Volume Demo CB

May 1, 2020

1 Introduction

Business problem: Pitney Bowes FDR business allows retailers to easily facilitate the return of merchandise (other services are Fullfilment and Delivery). Consumers can drop off the return parcels at any USPS office, and the parcel gets transported to the closest FDR hub to be inducted into the FDR network.

Once inducted into our network, the parcel get transported through the FDR network to the client's warehouse.

The volume of parcels that is being delivered to the clients' warehouse is driven by external factors like seasonal sales, but also by warehouse schedules, holidays etc.

Being able to give our clients reliable forecasts on what parcel volumes they can expect to arrive at their warehouse is very important for them, since the need these forecasts to staff the warehouse accordingly and efficiently manage the available warehouse space.

In this challenge, you are given 3 month of historic data for packages that have been delivered to a single client warehouse. The aim is to use this historic data to predict the parcel volumes to arrive at the client's facility over the next 5 days (i.e., 5 numbers for Monday through Friday).

Data: The data is aggregated by delivery or indiction date. The delivery column shows the total number of parcels that have neem delivered to the client on any given day. The induction columns give you the induction volume per day accross our FDR facilities. The time it takes a parcel to travel from the induction facility to the client facility depends largely on the distance between these facilities.

Output Format: TBD, but all we need are your 5 parcel-volume predictions for June, 3rd - June, 7th 2019. To evaluate your preditcions, we use the Mean Absolute Percentage Error.

$MAPE = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i-\hat{y}_i|}{y_i}$

[263]:
```
import os
import datetime as dt
import sys
module_path = os.path.abspath(os.path.join('work/shared'))
if module_path not in sys.path:
    sys.path.append(module_path)
```

[264]:
```
import warnings                                  # `do not disturbe` mode
warnings.filterwarnings('ignore')
import numpy as np                               # vectors and matrices
import pandas as pd                              # tables and data manipulations
import matplotlib.pyplot as plt                  # plots

from dateutil.relativedelta import relativedelta # working with dates with style
from scipy.optimize import minimize              # for function minimization

import scipy.stats as scs

from itertools import product                    # some useful functions

%matplotlib inline
```

[324]:
```
#Reading the provided csv file into a pandas Dataframe
Delivery = pd.read_csv('~/work/shared/Delivery_Volume.csv',␣
↪index_col=['DELIVERY_DATE'], parse_dates=['DELIVERY_DATE'])
Delivery.sort_index(inplace=True)
Delivery=Delivery.fillna(0)
```

[325]:
```
Delivery['Total_Inductioned'] = Delivery.iloc[:, 1:12].sum(axis=1)
Delivery['Weekday'] = Delivery.index.map(lambda x: x.weekday())
Delivery['Open'] = Delivery.index.map(lambda x: int(x.weekday() <= 4))
Delivery.loc[Delivery.index == '2019-5-27', 'Open'] = 0  # Could generalize for␣
↪other holidays
Delivery['Midweek Prox'] = Delivery.Weekday.map( lambda x: (abs(x  - 2) + 1) )␣
↪* Delivery.Open
Delivery['MidShift'] = Delivery['Midweek Prox'].shift(1)
Delivery = Delivery.assign(Midweek = lambda df: ( df['MidShift']==0 & df.Open)␣
↪+ (df.Open *df['Midweek Prox']))
Delivery.Midweek = Delivery.Midweek * Delivery.Open
Delivery['Mon'] = Delivery.index.map(lambda x: int(x.weekday() == 0))
Delivery['Tue'] = Delivery.index.map(lambda x: int(x.weekday() == 1))
Delivery['Wed'] = Delivery.index.map(lambda x: int(x.weekday() == 2))
Delivery['Thu'] = Delivery.index.map(lambda x: int(x.weekday() == 3))
Delivery['Fri'] = Delivery.index.map(lambda x: int(x.weekday() == 4))
Delivery['Weekend'] = Delivery.index.map(lambda x: int(x.weekday() > 4))
Delivery.loc[Delivery.index == '2019-3-18', 'Midweek'] = 4
```

[326]:
```
Delivery.head(20)
```

[326]:
```
               DELIVERED_VOLUME  Facility_A  Facility_B  Facility_C  \
DELIVERY_DATE
2019-03-13                  0.0         0.0         0.0         0.0
2019-03-14                  0.0         0.0         0.0         0.0
2019-03-15                  0.0         0.0         1.0         0.0
```

```
2019-03-18           0.0       0.0       0.0       0.0
2019-03-19           0.0      13.0       0.0       0.0
2019-03-20           0.0     275.0      32.0      51.0
2019-03-21           0.0     213.0      39.0     151.0
2019-03-22         840.0     143.0      19.0     126.0
2019-03-23           0.0       0.0       3.0      95.0
2019-03-24           0.0       0.0       0.0       0.0
2019-03-25        1694.0     213.0       0.0      45.0
2019-03-26        1183.0     202.0       0.0     208.0
2019-03-27         905.0     172.0       0.0     209.0
2019-03-28         682.0     226.0      25.0     184.0
2019-03-29        1463.0     158.0      45.0     175.0
2019-03-30           0.0       0.0       5.0      95.0
2019-03-31           0.0       0.0       3.0       0.0
2019-04-01        1860.0     237.0      42.0      52.0
2019-04-02         911.0     290.0      36.0     182.0
2019-04-03         780.0     285.0      28.0     163.0

               Facility_D  Facility_E  Facility_F  Facility_G  Facility_H  \
DELIVERY_DATE
2019-03-13            1.0         0.0         0.0         0.0         0.0
2019-03-14            8.0         0.0         0.0         0.0         0.0
2019-03-15            6.0         0.0         0.0         0.0         0.0
2019-03-18           55.0         3.0         0.0        45.0        86.0
2019-03-19            3.0         9.0        80.0       243.0        80.0
2019-03-20           37.0       322.0        60.0       264.0       111.0
2019-03-21           89.0       265.0        57.0       309.0        78.0
2019-03-22           16.0       226.0        45.0       214.0        81.0
2019-03-23            0.0       131.0        38.0         0.0         0.0
2019-03-24           21.0         0.0         0.0         0.0         0.0
2019-03-25           27.0       146.0         0.0       366.0       126.0
2019-03-26           23.0       286.0        87.0       180.0        88.0
2019-03-27           15.0       348.0        70.0       413.0       125.0
2019-03-28           27.0       286.0        54.0       290.0        89.0
2019-03-29           19.0       123.0        53.0       286.0        96.0
2019-03-30            0.0       218.0        53.0         0.0        54.0
2019-03-31           19.0         0.0         0.0         0.0         0.0
2019-04-01           29.0       216.0         0.0       438.0        67.0
2019-04-02           38.0       316.0        87.0       262.0        89.0
2019-04-03           10.0       319.0        58.0       354.0       119.0

               Facility_I  …  Open  Midweek Prox  MidShift  Midweek  Mon  \
DELIVERY_DATE              …
2019-03-13            0.0  …     1             1       NaN        1    0
2019-03-14            0.0  …     1             2       1.0        2    0
2019-03-15            0.0  …     1             3       2.0        3    0
2019-03-18            0.0  …     1             3       3.0        4    1
```

```
2019-03-19           0.0  …     1           2       3.0         2    0
2019-03-20           0.0  …     1           1       2.0         1    0
2019-03-21           0.0  …     1           2       1.0         2    0
2019-03-22           0.0  …     1           3       2.0         3    0
2019-03-23           0.0  …     0           0       3.0         0    0
2019-03-24           0.0  …     0           0       0.0         0    0
2019-03-25           0.0  …     1           3       0.0         4    1
2019-03-26           0.0  …     1           2       3.0         2    0
2019-03-27           0.0  …     1           1       2.0         1    0
2019-03-28           0.0  …     1           2       1.0         2    0
2019-03-29           0.0  …     1           3       2.0         3    0
2019-03-30           0.0  …     0           0       3.0         0    0
2019-03-31           0.0  …     0           0       0.0         0    0
2019-04-01           0.0  …     1           3       0.0         4    1
2019-04-02           8.0  …     1           2       3.0         2    0
2019-04-03           9.0  …     1           1       2.0         1    0

              Tue  Wed  Thu  Fri  Weekend
DELIVERY_DATE
2019-03-13      0    1    0    0        0
2019-03-14      0    0    1    0        0
2019-03-15      0    0    0    1        0
2019-03-18      0    0    0    0        0
2019-03-19      1    0    0    0        0
2019-03-20      0    1    0    0        0
2019-03-21      0    0    1    0        0
2019-03-22      0    0    0    1        0
2019-03-23      0    0    0    0        1
2019-03-24      0    0    0    0        1
2019-03-25      0    0    0    0        0
2019-03-26      1    0    0    0        0
2019-03-27      0    1    0    0        0
2019-03-28      0    0    1    0        0
2019-03-29      0    0    0    1        0
2019-03-30      0    0    0    0        1
2019-03-31      0    0    0    0        1
2019-04-01      0    0    0    0        0
2019-04-02      1    0    0    0        0
2019-04-03      0    1    0    0        0

[20 rows x 24 columns]
```

[355]:
```
weights = Delivery.iloc[:,1:12].mean(axis=0) / sum(Delivery.iloc[:,1:12].
 ↪mean(axis=0))
weighted_facilities = weights * Delivery.iloc[:, 1:12]
new_delivery = Delivery.join(weighted_facilities, rsuffix='_W')
new_delivery['Total_Inductioned_W'] = new_delivery.iloc[:, -11:].sum(axis=1)
```

```
new_delivery['Total_Inductioned_W x Days'] = new_delivery.iloc[:, -11:].
↪sum(axis=1) * new_delivery['Midweek']
new_delivery = new_delivery.join(new_delivery.iloc[:, -13:-2].shift(1),␣
↪rsuffix='_t-1').fillna(0)
new_delivery['Midweek1'] = new_delivery['Midweek']
new_delivery['Midweek2'] = new_delivery['Midweek'] ** 2
new_delivery['Midweek3'] = new_delivery['Midweek'] ** 3
new_delivery['Midweek4'] = new_delivery['Midweek'] ** 4
#new_delivery['Total_Inductioned_W x Day_num'] = new_deliver
#new_delivery['Total_Inductioned_W x Day_num'] =␣
↪new_delivery['Total_Inductioned_W'] * Delivery.index.map(lambda x: x.day)
new_delivery.columns
```

[355]:
```
Index(['DELIVERED_VOLUME', 'Facility_A', 'Facility_B', 'Facility_C',
       'Facility_D', 'Facility_E', 'Facility_F', 'Facility_G', 'Facility_H',
       'Facility_I', 'Facility_J', 'Facility_K', 'Total_Inductioned',
       'Weekday', 'Open', 'Midweek Prox', 'MidShift', 'Midweek', 'Mon', 'Tue',
       'Wed', 'Thu', 'Fri', 'Weekend', 'Facility_A_W', 'Facility_B_W',
       'Facility_C_W', 'Facility_D_W', 'Facility_E_W', 'Facility_F_W',
       'Facility_G_W', 'Facility_H_W', 'Facility_I_W', 'Facility_J_W',
       'Facility_K_W', 'Total_Inductioned_W', 'Total_Inductioned_W x Days',
       'Facility_A_W_t-1', 'Facility_B_W_t-1', 'Facility_C_W_t-1',
       'Facility_D_W_t-1', 'Facility_E_W_t-1', 'Facility_F_W_t-1',
       'Facility_G_W_t-1', 'Facility_H_W_t-1', 'Facility_I_W_t-1',
       'Facility_J_W_t-1', 'Facility_K_W_t-1', 'Midweek1', 'Midweek2',
       'Midweek3', 'Midweek4'],
      dtype='object')
```

[377]:
```
new_delivery.iloc[:,-28:]
```

[377]:
```
               Facility_A_W  Facility_B_W  Facility_C_W  Facility_D_W  \
DELIVERY_DATE
2019-03-13         0.000000      0.000000      0.000000      0.027245
2019-03-14         0.000000      0.000000      0.000000      0.217962
2019-03-15         0.000000      0.020452      0.000000      0.163471
2019-03-18         0.000000      0.000000      0.000000      1.498486
2019-03-19         2.414647      0.000000      0.000000      0.081736
…                       …            …            …            …
2019-05-29        52.750757      0.736260     27.043703      1.253280
2019-05-30        49.035915      0.429485     23.499140      0.245207
2019-05-31        38.634357      1.063487     21.398658      1.307770
2019-06-01         0.000000      0.000000      0.000000      0.000000
2019-06-02         0.000000      0.000000      0.000000      0.572149

               Facility_E_W  Facility_F_W  Facility_G_W  Facility_H_W  \
DELIVERY_DATE
2019-03-13         0.000000      0.000000      0.000000      0.000000
```

```
2019-03-14        0.000000    0.000000     0.000000    0.000000
2019-03-15        0.000000    0.000000     0.000000    0.000000
2019-03-18        0.690340    0.000000    11.590725    6.564802
2019-03-19        2.071019    4.073848    62.589915    6.106792
…                      …          …           …          …
2019-05-29       75.477154    4.022925   106.634670    5.267108
2019-05-30       87.903270    3.615540    80.104788    7.938830
2019-05-31       71.795341    4.022925   103.286238    8.320504
2019-06-01       40.269823    2.597078     0.000000    0.534344
2019-06-02        0.000000    0.000000     0.000000    0.000000

              Facility_I_W  Facility_J_W  …  Facility_F_W_t-1  \
DELIVERY_DATE                             …
2019-03-13        0.000000      0.000000  …          0.000000
2019-03-14        0.000000      0.000000  …          0.000000
2019-03-15        0.000000      0.000000  …          0.000000
2019-03-18        0.000000      0.138941  …          0.000000
2019-03-19        0.000000      0.052103  …          0.000000
…                      …          …  …                …
2019-05-29        0.128537      0.086838  …          0.000000
2019-05-30        0.082631      0.104205  …          4.022925
2019-05-31        0.073450      0.069470  …          3.615540
2019-06-01        0.000000      0.000000  …          4.022925
2019-06-02        0.000000      0.000000  …          2.597078

              Facility_G_W_t-1  Facility_H_W_t-1  Facility_I_W_t-1  \
DELIVERY_DATE
2019-03-13            0.000000          0.000000          0.000000
2019-03-14            0.000000          0.000000          0.000000
2019-03-15            0.000000          0.000000          0.000000
2019-03-18            0.000000          0.000000          0.000000
2019-03-19           11.590725          6.564802          0.000000
…                          …               …               …
2019-05-29          113.589105          3.053396          0.100993
2019-05-30          106.634670          5.267108          0.128537
2019-05-31           80.104788          7.938830          0.082631
2019-06-01          103.286238          8.320504          0.073450
2019-06-02            0.000000          0.534344          0.000000

              Facility_J_W_t-1  Facility_K_W_t-1  Midweek1  Midweek2  \
DELIVERY_DATE
2019-03-13            0.000000          0.000000         1         1
2019-03-14            0.000000          0.000000         2         4
2019-03-15            0.000000          0.000000         3         9
2019-03-18            0.000000          0.000000         4        16
2019-03-19            0.138941          0.000000         2         4
…                          …               …       …       …
```

```
2019-05-29             0.138941          0.017311         1          1
2019-05-30             0.086838          0.019784         2          4
2019-05-31             0.104205          0.019784         3          9
2019-06-01             0.069470          0.024730         0          0
2019-06-02             0.000000          0.000000         0          0

               Midweek3  Midweek4
DELIVERY_DATE
2019-03-13            1         1
2019-03-14            8        16
2019-03-15           27        81
2019-03-18           64       256
2019-03-19            8        16
…                   …        …
2019-05-29            1         1
2019-05-30            8        16
2019-05-31           27        81
2019-06-01            0         0
2019-06-02            0         0

[80 rows x 28 columns]
```

[358]:

```
new_delivery.columns
```

[358]:

```
Index(['DELIVERED_VOLUME', 'Facility_A', 'Facility_B', 'Facility_C',
       'Facility_D', 'Facility_E', 'Facility_F', 'Facility_G', 'Facility_H',
       'Facility_I', 'Facility_J', 'Facility_K', 'Total_Inductioned',
       'Weekday', 'Open', 'Midweek Prox', 'MidShift', 'Midweek', 'Mon', 'Tue',
       'Wed', 'Thu', 'Fri', 'Weekend', 'Facility_A_W', 'Facility_B_W',
       'Facility_C_W', 'Facility_D_W', 'Facility_E_W', 'Facility_F_W',
       'Facility_G_W', 'Facility_H_W', 'Facility_I_W', 'Facility_J_W',
       'Facility_K_W', 'Total_Inductioned_W', 'Total_Inductioned_W x Days',
       'Facility_A_W_t-1', 'Facility_B_W_t-1', 'Facility_C_W_t-1',
       'Facility_D_W_t-1', 'Facility_E_W_t-1', 'Facility_F_W_t-1',
       'Facility_G_W_t-1', 'Facility_H_W_t-1', 'Facility_I_W_t-1',
       'Facility_J_W_t-1', 'Facility_K_W_t-1', 'Midweek1', 'Midweek2',
       'Midweek3', 'Midweek4'],
      dtype='object')
```

[381]:

```
from sklearn import linear_model
x = new_delivery.iloc[:,-34:]
y = new_delivery['DELIVERED_VOLUME']
lm = linear_model.LinearRegression()
model = lm.fit(x,y)
predictions = lm.predict(x)
MAPE = 100 * np.mean(np.abs( np.array(y) - np.array(predictions)) / (np.
 ↪array(y) + 1 )) / len(y)
```

```
print(lm.coef_)
print(lm.intercept_)
print(MAPE)
lm.score(x,y)
```

```
[-1.00517163e+13  1.85987029e+11 -4.83686939e+12  1.85987029e+11
  1.85987028e+11 -1.00517163e+13  2.48869227e+02  2.39436872e+02
  2.57791918e+02  6.94722958e+01  2.35483340e+02  2.15274999e+02
  2.38854604e+02  2.55894181e+02  6.30253168e+02  3.79025179e+02
 -3.01074400e+03 -2.40561443e+02  9.37006859e-01 -2.29530199e+00
 -1.84347496e+01  8.96273845e+00  2.62678499e+01 -1.18407742e+00
 -3.67843833e+01  1.44783800e+00 -2.04378330e+01  5.65593547e+02
  1.12723879e+03  3.43074584e+03 -3.79654867e+12 -2.14344650e+12
  7.09149100e+11  1.59992044e+10]
10051716255928.87
41.40877722570868
```

[381]: 0.94676897022631

[389]:
```
avg_scenario1 = x.rolling(5,1).mean().iloc[-1,:]
avg_scenario2 = x.append(avg_scenario1).rolling(5,1).mean().iloc[-1,:]
avg_scenario3 = x.append([avg_scenario1, avg_scenario2]).rolling(5,1).mean().
 ↪iloc[-1,:]
avg_scenario4 = x.append([avg_scenario1, avg_scenario2, avg_scenario3]).
 ↪rolling(5,1).mean().iloc[-1,:]
avg_scenario5 = x.append([avg_scenario1, avg_scenario2, avg_scenario3,␣
 ↪avg_scenario4]).rolling(5,1).mean().iloc[-1,:]
print(avg_scenario5)
```

```
Mon                              0.000000
Tue                              0.000000
Wed                              0.069120
Thu                              0.126720
Fri                              0.174720
Weekend                          0.629440
Facility_A_W                    16.610158
Facility_B_W                     0.291127
Facility_C_W                     8.585845
Facility_D_W                     0.583475
Facility_E_W                    37.546902
Facility_F_W                     1.996756
Facility_G_W                    35.567639
Facility_H_W                     2.938564
Facility_I_W                     0.032189
Facility_J_W                     0.031345
Facility_K_W                     0.008195
Total_Inductioned_W            104.192194
```

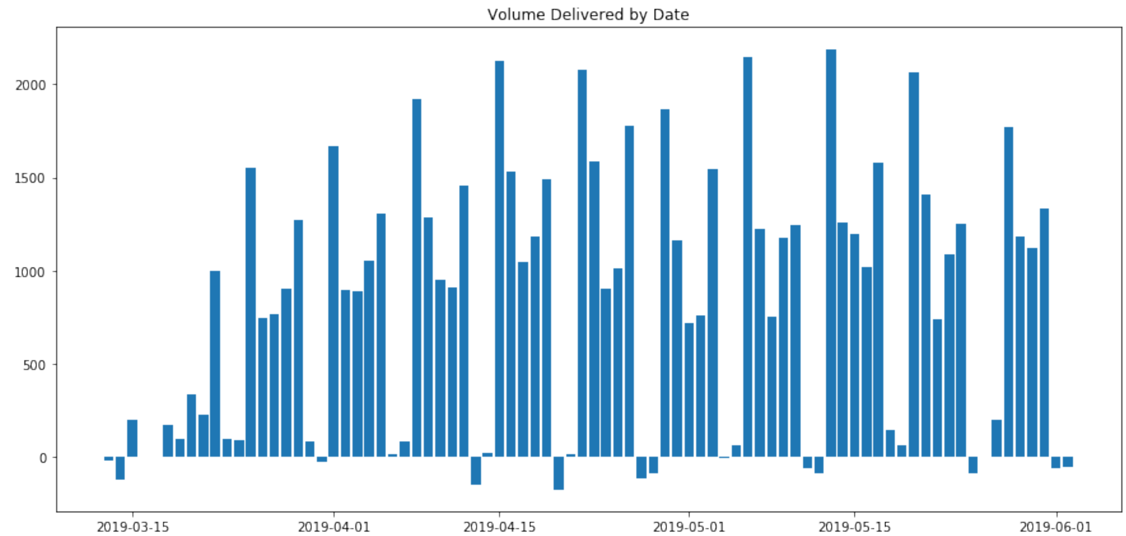
```
Total_Inductioned_W x Days    391.780015
Facility_A_W_t-1               26.654616
Facility_B_W_t-1                0.424963
Facility_C_W_t-1               13.443210
Facility_D_W_t-1                0.555907
Facility_E_W_t-1               58.805024
Facility_F_W_t-1                3.082355
Facility_G_W_t-1               57.537554
Facility_H_W_t-1                4.273753
Facility_I_W_t-1                0.053477
Facility_J_W_t-1                0.053731
Facility_K_W_t-1                0.012470
Midweek1                        0.846720
Midweek2                        2.148480
Midweek3                        5.800320
Midweek4                       16.248960
Name: 2019-06-02 00:00:00, dtype: float64
```

[390]:
```
scenario_dates = ['2019-6-03', '2019-6-04', '2019-6-05', '2019-6-06',␣
 ↪'2019-6-07']
scenario_days = [4, 2, 1, 2, 3]
df = pd.DataFrame( [avg_scenario1, avg_scenario2, avg_scenario3, avg_scenario4,␣
 ↪avg_scenario4] )
df.index = scenario_dates
df.Midweek1 = scenario_days
df.Midweek2 = df.Midweek1 ** 2
df.Midweek3 = df.Midweek1 ** 3
df.Midweek4 = df.Midweek1 ** 4
df.Mon = [1,0,0,0,0]
df.Tue = [0,1,0,0,0]
df.Wed = [0,0,1,0,0]
df.Thu = [0,0,0,1,0]
df.Fri = [0,0,0,0,1]
df.Weekend = [0,0,0,0,0]
predictions2 = lm.predict(df)
print(lm.coef_)
print(lm.intercept_)
print(predictions2)
# lm2.score(df, true_vol)
```

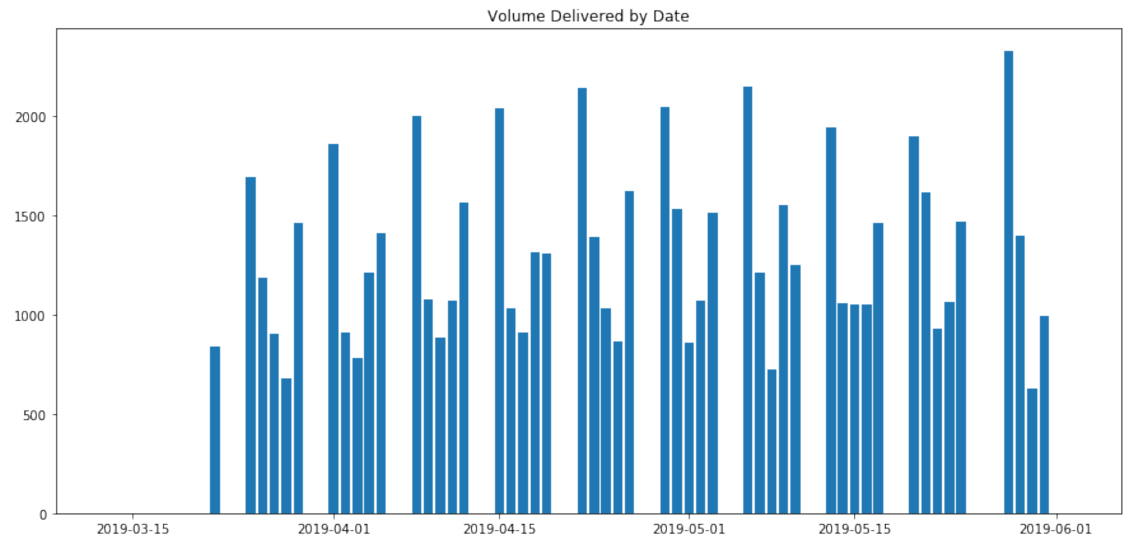
```
[-1.00517163e+13  1.85987029e+11 -4.83686939e+12  1.85987029e+11
  1.85987028e+11 -1.00517163e+13  2.48869227e+02  2.39436872e+02
  2.57791918e+02  6.94722958e+01  2.35483340e+02  2.15274999e+02
  2.38854604e+02  2.55894181e+02  6.30253168e+02  3.79025179e+02
 -3.01074400e+03 -2.40561443e+02  9.37006859e-01 -2.29530199e+00
 -1.84347496e+01  8.96273845e+00  2.62678499e+01 -1.18407742e+00
 -3.67843833e+01  1.44783800e+00 -2.04378330e+01  5.65593547e+02
  1.12723879e+03  3.43074584e+03 -3.79654867e+12 -2.14344650e+12
```

```
  7.09149100e+11  1.59992044e+10]
10051716255928.87
[798.01757812 408.27148438 488.41992188 144.96679688 239.109375  ]
```

[274]:
```
plt.figure(figsize=(15, 7))
#Delivery.plot.bar(figsize=(10,5))
plt.bar( new_delivery.index , predictions)
plt.title('Volume Delivered by Date')
plt.grid(False)
plt.show()
```



[275]:
```
plt.figure(figsize=(15, 7))
#Delivery.plot.bar(figsize=(10,5))
plt.bar(Delivery.index,Delivery.DELIVERED_VOLUME)
plt.title('Volume Delivered by Date')
plt.grid(False)
plt.show()
```



The plot above shows the delivery volumes at the client facility per day. The client facility is closed on the Saturday and Sundays, which is why you don't see any deliveries on the weekends. There were no deliveries on Memorial Day either. As you can see, there is a strong seasonal pattern the volume of parcels fluctuates quite a bit by weekday. This is why the clients are dependent on volume forecasts so they can manage their warehouse accordingly.

2 Example

2.1 Moving Averages

There are many different ways in which you could try to tackle this problem. A very simple approch would be to estimate the next days' volume by using the last days' volume. Given the daily fluctuation of volume, it is obvious that this is not a very promising approach. Another simple way would be to look at moving averages to smoothen out the predictions.

[276]:
```
def moving_average(series, n):
    """
        Calculate average of last n observations
    """
    return np.average(series[-n:])
```

[277]:
```
def plotMovingAverage(series, window, plot_intervals=False, scale=1.96,␣
 ↪plot_anomalies=False,day_out=5):

    """
        series - dataframe with timeseries
        window - rolling window size
        plot_intervals - show confidence intervals
```

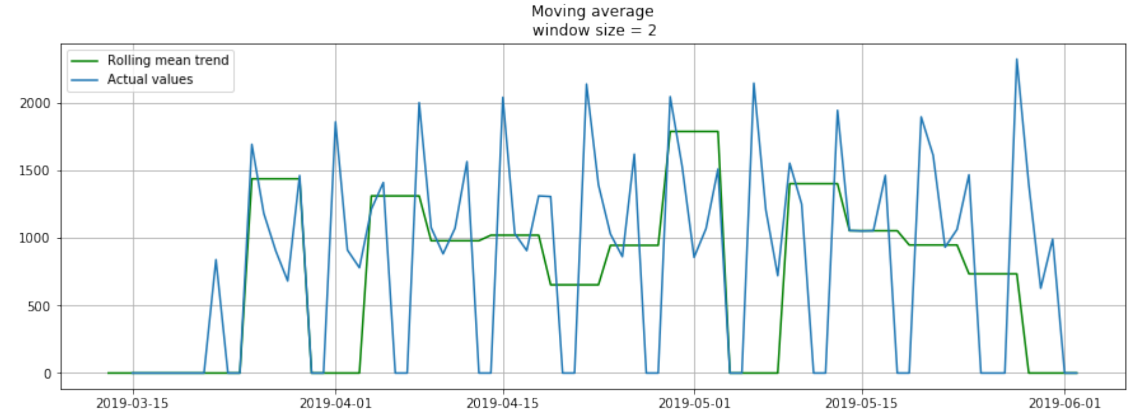
```
        plot_anomalies - show anomalies

    """

    Delivery['SMA_{}'.format(day_out)]=0
    for i in range(0,Delivery.shape[0]-day_out,day_out):
        for j in range(day_out):
            sum_val=0
            for k in range(window):
                sum_val+= Delivery.iloc[i+k,0]
            Delivery.loc[Delivery.index[i+j],'SMA_{}'.format(day_out)] = np.
↪round((sum_val/window),1)
    rolling_mean=Delivery['SMA_{}'.format(day_out)]

    plt.figure(figsize=(15,5))
    plt.title("Moving average\n window size = {}".format(window))
    plt.plot(rolling_mean, "g", label="Rolling mean trend")

    # Plot confidence intervals for smoothed values
    if plot_intervals:
        mae = mean_absolute_error(series[window:].iloc[:,0],␣
↪rolling_mean[window:])
        deviation = np.std(series[window:].iloc[:,0] - rolling_mean[window:])
        lower_bond = rolling_mean - (mae + scale * deviation)
        upper_bond = rolling_mean + (mae + scale * deviation)
        plt.plot(upper_bond, "r--", label="Upper Bond / Lower Bond")
        plt.plot(lower_bond, "r--")

        # Having the intervals, find abnormal values
        if plot_anomalies:
            anomalies = pd.DataFrame(index=series.index, columns=series.columns)
            anomalies[series<lower_bond] = series[series<lower_bond]
            anomalies[series>upper_bond] = series[series>upper_bond]
            plt.plot(anomalies, "ro", markersize=10)

    plt.plot(series[window:].iloc[:,0], label="Actual values")
    plt.legend(loc="upper left")
    plt.grid(True)
```

Parameters:

- window - no of days to consider when calculating the average
- day_out - the number of days for which you are generating the forecasts

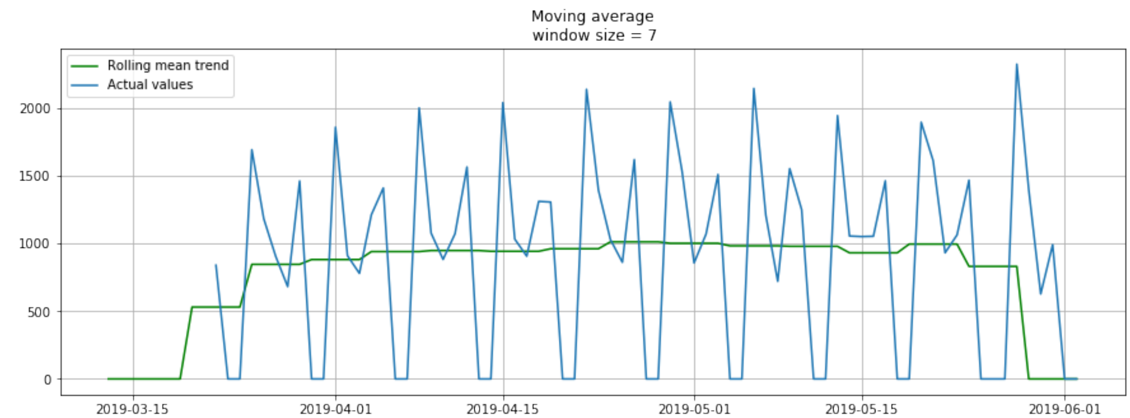
This is what you get when you calculate the average using the volumes delivered over the last 2 days

```
[278]: plotMovingAverage(Delivery,2,day_out=5)
```



As you can see, the forecast is a bit more smooth, but it misses the peaks in delivery volume. Now let's try doing so using the volumes over the previous 7 days.

```
[279]: plotMovingAverage(Delivery, 7,day_out=5)
```



As expected, the predictions smoothen out more, but the also become less usefull since they eventually converge to the average daily delivery volume. Our clients need more accurate forecasts for the individual days.

2.1.1 Notes and some ideas:

- You are not limited to the 13 columns (date, delivery total, induction totals for 11 facilities) provided in the data set, you can generate additional features that might be useful for your predictions. For instance, you could add a feature like weekday to your model. This might be helpful to capture weekday-specific patterns.

- Lag variables are very usefull for forecast problems. For instance, you could create additional columns like yesterdays volume (day-1), the day before yesterday (day-2), etc. This might be helpful when trying to use the induction volumes for your model, as they delivery volume follow the induction volumes (with a few days lag depending on where the induction happened).
- When building your model, make sure that you get a good understanding of the model's performance by using the historic data for back-testing. Common methods for forecasting problems are rolling window approaches.
- The examples above are very basic examples for illustration. You could for instance explore time-series packages that are readily availabe, or try to build a ML model that makes use of additional features you generate.

2.1.2 Reach out if you have questions:

Through which channel should they reach out?

2.1.3 Further reading:

- Time Series
- Jupyter Notebooks

2.2 Good luck!