Miras Mels

SE-2109

Endterm Project

https://github.com/laqube/endterm-parser

Source code:

```
import requests
import json
from bs4 import BeautifulSoup
import urllib3
import time
dataset = []  #An empty dataset
counter = 0   #A document counter
http = urllib3.PoolManager()
for p in range(1, 24):
    if p == 1:
        url = 'https://www.olx.kz/d/kk/dom-i-sad/mebel/mebel-dlya-vannoy-komnaty'                #First page of the category
    else:
        url = 'https://www.olx.kz/d/kk/dom-i-sad/mebel/mebel-dlya-vannoy-komnaty/?page=' + str(p) #Rest of the pages
    response = http.request('GET', url)
    print("Status: " + str(response.status))
    print("!____LESGOOOOOOOOOOOOOOOOOOOOOOOOO____!")     #indication of successful connection
    print("!_____________A NEW PAGE_____________!")
    print("PAGE NUMBER: " + str(p))                       #page counter
    soup = BeautifulSoup(response.data, "html.parser")
    names = soup.find_all("a", {"class": "css-rc5s2u"})     #all the links for certain products located on that page
    for name in names:
        purl = 'https://www.olx.kz/d/kk/obyavlenie' + name.get('href')
        print('PRODUCT+1_____________________')         #indication of successfull connection to a product page
        newresponse = http.request('GET', purl)
        newsoup = BeautifulSoup(newresponse.data, "html.parser")
  #I used try-except block to ignore errors in case if one of the variables is null or of wrong datatype
        try:
            pName = newsoup.find("h1", {"class": "css-1soizd2 er34gjf0"}).get_text()
        except ValueError:
            pName = "None"
        try:
            pPrice = int("".join(newsoup.find("h3", {"class": "css-ddweki er34gjf0"}).get_text().split()[:-1]))
        except ValueError:
            pPrice = 0
        pTags = newsoup.find("ul", {"class": "css-sfcl1s"})
        wTag = pTags.find_all("p")
        try:
            pTags = [i.get_text() for i in wTag]
        except:
            pTags = []
        product = {
            "Product_Name": pName,
            "Price": pPrice,
            "Tags": pTags
        }                                               #The structure of single document
        print(product)
        dataset.append(product)
        counter += 1
    time.sleep(1)                                       #Timer to avoid bot detection, actually in my case I believe it was useless
print(dataset)
print("BOLDY_____________________")                     #Indicator of successful completion of scrabbing
print("Product count: " + str(counter))                 #Document counter
print("ADIHAT_____________________")
with open("olx_dataset.json", "w") as outfile:          #Creating a json file
    json.dump(dataset, outfile, ensure_ascii=False)     #ensure_ascii=False is used in order to decode the dataset right away
```

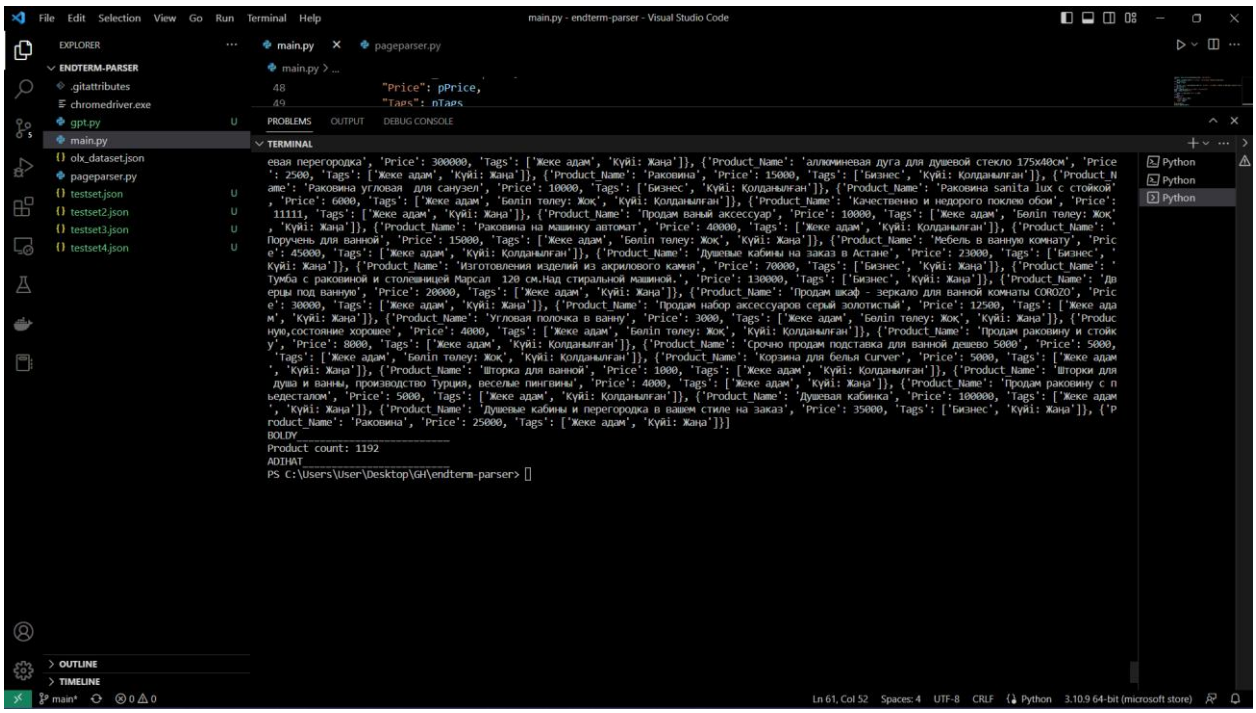
Result dataset.

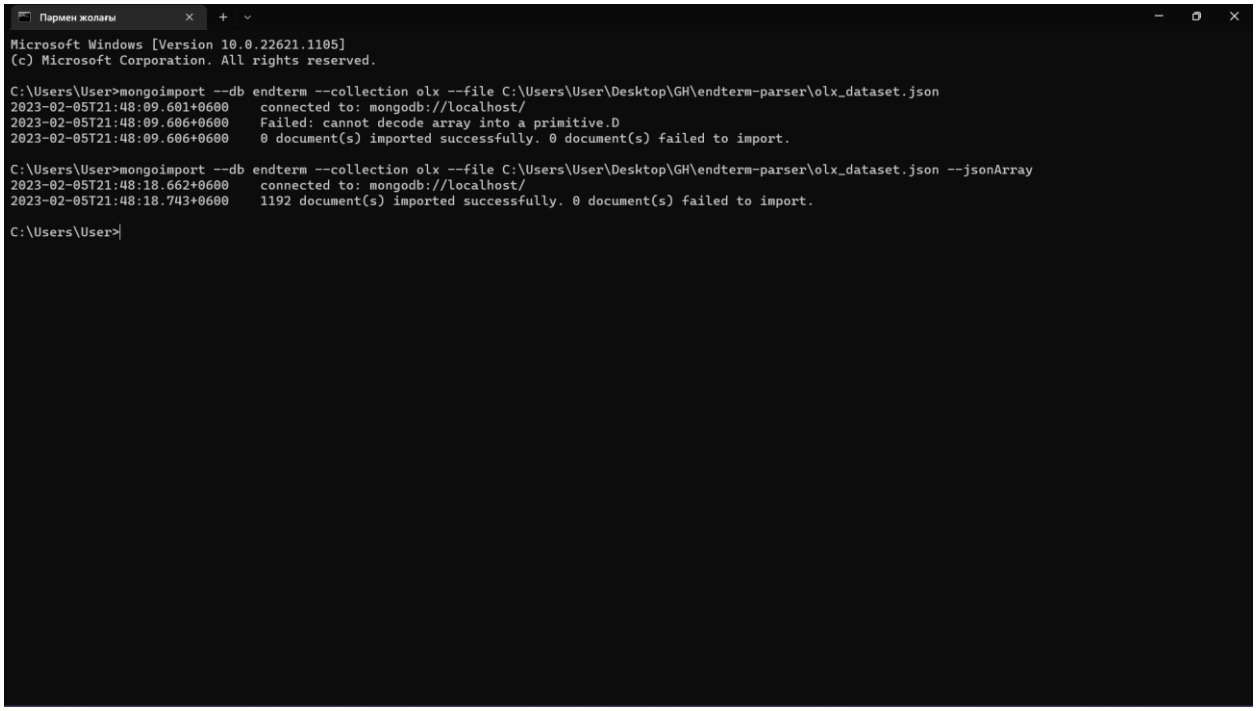
Dataset holds 1192 records of products published on olx.kz in a category “Furniture for the bathroom”. Document about a product holds the next structure:

Name of the product

Price

Tags(in tags, there is information about who is selling the product, its state and sometimes wether if it is possible to pay separately for the product).





I have successfully scrabbed the data and imported the json file into mongodb database, now I can work with it freely.

Analysis of the dataset

There are 378 offers from businesses, and 814 offers from ordinary people:

```
endterm> db.olx.count({ "Tags": "Жеке адам" })
814
endterm> db.olx.count({ "Tags": "Бизнес" })
378
```

557 of products have been previously used, and 635 are new:

```
endterm> db.olx.count({ "Tags": "Күйі: Қолданылған" })
557
endterm> db.olx.count({ "Tags": "Күйі: Жаңа" })
635
```

Now, lets find out how many businesses offer used products and how many ordinary people offer new products:

```
endterm> db.olx.count({Tags: {$all: ["Бизнес", "Күйі: Қолданылған"]}})
31
endterm> db.olx.count({Tags: {$all: ["Жеке адам", "Күйі: Жаңа"]}})
288
```

I am curious about he option of splitting the bill:

```
endterm> db.olx.count({ "Tags": "Бөліп төлеу: Жоқ" })
310
endterm> db.olx.count({ "Tags": "Бөліп төлеу: Бар" })
84
```

And is there any people who offer such option?

```
endterm> db.olx.count({Tags: {$all: ["Бөліп төлеу: Бар", "Жеке адам"]}})
30
endterm> db.olx.find({Tags: {$all: ["Бөліп төлеу: Бар", "Жеке адам"]}}).limit(5)
[
  {
    _id: ObjectId("63dfcfc229a14e3c87d3713c"),
    Product_Name: 'Ванная комната новая',
    Price: 250000,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Бар', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d3713e"),
    Product_Name: 'Распродажа мебели от 8.000тг',
    Price: 8000,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Бар', 'Күйі: Қолданылған' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37142"),
    Product_Name: 'Ванная комната новая',
    Price: 250000,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Бар', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37153"),
    Product_Name: 'Продам стеклянную шторку для ванны',
    Price: 20000,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Бар', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d3717a"),
    Product_Name: 'Распродажа мебели от 8.000тг',
    Price: 8000,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Бар', 'Күйі: Қолданылған' ]
  }
]
```

How cool is that? Some people even sell used products with option of splittin the bill. But it seems like some of them are businesses trying to look like people. And we can prove it also by repitition of data. They shamelessly publish several offers on the website.

Those are top 5 most expensive offers:

```
endterm> db.olx.find().sort({Price: -1}).limit(5)
[
  {
    _id: ObjectId("63dfcfc229a14e3c87d37253"),
    Product_Name: 'Мини сауна (Финская) для квартиры',
    Price: 950000,
    Tags: [ 'Жеке адам', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d372a1"),
    Product_Name: 'Набор для ванной комнаты',
    Price: 880000,
    Tags: [ 'Жеке адам', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37287"),
    Product_Name: 'Продадим 2 угловых джакузи в хорошем состоянии',
    Price: 330000,
    Tags: [ 'Жеке адам', 'Күйі: Қолданылған' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37205"),
    Product_Name: 'Продадим 2 угловых джакузи в хорошем состоянии',
    Price: 330000,
    Tags: [ 'Жеке адам', 'Күйі: Қолданылған' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37166"),
    Product_Name: 'Продадим 2 угловых джакузи в хорошем состоянии',
    Price: 330000,
    Tags: [ 'Жеке адам', 'Күйі: Қолданылған' ]
  }
]
```

And Those are top 5 most cheapest offers:

```
endterm> db.olx.find({Price: {$ne: 0}}).sort({Price: 1}).limit(5)
[
  {
    _id: ObjectId("63dfcfc229a14e3c87d3756e"),
    Product_Name: 'Коврик в ванну, душ 70х35см',
    Price: 500,
    Tags: [ 'Жеке адам', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d3736d"),
    Product_Name: 'Сиденье для ванны',
    Price: 500,
    Tags: [ 'Жеке адам', 'Бөліп төлеу: Жоқ', 'Күйі: Қолданылған' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37246"),
    Product_Name: 'Продам зеркало ручной работы!',
    Price: 500,
    Tags: [ 'Жеке адам', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d3751a"),
    Product_Name: 'Бачок  с крышкой . Два за 1500 т.',
    Price: 700,
    Tags: [ 'Жеке адам', 'Күйі: Қолданылған' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37279"),
    Product_Name: 'Продам Карниз для ванной комнаты',
    Price: 700,
    Tags: [
      'Жеке адам',
      'Бөліп төлеу: Жоқ, Барлық хабарландырулар',
      'Күйі: Жаңа'
    ]
  }
]
```

And those are two offers without a certain price:

```
endterm> db.olx.find({Price: 0})
[
  {
    _id: ObjectId("63dfcfc229a14e3c87d372d2"),
    Product_Name: 'Полки для Ванную LOFT',
    Price: 0,
    Tags: [ 'Бизнес', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63dfcfc229a14e3c87d37532"),
    Product_Name: 'Тройка ,раковина',
    Price: 0,
    Tags: [ 'Жеке адам', 'Күйі: Қолданылған' ]
  }
]
```

Enough of words, lets see some graphs!

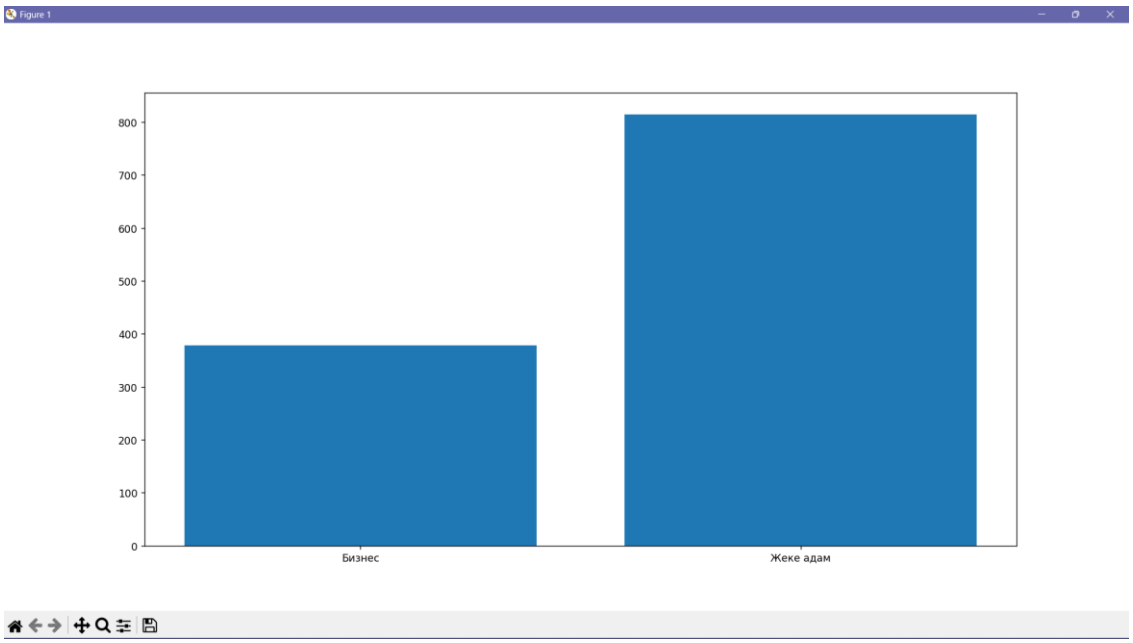


Fig.1 – The count of business offers and people’s offers

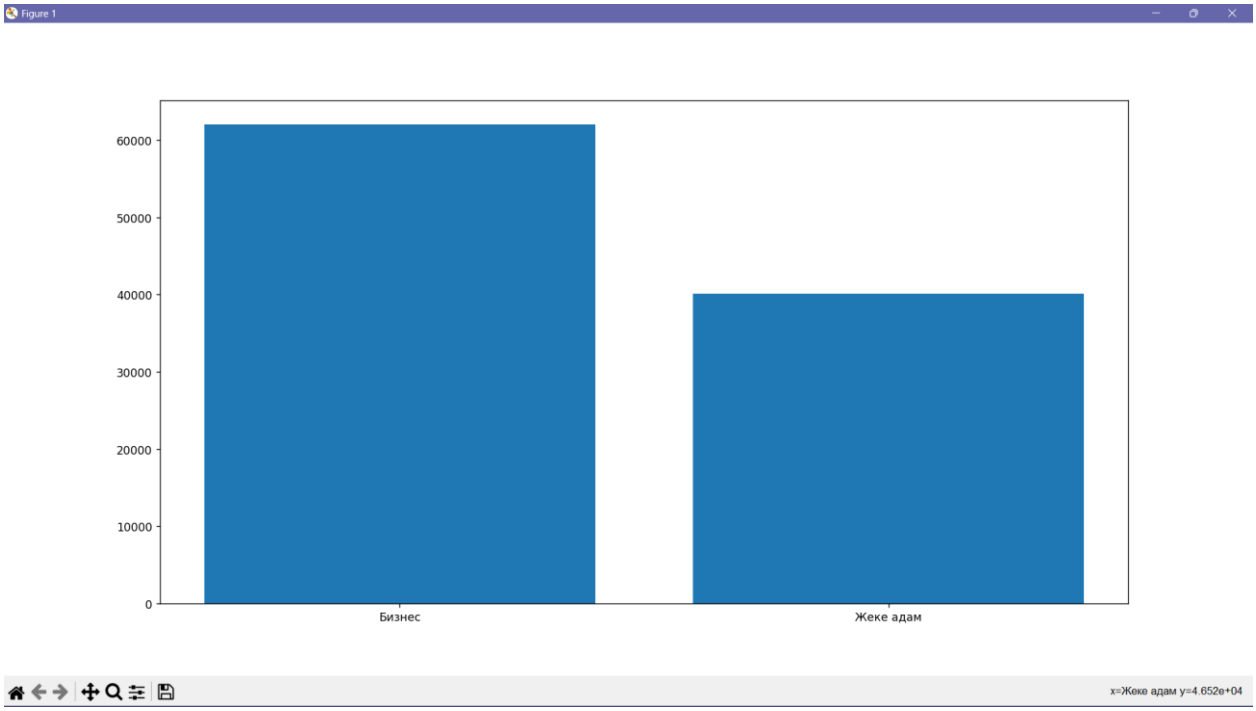


Fig. 2 – Average price for business category and the offers of regular people

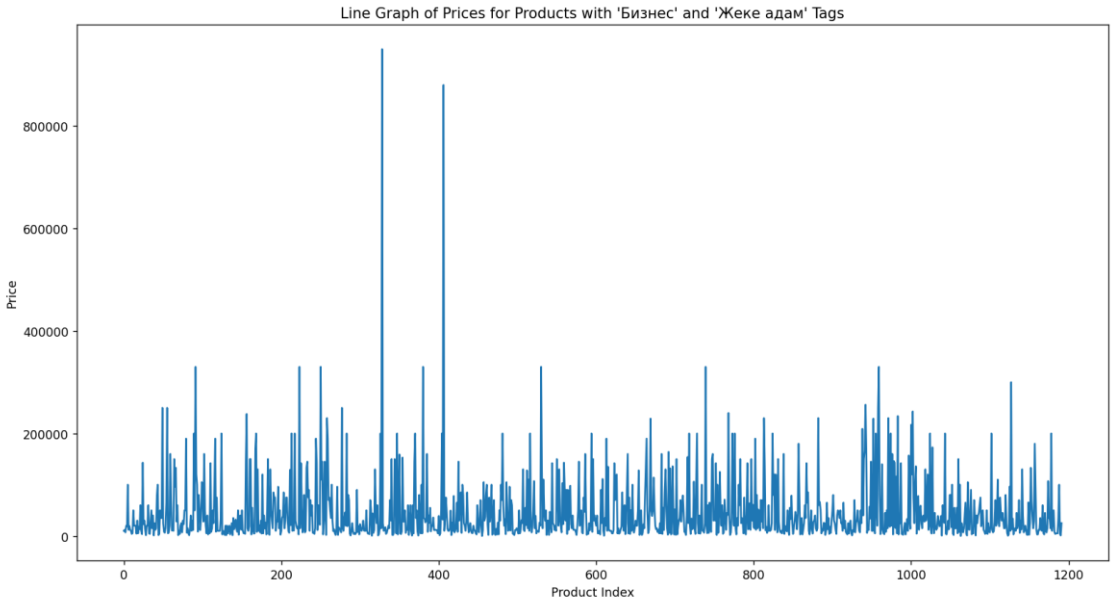


Fig. 3 The dynamic of prices of all products, we can see couple of anomalies

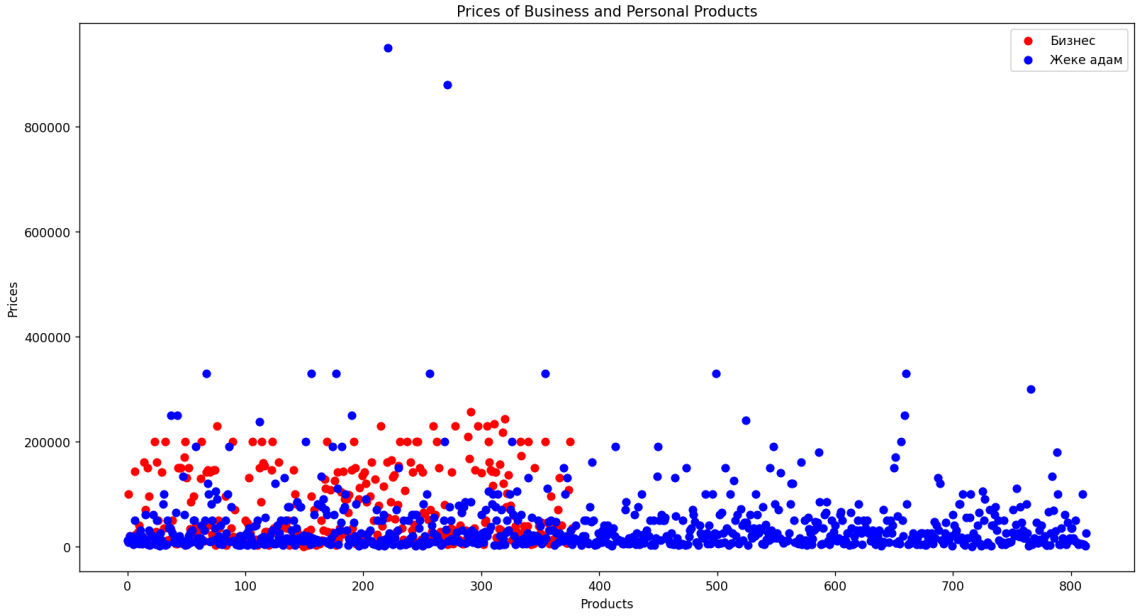


Fig. 4 From this diagram we can see how far can prices go for business offers and personal offers

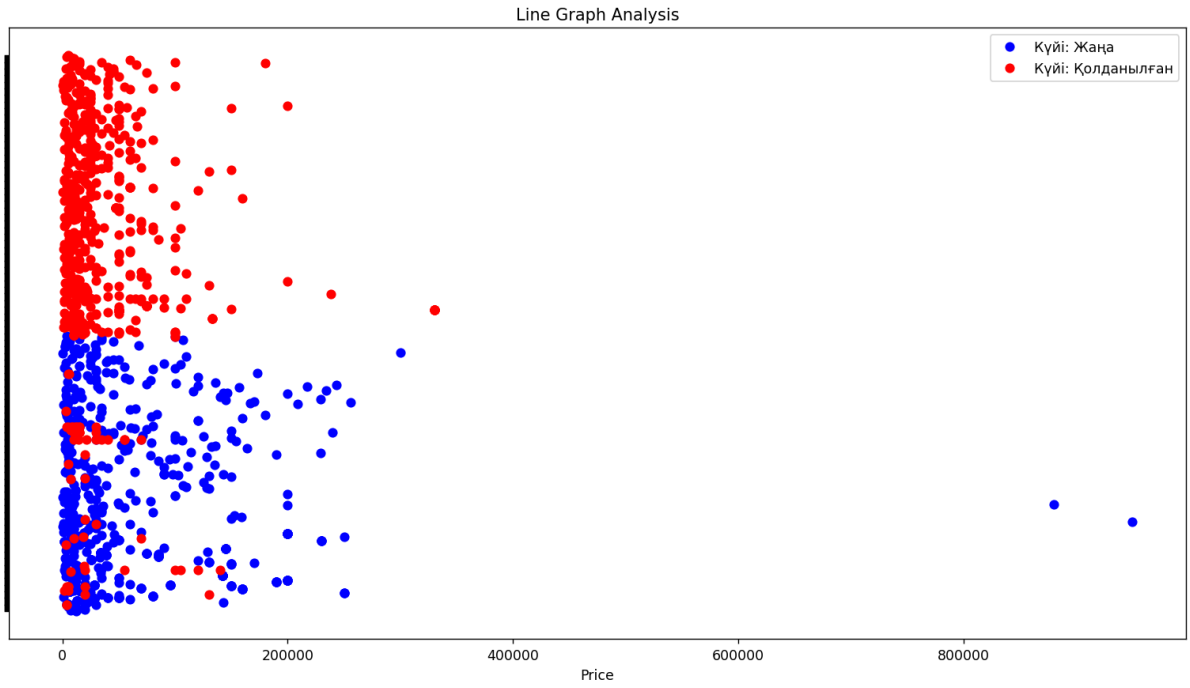


Fig. 5 Prices for New items and Used ones.

CRUD operations in MongoDB

Create/Read

```
endterm> db.olx.insertOne({Product_Name: "Раковина Non-Новая", Price: 66700, Tags:["Бизнес","Күйі: Жаңа"]})
{
  acknowledged: true,
  insertedId: ObjectId("63e3b1b8979cbcb21429445c")
}
endterm> db.olx.insertMany({Product_Name: "Раковина Сып-Свежая", Price: 66700, Tags:["Бизнес","Күйі: Жаңа"]},{Product_Name: "Душ Душевний", Price: 66700, Tags:["Жеке адам", "Күйі: Жаңа"]})
MongoInvalidArgumentError: Argument "docs" must be an array of documents
endterm> db.olx.insertMany([{Product_Name: "Раковина Сып-Свежая", Price: 66700, Tags:["Бизнес","Күйі: Жаңа"]},{Product_Name: "Душ Душевний", Price: 66700, Tags:["Жеке адам", "Күйі: Жаңа"]}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63e3b258979cbcb21429445d"),
    '1': ObjectId("63e3b258979cbcb21429445e")
  }
}
endterm> db.olx.find({Price: 67000})

endterm> db.olx.find({Price: 66700})
[
  {
    _id: ObjectId("63e3b1b8979cbcb21429445c"),
    Product_Name: 'Раковина Non-Новая',
    Price: 66700,
    Tags: [ 'Бизнес', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63e3b258979cbcb21429445d"),
    Product_Name: 'Раковина Сып-Свежая',
    Price: 66700,
    Tags: [ 'Бизнес', 'Күйі: Жаңа' ]
  },
  {
    _id: ObjectId("63e3b258979cbcb21429445e"),
    Product_Name: 'Душ Душевний',
    Price: 66700,
    Tags: [ 'Жеке адам', 'Күйі: Жаңа' ]
  }
]
endterm>
```

Update

Delete