Pyspark & AWS Glue

Colombu Gabriele (1075574) - Odajiu Romeo (1073807)

Lettura “watch_next_dataset”

Abbiamo creato il primo modello aggregato “**tags_dataset_agg”**, aggiungendo i tag al dataset “**tedx_dataset”**.

Abbiamo salvato i dati in “**watch_next_dataset”**, lo abbiamo ordinato per “**idx**” ed in fine fatto la join con “**tedx_dataset_agg”** in base all’id del video corrispondente.

```
##### READ TAGS DATASET
tags_dataset_path = "s3://tedx-perience-data/tags_dataset.csv"
tags_dataset = spark.read.option("header", "true").csv(tags_dataset_path)

##### CREATE THE AGGREGATE MODEL, ADD TAGS TO TEDX DATASET
tags_dataset_agg = tags_dataset.groupBy(col("idx").alias("idx_ref")).agg(collect_list("tag").alias("tags"))
tags_dataset_agg.printSchema()
tedx_dataset_agg = tedx_dataset.join(tags_dataset_agg, tedx_dataset.idx == tags_dataset_agg.idx_ref, "left") \
    .drop("idx_ref") \
    .select(col("idx").alias("_id"), col("*")) \
    .drop("idx") \

tedx_dataset_agg.printSchema()

##### READ WATCH NEXT DATASET
watch_next_dataset_path = "s3://tedx-perience-data/watch_next_dataset.csv"
watch_next_dataset = spark.read.option("header", "true").csv(watch_next_dataset_path)

##### CREATE THE AGGREGATE MODEL WATCH NEXT
watch_next_dataset_agg = watch_next_dataset.groupBy(col("idx").alias("idx_ref")).agg(collect_set("watch_next_idx").alias("watch_next"))
watch_next_dataset_agg.printSchema()
tedx_dataset_agg = tedx_dataset_agg.join(watch_next_dataset_agg, tedx_dataset_agg._id == watch_next_dataset_agg.idx_ref, "left") \
    .drop("idx_ref") \
```

Calcolo della media delle visualizzazioni

Abbiamo utilizzato dati provenienti da due dataset, nei quali i dati relativi alle visualizzazioni erano leggermente diversi. Abbiamo così deciso di creare una funzione per calcolare la media fra i due dati in nostro possesso.

```
##### READ FILE DATA
tedx_newdataset_path = "s3://tedx-perience-data/data.csv"
tedx_newdataset = spark.read \
    .option("header","true") \
    .option("quote","\"") \
    .option("escape","\"") \
    .option("delimiter", ",") \
    .csv(tedx_newdataset_path)
tedx_newdataset = tedx_newdataset.join(tedx_dataset_agg, tedx_newdataset.title == tedx_dataset_agg.title, "inner")

tedx_newdataset = (
    tedx_newdataset
    .withColumn('num_views', regexp_replace('num_views', ',', ''))
    .withColumn('num_views', col('num_views').cast("int"))
)

tedx_newdataset = (
    tedx_newdataset
    .withColumn('views', regexp_replace('views', ',', ''))
    .withColumn('views', col('views').cast("int"))
)

##### SAVE DATA AS LISTS
num_views = tedx_newdataset.select("num_views").rdd.flatMap(lambda x: x).map(lambda x: int(x) if x else 0).collect()
views =  tedx_newdataset.select("views").rdd.flatMap(lambda x: x).map(lambda x: int(x) if x else 0).collect()
```

Calcolo della media delle visualizzazioni

Funzione "**calc_avg_views**".
Nel caso uno dei due dati fosse nullo, la media delle visualizzazioni viene sostituita con l'unico dato valido fra i due disponibili.

```
############################## FUNZIONI ###################################

def calc_avg_views(num_views, views):
    for a in range(len(num_views)):
        if(num_views[a]==0):
            num_views[a]=views[a]
        if(views[a]==0):
            views[a]=num_views[a]
    return [(a+b)/2 for a, b in zip(num_views, views)]

###########################################################################
```

Aggiornamento del dataset con la media

Abbiamo aggiornato "**tedx_newdataset_complete**" inserendo il numero medio di visualizzazioni, facendo una join dove "**index**" fosse uguale fra gli speech del dataset "**tedx_newdataset**" e "**df_avg_views**".

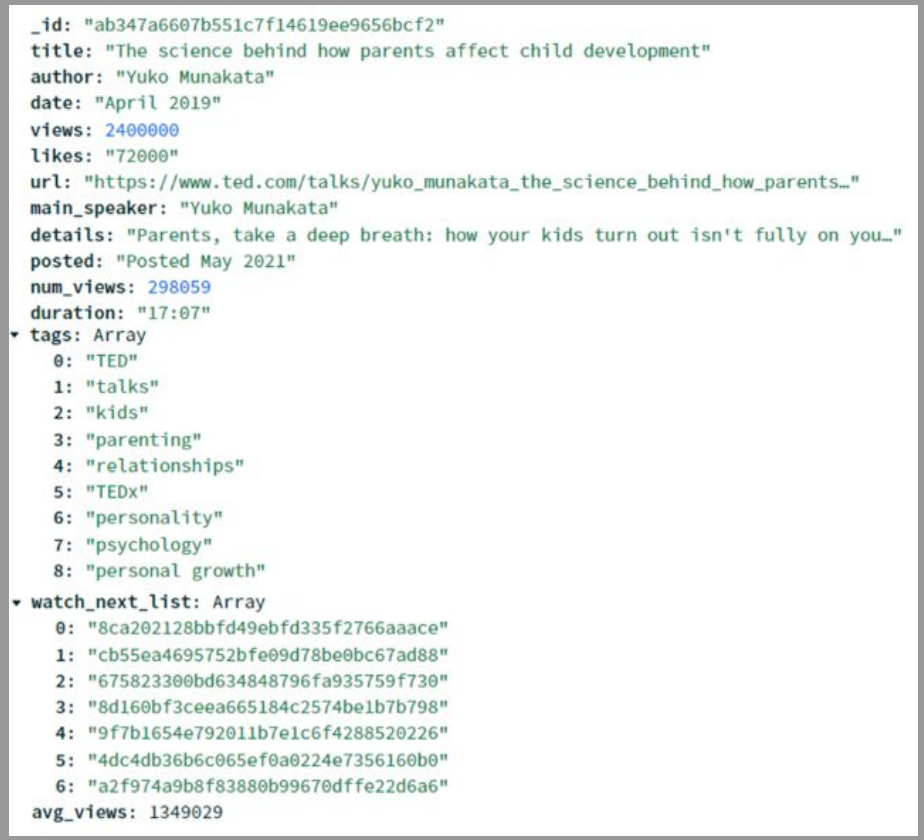
```
##### ADD INDEX TO DATA FRAME TO JOIN
df_avg_views = df_avg_views.withColumn("index", row_number().over(Window.orderBy(monotonically_increasing_id())))
tedx_newdataset = tedx_newdataset.withColumn("index", row_number().over(Window.orderBy(monotonically_increasing_id())))

##### JOIN DATA FRAMES
tedx_newdataset_complete = tedx_newdataset \
    .join(df_avg_views, tedx_newdataset.index == df_avg_views.index, "inner") \
    .drop("index") \
    .withColumnRenamed("idx", "idx_ref")
```

Visualizzazione dei dati in mongoDB

Il dataset in MongoDB si presenta in questo modo:

- _id : identificativo di ogni video.
- views: visual. del nuovo dataset.
- num_views: visual. del dataset fornito a lezione.
- watch_next_list: lista ordinata per n°visual.
- tags: lista dei tag del video.
- avg_views: media visual. calcolata.



Criticità tecniche

1. Formattazione errata dei valori di “num_views” in “tedx_dataset”

Abbiamo eliminato le virgole presenti nei valori del campo “**num_views**” e ri-formattato i valori, convertendoli in numeri interi.

2. dati duplicati presenti in “watch_next_dataset”

Abbiamo utilizzato il comando *collect_set* per eliminare i duplicati presenti nel dataset.

3. Lettura dei dataset

La differenza di formato dei dataset ha implicato la definizione di opzioni di lettura specifiche per per ognuno di essi.

4. Dataset obsoleti

I dataset non vengono periodicamente aggiornati, perciò i dati diventano obsoleti nel tempo.

5. mancanza di dati

In alcuni video il numero di visualizzazioni è pari a 0. Quindi abbiamo dovuto adattare la funzione *media* affinchè ne tenesse conto.

6. Formattazione errata dei valori di “likes” in “tedx_dataset”

Per assicurarci una corretta esecuzione del metodo .sort() nella funzione lambda, abbiamo fatto il cast di “**likes**” ad intero nello script glue.

Link Utili:

TRELLO:

- https://trello.com/invite/b/BKOvfpsB/ATTI5f54a246955ffbff71ce713d520e1ab97DDF7283/tedx-perience

REPOSITORY GITHUB:

- https://github.com/AndreaCremonesi4/TedX-Perience