https://www.netidee.at/preiswert

Preis.wert Technical Guide

Dieses Technical Guide beschreibt die **Installation und Benutzung des preis.wert** Tools, welches zur Erhebung einer Datenbasis mit derer dynamische und personalisierte Preisgestaltung auf Webshops potentiell feststellbar sind, entwickelt wurde. Die Installationsanleitung beschreibt sämtliche Schritte und Konfigurationsmöglichkeiten des Systems im Detail um das Tool selbst einsetzen und weiterentwickeln zu können. Die notwendigen Schritte zum Einsatz der entwickelten Methodik zur Auswertung eines erhobenen Datensatzes sind in diesem Dokument enthalten. Für ein detaillierte Auswertung und Analyse, am Beispiel des im Projekt erhobenen und anonymisierten Datensatzes, verweisen wir auf die ausführliche Dokumentation im **preis.wert User Guide.**

Inhalt

https://www.netidee.at/preiswert

Installationsanleitung

Konfiguration Scraping

Das preis.wert Tool wurde für den Einsatz in einer Linux Umgebung entwickelt und unter Ubuntu 18.04.3 LTS getestet. Alle im Einsatz befindlichen Komponenten stehen auch für einen Betrieb unter Windows zur Verfügung und eine Adaption des Tools sollte mit einem überschaubaren Aufwand für den Betrieb ab Windows 7 möglich sein.

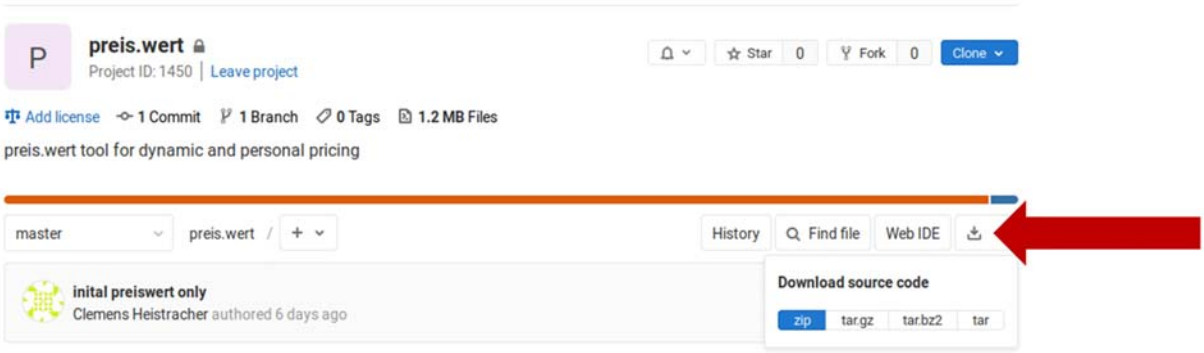
Es sind bereits Produkte und Webshops hinterlegt um das Tool sofort testen zu können. Einzig der Einsatz des Tools mit verschiedene VPNs erfordert eine anwenderspezifische Anpassung der Zugangsdaten zu den VPNs.

Installation

Clone git repository:

```
git clone https://git-service.ait.ac.at/im-public/preis.wert
```

Oder Download als zip und extrahieren:



Voraussetzung: Python >= 3.3

```
python3 --version
```

Pip installieren.

```
sudo apt update
```

```
sudo apt install python3-pip python3-dev
```

Pip Installation überprüfen.

```
pip3 --version
```

Benötigte Bibliotheken installieren.

```
pip3 install --user -r requirements.txt
```

Abmelden und Anmelden.

Konfiguration der zu untersuchenden Webshops

Die zu untersuchenden Webshops werden in der Datei “scapy_spider/configuration/define_shops.py" definiert. In “webshops_to_parse” werden alle beobachteten Produkte aller Shops, zu denen Spider-Konfigurationen vorhanden sind, festgelegt. Formal handelt es sich um ein Python Dictionary von der Struktur:

```
{website_name1: {product_name1:url1, product_name2:url2, ...},
 website_name2: {...},
  ....
}
```

- "**website_name***”: handelt es sich um die Identifier für die jeweiligen Webshops. Alle Produkte eines Webshops sollen demselben “website_name” zugeordnet sein, da dadurch auch die Extraktion des Preises definiert wird.
- “**product_name***”: bezeichnet den Namen eines Produktes, so, wie er auch in der Auswertung dargestellt wird. Des Weiteren sollen bei gleichen Produkten in verschieden Webshops dieselben Produktnamen gewählt werde, diese dann in einer einzelnen Abbildung dargestellt werden können.
- “**url***”: ist der Link zum Aufruf der einzelnen Produkte am jeweiligen Shop-System. (z.B. https://www.e-tec.at/details.php?artnr=267409)

Konfiguration der Preisextraktion

Preise werden Web Shop- oder produkt-spezifisch dargestellt. Daher muss die Preisextraktion der zu erhebenden Daten auch individuell für jeden Shop-Anbieter angepasst werden - dies erfolgt in Form von xpath Ausdrücken, die im Anschluss an einen Crawl angewandt werden um die Daten (z.B.: den Preis) aus dem archivierten HTML Code zu extrahieren und zu speichern. All dies erfolgt in “scapy_spider/configuration/define_xpath.py" durch die Funktion extract_price(scrapy.http.Response). “extract_price”: wendet hierbei die webshop-spezifische “xpath expression” an um den jeweiligen Preis zu extrahieren. “response.meta['site_name']” wird verwendet um, nach den im vorherigen Abschnitt definierten Shop Namen, zu selektieren. Im Anschluss wird mittels einer xpath Expression der preis aus dem html extrahiert.

```
elif response.meta['site_name'] == 'mediamarkt':
    price = response.xpath('//meta[@property="product:price:amount"]/@content').get()
```

Für eine Einführung zu xpath verweisen wir auf https://de.wikipedia.org/wiki/XPath.

Die xpath Expression zu einer html Seite kann getestet werden mit:

```
from scrapy.selector import Selector
body = '<html><body><span>good</span></body></html>'
Selector(text=body).xpath('//span/text()').get()
```

Des Weiteren kann das Modul try_xpath.py dazu verwendet werden um die Extraktion einzelner Preise beispielsweise bei der Aufnahme neuer Shop-Anbieter zu testen. Z.B:

```
from debug_price_extracction import try_price_extraction_single_url
default_url = 'https://www.universal.at/p/smartphone-samsung-galaxy-
```

```
s9/AKL10082545778#sku=2491545773-0-10082545778&ref=mba&nav-i=3&nav-
q=samsung%20galaxy%20s9%20duos&nav-p=T0'
default_site = {}
default_site["site_name"] = "universal"
default_site["product"] = "galaxy_s9"
try_price_extraction_single_url.main(url=default_url, site=default_site)


***************
Price collected 519,00
***************
html saved at try_xpath_html
```

Konfiguration der User Agents

Bei dem User Agent handelt es sich um eine Kennung, mit der sich ein Webbrowser gegenüber einer Webseite identifiziert. In den Standard Einstellungen sind jeweils der häufigste User Agent aus "Windows Notebook", "Apple Notebook", "IPhone" und "Android Phone" hinterlegt. Die User Agents können in "scapy_spider/configuration/define_shops.py" angepasst und beispielsweise um spezifischere User Agents (z.B. Samsung Android Geräte, etc.) erweitert werden.

```
useragent = {
    'windows': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.100 Safari/537.36',
    'macos': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.100 Safari/537.36',
    'ios': 'Mozilla/5.0 (iPhone; CPU iPhone OS 12_2 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.1 Mobile/15E148 Safari/604.1',
    'android': 'Mozilla/5.0 (Linux; U; Android 2.2) AppleWebKit/533.1 (KHTML, like Gecko) Version/4.0 Mobile Safari/533.1'
}
```

Test Scraping

```
python3 run_scrapy.py
```

Die Scraping Komponente kann durch den Aufruf "python3 run_scrapy.py" auf der Konsole getestet werden. Mit diesem Aufruf werden alle Produkte, mit allen User Agents, sowohl ohne gespeicherte Cookies als auch mit gespeicherten Cookies, abgefragt. Die Scraping Komponente selbst kann auch ohne der Verwendung des VPN Features betrieben werden.

For debugging in run_scrapy.py

```
settings["LOG_ENABLED"] = True
```

VPN Configuration

Um Abfragen aus verschiedenen Geolocations zu testen, können Virtual Private Networks (VPN) angebunden werden. Welche Standorte hierbei zur Verfügung stehen ist von VPN Anbieter zu Anbieter verschieden. Die einzige Voraussetzung ist, dass diese eine Open VPN Schnittstelle anbieten müssen, um mit preis.wert kompatibel zu sein. In unseren Tests wurde mit dem Anbieter Ghostpath VPN gearbeitet, da dieser verschiedene Standorte innerhalb Österreichs anbietet.

Installation openvpn am Hostbetriebssystem:

```
sudo apt-get install openvpn
```

Open vpn calls werden in "scapy_spider/configuration/define_shops.py" konfiguriert. Dabei wird der Ausdruck, der für eine openvpn Verbindung im Terminal verwendet wird, hinterlegt.

https://www.netidee.at/preiswert

REMARK: User Credentials für openvpn werden im Release nicht mitgeliefert.

```
# append the open vpn call (from the preis.wert root directory) to this list
openvpn_calls = []
openvpn_calls.append("sudo openvpn --config open_vpn/FRA/fra1.gpvpn.com.ovpn --auth-user-pass  open_vpn/user.txt")
openvpn_calls.append("sudo openvpn --config open_vpn/Graz/grz1.gpvpn.com.ovpn --auth-user-pass  open_vpn/user.txt")
```

Um das Scraping inklusive VPN handling zu starten:

```
sudo python3 start_preiswert.py
```

Root Privilegien sind notwendig, um mit openvpn eine Verbindung zu einem VPN aufbauen zu können. Falls SIe das Skript nicht als root ausführen möchten besteht die Möglichkeit openvpn als NOPASSWD in /etc/sudoers zu hinterlegen.

Scheduling

Um Preismessungen in regelmäßigen Intervallen durchzuführen, wird auf Scheduling mittels crontab zurückgegriffen:

```
sudo crontab -e
```

Auswahl des bevorzugten Editors:

Durch einen Zahlencode wird die Häufigkeit eines Aufrufes festgelegt.

```
# m h dom mon dow   command
SHELL=/bin/sh
*/50 * * * * cd preis.wert/preiswert_spider && python3 start_preiswert.py >> preiswert.log
```

Der obige Ausdruck bedeutet Scraping alle 50 Minuten, jede Stunde, jeden Tag des Monats, jeden Monat, jeden Wochentag.

Weiter Beispiele unter (https://de.wikipedia.org/wiki/Cron#Beispiele)

Datenstruktur

Bei erfolgreicher Abfrage werden die Ergebnisse im Ordner „results“ abgespeichert. Alle für die Auswertung relevanten Daten werden in „results.csv“ gespeichert. Zusätzlich werden auch alle gescrapten html Seiten in im Unterordner „html“ gespeichert. Dadurch lässt sich die Extraktion der Preise reproduzieren und es ist möglich rückwirkend auf Änderungen im Aufbau der Webseiten zu reagieren.

- **„results.csv“** In dieser Datei werden alle relevanten Daten zu den Preisabfragen gespeichert. Die Daten werden als „comma-separated value“ gespeichert und sind daher sowohl automatisiert als auch mit Tabellenkalkulationsprogrammen leicht auswertbar. Die folgenden Spalten wurden verwendet.

Spaltenname	**Bedeutung**
product	Produktname wie in “scapy_spider/configuration/define_shops.py" definiert
site_name	Websitename wie in “scapy_spider/configuration/define_shops.py" definiert

url	Tatsächlich verwendete url
price	Der extrahierte Preis
screenshotname	disabled
html_name	Pfad des htmls zu dieser Zeile
Cookies	Die empfangenen Cookies
cookies_sent	Die empfangenen Cookies
User agent	Der verwendete User Agent
configuration	The full scrapy Konfiguration
timestamp_start_crawl	Timestamp of start of this crawl
ip	The public ip used for this line

- **"cookie_storage.csv"** dient dazu auch wiederkehren User simulieren zu können. Dazu werden die empfangenen Cookies des jeweils letzten Aufrufs gespeichert.
- **„html"** enthält die html Seiten aller Anfragen. Bei geringem verfügbarem Speicherplatz kann es praktisch sein diesen Folder regelmäßig zu archivieren oder zu löschen. Eventuell auch als crontab Job.
- **„evaluation"** wird durch preiswert_evaluation.ipynb erstellt und enthält Abbildungen und Tabellen der Auswertung (siehe folgender Abschnitt).

Auswertung und Evaluierung

Um eine **graphische Übersicht der gemessenen Preispunkte** zu erhalten wurde das **jupyter notebook eval-personal_pricing.ipynb** entwickelt. Dieses beschreibt sämtliche Verarbeitungsschritte in chronologischer Form und erlaubt es die jeweiligen Code-Segmente auch einzeln auszuführen.

Jupyter wurde bereits mit requirements.txt installiert. Um jupyter zu starten

```
jupyter notebook evaluation/preiswert_evaluation.ipynb
```

In der zweiten Zelle kann der Pfad des Ergebnisfiles angepasst werden. Falls Untersuchungszeiträume voneinander abgegrenzt werden sollen empfiehlt es sich den Namen von „results.csv" manuell zu ändern. Für weite Abfragen wird dann ein neues „results.csv" erstellt.

Evaluation preis.wert

This notebook is used for post-processing and visualisation of data scraped for preis.wert.

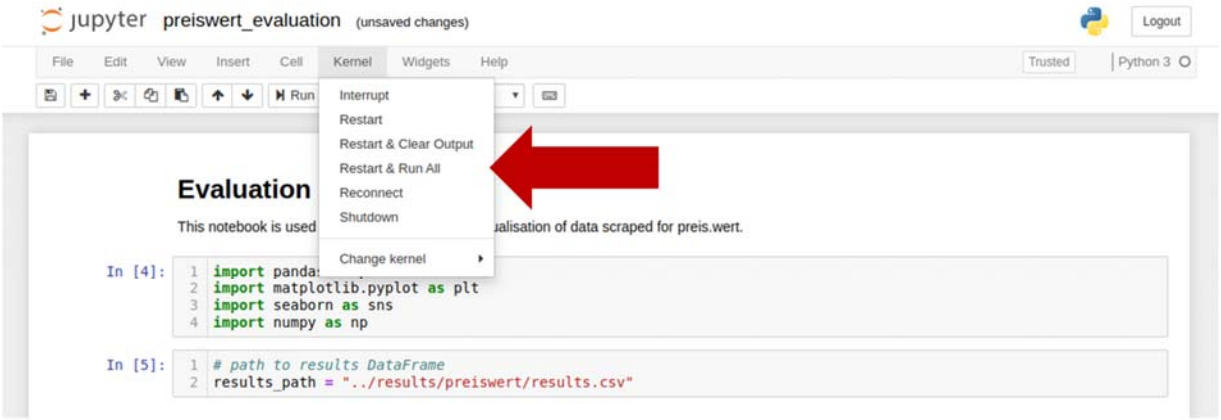
In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In [2]:
```
# path to results DataFrame
results_path = "../results/preiswert/results.csv"
```

In einem Jupyter Notebook werden einzelne Zellen durch „STRG" + „ENTER" ausgeführt. Dazu muss eine einzelne Zelle durch einen Mausklick ausgewählt werden (blaue Markierung).

Um zu überprüfen ob der Pfad korrekt ist können die ersten drei Zellen ausgeführt werden. Sofern nun die aktuellsten Preise aller Produkte angezeigt werden wurde das Ergebnisfile korrekt geladen.

Um alle Zellen des Jupyter Notebooks auszuführen klicken Sie auf „Kernel“ → „Restart & Run All“.



Die Ergebnisse der ein einzelnen Zellen werden direkt unter der jeweiligen Zelle dargestellt. Dadurch eignet sich ein Jupyter Notebook zur interaktiven Analyse. Zusätzlich werden Abbildungen und Tabellen in dem in Zelle 2 definierten Pfad ("results/preiswert/evaluation") abgespeichert.

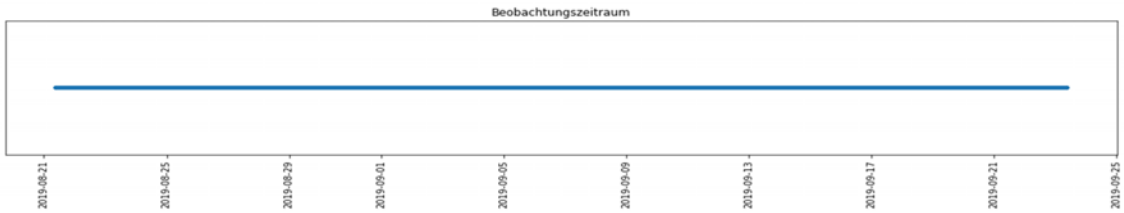
Das Inhaltsverzeichnis kann zur Navigation durch das Notebook verwendet werde

Exemplarische Analyse eines Datensatzes

In diesem Abschnitt wir die Analyse eines exemplarischen Datensatzes (results_190923_anon.csv) vorgestellt.

Beobachtungszeitraum

Der Fokus des ersten Abschnittes des Analyse Notebooks besteht darin einen Überblick über die gewonnenen Daten zu gewinnen. Dazu wird der zunächst der Beobachtungszeitraum dargestellt.



Übersicht aktuellste Preise

Es werden die jeweils aktuellsten Preise zu einem Shop dargestellt.

Show newest Price

In [6]:
```
print("number of prices are %s."%len(df))
df.groupby(['product','site_name']).last().reset_index()[['site_name','product','price']].sort_values('site_nam
```

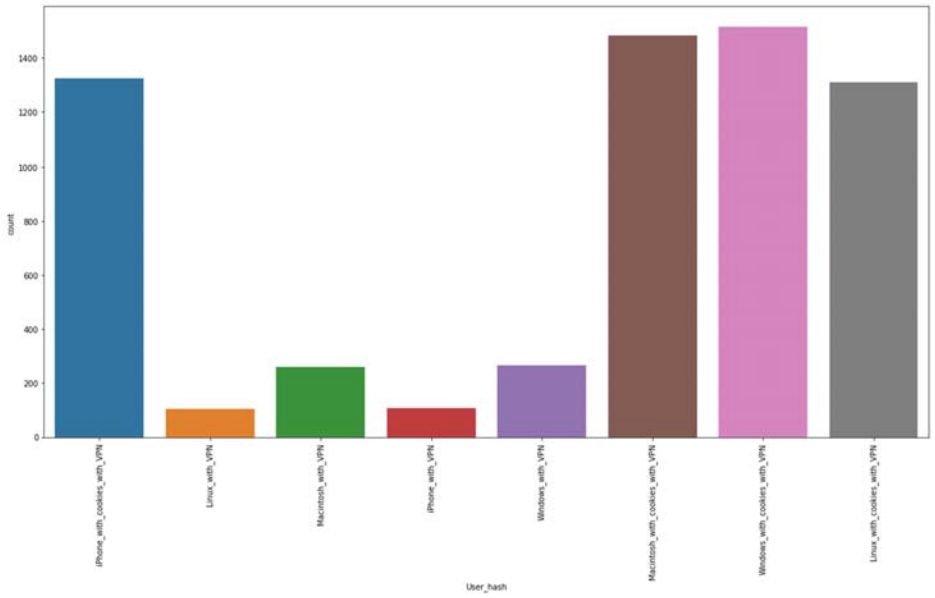
```
number of prices are 223377.
```

Out[6]:

	site_name	product	price
0	Shop_A	Jack Wolfskin Funktionsjacke »365 INFLUENCER J...	139.99
24	Shop_A	switch	369.99
21	Shop_A	samsung_fernseher	1269.99
19	Shop_A	matebook	1699.99
16	Shop_A	galaxy_s9	519.00
11	Shop_A	Seidensticker Hemd Damen (hellblau)	59.99
9	Shop_A	Reebok Classic, weiße Sneakers, Club C85	79.99
13	Shop_A	Superdry Kapuzensweatshirt »VINTAGE LOGO POP E...	69.95
5	Shop_A	Lee Damen Jeans Marion Straight Legs	89.95
2	Shop_A	Jeans Herren, G-Star Raw Slim Fit Denim	54.99
7	Shop_A	Nike Kapuzensweatshirt »Team Club«	37.89
17	Shop_B	galaxy_s9	638.40
22	Shop_B	samsung_fernseher	1402.50
25	Shop_B	switch	319.00
23	Shop_C	samsung_fernseher	1289.00
20	Shop_C	matebook	1699.00
18	Shop_C	galaxy_s9	649.00
26	Shop_C	switch	311.00
3	Shop_E	Jeans Herren, G-Star Raw Slim Fit Denim	119.95
14	Shop_E	Superdry Kapuzensweatshirt »VINTAGE LOGO POP E...	69.95
15	Shop_F	Superdry Kapuzensweatshirt »VINTAGE LOGO POP E...	69.90
12	Shop_F	Seidensticker Hemd Damen (hellblau)	49.90
10	Shop_F	Reebok Classic, weiße Sneakers, Club C85	99.90
6	Shop_F	Lee Damen Jeans Marion Straight Legs	69.90
1	Shop_F	Jack Wolfskin Funktionsjacke »365 INFLUENCER J...	145.00
4	Shop_F	Jeans Herren, G-Star Raw Slim Fit Denim	84.90
8	Shop_F	Nike Sweatshirt 'PO FLC CLUB'	49.90

Anzahl der Abfragen je Identität

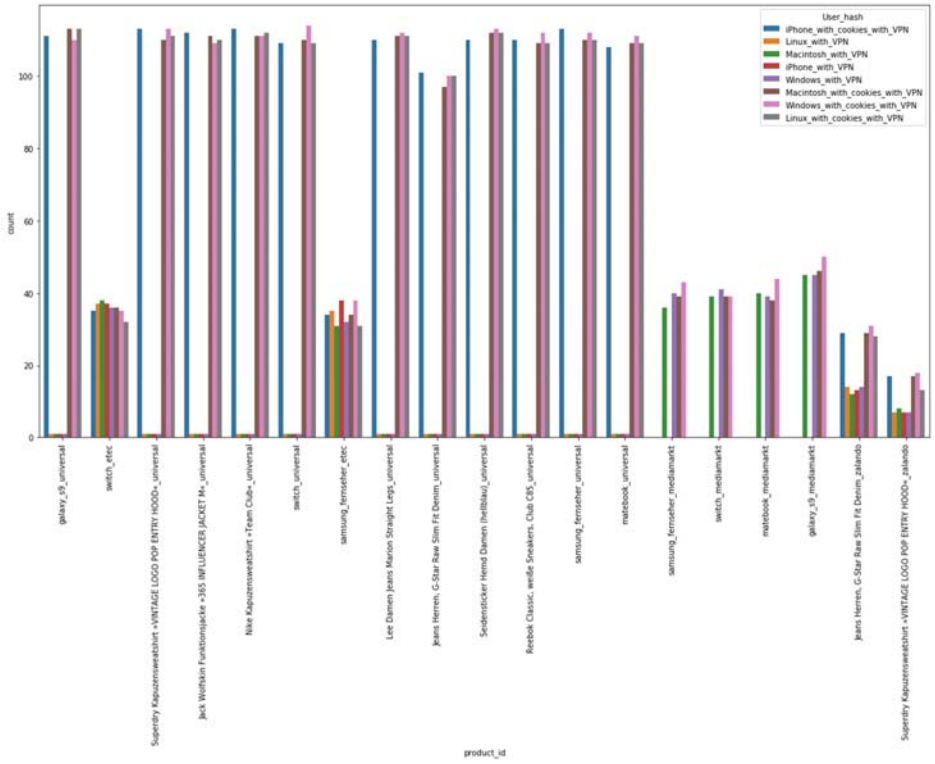
Es wird eine Übersicht zu allen verwendeten Identitäten erstellt.



https://www.netidee.at/preiswert

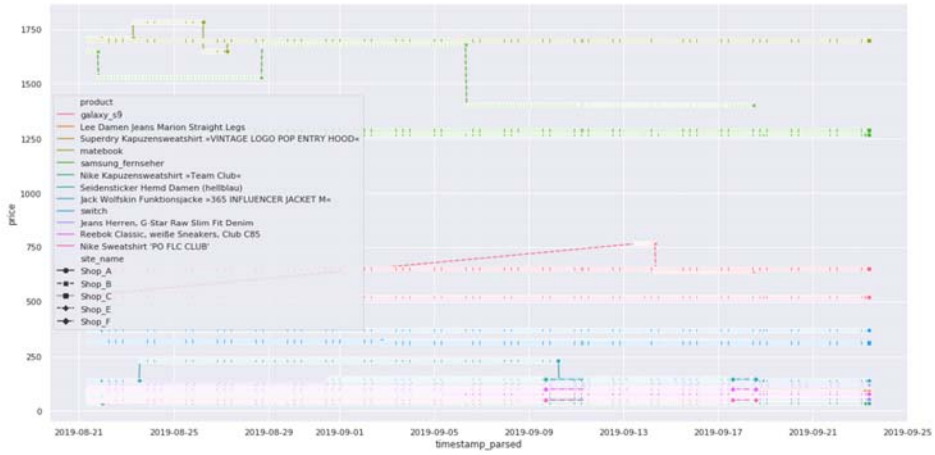
Anzahl der Abfragen und je Identität und Produkt

Die Anzahl der erfolgreichen Abfragen je Identität und Produkt werden in einem Balkendiagramm dargestellt.



Preisverläufe zu allen Produkten

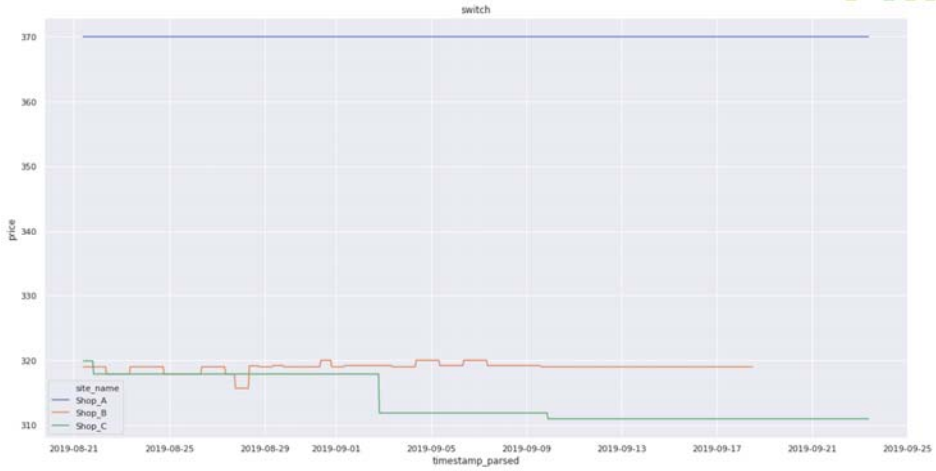
In dieser Abbildung werden alle erfassten Preise zusammengefasst dargestellt.



Preisverläufe zu einem Produkt

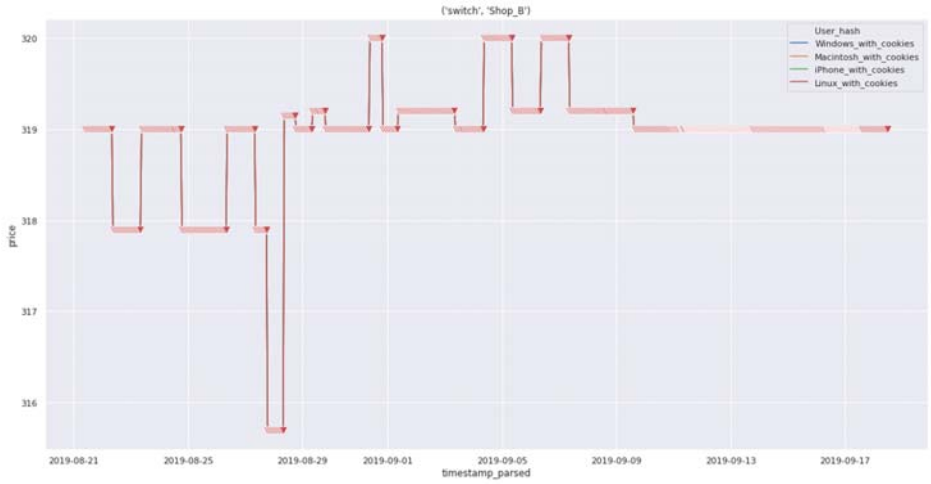
Es können auch die Preisverläufe verschiedener Shops mit dem selben Produkt dargestellt werden.

https://www.netidee.at/preiswert



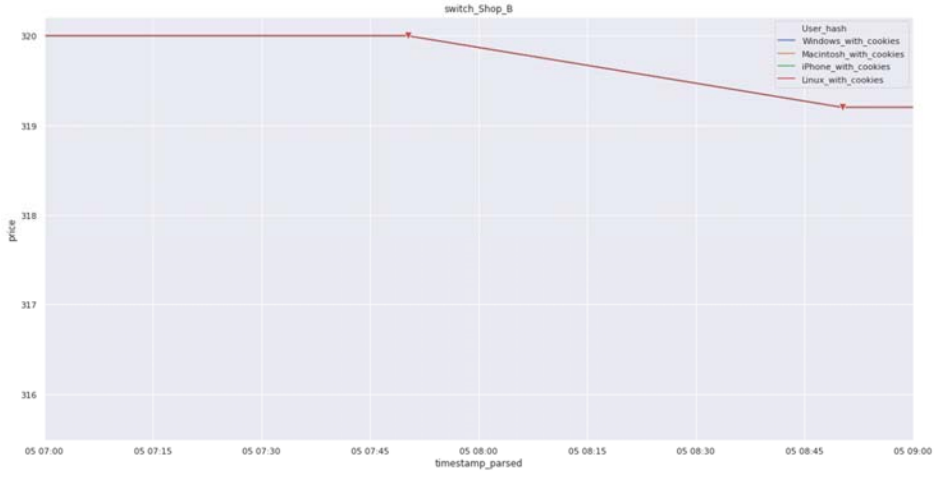
Preisverlauf eines Produktes mit verschiedenen Identitäten

Um personalisierte Preisgestaltung zu detektieren wird der Preis für alle Identitäten dargestellt.



Zoom zu Preissprung

Ein vergrößerter Ausschnitt der obigen Abbildung.



Eine Analyse und Auswertung solch eines Datensatzes finden Sie im **preis.wert User Guide**

Lizenz

Das entwickelte preis.wert Tool wurde unter der GPL-V3.0 veröffentlicht. Software, die unter der GNU General Public License lizenziert ist, ist freie Software, und alle Software, die auf GPL-lizenzierten Komponenten aufbaut, ist ebenso frei und muss ebenso unter der GPL lizenziert werden. Für Details siehe: https://git-service.ait.ac.at/im-public/preis.wert/blob/master/LICENSE

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.
[...]
```