improvibar

February 15, 2019

1 shop results

1.1 goal

- find factors influencing this shop's results
- predict results

```
In [1]: from itertools import product

        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns

        %matplotlib inline
```

1.2 Load data

```
In [2]: from datetime import datetime
        from os import path, scandir

        daily_datadir = "./data/CaisseJour/"
        datadirs = [path.join(daily_datadir, d.name) for d in scandir(daily_datadir)]
        data_files = [
            path.join(datadir, file.name)
            for datadir in datadirs
            for file in scandir(datadir)
        ]
```

```
In [3]: def parse_caisse(filename, keywords=("Chiffre", "TVA", "nombre", "moyen", "ticket")):
            """Parse file "caisse jour"

            Args:
                filename (string): file to parse
                keywords (list): list of keywords for  one line data
            """
            data = {}
            with open(filename, "br") as fd:
```

```
            for line in fd:
                line = line.decode("Windows-1252",errors="ignore")
                if "à" in line:
                    # try with date
                    try:
                        date = [int(d) for d in  line.split(" ")[0].split("/")]
                    except ValueError:
                        # "à" in cocktail name
                        continue
                    data["date"] = datetime(date[2], date[1], date[0])
                elif any(keyword in line for keyword in keywords):
                    value = line.split(";")[1]
                    value = value.strip(" \r\n")
                    try:
                        # parse french number representation
                        value = value.replace(",", ".")
                        value = float(value)
                    except ValueError:
                        # not a number, cannot convertto float
                        pass
                    data[line.split(";")[0].strip()] = value
            #TODO: add small tables
        return data

In [4]: daily = pd.DataFrame(parse_caisse(f) for f in data_files)
        daily.index = daily["date"]  # keep date and index

In [5]: daily.head()

Out[5]:            Chiffre d'Affaires HT  Chiffre d'Affaires TTC  Coefficient moyen  \
        date
        2018-11-09                 545.26                   633.9                0.0
        2018-11-03                 242.34                   285.8                0.0
        2018-11-10                1370.21                  1616.0                0.0
        2018-11-22                 153.58                   182.0                0.0
        2018-11-02                 394.43                   459.7                0.0

                     Nom TVA  Nombre moyen de produits / Ticket  TVA Collecté  \
        date
        2018-11-09  Taux TVA                                1.6         88.64
        2018-11-03  Taux TVA                                1.6         43.46
        2018-11-10  Taux TVA                                1.8        245.79
        2018-11-22  Taux TVA                                1.2         28.42
        2018-11-02  Taux TVA                                1.5         65.27

                   TVA Vente 10% TVA Vente 20%  Ticket moyen TTC       date
        date
        2018-11-09          10 %          20 %             10.06 2018-11-09
```

```
2018-11-03          10 %          20 %             10.99 2018-11-03
2018-11-10          10 %          20 %             11.14 2018-11-10
2018-11-22          10 %          20 %              7.00 2018-11-22
2018-11-02          10 %          20 %             10.95 2018-11-02
```

1.3 Calendar

```
In [6]: start_date = min(daily["date"])
        end_date = max(daily["date"])

        #start_date = datetime(2018, 09, 01)
        #end_date = datetime(2019, 09, 01)
```

```
In [7]: def date_to_monthweek(date):
            """Return the week number of the month, i.e. the number of mondays before this dat

            Args:
                date: (datetime.datetime)

            Return:
                int: the week number
            """
            return len(
                [
                    day
                    for day in pd.date_range(datetime(date.year, date.month, 1), date)
                    if day.weekday() == 0
                ]
            )
```

```
In [8]: calendar = pd.DataFrame(pd.date_range(start_date, end_date), columns=("date",))
        #calendar["day", "month", "year", "wod"] = list(map(lambda x: (x.day, x.month, x.year,
        calendar["day"] = list(map(lambda x: x.day, calendar["date"]))
        calendar["month"] = list(map(lambda x: x.month, calendar["date"]))
        calendar["year"] = list(map(lambda x: x.year, calendar["date"]))
        calendar["dow"] = list(map(lambda x: x.weekday(), calendar["date"]))
        calendar["week number"] = list(map(lambda x: x.isocalendar()[1], calendar["date"]))
        calendar["month week number"] = list(map(date_to_monthweek, calendar["date"]))
```

1.3.1 Holidays

from https://date.nager.at/PublicHoliday/DownloadCSV/FR/2018

```
In [9]: datadir = "./data/calendars"
        data_files = [path.join(datadir, file.name) for file in scandir(datadir)]

        holidays = pd.concat(
            [
                pd.read_csv(file)
```

```
            for file in data_files
        ]
    )

    # reformat date
    holidays["Date"] = pd.Series(
        [
            datetime(int(x.split("-")[0]), int(x.split("-")[1]), int(x.split("-")[2]))
            for x in holidays["Date"]
        ]
    )
```

In [10]:
```
holidays.head()
```

Out[10]:

	Date	LocalName	Name	CountryCode	Fixed	Global	LaunchYear
0	2018-01-01	Jour de l'an	New Year's Day	FR	True	True	1967.0
1	2018-03-30	Vendredi saint	Good Friday	FR	False	False	NaN
2	2018-04-02	Lundi de Pâques	Easter Monday	FR	False	True	1642.0
3	2018-05-01	Fête du premier mai	Labour Day	FR	True	True	NaN
4	2018-05-08	Fête de la Victoire	Victory in Europe Day	FR	True	True	NaN

In [11]:
```
calendar["public holidays"] = list(map(lambda x: x in list(holidays["Date"]), calendar
```

In [12]:
```
calendar = calendar.set_index("date")
```

In [13]:
```
calendar.head()
```

Out[13]:

date	day	month	year	dow	week number	month week number	public holidays
2018-08-29	29	8	2018	2	35	4	False
2018-08-30	30	8	2018	3	35	4	False
2018-08-31	31	8	2018	4	35	4	False
2018-09-01	1	9	2018	5	35	0	False
2018-09-02	2	9	2018	6	35	0	False

join data

```
In [14]: # the pandas way
         daily = daily.join([calendar])

         # the spark.sql way

         daily.head()
```

Out[14]:

date	Chiffre d'Affaires HT	Chiffre d'Affaires TTC	Coefficient moyen	Nom TVA	Nombre moyen de produits / Ticket	TVA Collecté	TVA Vente 10%	TVA Vente 20%	Ticket moyen TTC	date	day	month	year	dow	week number	month week number	public holidays
2018-08-29	88.99	105.6	0.0	Taux TVA	1.3	16.61	10 %	20 %	7.04	2018-08-29	29	8	2018	2	35	4	False
2018-08-30	115.37	134.1	0.0	Taux TVA	2.7	18.73	10 %	20 %	19.16	2018-08-30	30	8	2018	3	35	4	False
2018-08-31	91.39	108.3	0.0	Taux TVA	2.1	16.91	10 %	20 %	10.83	2018-08-31	31	8	2018	4	35	4	False
2018-09-01	196.80	231.7	0.0	Taux TVA	2.2	34.90	10 %	20 %	14.48	2018-09-01	1	9	2018	5	35	0	False
2018-09-05	56.00	67.2	0.0	Taux TVA	10.0	11.20	NaN	20 %	67.20	2018-09-05	5	9	2018	2	36	1	False

1.4 Weather

from meteofrance

```
In [ ]:
```

1.5 Data exploration

Describe and restrict features

In [15]:
```
from pandas.plotting import scatter_matrix
```

In [16]:
```
# data description
columns_descr = {
    "Chiffre d'Affaires HT": "(float) Income (taxes excluded)",
    "Chiffre d'Affaires TTC": "(float) Income (taxes included)",
    "Nombre moyen de produits / Ticket": "(float) Mean good numbers per transaction",
    "Ticket moyen TTC": "(float) Mean transaction value",
    "date": "(date) date of the day",
    "dow": "(int) day of week, 0..7",
    "day": "(int) day in month",
    "month": "(int) month number",
    "week number": "(int) iso week number (0..53)",
    "month week number": "(int) month week number (0..5)",
    "year": " (int) year",
    "public holidays": "(bool) Public holiday in France",
}
```

In [17]:
```
cols = columns_descr.keys()
daily = daily.loc[:, columns_descr.keys()]
daily["public holidays"] = daily["public holidays"].apply(lambda x: 1 if x is True els
daily.head()
```

Out[17]:

date	Chiffre d'Affaires HT	Chiffre d'Affaires TTC
2018-08-29	88.99	105.6
2018-08-30	115.37	134.1
2018-08-31	91.39	108.3
2018-09-01	196.80	231.7
2018-09-05	56.00	67.2

date	Nombre moyen de produits / Ticket	Ticket moyen TTC	date
2018-08-29	1.3	7.04	2018-08-29
2018-08-30	2.7	19.16	2018-08-30
2018-08-31	2.1	10.83	2018-08-31
2018-09-01	2.2	14.48	2018-09-01
2018-09-05	10.0	67.20	2018-09-05

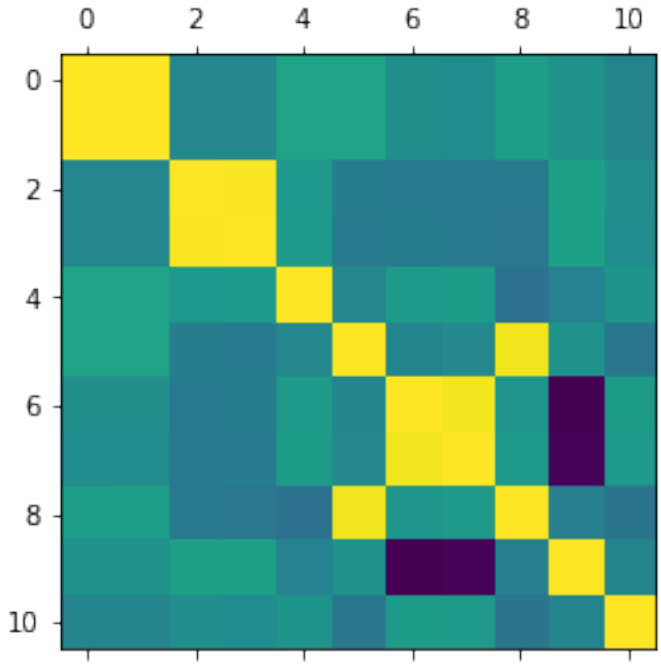
date	dow	day	month	week number	month week number	year
2018-08-29	2	29	8	35	4	2018
2018-08-30	3	30	8	35	4	2018
2018-08-31	4	31	8	35	4	2018
2018-09-01	5	1	9	35	0	2018
2018-09-05	2	5	9	36	1	2018

date	public holidays

```
date
2018-08-29                0
2018-08-30                0
2018-08-31                0
2018-09-01                0
2018-09-05                0
```

```
In [18]: plt.matshow(daily.corr())
```

```
Out[18]: <matplotlib.image.AxesImage at 0x7f07c87a5860>
```



There are highly correlate features, some may be discarded. Moreover, some features are redondant.

```
In [19]: del columns_descr["Chiffre d'Affaires HT"]
```

```
In [20]: # extract numeric features
         num_cols = [
             col
             for col in columns_descr.keys()
             if np.isreal(daily.loc[start_date, col])
         ]
         num_cols
         #scatter_matrix(daily)
```

```
Out[20]: ["Chiffre d'Affaires TTC",
          'Nombre moyen de produits / Ticket',
```

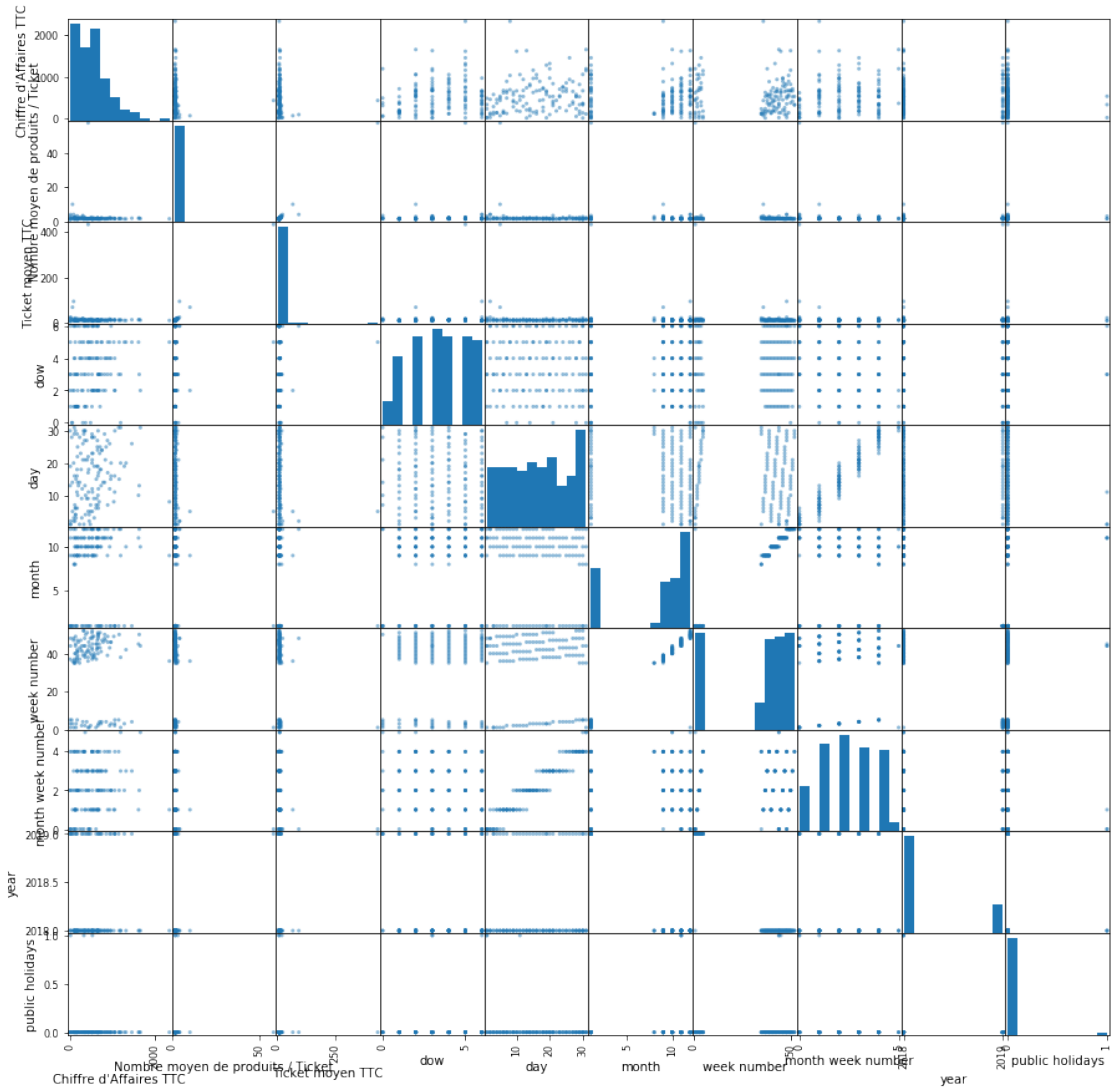
```
          'Ticket moyen TTC',
          'date',
          'dow',
          'day',
          'month',
          'week number',
          'month week number',
          'year',
          'public holidays']
```

```
In [21]: scatter_matrix(daily.loc[:, num_cols], figsize=(15, 15))
```

```
Out[21]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8724208>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8744e10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c86f4358>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c871d8d0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c86c5e48>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8673400>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8699978>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8644f28>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8644f60>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c861ba20>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c85c1f98>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8571550>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c859aac8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c854b080>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c84f45f8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c851cb70>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c84cd128>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c84736a0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8499c18>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c844b1d0>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c83f4748>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c839fcc0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c83ce278>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c83767f0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c831ed68>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c834e320>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c82f6898>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c829ee10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c82d03c8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8276940>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8221eb8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8250470>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c81f59e8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c81a1f60>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c81d2518>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8177a90>,
```

```
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c812d048>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c81535c0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c80fbb38>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c80ad0f0>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c80d6668>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c807dbe0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c802d198>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c8055710>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7ffcc88>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7fae240>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7fd57b8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7f7ed30>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7f2f2e8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7f56860>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7f00dd8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7eb0390>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7ed8908>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7e81e80>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7e30438>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7e589b0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7dfff28>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7db14e0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7dd9a58>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7d83fd0>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7d32588>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7d59b00>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7d0b0b8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7cb2630>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7cdbba8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7c8c160>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7c346d8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7bdec50>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7c0d208>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7bb4780>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7bdccf8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7b8e2b0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7b36828>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7adfda0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7b10358>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7ab88d0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7a5fe48>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7a92400>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7a38978>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c79dfef0>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7a114a8>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c79b9a20>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7960f98>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7992550>,
```

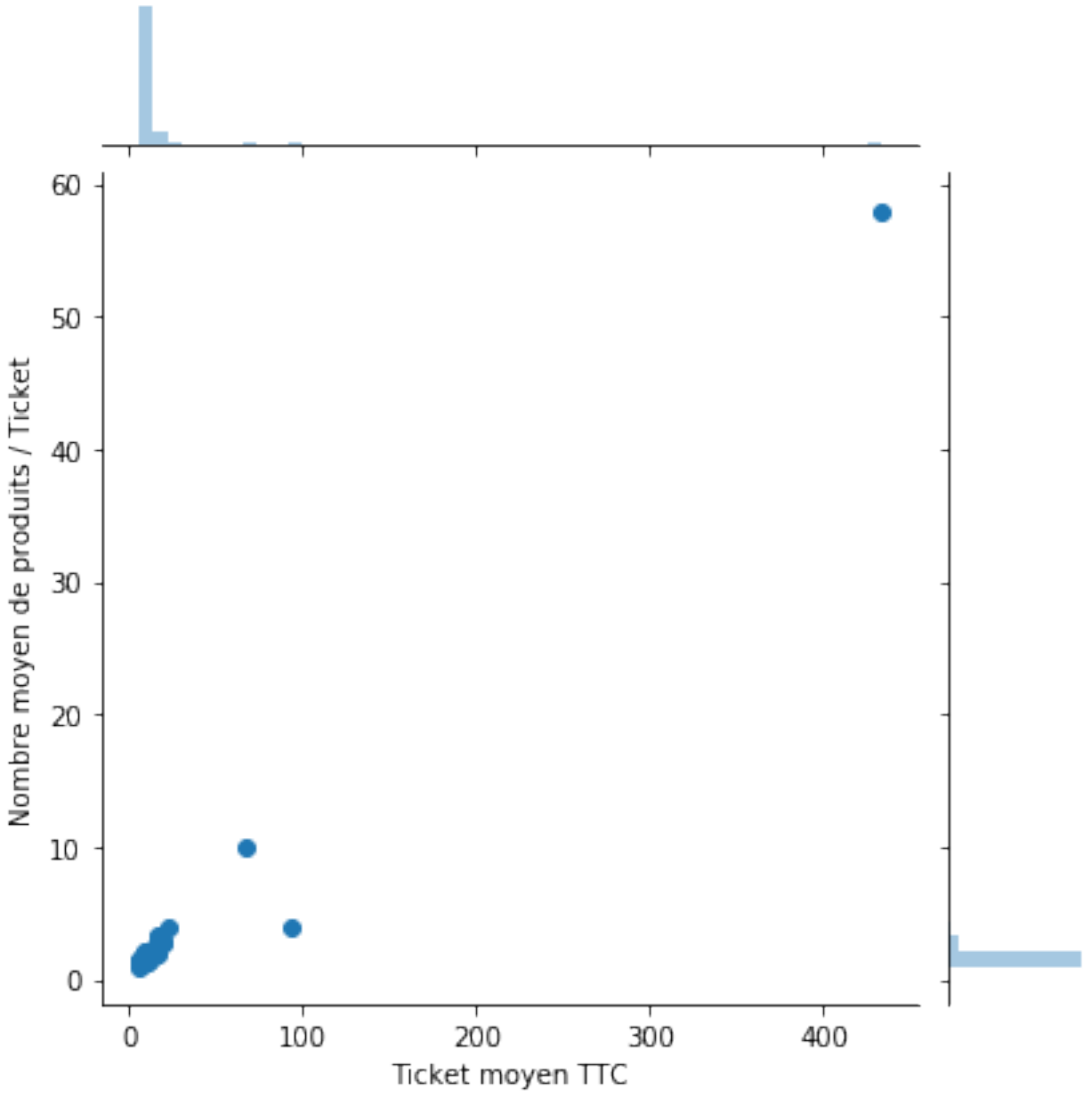
```
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c793aac8>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c78eb080>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c79145f8>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c78bbb70>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c786d128>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c78926a0>],
 [<matplotlib.axes._subplots.AxesSubplot object at 0x7f07c783dc18>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c77ed1d0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7813748>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c77becc0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c776f278>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c77947f0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c773fd68>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c76f0320>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c7716898>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x7f07c76bee10>]],
dtype=object)
```



- The Income seems to vary each month
- There are outliers in income
- Mean product per transaction and mean transaction number is highly correlated (as expected) except for some days (must be treated separately)

```
In [22]: sns.jointplot(
             data=daily,
             x="Ticket moyen TTC",
             y="Nombre moyen de produits / Ticket",
         )
```

```
Out[22]: <seaborn.axisgrid.JointGrid at 0x7f07c6ad7dd8>
```



- 3 days with higher mean transaction
- 1 day with the mean product price is higher than usual

1.5.1 Transaction outliers

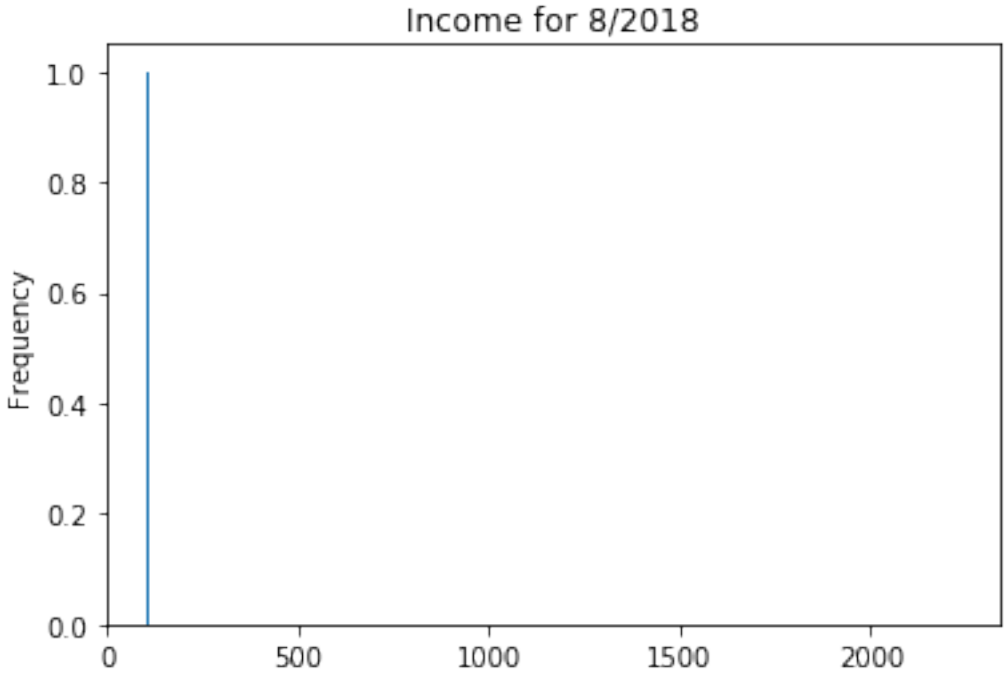
The goal of this section is to know if this outliers should be discarded.

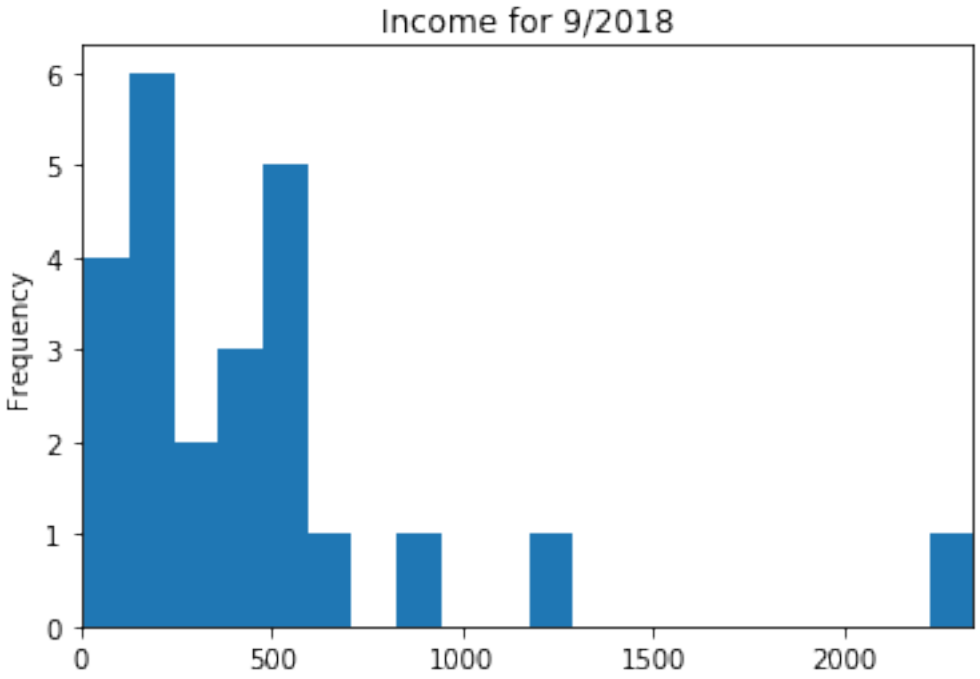
In [ ]:

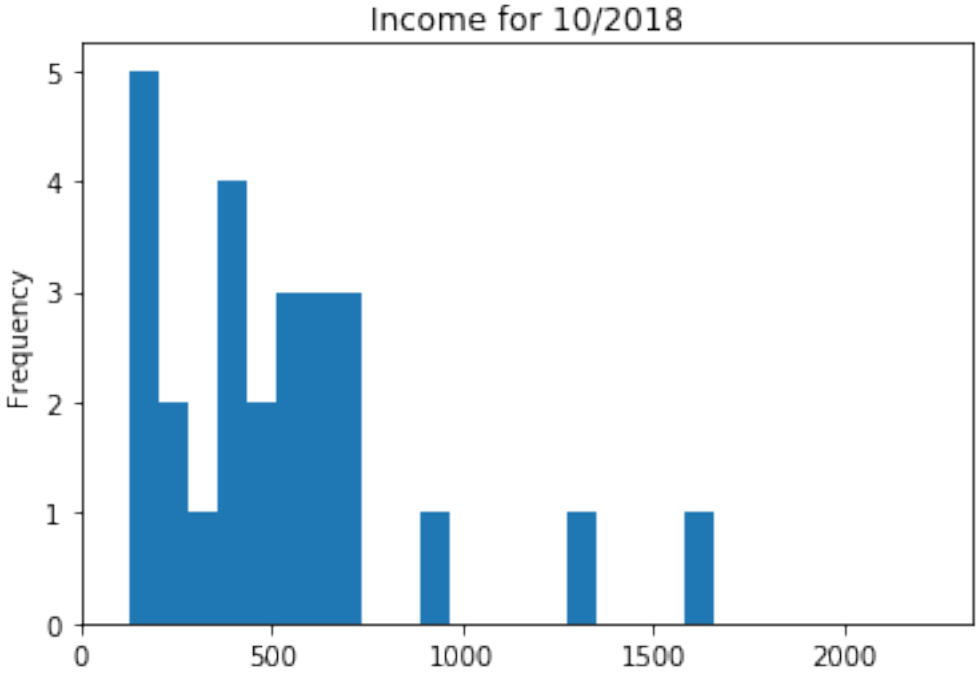
1.5.2 Per month income

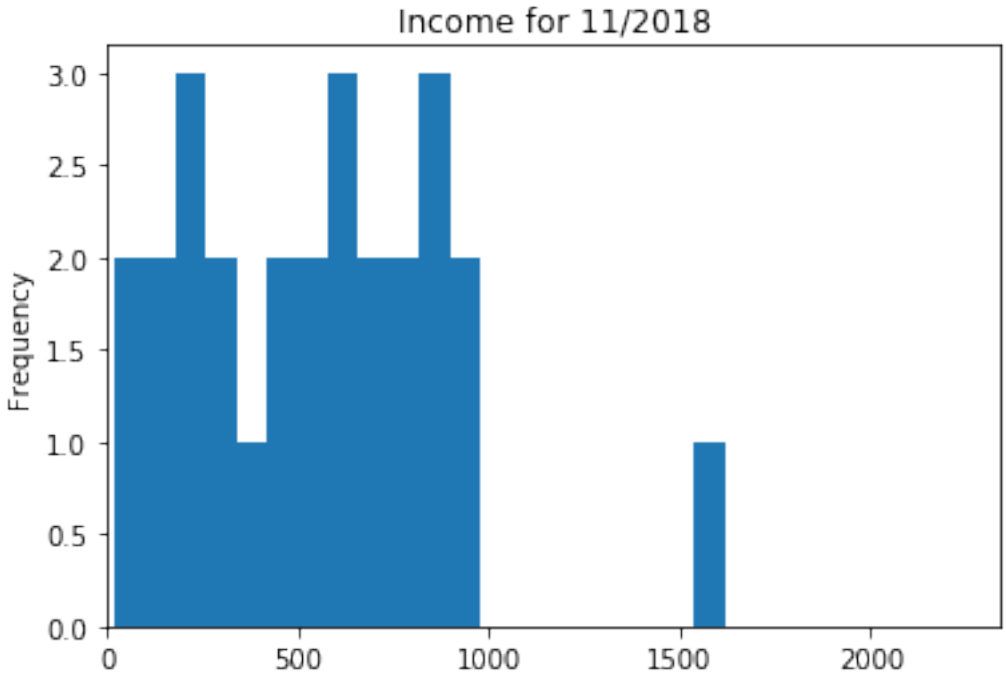
```
In [23]: xlim = (0, max(daily["Chiffre d'Affaires TTC"]))
         for year, month in product(range(2018, 2020), range(1, 13)):
```

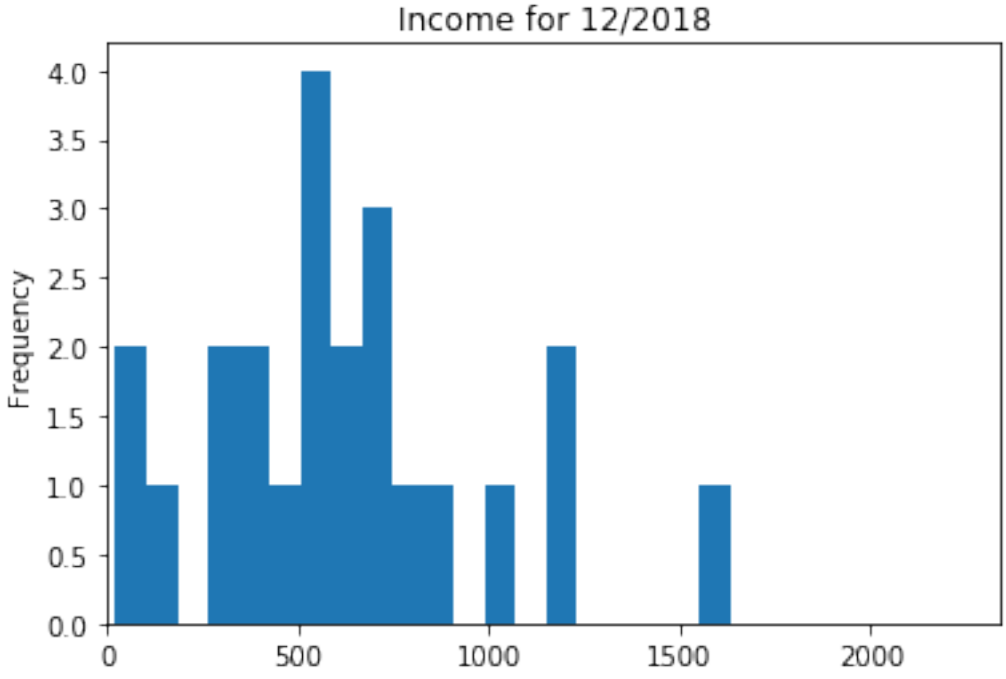
```
cur_data = daily.loc[(daily["month"] == month) & (daily["year"] == year)]
if len(cur_data) == 0:
    continue
plt.figure()
#cur_data["Chiffre d'Affaires TTC"].plot(kind="box",)
cur_data["Chiffre d'Affaires TTC"].plot(
    kind="hist",
    xlim=xlim,
    title="Income for {}/{}".format(month, year),
    bins=20
)
```

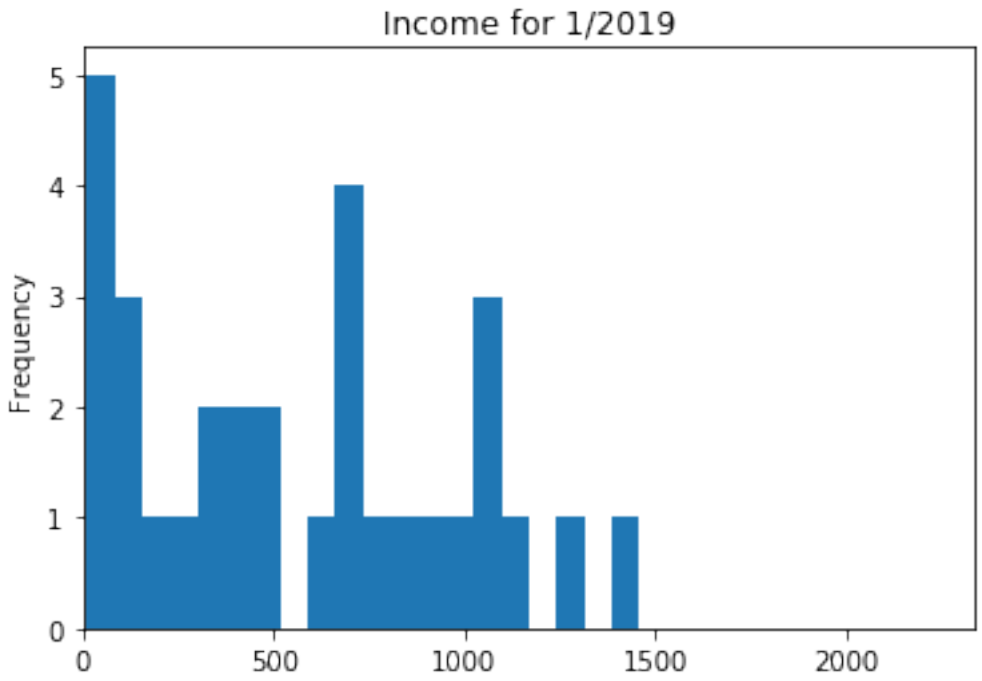



Income for 9/2018
Frequency
0
1
2
3
4
5
6
0
500
1000
1500
2000

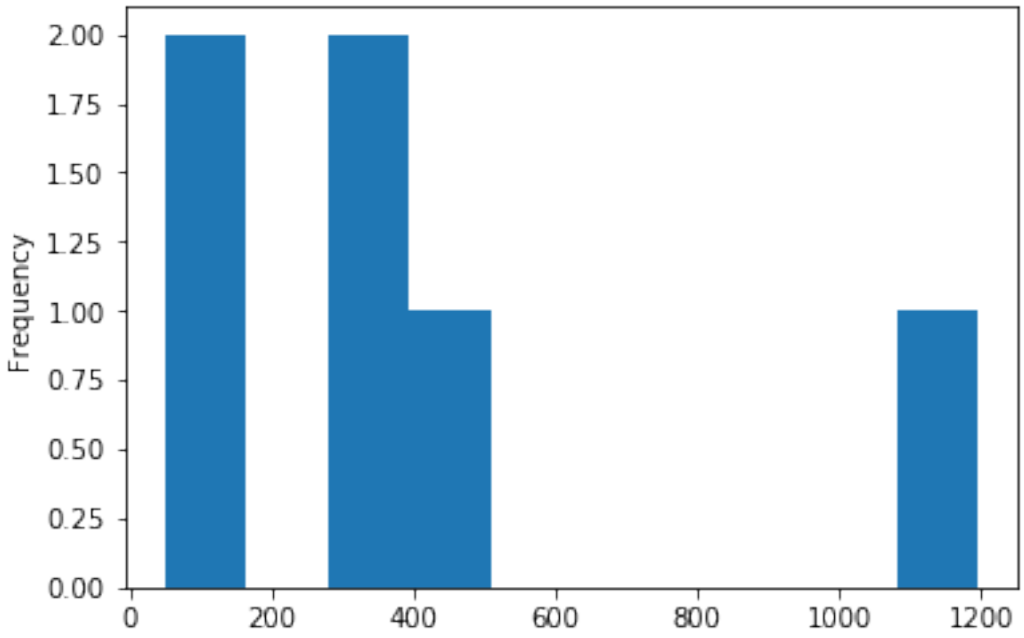

Income for 10/2018
Frequency
0
1
2
3
4
5
0
500
1000
1500
2000

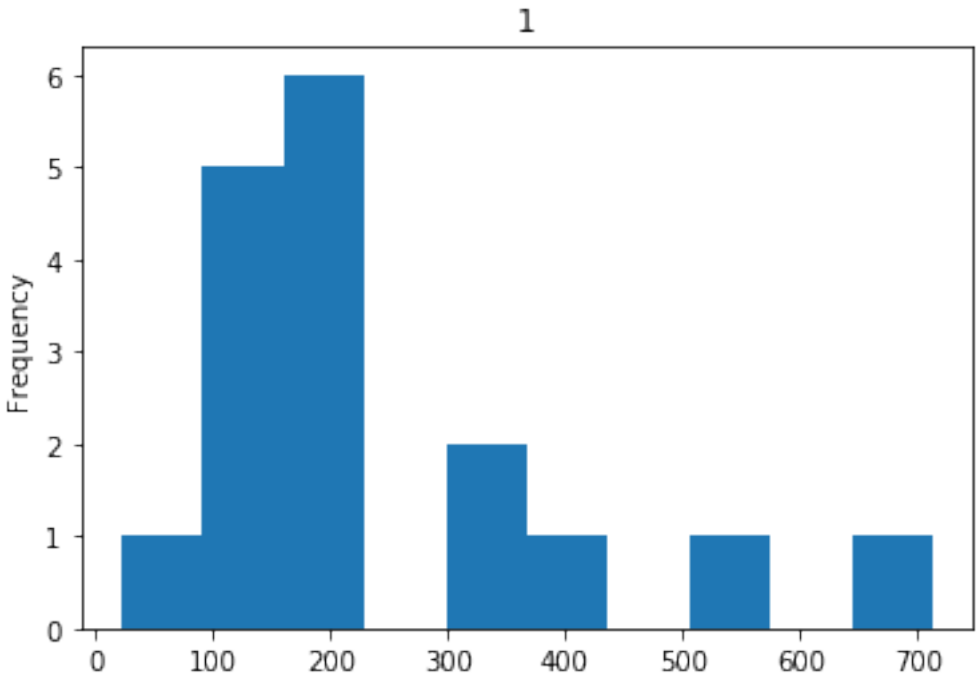

Income for 11/2018
Frequency
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
500
1000
1500
2000

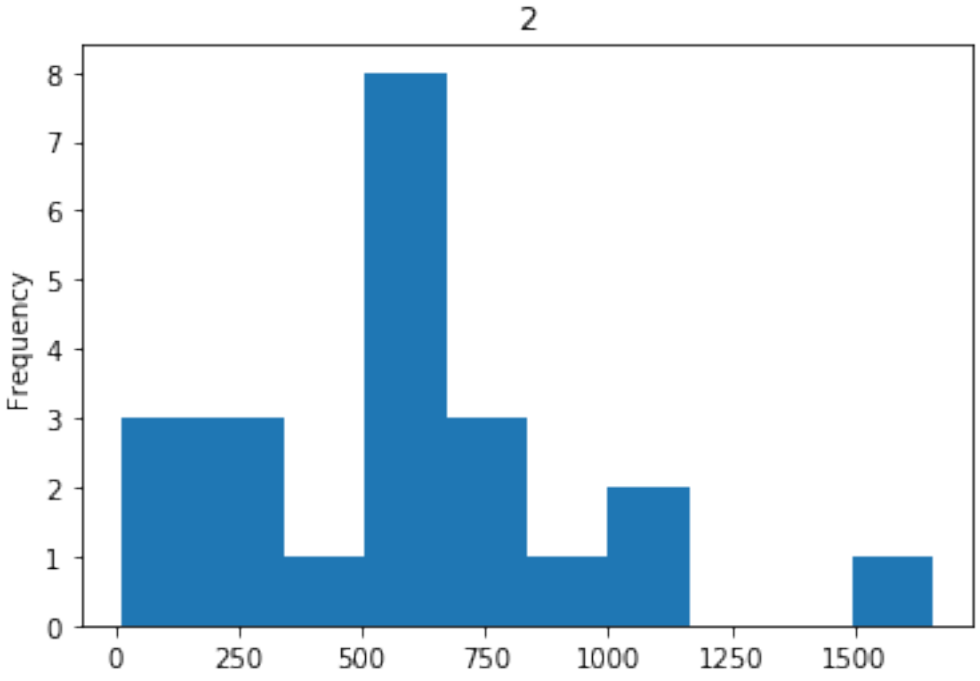

Income for 12/2018
Frequency
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
500
1000
1500
2000

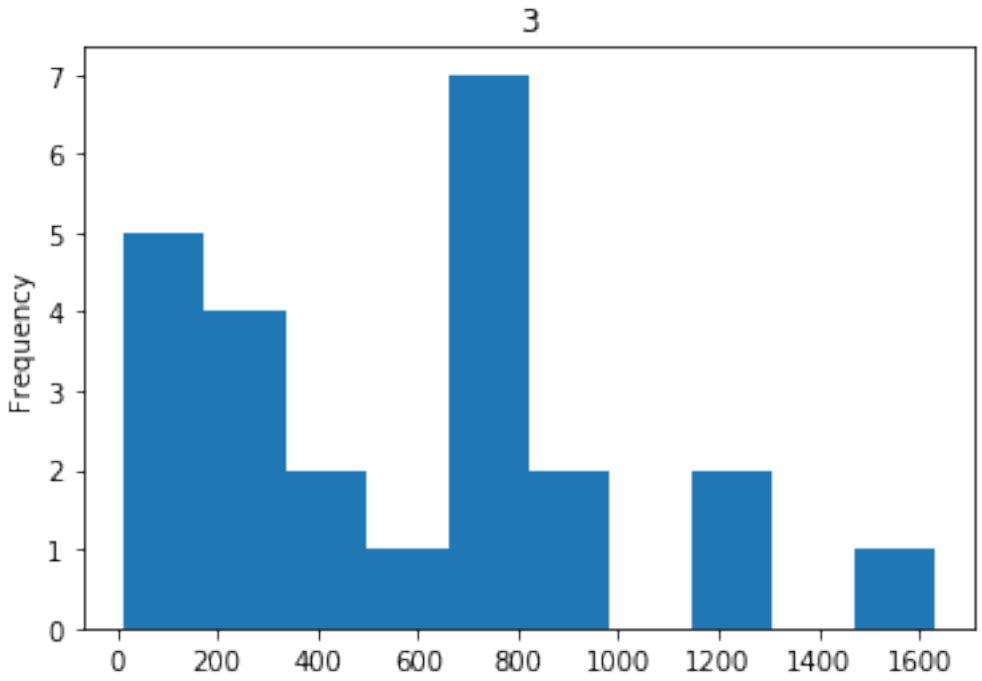


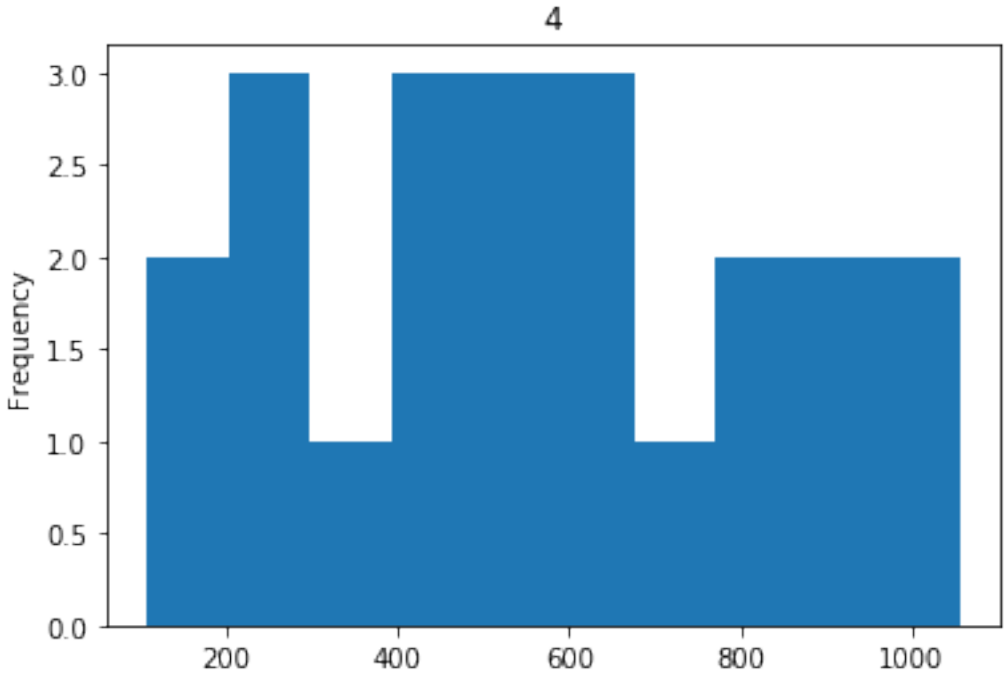

```
In [24]: for title, group in daily.groupby("dow"):
             plt.figure()
             group["Chiffre d'Affaires TTC"].plot(kind="hist", title=title)
```

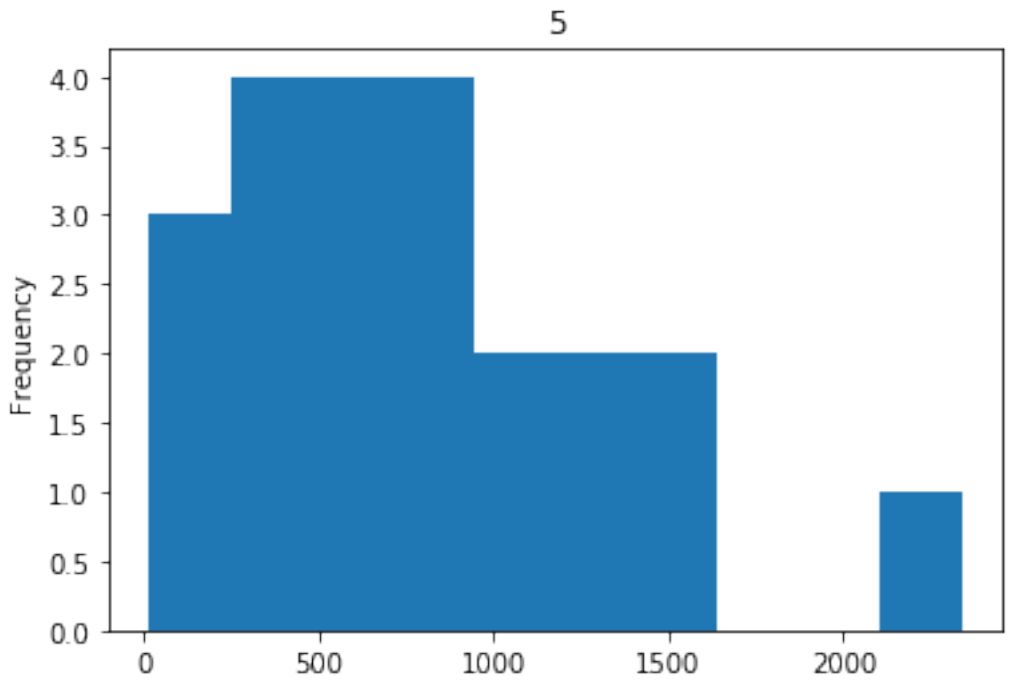


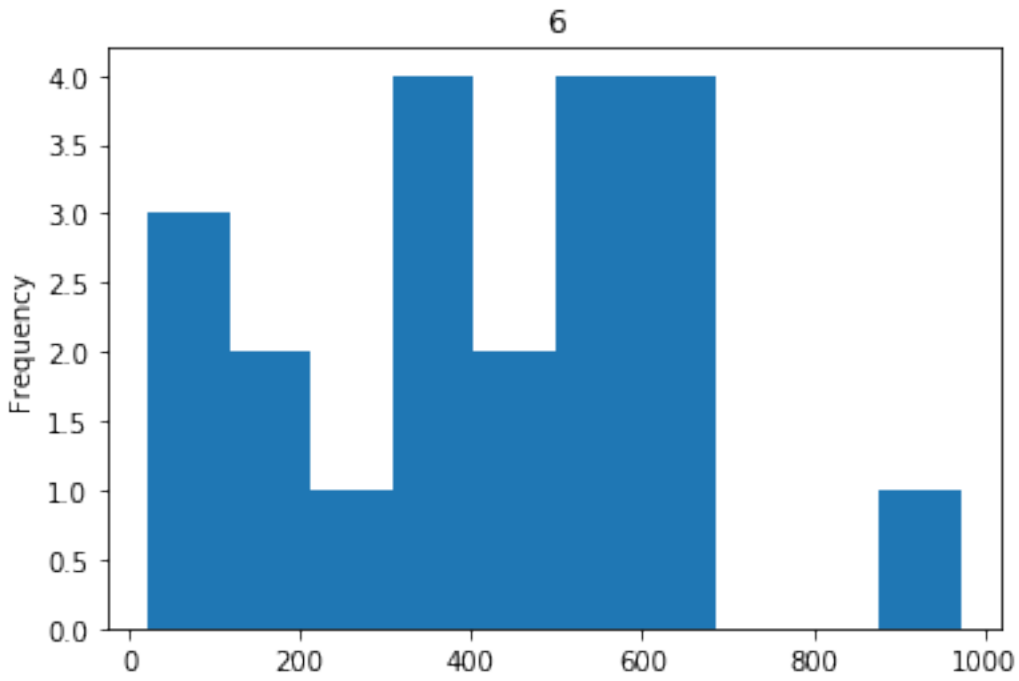

1
Frequency
0
1
2
3
4
5
6
0
100
200
300
400
500
600
700


2
Frequency
0
1
2
3
4
5
6
7
8
0
250
500
750
1000
1250
1500


3
Frequency
0
1
2
3
4
5
6
7
0
200
400
600
800
1000
1200
1400
1600

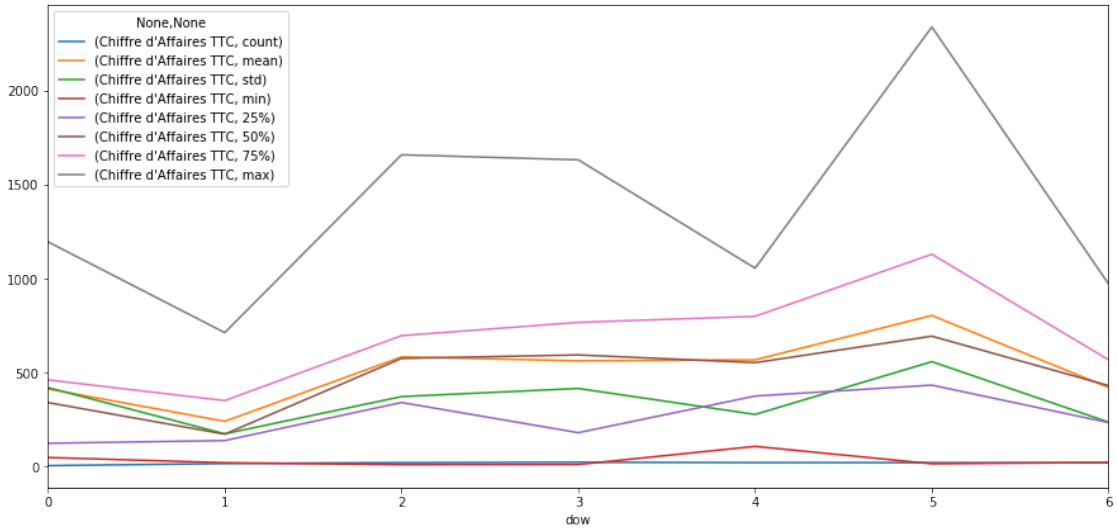

4
Frequency
0.0
0.5
1.0
1.5
2.0
2.5
3.0
200
400
600
800
1000





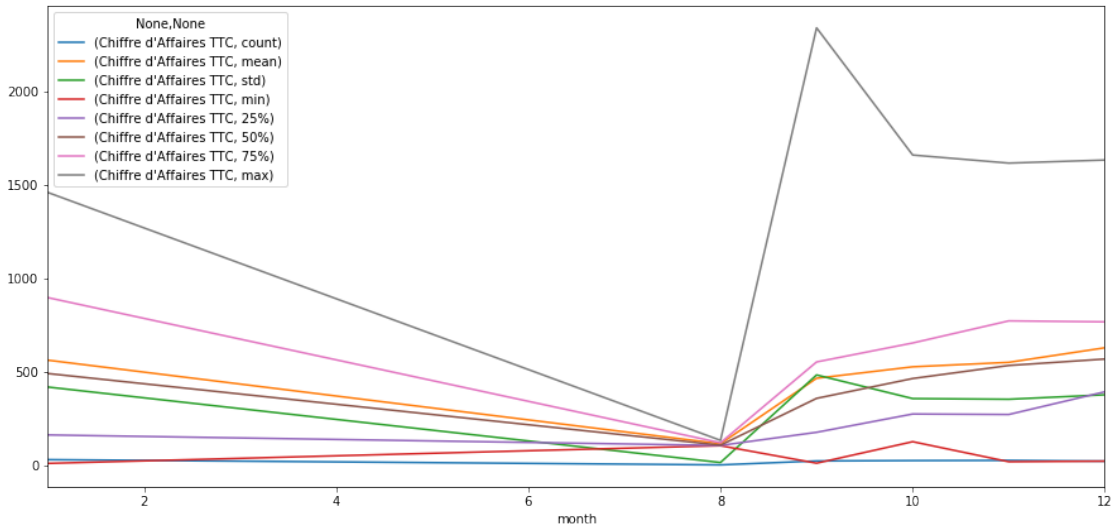

```
In [25]: t = daily.loc[:, ["dow", "Chiffre d'Affaires TTC"]]
         t = t.groupby("dow").describe()
         t.plot(figsize=(15,7))
```

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x7f07c6dbf898>



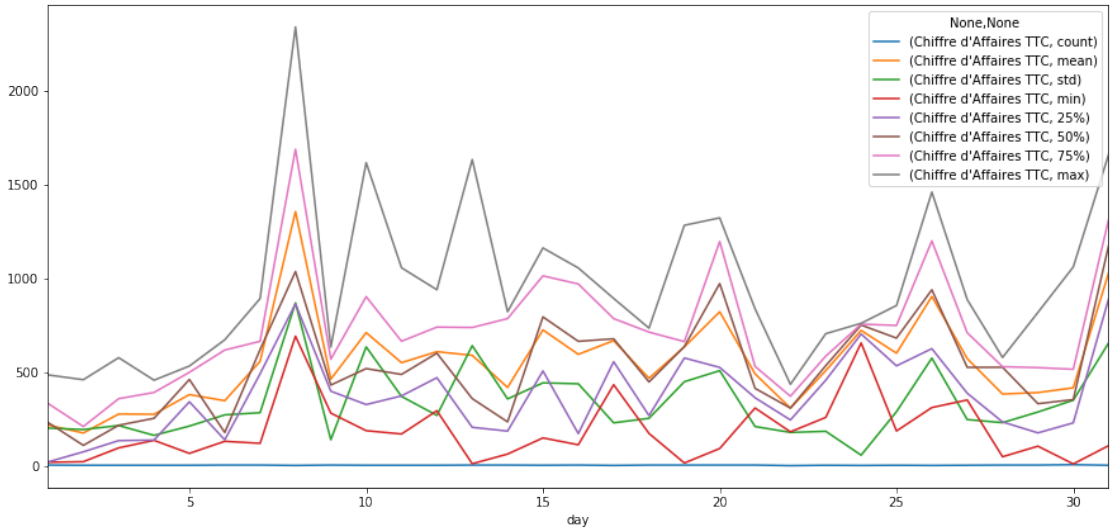
In [26]:
```
t = daily.loc[:, ["month", "Chiffre d'Affaires TTC"]]
t = t.groupby("month").describe()
t.plot(figsize=(15,7))
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f07c4e86d30>

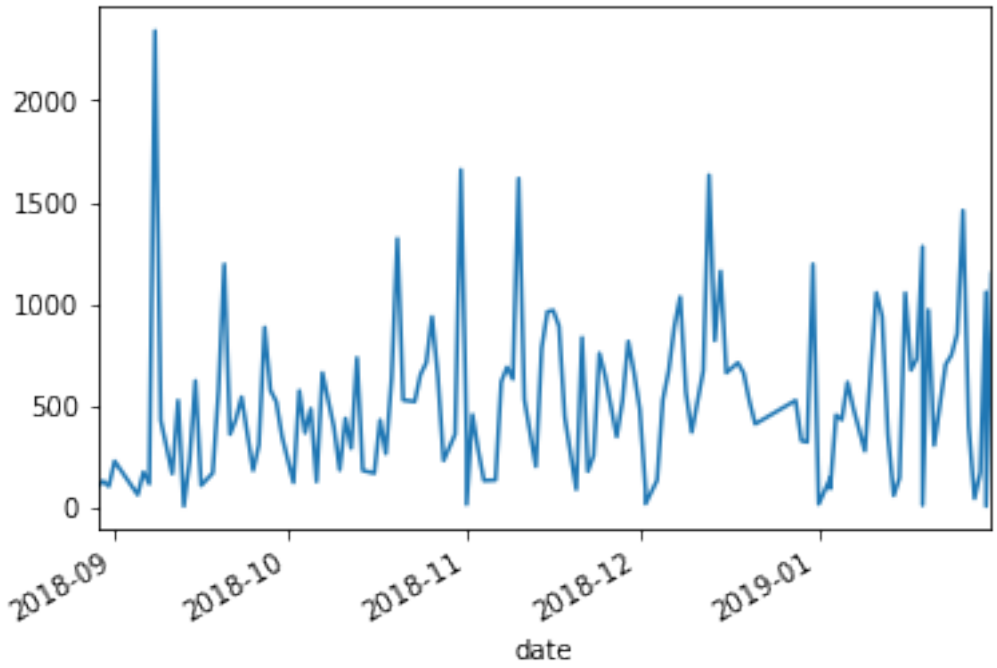


In [27]:
```
t = daily.loc[:, ["day", "Chiffre d'Affaires TTC"]]
t = t.groupby("day").describe()
t.plot(figsize=(15,7))
```

```
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7f07c4e00ac8>
```



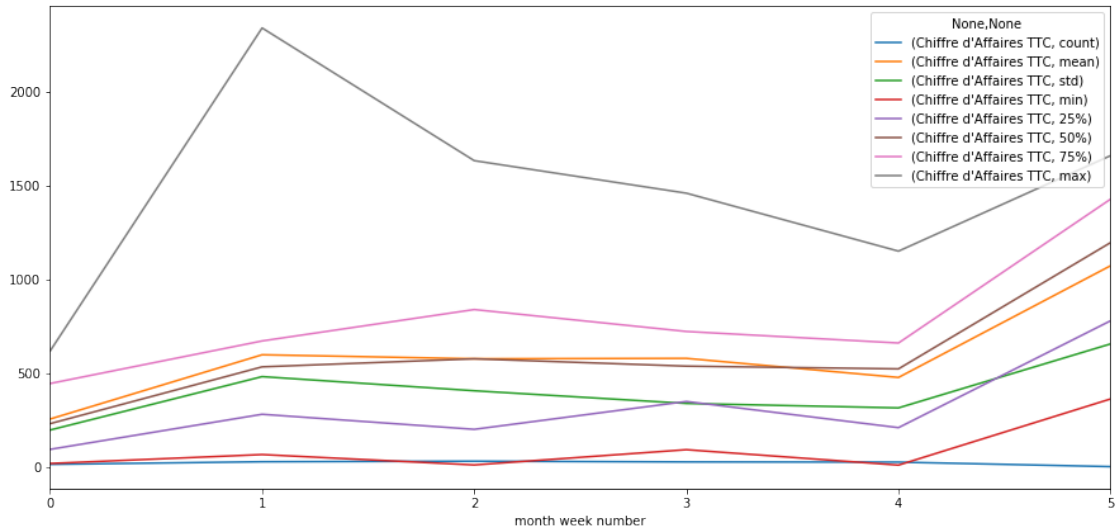

```
In [28]: daily["Chiffre d'Affaires TTC"].plot()

Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x7f07c6d03780>
```



This graph denote weekly seasonalities

In [29]:
```
t = daily.loc[:, ["month week number", "Chiffre d'Affaires TTC"]]
t = t.groupby("month week number").describe()
t.plot(figsize=(15,7))
```

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x7f07c4d16240>



In [ ]: