Final

January 14, 2020

1 Init project

```
In [1]: import herepy

        import folium
        import pygal
        import pandas as pd

        import numpy as np
        import os
        import re
        import time
        import json
        import datetime

        from pyspark.sql import SQLContext
        import matplotlib.pyplot as plt

        from sklearn.model_selection import train_test_split
        from shapely.geometry import Point
        from shapely.geometry.polygon import Polygon
```

```
In [2]: import findspark
        findspark.init()

        import pyspark
        import random
        import math

        # sc.stop()
        sc = pyspark.SparkContext(appName="Valeur Fonctiere")
```

2 Create Data File

No need to run it if final data file provided // Take few minutes

```
In [3]: class txt_reader_valeurs_foncieres():
    def __init__(self, full_path_name_txt):
        self.txt_paths=full_path_name_txt
        self.dataframes=self.fill_dataframes()

    def fill_dataframes(self):
        dataframes=list()
        for path in self.txt_paths:
            df=pd.read_csv(path,sep='|',low_memory=False)
            dataframes.append(df)
        return dataframes


    def get_columns(self):
        columns=None
        for df in self.dataframes:
            if columns is None:
                columns=df.columns
            else:
                if columns.all()!=df.columns.all():
                    print('TXT format columns not matching')

        return columns

    def combine_all_csv(self,columns_selected):
        df_joined=pd.DataFrame()
        for df in self.dataframes:
            df_joined=df_joined.append(df[columns_selected],ignore_index=True)
        df_joined.fillna(0, inplace=True)
        df_joined.to_csv("Valeurs foncières/Joined/valeurs_foncieres_joined.csv",index=

    def retrieve_df(self,item_num):
        return self.dataframes[item_num]
```

```
In [4]: txt_files=list()
        directory='Valeurs foncières/'
        for filename in os.listdir(directory):
            if filename.endswith(".txt"):
                txt_files.append(directory+ '/' +filename)

        reader_txt=txt_reader_valeurs_foncieres(txt_files)


        # We want to keep few columns for analysis:
        #     - Date mutation
        #     -'Valeur fonciere'
        #     -'No voie'
        #     -'Type de voie'
        #     -'Code voie'
        #     -'Voie'
        #     -'Code postal'
        #     -'Commune'
        #     -'Code departement'
        #     -'Code commune'
        #     -'Surface reelle bati'
        #     -'Nombre pieces principales'


        columns=['Date mutation','Valeur fonciere','No voie','Type de voie','Code voie','Voie'
            'Code postal','Commune','Code departement','Code commune','Surface reelle bati',
            'Nombre pieces principales']
        reader_txt.combine_all_csv(columns)
```

3 Read final data file as pd

```
In [3]: df=pd.read_csv('valeurs_foncieres_joined.csv',dtype=str,low_memory=False)

        # Select only ile de france
        df_IleDeFrance=df[df['Code postal'].str.contains('^75[0-9]{3}.',regex=True, na=False)]

In [4]: list_arrondissements=df_IleDeFrance['Code postal'].unique()

In [5]: df_IleDeFrance.head()

Out[5]:        Date mutation Valeur fonciere No voie Type de voie Code voie  \
        922095    16/01/2019       650000,00    25.0           AV      3520
        922162    24/01/2019       503029,00    35.0           AV      3520
        922283    24/01/2019       791600,00     3.0          RUE      1480
        922482    18/01/2019       607600,00    20.0          RUE      6660
        924012    15/03/2019       860000,00    30.0          RUE      6660

                                 Voie Code postal              Commune  \
        922095        FERDINAND BUISSON     75016.0  BOULOGNE-BILLANCOURT
        922162        FERDINAND BUISSON     75016.0  BOULOGNE-BILLANCOURT
        922283  DU CDT GUILBAUD A PARIS     75016.0  BOULOGNE-BILLANCOURT
        922482        NUNGESSER ET COLI     75016.0  BOULOGNE-BILLANCOURT
        924012        NUNGESSER ET COLI     75016.0  BOULOGNE-BILLANCOURT

               Code departement Code commune Surface reelle bati  \
        922095               92           12                89.0
        922162               92           12                55.0
        922283               92           12                90.0
        922482               92           12                66.0
        924012               92           12               105.0

               Nombre pieces principales
        922095                       3.0
```

```
922162                       3.0
922283                       4.0
922482                       3.0
924012                       3.0
```

```
In [6]: df_IleDeFrance[df_IleDeFrance['Voie']=='DUPHOT']
```

```
Out[6]:         Date mutation Valeur fonciere No voie Type de voie Code voie    Voie  \
        995783     04/01/2019      1196000,00    17.0          RUE      2999  DUPHOT
        996804     12/04/2019      2691110,00    15.0          RUE      2999  DUPHOT
        996808     03/04/2019       540500,00    10.0          RUE      2999  DUPHOT
        996809     03/04/2019       540500,00    10.0          RUE      2999  DUPHOT
        996810     03/04/2019       439500,00    10.0          RUE      2999  DUPHOT
        996811     03/04/2019       439500,00    10.0          RUE      2999  DUPHOT
        997406     21/05/2019       434000,00    19.0          RUE      2999  DUPHOT
        997407     21/05/2019       434000,00    19.0          RUE      2999  DUPHOT
        997408     21/05/2019       148000,00    19.0          RUE      2999  DUPHOT
        4016535    29/06/2018        50000,00    17.0          RUE      2999  DUPHOT
        4017503    24/09/2018      2200000,00    15.0          RUE      2999  DUPHOT
        4017504    24/09/2018      2200000,00    15.0          RUE      2999  DUPHOT
        4017505    24/09/2018      2200000,00    15.0          RUE      2999  DUPHOT
        4018099    26/10/2018     35000000,00     3.0          RUE      2999  DUPHOT
        4018114    16/10/2018       870000,00    10.0          RUE      2999  DUPHOT
        7388695    27/02/2017      2480000,00     1.0          RUE      2999  DUPHOT
        7388866    24/02/2017       565000,00    10.0          RUE      2999  DUPHOT

                Code postal   Commune Code departement Code commune  \
        995783      75001.0  PARIS 01               75          101
        996804      75001.0  PARIS 01               75          101
        996808      75001.0  PARIS 01               75          101
        996809      75001.0  PARIS 01               75          101
        996810      75001.0  PARIS 01               75          101
        996811      75001.0  PARIS 01               75          101
        997406      75001.0  PARIS 01               75          101
        997407      75001.0  PARIS 01               75          101
        997408      75001.0  PARIS 01               75          101
        4016535     75001.0  PARIS 01               75          101
        4017503     75001.0  PARIS 01               75          101
        4017504     75001.0  PARIS 01               75          101
        4017505     75001.0  PARIS 01               75          101
        4018099     75001.0  PARIS 01               75          101
        4018114     75001.0  PARIS 01               75          101
        7388695     75001.0  PARIS 01               75          101
        7388866     75001.0  PARIS 01               75          101

                Surface reelle bati Nombre pieces principales
        995783                112.0                       3.0
        996804                160.0                       0.0
```

```
996808                     43.0                       2.0
996809                     45.0                       2.0
996810                     37.0                       0.0
996811                     43.0                       2.0
997406                     92.0                       0.0
997407                      0.0                       0.0
997408                     91.0                       0.0
4016535                     0.0                       0.0
4017503                   160.0                       5.0
4017504                     0.0                       0.0
4017505                     0.0                       0.0
4018099                   147.0                       0.0
4018114                    39.0                       2.0
7388695                   119.0                       4.0
7388866                    48.0                       2.0
```

4 Read final data file as RDD // pyspark

```
In [7]: file_france = sc.textFile("valeurs_foncieres_joined.csv")
```

```
In [8]: future_pattern = re.compile("""([^,"]+|"[^"]+")(?=,|$)""")

        def parseCSV(line):
            return future_pattern.findall(line)
```

```
In [9]: data_France = file_france.map(parseCSV).filter(lambda x: x[0]!="Date mutation")
```

```
In [10]: iledefrance_pattern=re.compile("""^75[0-9]{3}..""")
         data_IleDeFrance=data_France.filter(lambda x: iledefrance_pattern.findall(x[6]))
```

```
In [11]: data_IleDeFrance_map=data_IleDeFrance.map(lambda x: ((x[2],x[3],x[5],x[7],x[6]),(x[0]
```

```
In [12]: data_IleDeFrance_map.cache().count()
```

```
Out[12]: 146878
```

```
In [13]: data_IleDeFrance_map.take(5)
```

```
Out[13]: [(('25.0', 'AV', 'FERDINAND BUISSON', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('16/01/2019', 650000.0, 89.0)),
          (('35.0', 'AV', 'FERDINAND BUISSON', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('24/01/2019', 503029.0, 55.0)),
          (('3.0', 'RUE', 'DU CDT GUILBAUD A PARIS', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('24/01/2019', 791600.0, 90.0)),
          (('20.0', 'RUE', 'NUNGESSER ET COLI', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('18/01/2019', 607600.0, 66.0)),
          (('30.0', 'RUE', 'NUNGESSER ET COLI', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('15/03/2019', 860000.0, 105.0))]
```

```
In [14]: list_arrondissements=df_IleDeFrance['Code postal'].unique()
```

```
In [15]: for arrondissement in list_arrondissements:
             data_arrondissement=data_IleDeFrance_map.filter(lambda x: arrondissement==x[0][4]
             print(arrondissement, ' : ',data_arrondissement.count())
```

```
75016.0  :  15455
75015.0  :  9254
75001.0  :  1730
75003.0  :  3226
75008.0  :  5611
75006.0  :  3569
75007.0  :  4554
75005.0  :  3239
75009.0  :  5966
75010.0  :  7957
75011.0  :  9275
75012.0  :  6953
75013.0  :  6936
75014.0  :  8904
75017.0  :  13760
75018.0  :  15296
75020.0  :  9201
75019.0  :  10522
75004.0  :  2843
75002.0  :  2627
```

5 Get long/lat from address with GeocoderApi

5.0.1 Test on single request

```
In [18]: # API init with Key
         geocoderApi = herepy.GeocoderApi('8UDt2_0vw7gJ2ffM2I2I17GIzMz5e8H980InYedeLcU')

         # s5tGP7hB1Md6T9HzvUm72xqF87Dur5vE119ARWilHPk    // guillaume_lehericy@hotmail.fr
         # 8UDt2_0vw7gJ2ffM2I2I17GIzMz5e8H980InYedeLcU
```

```
In [19]: arron='75016.0'
         # test_arron.unpersist()
         test_arron=data_IleDeFrance_map.filter(lambda x: arron==x[0][4])
         test_arron.cache().count()
```

```
Out[19]: 15455
```

```
In [20]: first_item=test_arron.take(1)
         first_item
```

```
Out[20]: [(('25.0', 'AV', 'FERDINAND BUISSON', 'BOULOGNE-BILLANCOURT', '75016.0'),
           ('16/01/2019', 650000.0, 89.0))]
```

5.0.2 If request correct

```
In [21]: response = geocoderApi.address_with_details(first_item[0][0][0],first_item[0][0][2] ,
         print(response)
```

```
{"Response": {"MetaInfo": {"Timestamp": "2020-01-10T13:04:40.340+0000"}, "View": [{"Result": [
```

5.0.3 If request wrong (no return)

```
In [22]: response = geocoderApi.address_with_details(first_item[0][0][0],first_item[0][0][2] ,
         print(response)
```

```
{"Response": {"MetaInfo": {"Timestamp": "2020-01-12T15:43:24.140+0000"}, "View": []}}
```

We had to identify both cases in order to deal with missing info

6 Retrieve all coord for items in Ile de France

```
In [23]: # Request api for each rows in RDD
         def geoLoca(x):
             return x[0][0],x[0][1],x[0][2],x[0][3],x[0][4], x[1][0], x[1][1], x[1][2], geocod

         # Retrieve only Lat and Long from API response
         def getOnlyLatLong(x):
             try:
                 lat=x[8]['Response']['View'][0]['Result'][0]['Location']['DisplayPosition']['
                 long=x[8]['Response']['View'][0]['Result'][0]['Location']['DisplayPosition'][
                 return x[0], x[1], x[2], x[3],x[4],x[5],x[6],x[7],lat,long
             except:
                 try:
                     return x[0], x[1], x[2], x[3],x[4],x[5],x[6],x[7],0,0
                 except:
                     return x

In [24]: # for each 'arrondissement', request API (high chances to crash if all requested at o
         for arrondissement in list_arrondissements:
             if not os.path.exists('arrondissement/' + arrondissement):
                 start=time.process_time()
                 print(arrondissement, ' , ', data_IleDeFrance_map.filter(lambda x: arrondisse
                 data_arrondissement=data_IleDeFrance_map.filter(lambda x: arrondissement==x[0
                 data_geo=data_arrondissement.map(lambda x: geoLoca(x)).map(lambda x: getOnlyL
                 data_geo.cache().collect()
                 data_geo.saveAsTextFile('arrondissement/' + arrondissement)
                 print(arrondissement, ' , ', data_geo.count(),' : ', time.process_time()-star
```

```
75011.0  ,  9275 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=20, tm_min=25, t
75011.0  ,  9275  :  0.1318980000000014
```

```
75012.0   ,  6953 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=20, tm_min=39, tm
75012.0   ,  6953  :  0.11816866100000034
75013.0   ,  6936 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=20, tm_min=50, tm
75013.0   ,  6936  :  0.10023059000000067
75014.0   ,  8904 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=21, tm_min=0, tm_
75014.0   ,  8904  :  0.14788003900000035
75017.0   ,  13760 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=21, tm_min=15, t
75017.0   ,  13760  :  0.2134044700000004
75018.0   ,  15296 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=21, tm_min=38, t
75018.0   ,  15296  :  0.20801423600000035
75020.0   ,  9201 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=21, tm_min=59, tm
75020.0   ,  9201  :  0.15402410899999985
75019.0   ,  10522 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=22, tm_min=9, tm
75019.0   ,  10522  :  0.17769891600000065
75004.0   ,  2843 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=22, tm_min=25, tm
75004.0   ,  2843  :  0.09365647799999977
75002.0   ,  2627 time.struct_time(tm_year=2020, tm_mon=1, tm_mday=12, tm_hour=22, tm_min=30, tm
75002.0   ,  2627  :  0.07415120799999997
```

7 Read saved files of all 'arrondissement'

```
In [20]: # retrieve all folders in 'arrondissement' and then read them
         list_arrondissement_path=list()
         for item in os.listdir('arrondissement'):
             list_arrondissement_path.append('arrondissement/'+item)
         data=sc.textFile(','.join(list_arrondissement_path))
```

```
In [21]: data=data.map(parseCSV)
```

```
In [22]: # read json polygons
         with open('quartier_paris.geojson') as json_file:
             quartier_info = json.load(json_file)

         # check if localisation belongs to one of the polygons
         def returnQuartier(x):
             list_return=list()
             for item in x:
                 list_return.append(item.replace('(', '').replace(')', '').replace("'","").repl
             for neighbor in quartier_info['features']:
                 polygon = Polygon(neighbor['geometry']['coordinates'][0]) # create polygon
                 point = Point(float(x[9].replace(' ','').replace(')','')),float(x[8].replace(
                 if polygon.contains(point):
                     list_return.append(neighbor['properties']['l_qu'])
                     return list_return
             list_return.append('NaN')
             return list_return
```

```
In [23]: with_neigbhor=data.map(lambda x: returnQuartier(x))
         with_neigbhor.take(2)

Out[23]: [['32.0',
           'RUE',
           'SAUSSURE',
           'PARIS17',
           '75017.0',
           '09/01/2017',
           '571000.0',
           '59.0',
           '48.88446',
           '2.31607',
           'Batignolles'],
          ['171.0',
           'RUE',
           'LEGENDRE',
           'PARIS17',
           '75017.0',
           '06/01/2017',
           '210000.0',
           '30.0',
           '48.89153',
           '2.32563',
           'Epinettes']]

In [24]: test=data.map(lambda x: returnQuartier(x)).map(lambda x: returnQuartier(x)).filter(la
         test.take(3)

Out[24]: [('1/2017', 9677.966101694916),
          ('1/2017', 7000.0),
          ('1/2017', 7272.727272727273)]

In [25]: sum_neigbhor=data.map(lambda x: returnQuartier(x)).filter(lambda x: float(x[6])!=0 an
         sum_neigbhor.collect()
         # Col1 = Quartier ; Col2 = Prix total desz ventes dans le quartier ; Col3= nombre de

Out[25]: [('Saint-Gervais', (1022177282.2, 39336.0, 758)),
          ('Batignolles', (4172587815.63, 157231.0, 2685)),
          ('Gare', (2530983561.4700003, 109291.0, 1380)),
          ('La Chapelle', (3267770059.73, 81298.0, 1227)),
          ('Pont-de-Flandre', (1210814087.4599998, 63560.0, 852)),
          ('Saint-Vincent-de-Paul', (677408867.99, 58752.0, 1001)),
          ('Auteuil', (3160521035.91, 252718.0, 3543)),
          ('Petit-Montrouge', (1330424707.3599997, 112280.0, 1943)),
          ('Europe', (8489775462.81, 159125.0, 1365)),
          ('Sainte-Marguerite', (1121258575.27, 67488.0, 1256)),
          ('Montparnasse', (2088615716.76, 67485.0, 1077)),
          ('Clignancourt', (4251286244.829999, 251958.0, 5052)),
          ('Père-Lachaise', (973676046.46, 86413.0, 1824)),
          ('Mail', (4077192480.8199997, 78994.0, 799)),
          ('Arts-et-Métiers', (591727724.05, 56305.0, 1073)),
          ('Grenelle', (2138371131.79, 113073.0, 1584)),
          ('Notre-Dame', (436979203.0, 19160.0, 262)),
          ('Madeleine', (3887107943.6, 81555.0, 728)),
          ('Place-Vendôme', (4156316156.12, 39770.0, 289)),
          ('Sorbonne', (868185387.54, 28758.0, 540)),
          ('Bel-Air', (1677691091.15, 63579.0, 1169)),
          ('Saint-Ambroise', (2959569375.0, 109671.0, 1717)),
          ('Saint-Victor', (758541729.1, 32763.0, 624)),
          ('Salpêtrière', (13000408715.730001, 35251.0, 752)),
          ('Notre-Dame-des-Champs', (1584606435.69, 92588.0, 1390)),
          ('Gros-Caillou', (10092712026.57, 105154.0, 1405)),
          ('Bonne-Nouvelle', (2203028212.04, 88263.0, 1424)),
          ('Javel', (1522680083.4999995, 79263.0, 1265)),
          ('Villette', (5385066645.330003, 112919.0, 2226)),
          ('Sainte-Avoie', (1129497692.62, 65316.0, 841)),
          ('Quinze-Vingts', (60517537506.13, 68714.0, 1075)),
          ("Saint-Thomas-d'Aquin", (2283651122.8, 64422.0, 698)),
          ('Saint-Germain-des-Prés', (544100908.62, 30599.0, 376)),
          ('Porte-Saint-Denis', (1276022819.3400002, 76487.0, 1160)),
          ('Saint-Georges', (16317705273.769999, 143089.0, 1643)),
          ('NaN', (470389571.94, 36775.0, 466)),
          ('Monnaie', (428690609.51, 30058.0, 580)),
          ('Archives', (872762827.3799999, 45205.0, 731)),
          ('Roquette', (47562024207.25999, 118089.0, 2158)),
          ("Chaussée-d'Antin", (10113948540.720001, 90028.0, 379)),
          ('Combat', (2374642469.47, 103403.0, 1964)),
          ('Saint-Merri', (2412211434.38, 30382.0, 530)),
          ('Croulebarbe', (581094006.1800001, 36418.0, 736)),
          ('Hôpital-Saint-Louis', (693512370.68, 59954.0, 1279)),
          ('Necker', (4713257042.02, 151373.0, 1399)),
          ('Rochechouart', (1066465334.1999999, 88713.0, 1557)),
          ('Porte-Saint-Martin', (1103712892.34, 99446.0, 1392)),
          ('Chaillot', (3338514343.76, 189362.0, 1498)),
          ('Saint-Lambert', (5282201383.21, 130499.0, 2501)),
          ('Faubourg-du-Roule', (35702266451.83, 170918.0, 1157)),
          ('Porte-Dauphine', (4562160269.620001, 173228.0, 1754)),
          ('Muette', (6568558381.48, 253526.0, 2515)),
          ('Jardin-des-Plantes', (1040987948.14, 43152.0, 850)),
          ('Champs-Elysées', (12463025008.01, 83827.0, 578)),
          ('Plaine de Monceaux', (25765547876.829998, 241273.0, 2122)),
          ('Charonne', (1092346394.43, 96080.0, 1961)),
          ('Saint-Fargeau', (581341004.14, 73052.0, 1424)),
          ('Odéon', (536566374.34000003, 31811.0, 480)),
          ('Epinettes', (3365159672.37, 124655.0, 2659)),
          ('Arsenal', (1533165137.4300003, 42279.0, 632)),
           ('Invalides', (4948450740.950001, 46545.0, 392)),
           ('Parc-de-Montsouris', (6066536566.889999, 55145.0, 704)),
           ('Folie-Méricourt', (28420205296.24, 89390.0, 1688)),
           ('Gaillon', (2764569629.0, 31885.0, 225)),
           ('Maison-Blanche', (2947950527.64, 145706.0, 2011)),
           ('Faubourg-Montmartre', (3070316722.2800007, 74030.0, 880)),
           ('Enfants-Rouges', (1265765266.45, 57426.0, 930)),
           ('Belleville', (933322692.26, 74239.0, 1512)),
           ('Vivienne', (2822761610.0, 67210.0, 265)),
           ('Halles', (4990056759.05, 63199.0, 1099)),
           ('Grandes-Carrières', (3066347543.87, 211323.0, 4294)),
           ('Ecole-Militaire', (1017883240.5799999, 53014.0, 566)),
           ('Amérique', (2144287655.57, 124016.0, 1783)),
           ('Palais-Royal', (1537140684.49, 26545.0, 381)),
           ('Bercy', (2702882725.12, 24232.0, 448)),
           ("Saint-Germain-l'Auxerrois", (133099279.77, 8015.0, 93)),
           ('Ternes', (15178646014.21, 248715.0, 2544)),
           ('Val-de-Grâce', (756650278.76, 47194.0, 765)),
           ('Picpus', (74016050000.17, 134411.0, 2222)),
           ('Plaisance', (10196990778.52, 127483.0, 2503)),
           ("Goutte-d'Or", (666858727.99, 66608.0, 1443))]
```

```
In [26]: mean_price_per_neigbors=sum_neigbhor.map(lambda x: (x[0],x[1][0]/x[1][2],x[1][0]/x[1]
         mean_price_per_neigbors.collect()
         # Col1 = Quartier ; Col2 = Prix moyen ; Col3 = Prix m2 : Col4= nombre de ventes
```

```
Out[26]: [('Saint-Gervais', 1348518.8419525067, 25985.796273134027, 758),
          ('Batignolles', 1554036.4304022347, 26537.94617874338, 2685),
          ('Gare', 1834046.059036232, 23158.206636136554, 1380),
          ('La Chapelle', 2663219.282583537, 40194.96248038082, 1227),
          ('Pont-de-Flandre', 1421143.2951408448, 19049.9384433606, 852),
          ('Saint-Vincent-de-Paul', 676732.1358541459, 11529.971200810185, 1001),
          ('Auteuil', 892046.5808382726, 12506.117632736885, 3543),
          ('Petit-Montrouge', 684727.0753268141, 11849.169107231917, 1943),
          ('Europe', 6219615.72367033, 53352.87015120189, 1365),
          ('Sainte-Marguerite', 892721.7955971337, 16614.19178624348, 1256),
          ('Montparnasse', 1939290.359108635, 30949.332692598357, 1077),
          ('Clignancourt', 841505.5908214566, 16872.99567717635, 5052),
          ('Père-Lachaise', 533813.6219627194, 11267.70331385324, 1824),
          ('Mail', 5102869.187509387, 51613.95144973035, 799),
          ('Arts-et-Métiers', 551470.3858807082, 10509.328195542135, 1073),
          ('Grenelle', 1349981.7751199494, 18911.421221600205, 1584),
          ('Notre-Dame', 1667859.5534351144, 22806.847755741128, 262),
          ('Madeleine', 5339433.988461538, 47662.411177732814, 728),
          ('Place-Vendôme', 14381716.80318339, 104508.82967362333, 289),
          ('Sorbonne', 1607750.7176666665, 30189.352094721467, 540),
          ('Bel-Air', 1435150.6340034218, 26387.50359631325, 1169),
          ('Saint-Ambroise', 1723686.2987769365, 26985.888475531363, 1717),
('Saint-Victor', 1215611.7453525641, 23152.3892531209, 624),
('Salpêtrière', 17287777.5475133, 368795.45873109985, 752),
('Notre-Dame-des-Champs', 1140004.629992806, 17114.598389532122, 1390),
('Gros-Caillou', 7183424.9299430605, 95980.29581917949, 1405),
('Bonne-Nouvelle', 1547070.3736235956, 24959.81568766074, 1424),
('Javel', 1203699.6707509877, 19210.477568348404, 1265),
('Villette', 2419167.4058086267, 47689.64164870396, 2226),
('Sainte-Avoie', 1343041.2516290129, 17292.817879539467, 841),
('Quinze-Vingts', 56295383.72663256, 880716.2660612102, 1075),
("Saint-Thomas-d'Aquin", 3271706.479656161, 35448.311489863714, 698),
('Saint-Germain-des-Prés', 1447076.8846276596, 17781.65654498513, 376),
('Porte-Saint-Denis', 1100019.6718448277, 16682.871851948697, 1160),
('Saint-Georges', 9931652.631631162, 114038.8518598215, 1643),
('NaN', 1009419.6822746781, 12791.014872603671, 466),
('Monnaie', 739121.7405344828, 14262.113564109388, 580),
('Archives', 1193929.9964158684, 19306.776404822474, 731),
('Roquette', 22039862.932001848, 402764.22196190996, 2158),
("Chaussée-d'Antin", 26685880.05467019, 112342.25508419605, 379),
('Combat', 1209084.7604226067, 22964.928188447142, 1964),
('Saint-Merri', 4551342.3290188685, 79396.071173063, 530),
('Croulebarbe', 789529.899701087, 15956.230605195235, 736),
('Hôpital-Saint-Louis', 542230.156903831, 11567.407857357306, 1279),
('Necker', 3369018.6147390995, 31136.708937657313, 1399),
('Rochechouart', 684948.8337829158, 12021.522597589981, 1557),
('Porte-Saint-Martin', 792897.1927729885, 11098.6152518955, 1392),
('Chaillot', 2228647.759519359, 17630.328913720812, 1498),
('Saint-Lambert', 2112035.7389884046, 40476.94911999326, 2501),
('Faubourg-du-Roule', 30857620.096655145, 208885.35117325268, 1157),
('Porte-Dauphine', 2601003.574469784, 26336.159683307553, 1754),
('Muette', 2611752.8355785287, 25908.815590826973, 2515),
('Jardin-des-Plantes', 1224691.7036941177, 24123.74740776789, 850),
('Champs-Elysées', 21562327.00347751, 148675.54616066423, 578),
('Plaine de Monceaux', 12142105.502747407, 106790.01743597501, 2122),
('Charonne', 557035.3872667007, 11369.133996981682, 1961),
('Saint-Fargeau', 408245.0871769663, 7957.906753271642, 1424),
('Odéon', 1117846.6132083335, 16867.32181761026, 480),
('Epinettes', 1265573.4006656636, 26995.78574762344, 2659),
('Arsenal', 2425894.2047943044, 36263.04163840205, 632),
('Invalides', 12623598.828954084, 106315.40962401978, 392),
('Parc-de-Montsouris', 8617239.441605113, 110010.63681004623, 704),
('Folie-Méricourt', 16836614.51199052, 317934.9512947757, 1688),
('Gaillon', 12286976.128888888, 86704.3948251529, 225),
('Maison-Blanche', 1465912.7437294878, 20232.183490316114, 2011),
('Faubourg-Montmartre', 3488996.2753181825, 41473.95275266784, 880),
('Enfants-Rouges', 1361037.9209139785, 22041.675659979803, 930),
('Belleville', 617276.9128703703, 12571.865087891809, 1512),
('Vivienne', 10651930.603773585, 41999.131230471656, 265),
('Halles', 4540543.001865332, 78957.84362173453, 1099),
('Grandes-Carrières', 714100.4992710759, 14510.240455937119, 4294),
('Ecole-Militaire', 1798380.2837102471, 19200.27239182103, 566),
('Amérique', 1202629.0833258552, 17290.411362808023, 1783),
('Palais-Royal', 4034489.985538058, 57906.97624750424, 381),
('Bercy', 6033220.368571429, 111541.87541762958, 448),
("Saint-Germain-l'Auxerrois", 1431175.0512903226, 16606.273208983155, 93),
('Ternes', 5966448.904956761, 61028.26936135737, 2544),
('Val-de-Grâce', 989085.3317124182, 16032.764308174768, 765),
('Picpus', 33310553.555432044, 550669.5880558139, 2222),
('Plaisance', 4073907.6222612867, 79987.06320466258, 2503),
("Goutte-d'Or", 462133.56063063064, 10011.69120811314, 1443)]
```

8 Analysis

```
In [17]: sqlContext = SQLContext(sc)
```

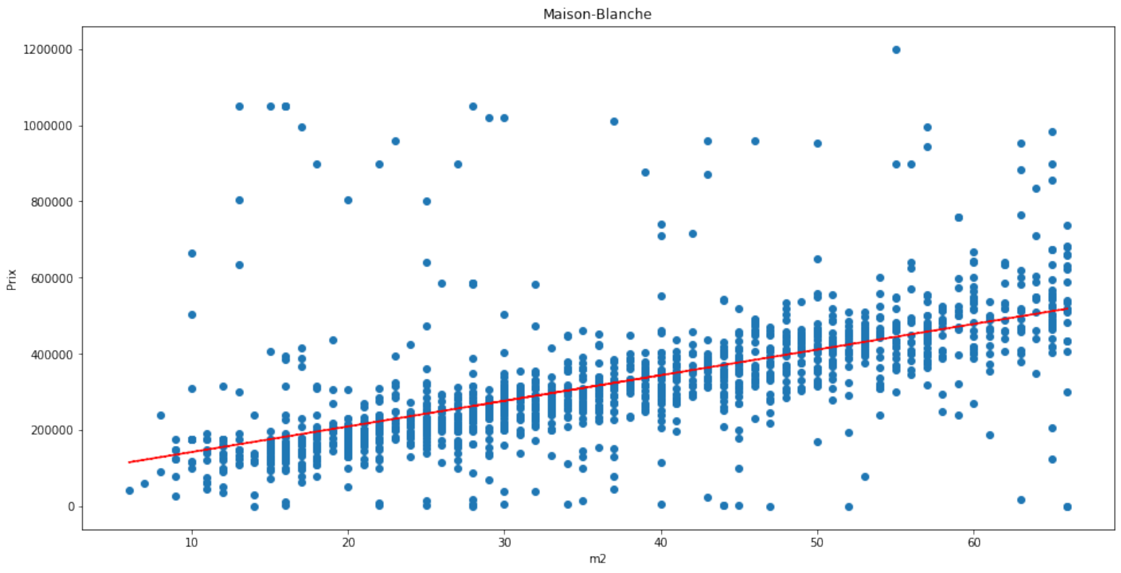
8.1 Prix d'un quartier en fonction de sa taille en m2

```
In [27]: rdd = sqlContext.createDataFrame(with_neigbhor)
         df=rdd.toPandas()

In [28]: df.columns=['Num','Type','Adr','Zone','Code','Date','Prix','m2','Lat','Long','Quartie

In [29]: df_pandas=df

In [30]: df_pandas['Prix']=df_pandas['Prix'].astype(float)
         df_pandas['m2']=df_pandas['m2'].astype(float)
         df_pandas['Date']=pd.to_datetime(df['Date'], format="%d/%m/%Y")
         df_pandas=df_pandas[df_pandas['Prix']!=0]
         df_pandas=df_pandas[df_pandas['m2']!=0]
         df_pandas["price_m2"]=df_pandas["Prix"]/df_pandas["m2"]

In [31]: df_pandas.head()

Out[31]:      Num Type             Adr     Zone     Code       Date      Prix    m2  \
         0   32.0  RUE        SAUSSURE  PARIS17  75017.0 2017-01-09  571000.0  59.0
         1  171.0  RUE        LEGENDRE  PARIS17  75017.0 2017-01-06  210000.0  30.0
         3   46.0  RUE       DESMOINES  PARIS17  75017.0 2017-01-06  160000.0  22.0
         6   37.0  RUE          POUCHET  PARIS17  75017.0 2017-01-02  179000.0  24.0
         7    6.0  RUE  DEODATDESEVERAC  PARIS17  75017.0 2017-01-11  225000.0  30.0

                 Lat     Long    Quartier     price_m2
         0  48.88446  2.31607  Batignolles  9677.966102
         1  48.89153  2.32563    Epinettes  7000.000000
         3   48.8896  2.32024    Epinettes  7272.727273
         6  48.89341  2.31984    Epinettes  7458.333333
         7  48.88593  2.31046  Batignolles  7500.000000

In [32]: df_quantile=df_pandas
In [33]: quantile_m2= df["m2"].quantile(0.80)
         quantile_Prix= df["Prix"].quantile(0.80)
         df_quantile=df_quantile[df_pandas["Prix"]<quantile_Prix]
         df_quantile=df_quantile[df_pandas["m2"]<quantile_m2]
```

```
/home/guillaume/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:4: UserWarning: Boo
  after removing the cwd from sys.path.
```

```
In [272]: quartier='Maison-Blanche'
          df_quartier=df_quantile[df_quantile['Quartier']==quartier]
```

```
In [293]: fig, ax = plt.subplots(figsize=(16,8))
          ax.scatter(x=df_quartier['m2'],y=df_quartier['Prix'])
          ax.set_xlabel('m2')
          ax.set_ylabel('Prix')

          z = np.polyfit(df_quartier['m2'], df_quartier['Prix'], 1)
          p = np.poly1d(z)

          ax.plot(df_quartier['m2'],p(df_quartier['m2']),"r--")

          plt.title(quartier)
          plt.show()
```



8.2 Evolution des prix a Paris? 2017-2018

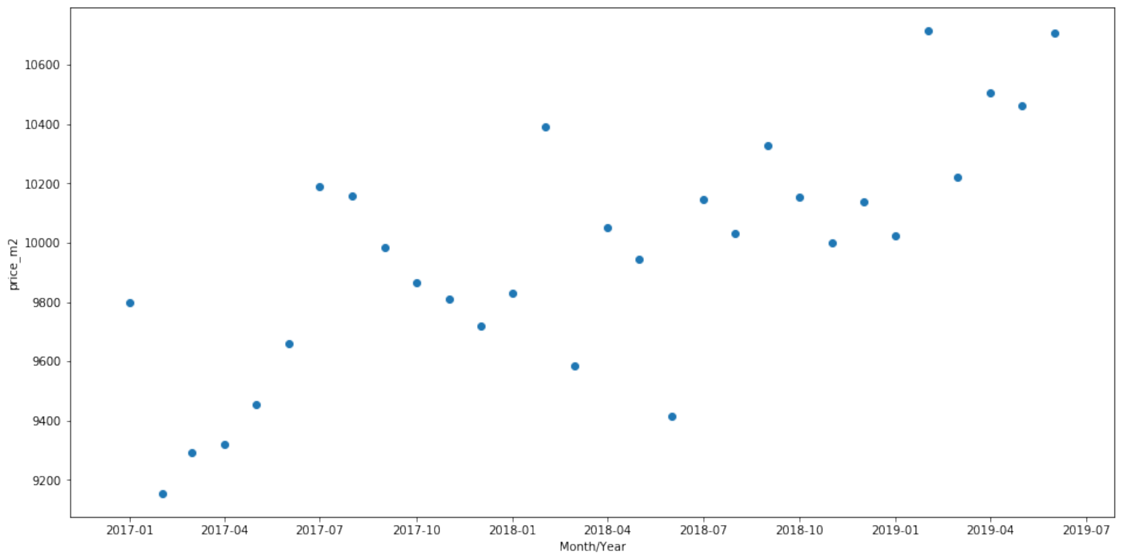
```
In [294]: df_quantile=df_quantile.sort_values(by=['Date'])
          df_quantile['Month']=pd.DatetimeIndex(df_quantile['Date']).month
```

```
        df_quantile['Year']=pd.DatetimeIndex(df_quantile['Date']).year
        df_quantile['Month/Year']=df_quantile['Month'].map(str) + '/' + df_quantile['Year'].m
        df_quantile['Month/Year']=pd.to_datetime(df_quantile['Month/Year'], format="%m/%Y")
```

```
In [263]: df_quantile=df_quantile.sort_values(by=['Month/Year'])
          df_quantile
```

```
Out[263]:          Num Type                 Adr     Zone     Code        Date      Prix  \
          27716   13.0  RUE        LOUISBRAILLE  PARIS12  75012.0  2017-01-02  193300.0
          27982   91.0  RUE           DEREUILLY  PARIS12  75012.0  2017-01-23  371000.0
          89574    5.0   PL           DESTERNES  PARIS17  75017.0  2017-01-23   60000.0
          55456    5.0  RUE              LITTRE  PARIS06  75006.0  2017-01-24  214000.0
          50476   19.0  RUE         DERICHEMONT  PARIS13  75013.0  2017-01-24   85000.0
          ...      ...  ...                 ...      ...      ...         ...       ...
          138884  58.0  RUE  DELACHAUSS.DANTIN  PARIS09  75009.0  2019-06-12       1.0
          138882  58.0  RUE  DELACHAUSS.DANTIN  PARIS09  75009.0  2019-06-12       1.0
          138881  58.0  RUE  DELACHAUSS.DANTIN  PARIS09  75009.0  2019-06-12       1.0
          89063   39.0  RUE             LAUGIER  PARIS17  75017.0  2019-06-12  287500.0
          103705  70.0  RUE       DEMENILMONTANT  PARIS20  75020.0  2019-06-29  208200.0

                    m2       Lat     Long             Quartier      price_m2  Month  \
          27716   28.0  48.83979  2.40189              Bel-Air   6903.571429      1
          27982   30.0  48.84447  2.39012               Picpus  12366.666667      1
          89574    9.0   48.8784  2.29765               Ternes   6666.666667      1
          55456   20.0  48.84576  2.32391  Notre-Dame-des-Champs  10700.000000      1
          50476   15.0  48.82783   2.3694                 Gare   5666.666667      1
          ...      ...       ...      ...                  ...           ...    ...
          138884  10.0  48.87463  2.33268     Chaussée-d'Antin      0.100000      6
          138882  10.0  48.87463  2.33268     Chaussée-d'Antin      0.100000      6
          138881  10.0  48.87463  2.33268     Chaussée-d'Antin      0.100000      6
          89063   29.0  48.88188  2.29474               Ternes   9913.793103      6
          103705  25.0  48.86403   2.3866        Père-Lachaise   8328.000000      6

                  Year Month/Year
          27716   2017 2017-01-01
          27982   2017 2017-01-01
          89574   2017 2017-01-01
          55456   2017 2017-01-01
          50476   2017 2017-01-01
          ...      ...        ...
          138884  2019 2019-06-01
          138882  2019 2019-06-01
          138881  2019 2019-06-01
          89063   2019 2019-06-01
          103705  2019 2019-06-01

          [27215 rows x 15 columns]
```

```
In [264]: df_final=df_quantile.groupby(['Month/Year']).mean().reset_index()
In [265]: fig, ax = plt.subplots(figsize=(16,8))
          ax.scatter(x=df_final['Month/Year'],y=df_final['price_m2'])
          ax.set_xlabel('Month/Year')
          ax.set_ylabel('price_m2')

          plt.show()
```



8.2.1 On remarque une tendance haussière sur les prix en fonction du temps

9 Folium map

Colore les quartiers en fonction du prix moyen au m2

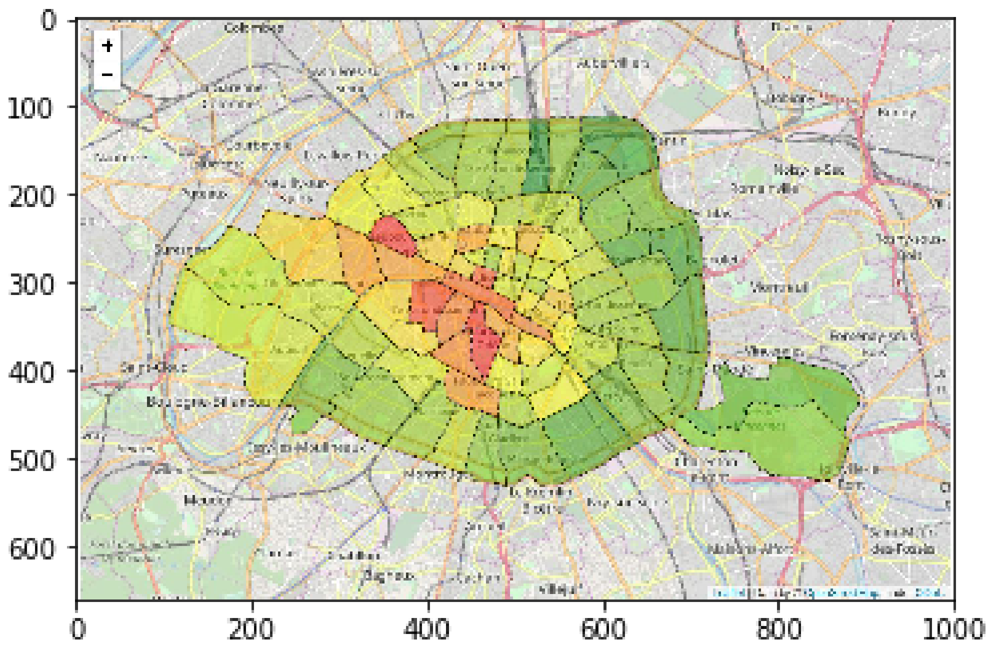
```
In [39]: import folium #mappingas
         import branca.colormap as cm
```

```
In [35]: df_mean_quartier=df_quantile.groupby(['Quartier']).mean().reset_index()
```

```
In [37]: print(df_mean_quartier[df_mean_quartier['Quartier']=='Sorbonne']['price_m2'].min())
         print(df_mean_quartier[df_mean_quartier['Quartier']=='Sorbonne']['price_m2'].max())
```

```
13837.179534327366
13837.179534327366
```

```
In [40]: coords = (48.864716,2.349014)
         map = folium.Map(location=coords, tiles='OpenStreetMap', zoom_start=12)

         linear = cm.LinearColormap(
             ['green','yellow', 'red'],
             vmin=df_mean_quartier['price_m2'].min(), vmax=df_mean_quartier['price_m2'].max()
         )

         #style function

         def style_function(feature):
             if df_mean_quartier['Quartier'].isin([feature['properties']['l_qu']]).any().any()
                 value=df_mean_quartier.loc[df_mean_quartier['Quartier'] == feature['propertie
                 return {
                     'fillColor': linear(value),
                     'fillOpacity': 0.5,
                     'color': 'black',
                     'weight': 1,
                     'dashArray': '5, 5'
                 }
             return {
                 'fillColor': linear(0),
                 'color': 'black',
                 'weight': 2,
                 'dashArray': '5, 5'
                 }
         folium.GeoJson(
             data=quartier_info,
             name=quartier_info,
             style_function= style_function
         ).add_to(map)


         map.save(outfile='map.html')

         from IPython.display import IFrame
         IFrame(src='map.html', width=900, height=600)

Out[40]: <IPython.lib.display.IFrame at 0x7fc99547bc18>

In [43]: import matplotlib.pyplot as plt
         import matplotlib.image as mpimg
         img=mpimg.imread('Map.jpg')
         imgplot = plt.imshow(img)
         plt.show()
```



0
100
200
300
400
500
600
0
200
400
600
800
1000