

UNIVERSIDAD
COMPLUTENSE
DE MADRID



Scala: Trabajo Final

Análisis exploratorio con Scala

Introducción:

Queremos realizar un análisis exploratorio de un dataset en formato csv. Este dataset suele ser usado para procesos de ML en el que se intenta predecir si los ingresos de una persona serán superiores a los 50K anuales.

Descripción:

Se solicita implementar un programa en Scala que sea capaz de hacer un análisis exploratorio básico y sacar algunas conclusiones del dataset. Para ello se hará un uso de las funciones que provee la API de las colecciones de Scala.

También se pedirá en algún apartado que se defina un companion object para poder disponer de funciones de utilidades.

Se va a proveer de un código fuente del programa del cual se tiene que partir y completarlo para su correcto funcionamiento.

En el código provisto se dispone de una función de lectura del fichero csv y la definición de funciones que se deben implementar. Así como también la definición de la clase a la que se mapea cada una de las filas del csv.

El código fuente provisto está formado por 4 ficheros .scala y 1 fichero .csv y tiene la siguiente estructura:

```
src
├── adult.data.clean.csv
├── main
│   └── scala
│       ├── AnalisisExploratorio.scala
│       ├── Analizador.scala
│       ├── Contribuyente.scala
│       └── Utilidades.scala
└── test
    └── scala
```

- AnalisisExploratorio.scala -> Es el fichero del programa principal
- Analizador.scala -> Contiene la definición de un trait que se deberá usar para completar una de los ejercicios.
- Contribuyente.scala -> Definición de la clase Contribuyente a la que se mapea cada fila del fichero .csv

- Utilidades.scala -> Definición de un objeto que tiene implementada la función de lectura del csv y devuelve una secuencia de instancias de la clase Contribuyente: Seq[Contribuyente]
- adult.data.clean.csv -> dataset que tiene la siguientes columnas:

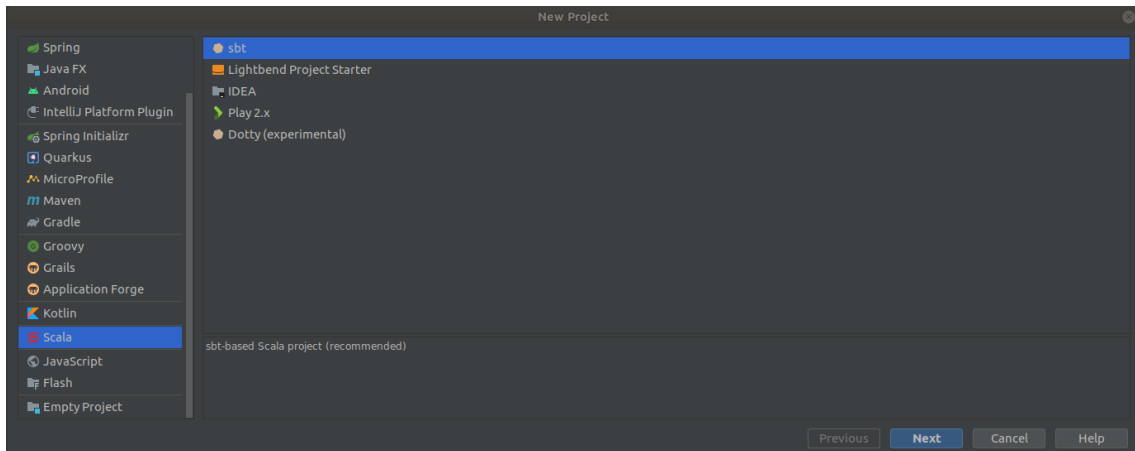
age: int.
workclass: string.
education: String
education-num: int
marital-status: string
occupation: string
relationship: string
race: string
sex: string
capital-gain: string
capital-loss: string
hours-per-week: string
native-country: string
income: string

Para poder hacer uso del código fuente que se provee, hace falta crear un proyecto nuevo de Scala en IntelliJ, lo podemos hacer desde la vista inicial:

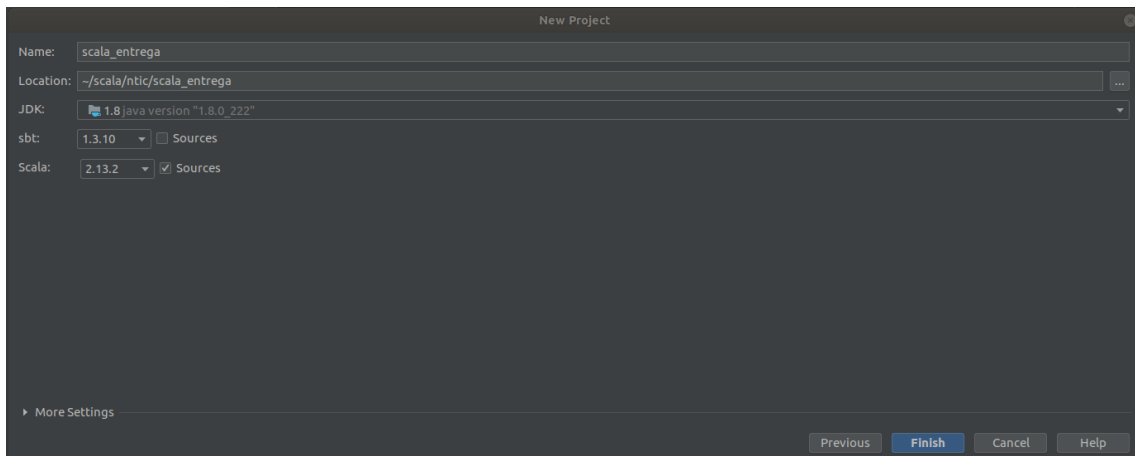


o si lo teníamos previamente abierto: File -> New -> Project.

El proyecto tiene que ser de Scala y con sbt:

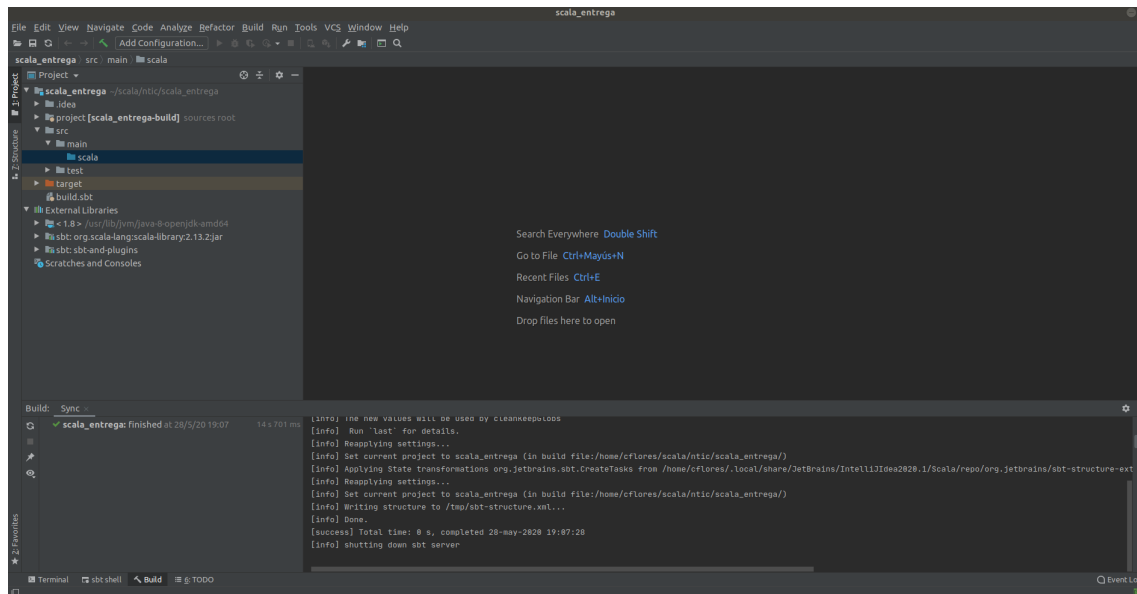


se establece un nombre al proyecto:



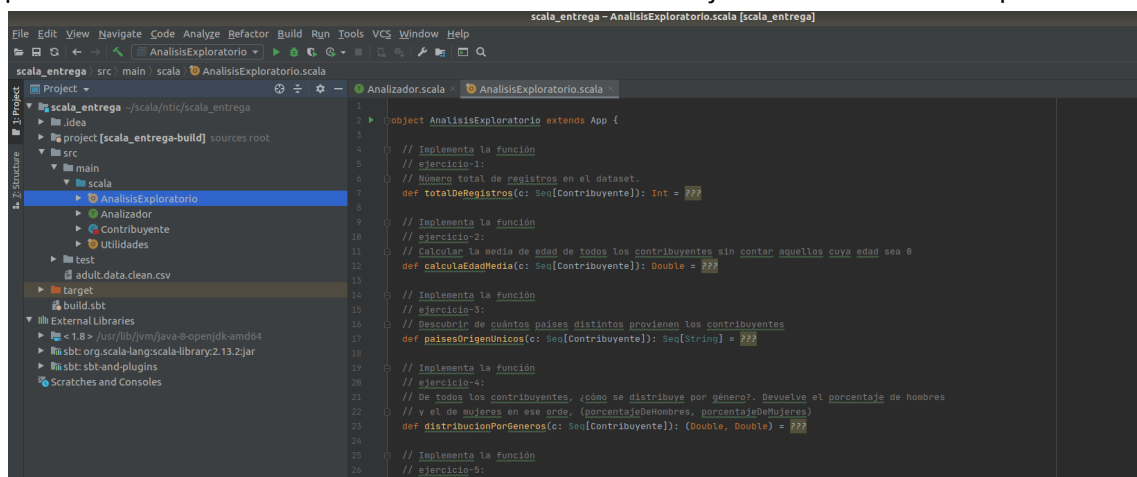
Y una vez establecido el nombre, se puede finalizar clickando sobre el botón Finish

Una vez IntelliJ ha terminado de generar el proyecto:

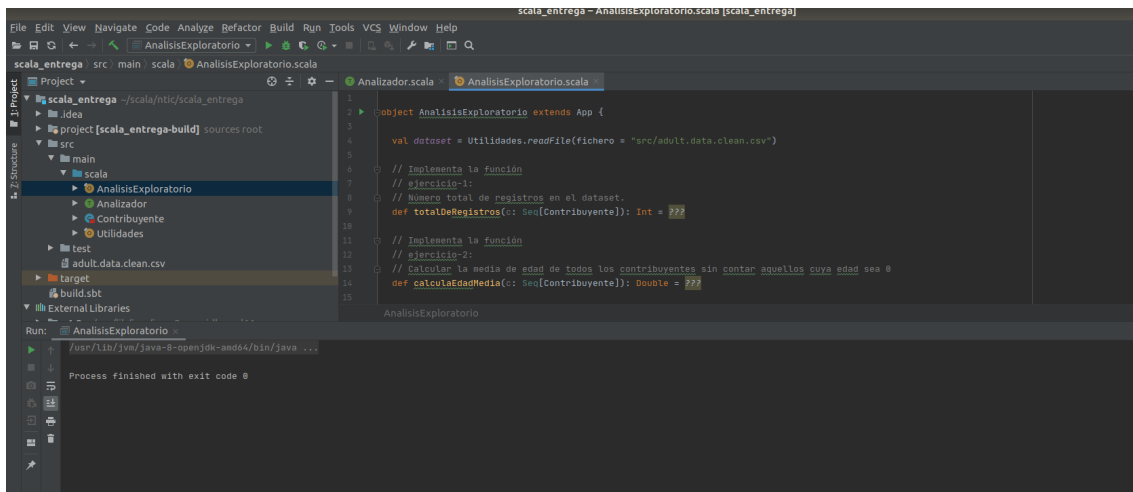


IntelliJ por defecto nos crea una carpeta **src** vacía, esta carpeta debemos reemplazarla por la que se proporciona dentro del fichero src.zip

Una vez reemplazada la carpeta **src** por la que se encuentra dentro del src.zip, podemos hacer una prueba rápida para comprobar que se ha cargado correctamente, para eso abrimos el objeto **AnálisisExploratorio**:



Y lo ejecutamos, para ello podemos darle al botón de Run (situado a la derecha del número de línea), situado a la izquierda de la definición del objeto. Al darle a Run, es probable que aparezcan varias opciones de las cuales debemos elegir Run, esto ejecutará el programa terminará dejando una salida como la siguiente:



```
scala_entrega - AnalisisExploratorio.scala [scala_entrega]
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
scala_entrega src main scala AnalisisExploratorio.scala
Project
  scala_entrega /scala/ntic/scala_entrega
  .idea
  project [scala_entrega-build] sources root
  src
    main
      scala
        AnalisisExploratorio
        Analizador
        Contribuyente
        Utilidades
      test
      adult.data.clean.csv
      target
      build.sbt
  External Libraries
Run: AnalisisExploratorio
  /usr/lib/jvm/java-8-openjdk-amd64/bin/java ...
  Process finished with exit code 0
```

Llegados hasta aquí, ya se puede empezar a desarrollar.

Si exploramos el código de Análisis exploratorio y nos centramos en las líneas de la 39 a la 44, vemos que las llamadas a la función `println` están comentadas. Estas líneas serán descomentadas a la hora de la evaluación de los proyectos. Ya que llamarán a las funciones que se pide que se implementen.

Ejercicios:

1. (0.5 puntos) Número total de registros en el dataset.
2. (0.5 puntos) Calcular la media de edad de todos los contribuyentes.
3. (1 puntos) Calcular la media de edad de todos los contribuyentes sin contar aquellos cuya edad sea 0
4. (1 punto) Descubrir de cuántos países distintos provienen los contribuyentes.
5. (1 punto) De todos los contribuyentes, cómo se distribuye por género? cuál es el porcentaje de hombres? y el de mujeres?
6. (2 puntos)Cuál es el tipo de trabajo (workclass) cuyos ingresos son mayoritariamente superiores a ">50K"
7. (2 puntos)Cuál es la media de años de educación (educationNum) de aquellos contribuyentes cuyo país de origen no es United-States.
8. (2 puntos, 1 cada sub-apartado) Dada la clase 'Contribuyente' (viene en el código fuente del que se parte) y que mapea cada columna del csv, se pide que se cree su companion object y defina la función:

- **imprimeDatos** que muestre por consola el siguiente formato:
"\$workclass - \$occupation - \$nativeCountry - \$income"
 - **apply**, que no reciba ningún parámetro y que devolverá una instancia de la clase Contribuyente con aquellos campos que sean:
 - del tipo Int inicializados a -1
 - del tipo String inicializado a "desconocido"
9. (0.5 puntos. Opcional) Hacer que el objeto AnalisisExploratorio extienda del trait Analizador (provisto en el código) en lugar de App.
10. (1 punto. Opcional) Al extender Analizador, es necesario que se implemente la función `imprimeContribuyente`, esta función tiene que imprimir cada instancia de Contribuyente que contenga la colección que recibe por parámetro, con el siguiente formato: "\$workclass - \$occupation - \$nativeCountry - \$income".
11. (0.5 punto. Opcional) Llama a la función `imprimeContribuyente` al final del programa.

Entrega:

Se enviará el contenido de la carpeta **src** del proyecto con la implementación de las funcionalidades pedidas para cada apartado en un fichero zip cuyo nombre tiene que seguir el siguiente formato: PrimerApellidoSegundoApellidoNombre.zip, por ejemplo: FloresEspinozaCharles.zip

Hay 12 puntos en total en todos los apartados, pero la evaluación será sobre 10.