MANIPAL INSTITUTE OF TECHNOLOGY
Manipal – 576 104

DEPARTMENT OF COMPUTER SCIENCE & ENGG.

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

CERTIFICATE

This is to certify that Ms./Mr. …………………...…………………………………… Reg. No. …..…………………… Section: ……………… Roll No ....has satisfactorily completed the lab exercises prescribed for Big Data Analytics Lab [CSE 3263] of Third Year B. Tech. Degree at MIT, Manipal, in the academic year 2023-2024.

Date: ……...................................

Signature
Faculty in Charge

Signature
Head of the Department

CONTENTS

Course objectives:

This laboratory course enable students to

- Understand usage HDFS Commands, HIVE Queries and map reduce programs
- Implement recommendation system for a real-world problem
- Understand usage of classification and clustering algorithm on Big Data.
- Understand how graph data can be analysed
- Demonstrate risk estimation using simulation

Course outcomes:

At the end of this course, students will gain the

- Ability to use HDFS Commands, query HIVE and develop map reduce programs
- To develop a recommendation system for a real-world problem
- To classify big data by applying algorithms based on supervised and unsupervised techniques.
- To analyse graph data using GraphX.
- To apply Monte-Carlo simulation for estimating risk.

Evaluation plan

- Internal Assessment Marks : 60%

  - Continuous Evaluation : 60%
    4*10=40M for (2*10=20M for observation book, + 2*10=20M for Execution and viva)
    20M for midterm evaluation

    The assessment will depend on punctuality, program execution, maintaining the observation note and answering the questions in viva voce.

- End semester assessment of 2 hours duration: 40 %

INSTRUCTIONS TO THE STUDENTS

Pre- Lab Session Instructions

1. Students should carry the Lab Manual Book and the required stationery to every lab session.
2. Be in time and follow the institution dress code.
3. Must Sign in the log register provided.
4. Make sure to occupy the allotted seat and answer the attendance
5. Adhere to the rules and maintain the decorum.
6. Students must come prepared for the lab in advance.

In- Lab Session Instructions

- Follow the instructions on the allotted exercises.
- Show the program and results to the instructors on completion of experiments.
- On receiving approval from the instructor, copy the program and results in the Lab record
- Prescribed textbooks and class notes can be kept ready for reference if required.

General Instructions for the exercises in Lab

- Implement the given exercise individually and not in a group.
- Observation book should be complete with program, proper input output clearly showing the parallel execution in each process. Plagiarism (copying from others) is strictly prohibited and would invite severe penalty in evaluation.
- The exercises for each week are divided under three sets:
  - ➢ Solved example
  - ➢ Lab exercises - to be completed during lab hours
  - ➢ Additional Exercises - to be completed outside the lab or in the lab to enhance the skill
- In case a student misses a lab class, he/ she must ensure that the experiment is completed during the repetition class with the permission of the faculty concerned but credit will be given only to one day’s experiment(s).
- Questions for lab tests and examination are not necessarily limited to the questions in the manual, but may involve some variations and / or combinations of the questions.

THE STUDENTS SHOULD NOT

- Bring mobile phones or any other electronic gadgets to the lab.
- Go out of the lab without permission.

Lab No 1: **Date:**

Introduction to Basic PySpark Programs

I. Introduction

PySpark is a Python library for Apache Spark, an open-source, distributed computing system. It provides an interface for programming Spark with Python, making it accessible for data engineers and data scientists. Key basics of PySpark include:

1. Resilient Distributed Datasets (RDDs): The fundamental data structure in Spark, RDDs represent distributed collections of data that can be processed in parallel.

2. DataFrames: PySpark introduces DataFrames, which offer a more user-friendly and optimized API for structured data processing. They are similar to SQL tables or Pandas DataFrames.

3. Transformations and Actions: PySpark operations are categorized into transformations (e.g., map, filter) and actions (e.g., count, collect). Transformations are lazily evaluated, and actions trigger computation.

4. Spark Context and SparkSession: PySpark requires a SparkContext to connect to a Spark cluster, and SparkSession simplifies the creation and configuration of Spark jobs.

5. SQL and Machine Learning Libraries: PySpark supports SQL queries for structured data and includes libraries for machine learning (MLlib) and graph processing (GraphFrames).

Understanding these basics empowers users to harness the power of PySpark for distributed data processing and analysis.

Lab Exercises:

1. Write a PySpark program to square set of integers.
2. Write a PySpark program to find the maximum of given set of numbers.
3. Write a PySpark program to find average of N numbers.
4. Demonstrate how to read a CSV file into a PySpark DataFrame.
5. Use PySpark commands to display the first few rows and schema of a DataFrame.
6. Calculate basic summary statistics for a specific column in the DataFrame.

Lab No 2: **Date:**

Simple PySpark Programs

Transformations in PySpark are operations that produce a new Resilient Distributed Dataset (RDD) or DataFrame from an existing one. They are the building blocks of PySpark's lazy evaluation paradigm, meaning they are not executed immediately, but their execution is deferred until an action is called. Some common transformations in PySpark include:

1. Map: Applies a function to each element in the RDD or DataFrame.
2. Filter: Selects elements that satisfy a given condition.
3. GroupBy: Groups data based on a specified key or keys.
4. Join: Combines two RDDs or DataFrames based on a common key.
5. Union: Combines two RDDs or DataFrames into a single one.
6. Distinct: Removes duplicate elements from the dataset.
7. FlatMap: Similar to map, but each input item can be mapped to zero or more output items.

Understanding and using these transformations allow for the creation of complex data processing pipelines in PySpark, facilitating scalable and distributed data manipulation.

Lab Exercises:

1) Implement a PySpark script that applies transformations like filter and withColumn on a DataFrame.

2) Write a PySpark script that performs actions like count and show on a DataFrame.

3) Demonstrate how to perform basic aggregations (e.g., sum, average) on a PySpark DataFrame.

4) Show how to write a PySpark DataFrame to a CSV file.

5) Implement wordcount program in PySpark.

Lab No 3: **Date:**

Entity Resolution Application using PySpark

Entity resolution, also known as record linkage or deduplication, is a data integration process that identifies and links records that refer to the same real-world entity across diverse data sources. The goal is to reconcile and merge information about entities, such as individuals or businesses, even when they are represented inconsistently or incompletely in different datasets. Entity resolution involves comparing and analyzing attributes like names, addresses, and other identifying information to determine the likelihood of a match. This process is crucial in various domains, including customer relationship management, healthcare, finance, and law enforcement, where accurate and consolidated data is essential. Advanced techniques, such as probabilistic matching and machine learning algorithms, are often employed to enhance the accuracy and efficiency of entity resolution in handling large and complex datasets.

Lab Exercises:

1) Develop a PySpark script to clean and preprocess data before performing entity resolution. Include steps like tokenization and normalization.
2) Implement a PySpark program that computes similarity scores between records using a chosen similarity metric.
3) Implement a PySpark program to evaluate the precision, recall, and F1-score of an entity resolution model.

Lab No 4: **Date:**

Recommendation System using PySpark

A recommendation system is a software application that suggests items or content to users based on their preferences, behaviors, and historical interactions. It leverages algorithms to analyze user data and identify patterns, aiming to provide personalized and relevant recommendations. There are two primary types of recommendation systems: collaborative filtering, which recommends items based on the preferences of users with similar tastes, and content-based filtering, which suggests items similar to those the user has previously liked. Hybrid approaches combine these methods for more robust and accurate suggestions. Recommendation systems are widely used in e-commerce platforms, streaming services, social media, and other online applications to enhance user experience, engagement, and satisfaction by delivering tailored content or product suggestions.

Lab Exercises:

1) Demonstrate how to load a dataset suitable for recommendation systems into a PySpark DataFrame.
2) Implement a PySpark script that splits the data and trains a recommendation model.
3) Implement a PySpark script using the ALS algorithm for collaborative filtering.
4) Implement code to evaluate the performance of the recommendation model using appropriate metrics.

Lab No 5: **Date:**

Mini Project Phase-1

Choose a problem statement, define objectives, Review Literature, Define project structure, technical specifications, time line, start design and implementation.

Lab No 6: **Date:**

Prediction with Decision Trees

Prediction using decision trees involves utilizing a decision tree algorithm to make informed predictions or classifications based on input features. Decision trees are hierarchical structures that recursively split data into subsets, guided by feature attributes, to ultimately reach a decision or prediction at the tree's leaves. The algorithm learns from training data, where it identifies optimal feature splits to create a tree that best captures patterns and relationships within the dataset. In predictive modeling, decision trees are employed for tasks such as classification or regression. For classification, the tree assigns instances to predefined categories, while in regression, it predicts numerical values. Decision trees are interpretable and effective in handling complex relationships, making them valuable in various domains such as finance, healthcare, and marketing for making accurate predictions and aiding decision-making processes.

Lab Exercises:

1) Demonstrate how to load a dataset suitable for prediction into a PySpark DataFrame and Display basic statistics and information about the dataset.
2) Implement a PySpark script to handle missing values and categorical features in the dataset.
3) Develop a PySpark script that trains a decision tree model on the training dataset.
4) Implement code to evaluate the decision tree model using metrics such as accuracy, precision, and recall.

Lab No 7: **Date:**

Anomaly detection with K-means Clustering

Anomaly detection in a network refers to the identification of unusual or suspicious patterns, behaviors, or activities that deviate from normal operational norms. It involves analyzing network traffic, system logs, or user behavior to detect abnormalities that may indicate security threats, malfunctions, or potential risks. Various techniques are employed, including statistical methods, machine learning algorithms, and heuristic approaches, to establish a baseline of normal behavior and flag deviations. Anomalies could signify security breaches, malware infections, or system faults. Effective anomaly detection enhances network security by enabling prompt identification and response to potential threats, thereby minimizing the impact of malicious activities and ensuring the integrity and reliability of the network infrastructure. It plays a critical role in cybersecurity strategies, helping organizations proactively safeguard their networks against evolving threats and vulnerabilities.

Lab Exercises:

1. Implement a PySpark script to handle any missing values and scale numerical features.
2. Develop a PySpark script that uses the K-means algorithm to cluster data points.
3. Develop a PySpark script that labels data points as anomalies based on their cluster assignments.
4. Implement code to evaluate the effectiveness of the K-means clustering model in detecting anomalies.

Lab No 8: **Date:**

Mini Project Phase-2

Start documentation of the implementation in IEEE paper format using Latex template.

Lab No 9: **Date:**

Estimating Risk Through Monte Carlo Simulation

Financial stock risk analysis involves evaluating the potential uncertainties and volatility associated with investing in a particular stock or the overall market. Investors and analysts use various tools and methodologies to assess factors that may impact stock prices, such as market trends, economic indicators, company performance, and geopolitical events. Quantitative measures, including standard deviation, beta, and value at risk (VaR), are often employed to quantify and analyze the level of risk in a stock or portfolio. Fundamental analysis examines a company's financial health, earnings, and growth potential, while technical analysis studies historical price and trading volume patterns. Additionally, sentiment analysis considers market sentiment and news to gauge investor behavior. By conducting comprehensive risk analysis, investors can make more informed decisions, develop risk management strategies, and optimize their investment portfolios in response to changing market conditions.

Lab Exercises:

1) Implement a PySpark script that runs Monte Carlo simulations in parallel.
2) Demonstrate how to define and apply probability distributions to input parameters using PySpark.

Lab No 10: **Date:**

PySpark Cluster usage Demo and Mini Project Evaluations

1. Submit the paper prepared according to IEEE format pertaining to the Mini Project.
2. Explore the usage of cluster.

References:

1. Tom White, *Hadoop: The definitive guide (4e)*, O’Reilly, 2015.
2. Vignesh Prajapathi, *Big Data Analytics with R and Hadoop*, Packt Publishing, 2013.
3. Jeffery Aven, *Data Analytics with Spark using Python*, Pearson, 2018
4. Sandya Ryza, Uri Laserson, Sean Owen and Josh Wills, *Advanced Analytics with spark (2e)*, O’Reilly Media Inc, 2017.
5. Holden Karau, Andy Konwinski, Patrick Wendell and Matei Zaharia, *Learning Spark: Lightning-Fast Big Data Analysis (2e)*, O’Reilly Media Inc, 2020.
6. Github link to sample code