Agmatix - Noam Natan

August 18, 2021

1 Intro

```
In [1]: # Since purpose and design (including tools, best practices and future suggestions)
        # had already been presented and writtend in first interview/test on premises
        # , then jumping into actual code w/ clarifications

In [2]: # So, this code is to be invoked in the env where source files are located
        # digesting and cleaning the data
        # uploading to db
```

2 Imports

```
In [3]: import pandas as pd
        import os
        import re
        from datetime import datetime
        from IPython.display import display
        import numpy as np

In [4]: pd.set_option('display.max_columns', None)
        pd.set_option('display.expand_frame_repr', False)
        pd.set_option('max_colwidth', -1)
```

3 Start logging report

```
In [5]: log_df_cols = ['timestamp', 'severity', 'flag', 'msg']

        log_df = pd.DataFrame(data=[[datetime.now(), 'high', 'success','started log_df']]
                              ,columns=log_df_cols)

        def add_log(log_vals):
            global log_df
            log_vals = [datetime.now()]+log_vals
            log_df = pd.concat([pd.DataFrame(data=[log_vals], columns=log_df_cols), log_df])
            log_df.head()
            return log_df
```

```
log_df
```

Out[5]:

	timestamp	severity	flag	msg
0	2021-08-18 20:41:20.125081	high	success	started log_df

4 Load all files in desitination folder

```
In [6]: # folder_path  = input('enter folder_path')
        folder_path  = r'C:\Users\Noam\Desktop\Agmatix\agmatix-data-interview\agriculture'
        # C:\Users\Noam\Desktop\Agmatix\agmatix-data-interview\agriculture

In [7]: file_names = os.listdir(folder_path)
        file_names = [f for f in file_names if (f[-3:] == 'xls' or f[-4:] == 'xlsx') and f[0]
        print (file_names)
```

```
['climate.xlsx', 'samples.xlsx', 'sensors.xlsx']
```

```
In [8]: add_log(['high', 'success','folder path entered and f_names parsed, total: '+str(len(f
```

Out[8]:

	timestamp	severity	flag	
0	2021-08-18 20:41:20.174116	high	success	folder path entered and f_names parsed
0	2021-08-18 20:41:20.125081	high	success	started log_df

4.1 convert files to Dataframe

4.2 w/ files integrity check : readable and shape

```
In [9]: df_cnt = 0
        src_list = []
        for f in file_names:
            df_name = f.split('.')[0]+'_df'
            path = folder_path+'\\'+f
            path = path.replace('\\','/')

            file = pd.ExcelFile(path)
            sheets = file.sheet_names

            for i in range(len(sheets)):
                s = sheets[i]
                df_s_name = df_name+'_'+s
                df_s_name = df_s_name.replace(' ','').replace('-','_')
                try:
                    cmd  = r'''{df_s_name} = pd.read_excel('{path}', sheet_name={i})'''.format
                    exec(cmd)
                    df_cnt += 1
                    exec(r'''df_shp = {df_s_name}.shape'''.format(df_s_name=df_s_name))
                    assert df_shp!=(0,0), r'''{df_s_name} is empty'''.format(df_s_name=df_s_na
```

```
                    add_log(['high', 'success','loaded file to dataframe: '+df_s_name+' shape:
                    src_list.append(df_s_name)
                    print(df_s_name)
                    exec(r'''display({df_s_name}.head(2))'''.format(df_s_name=df_s_name))
                except:
                    add_log(['high', 'fail','fail to load file to dataframe: '+df_s_name])
        display(log_df)
```

climate_df_Recovered_Sheet1

	Latitude	Longitude	Farm A	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7
0	42.439999	-76.459999	ET	Rain	Tmin	Tmax	Hr	Date
1	42.439999	-76.459999	mm	mm	Cř	Cř	NaN	NaN

samples_df_metadata_treatments

	Research_ID	trial_ID	treatment ID	tratment timing	treatment date
0	51	1	1	Oat	2012-05-18 00:00:00
1	51	1	2	Pre-plant	2012-08-28 00:00:00

samples_df_sampels

	Latitude	Longitude	trial	Repetition	treatment ID	N timing	Grain yield	Total Bioma
0	42.439999	-76.459999	1	1	1	Oat tillering	11.9	27.0
1	42.439999	-76.459999	1	2	1	Oat tillering	NaN	25.1

sensors_df_25cm

	Latitude	Longitude	Date	Date.1	Time	Value (Kpa)
0	42.439999	-76.459999	2018-05-13 19:46:10	2018-05-13	19:46:10	5.812
1	42.439999	-76.459999	2018-05-13 20:01:10	2018-05-13	20:01:10	5.812

sensors_df_50cm

	Date	Date.1	Time	Value (Kpa)
0	2018-05-13 19:46:10	2018-05-13	19:46:10	25.412
1	2018-05-13 20:01:10	2018-05-13	20:01:10	25.319

sensors_df_75cm

```
                 Date     Date.1      Time  Value (Kpa)
0 2018-05-13 19:46:10 2018-05-13  19:46:10  23.087
1 2018-05-13 20:01:10 2018-05-13  20:01:10  23.134
```

```
                   timestamp severity     flag
0 2021-08-18 20:41:27.074038  high     success  loaded file to dataframe: sensors_df_75cm shape
0 2021-08-18 20:41:25.493915  high     success  loaded file to dataframe: sensors_df_50cm shape
0 2021-08-18 20:41:23.903767  high     success  loaded file to dataframe: sensors_df_25cm shape
0 2021-08-18 20:41:20.891644  high     success  loaded file to dataframe: samples_df_sampels sh
0 2021-08-18 20:41:20.692484  high     success  loaded file to dataframe: samples_df_metadata_t
0 2021-08-18 20:41:20.355245  high     success  loaded file to dataframe: climate_df_Recovered_
0 2021-08-18 20:41:20.174116  high     success  folder path entered and f_names parsed, total:
0 2021-08-18 20:41:20.125081  high     success  started log_df
```

In [10]:
```
# delete irrelevant sources according to assignment info

src_del = ['sensors_df_50cm', 'sensors_df_75cm']
src_list = [s for s in src_list if s not in src_del]
print(src_list)

add_log(['low', 'success','delete irrelevant sources'])
```

```
['climate_df_Recovered_Sheet1', 'samples_df_metadata_treatments', 'samples_df_sampels', 'sensor
```

Out[10]:
```
                   timestamp severity     flag
0 2021-08-18 20:41:27.095034  low      success  delete irrelevant sources
0 2021-08-18 20:41:27.074038  high     success  loaded file to dataframe: sensors_df_7
0 2021-08-18 20:41:25.493915  high     success  loaded file to dataframe: sensors_df_5
0 2021-08-18 20:41:23.903767  high     success  loaded file to dataframe: sensors_df_2
0 2021-08-18 20:41:20.891644  high     success  loaded file to dataframe: samples_df_s
0 2021-08-18 20:41:20.692484  high     success  loaded file to dataframe: samples_df_m
0 2021-08-18 20:41:20.355245  high     success  loaded file to dataframe: climate_df_R
0 2021-08-18 20:41:20.174116  high     success  folder path entered and f_names parsed
0 2021-08-18 20:41:20.125081  high     success  started log_df
```

5 Reformat sets & Data validaiton

5.0.1 Null records

In [11]:
```
# I didn't remove records according to Null values
# Since there are several fillin practices for that
# and there missig professional contex to understand which columns are must-have or n
```

In [12]:
```
# sample code for removing all Null records from sources and add to dedicated 'BAD' r

# for src in src_list:
```

```
#     cmd  = r'''{src}_BAD = {src}[{src}.isnull().any(axis=1)]'''.format(src=src)
#     exec(cmd)
#     cmd  = r'''index_with_null = {src}.index[{src}.isnull().any(axis=1)]'''.format(
#     exec(cmd)
#     cmd  = r'''{src}.drop(index_with_null,0, inplace=True)'''.format(src=src)
#     exec(cmd)
```

In [13]:
```
def check_df_stats(df):
    cols = df.select_dtypes(include='float64').columns.tolist()
    df_dict = {}
    for c in cols:
        df_dict[c] = {'mean' : np.mean(df[c]), 'std' : np.std(df[c])}

    def check_value_stats(v):
        threshold = 3
        mean = df_dict[c]['mean']
        std = df_dict[c]['std']
        z_score= (v - mean)/std
        if np.abs(z_score) > threshold: return 0
        else: return 1

    for c in cols:
        df[c+'_c'] = df[c].apply(check_value_stats)

    cols_c = [c+'_c' for c in cols]
    df['stats_check'] = df[cols_c].sum(axis=1)==len(cols_c)
    # df[df['stats_check']==True]

    return
```

5.1 climate_df_Recovered_Sheet1

In [14]:
```
# fixing climate_df columns header
climate_df_Recovered_Sheet1.columns = climate_df_Recovered_Sheet1.columns.tolist()[:2]

# removing climate_df_Recovered_Sheet1 invalid rows
climate_df_Recovered_Sheet1_BAD = climate_df_Recovered_Sheet1.iloc[:2,:].copy()
climate_df_Recovered_Sheet1 = climate_df_Recovered_Sheet1.iloc[2:,:]
# climate_df_Recovered_Sheet1.head(2)

# data types relevant to values, reformat and remove mismatches if needed
climate_df_Recovered_Sheet1 = climate_df_Recovered_Sheet1.astype({k:'float64' for k in
climate_df_Recovered_Sheet1['HR'] = pd.to_datetime(climate_df_Recovered_Sheet1['Hr'],
del climate_df_Recovered_Sheet1['Hr']
climate_df_Recovered_Sheet1['DATE'] = pd.to_datetime(climate_df_Recovered_Sheet1['Date
del climate_df_Recovered_Sheet1['Date']
display(climate_df_Recovered_Sheet1.head(2))
climate_df_Recovered_Sheet1.info()
```

```
    Latitude  Longitude    ET  Rain  Tmin  Tmax        HR       DATE
2  42.439999 -76.459999  3.29  1.1   13.9  21.7  00:00:00 2018-03-24
3  42.439999 -76.459999  4.25  0.0   7.7   24.9  00:00:00 2018-03-25


<class 'pandas.core.frame.DataFrame'>
RangeIndex: 160 entries, 2 to 161
Data columns (total 8 columns):
Latitude     160 non-null float64
Longitude    160 non-null float64
ET           158 non-null float64
Rain         157 non-null float64
Tmin         160 non-null float64
Tmax         158 non-null float64
HR           160 non-null object
DATE         160 non-null datetime64[ns]
dtypes: datetime64[ns](1), float64(6), object(1)
memory usage: 10.1+ KB


In [15]: # check Date delta

         climate_df_Recovered_Sheet1['date_diff'] = climate_df_Recovered_Sheet1['DATE'].dt.date
         climate_df_Recovered_Sheet1['date_diff'].iloc[0] = pd.Timedelta('1 days 00:00:00')


         # add possaible rows with longer date diff (of 1 day) to BAD_df
         climate_df_Recovered_Sheet1_BAD = pd.concat([climate_df_Recovered_Sheet1_BAD
                                                      , climate_df_Recovered_Sheet1[climate_df_
         climate_df_Recovered_Sheet1 = climate_df_Recovered_Sheet1[climate_df_Recovered_Sheet1

C:\Program Files\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  self._setitem_with_indexer(indexer, value)
C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:9: FutureWarning: Sorting be
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.


In [16]: # check_df_stats

         check_df_stats(climate_df_Recovered_Sheet1)
```

```
climate_df_Recovered_Sheet1_BAD = climate_df_Recovered_Sheet1[climate_df_Recovered_She
climate_df_Recovered_Sheet1 = climate_df_Recovered_Sheet1[climate_df_Recovered_Sheet1
climate_df_Recovered_Sheet1.head()
```

Out[16]:

	Latitude	Longitude	ET	Rain	Tmin	Tmax	HR	DATE	date_diff	Latitu
2	42.439999	-76.459999	3.29	1.1	13.9	21.7	00:00:00	2018-03-24	1 days	1
3	42.439999	-76.459999	4.25	0.0	7.7	24.9	00:00:00	2018-03-25	1 days	1
4	42.439999	-76.459999	NaN	0.0	4.2	31.2	00:00:00	2018-03-26	1 days	1
5	42.439999	-76.459999	4.08	0.0	12.9	32.3	00:00:00	2018-03-27	1 days	1
6	42.439999	-76.459999	3.16	1.1	13.7	29.8	00:00:00	2018-03-28	1 days	1

In [17]:
```
add_log(['low', 'success','finished work on set: climate_df_Recovered_Sheet1'])
```

Out[17]:

	timestamp	severity	flag	
0	2021-08-18 20:41:27.508328	low	success	finished work on set: climate_df_Recov
0	2021-08-18 20:41:27.095034	low	success	delete irrelevant sources
0	2021-08-18 20:41:27.074038	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:25.493915	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:23.903767	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:20.891644	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.692484	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.355245	high	success	loaded file to dataframe: climate_df_
0	2021-08-18 20:41:20.174116	high	success	folder path entered and f_names parse
0	2021-08-18 20:41:20.125081	high	success	started log_df

5.1.1 samples_df_metadata_treatments

In [18]:
```
samples_df_metadata_treatments
```

Out[18]:

	Research_ID	trial_ID	treatment ID	tratment timing	treatme
0	51	1	1	Oat	2012-05-18 00:00:00
1	51	1	2	Pre-plant	2012-08-28 00:00:00
2	51	1	3	Planting	2012-09-22 00:00:00
3	51	1	4	V3 (three-leaf) corn	2012-10-16 00:00:00
4	51	1	5	V3 / V6 (six-leaf) corn	Oct, 16 , 2012/ Nov,
5	51	2	1	Oat	2012-05-20 00:00:00
6	51	2	2	Pre-plant	2012-10-30 00:00:00
7	51	2	3	Planting	2012-11-12 00:00:00
8	51	2	4	V3 (three-leaf) corn	2012-12-04 00:00:00
9	51	2	5	V3 / V6 (six-leaf) corn	12/4/2012, 16/4/2012

In [19]:
```
row = samples_df_metadata_treatments.iloc[[4,]].copy()
new_row = pd.concat([row,row]).reset_index(drop=True)

# redesign rows

treat_time = new_row.iloc[0,3].split('/')
```

```
        treat_time = [i.strip() for i in treat_time]
        for i in range(len(treat_time)):
            new_row.iloc[i,3] = treat_time[i]

        treat_date = new_row.iloc[0,4].split('/')
        treat_date = [datetime.strptime(i.replace(' ',''), "%b,%d,%Y") for i in treat_date]
        for i in range(len(treat_date)):
            new_row.iloc[i,4] = treat_date[i]

        samples_df_metadata_treatments = pd.concat([samples_df_metadata_treatments.iloc[:4,]
                                                    , new_row
                                                    , samples_df_metadata_treatments.iloc[5:,]
        display(samples_df_metadata_treatments)
```

	Research_ID	trial_ID	treatment ID	tratment timing	treatment date
0	51	1	1	Oat	2012-05-18 00:00:00
1	51	1	2	Pre-plant	2012-08-28 00:00:00
2	51	1	3	Planting	2012-09-22 00:00:00
3	51	1	4	V3 (three-leaf) corn	2012-10-16 00:00:00
4	51	1	5	V3	2012-10-16 00:00:00
5	51	1	5	V6 (six-leaf) corn	2012-11-01 00:00:00
6	51	2	1	Oat	2012-05-20 00:00:00
7	51	2	2	Pre-plant	2012-10-30 00:00:00
8	51	2	3	Planting	2012-11-12 00:00:00
9	51	2	4	V3 (three-leaf) corn	2012-12-04 00:00:00
10	51	2	5	V3 / V6 (six-leaf) corn	12/4/2012, 16/4/2012

In [20]:
```
row = samples_df_metadata_treatments.iloc[[10,]].copy()
new_row = pd.concat([row,row]).reset_index(drop=True)

# redesign rows

treat_time = new_row.iloc[0,3].split('/')
treat_time = [i.strip() for i in treat_time]
for i in range(len(treat_time)):
    new_row.iloc[i,3] = treat_time[i]

treat_date = new_row.iloc[0,4].split(',')
print(treat_date)
treat_date = [datetime.strptime(i.replace(' ',''), "%d/%m/%Y") for i in treat_date]
for i in range(len(treat_date)):
    new_row.iloc[i,4] = treat_date[i]

samples_df_metadata_treatments = pd.concat([samples_df_metadata_treatments.iloc[:10,]
                                            , new_row]).reset_index(drop=True)
display(samples_df_metadata_treatments)
```

```
['12/4/2012', ' 16/4/2012']
```

	Research_ID	trial_ID	treatment ID	tratment timing	treatment date
0	51	1	1	Oat	2012-05-18 00:00:00
1	51	1	2	Pre-plant	2012-08-28 00:00:00
2	51	1	3	Planting	2012-09-22 00:00:00
3	51	1	4	V3 (three-leaf) corn	2012-10-16 00:00:00
4	51	1	5	V3	2012-10-16 00:00:00
5	51	1	5	V6 (six-leaf) corn	2012-11-01 00:00:00
6	51	2	1	Oat	2012-05-20 00:00:00
7	51	2	2	Pre-plant	2012-10-30 00:00:00
8	51	2	3	Planting	2012-11-12 00:00:00
9	51	2	4	V3 (three-leaf) corn	2012-12-04 00:00:00
10	51	2	5	V3	2012-04-12 00:00:00
11	51	2	5	V6 (six-leaf) corn	2012-04-16 00:00:00

In [21]:
```
add_log(['low', 'success','finished work on set: samples_df_metadata_treatments'])
```

Out[21]:

	timestamp	severity	flag	
0	2021-08-18 20:41:27.679451	low	success	finished work on set: samples_df_meta
0	2021-08-18 20:41:27.508328	low	success	finished work on set: climate_df_Reco
0	2021-08-18 20:41:27.095034	low	success	delete irrelevant sources
0	2021-08-18 20:41:27.074038	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:25.493915	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:23.903767	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:20.891644	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.692484	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.355245	high	success	loaded file to dataframe: climate_df_
0	2021-08-18 20:41:20.174116	high	success	folder path entered and f_names parse
0	2021-08-18 20:41:20.125081	high	success	started log_df

5.1.2 samples_df_sampels

In [22]:
```
# check for outlier by value counts

outlier_dict = {}
for c in samples_df_sampels.select_dtypes(include=['int64', 'object']).columns.tolist
    df = samples_df_sampels[c].value_counts().to_frame().reset_index()
    outs = df['index'][df[c]==1].values.tolist()
    if len(outs)!=0: outlier_dict[c] = outs

outlier_dict.keys()
```

Out[22]:
```
dict_keys(['Total N content', 'Grain yield', 'treatment ID'])
```

In [23]:
```
outlier_dict['treatment ID']
```

Out[23]: ['#']

In [24]:
```
samples_df_sampels_BAD = samples_df_sampels[samples_df_sampels['treatment ID'].isin(ou
samples_df_sampels = samples_df_sampels[~samples_df_sampels['treatment ID'].isin(outli
```

In [25]:
```
# check date columns in set
# looks like they have good readable format

cols = samples_df_sampels.columns.tolist()
date_cols = [c for c in cols if re.search("date", c,  re.IGNORECASE)]
samples_df_sampels[date_cols].tail(2)
```

Out[25]:

	Oat Planting Date	Corn Planting Date	N date application	second N date application
38	2012-05-03	2012-11-13	2012-12-04	16/4/2012
39	2012-05-03	2012-11-13	2012-12-04	16/4/2012

In [26]:
```
# object columns check

obj_cols = samples_df_sampels.select_dtypes(include=['object']).columns.tolist()
samples_df_sampels[obj_cols].head(2)
```

Out[26]:

	treatment ID	N timing	Grain yield	Total N content	Oat	Corn Hybrids
0	1	Oat tillering	11.9	232.1	Avena sativa	AS1555
1	1	Oat tillering	NaN	311.4	Avena sativa	AS1555

In [27]:
```
obj_cols
```

Out[27]:
```
['treatment ID',
 'N timing',
 'Grain yield',
 'Total N content',
 'Oat',
 'Corn Hybrids',
 'product name',
 'second N date application']
```

In [28]:
```
# converting

samples_df_sampels['Grain yield'] = samples_df_sampels['Grain yield'].apply(lambda x:
samples_df_sampels['Total N content'] = samples_df_sampels['Total N content'].apply(la

samples_df_sampels = samples_df_sampels.astype({'treatment ID':'int64', 'Grain yield'
```

In [29]:
```
# int columns check
# looks good
int_cols = samples_df_sampels.select_dtypes(include=['int64']).columns.tolist()
samples_df_sampels[int_cols].head(2)
```

Out[29]:

	trial	Repetition	treatment ID	Clay	Silt	Sand	Precipitation	Plant populati
0	1	1	1	670	230	100	2154	70000
1	1	2	1	670	230	100	2154	70000

In [30]:
```
# check_df_stats

check_df_stats(samples_df_sampels)

samples_df_sampels_BAD = samples_df_sampels[samples_df_sampels['stats_check']==False]
samples_df_sampels = samples_df_sampels[samples_df_sampels['stats_check']==True]
samples_df_sampels.head()
```

Out[30]:

	Latitude	Longitude	trial	Repetition	treatment ID	N timing	Grain yield
0	42.439999	-76.459999	1	1	1	Oat tillering	11.9
1	42.439999	-76.459999	1	2	1	Oat tillering	NaN
2	42.439999	-76.459999	1	3	1	Oat tillering	14.2
4	42.439999	-76.459999	1	1	2	Pre-planting	13.9
5	42.439999	-76.459999	1	2	2	Pre-planting	18.5

In [31]:
```
add_log(['low', 'success','finished work on set: samples_df_sampels'])
```

Out[31]:

	timestamp	severity	flag	
0	2021-08-18 20:41:28.236846	low	success	finished work on set: samples_df_samp
0	2021-08-18 20:41:27.679451	low	success	finished work on set: samples_df_meta
0	2021-08-18 20:41:27.508328	low	success	finished work on set: climate_df_Reco
0	2021-08-18 20:41:27.095034	low	success	delete irrelevant sources
0	2021-08-18 20:41:27.074038	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:25.493915	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:23.903767	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:20.891644	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.692484	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.355245	high	success	loaded file to dataframe: climate_df_
0	2021-08-18 20:41:20.174116	high	success	folder path entered and f_names parse
0	2021-08-18 20:41:20.125081	high	success	started log_df

5.1.3 sensors_df_25cm

In [32]:
```
sensors_df_25cm.head()
```

Out[32]:

	Latitude	Longitude	Date	Date.1	Time	Value (Kpa)
0	42.439999	-76.459999	2018-05-13 19:46:10	2018-05-13	19:46:10	5.812
1	42.439999	-76.459999	2018-05-13 20:01:10	2018-05-13	20:01:10	5.812
2	42.439999	-76.459999	2018-05-13 20:16:10	2018-05-13	20:16:10	5.859
3	42.439999	-76.459999	2018-05-13 20:31:10	2018-05-13	20:31:10	5.859
4	42.439999	-76.459999	2018-05-13 20:46:10	2018-05-13	20:46:10	5.929

In [33]:
```
# since 'Date' stores full timestamp, then there's no need (currently) to divide to t
del sensors_df_25cm['Date.1']
del sensors_df_25cm['Time']
```

In [34]:
```
# check_df_stats

check_df_stats(sensors_df_25cm)

sensors_df_25cm_BAD = sensors_df_25cm[sensors_df_25cm['stats_check']==False]
sensors_df_25cm = sensors_df_25cm[sensors_df_25cm['stats_check']==True]
sensors_df_25cm.head()
```

Out[34]:

	Latitude	Longitude	Date	Value (Kpa)	Latitude_c	Longitude_c	Va
0	42.439999	-76.459999	2018-05-13 19:46:10	5.812	1	1	1
1	42.439999	-76.459999	2018-05-13 20:01:10	5.812	1	1	1
2	42.439999	-76.459999	2018-05-13 20:16:10	5.859	1	1	1
3	42.439999	-76.459999	2018-05-13 20:31:10	5.859	1	1	1
4	42.439999	-76.459999	2018-05-13 20:46:10	5.929	1	1	1

In [35]:
```
sensors_df_25cm_BAD
```

Out[35]:

	Latitude	Longitude	Date	Value (Kpa)	Latitude_c	Longitude_c
544	42.439999	-76.459999	2018-05-19 19:16:20	67.495	1	1
545	42.439999	-76.459999	2018-05-19 19:31:20	68.262	1	1
546	42.439999	-76.459999	2018-05-19 19:46:20	68.588	1	1
547	42.439999	-76.459999	2018-05-19 20:01:20	68.913	1	1
548	42.439999	-76.459999	2018-05-19 20:16:20	69.238	1	1
549	42.439999	-76.459999	2018-05-19 20:31:20	69.262	1	1
550	42.439999	-76.459999	2018-05-19 20:46:20	69.494	1	1
551	42.439999	-76.459999	2018-05-19 21:01:20	69.448	1	1
552	42.439999	-76.459999	2018-05-19 21:16:20	69.448	1	1
553	42.439999	-76.459999	2018-05-19 21:31:20	69.587	1	1
554	42.439999	-76.459999	2018-05-19 21:46:20	69.494	1	1
555	42.439999	-76.459999	2018-05-19 22:01:20	69.494	1	1
556	42.439999	-76.459999	2018-05-19 22:16:20	69.448	1	1
557	42.439999	-76.459999	2018-05-19 22:31:20	69.308	1	1
558	42.439999	-76.459999	2018-05-19 22:46:20	69.169	1	1
559	42.439999	-76.459999	2018-05-19 23:01:20	68.936	1	1
560	42.439999	-76.459999	2018-05-19 23:16:20	68.657	1	1
561	42.439999	-76.459999	2018-05-19 23:31:20	68.355	1	1
562	42.439999	-76.459999	2018-05-19 23:46:20	68.192	1	1
563	42.439999	-76.459999	2018-05-20 00:01:20	67.844	1	1
564	42.439999	-76.459999	2018-05-20 00:16:20	67.518	1	1
565	42.439999	-76.459999	2018-05-20 00:31:20	67.216	1	1
7593	42.439999	-76.459999	2018-08-01 08:29:30	68.820	1	1
7594	42.439999	-76.459999	2018-08-01 08:44:30	70.634	1	1
7595	42.439999	-76.459999	2018-08-01 08:59:30	72.935	1	1
7596	42.439999	-76.459999	2018-08-01 09:14:30	74.795	1	1
7597	42.439999	-76.459999	2018-08-01 09:29:30	75.934	1	1

In [36]:
```
add_log(['low', 'success','finished work on set: sensors_df_25cm'])
```

Out[36]:

	timestamp	severity	flag	
0	2021-08-18 20:41:28.476021	low	success	finished work on set: sensors_df_25cm

```
0 2021-08-18 20:41:28.236846  low      success  finished work on set: samples_df_sampe
0 2021-08-18 20:41:27.679451  low      success  finished work on set: samples_df_metac
0 2021-08-18 20:41:27.508328  low      success  finished work on set: climate_df_Recov
0 2021-08-18 20:41:27.095034  low      success  delete irrelevant sources
0 2021-08-18 20:41:27.074038  high     success  loaded file to dataframe: sensors_df_1
0 2021-08-18 20:41:25.493915  high     success  loaded file to dataframe: sensors_df_5
0 2021-08-18 20:41:23.903767  high     success  loaded file to dataframe: sensors_df_2
0 2021-08-18 20:41:20.891644  high     success  loaded file to dataframe: samples_df_s
0 2021-08-18 20:41:20.692484  high     success  loaded file to dataframe: samples_df_m
0 2021-08-18 20:41:20.355245  high     success  loaded file to dataframe: climate_df_R
0 2021-08-18 20:41:20.174116  high     success  folder path entered and f_names parsed
0 2021-08-18 20:41:20.125081  high     success  started log_df
```

---

6 Storing to DB

SQLite is an open-source, zero-configuration, self-contained, stand-alone, transaction relational database engine

```
In [37]: add_sets = ['log_df', 'climate_df_Recovered_Sheet1_BAD', 'samples_df_sampels_BAD', 'se
         src_list += add_sets
         src_list
```

```
Out[37]: ['climate_df_Recovered_Sheet1',
          'samples_df_metadata_treatments',
          'samples_df_sampels',
          'sensors_df_25cm',
          'log_df',
          'climate_df_Recovered_Sheet1_BAD',
          'samples_df_sampels_BAD',
          'sensors_df_25cm_BAD']
```

```
In [38]: import sqlite3
         conn = sqlite3.connect('noam_db.sqlite')

         for s in src_list:
             cmd  = r'''{s}.to_sql('{s}', conn, if_exists="replace")'''.format(s=s)
             exec(cmd)

         conn.commit()
```

```
C:\Program Files\Anaconda3\lib\site-packages\pandas\io\sql.py:450: UserWarning: the 'timedelta
  chunksize=chunksize, dtype=dtype)
C:\Program Files\Anaconda3\lib\site-packages\pandas\core\generic.py:2130: UserWarning: The spac
  dtype=dtype)
```

```
In [39]: add_log(['low', 'success','finished storing all sets to DB'])
```

Out[39]:

	timestamp	severity	flag	
0	2021-08-18 20:41:29.081446	low	success	finished storing all sets to DB
0	2021-08-18 20:41:28.476021	low	success	finished work on set: sensors_df_25cm
0	2021-08-18 20:41:28.236846	low	success	finished work on set: samples_df_samp
0	2021-08-18 20:41:27.679451	low	success	finished work on set: samples_df_meta
0	2021-08-18 20:41:27.508328	low	success	finished work on set: climate_df_Reco
0	2021-08-18 20:41:27.095034	low	success	delete irrelevant sources
0	2021-08-18 20:41:27.074038	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:25.493915	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:23.903767	high	success	loaded file to dataframe: sensors_df_
0	2021-08-18 20:41:20.891644	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.692484	high	success	loaded file to dataframe: samples_df_
0	2021-08-18 20:41:20.355245	high	success	loaded file to dataframe: climate_df_
0	2021-08-18 20:41:20.174116	high	success	folder path entered and f_names parse
0	2021-08-18 20:41:20.125081	high	success	started log_df