Assignment1

June 27, 2019

1 Assignment 1

1.0.1 Author: Rajat Shrivastava

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

In [12]:
```
#Data of Stations
df = pd.read_csv("example_sprit_cut_station.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	ID	VERSION	VERSION_TIM
0	d37bee76-91b6-485a-b974-24f1b0d391fa	4	2015-01-09 10:26:15.00000
1	51d4b652-a095-1aa0-e100-80009459e03a	796	2015-09-28 21:00:13.00000
2	c7caf231-0e47-40db-92de-9349a2eb2bef	4	2014-09-26 13:24:57.00000
3	51d4b53b-a095-1aa0-e100-80009459e03a	796	2015-09-28 21:00:13.00000
4	51d4b70f-a095-1aa0-e100-80009459e03a	796	2015-09-28 21:00:13.00000

	NAME	BRAND	STREET	\
0	Aral Tankstelle	ARAL	An der Tagweide	
1	JET MANNHEIM UNTERMUEHLAUSTRASSE 83	JET	UNTERMUEHLAUSTRASSE 83	
2	Aral Tankstelle	ARAL	Trippstadter Straße	
3	JET KAISERSLAUTERN PARISER STR. 167	JET	PARISER STR. 167	
4	JET GERMERSHEIM MUENCHENER STRASSE 12	JET	MUENCHENER STRASSE 12	

	HOUSE_NUMBER	POST_CODE	PLACE	PUBLIC_HOLIDAY_IDENTIFIER	I
0	2	76139	Karlsruhe	\N	49.0202
1	\N	68169	MANNHEIM	\N	49.5089
2	69	67663	Kaiserslautern	\N	49.4337
3	\N	67655	KAISERSLAUTERN	\N	49.4422
4	\N	76726	GERMERSHEIM	\N	49.2268

	LNG
0	8.459429
1	8.467691

```
2  7.757465
3  7.748360
4  8.374400
```

```
In [4]: df.shape
```

```
Out[4]: (824, 12)
```

```
In [5]: #check date data type
        type(df.VERSION_TIME[0])
```

```
Out[5]: str
```

1.0.2 1.How many different stations exist in the data set and what is the existing history in days (bar chart)?

There exist 55 - 2 = 53 different Stations(21 and 40th Row Data is not useful and will be removed from further analysis)

```
In [6]: df.describe()
```

```
Out[6]:           VERSION         LAT         LNG
        count  824.000000  824.000000  824.000000
        mean   131.427184   49.322226    8.191227
        std    269.682766    0.254722    0.283228
        min      1.000000   48.800790    7.505350
        25%      1.000000   49.106052    8.106655
        50%      4.000000   49.374500    8.289707
        75%     22.250000   49.509410    8.404923
        max    881.000000   49.799397    8.499941
```

```
In [7]: station_names = df['BRAND'].value_counts().to_frame()
```

```
In [8]: #One of the best things I learnt today: Converting the counted values into
        station_names = df['BRAND'].value_counts().rename_axis('BRAND').reset_index
```

```
In [9]: station_names
```

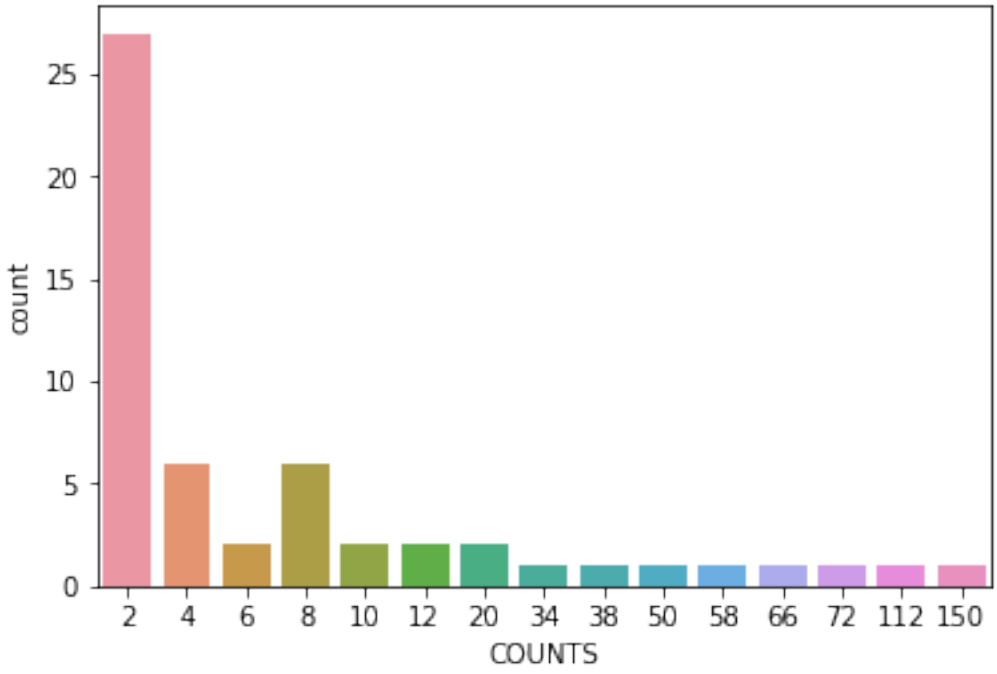
```
Out[9]:
                                                BRAND  COUNTS
        0                                        ARAL     150
        1                                       Shell     112
        2                                        ESSO      72
        3                                        AVIA      66
        4                                       Total      58
        5                                         JET      50
        6                                         BFT      38
        7                                        Agip      34
        8                                         OMV      20
        9          Supermarkt-Tankstelle am real- Markt      20
        10                              Frühmesser GmbH      12
```

11		OIL!	12
12		HEM	10
13		bft	10
14		Freie	8
15		T	8
16		SB	8
17		Supermarkt-Tankstelle	8
18		Schuster & Sohn KG	8
19		ED	8
20		Freie Tankstelle	6
21		\N	6
22		Schiffer & Nicklaus GmbH	4
23		Winkler	4
24		ZG Raiffeisen Energie	4
25		Preis	4
26		freie Tankstelle	4
27		Tankcenter	4
28		TOP	2
29		CLASSIC	2
30		Markant	2
31		Markenfreie TS	2
32		Eberhardt Jöhlingen	2
33		mtb	2
34		E Center	2
35		Sefrin Heizöl & Kraftstoffe	2
36		frei	2
37		rnt	2
38		Tankhof Iffezheim	2
39		SB Markt	2
40	AUTO ZOTZ Herxheim b.LD.	Tanken an L.493	2
41		Raiffeisen Tankstelle	2
42		TOTAL	2
43		nicht mehr aktiv	2
44		Bft	2
45		Raiffeisen	2
46		ELAN	2
47		Fritz Walter GmbH	2
48		TAMOIL	2
49		SB-Markttankstelle	2
50	AUTO ZOTZ Landau in der Pfalz	Tanken und was...	2
51		Mr. Wash Autoservice AG	2
52		Supermarkt Tankstelle	2
53		TS ARNT	2
54		Tankstelle Heinz	2

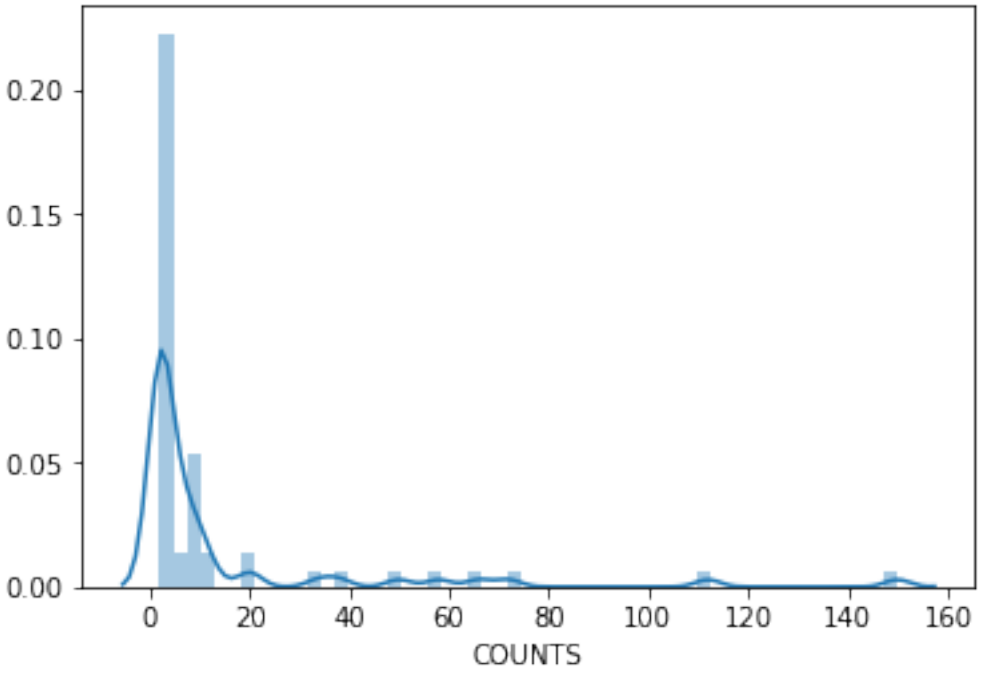
```
In [13]: type(station_names.COUNTS[0])
```

```
Out[13]: numpy.int64
```

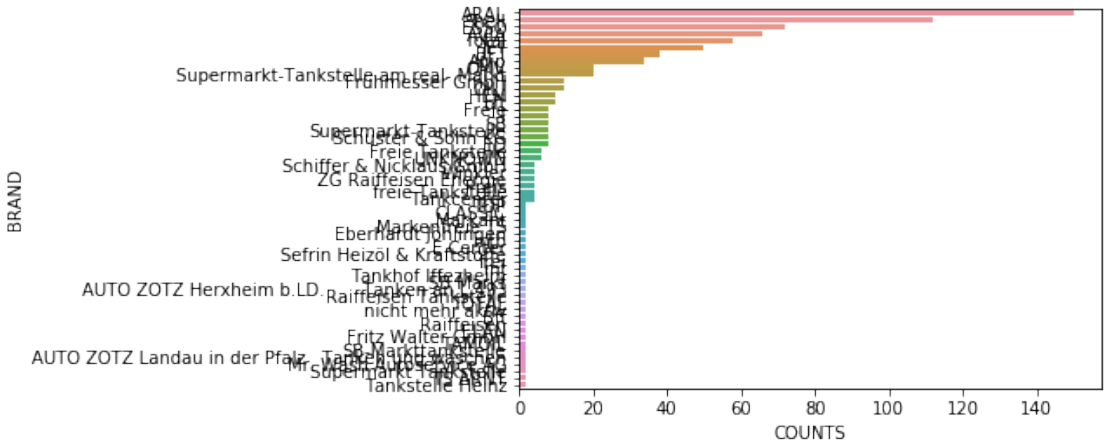
```
In [14]: type(station_names['COUNTS'][0])

Out[14]: numpy.int64

In [15]: sns.countplot(x='COUNTS',data=station_names)
         plt.show()
```




```
In [16]: sns.distplot(station_names['COUNTS'],bins = 55)
         plt.show()
```



In [17]:
```
sns.barplot(x='COUNTS', y='BRAND', data = station_names)
plt.show()
```



In [ ]:

In [18]:
```
#Data of Prices
df_2 = pd.read_csv('example_sprit_cut_prices.csv')
```

```
In [19]: #It shows the values before the dataset is cleaned
         df_2.describe()
```

Out[19]:

	E5	E10	DIESEL	CHANGED
count	1.033471e+06	1.033471e+06	1.033471e+06	1.033471e+06
mean	1.455434e+03	1.427606e+03	1.245517e+03	1.895458e+01
std	1.011221e+02	9.689516e+01	1.004966e+02	7.722413e+00
min	8.000000e+00	1.129000e+03	9.990000e+02	1.000000e+00
25%	1.379000e+03	1.359000e+03	1.169000e+03	2.100000e+01
50%	1.459000e+03	1.429000e+03	1.229000e+03	2.100000e+01
75%	1.529000e+03	1.499000e+03	1.319000e+03	2.100000e+01
max	1.759000e+03	1.729000e+03	1.129000e+04	6.300000e+01

Here, we see that : min 'E5', min 'E10' and max 'E5' & 'E10' don't seem right. So our data is still not cleaned properly. Lets clean the data and check the result again.

```
In [20]: df_2.describe()
```

Out[20]:

	E5	E10	DIESEL	CHANGED
count	1.033471e+06	1.033471e+06	1.033471e+06	1.033471e+06
mean	1.455434e+03	1.427606e+03	1.245517e+03	1.895458e+01
std	1.011221e+02	9.689516e+01	1.004966e+02	7.722413e+00
min	8.000000e+00	1.129000e+03	9.990000e+02	1.000000e+00
25%	1.379000e+03	1.359000e+03	1.169000e+03	2.100000e+01
50%	1.459000e+03	1.429000e+03	1.229000e+03	2.100000e+01
75%	1.529000e+03	1.499000e+03	1.319000e+03	2.100000e+01
max	1.759000e+03	1.729000e+03	1.129000e+04	6.300000e+01

```
In [19]: #data = data.set_index("Area")
         #data = data.drop("Ireland", axis=0)
         #data[data["Area"] == "Ireland"]

         #df.drop(df[df.score < 50].index, inplace=True)

         #df_2 = df_2.drop(df_2[df_2.E5 > 1700].index)
         #df_2 = df_2.set_index('E5')
         #df_2 = df_2.drop( > 1700, axis=0)

In [21]: #filter dataset for E5 values
         df_2 = df_2[df_2.E5 < 1700]
         df_2 = df_2[df_2.E5 > 1000]

In [22]: #filter dataset for E10 values
         df_2 = df_2[df_2.E10 < 1700]
         df_2 = df_2[df_2.E10 > 1000]

In [23]: #filter dataset for E10 values
         df_2 = df_2[df_2.DIESEL < 1600]
         df_2 = df_2[df_2.DIESEL > 800]
```

```
In [24]: len(df_2)

Out[24]: 1032518

In [25]: df_2.describe()

Out[25]:                 E5           E10        DIESEL       CHANGED
         count  1.032518e+06  1.032518e+06  1.032518e+06  1.032518e+06
         mean   1.455199e+03  1.427381e+03  1.245258e+03  1.895091e+01
         std    1.008456e+02  9.665279e+01  9.924767e+01  7.718604e+00
         min    1.159000e+03  1.129000e+03  9.990000e+02  1.000000e+00
         25%    1.379000e+03  1.359000e+03  1.169000e+03  2.100000e+01
         50%    1.459000e+03  1.429000e+03  1.229000e+03  2.100000e+01
         75%    1.529000e+03  1.499000e+03  1.319000e+03  2.100000e+01
         max    1.699000e+03  1.690000e+03  1.589000e+03  6.300000e+01
```

Now the data looks clean enough to work with.

1.0.3 2. What is the min, mean, max price for each gasoline type and station weekly (time series graph).

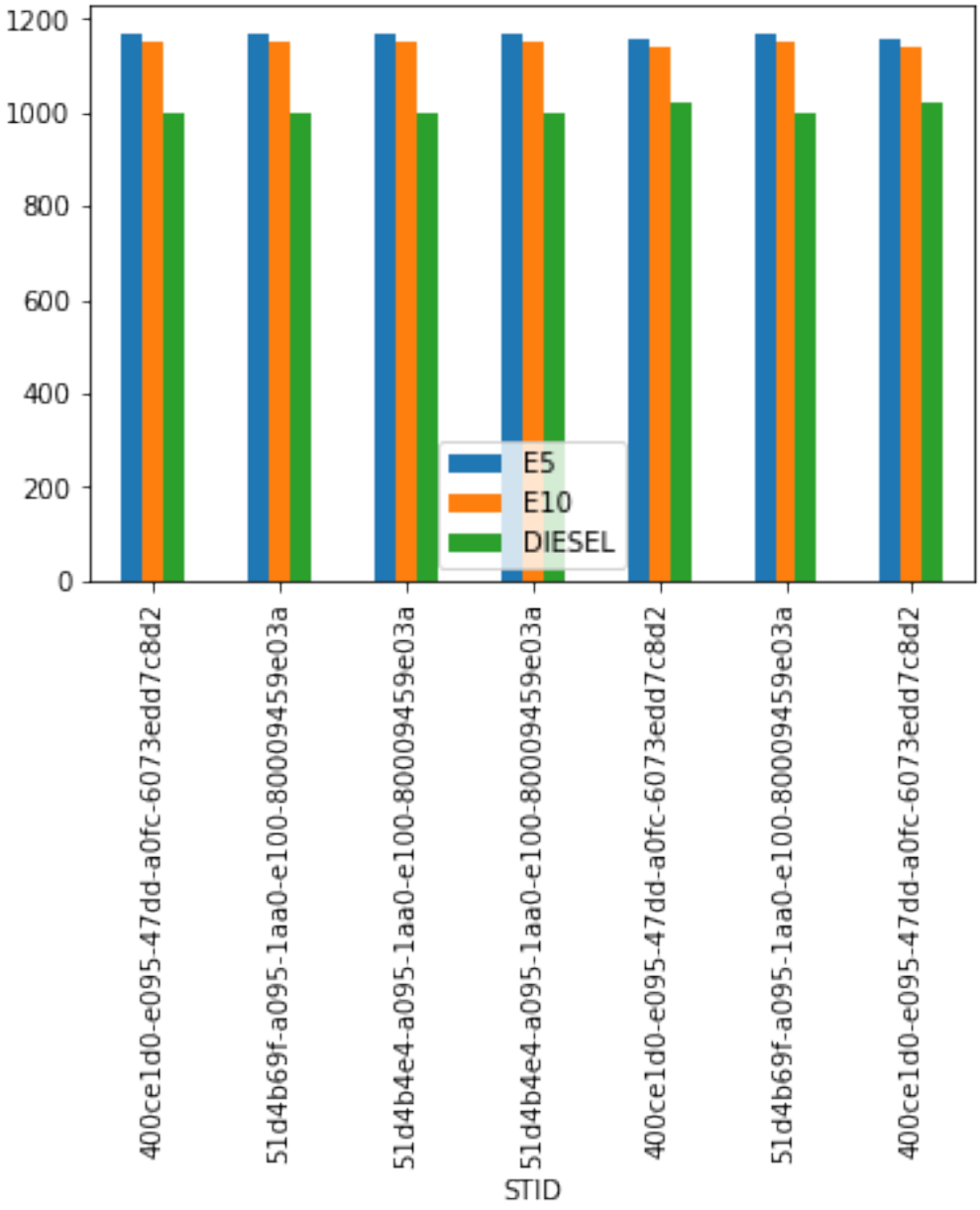
```
/     E5      E10        Diesel
Min   1159    1129       999
Mean  1455    1427       1245
Max   1699    1690       1589
```

1.0.4 3. What is cheapest station (in average) and why?

```
In [26]: cheap_stations = df_2[df_2['E5'] < 1190]
         cheap_stations = cheap_stations[cheap_stations['E10'] < 1155]
         cheap_stations = cheap_stations[cheap_stations['DIESEL'] < 1020]
         print(cheap_stations)

                                          STID    E5   E10  DIESEL  \
75415   400ce1d0-e095-47dd-a0fc-6073edd7c8d2  1169  1149     999
113510  51d4b69f-a095-1aa0-e100-80009459e03a  1169  1149     999
115827  51d4b4e4-a095-1aa0-e100-80009459e03a  1169  1149     999
175750  51d4b4e4-a095-1aa0-e100-80009459e03a  1169  1149     999
538319  400ce1d0-e095-47dd-a0fc-6073edd7c8d2  1159  1139    1019
596003  51d4b69f-a095-1aa0-e100-80009459e03a  1169  1149     999
846787  400ce1d0-e095-47dd-a0fc-6073edd7c8d2  1159  1139    1019

                      DATE_CHANGED  CHANGED
75415   2015-01-22 19:25:01.000000       21
113510  2015-01-22 19:09:01.000000       21
115827  2015-01-22 19:09:01.000000       21
175750  2015-01-22 19:09:01.000000       21
538319  2015-01-16 13:26:01.000000       21
596003  2015-01-22 19:09:01.000000       21
```

```
846787  2015-01-16 13:26:01.000000       21
```

```
In [27]: #lets plot this data
         cheap_stations.plot(x='STID', y=["E5", "E10", "DIESEL"], kind="bar")
         plt.show()
```



From the graph above, we can see that the Gas Station with ID:'400ce1d0-e095-47dd-a0fc-6073edd7c8d2' (Globus Handelshof GmbH & Co.KG Betriebsstätte) is the cheapest.

1.0.5 4. At which day of a week is the price most likely the cheapest (week profile)?

On Thursdays, we see that the prices for 'E5' and 'E10' is the cheapest compared to other days.
For 'DIESEL', its again cheaper on Thursdays but the price difference is not so much.

```
In [28]: #Convert date given to us in dataset according to our required format
         df_2['formatted_date'] = pd.to_datetime(df_2['DATE_CHANGED'])

In [29]: df_2.head()
```

Out[29]:

	STID	E5	E10	DIESEL	DATE_CHANGED	CHANGED	formatted_date
0	01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1369	2014-10-02 05:26:01.000000	21	2014-10-02 05:26:01
1	01bf5a06-4248-43a5-9524-56123fa3ab2f	1589	1549	1389	2014-10-02 05:22:01.000000	21	2014-10-02 05:22:01
2	01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1359	2014-10-02 09:06:01.000000	1	2014-10-02 09:06:01
3	01bf5a06-4248-43a5-9524-56123fa3ab2f	1539	1499	1339	2014-10-03 12:14:01.000000	21	2014-10-03 12:14:01
4	01bf5a06-4248-43a5-9524-56123fa3ab2f	1529	1489	1329	2014-10-03 14:14:01.000000	21	2014-10-03 14:14:01

```
In [30]: #Converting the date data to Days of the week
         df_2['DAY_OF_WEEK'] = df_2.formatted_date.apply(lambda x: x.dayofweek)

In [31]: df_2
```

Out[31]:

	STID	E5	E10	DIESEL
0	01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1369
1	01bf5a06-4248-43a5-9524-56123fa3ab2f	1589	1549	1389
2	01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1359
3	01bf5a06-4248-43a5-9524-56123fa3ab2f	1539	1499	1339
4	01bf5a06-4248-43a5-9524-56123fa3ab2f	1529	1489	1329
5	01bf5a06-4248-43a5-9524-56123fa3ab2f	1629	1589	1429
6	01bf5a06-4248-43a5-9524-56123fa3ab2f	1549	1509	1389
7	01bf5a06-4248-43a5-9524-56123fa3ab2f	1549	1509	1379
8	01bf5a06-4248-43a5-9524-56123fa3ab2f	1549	1509	1339
9	01bf5a06-4248-43a5-9524-56123fa3ab2f	1559	1519	1309
10	01bf5a06-4248-43a5-9524-56123fa3ab2f	1559	1519	1299
11	01bf5a06-4248-43a5-9524-56123fa3ab2f	1529	1489	1299
12	01bf5a06-4248-43a5-9524-56123fa3ab2f	1619	1579	1419
13	01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1369
14	01bf5a06-4248-43a5-9524-56123fa3ab2f	1619	1579	1419
15	01bf5a06-4248-43a5-9524-56123fa3ab2f	1509	1469	1309
16	01bf5a06-4248-43a5-9524-56123fa3ab2f	1519	1479	1309
17	01bf5a06-4248-43a5-9524-56123fa3ab2f	1509	1469	1309
18	01bf5a06-4248-43a5-9524-56123fa3ab2f	1499	1459	1309

19	01bf5a06-4248-43a5-9524-56123fa3ab2f	1489	1449	1299
20	01bf5a06-4248-43a5-9524-56123fa3ab2f	1479	1439	1289
21	01bf5a06-4248-43a5-9524-56123fa3ab2f	1559	1519	1359
22	01bf5a06-4248-43a5-9524-56123fa3ab2f	1509	1469	1329
23	01bf5a06-4248-43a5-9524-56123fa3ab2f	1499	1459	1319
24	01bf5a06-4248-43a5-9524-56123fa3ab2f	1469	1429	1289
25	01bf5a06-4248-43a5-9524-56123fa3ab2f	1459	1419	1279
26	01bf5a06-4248-43a5-9524-56123fa3ab2f	1449	1409	1279
27	01bf5a06-4248-43a5-9524-56123fa3ab2f	1439	1399	1269
28	01bf5a06-4248-43a5-9524-56123fa3ab2f	1429	1389	1259
29	01bf5a06-4248-43a5-9524-56123fa3ab2f	1579	1539	1399
...	...	...	...	...
1033441	b48682b4-d344-4bb9-8890-a8504912119d	1449	1429	1229
1033442	b48682b4-d344-4bb9-8890-a8504912119d	1479	1459	1259
1033443	b48682b4-d344-4bb9-8890-a8504912119d	1599	1579	1369
1033444	b48682b4-d344-4bb9-8890-a8504912119d	1589	1569	1359
1033445	b48682b4-d344-4bb9-8890-a8504912119d	1519	1499	1289
1033446	b48682b4-d344-4bb9-8890-a8504912119d	1479	1459	1239
1033447	b48682b4-d344-4bb9-8890-a8504912119d	1469	1449	1239
1033448	b48682b4-d344-4bb9-8890-a8504912119d	1449	1429	1229
1033449	b48682b4-d344-4bb9-8890-a8504912119d	1449	1429	1229
1033450	b48682b4-d344-4bb9-8890-a8504912119d	1469	1449	1249
1033451	b48682b4-d344-4bb9-8890-a8504912119d	1449	1429	1229
1033452	b48682b4-d344-4bb9-8890-a8504912119d	1469	1449	1249
1033453	b48682b4-d344-4bb9-8890-a8504912119d	1499	1479	1279
1033454	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1279
1033455	b48682b4-d344-4bb9-8890-a8504912119d	1599	1579	1389
1033456	b48682b4-d344-4bb9-8890-a8504912119d	1459	1439	1239
1033457	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1269
1033458	b48682b4-d344-4bb9-8890-a8504912119d	1479	1459	1259
1033459	b48682b4-d344-4bb9-8890-a8504912119d	1599	1579	1389
1033460	b48682b4-d344-4bb9-8890-a8504912119d	1599	1579	1369
1033461	b48682b4-d344-4bb9-8890-a8504912119d	1499	1479	1269
1033462	b48682b4-d344-4bb9-8890-a8504912119d	1589	1569	1379
1033463	b48682b4-d344-4bb9-8890-a8504912119d	1579	1559	1359
1033464	b48682b4-d344-4bb9-8890-a8504912119d	1579	1559	1359
1033465	b48682b4-d344-4bb9-8890-a8504912119d	1529	1509	1309
1033466	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1279
1033467	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1279
1033468	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1269
1033469	b48682b4-d344-4bb9-8890-a8504912119d	1489	1469	1269
1033470	b48682b4-d344-4bb9-8890-a8504912119d	1469	1449	1249

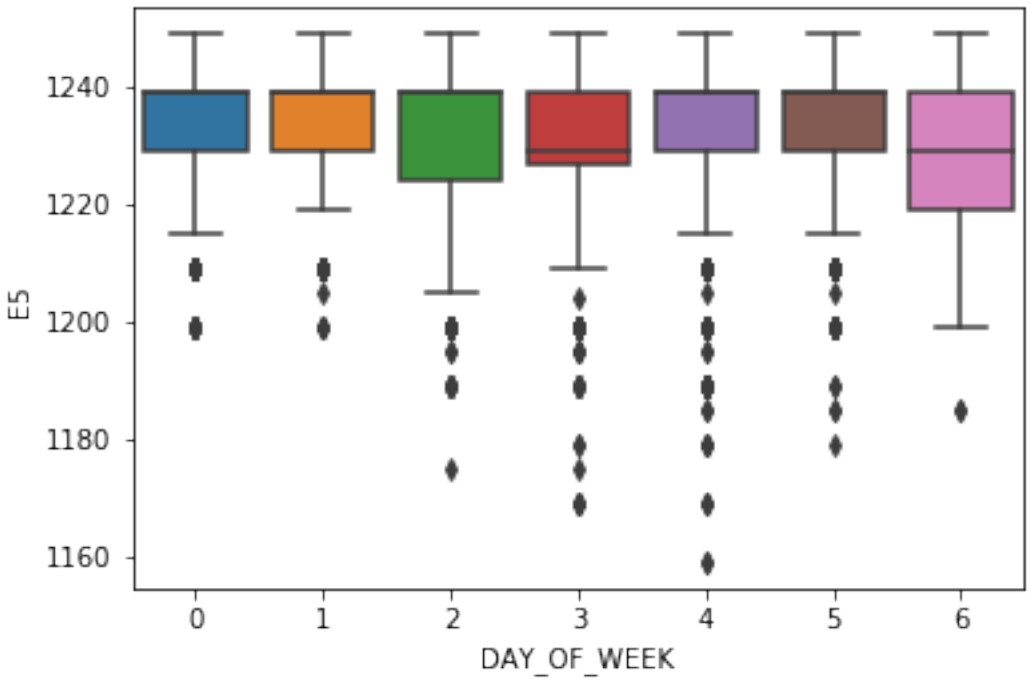
	DATE_CHANGED	CHANGED	formatted_date	DAY_OF_W
0	2014-10-02 05:26:01.000000	21	2014-10-02 05:26:01	
1	2014-10-02 05:22:01.000000	21	2014-10-02 05:22:01	
2	2014-10-02 09:06:01.000000	1	2014-10-02 09:06:01	
3	2014-10-03 12:14:01.000000	21	2014-10-03 12:14:01	

4	2014-10-03 14:14:01.000000	21	2014-10-03	14:14:01
5	2014-10-04 23:10:01.000000	21	2014-10-04	23:10:01
6	2014-10-06 06:02:01.000000	20	2014-10-06	06:02:01
7	2014-10-06 07:38:01.000000	1	2014-10-06	07:38:01
8	2014-10-06 08:26:01.000000	1	2014-10-06	08:26:01
9	2014-10-07 07:38:01.000000	1	2014-10-07	07:38:01
10	2014-10-07 10:18:01.000000	1	2014-10-07	10:18:01
11	2014-10-07 10:22:01.000000	20	2014-10-07	10:22:01
12	2014-10-10 23:10:01.000000	21	2014-10-10	23:10:01
13	2014-10-11 06:22:02.000000	21	2014-10-11	06:22:02
14	2014-10-11 23:10:01.000000	21	2014-10-11	23:10:01
15	2014-10-13 11:34:01.000000	21	2014-10-13	11:34:01
16	2014-10-14 12:14:01.000000	21	2014-10-14	12:14:01
17	2014-10-14 12:18:01.000000	20	2014-10-14	12:18:01
18	2014-10-15 14:10:01.000000	21	2014-10-15	14:10:01
19	2014-10-15 14:18:01.000000	21	2014-10-15	14:18:01
20	2014-10-15 14:46:01.000000	21	2014-10-15	14:46:01
21	2014-10-17 23:06:01.000000	21	2014-10-17	23:06:01
22	2014-10-19 09:46:01.000000	21	2014-10-19	09:46:01
23	2014-10-19 09:50:01.000000	21	2014-10-19	09:50:01
24	2014-10-20 12:18:01.000000	21	2014-10-20	12:18:01
25	2014-10-20 13:10:01.000000	21	2014-10-20	13:10:01
26	2014-10-20 14:14:01.000000	20	2014-10-20	14:14:01
27	2014-10-21 14:18:01.000000	21	2014-10-21	14:18:01
28	2014-10-21 18:14:01.000000	21	2014-10-21	18:14:01
29	2014-10-23 23:10:01.000000	21	2014-10-23	23:10:01
...	...	...	...	...
1033441	2015-04-30 16:54:01.000000	21	2015-04-30	16:54:01
1033442	2015-04-30 20:02:01.000000	21	2015-04-30	20:02:01
1033443	2015-04-30 22:06:01.000000	21	2015-04-30	22:06:01
1033444	2015-05-02 05:22:01.000000	21	2015-05-02	05:22:01
1033445	2015-05-02 06:22:01.000000	21	2015-05-02	06:22:01
1033446	2015-05-02 09:22:01.000000	21	2015-05-02	09:22:01
1033447	2015-05-03 12:22:01.000000	21	2015-05-03	12:22:01
1033448	2015-05-03 13:54:01.000000	21	2015-05-03	13:54:01
1033449	2015-05-04 16:50:01.000000	20	2015-05-04	16:50:01
1033450	2015-05-05 16:38:01.000000	20	2015-05-05	16:38:01
1033451	2015-05-05 16:58:01.000000	21	2015-05-05	16:58:01
1033452	2015-05-06 17:18:01.000000	20	2015-05-06	17:18:01
1033453	2015-05-06 20:02:01.000000	21	2015-05-06	20:02:01
1033454	2015-05-06 20:22:01.000000	20	2015-05-06	20:22:01
1033455	2015-05-06 22:02:01.000000	21	2015-05-06	22:02:01
1033456	2015-05-07 18:42:01.000000	21	2015-05-07	18:42:01
1033457	2015-05-07 20:02:01.000000	21	2015-05-07	20:02:01
1033458	2015-05-07 20:26:01.000000	21	2015-05-07	20:26:01
1033459	2015-05-07 22:06:01.000000	21	2015-05-07	22:06:01
1033460	2015-05-08 05:22:01.000000	1	2015-05-08	05:22:01
1033461	2015-05-08 20:02:01.000000	21	2015-05-08	20:02:01

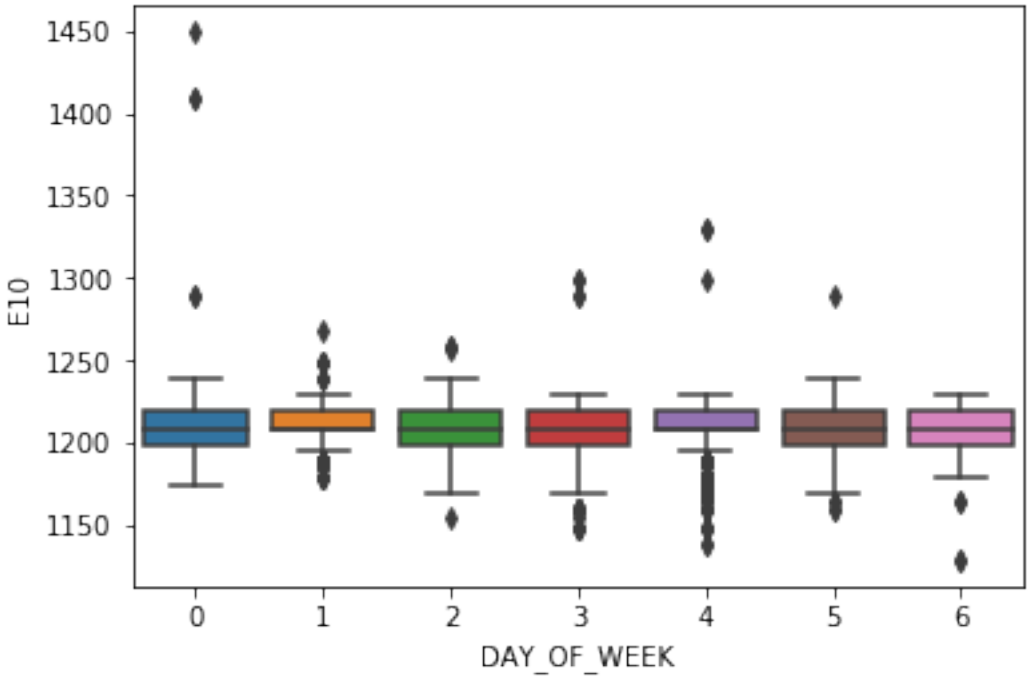
```
1033462  2015-05-08 22:02:01.000000       21 2015-05-08 22:02:01
1033463  2015-05-09 05:22:01.000000       21 2015-05-09 05:22:01
1033464  2015-05-10 05:22:01.000000       21 2015-05-10 05:22:01
1033465  2015-05-10 06:54:01.000000       21 2015-05-10 06:54:01
1033466  2015-05-11 12:22:01.000000       21 2015-05-11 12:22:01
1033467  2015-05-12 12:58:01.000000       21 2015-05-12 12:58:01
1033468  2015-05-12 14:42:01.000000        1 2015-05-12 14:42:01
1033469  2015-05-13 12:22:01.000000       21 2015-05-13 12:22:01
1033470  2015-05-13 16:46:01.000000       21 2015-05-13 16:46:01

[1032518 rows x 8 columns]
```

In [32]:
```
# So the approach here is to take 10000 cheapest price and then visualize
# using box plot
# Here days are denotted by numbers 0-6(Monday-Sunday)
df_cheap = df_2.nsmallest(10000,['E5','E10','DIESEL'],keep = 'all')
```

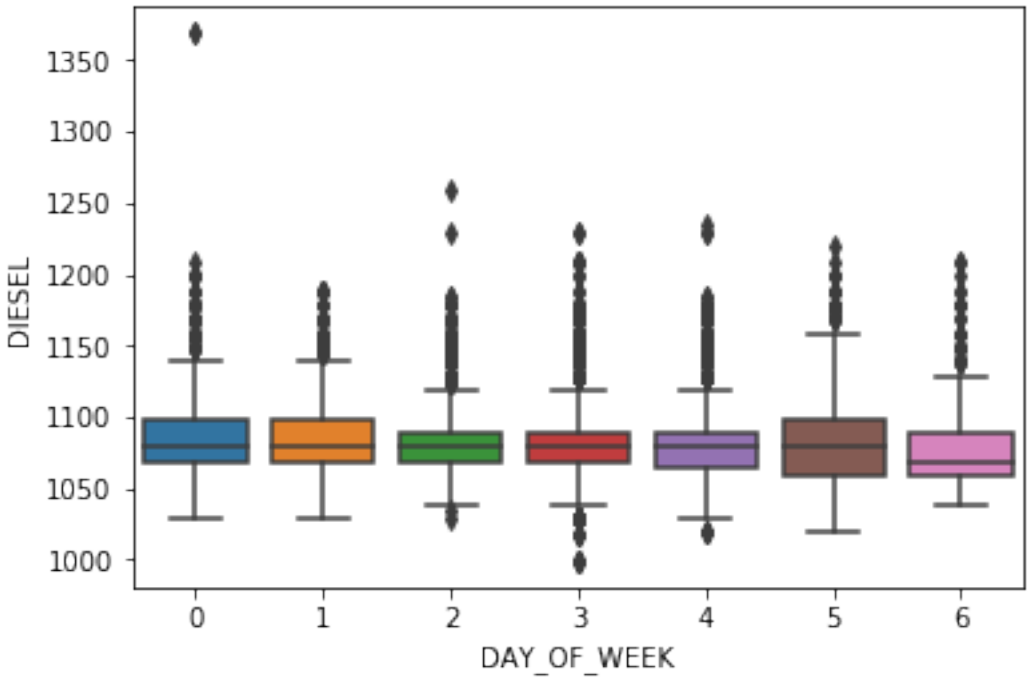
In [33]:
```
#Lets make boxplot of the data for E5
sns.boxplot(x='DAY_OF_WEEK', y = 'E5', data = df_cheap)
plt.show()
```



In [35]:
```
#Lets make boxplot of the data for E10
sns.boxplot(x='DAY_OF_WEEK', y = 'E10', data = df_cheap)
plt.show()
```



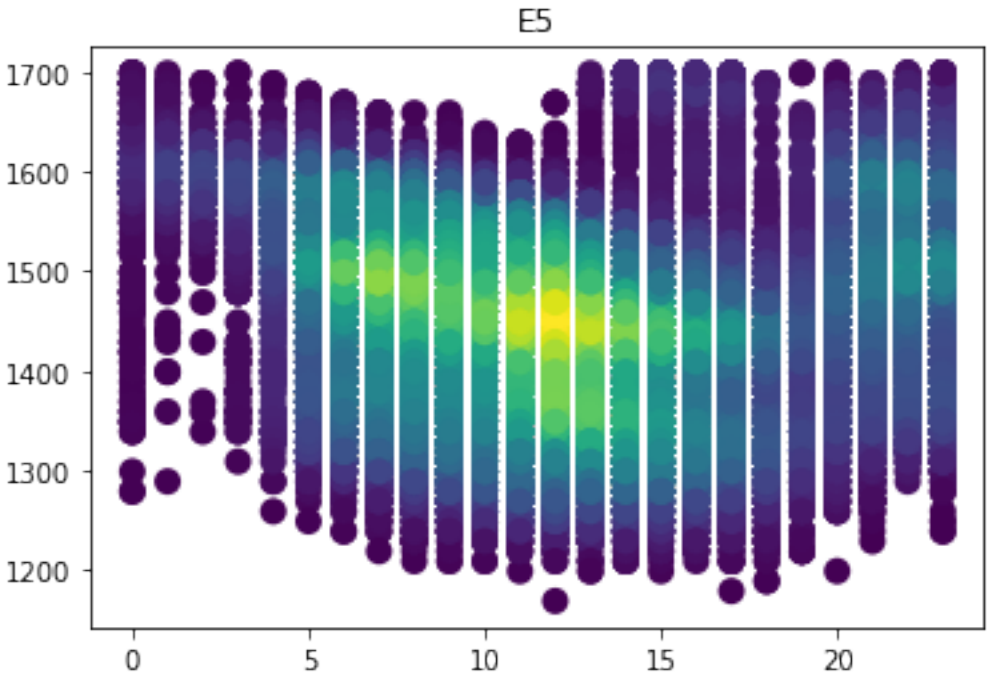
In [36]:
```
#Lets make boxplot of the data for DIESEL
sns.boxplot(x='DAY_OF_WEEK', y = 'DIESEL', data = df_cheap)
plt.show()
```



In [ ]:

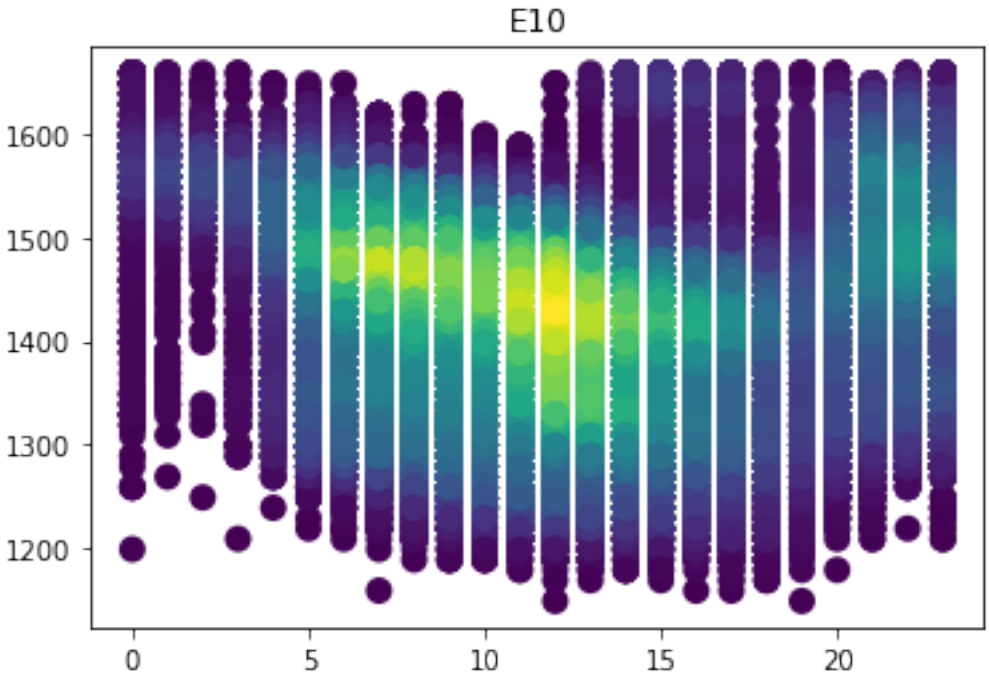
1.0.6 5. At which hour during a day is the price the cheapest in average (hour profile)?

```
In [37]: #Extract only hour from dataframe
         #df['hour'] = pd.to_datetime(df['time'], format='%H:%M').dt.hour
         df_2['formatted_time'] = pd.to_datetime(df_2['formatted_date']).dt.hour

In [38]: hourly = pd.DataFrame()
         hourly = df_2

In [39]: hourly['E5'] = df_2['E5']
         hourly['E10'] = df_2['E10']
         hourly['DIESEL'] = df_2['DIESEL']
         hourly['formatted_date'] = df_2['formatted_date']
         hourly['DATE_CHANGED'] = df_2['DATE_CHANGED']
         hourly['formatted_time'] = df_2['formatted_time']

In [40]: df_2['formatted_time'].head()

Out[40]: 0     5
         1     5
         2     9
         3    12
         4    14
         Name: formatted_time, dtype: int64

In [101]: from scipy.stats import gaussian_kde
          # Calculate the point density
          x = hourly['formatted_time'][0:50000]
          y = hourly['E5'][0:50000]
          xy = np.vstack([x,y])
          z = gaussian_kde(xy)(xy)
          plt.title('E5')

          plt.scatter(x, y, c=z, s=100, edgecolor='')
          plt.show()
```



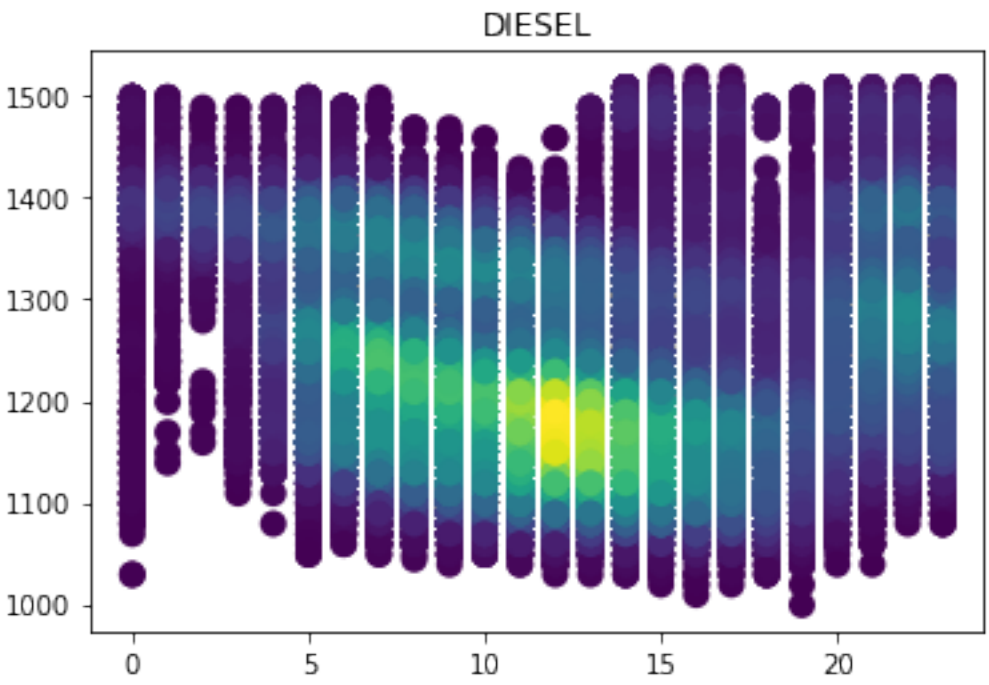
According to the density graph above, we see that the E5 gasoline is more frequently cheaper from 11-13hr.

```
In [85]: x = hourly['formatted_time'][0:100000]
         y = hourly['E10'][0:100000]
         xy = np.vstack([x,y])
         z = gaussian_kde(xy)(xy)
         plt.title('E10')

         plt.scatter(x, y, c=z, s=100, edgecolor='')
         #plt.scatter(hourly['formatted_time'][0:100000],hourly['E10'][0:100000])
         plt.show()
```



Same as above graph has cheaper E10 gasoline frequently around 11-13hr.

```
In [86]: x = hourly['formatted_time'][0:100000]
         y = hourly['DIESEL'][0:100000]
         xy = np.vstack([x,y])
         z = gaussian_kde(xy)(xy)
         plt.title('DIESEL')

         plt.scatter(x, y, c=z, s=100, edgecolor='')
         #plt.scatter(hourly['formatted_time'][0:100000],hourly['E10'][0:100000])
         plt.show()
```



We see that prices have been cheapest around 15-16hr (DIESEL).

In [ ]:

1.0.7 6. How many different station locations are present in the data (visualize via a map)?

In [ ]:

1.0.8 7. What is the gas station which has most price data points, choose one and draw the time series for all 3 gasoline types.

In [43]:
```
# Find the frequency of each station in the price data
station_freq_prices = df_2['STID'].value_counts().rename_axis('ID').reset_
```

In [44]:
```
station_freq_prices
```

Out[44]:

	ID	COUNTS
0	e50f3cfa-f32b-48e7-8304-a9f3d40d9996	6520
1	7aafc62a-a9e0-4cd2-83ad-345dfec172a0	6290
2	e6eb9adc-77d0-4f03-9549-799b691f3ff4	6009
3	daf56df5-bb5f-4297-a4bb-a0f444139d7f	5982
4	d133b8ca-83eb-431c-8b35-0ae3b955ebbe	5978
5	e991a489-6456-38af-e040-0b0a3dfe7c4f	5833
6	51d4b551-a095-1aa0-e100-80009459e03a	5822
7	51d4b553-a095-1aa0-e100-80009459e03a	5744
8	51d4b4e4-a095-1aa0-e100-80009459e03a	5700

9	14ec4b5b-18f3-4664-b768-1958f9e3b49f	5698
10	8d2cc77a-e636-403d-a4f6-9260ff4c9c16	5645
11	8607b08c-fdc8-4e54-8c0e-3403b333437d	5587
12	9247742f-2957-4777-b8b5-06786141aa14	5488
13	c393f0a6-2197-403a-ac89-e6a59be3ff50	5388
14	061716a5-f4a6-4983-a36e-8ef1e8ce17e0	5243
15	fdca8d3a-c7e7-4fe1-ad17-27772aab7d62	5219
16	7c848417-1490-4752-9766-d05859b23f72	5160
17	51d4b598-a095-1aa0-e100-80009459e03a	5144
18	0f2e0c50-51c1-4a4f-bfdb-c6f2eb676893	5092
19	51b4b85c-a7fc-4b60-bde2-bbee6031bb8d	5052
20	e935427e-bd37-4daf-979c-b4fe2d112981	4980
21	5374ea5c-bf3e-4b0a-8504-049368adcdba	4978
22	dbb0a607-7b91-42bf-af87-0c06c45991ed	4937
23	8a0bc647-d690-4e43-9e80-479e422af7df	4910
24	51d4b71a-a095-1aa0-e100-80009459e03a	4880
25	51d4b48c-a095-1aa0-e100-80009459e03a	4859
26	51d4b63b-a095-1aa0-e100-80009459e03a	4839
27	51d4b6bb-a095-1aa0-e100-80009459e03a	4823
28	17ac8e20-e554-4819-873d-0e6776201823	4786
29	51d4b432-a095-1aa0-e100-80009459e03a	4776
..	...	...
370	e80169b2-1a75-457c-9e92-8da3d77a0f5d	791
371	c615c080-915b-4d1e-8d65-54286e3c4acc	790
372	f0f73c6d-a1a6-4e06-dbb9-6fd2d1cf72df	785
373	a36a053d-2384-4d6b-a677-7f2d8abc3f01	776
374	044b327b-8726-428a-867c-a48235121cf2	756
375	8e04a261-815f-43f8-a5ba-20d12c5b3273	751
376	8210437c-c3e9-4b5a-a62a-8f18c2cee833	735
377	00060562-0001-4444-8888-acdc00000001	647
378	805a5d72-50bb-438a-a243-978a19ec2691	604
379	85aae4a5-89c0-44c7-b994-d8fdc325ca48	540
380	e8729196-1156-4f0d-e325-440e5a3745b3	487
381	f34f9241-6f55-48c6-aa4c-ee19f7272a22	482
382	79f03b0b-7324-402b-853b-16b667228112	481
383	00060710-0001-4444-8888-acdc00000001	466
384	23550389-c41c-43d1-9f34-a91270c59530	398
385	72046e37-447b-456e-86d8-7832c05aa88f	397
386	a00a98c1-6216-4ead-86ee-ae5df2035e1b	378
387	4b3c04a2-dc5a-44b9-ab23-dfb9072e115c	341
388	a0ad7e17-ff61-47cf-80b9-c17183527ee5	308
389	ba10c89e-d2f8-46d9-8ec5-6060bc32e89a	265
390	0c94d112-d3d4-4a98-a20d-138f7c5e1064	196
391	00060385-0001-4444-8888-acdc00000001	181
392	de1e26ff-6fb8-436e-b819-877ab581527a	109
393	eef314cf-a84a-4648-9284-c2df1efa36c9	96
394	c0cd87f5-64c3-4cb9-b424-9f3ac71e0678	80
395	f4f11a3c-a8d0-4742-874d-4f2d8d6b3572	80

```
396  12df4ed4-df41-45ff-8189-4b8a31236b2c      46
397  ef71bc50-b1f6-4fc6-ac96-6340c71c9bb9      46
398  01ddc2f2-35ca-4a02-a47c-190ebabe558b      18
399  82e92aca-b1ec-4bcf-3ab5-f192910c2716       9

[400 rows x 2 columns]
```

So we see that ID: 'e50f3cfa-f32b-48e7-8304-a9f3d40d9996'("AVIA Servicestation",AVIA) has the highest frequency of price data points. So lets plot the time series plot to visualize the price change in this gas station.

In [48]:
```
#Copy of df_2
df_22 = df_2
#df_22 = df_22.set_index('STID', inplace = True)
```

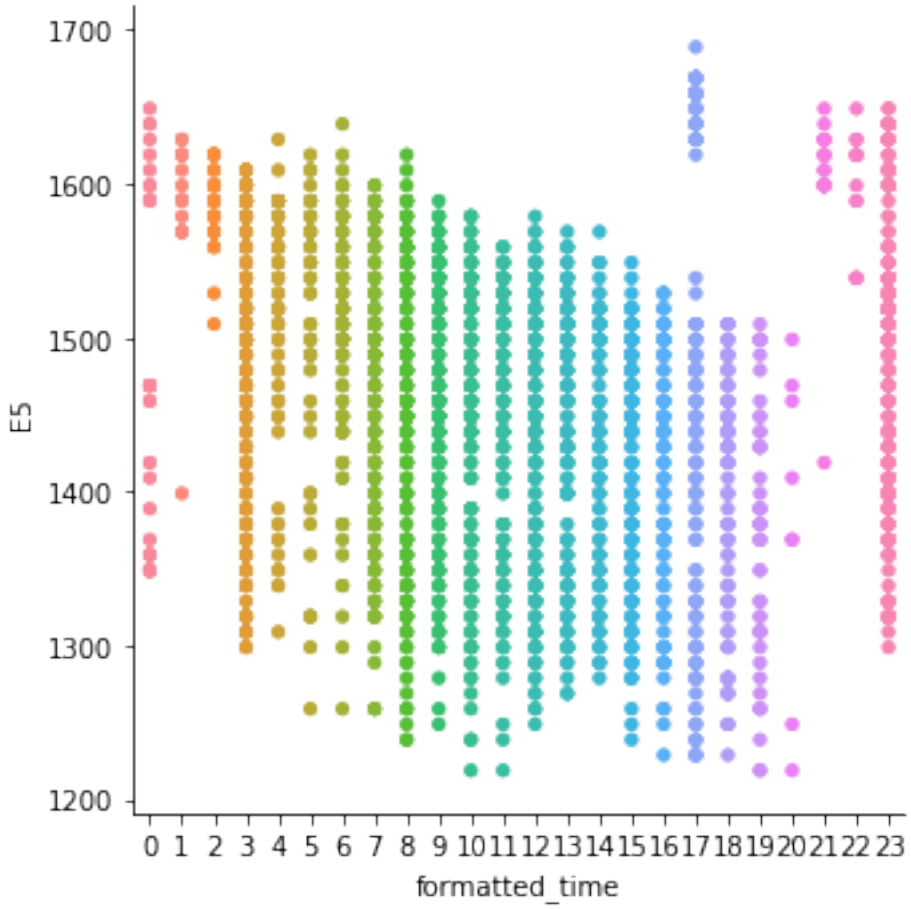
In [49]:
```
df_22.head()
```

Out[49]:

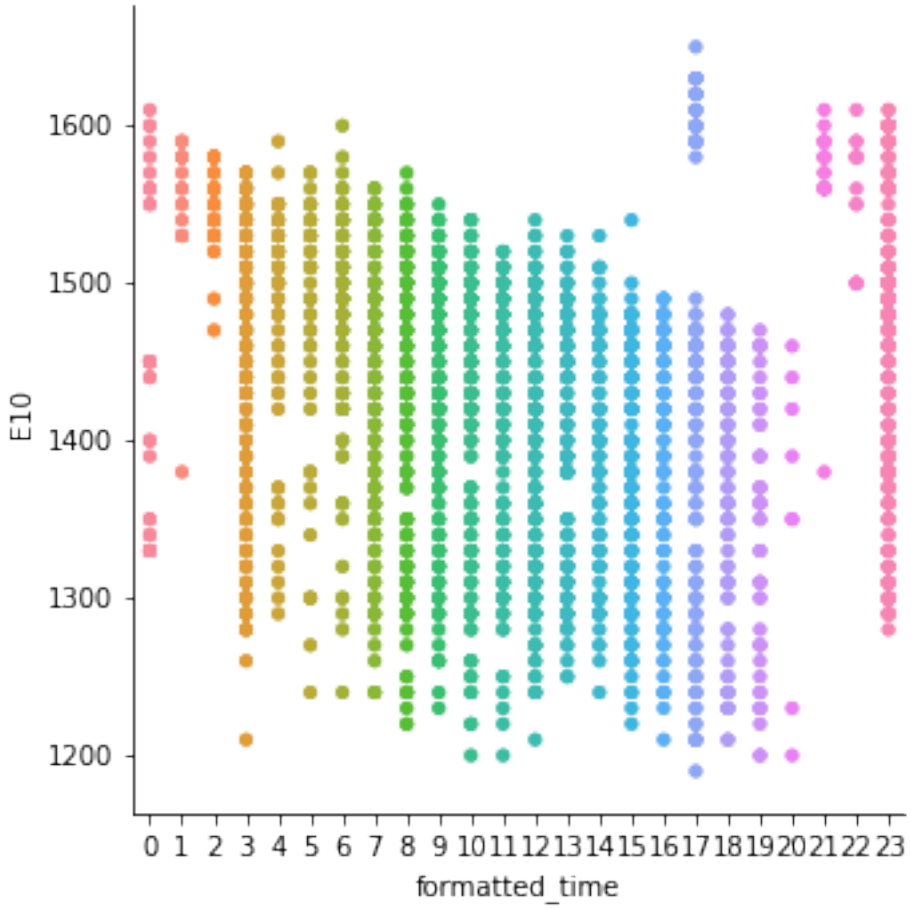
STID	E5	E10	DIESEL	DATE_CHANGED	CHANGED
01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1369	2014-10-02 05:26:01.000000	21
01bf5a06-4248-43a5-9524-56123fa3ab2f	1589	1549	1389	2014-10-02 05:22:01.000000	21
01bf5a06-4248-43a5-9524-56123fa3ab2f	1569	1529	1359	2014-10-02 09:06:01.000000	1
01bf5a06-4248-43a5-9524-56123fa3ab2f	1539	1499	1339	2014-10-03 12:14:01.000000	21
01bf5a06-4248-43a5-9524-56123fa3ab2f	1529	1489	1329	2014-10-03 14:14:01.000000	21

STID	formatted_date	DAY_OF_WEEK	formatted_time
01bf5a06-4248-43a5-9524-56123fa3ab2f	2014-10-02 05:26:01	3	5
01bf5a06-4248-43a5-9524-56123fa3ab2f	2014-10-02 05:22:01	3	5
01bf5a06-4248-43a5-9524-56123fa3ab2f	2014-10-02 09:06:01	3	9
01bf5a06-4248-43a5-9524-56123fa3ab2f	2014-10-03 12:14:01	4	12
01bf5a06-4248-43a5-9524-56123fa3ab2f	2014-10-03 14:14:01	4	14

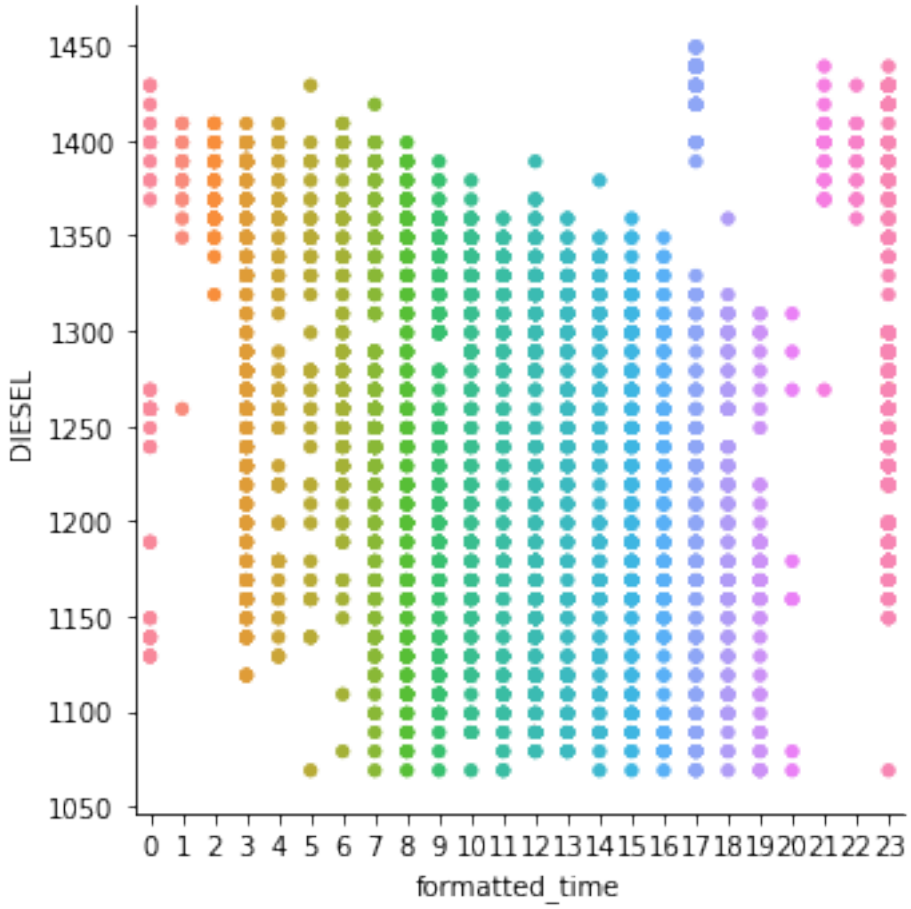
```
In [50]: # High freq data contains all the values of station 'AVIA' which has the h
         # price change
         high_freq_data = df_22.loc['e50f3cfa-f32b-48e7-8304-a9f3d40d9996']

In [51]: len(high_freq_data)

Out[51]: 6520

In [52]: high_freq_data.head()

Out[52]:                                          E5   E10  DIESEL  \
         STID
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  1369  1329    1219
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  1309  1269    1159
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  1309  1269    1149
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  1299  1259    1139
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  1419  1379    1269

                                                        DATE_CHANGED  CHANGED
         STID
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  2014-12-15 12:42:01.000000       21
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  2014-12-16 19:02:01.000000       21
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  2014-12-16 19:22:01.000000        1
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  2014-12-17 15:58:01.000000       21
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996  2014-11-03 21:17:00.000000       63

                                                    formatted_date  DAY_OF_WEEK  \
         STID
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996 2014-12-15 12:42:01            0
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996 2014-12-16 19:02:01            1
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996 2014-12-16 19:22:01            1
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996 2014-12-17 15:58:01            2
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996 2014-11-03 21:17:00            0

                                              formatted_time
         STID
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996              12
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996              19
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996              19
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996              15
         e50f3cfa-f32b-48e7-8304-a9f3d40d9996              21

In [54]: sns.catplot(x='formatted_time', y = 'E5', jitter = False, data = high_freq
         plt.show()
```



In [56]:
```
sns.catplot(x='formatted_time', y = 'E10', jitter = False, data = high_fre
plt.show()
```




```
In [57]: sns.catplot(x='formatted_time', y = 'DIESEL', jitter = False, data = high_
         plt.show()
```



In [ ]:

1.0.9 8. At which hour during a day do we have the most price changes?

In [ ]:

1.0.10 9. Select 20 gas stations having the longest time history and visualize the average price per month. Use heatmap and only the prices between 12:00-13:00 of e10 and diesel.

```
In [58]: #Top 20 stations with longest time history
         top_20 = df_22.loc[['e50f3cfa-f32b-48e7-8304-a9f3d40d9996','7aafc62a-a9e0-
                             'daf56df5-bb5f-4297-a4bb-a0f444139d7f','d133b8ca-83eb-4
                             '51d4b553-a095-1aa0-e100-80009459e03a','14ec4b5b-18f3-4
                             '8d2cc77a-e636-403d-a4f6-9260ff4c9c16','51d4b4e4-a095-1
                             '9247742f-2957-4777-b8b5-06786141aa14','c393f0a6-2197-4
                             'fdca8d3a-c7e7-4fe1-ad17-27772aab7d62','7c848417-1490-4
                             '51d4b598-a095-1aa0-e100-80009459e03a','51b4b85c-a7fc-4

In [59]: top_20.head()
```

Out[59]:

STID	E5	E10	DIESEL	DATE_CHANGED	CHANGED
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	1369	1329	1219		
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	1309	1269	1159		
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	1309	1269	1149		
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	1299	1259	1139		
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	1419	1379	1269		
e50f3cfa-f32b-48e7-8304-a9f3d40d9996				2014-12-15 12:42:01.000000	21
e50f3cfa-f32b-48e7-8304-a9f3d40d9996				2014-12-16 19:02:01.000000	21
e50f3cfa-f32b-48e7-8304-a9f3d40d9996				2014-12-16 19:22:01.000000	1
e50f3cfa-f32b-48e7-8304-a9f3d40d9996				2014-12-17 15:58:01.000000	21
e50f3cfa-f32b-48e7-8304-a9f3d40d9996				2014-11-03 21:17:00.000000	63

STID	formatted_date	DAY_OF_WEEK
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	2014-12-15 12:42:01	0
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	2014-12-16 19:02:01	1
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	2014-12-16 19:22:01	1
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	2014-12-17 15:58:01	2
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	2014-11-03 21:17:00	0

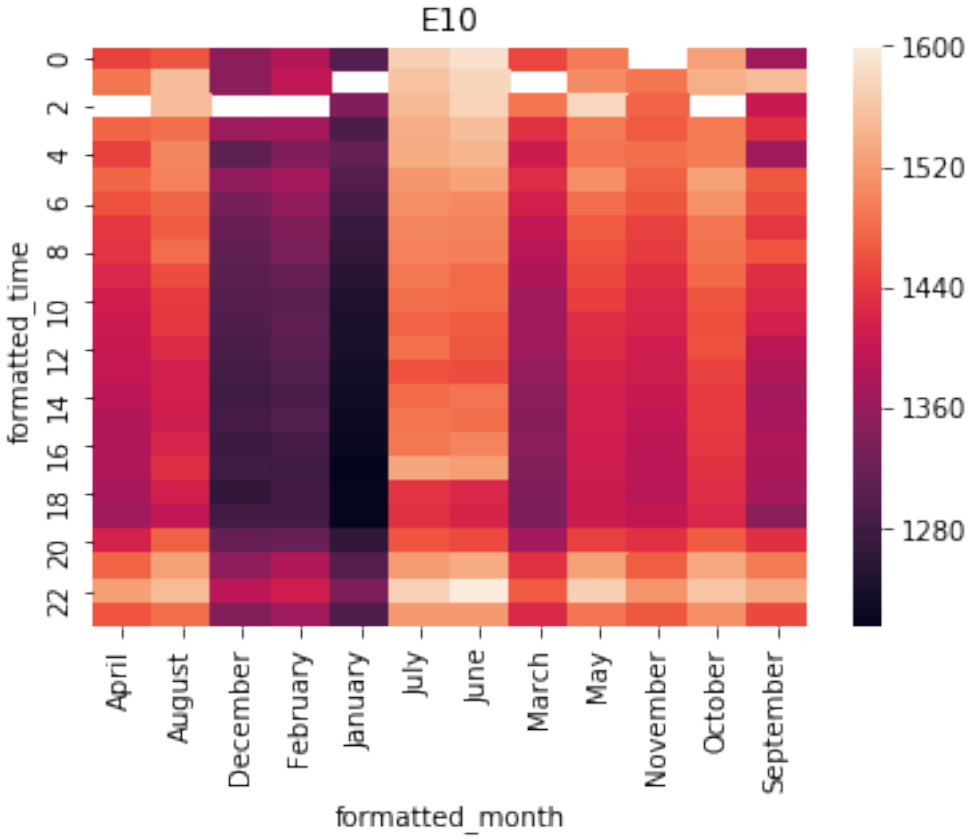
STID	formatted_time
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	12
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	19
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	19
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	15
e50f3cfa-f32b-48e7-8304-a9f3d40d9996	21

In [60]:
```
#Lets plot Heatmap
#df_2['formatted_time'] = pd.to_datetime(df_2['formatted_date']).dt.hour
top_20['formatted_month'] = pd.to_datetime(top_20['formatted_date']).dt.mo
#top_20['formatted_month'] = pd.to_datetime(top_20['formatted_date']).dt.m
top_20 = top_20.reset_index()
```

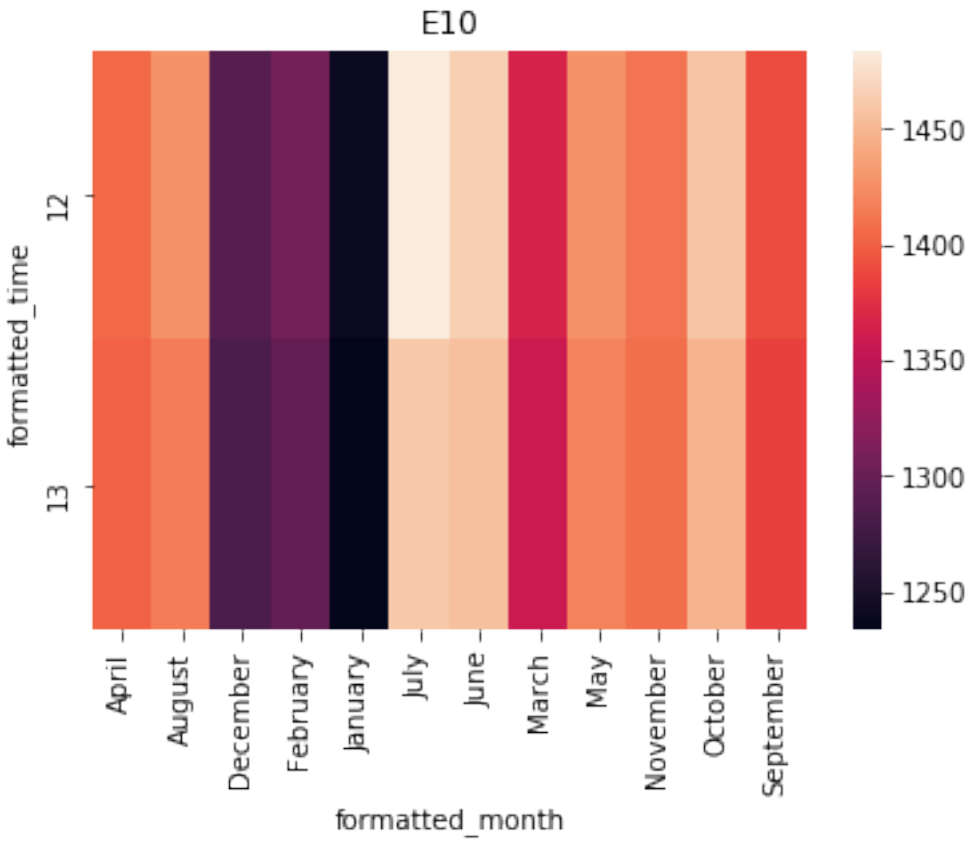
In [89]:
```
top_20_E10 = top_20[['formatted_time','formatted_month','E10']]
#Between 12-13hr
top_20_E10_12_13 = top_20_E10[top_20_E10['formatted_time'] >11]
top_20_E10_12_13 = top_20_E10_12_13[top_20_E10_12_13['formatted_time'] < 1
#sns.heatmap(top_20_E5)
#plt.matshow(top_20_E5)
#plt.show()
#type(top_20_E5['E5'][1])
#plt.matshow(top_20_E5)
```

In [73]:
```
top_20_E10 = top_20_E10.pivot_table(index = 'formatted_time', columns = 'f
```

```
In [74]: sns.heatmap(top_20_E10)
         plt.title("E10")
         plt.show()
```

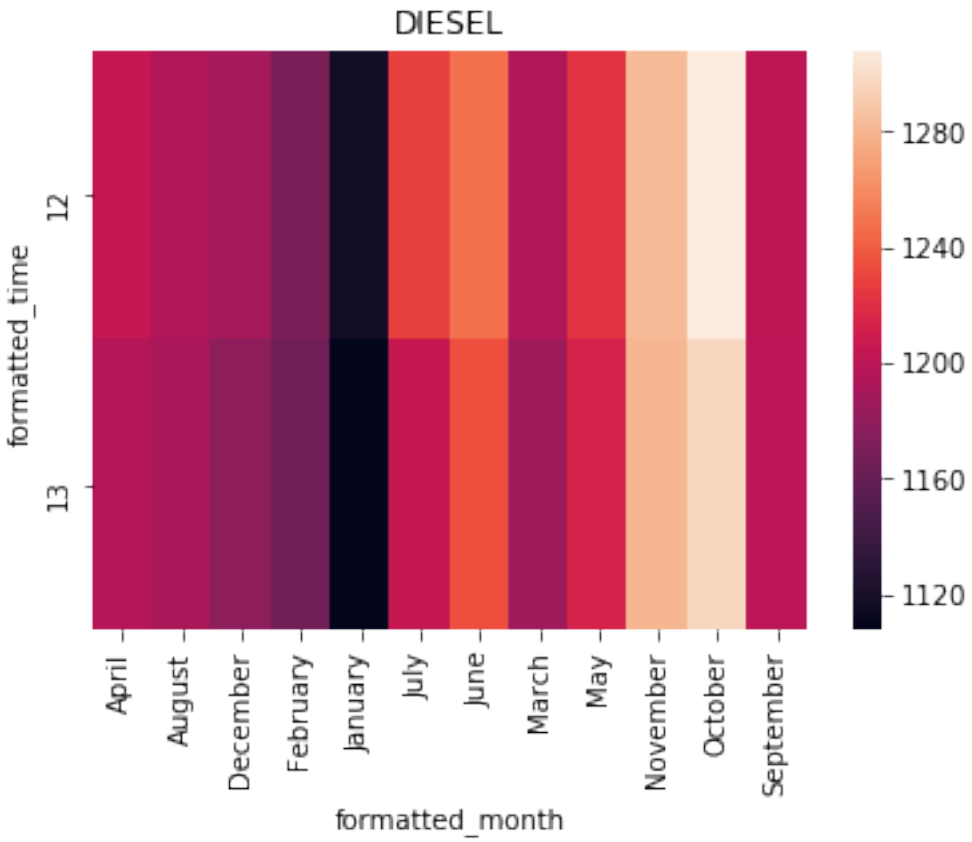



```
In [97]: top_20_E10 = top_20[['formatted_time','formatted_month','E10']]
         top_20_E10_12_13 = top_20_E10[top_20_E10['formatted_time'] >11]
         top_20_E10_12_13 = top_20_E10_12_13[top_20_E10_12_13['formatted_time'] < 1

In [98]: top_20_E10 = top_20_E10.pivot_table(index = 'formatted_time', columns = 'f

In [99]: top_20_E10_12_13 = top_20_E10_12_13.pivot_table(index = 'formatted_time',
         sns.heatmap(top_20_E10_12_13)
         plt.title("E10")
         plt.show()
```



We can see that the Price is highest in July, June and October.

```
In [69]: #Same can be done for DIESEL
         top_20_DIESEL = top_20[['formatted_time','formatted_month','DIESEL']]
         top_20_DIESEL_12_13 = top_20_DIESEL[top_20_DIESEL['formatted_time'] >11]
         top_20_DIESEL_12_13 = top_20_DIESEL_12_13[top_20_DIESEL_12_13['formatted_t

In [70]: top_20_DIESEL = top_20_DIESEL.pivot_table(index = 'formatted_time', columr

In [71]: top_20_DIESEL_12_13 = top_20_DIESEL_12_13.pivot_table(index = 'formatted_t
         sns.heatmap(top_20_DIESEL_12_13)
         plt.title("DIESEL")
         plt.show()
```



We see that prices are higheset in October and November.

In [ ]:

1.0.11 10. Describe a possible business potential in € for the customer (textual description in the ipyhton file). Define the constraints of the business case 5 lines, the answer max 15 lines (high level summary)

The Customers can enjoy benefits of lower gasoline prices by following few guidelines: 1. For DIESEL (Oct-Nov) and for E10 (Jun-Jul & Oct) have been the costliest. Be prepared and fill your gas tanks in advance or try to minimize your gasoline use in these months. 2. Thursdays are the cheapest on average, so fill your gas tanks on Thursdays(Don't wait for Weekend). 3. E5 and E10 are in general cheaper around 11-13hr and DIESEL around 15-16hr. 4. Globus Handelshof GmbH & Co.KG Betriebsstätte is the cheapest gas station in general.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: