

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("!" + str(response.status) + "!") #Indication of successful connection
    print("!" + str(response.status) + "!")
    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 successful 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:-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")
print("Product count: " + str(counter)) #Indicator of successful completion of scrabbing
print("ADIHAT") #Document counter
with open("olx_dataset.json", "w") as outfile:
    json.dump(dataset, outfile, ensure_ascii=False) #Creating a json file
#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 whether if it is possible to pay separately for the product).

```
main.py - endterm-parser - Visual Studio Code

EXPLORER
ENDTERM-PARSER
  .gitattributes
  chromedriver.exe
  gpt.py
  main.py
  olx_dataset.json
  pageparser.py
  testset.json
  testset2.json
  testset3.json
  testset4.json

main.py
48 "Price": pPrice,
49 "Tags": nTags,

TERMINAL
python pageparser.py
[{"Product Name": "Алюминевая дуга для душевой стекло 175x100cm", "Price": 2500, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Раковина", "Price": 15000, "Tags": ["Бизнес", "Күйі: Колданылган"]}, {"Product Name": "Раковина угловая для санузла", "Price": 10000, "Tags": ["Бизнес", "Күйі: Колданылган"]}, {"Product Name": "Раковина санита lux с стойкой", "Price": 6000, "Tags": ["Жеке адам", "Бөліп төлеу: Жок", "Күйі: Колданылган"]}, {"Product Name": "Качественно и недорого поклей обои", "Price": 11111, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Продам ваный аксессуар", "Price": 10000, "Tags": ["Жеке адам", "Бөліп төлеу: Жок", "Күйі: Жана"]}, {"Product Name": "Раковина на машинку автомат", "Price": 40000, "Tags": ["Жеке адам", "Күйі: Колданылган"]}, {"Product Name": "Поручень для ванной", "Price": 15000, "Tags": ["Жеке адам", "Бөліп төлеу: Жок", "Күйі: Жана"]}, {"Product Name": "Мебель в ванную комнату", "Price": 45000, "Tags": ["Жеке адам", "Күйі: Колданылган"]}, {"Product Name": "Душевые кабины на заказ в Астане", "Price": 23000, "Tags": ["Бизнес", "Күйі: Жана"]}, {"Product Name": "Изготовление изделий из акрилового камня", "Price": 70000, "Tags": ["Бизнес", "Күйі: Жана"]}, {"Product Name": "Тумба с раковиной и столешницей Марсала 120 см.Над стиральной машиной.", "Price": 130000, "Tags": ["Бизнес", "Күйі: Жана"]}, {"Product Name": "Двери под ванную", "Price": 20000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Продам шкаф - зеркало для ванной комнаты CORO20", "Price": 30000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Продам набор аксессуаров серый холщистый", "Price": 12000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Угловая полка в ванну", "Price": 3000, "Tags": ["Жеке адам", "Бөліп төлеу: Жок", "Күйі: Жана"]}, {"Product Name": "Продам раковину и стойку", "Price": 8000, "Tags": ["Жеке адам", "Күйі: Колданылган"]}, {"Product Name": "Срочно продам подставка для ванной дешево 5000", "Price": 5000, "Tags": ["Жеке адам", "Бөліп төлеу: Жок", "Күйі: Колданылган"]}, {"Product Name": "Корзина для белья silver", "Price": 5000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Игорка для ванной", "Price": 10000, "Tags": ["Жеке адам", "Күйі: Колданылган"]}, {"Product Name": "Второй для душа и ванны, производство Турции, веселые пингвины", "Price": 4000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Продам раковину с пьедесталом", "Price": 5000, "Tags": ["Жеке адам", "Күйі: Колданылган"]}, {"Product Name": "Душевая кабинка", "Price": 100000, "Tags": ["Жеке адам", "Күйі: Жана"]}, {"Product Name": "Душевые кабины и перегородка в вашем стиле на заказ", "Price": 35000, "Tags": ["Бизнес", "Күйі: Жана"]}, {"Product Name": "Раковина", "Price": 25000, "Tags": ["Жеке адам", "Күйі: Жана"]}

BOLD
Product count: 1192
ADINAT
PS C:\Users\User\Desktop\GH\endterm-parser> ]
```

```
Пармен жолағы
Microsoft Windows [Version 10.0.22621.1105]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>mongoimport --db endterm --collection olx --file C:\Users\User\Desktop\GH\endterm-parser\olx_dataset.json
connected to: mongodb://localhost/
Failed: cannot decode array into a primitive.D
2023-02-05T21:48:09.601+0600
2023-02-05T21:48:09.606+0600
0 document(s) imported successfully. 0 document(s) failed to import.

C:\Users\User>mongoimport --db endterm --collection olx --file C:\Users\User\Desktop\GH\endterm-parser\olx_dataset.json --jsonArray
connected to: mongodb://localhost/
2023-02-05T21:48:18.662+0600
2023-02-05T21:48:18.743+0600
1192 document(s) imported successfully. 0 document(s) failed to import.

C:\Users\User>
```

I have successfully scrapped 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, let's 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 the 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 person who offers such an 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 splitting the bill. But it seems like some of them are businesses trying to look like people. And we can prove it also by repetition 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 70x35см',
    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!

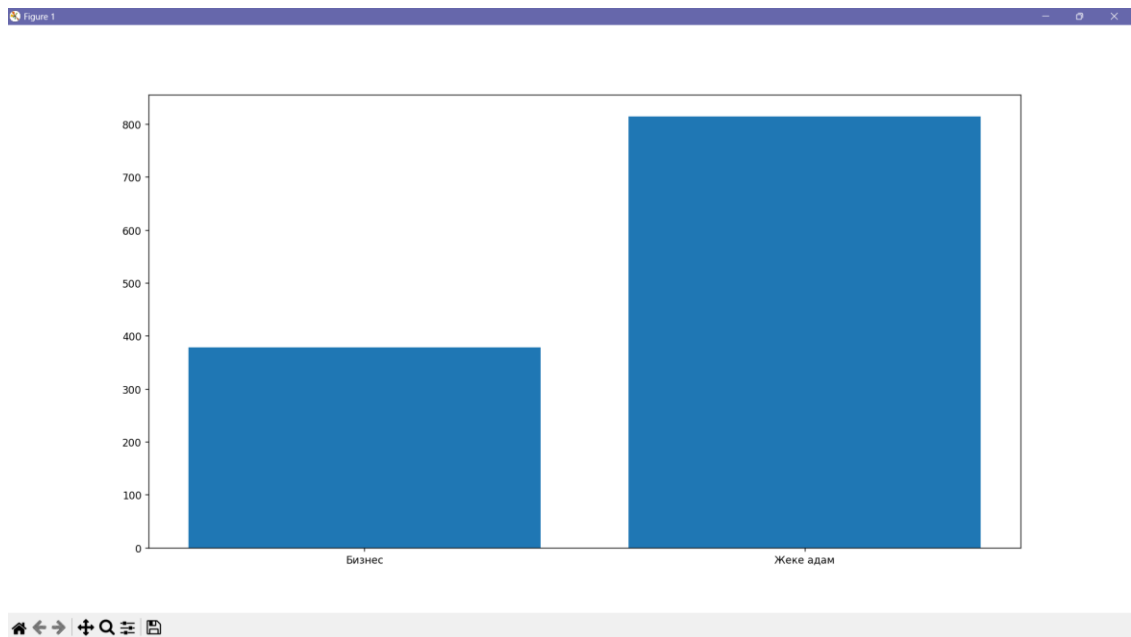


Fig.1 – The count of business offers and people's offers

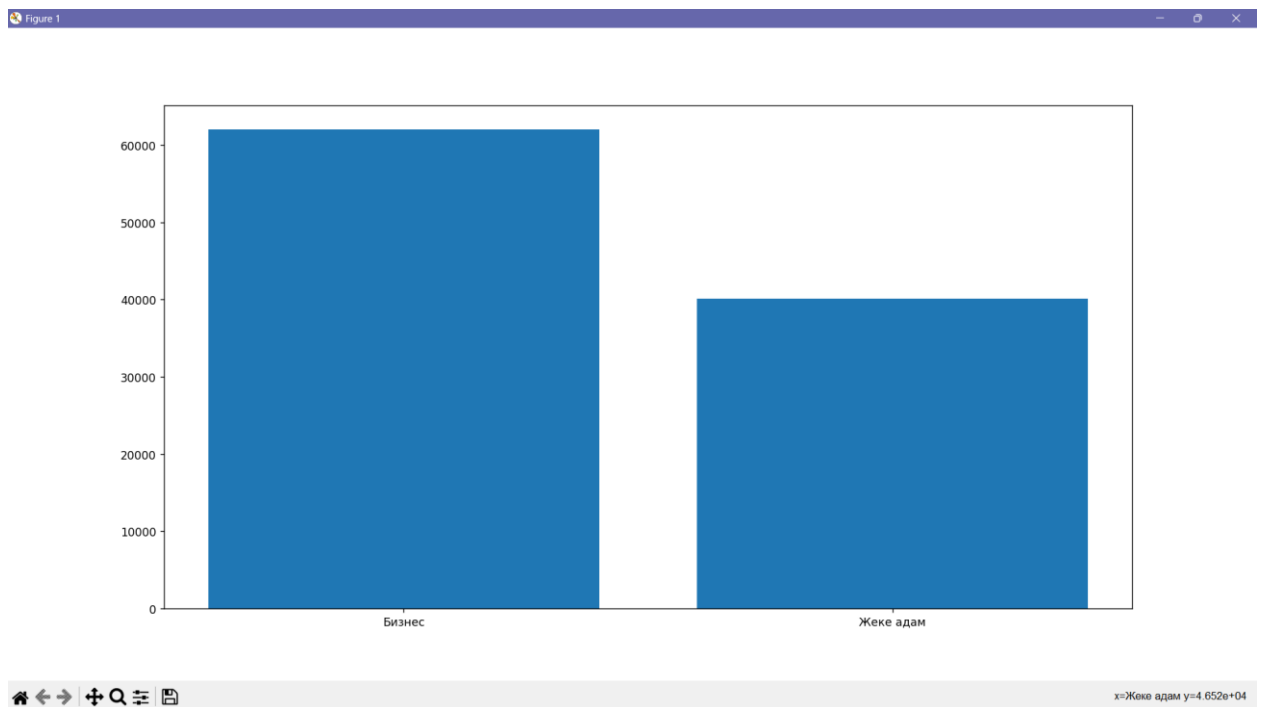


Fig. 2 – Average price for business category and the offers of regular people

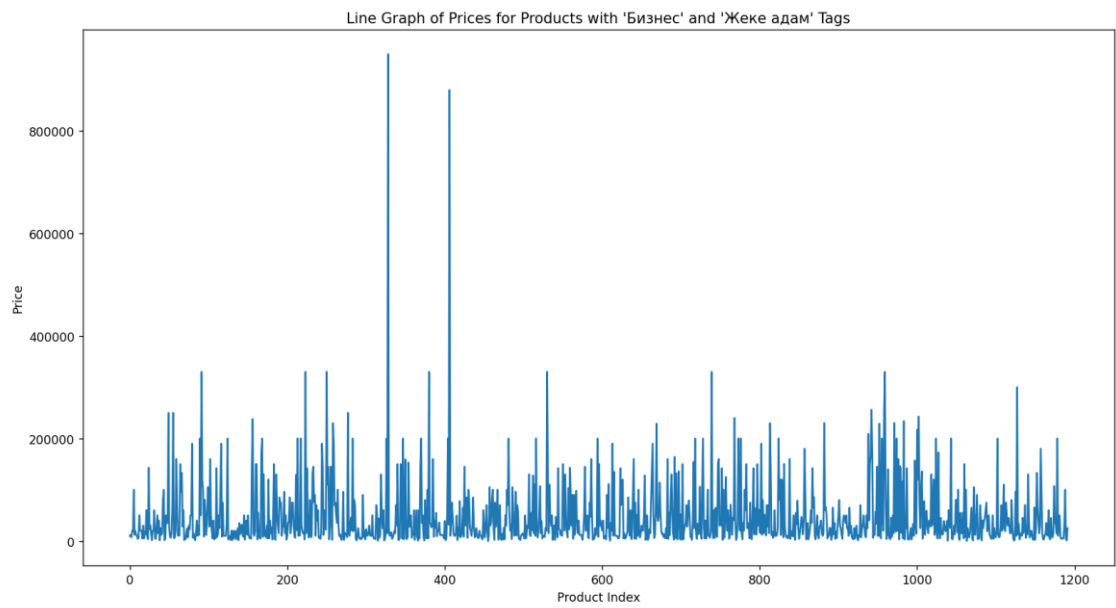


Fig. 3 The dynamic of prices of all products, we can see couple of anomalies

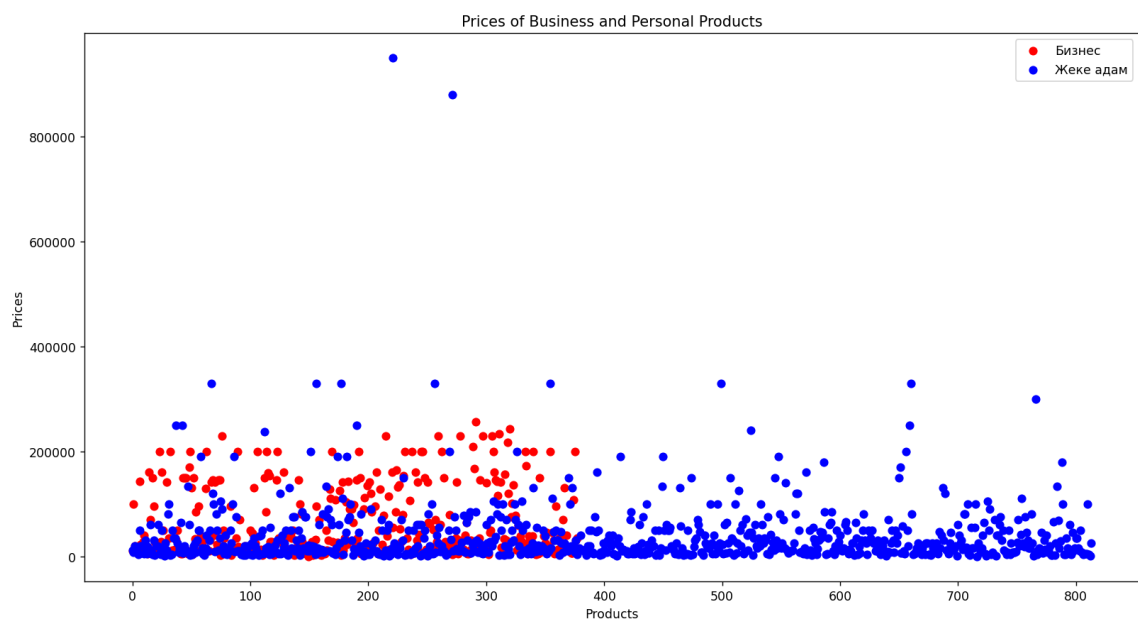


Fig. 4 From this diagram we can see how far can prices go for business offers and personal offers

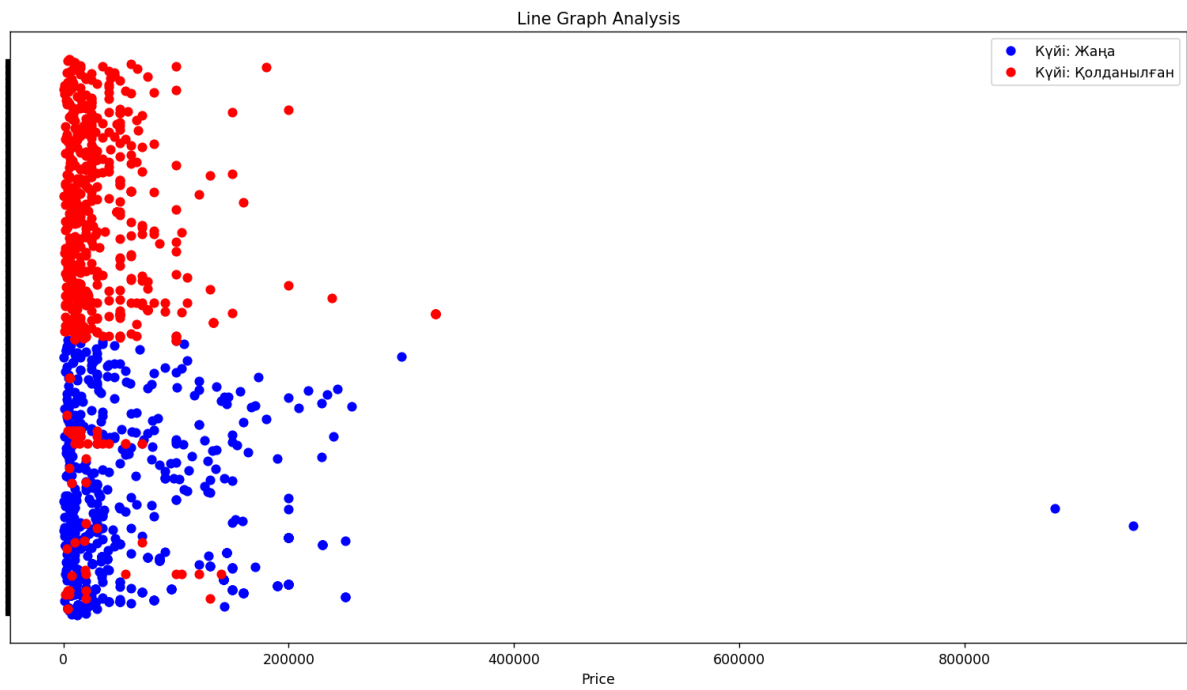


Fig. 5 Prices for New items and Used ones.