PySpark job

Michele Dentella - 1063244

analisi dati da importare

come operazione preliminare ho analizzato i dati per avere un’idea del materiale con cui avrei avuto a che fare.

watch_next.csv

idx	url	watch_next_idx

watch_next.csv

idx	altri campi	url	altri campi	tags [array]

su entrambi i dataset ho fatto un’analisi per cercare errori provenienti dallo scraping.

in questo caso le righe scorrette sono state semplicemente rimosse, in un contesto più ampio si potrebbero analizzare per migliorare lo script di scraping.

analisi: tedx_dataset

identificazione campi critici, che devono esistere corretti, per il buon funzionamento dell'app finale: idx, url.

filtro oggetti con url vuoto o mal formattato:

```
count_items = tedx_dataset.count()
count_items_null = tedx_dataset.filter("url like 'http%'").count()

print(f"Number of items from RAW DATA {count_items}")
print(f"Number of items from RAW DATA with NOT NULL KEY {count_items_null}")
print(f"Number of items filtered: {count_items-count_items_null}")
```

```
Number of items from RAW DATA 4494
Number of items from RAW DATA with NOT NULL KEY 4444
Number of items filtered: 50
```

50 oggetti filtrati

filtro finale nel job PySpark:

```
48 #### filter TEDx dataset on url field
49 tedx_dataset_filtered = tedx_dataset.filter("url like 'http%'")
```

analisi: watch_next dataset

Campi critici: idx, watch_next_idx.
L'url non è stato preso perchè già presente nel dataset dei talk, ho preferito mantenere più snello il DB.

Da un'analisi preliminare tramite editor di testo, ho individuato due principali problemi:

- presenza di duplicati
- oggetti con url malformattato (forse rimando ad una playlist?) che ha un id non presente nella lista dei talk

analisi: watch_next dataset: duplicati

drop duplicate:

```
#### DROPPING DUPLICATE

wn_count = wn_dataset.count()
print(f"Pre drop duplicated: {wn_count}")
wn_dataset = wn_dataset.dropDuplicates()
print(f"After drop duplicated: {wn_dataset.count()}")
print(f"Dropped: {wn_count-wn_dataset.count()}")
```

```
Pre drop duplicated: 77364
After drop duplicated: 30254
Dropped: 47110
```

rimossi 47110 duplicati

analisi: watch_next dataset: url mal formattato

filter bad url:

```
#### FILTER BAD URL

count_items = wn_dataset.count()
count_items_good = wn_dataset.filter("url not like 'https://www.ted.com/session/new?%'").count
()

print(f"Number of items PRE {count_items}")
print(f"Number of items AFTER {count_items_good}")
print(f"Number of items filtered: {count_items-count_items_good}")

#### bad url row has a watch_next_id that's not in talk id list - can be safely removed, no inf
o lost
bad_id = wn_dataset.filter("url like 'https://www.ted.com/session/new?%'").select("watch_next_i
dx").dropDuplicates()
print(f"different bad id: {bad_id.count()}")
print(f"bad watch_next_id: ")
bad_id.show(1, False)
#### search bad_id in talks
print(f"search for bad_id in talks: ")
tedx_dataset_agg.filter("idx = '9f7b1654e792011b7e1c6f4288520226'").show()

#### filter bad id
wn_dataset = wn_dataset.filter("url not like 'https://www.ted.com/session/new?%'")
```

```
Number of items PRE 30254
Number of items AFTER 25788
Number of items filtered: 4466
different bad id: 1
bad watch_next_id:
+--------------------------------+
|watch_next_idx                  |
+--------------------------------+
|9f7b1654e792011b7e1c6f4288520226|
+--------------------------------+

search for bad_id in talks:
+---+------------+-----+-------+------+---+---------+----+
|idx|main_speaker|title|details|posted|url|num_views|tags|
+---+------------+-----+-------+------+---+---------+----+
+---+------------+-----+-------+------+---+---------+----+
```

- dopo alcune prove ho deciso di usare un filtro “positivo” - mantiene gli oggetti con url ben formattato
- controllo quali id sto escludendo
  - l’id è uno solo
  - non è presente tra i talks
  - posso rimuoverlo senza perdere informazioni utili

schema finale

```
tedx_dataset.printSchema()
```

```
root
 |-- _id: string (nullable = true)
 |-- main_speaker: string (nullable = true)
 |-- title: string (nullable = true)
 |-- details: string (nullable = true)
 |-- posted: string (nullable = true)
 |-- url: string (nullable = true)
 |-- num_views: string (nullable = true)
 |-- tags: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- watch_next_ids: array (nullable = true)
 |    |-- element: string (containsNull = true)
```


```
_id: "8d2005ec35280deb6a438dc87b225f89"
main_speaker: "Alexandra Auer"
title: "The intangible effects of walls"
details: "More barriers exist now than at the end of World War II, says designer..."
posted: "Posted Apr 2020"
url: "https://www.ted.com/talks/alexandra_auer_the_intangible_effects_of_wal..."
tags: Array
  0: "TED"
  1: "talks"
  2: "design"
  3: "society"
  4: "identity"
  5: "social change"
  6: "community"
  7: "humanity"
  8: "TEDx"
watch_next_ids: Array
  0: "fe35edd737282ab3a325f2387cf1b50b"
  1: "8576654442b6633b1dc0eb48a989172a"
  2: "5134ae81a27c94354173f38e84289ad5"
  3: "5bd34fcc55d9e1267f605fa0c060d54e"
  4: "d9896b41b372ec60cdd3c662e57caad3"
  5: "078766d6cc461cf71d45dc268b66db95"
```

job PySpark

per evitare le lunghe attese dovute all'esecuzione su AWS, ho usato il container jupyter/pyspark-notebook presentato da Cattaneo e Locatelli.

Il file jupyter con l'analisi sui dati si trova nel repo github contenente tutto il materiale del secondo compito: https://github.com/abeteverde/laboratorio-TCM

criticità

- attese esecuzione su aws
- pulizia linee con errori nei file .csv
- duplicati
- strutturare il dataset senza avere un’idea precisa dell’app finale (e degli utilizzi di questa)

possibili evoluzioni

- maggiori controlli sulla qualità dei dati
  - analizzare righe scorrette provenienti dallo scraper
  - filtri sui dati importati nel db per garantire maggiore robustezza