Auto Scout24 scraping

Simone Zambetti

Background economico

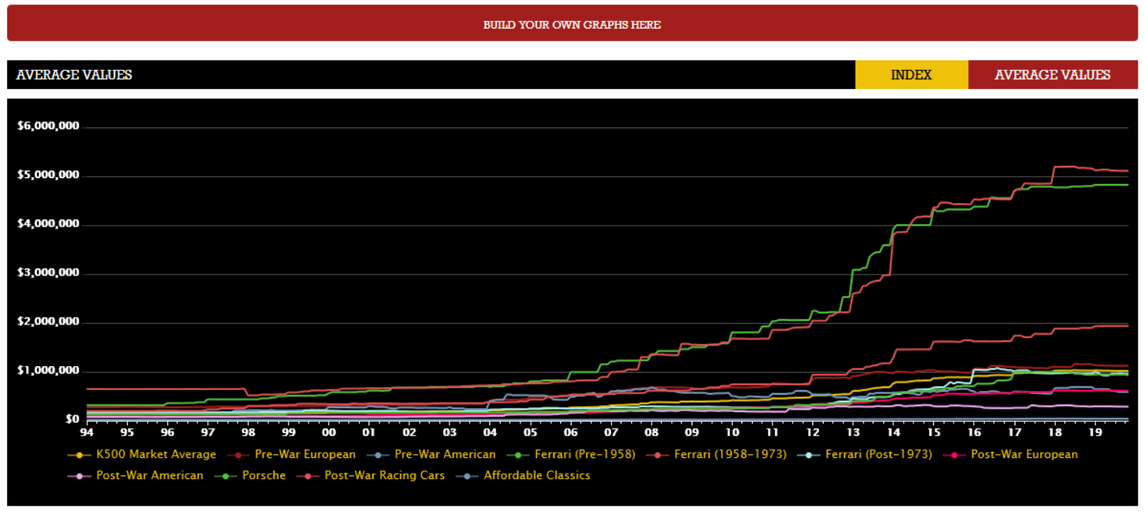
3/7/2020

BI&BDA

Github link

Business target

- Target: scraping degli annunci di autoscout24, il portale leader in italia nel commercio di autovetture nuove o usate
- Focalizzandosi sul mercato emergente dei veicoli elettrici (EV), recuperare tutte le informazioni disponibili sulle pagine degli annunci elaborando anche i singoli annunci
- La base dati risultante può essere eventualmente usata per risolvere le seguenti domande di business, ad esempio:
    - E' vero che i deprezzamento delle auto elettriche inferiore rispetto alle endotermiche? Una domanda particolarmente interessante visto che, intuitivamente, il tasso di evoluzione tecnologico delle EV è molto più veloce oggi rispetto alle ICE (interal combustion engine), dunque si ci aspetterebbe il contrario.
    - Quale è lo share di mercato dei diversi marchi?
    - Quali variabili influenzano il valore di una vettura elettrica?
    - Come stanno influenzando gli ingenti incentivi statali e regionali il pricing delle vetture elettriche?
    - Costruire un indice di prezzo di mercato per vari modelli, come sottostante:

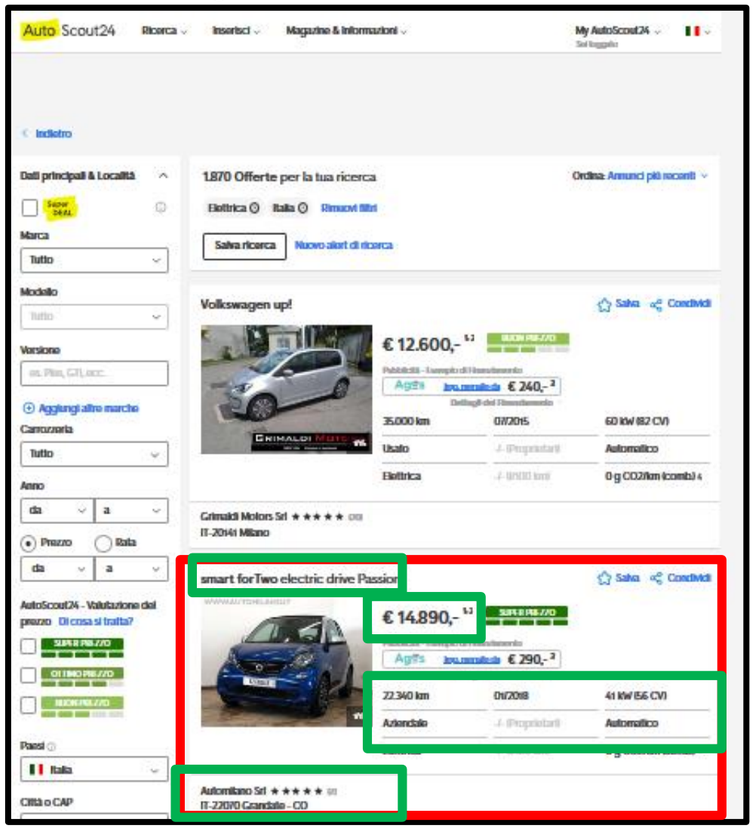


Indice di mercato per le vetture classiche k500, più info gui

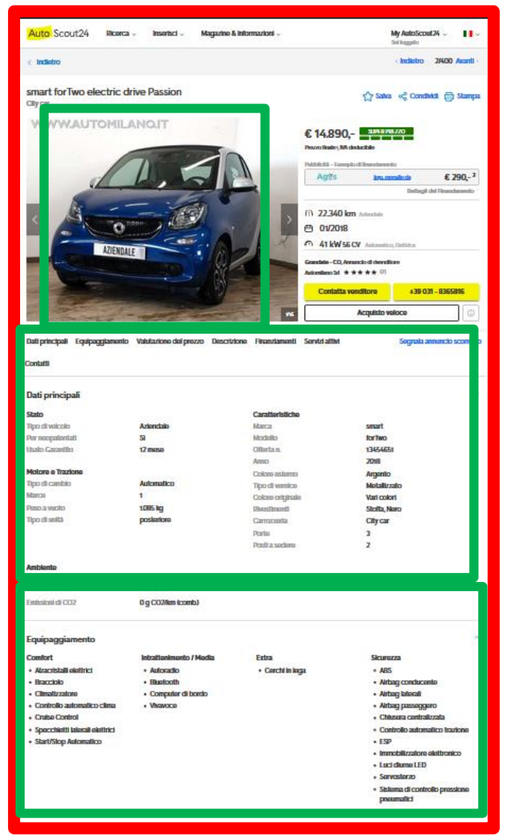
Scraping

- Partendo dalla pagina principale riferimento annunci, già filtrata per EV (fuel=E), massimo valori per pagina 20, paese italia, sorted by più recenti. Link di esempio: https://www.autoscout24.it/lst?sort=age&desc=1&fuel=E&ustate=N%2CU&size=20&page=1&cy=I (solo le prime 20 pagine e annunci correlati sono stati scaricati per ridurre cardinalità)
- Utilizzando **colab**, tutte le informazioni nei riquadri verdi sono «scraped» :
  - Dati non strutturati: in alcune annuncio abbiamo caratteristiche/descrizioni che possono essere completamente mancati in altri
  - ci sono anche immagini
  - Alcuni campi sono ripetuti, quindi non è necessario che vengano scaricati anche dalle singole pagine degli annunci

Pagina di ricerca annunci



Singolo annuncio



Scraping (pt. II)

- Output dello scraping:
- 2 file json

Git | export_output_scraping.json

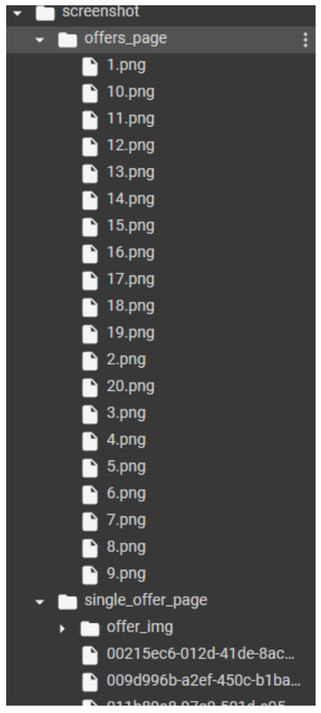

```
[
  {
    "id_annuncio": "084037c1-3fbe-48ce-b5b2-f1a278522e22",
    "link_annuncio": "https://www.autoscout24.it/annunci/smart-fortwo-electric-drive-coupe-elettrica-nero-084037c1-3fbe-48ce-b5b2-f1a278522e22?cldtidx=1&cldtsrc=listPage",
    "vehicle": "smart forTwo",
    "vehicle_user_desc": "electric drive coup\u00e9",
    "asking_price": "\u20ac 9.000,- 12",
    "vehicle_data": [
      "28.015 km",
      "02/2014",
      "35 kW (48 CV)",
      "Usato",
      "1 proprietario",
      "Automatico",
      "Elettrica",
      "-/- (l/100 km)",
      "0 g CO2/km (comb.) 4"
    ],
    "seller": "Unique Car Srl",
    "country": "IT",
    "address": "00166 Roma - Rm"
  },
  {
    "id_annuncio": "9dc886a9-480b-d914-e053-0100007f3eeb",
    "link_annuncio": "https://www.autoscout24.it/annunci/smart-fortwo-eq-edition-one-22kw-elettrica-grigio-9dc886a9-480b-d914-e053-0100007f3eeb?cldtidx=2&cldtsrc=listPage",
```


Git | export_output_single_offers_scraping.json


```
[
  {
    "id_annuncio": "084037c1-3fbe-48ce-b5b2-f1a278522e22",
    "user_description": "vettura usata ottimo stato aziendale
    "equipment": [
      "Alzacristalli elettrici",
      "Climatizzatore",
      "Autoradio",
      "Cerchi in lega",
      "ABS",
      "Airbag conducente",
      "Airbag laterali",
      "Airbag passeggero",
      "Chiusura centralizzata",
      "Controllo automatico trazione",
      "ESP",
      "Fendinebbia",
      "Immobilizzatore elettronico",
      "Servosterzo"
    ],
    "specs": {
      "Tipo di veicolo": "Usato",
      "Per neopatentati": "S\u00ec",
      "Proprietari": "1",
      "Usato Garantito": "12 mese",
      "Tagliandi certificati": "S\u00ec",
      "Veicolo per non fumatori": "S\u00ec",
      "Marca": "smart",
      "Modello": "forTwo",
```


- Immagini annunci

+ screenshot delle pagine:



BI&BDA

Extraction

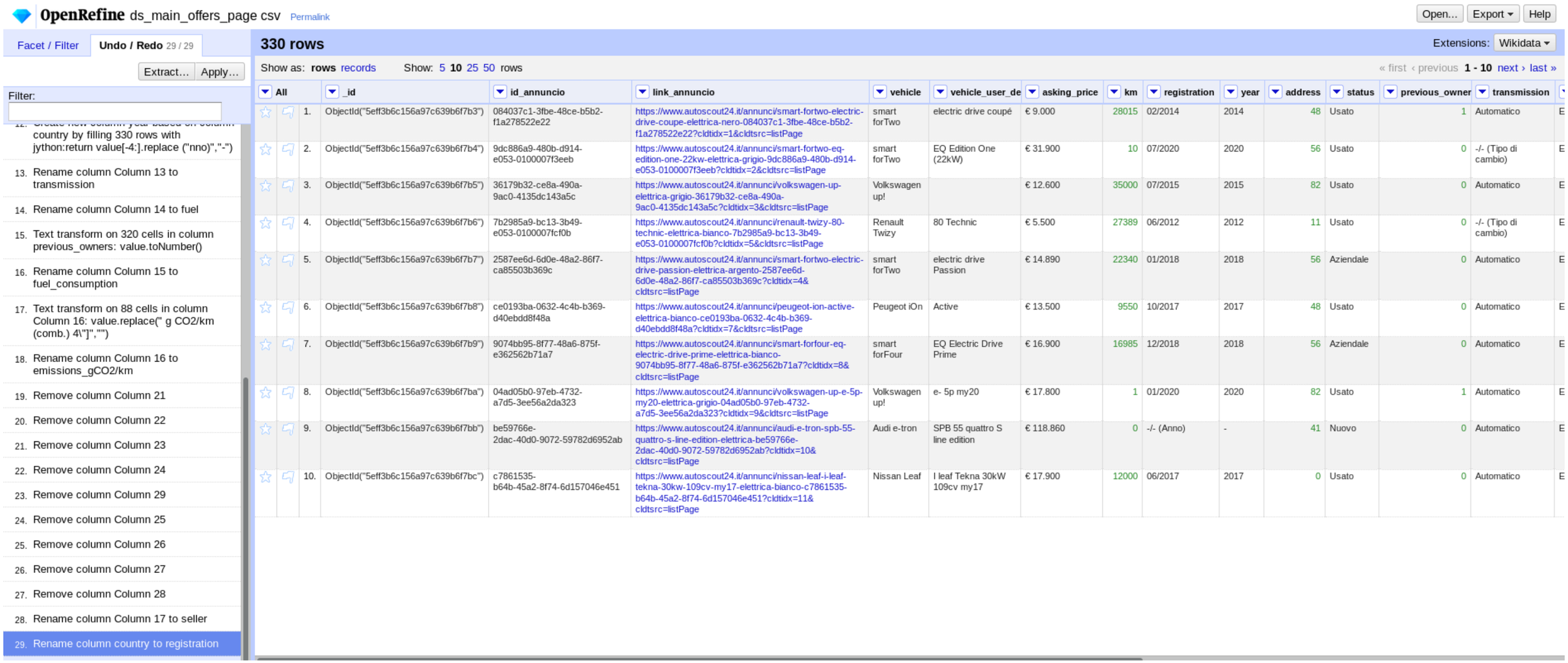
- I json sono caricati tramite shell su MongoDB, nel database autoscout_scraping, creando 2 collections con il prefix ds_*

```
#!/bin/bash

cd /media/sf_Condivisa #go to mounted share drive of master if running on virtual machine
sudo service mongod start
mongo --eval "use autoscout_scraping" #crea database

#import 1mo json in mongo
mongoimport --db autoscout_scraping --collection ds_main_offers_page --file export_output_scraping.json --jsonArray
mongo --eval "db.sc_main_offers_page.find({})" autoscout_scraping

#import 2ndo json in mongo
mongoimport --db autoscout_scraping --collection ds_single_offers_page --file export_output_single_offers_scraping.json --jsonArray

mongo --eval "db.sc_single_offers_page.find({})" autoscout_scraping
#the i use Mongodb compass to export to csv to process with OpenRefine as it is lengthy to specify all fields in bash
```

- MongoDB è stato montato su una VPS debian hostata su aruba.it
- Una volta caricate, le collections sono esportate tramite Mongo Compass in *.csv poichè possano essere letti correttamente da OpenRefine, visto che non riesce a leggere .json nel formato corretto

Transformation (pt. I)

- Transformazione su OpenRefine dei csv esportati:

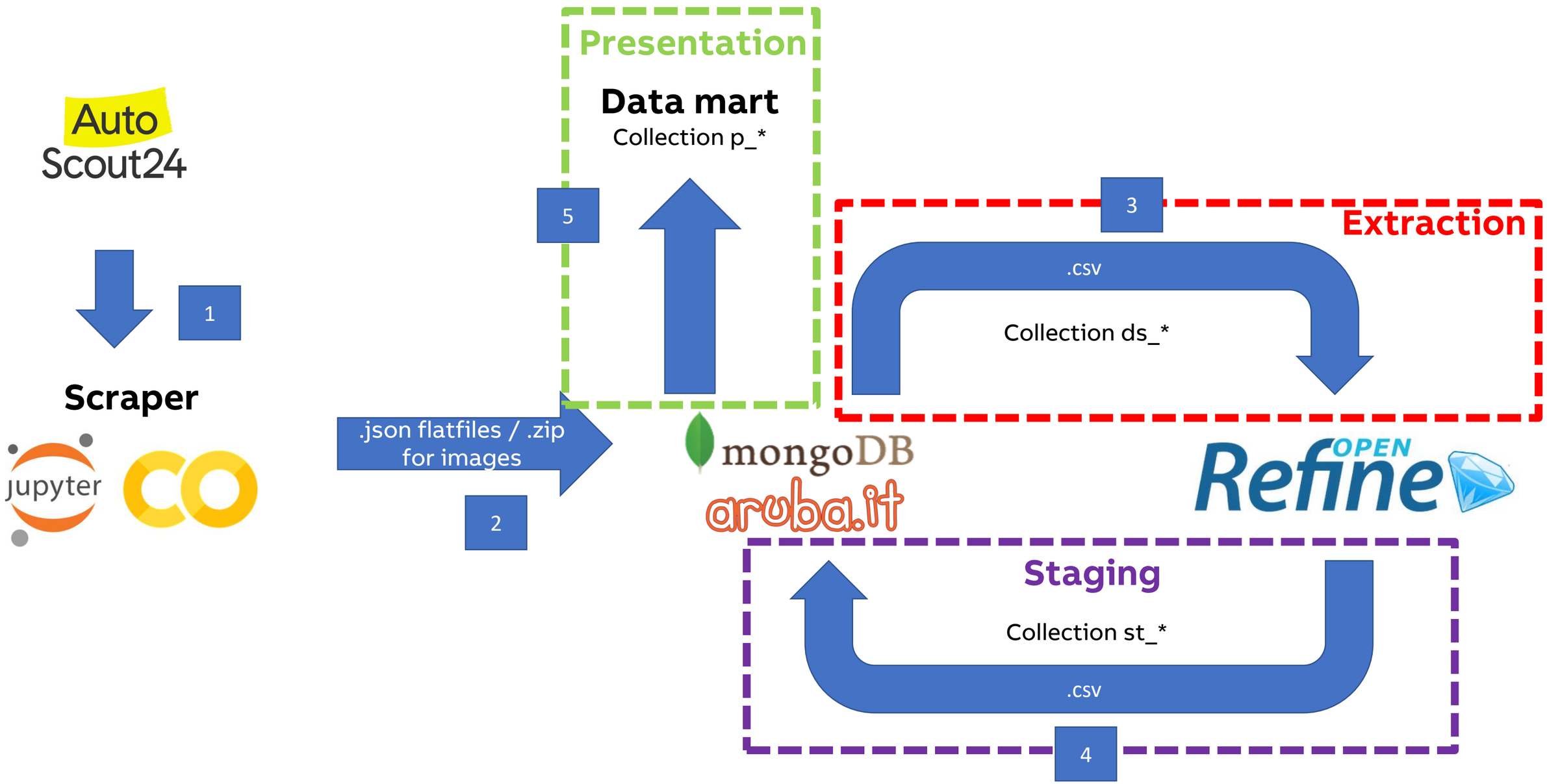


Transformation (pt. II)

- Ad esempio, i seguenti problem sono stati risolti:
  - Annunci con id duplicate con facet (sono tutti singoli)
  - Replacement heading e cleanup dei valori in alcune colonne, ad esempio classe emissioni, campo proprietary conversion a numerico
  - Alcuni annunci non sono auto elettriche (errori di input da parte degli utenti), per risolvere questo problema le seguenti logiche sono stati applicate:
    - Filtrati solo gli annunci di veicoili immatricolati dopo il 2005
    - Esclusi annunci che indicavano emissioni > 0
    - Escluse alcune marche che al momento non possiedono veicoli elettrici puri (e.g. fiat)

- Una volta ultimati questi cambiamenti, i 2 file .csv sono stati esportati su MongoDB nuovamente, su collections con prefisso st_*

Overall architecture



BI&BDA

Migliorie e criticità

- Possibili migliorie:
  - Automatizzare il lancio dello scraper ogni giorno, possibilmente passandolo alla VPS di aruba da Colab
  - Automatizzare in python il passaggio di json -> csv e il processing di OpenRefine
  - Usare le restful API di autoscout, chiedendo account developer per avere accesso a tutte le pagine (per il momento lo scraper scarica solo le prime 20 pagine, un workaround sarebbe scaricare solo gli annunci pubblicati in giornata, per cui autoscout ha un filtro)
  - Utilizzare tutte le immagini delle annunci scaricati

- Criticità:
  - La struttura dell'output dello scraper andrebbe cambiato in modo tale da non convertire in csv i file in json
  - Anche se gli id annunci sono diversi, al momento non vi è un modo per capire 2 o più annunci siano effetivamente duplicati (ad esempio: carico la stessa offerta in molti annunci per avere migliore visibilità)

Github

- Tutto il materiale è disponibile sulla repo:

https://github.com/szambetti/Data-science-Master-UNIMIB/tree/master/Module%204%20-%20Big%20Data%20%26%20Analytics/Homework