Universidad Central del Ecuador

Facultad de Ingeniería y Ciencias Aplicadas

Carrera de Ingeniería en Ciencias de la Computación

Programación avanzada III - Trabajo Grupal Final

Desarrollo de un Proyecto Scala con Apache Spark en IntelliJ: Procesamiento de Datos y Aplicación de Algoritmos de Aprendizaje Automático

Autores:

- Altamirano Ortiz Jonathan Danilo
- Flórez Rivera Yaniry Mabely
- Gualoto Tigrero, Erika Paola

Semestre: 7 mo

Quito-2024

Índice general de contenidos

CAPÍTULO 1: 2
1. Generalidades.....2
1.1. Introducción 2
1.2. Objetivos.....2
1.2.1. Objetivo general 2
1.2.2. Objetivos específicos 3
CAPÍTULO 2: 3
2. Marco Metodológico 3
2.1. Diseño del sistema 3
2.2. Desarrollo del sistema.....3
2.2.1. Plataforma y herramientas.....3
3. Bibliografía 5
4. Anexos: 6

CAPÍTULO 1:

1. Generalidades

1.1. Introducción

En este proyecto, se explorará el uso de Apache Spark y Scala para el procesamiento y análisis de datos, con un enfoque particular en el uso de un conjunto de datos proveniente del Censo de Población y Vivienda de 2010 de Ecuador. Apache Spark se utilizará como motor de procesamiento de datos para manejar grandes volúmenes de información de manera eficiente. El objetivo principal es desarrollar un sistema integrado que permita el procesamiento de datos, la aplicación de algoritmos de aprendizaje automático.

El conjunto de datos en cuestión incluye información detallada sobre viviendas, hogares, personas y la división político-administrativa de Ecuador. Este archivo CSV contiene una alta dimensionalidad y diversos atributos categóricos y numéricos que serán utilizados para:

- Preprocesamiento de Datos: Cargar y limpiar el archivo CSV para eliminar datos inconsistentes y preparar los datos para el análisis.
- Análisis Exploratorio: Realizar operaciones de filtrado, agrupamiento y cálculo de estadísticas descriptivas para entender mejor la distribución y características de los datos.
- Aplicación de Algoritmos de Aprendizaje Automático: Utilizar los datos preprocesados para entrenar modelos de machine learning y obtener predicciones útiles.

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar un sistema utilizando Apache Spark y Scala para el procesamiento y análisis de datos del Censo de Población y Vivienda de 2010 de Ecuador, con el fin de aplicar técnicas de aprendizaje automático.

1.2.2. Objetivos específicos

a) Leer el archivo CSV del censo y realizar la limpieza y transformación de datos necesarias para el análisis.

b) Aplicar técnicas de filtrado, agrupamiento y cálculo de estadísticas descriptivas sobre los datos del censo.

c) Implementar y evaluar algoritmos de machine learning utilizando los datos preprocesados.

CAPÍTULO 2:

2. Marco Metodológico

2.1. Diseño del sistema

El diseño del sistema se centrará en el uso de Apache Spark para el procesamiento eficiente de grandes volúmenes de datos. Spark permitirá realizar operaciones de limpieza, transformación y análisis de datos de manera rápida y distribuida. Scala, como lenguaje funcional, facilitará la implementación de soluciones limpias y mantenibles.

El sistema se enfocará en preparar los datos para la aplicación de algoritmos de aprendizaje automático. Esto incluye la integración de módulos de Spark para leer, transformar y analizar datos, así como la implementación de funciones para calcular estadísticas descriptivas y aplicar modelos predictivos.

2.2. Desarrollo del sistema

2.2.1. Plataforma y herramientas

- **Apache Spark:**

Motor de procesamiento de datos en clúster que permite realizar análisis a gran escala con alta eficiencia. En este proyecto, se utilizará para procesar y analizar grandes conjuntos de datos relacionados con el dominio de interés.

- **Scala:**

Lenguaje de programación funcional que se empleará para implementar el procesamiento de datos y los algoritmos de aprendizaje automático. Scala ofrece soporte para programación funcional y orientada a objetos, ideal para el desarrollo de soluciones robustas y eficientes.

- **IntelliJ IDEA:**

Entorno de desarrollo integrado (IDE) utilizado para el desarrollo del proyecto. IntelliJ IDEA proporciona un entorno completo para trabajar con Scala y Apache Spark, facilitando la escritura, depuración y ejecución del código.

- **SBT (Scala Build Tool):**

Herramienta de construcción utilizada para gestionar dependencias y construir el proyecto. SBT simplifica la configuración del proyecto y la integración de bibliotecas necesarias para el desarrollo con Spark y Scala.

- **Algoritmos de Aprendizaje Automático:**

Modelos y técnicas de machine learning aplicados a los datos preprocesados para obtener insights y predicciones. Se implementarán modelos adecuados para el dominio específico del proyecto, utilizando las capacidades de Spark MLlib.

- **Librerias implementadas:**

  Spark- core. Version 3.5.1

  Spark- sql. Version 3.5.1

  Spark- mllib. Version 3.5.1

  Breezer. Version 2.1.0

  Breezer-viz. Version 2.1.0

3. Bibliografía

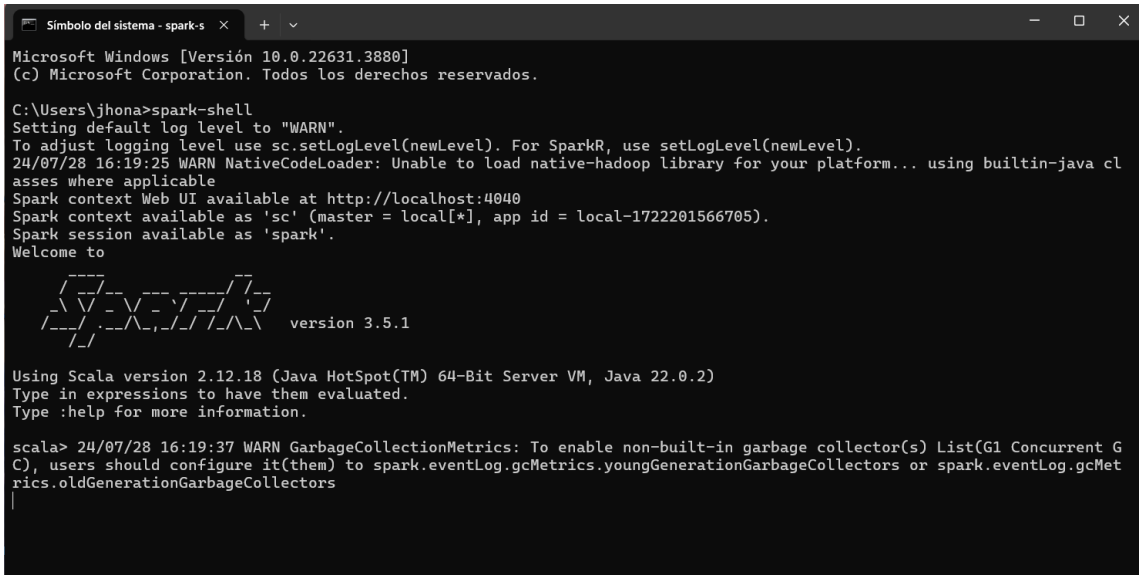
(S/f). Rua.ua.es. Recuperado el 27 de julio de 2024, de https://rua.ua.es/dspace/bitstream/10045/95608/1/tesis_zoila_ruiz.pdf

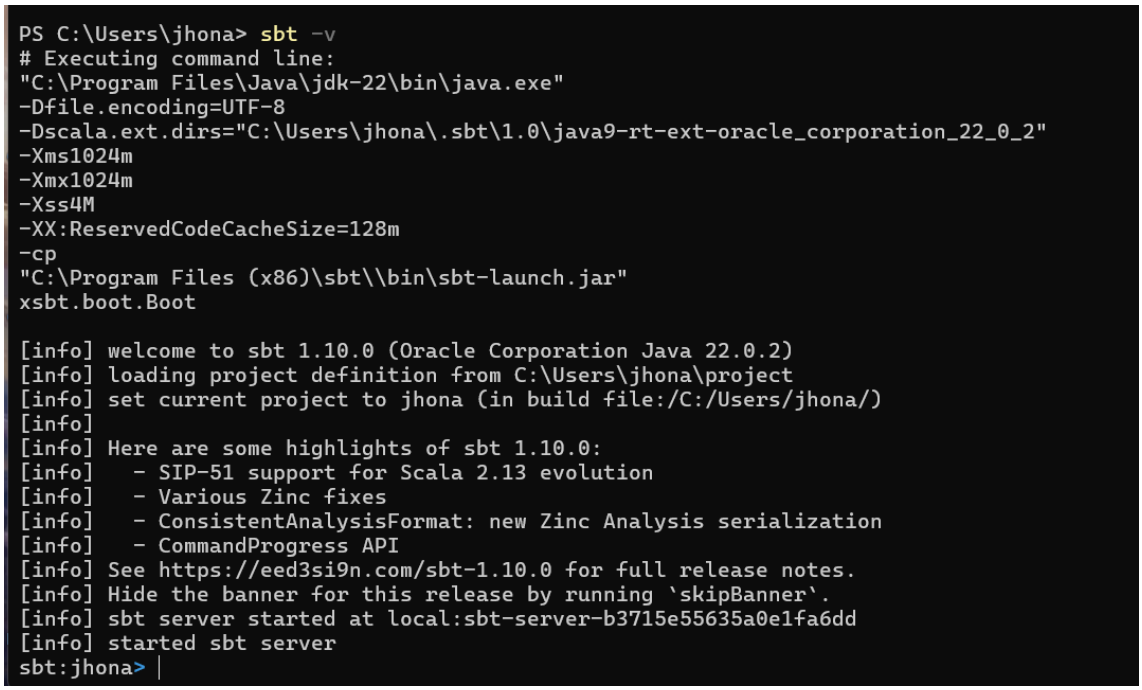
Introducción. (s/f). Scala Documentation. Recuperado el 27 de julio de 2024, de https://docs.scala-lang.org/es/tour/tour-of-scala.html

Download IntelliJ IDEA – the leading java and kotlin IDE. (s/f). JetBrains. Recuperado el 27 de julio de 2024, de https://www.jetbrains.com/idea/download/?section=windows

Download. (s/f). Scala-sbt.org. Recuperado el 27 de julio de 2024, de https://www.scala-sbt.org/download/

4. Anexos:


Símbolo del sistema - spark-s
Microsoft Windows [Versión 10.0.22631.3880]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\jhona>spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/07/28 16:19:25 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://localhost:4040
Spark context available as 'sc' (master = local[*], app id = local-1722201566705).
Spark session available as 'spark'.
Welcome to
version 3.5.1
Using Scala version 2.12.18 (Java HotSpot(TM) 64-Bit Server VM, Java 22.0.2)
Type in expressions to have them evaluated.
Type :help for more information.
scala> 24/07/28 16:19:37 WARN GarbageCollectionMetrics: To enable non-built-in garbage collector(s) List(G1 Concurrent GC), users should configure it(them) to spark.eventLog.gcMetrics.youngGenerationGarbageCollectors or spark.eventLog.gcMetrics.oldGenerationGarbageCollectors


PS C:\Users\jhona> sbt -v
Executing command line:
"C:\Program Files\Java\jdk-22\bin\java.exe"
-Dfile.encoding=UTF-8
-Dscala.ext.dirs="C:\Users\jhona\.sbt\1.0\java9-rt-ext-oracle_corporation_22_0_2"
-Xms1024m
-Xmx1024m
-Xss4M
-XX:ReservedCodeCacheSize=128m
-cp
"C:\Program Files (x86)\sbt\\bin\sbt-launch.jar"
xsbt.boot.Boot
[info] welcome to sbt 1.10.0 (Oracle Corporation Java 22.0.2)
[info] loading project definition from C:\Users\jhona\project
[info] set current project to jhona (in build file:/C:/Users/jhona/)
[info]
[info] Here are some highlights of sbt 1.10.0:
[info] - SIP-51 support for Scala 2.13 evolution
[info] - Various Zinc fixes
[info] - ConsistentAnalysisFormat: new Zinc Analysis serialization
[info] - CommandProgress API
[info] See https://eed3si9n.com/sbt-1.10.0 for full release notes.
[info] Hide the banner for this release by running `skipBanner`.
[info] sbt server started at local:sbt-server-b3715e55635a0e1fa6dd
[info] started sbt server
sbt:jhona>

Main.scala | Grafica.scala | Row.class | Lista.scala | UFunc.scala | build.sbt | sbt.json

```
ThisBuild / version := "0.1.0-SNAPSHOT"

ThisBuild / scalaVersion := "2.13.12"

lazy val root = (project in file("."))
  .settings(
    name := "grupal"
  )

libraryDependencies ++= Seq(
  "org.apache.spark" %% "spark-core" % "3.5.1",
  "org.apache.spark" %% "spark-sql" % "3.5.1" % "provided",
  "org.apache.spark" %% "spark-mllib" % "3.5.1",
  "org.scalanlp" %% "breeze-viz" % "2.1.0",

    "org.scalanlp" %% "breeze" % "2.1.0",



)

```