:

* **Containerization of Services**: Docker was used to create isolated containers for each of the services needed for your project, including the Hadoop ecosystem and Jupyter Notebook. This allows for a consistent environment across different systems.
* **Environment Consistency**: Ensure that the necessary dependencies for services (Jupyter Notebook and HDFS) are properly configured without installing them directly on the host machine.

**Docker Compose**:

* **Service Orchestration**: Docker Compose allows to define and manage multiple services (like Jupyter Notebook, Hadoop’s Namenode and Datanode) in a single YAML file.
* **Multi-Service Management**: Run several containers simultaneously (Notebook, Namenode, and Datanode), so these services can communicate with one another if needed.

**HDFS (Hadoop Distributed File System)**:

* **Data Storage and Management**: Distributed storage used for storing the MovieLens datasets**:** ratings.csv, movies.csv, links.csv (not used in analysis), and tags.csv inside HDFS.

**Spark:**

* **Distributed Data Processing:** Apache Spark was used to process and analyze large-scale data in a distributed environment. The Spark cluster, consisting of a Spark master and worker nodes, is responsible for parallelizing the data operations across nodes, speeding up computations.
* **Cluster-based Machine Learning and Analysis:** Using Spark’s built-in libraries (PySpark, MLlib), I was able to load data from HDFS, perform data transformations, and run machine learning algorithms on distributed datasets efficiently.
* **Integration with Jupyter Notebook:** Spark was integrated with the Jupyter Notebook to provide an interactive platform for writing Spark code, performing big data queries, and executing machine learning tasks with Spark's distributed capabilities. This enabled streamlined access to Spark’s power through familiar Python libraries.

# Monthly cost estimation:

A screenshot of a computer

Description automatically generated

# Workflow (made by draw.io):

A diagram of a diagram

Description automatically generated with medium confidence