STA 141 Project

December 9, 2019

1 STA 141A Project Python Code

2 (James Hizon)

3 Import essential packages

[2]:
```
import os
import pandas as pd
import linecache
import itertools as it
import functools
import numpy as np
import matplotlib.pyplot as plt
import mpld3
import plotly
import plotly.offline as pyo
# ENABLE INTERACTIVE MATPLOTLIB INTERACTION:
mpld3.enable_notebook()
%matplotlib inline
```

[3]:
```
# Work Directory:
os.chdir('/home/james/RS Directory Path Folder/
 ↪9_23_2019_proto_board_10_A_10_,1pbs_trial1') # path -> working directory
os.scandir(path='/home/james/RS Directory Path Folder/
 ↪9_23_2019_proto_board_10_A_10_,1pbs_trial1')
print("Current Working Directory", os.getcwd())
```

```
Current Working Directory /home/james/RS Directory Path
Folder/9_23_2019_proto_board_10_A_10_,1pbs_trial1
```

[4]:
```
def preprocess_fromdf(filename):
    """
    Function:
    - Reads a .csv file.
    - Creates a dataframe as needed.
    """
```

```
    df = pd.read_csv(filename)
    # MAYBE, have to drop columns only if the values inside are NULL.
    # UPDATE: I removed Phase from dropping, b/c exists inside raw data file.␣
 ↪ASK ABOUT IT.
    df = df.drop(['Bias Voltage','Phase','Conductance', 'Susceptance',␣
 ↪'Resistance','Reactance', 'Capacitance Parallel','Quality␣
 ↪Factor','Capacitance Series','Dissipation Factor','Inductance␣
 ↪Series','ESR'], axis=1)
    df['Frequency'] = df['Frequency'].map(lambda x: x.rstrip('KHZ'))
    df['Voltage'] = df['Voltage'].map(lambda x: x.rstrip(' MV'))
    df['Frequency'] = pd.to_numeric(df['Frequency'])
    df['Voltage'] = pd.to_numeric(df['Voltage'])

    return df

# LIST OF CSV FILES:
csvFiles_List = []
for i in range(10):
    csvFiles_List += ['Well_'+str(i+1)+'.csv']
csvFiles_List
```

[4]:
```
['Well_1.csv',
 'Well_2.csv',
 'Well_3.csv',
 'Well_4.csv',
 'Well_5.csv',
 'Well_6.csv',
 'Well_7.csv',
 'Well_8.csv',
 'Well_9.csv',
 'Well_10.csv']
```

[5]:
```
# USE FOR STA 141A:

df_well_1 = preprocess_fromdf('Well_1.csv')
df_well_1.head(18)
```

[5]:

	Time	Voltage	Frequency	Impedance
0	2019-09-23 12:28:01.228297	10	10	60979.4
1	2019-09-23 12:28:01.650156	10	10	60443.1
2	2019-09-23 12:28:02.056392	10	10	61510.8
3	2019-09-23 12:28:02.556371	10	20	47860.5
4	2019-09-23 12:28:02.962614	10	20	49085.0
5	2019-09-23 12:28:03.368849	10	20	47314.9
6	2019-09-23 12:28:03.868838	10	30	44268.0
7	2019-09-23 12:28:04.275068	10	30	44333.0
8	2019-09-23 12:28:04.681307	10	30	45133.2

```
9   2019-09-23 12:28:05.196902       10          40    43039.3
10  2019-09-23 12:28:05.618770       10          40    42246.6
11  2019-09-23 12:28:06.071881       10          40    42288.1
12  2019-09-23 12:28:06.571866       10          50    42686.6
13  2019-09-23 12:28:06.978105       10          50    41411.2
14  2019-09-23 12:28:07.399964       10          50    40978.4
15  2019-09-23 12:28:07.915572       10          60    41864.4
16  2019-09-23 12:28:08.321809       10          60    40920.7
17  2019-09-23 12:28:08.774919       10          60    41186.5
```

[6]:
```
df_well_1.mean()
```

[6]:
```
Voltage         150.000000
Frequency       495.000000
Impedance     14894.940753
dtype: float64
```

[7]:
```
df_well_1['Impedance'].min()
```

[7]:
```
2769.83
```

[9]:
```
df_well_1['Impedance'].max()
```

[9]:
```
69248.5
```

[10]:
```
# Explore Impedance: min, max, median, mode, std
# to try and categorize variables.
df_well_1['Impedance'].mean()
```

[10]:
```
14894.940752990811
```

[11]:
```
df_well_1['Impedance'].std()
```

[11]:
```
14987.552044157746
```

[12]:
```
# Create outlier bound:

df_outlier_bound = df_well_1['Impedance'].mean() + df_well_1['Impedance'].std()
df_outlier_bound

# outlier bound is 29882.5.
```

[12]:
```
29882.49279714856
```

[13]:
```
# Check results using index values: Voltage = 90 MV, Frequency = 90 MV.
df_well_1_V90F90 =␣
 ↪df_well_1[df_well_1['Voltage']==90][df_well_1['Frequency']==90]
df_well_1_V90F90
```

```
/home/james/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:2:
UserWarning:

Boolean Series key will be reindexed to match DataFrame index.
```

[13]:
```
                          Time  Voltage  Frequency  Impedance
2376  2019-09-23 12:45:41.411679       90         90    39949.4
2377  2019-09-23 12:45:41.833541       90         90    39927.8
2378  2019-09-23 12:45:42.239784       90         90    39938.5
```

[14]:
```
# NEXT: Let's transform data from continuous to categorical variables!

df_well_1_V90F90.mean()
```

[14]:
```
Voltage          90.000000
Frequency        90.000000
Impedance    39938.566667
dtype: float64
```

4 SET UP BINS for categorical labeling:

[49]:
```
bin = [0, 14894.940752990811, 29882.49279714856, 80000]

# Use pd.cut function to attribute values into specific bins.

# How should we categorize our variables?
# "Low, Med, High" or "Low, High, Outlier"?
category = pd.cut(df_well_1['Impedance'], bin, labels = ['Low','High',␣
 ↪'Outlier'])
category = category.to_frame()
category.columns = ['Impedance']

# Concatenate age and its' bin

df_new = pd.concat([df_well_1, category], axis=1)
df_new
```

[49]:
```
                          Time  Voltage  Frequency Impedance Impedance
0     2019-09-23 12:28:01.228297       10         10  60979.40   Outlier
1     2019-09-23 12:28:01.650156       10         10  60443.10   Outlier
2     2019-09-23 12:28:02.056392       10         10  61510.80   Outlier
3     2019-09-23 12:28:02.556371       10         20  47860.50   Outlier
4     2019-09-23 12:28:02.962614       10         20  49085.00   Outlier
5     2019-09-23 12:28:03.368849       10         20  47314.90   Outlier
6     2019-09-23 12:28:03.868838       10         30  44268.00   Outlier
7     2019-09-23 12:28:04.275068       10         30  44333.00   Outlier
8     2019-09-23 12:28:04.681307       10         30  45133.20   Outlier
9     2019-09-23 12:28:05.196902       10         40  43039.30   Outlier
10    2019-09-23 12:28:05.618770       10         40  42246.60   Outlier
11    2019-09-23 12:28:06.071881       10         40  42288.10   Outlier
12    2019-09-23 12:28:06.571866       10         50  42686.60   Outlier
13    2019-09-23 12:28:06.978105       10         50  41411.20   Outlier
```

14	2019-09-23 12:28:07.399964	10	50	40978.40	Outlier
15	2019-09-23 12:28:07.915572	10	60	41864.40	Outlier
16	2019-09-23 12:28:08.321809	10	60	40920.70	Outlier
17	2019-09-23 12:28:08.774919	10	60	41186.50	Outlier
18	2019-09-23 12:28:09.274908	10	70	40670.30	Outlier
19	2019-09-23 12:28:09.696782	10	70	40805.90	Outlier
20	2019-09-23 12:28:10.102997	10	70	41168.50	Outlier
21	2019-09-23 12:28:10.602980	10	80	38783.80	Outlier
22	2019-09-23 12:28:11.009216	10	80	40336.80	Outlier
23	2019-09-23 12:28:11.446709	10	80	39110.10	Outlier
24	2019-09-23 12:28:11.962312	10	90	40043.10	Outlier
25	2019-09-23 12:28:12.384171	10	90	40472.00	Outlier
26	2019-09-23 12:28:12.790407	10	90	40129.30	Outlier
27	2019-09-23 12:28:13.306018	10	100	39680.50	Outlier
28	2019-09-23 12:28:13.727878	10	100	39463.70	Outlier
29	2019-09-23 12:28:14.149739	10	100	38786.10	Outlier
...	...	...	...	...	...
8496	2019-09-23 13:29:02.512415	290	890	3019.04	Low
8497	2019-09-23 13:29:02.887403	290	890	3019.43	Low
8498	2019-09-23 13:29:03.278014	290	890	3019.24	Low
8499	2019-09-23 13:29:03.746753	290	900	3018.65	Low
8500	2019-09-23 13:29:04.121737	290	900	3019.97	Low
8501	2019-09-23 13:29:04.496725	290	900	3019.64	Low
8502	2019-09-23 13:29:04.996709	290	910	3019.70	Low
8503	2019-09-23 13:29:05.387328	290	910	3019.27	Low
8504	2019-09-23 13:29:05.762323	290	910	3019.68	Low
8505	2019-09-23 13:29:06.215418	290	920	3019.56	Low
8506	2019-09-23 13:29:06.590408	290	920	3019.03	Low
8507	2019-09-23 13:29:06.965392	290	920	3019.35	Low
8508	2019-09-23 13:29:07.418497	290	930	3019.22	Low
8509	2019-09-23 13:29:07.777861	290	930	3019.81	Low
8510	2019-09-23 13:29:08.152857	290	930	3019.30	Low
8511	2019-09-23 13:29:08.605964	290	940	3019.58	Low
8512	2019-09-23 13:29:08.980950	290	940	3018.83	Low
8513	2019-09-23 13:29:09.355940	290	940	3019.23	Low
8514	2019-09-23 13:29:09.809050	290	950	3019.42	Low
8515	2019-09-23 13:29:10.184036	290	950	3019.63	Low
8516	2019-09-23 13:29:10.559019	290	950	3019.60	Low
8517	2019-09-23 13:29:11.012144	290	960	3018.79	Low
8518	2019-09-23 13:29:11.387139	290	960	3019.53	Low
8519	2019-09-23 13:29:11.762108	290	960	3019.25	Low
8520	2019-09-23 13:29:12.215219	290	970	3019.91	Low
8521	2019-09-23 13:29:12.590201	290	970	3019.47	Low
8522	2019-09-23 13:29:12.965189	290	970	3020.00	Low
8523	2019-09-23 13:29:13.433926	290	980	2833.69	Low
8524	2019-09-23 13:29:13.840164	290	980	2834.21	Low
8525	2019-09-23 13:29:14.230776	290	980	2833.65	Low

[8526 rows x 5 columns]

5 TRY function to create new columns for categorical label:

[51]:
```
def df_column_uniquify(df):
    df_columns = df.columns
    new_columns = []
    for item in df_columns:
        counter = 0
        newitem = item
        while newitem in new_columns:
            counter += 1
            newitem = "{}_{}".format(item, counter)
        new_columns.append(newitem)
    df.columns = new_columns
    return df
```

[52]:
```
df_new = df_column_uniquify(df_new)
df_new.columns=['Time','Voltage','Frequency','Impedance Value','Low/High/
 ↪Outlier Imp. Label']
df_new.head(300)
```

[52]:

	Time	Voltage	Frequency	Impedance Value	\
0	2019-09-23 12:28:01.228297	10	10	60979.40	
1	2019-09-23 12:28:01.650156	10	10	60443.10	
2	2019-09-23 12:28:02.056392	10	10	61510.80	
3	2019-09-23 12:28:02.556371	10	20	47860.50	
4	2019-09-23 12:28:02.962614	10	20	49085.00	
5	2019-09-23 12:28:03.368849	10	20	47314.90	
6	2019-09-23 12:28:03.868838	10	30	44268.00	
7	2019-09-23 12:28:04.275068	10	30	44333.00	
8	2019-09-23 12:28:04.681307	10	30	45133.20	
9	2019-09-23 12:28:05.196902	10	40	43039.30	
10	2019-09-23 12:28:05.618770	10	40	42246.60	
11	2019-09-23 12:28:06.071881	10	40	42288.10	
12	2019-09-23 12:28:06.571866	10	50	42686.60	
13	2019-09-23 12:28:06.978105	10	50	41411.20	
14	2019-09-23 12:28:07.399964	10	50	40978.40	
15	2019-09-23 12:28:07.915572	10	60	41864.40	
16	2019-09-23 12:28:08.321809	10	60	40920.70	
17	2019-09-23 12:28:08.774919	10	60	41186.50	
18	2019-09-23 12:28:09.274908	10	70	40670.30	
19	2019-09-23 12:28:09.696782	10	70	40805.90	
20	2019-09-23 12:28:10.102997	10	70	41168.50	
21	2019-09-23 12:28:10.602980	10	80	38783.80	
22	2019-09-23 12:28:11.009216	10	80	40336.80	

23	2019-09-23	12:28:11.446709	10	80	39110.10
24	2019-09-23	12:28:11.962312	10	90	40043.10
25	2019-09-23	12:28:12.384171	10	90	40472.00
26	2019-09-23	12:28:12.790407	10	90	40129.30
27	2019-09-23	12:28:13.306018	10	100	39680.50
28	2019-09-23	12:28:13.727878	10	100	39463.70
29	2019-09-23	12:28:14.149739	10	100	38786.10
..		...	...	...	...
270	2019-09-23	12:30:01.036796	10	910	2972.50
271	2019-09-23	12:30:01.443040	10	910	2949.03
272	2019-09-23	12:30:01.849268	10	910	2966.08
273	2019-09-23	12:30:02.349255	10	920	2962.77
274	2019-09-23	12:30:02.755491	10	920	2952.78
275	2019-09-23	12:30:03.177350	10	920	2974.26
276	2019-09-23	12:30:03.692960	10	930	2954.87
277	2019-09-23	12:30:04.114829	10	930	2961.25
278	2019-09-23	12:30:04.521076	10	930	2964.70
279	2019-09-23	12:30:05.021040	10	940	2952.68
280	2019-09-23	12:30:05.427278	10	940	2964.37
281	2019-09-23	12:30:05.849134	10	940	2976.61
282	2019-09-23	12:30:06.364742	10	950	2959.72
283	2019-09-23	12:30:06.786609	10	950	2955.97
284	2019-09-23	12:30:07.192842	10	950	2965.95
285	2019-09-23	12:30:07.692827	10	960	2951.59
286	2019-09-23	12:30:08.099061	10	960	2971.93
287	2019-09-23	12:30:08.520921	10	960	2986.69
288	2019-09-23	12:30:09.036532	10	970	2965.47
289	2019-09-23	12:30:09.442761	10	970	2948.09
290	2019-09-23	12:30:09.849018	10	970	2962.54
291	2019-09-23	12:30:10.317737	10	980	2788.37
292	2019-09-23	12:30:10.708350	10	980	2786.32
293	2019-09-23	12:30:11.083336	10	980	2791.25
294	2019-09-23	12:30:12.473916	20	10	64740.70
295	2019-09-23	12:30:12.880153	20	10	65455.30
296	2019-09-23	12:30:13.286387	20	10	64226.30
297	2019-09-23	12:30:13.786375	20	20	49743.40
298	2019-09-23	12:30:14.208234	20	20	49802.60
299	2019-09-23	12:30:14.630097	20	20	50145.70

	Low/High/Outlier Imp. Label
0	Outlier
1	Outlier
2	Outlier
3	Outlier
4	Outlier
5	Outlier
6	Outlier

```
7                    Outlier
8                    Outlier
9                    Outlier
10                   Outlier
11                   Outlier
12                   Outlier
13                   Outlier
14                   Outlier
15                   Outlier
16                   Outlier
17                   Outlier
18                   Outlier
19                   Outlier
20                   Outlier
21                   Outlier
22                   Outlier
23                   Outlier
24                   Outlier
25                   Outlier
26                   Outlier
27                   Outlier
28                   Outlier
29                   Outlier
..                       ...
270                      Low
271                      Low
272                      Low
273                      Low
274                      Low
275                      Low
276                      Low
277                      Low
278                      Low
279                      Low
280                      Low
281                      Low
282                      Low
283                      Low
284                      Low
285                      Low
286                      Low
287                      Low
288                      Low
289                      Low
290                      Low
291                      Low
292                      Low
```

```
293                        Low
294                    Outlier
295                    Outlier
296                    Outlier
297                    Outlier
298                    Outlier
299                    Outlier

[300 rows x 5 columns]
```

[53]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier'].head(18)
```

[53]:
```
                          Time  Voltage  Frequency  Impedance Value  \
0   2019-09-23 12:28:01.228297       10         10          60979.4
1   2019-09-23 12:28:01.650156       10         10          60443.1
2   2019-09-23 12:28:02.056392       10         10          61510.8
3   2019-09-23 12:28:02.556371       10         20          47860.5
4   2019-09-23 12:28:02.962614       10         20          49085.0
5   2019-09-23 12:28:03.368849       10         20          47314.9
6   2019-09-23 12:28:03.868838       10         30          44268.0
7   2019-09-23 12:28:04.275068       10         30          44333.0
8   2019-09-23 12:28:04.681307       10         30          45133.2
9   2019-09-23 12:28:05.196902       10         40          43039.3
10  2019-09-23 12:28:05.618770       10         40          42246.6
11  2019-09-23 12:28:06.071881       10         40          42288.1
12  2019-09-23 12:28:06.571866       10         50          42686.6
13  2019-09-23 12:28:06.978105       10         50          41411.2
14  2019-09-23 12:28:07.399964       10         50          40978.4
15  2019-09-23 12:28:07.915572       10         60          41864.4
16  2019-09-23 12:28:08.321809       10         60          40920.7
17  2019-09-23 12:28:08.774919       10         60          41186.5

   Low/High/Outlier Imp. Label
0                      Outlier
1                      Outlier
2                      Outlier
3                      Outlier
4                      Outlier
5                      Outlier
6                      Outlier
7                      Outlier
8                      Outlier
9                      Outlier
10                     Outlier
11                     Outlier
12                     Outlier
13                     Outlier
14                     Outlier
```

```
15                    Outlier
16                    Outlier
17                    Outlier
```

[54]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].max()
```

[54]:
```
220
```

[20]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].min()
```

[20]:
```
10
```

[21]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].mean()
```

[21]:
```
115.0
```

[22]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].std()
```

[22]:
```
63.45946757648916
```

[23]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Voltage"].max()
```

[23]:
```
290
```

[38]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Voltage"].min()
```

[38]:
```
10
```

6 Use new bins for new range of values for categorical labeling.

[30]:
```
# NEW BINS:

# ADD VALUE TO GET MEDIAN:(46-19)/2 = 13.5
# LOW: 0
# NEW MEDIAN: 19K + 13.5K = 32.5
# NEW HIGH: 46K
# OUTLIERS: ABOVE 46K

new_bin = [0, 32500, 46000, 80000]

# Use pd.cut function to attribute values into specific bins.

# How should we categorize our variables?
# "Low, Med, High" or "Low, High, Outlier"?
category = pd.cut(df_well_1['Impedance'], new_bin, labels = ['Low','High',␣
 ↪'Outlier'])
category = category.to_frame()
category.columns = ['Impedance']

# Concatenate age and its' bin

df_new = pd.concat([df_well_1, category], axis=1)
```

```
df_new.head(18)
```

[30]:

	Time	Voltage	Frequency	Impedance	Impedance
0	2019-09-23 12:28:01.228297	10	10	60979.4	Outlier
1	2019-09-23 12:28:01.650156	10	10	60443.1	Outlier
2	2019-09-23 12:28:02.056392	10	10	61510.8	Outlier
3	2019-09-23 12:28:02.556371	10	20	47860.5	Outlier
4	2019-09-23 12:28:02.962614	10	20	49085.0	Outlier
5	2019-09-23 12:28:03.368849	10	20	47314.9	Outlier
6	2019-09-23 12:28:03.868838	10	30	44268.0	High
7	2019-09-23 12:28:04.275068	10	30	44333.0	High
8	2019-09-23 12:28:04.681307	10	30	45133.2	High
9	2019-09-23 12:28:05.196902	10	40	43039.3	High
10	2019-09-23 12:28:05.618770	10	40	42246.6	High
11	2019-09-23 12:28:06.071881	10	40	42288.1	High
12	2019-09-23 12:28:06.571866	10	50	42686.6	High
13	2019-09-23 12:28:06.978105	10	50	41411.2	High
14	2019-09-23 12:28:07.399964	10	50	40978.4	High
15	2019-09-23 12:28:07.915572	10	60	41864.4	High
16	2019-09-23 12:28:08.321809	10	60	40920.7	High
17	2019-09-23 12:28:08.774919	10	60	41186.5	High

[31]:
```
df_new = df_column_uniquify(df_new)
df_new.columns=['Time','Voltage','Frequency','Impedance Value','Low/High/
 ↪Outlier Imp. Label']
df_new.head(300)
```

[31]:

	Time	Voltage	Frequency	Impedance Value	\
0	2019-09-23 12:28:01.228297	10	10	60979.40	
1	2019-09-23 12:28:01.650156	10	10	60443.10	
2	2019-09-23 12:28:02.056392	10	10	61510.80	
3	2019-09-23 12:28:02.556371	10	20	47860.50	
4	2019-09-23 12:28:02.962614	10	20	49085.00	
5	2019-09-23 12:28:03.368849	10	20	47314.90	
6	2019-09-23 12:28:03.868838	10	30	44268.00	
7	2019-09-23 12:28:04.275068	10	30	44333.00	
8	2019-09-23 12:28:04.681307	10	30	45133.20	
9	2019-09-23 12:28:05.196902	10	40	43039.30	
10	2019-09-23 12:28:05.618770	10	40	42246.60	
11	2019-09-23 12:28:06.071881	10	40	42288.10	
12	2019-09-23 12:28:06.571866	10	50	42686.60	
13	2019-09-23 12:28:06.978105	10	50	41411.20	
14	2019-09-23 12:28:07.399964	10	50	40978.40	
15	2019-09-23 12:28:07.915572	10	60	41864.40	
16	2019-09-23 12:28:08.321809	10	60	40920.70	
17	2019-09-23 12:28:08.774919	10	60	41186.50	
18	2019-09-23 12:28:09.274908	10	70	40670.30	
19	2019-09-23 12:28:09.696782	10	70	40805.90	

20	2019-09-23	12:28:10.102997	10	70	41168.50
21	2019-09-23	12:28:10.602980	10	80	38783.80
22	2019-09-23	12:28:11.009216	10	80	40336.80
23	2019-09-23	12:28:11.446709	10	80	39110.10
24	2019-09-23	12:28:11.962312	10	90	40043.10
25	2019-09-23	12:28:12.384171	10	90	40472.00
26	2019-09-23	12:28:12.790407	10	90	40129.30
27	2019-09-23	12:28:13.306018	10	100	39680.50
28	2019-09-23	12:28:13.727878	10	100	39463.70
29	2019-09-23	12:28:14.149739	10	100	38786.10
..		...	...	...	...
270	2019-09-23	12:30:01.036796	10	910	2972.50
271	2019-09-23	12:30:01.443040	10	910	2949.03
272	2019-09-23	12:30:01.849268	10	910	2966.08
273	2019-09-23	12:30:02.349255	10	920	2962.77
274	2019-09-23	12:30:02.755491	10	920	2952.78
275	2019-09-23	12:30:03.177350	10	920	2974.26
276	2019-09-23	12:30:03.692960	10	930	2954.87
277	2019-09-23	12:30:04.114829	10	930	2961.25
278	2019-09-23	12:30:04.521076	10	930	2964.70
279	2019-09-23	12:30:05.021040	10	940	2952.68
280	2019-09-23	12:30:05.427278	10	940	2964.37
281	2019-09-23	12:30:05.849134	10	940	2976.61
282	2019-09-23	12:30:06.364742	10	950	2959.72
283	2019-09-23	12:30:06.786609	10	950	2955.97
284	2019-09-23	12:30:07.192842	10	950	2965.95
285	2019-09-23	12:30:07.692827	10	960	2951.59
286	2019-09-23	12:30:08.099061	10	960	2971.93
287	2019-09-23	12:30:08.520921	10	960	2986.69
288	2019-09-23	12:30:09.036532	10	970	2965.47
289	2019-09-23	12:30:09.442761	10	970	2948.09
290	2019-09-23	12:30:09.849018	10	970	2962.54
291	2019-09-23	12:30:10.317737	10	980	2788.37
292	2019-09-23	12:30:10.708350	10	980	2786.32
293	2019-09-23	12:30:11.083336	10	980	2791.25
294	2019-09-23	12:30:12.473916	20	10	64740.70
295	2019-09-23	12:30:12.880153	20	10	65455.30
296	2019-09-23	12:30:13.286387	20	10	64226.30
297	2019-09-23	12:30:13.786375	20	20	49743.40
298	2019-09-23	12:30:14.208234	20	20	49802.60
299	2019-09-23	12:30:14.630097	20	20	50145.70

	Low/High/Outlier Imp. Label
0	Outlier
1	Outlier
2	Outlier
3	Outlier

```
4                       Outlier
5                       Outlier
6                          High
7                          High
8                          High
9                          High
10                         High
11                         High
12                         High
13                         High
14                         High
15                         High
16                         High
17                         High
18                         High
19                         High
20                         High
21                         High
22                         High
23                         High
24                         High
25                         High
26                         High
27                         High
28                         High
29                         High
..                          ...
270                         Low
271                         Low
272                         Low
273                         Low
274                         Low
275                         Low
276                         Low
277                         Low
278                         Low
279                         Low
280                         Low
281                         Low
282                         Low
283                         Low
284                         Low
285                         Low
286                         Low
287                         Low
288                         Low
289                         Low
```

```
290                          Low
291                          Low
292                          Low
293                          Low
294                      Outlier
295                      Outlier
296                      Outlier
297                      Outlier
298                      Outlier
299                      Outlier

[300 rows x 5 columns]
```

[ ]:

[32]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].max()
```

[32]:
```
40
```

[33]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].min()
```

[33]:
```
10
```

[34]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].mean()
```

[34]:
```
17.740384615384617
```

[35]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Frequency"].std()
```

[35]:
```
7.875228934401255
```

[36]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Voltage"].max()
```

[36]:
```
290
```

[37]:
```
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier']["Voltage"].min()
```

[37]:
```
10
```

[39]:
```
# Total # of Impedance values: 8526
df_new.count()
```

[39]:
```
Time                          8526
Voltage                       8526
Frequency                     8526
Impedance Value               8526
Low/High/Outlier Imp. Label   8526
dtype: int64
```

[42]:
```
# Use .count() to find total # of "High" values.
df_new[df_new['Low/High/Outlier Imp. Label']=='High'].count() # Total: 1457
```

[42]:
```
Time                          1457
Voltage                       1457
Frequency                     1457
```

```
Impedance Value                1457
Low/High/Outlier Imp. Label    1457
dtype: int64
```

[41]:
```
# Use .count() to find total # of "Low" values.
df_new[df_new['Low/High/Outlier Imp. Label']=='Low'].count() # Total: 6861
```

[41]:
```
Time                           6861
Voltage                        6861
Frequency                      6861
Impedance Value                6861
Low/High/Outlier Imp. Label    6861
dtype: int64
```

[43]:
```
# Use .count() to find total number of "Outlier" values.
df_new[df_new['Low/High/Outlier Imp. Label']=='Outlier'].count() # Total: 208
```

[43]:
```
Time                           208
Voltage                        208
Frequency                      208
Impedance Value                208
Low/High/Outlier Imp. Label    208
dtype: int64
```

7 Using Classification help determine percentage of values inside low, high and outlier ranges:

[44]:
```
# High Imp. Value Pct: 17.09 %
1457/8526 * 100
```

[44]:
```
17.08890452732817
```

[45]:
```
# Low Imp. Value Pct: 80.4%
6861/8526 * 100
```

[45]:
```
80.47149894440535
```

[46]:
```
# Outlier Imp. Value Pct: 2.44%
208/8526 * 100
```

[46]:
```
2.439596528266479
```

8 Check results using index values: Voltage = 90 MV, Frequency = 90 MV.

[47]:
```
df_new_V90F90 = df_new[df_new['Voltage']==90][df_new['Frequency']==90]
df_new_V90F90
```

```
/home/james/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:2:
UserWarning:

Boolean Series key will be reindexed to match DataFrame index.
```

[47]:

	Time	Voltage	Frequency	Impedance Value	\
2376	2019-09-23 12:45:41.411679	90	90	39949.4	
2377	2019-09-23 12:45:41.833541	90	90	39927.8	
2378	2019-09-23 12:45:42.239784	90	90	39938.5	

	Low/High/Outlier Imp. Label
2376	High
2377	High
2378	High

[ ]: