# (7082CEM)

# Coursework

Demonstration of a Big Data Program

MODULE LEADER: Dr. Marwan Fuad

Student Name: Manoj Sibi Mogan

SID: 10224061

**INDIVIDUAL’S ANNUAL INCOME ANALYSIS THROUGH MACHINE LEARNING USING PYSPARK**

I can confirm that all work submitted is my own: Yes

**Table of Contents**

1. **Introduction ---------------------------------------------------------------------- 3**
2. **Implementation ------------------------------------------------------------------ 4**
   1. **Background study ----------------------------------------------------------- 4**
      1. **SPARK ------------------------------------------------------------------- 4**
      2. **SparkSession ----------------------------------------------------------- 4**
      3. **DataFrame ------------------------------------------------------------- 4**
      4. **PySpark ----------------------------------------------------------------- 5**
   2. **Dataset ------------------------------------------------------------------------ 6**
   3. **Installation Steps ----------------------------------------------------------- 7**
   4. **Pre-processing the Dataset ------------------------------------------------ 10**
   5. **Data Visualization ---------------------------------------------------------- 16**
   6. **Machine Learning – Linear Regression -------------------------------- 22**
   7. **Discussion --------------------------------------------------------------------- 26**
   8. **Conclusion -------------------------------------------------------------------- 26**
   9. **References -------------------------------------------------------------------- 27**
3. **Appendix -------------------------------------------------------------------------- 27**

1. **Introduction**

In this paper I am going to explain and demonstrate how to setup and use PySpark realizing the entire project. The result of this project are also reported and discussed. Sophisticated platforms have started to develop and handle a massive amount of data. Spark is one of the famous distributed computation platform for big data analysis which comprises outstanding functionalities such as performing faster in large dataset, ease of use, fault tolerance and overcoming memory latency.

We start with introduction and detailed explanation of the stack of programs that are going to be used throughout this project and how to install them. Spark offers programming for different programming languages, Python has been considered for this study for its special qualities such as real time screen analytics, facilitates data visualisation and process faster on the framework. PySpark which is a combination of Spark and Python is implemented in this study for Spark data processing through PySpark which easily integrate and collaborate with RDD through navigating Py4J library. In this study we have used Python, Spark, Machine learning (Linear Regressing) to identify the Correlation, Accuracy and Error rate along with this Data visual representation was demonstrated in Tableau.

Data for this study is based on Individual’s Annual income which is covered worldwide. However due to time limitations part of the dataset is extracted from Kaggle dataset and used in this study for Big Data analysis and Data visualisation. The collected data contains eight attributes which are best suited for this study. Education Qualification, Age, Occupation, Income are some of the attributes chosen for the study. After the selection of dataset PySpark was launched using number on installation steps. Next, pre-processing for dataset was completed through loading data (Loading Dataframe). Following this, duplications were removed and columns were dropped including handing missing values. Then Dataframe operations were completed through different methods: Groupby, Distinct, Orderby, Built-in-functions, describe function and check value for specific column. This then led to exploratory analysis. The explanation for visualisations was shown in Tableau. More information about the study and results are discussed in details in the following sections.

1. **Implementation**
   1. **Background study**

**SPARK**

Spark is an in-memory, distributed, fault-tolerant processing framework. It is implemented in the Java virtual-machine (JVM) compatible programming language Scala, it provides higher-level abstractions than MapReduce and thus enhances developer productivity. As an in-memory solution, Spark excels at tasks that cause bottlenecks on disk IO in MapReduce. Spark is referred to as a replacement to Hadoop, but in real-time Spark and other elements of the BDAS were designed to work closely with Hadoop HDFS and YARN, and several Spark implementations use HDFS for persistent storage.

In Spark, data is represented as resilient distributed datasets (RDD). RDDs are collections of objects that can be partitioned across ‘n’ number nodes of the cluster. The partitioning and subsequent distribution of processing are handled automatically by the Spark framework. The Spark API defines high-level methods that perform operations on RDDs. Operations such as joining, filtering, and aggregation, which would entail hundreds of lines of Java code in MapReduce, are expressed as simple method calls in Spark.

**SPARK SESSION**

Spark session is the entry point for the Spark SQL. It’s the initial object to be created while developing a Spark SQL application. allows for creating a DataFrame creating a Dataset, accessing the Spark SQL services, executing a SQL query, loading a table and accessing DataFrameReader interface to load a dataset of the format of the developer’s choice

**DATAFRAME**

Dataframe in PySpark is the distributed collection of structured or semi-structured data. This data in Dataframe is stored in rows under named columns which is similar to the relational database tables or excel sheets. It also shares some common attributes with RDD like Immutable in nature, follows lazy evaluations and is distributed in nature. It supports a wide range of formats like JSON, CSV, TXT and many more.

**PYSPARK**

PySpark is a great language for performing exploratory data analysis at scale, building machine learning pipelines, and creating ETLs for a data platform. Spark is a popular open source framework that ensures data processing with lightning speed and supports various languages like Scala, Python, Java, and R. It then boils down to your language preference and scope of work. The goal of this post is to show how to get up and running with PySpark and to perform common tasks.

The diagram below shows how both PySpark and Spark contexts are managed by Spark driver with the aid of local file system and through communicating with Spark worker through cluster manager.

Diagram

Description automatically generated

**2.2 Dataset**

An individual’s annual income results from various factors. Intuitively, it is influenced by the individual’s education level, age, gender, occupation, and etc. This dataset which is from US Census Reports (1994 – 1996). Therefore, a version of this dataset has been extracted from Kaggle dataset and used in this study for Big Data analysis and Data visualisation. This chosen dataset includes information on Education Qualification, Age, Occupation, Hours Per Week, Income etc. and demographics. There are 13 attributes and contains 32, 561 number of records in this study. The main aim of this study is to analyse income status of different employees through analysing the factors such income, educational qualification, working sector etc.

The table below shows the attributes from the dataset including its description and data types.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Description** | **Data Types** |
| Age | Age of the employee | integer |
| Work Sector | Employee working sector | string |
| Educational\_Qualification | Employee educational qualification details | string |
| Marital\_Status | Employee Marital details | string |
| Occupation | Employee work nature | string |
| Relationship | Employee’s relationship with their family | string |
| Country | Country to which the employee belongs | string |
| Income | Annual income of the employee | integer |

**2.3 Installation Steps**

Machine used to perform this study has the hard disk drive with the capacity of 1TB, RAM with 16GB and the used operating system is macOS Catalina version 10.15.7. The steps installation steps followed are explained below:

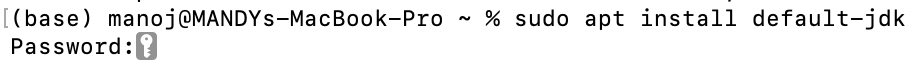
**Step 1:** Initially, it is important to check whether java is installed within the machine through looking at the current java version by entering the below mentioned command in Terminal. If the java is installed it will display the below message.

**Text

Description automatically generated**

And if the java is not installed below mentioned command needs to be used followed by the system password.

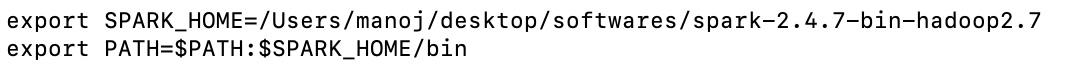
*sudo apt install default-jdk*



**Step 2**:- Download the latest version Spark from (<https://spark.apache.org/downloads.html>) and save the zip folder in the desktop or any folder. Enter the below command in the terminal to unzip the Spark file.

*tar -xzf spark-2.4.7-bin-hadoop2.7.tar*

**Step 3**:- under bash profile setting up the Spark Environment by using the command shown below.



**Step 4**:- To ensure whether the Spark is installed correctly with the below command.

*spark-shell*

Graphical user interface, text, application, letter, email

Description automatically generated

**Step 5**:- After the installation of Spark, need to install Python, Jupyter Notebook by downloading Anaconda (<https://www.anaconda.com/products/individual#macos>) and install it.

**Step 6**:- After the installation of Anaconda, it is important to check whether Python is installed in the machine. The following command will be used to check the Python version.

*python -version*



**Step 7:** After checking the Python version make sure whether Jupyter notebook is successfully installed the following command is used.

*jupyter notebook*

Graphical user interface, text, application

Description automatically generated

**2.4 Pre-processing the Dataset**

Dataset chosen for this study is not clean. In other words, it includes missing values, null values and duplications. Therefore, these inappropriate values are removed through pre-processing step.

* **Load the Data**

Load Dataset through DataFrame

The data set is loaded through the following steps:

* **Load the csv file into Dataframe with the code given below:**

*dtFrame = spark.read.csv('/Users/manoj/desktop/employeeincomedetails.csv', header = True)*

* **Display the uploaded csv file by using below code.**

*dtFrame.show (5)*

Table

Description automatically generated

* **To change the column name and display it from the uploaded file by using the below code.**

*dtFrame2 = dtFrame.withColumnRenamed("age", "Age").withColumnRenamed("workclass", "Sector").withColumnRenamed("fnlwgt", "UID").withColumnRenamed("education", "Educational\_Qualification").withColumnRenamed("education.num", "Education Grade").withColumnRenamed("marital.status", "Marital\_Status").withColumnRenamed("occupation", "Occupation").withColumnRenamed("relationship", "Relationship").withColumnRenamed("race", "Race").withColumnRenamed("sex", "Gender").withColumnRenamed("capital.gain", "Capital Gain").withColumnRenamed("capital.loss", "Capital Loss").withColumnRenamed("native.country", "Country").withColumnRenamed('hours.per.week', 'Hours Per Week').withColumnRenamed("income", "Income")*

Table

Description automatically generated

* **Count the number of records.**

To count the number of records in the dataset by using the code shown below

*dtFrame2.count()*

Graphical user interface, text, application

Description automatically generated

* **Dropping the Columns:**

After identifying all the counts in each column in the data by using the above mentioned code. We can drop the column which is not necessary for our analysis. From the updated data I have dropped some of the columns which is not useful for this analysis work. The dropped columns are **“Capital Gain, Capital Loss, UID, Hours Per Week, Education Grade”** and the code is mentioned below and the updated data is displayed as,

*dtFrame2 = dtFrame2.drop('Capital Gain', 'Capital Loss', 'UID', 'Hours Per Week', 'Education Grade')*

Table

Description automatically generated

* **Replacing the values in Machine readable form:**

The value in Income column in the defined data is mentioned as **“<=50K”** and **“>50K”** in-order to understand it I have assigned those values as **“<=50K as 0”** and **“>50K as 1”**. I have mentioned the code below.

*dtFrame3 = dtFrame2.replace(['<=50K', '>50K'], ['0', '1'], 'Income')*

Table

Description automatically generated

Also, replacing the **“?”** value in the entire data with **“Null”** value.

**Table

Description automatically generated**

* **Replacing the DataTypes:**

Replacing the default datatypes for all the attributes to the preferred datatypes using the code mentioned below and displayed the datatypes for each column.

*dtFrame4 = dtFrame3.withColumn("Age", dtFrame3["Age"].cast(IntegerType())) .withColumn("Sector", dtFrame3["Sector"].cast(StringType())) .withColumn("Educational\_Qualification", dtFrame3["Educational\_Qualification"].cast(StringType())) .withColumn("Marital\_Status", dtFrame3["Marital\_Status"].cast(StringType())) .withColumn("Occupation", dtFrame3["Occupation"].cast(StringType())) .withColumn("Relationship", dtFrame3["Relationship"].cast(StringType())) .withColumn("Gender", dtFrame3["Gender"].cast(StringType())) .withColumn("Country", dtFrame3["Country"].cast(StringType())) .withColumn("Income", dtFrame3["Income"].cast(IntegerType())) .withColumn("Race", dtFrame3["Race"].cast(StringType()))*

Graphical user interface, text, application, email

Description automatically generated

* **Removing the Unwanted Row:**

Removing the “Null” present in few cell and deleting the entire row by using the code mentioned below.

*dtFrame5 = dtFrame4.filter((dtFrame4.Sector != 'Null') & (dtFrame4.Occupation != 'Null') & (dtFrame4.Relationship != 'Null') & (dtFrame4.Gender != 'Null') & (dtFrame4.Country != 'Null') & (dtFrame4.Educational\_Qualification != 'Null') & (dtFrame4.Marital\_Status != 'Null'))*  
  
Table

Description automatically generated

* **Grouping the Data Values:**

Grouping the all the column values in the dataset and arranging them in the descending order using the code mentioned below.

*dtFrame5.groupBy('Age', 'Sector', 'Educational\_Qualification', 'Marital\_Status', 'Relationship', 'Gender', 'Country', 'Income', 'Race').count().sort(desc("count")).show(10)*

Table

Description automatically generated

* **Downloading the Processed Dataset:**

After removing all the unwanted data and changing the values the **“Cleaned Data”** can be saved into the system as **“CSV file”** using the below code.

*dtFrame5.write.csv('/Users/manoj/desktop/Course Work -- BD & DV.csv',header=True)*

**

This **.csv file** is used for data visualisation in **Tableau** and it is further discussed and analysed below.

* 1. **Data Visualisation**

The processed dataset is used which is obtained from the Jupyter Notebook is visualised using Tableau. Let us discuss in detail about the installation procedure of Tableau, how the data is visualised in Tableau.

**Installation Steps (Tabulae):**

Tableau is an interactive data visualization software . It will help to generate graph-type data visualizations. Download the Tableau Desktop 2020.3 from their website and install it in the Desktop/laptop which is used to generate the visualisation. After installing the Tableau upload the processed dataset in to it. Then select “Use Data Interpreter” checkbox to remove the “Null” rows in the processed dataset. After this process the data can be used for visualisation.

**Countries Geographical Representation:**

Location Analysis is a study about different economic factors associated to locations in a country. In this course work, individual employee’s income status of different countries are chosen to find the relationship among the factors and to predict the employee’s income status of each country.

The study has consider specific country and that is represented in the map below with the aid of actual “Longitude and Latitude” also the legends indicates the specific country with colour assigned to it.

Map

Description automatically generated

**Location vs. Gender:**

Based on this study, the below chart explains link between the countries and total number of Male and the Female who were employed in the particular countries. In this percentage of male or female were represented in blue and orange “Dots” respectively.

Map

Description automatically generated

**Educational Qualification vs. Income Count:**

Next, the link between Educational Qualification and Income count attributes were explored and it is illustrated in the “Bar chart” shown below.

Chart, bar chart

Description automatically generated

**Marital Status vs. Income Count:**

Next, the link between Marital Status of the people and their Income count were explored and it is demonstrated in the “Bar chart” shown below.

A picture containing chart

Description automatically generated  
 **Occupation vs. Age vs. Income Count:** Following this, the link between Occupation of the people and total Age count of the people were explored and it is demonstrated in the “Bar chart” and the “Line chart” which demonstrate the link between Occupation of the people and total Age count of the people along with their Income count.

Chart, histogram

Description automatically generated

**Race vs. Work Sector vs. Income Count:** Following this, the link between Race based on region of the people and the working sector of those people and their Income Count were explored and it is demonstrated in the “Bar chart”.

**A picture containing graphical user interface

Description automatically generated**

**﻿Work Sector vs. Gender vs. Income Count:**

The below describes about the connections between three attributes which are Sector, Gender and Income. This is shown in the following “Bar chart” with the comparison between the Male and Female gender.

**Chart, bar chart

Description automatically generated**

﻿**Race vs. Occupation vs. Count of Educational Qualification:**

The below chart describes about the connections between Race based on region of the people, Occupation and Educational Qualification. This is shown in the following “Shape (circle) chart” with different colours illustrates the Race.

Chart, scatter chart

Description automatically generated

**Occupation vs. Sector vs. Gender vs. Income:**

Then this analysis has moved further advance through looking at the connections between four attributes which are Occupation, Sector, Gender and Income, . This is shown in the following different “Shape chart”, where the X-axis represents the Count of the Gender and Y-axis represents the Income.

**A picture containing scatter chart

Description automatically generated**

**Country vs. Educational Qualification vs. Gender vs. Work Sector vs. Occupation:**

Again, this analysis has moved further advance through looking at the connections between five attributes which are Country, Educational Qualification, Gender, Work Sector and Occupation . This is shown in the following using “Bubble chart”.

Chart, bubble chart

Description automatically generated

* 1. **Machine Learning – Linear Regression**

Machine learning is the process by which a computer can work more accurately as it collects and learns from the data it is given. Linear Regression is a type of supervised machine learning algorithm for predicting continuous-valued output where the predicted output is continuous and has a constant slope. It’s used to predict values more effectively and efficiently. In this study we have generated prediction value and correlation matrix to identify the Linear Regression between “Age” and “Income” which are Integer datatypes. The dataset has been split into 70% training dataset and 30% testing dataset and the transformation through vector assembler to setup “Income”, “Age” as features and labels. This is explained in the below code.

The following code has been used for it.

* Importing the cleaned dataset to identify the prediction value and correlation matrix. Here the cleaned dataset has been read using spark and saved in the Data Frame “dtFrame”. Using the below mentioned code the rows and the column of the dataset has been displayed.

*print((dtFrame.count(), len(dtFrame.columns)))*

Graphical user interface, text, application, email

Description automatically generated

* To identify and to calculate the summary statistics of numerical column from the dataset the below mention code is used.

*dfFrame1 = dtFrame.describe().show(5)*

*Table

Description automatically generated*

* To identify the correlation between the two numeric columns in the dataset by using the below mentioned code.

*dtFrame1.select(corr('Age','Income')).show()*

A picture containing graphical user interface, table

Description automatically generated

* Assigning the input value and output value in assembler using VectorAssembler feature. Then assigning the random split value as (0.7 and 0.3) for features and Age respectively. The code for this is mentioned below.

*assembler = VectorAssembler(inputCols=['Income'], outputCol='features')*

*dtFrame2 = assembler.transform(dtFrame)*

*dtFrame2 = dtFrame2.select(['features','Age'])*

*train,test = dtFrame2.randomSplit([0.7, 0.3])*

*train.show(5)*

*test.show(5)*

A picture containing text

Description automatically generated

* Calculating Intercept and Coefficient using the assigned LinearRegression value.

Graphical user interface, text, application

Description automatically generated

* Calculating the prediction and accuracy value using the assigned feature and label column using the below mentioned code.

Table

Description automatically generated

* After then the predicted values were evaluated through R2 values and achieved approximately 0.06%. Higher the R2 better the model.

Graphical user interface, text, application, email

Description automatically generated

* 1. **Discussion:**
  2. **Conclusion:**

* 1. **References:**

Guy Harrison (2015), “*Next Generation Databases*” [Online]

Flannery O'Connor, “*The Internals of Spark SQL (Apache Spark 2.4.5)*” [Online]

Swatee Chand (2020), “*PySpark Programming – Integrating Speed With Simplicity*” [Online]

Chris A. Mattmann (2020), “*Machine Learning with TensorFlow, Second Edition*” [Online]

Davy Cielen, Arno Meysman, Mohamed Ali (2016), “*Introducing Data Science - Big Data, Machine Learning, and More, Using Python Tools*” [Online]

*Adult income dataset*. kaggle. [Online] Available from: <<https://www.kaggle.com/wenruliu/adult-income-dataset>> [06.10. 2016].

*Java 1.8.0\_261* [Online] < <https://www.java.com/en/download/>>

*Spark version 3.0.1* [Online] <<https://spark.apache.org/downloads.html>>

*Anaconda Installers (Python 3.8)* [Online] <<https://www.anaconda.com/products/individual#macos>>

*Tableau Desktop 2020.*3 [Online] <<https://www.tableau.com/en-gb/products/desktop/download>>

*PySpark Flow Diagram* [Online] <<https://www.researchgate.net/figure/An-illustration-of-PySpark-data-flow-The-Python-environments-were-shaded-in-white-and_fig2_325055629>>

[Somesh Routray](https://medium.com/@somesh.routray11) (2020), “*Machine Learning : Linear Regression using Pyspark*” [Online] <<https://towardsdatascience.com/machine-learning-linear-regression-using-pyspark-9d5d5c772b42>>

1. **Appendix**

Please see the link to access the codes used for this study -- [CODE](https://github.com/Manojsibi/Manoj_SM-Projects.git)