Works related to Data Visualization, Data Analysis/Mining/Scraping

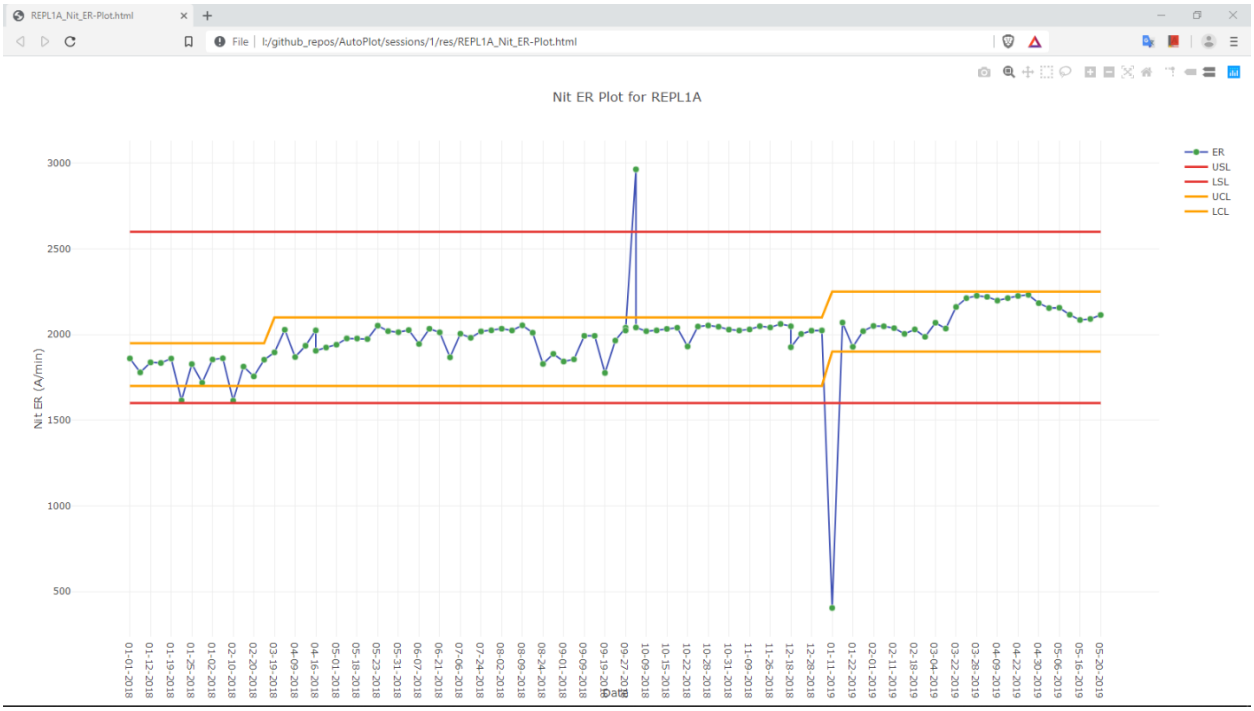


Fig. 1: Line Plot with single main trace

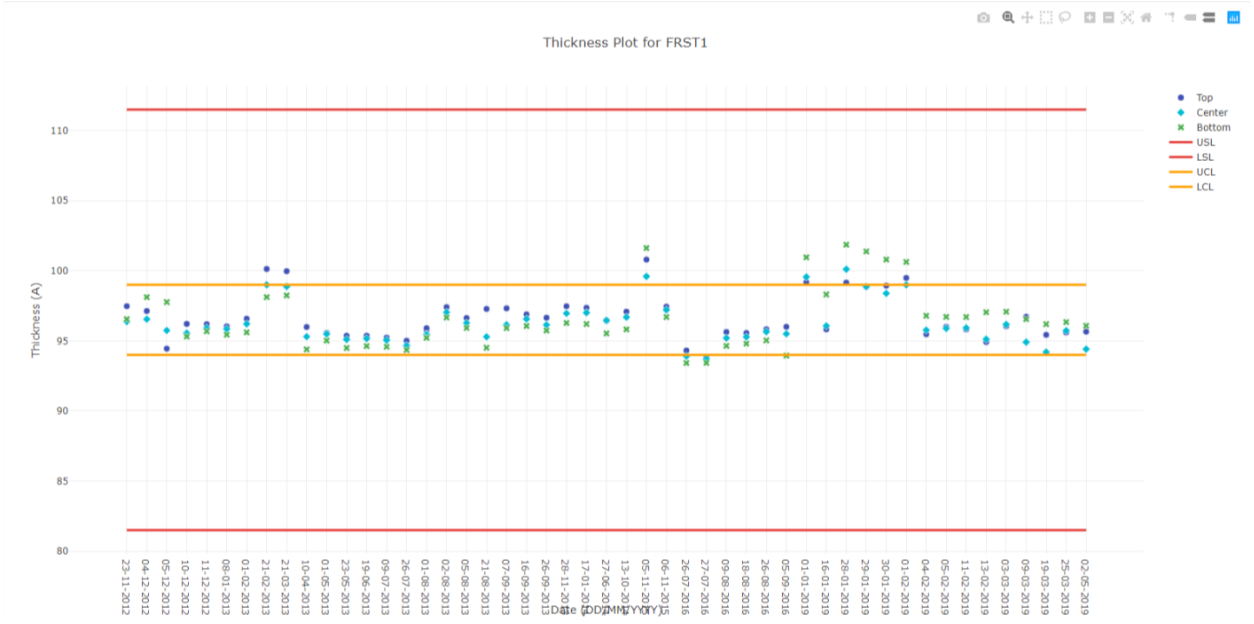


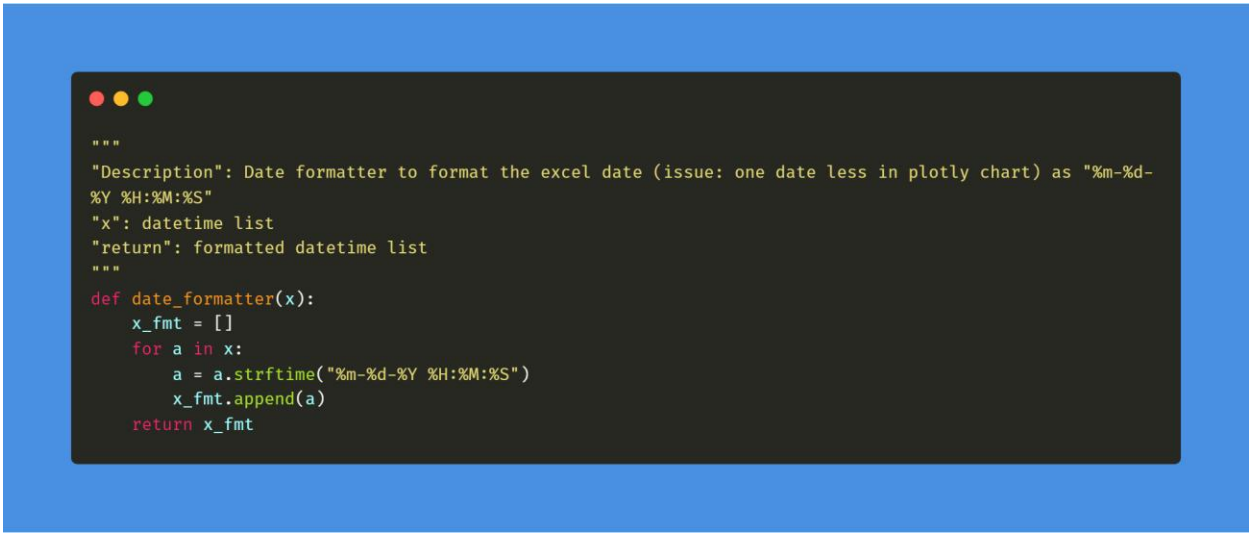
Fig. 2: Scatter Plot with multiple main traces

```
"""
"Description": This function plots CP Chart with traces v/s
Data
"draw_plotly_resp1b_cp_plot": Draw Plotly's Plot for RESP1B CP
"x": Date (x-axis) for CP Chart
"y1": Delta-CP (y-axis) for CP Chart
"y2": USL (y-axis) for CP Chart
# "y3": UCL (y-axis) for CP Chart
"""
def draw_plotly_resp1b_cp_plot(x, y1, y2, remarks):
    trace1 = go.Scatter(
            x = x,
            y = y1,
            name = 'delta-CP',
            mode = 'lines+markers',
            line = dict(
                    color = line_color,
                    width = 2),
            marker = dict(
                    color = marker_color,
                    size = 8,
                    line = dict(
                        color = marker_border_color,
                        width = 0.5),
                    ),
            text = remarks
    )

    trace2 = go.Scatter(
            x = x,
            y = y2,
            name = 'USL',
            mode = 'lines',
            line = dict(
                    color = sl_color,
                    width = 3)
    )

    trace3 = go.Scatter(
            x = x,
            y = y3,
            name = 'UCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    data = [trace1, trace2, trace3]
    layout = dict(
            title = cp_plot_title,
            xaxis = dict(title= cp_plot_xlabel),
            yaxis = dict(title= cp_plot_ylabel)
        )
    fig = dict(data= data, layout= layout)
    py.offline.plot(fig, filename= cp_plot_html_file)
```

Fig. 3: Python Script for Fig. 1 shown above

```
"""
"Description": This function plots ER Chart with traces v/s Date.
"draw_plotly_resp1b_er_barc_plot": Draw Plotly's Plot for RESP1B BARC
ER": Date (x-axis) for ER Chart
"y1": ER (y-axis) for ER Chart
"y2": USL (y-axis) for ER Chart
"y3": LSL (y-axis) for ER Chart
"y4": UCL (y-axis) for ER Chart
"y5": LCL (y-axis) for ER Chart
"""
def draw_plotly_resp1b_er_barc_plot(x, y1, y2, y3, y4, y5, remarks):
    trace1 = go.Scatter(
            x = x,
            y = y1,
            name = 'ER',
            mode = 'lines+markers',
            line = dict(
                    color = line_color,
                    width = 2),
            marker = dict(
                    color = marker_color,
                    size = 8,
                    line = dict(
                        color = marker_border_color,
                        width = 0.5),
                    ),
            text = remarks
    )

    trace2 = go.Scatter(
            x = x,
            y = y2,
            name = 'USL',
            mode = 'lines',
            line = dict(
                    color = sl_color,
                    width = 3)
    )

    trace3 = go.Scatter(
            x = x,
            y = y3,
            name = 'LSL',
            mode = 'lines',
            line = dict(
                    color = sl_color,
                    width = 3)
    )

    trace4 = go.Scatter(
            x = x,
            y = y4,
            name = 'UCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    trace5 = go.Scatter(
            x = x,
            y = y5,
            name = 'LCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    data = [trace1, trace2, trace3, trace4, trace5]
    layout = dict(
            title = er_barc_plot_title,
            xaxis = dict(title= er_barc_plot_xlabel),
            yaxis = dict(title= er_barc_plot_ylabel)
        )
    fig = dict(data= data, layout= layout)
    py.offline.plot(fig, filename= er_barc_plot_html_file)
```

Fig. 4: Python script used for Line & Scatter Plot



```
"""
"Description": Date formatter to format the excel date (issue: one date less in plotly chart) as "%m-%d-
%Y %H:%M:%S"
"x": datetime list
"return": formatted datetime list
"""
def date_formatter(x):
    x_fmt = []
    for a in x:
        a = a.strftime("%m-%d-%Y %H:%M:%S")
        x_fmt.append(a)
    return x_fmt
```

Fig. 5: Automatic Date formatter function for formatting ‘Date’ column into a custom one

Works related to Telegram Bot

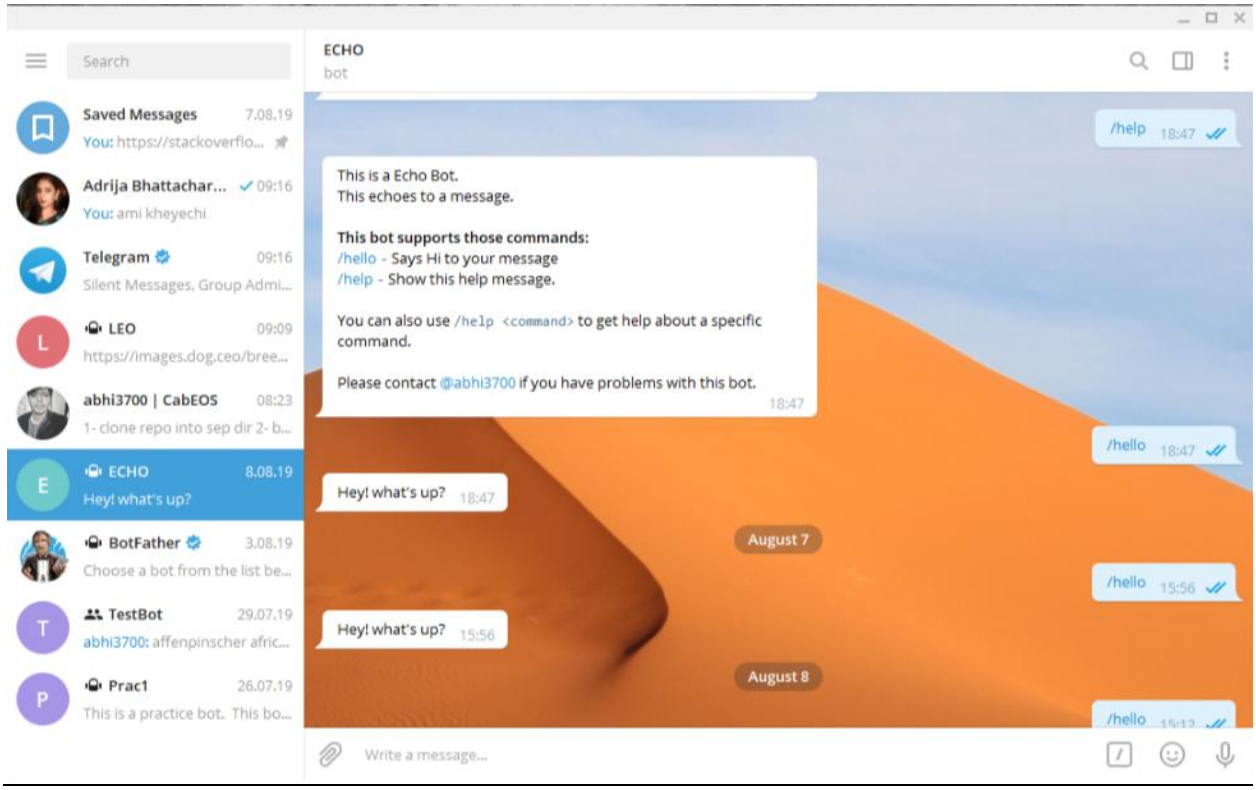


Fig. 6: Echo Bot – Echoes "Hey! What's up?" for a user command – '/hello'. Also can be used for FAQs application for creating smooth interaction with clients/customers

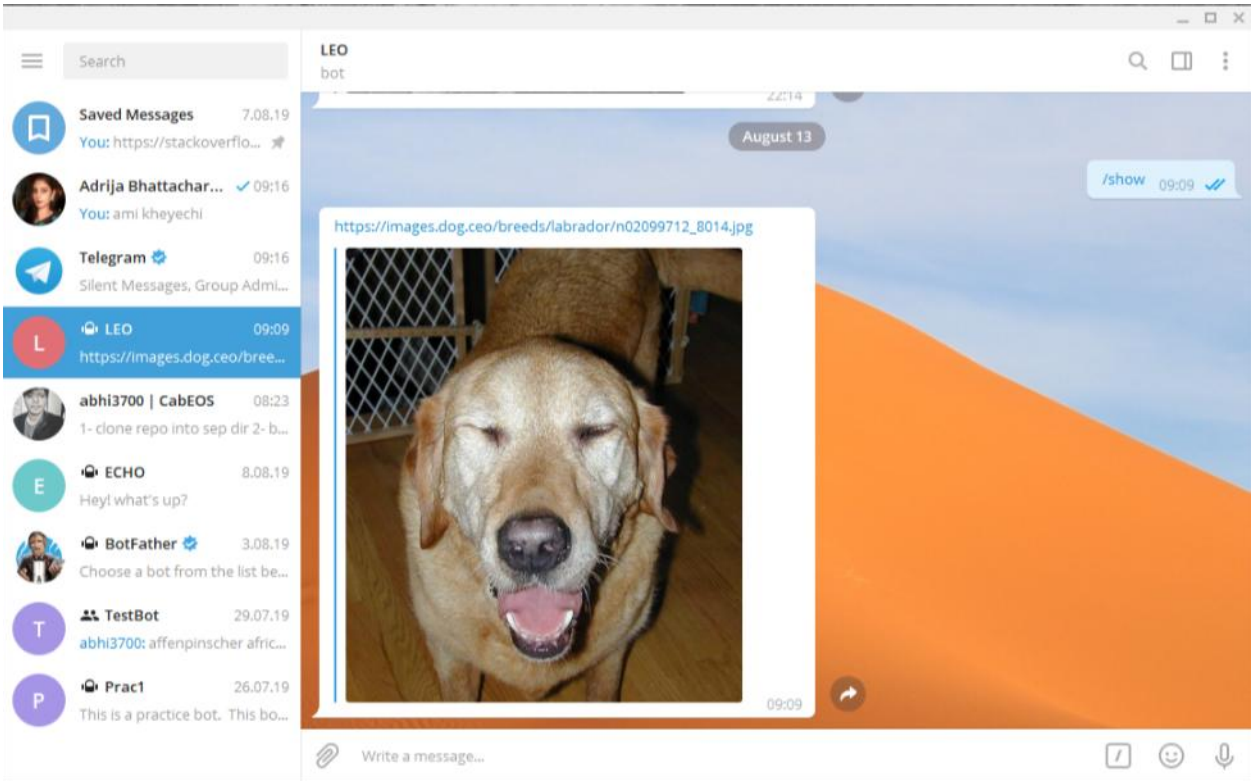


Fig. 7: Dog Bot – Shows random images based on 'breed' asked by a user