Engineering for everyday world

AceEngineer

DATA VISUALIZATIONS

Development Document

9th August 2018

19-Dec-2014	01	New Issue	VA	-	-
DATE	**REV**	**DESCRIPTION**	**ORIG**	**CHK**	**APPR**

DOCUMENT CONTROL NO	**Project**	**Type**	**Area**	**Client**	**-**	**-**	**Sequence**	**Revision**
	0026	**PY**	**-**	**-**	**-**	**-**	**0001**	**01**

Revision History:

REV	DATE	DESCRIPTION	ORIG	CHK	APPR
01	9th Dec 2014	Manual for python coding	MP	VA	VA

Change Log

REV	SECTION	CHANGE DESCRIPTION

Document Holds

Hold	DESCRIPTION
HOLD 01	

CONTENTS

1 INTRODUCTION

Data visualization is key to grasp large amount of data and infer decisions.

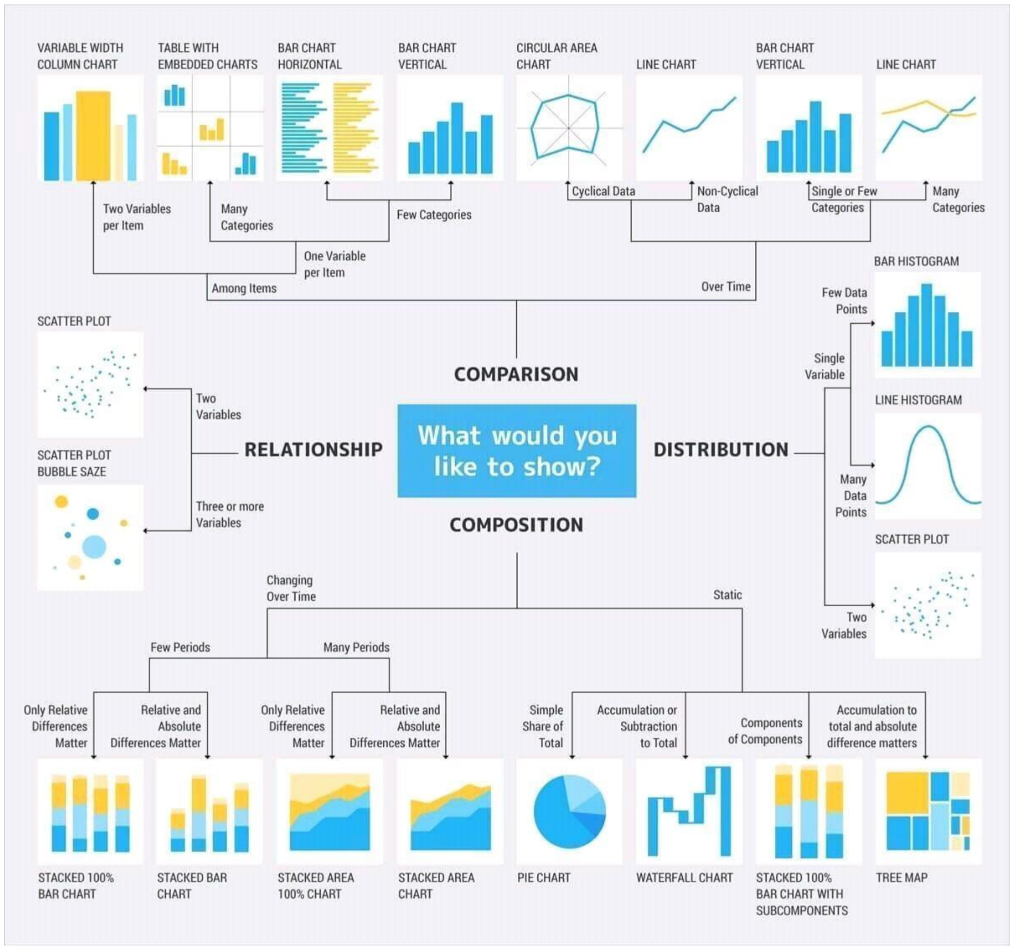
1.1 Highlight Features

The key Python features worth noting are:

1.2 High Level References

http://flowingdata.com/2018/02/23/beginners-guide-to-visualization-literacy/
http://dataviz-literacy.wmflabs.org/ Introduction to Visualization

https://go.anaconda.com/confirmation-taming-python-visualization-jungle/?first_name=Vamsee&last_name=Achanta&company=Ind.%20Consultant&work_email=vamsee.achanta%40aceengineer.com&job_title=Ind.%20Consultant&phone_number=7133069029&referrer=NoReferrer&utm_source=&utm_campaign=&utm_name=&utm_medium=
https://github.com/plotly/dash Dash. Reactive or interactive web plots in Python
https://plot.ly/dash/getting-started



2 SUMMARY

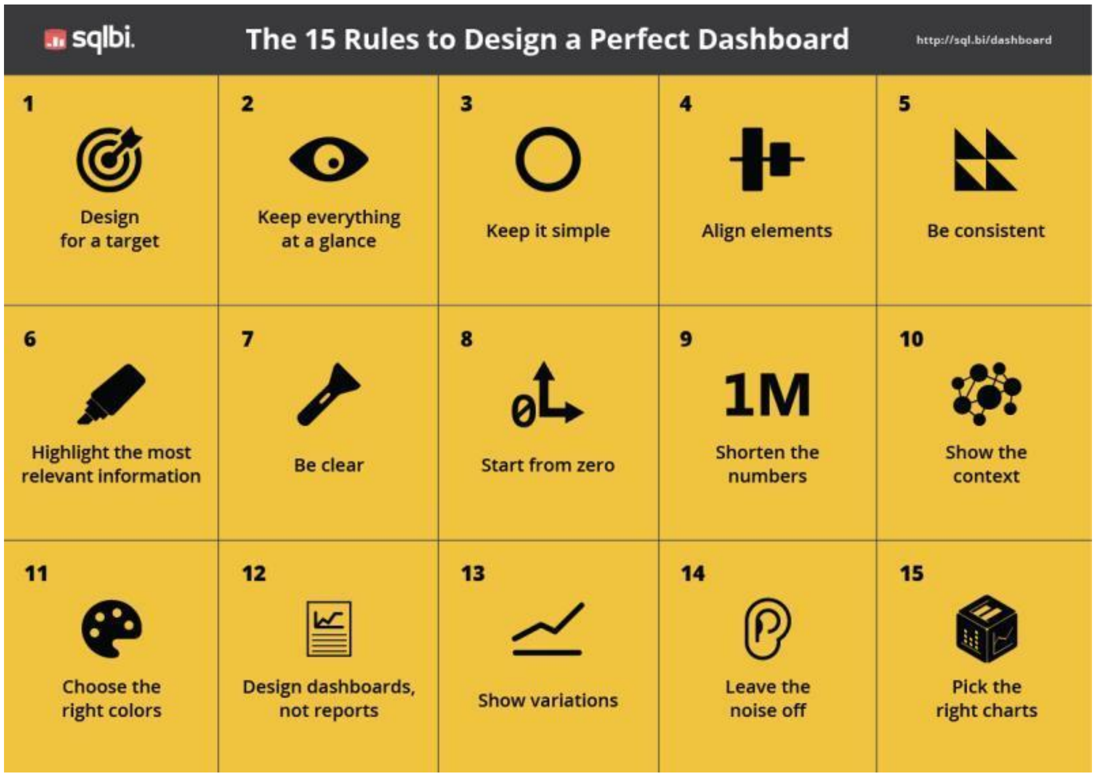
Python Fatigue analysis package

2.1 Improvements

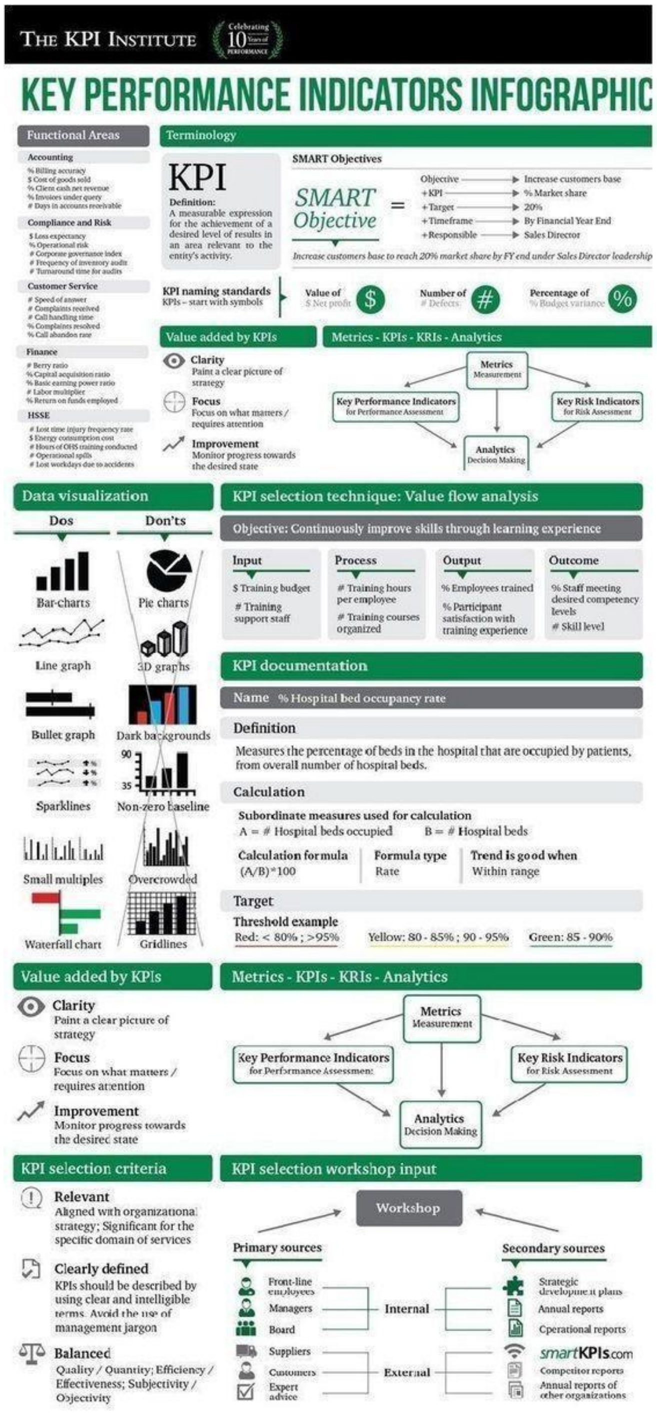
2.1.1 Close out Actions Reporting

S.No	Name	Description	Scope	Status
1	Plot_settings to plt_single	Change plot_settings to plot_single in yml file and code.		
2				
3				
4				
5				
6				
7				
8				
9				
4				

2.2 Dashboards



2.3 Infographics


THE KPI INSTITUTE
Celebrating 10 Years of PERFORMANCE
KEY PERFORMANCE INDICATORS INFOGRAPHIC
Functional Areas
Accounting
Compliance and Risk
Customer Service
Finance
HSSE
Terminology
KPI
Definition: A measurable expression for the achievement of a desired level of results in an area relevant to the entity's activity.
SMART Objectives
SMART Objective =
Objective → Increase customers base
+KPI → % Market share
+Target → 20%
+Timeframe → By Financial Year End
+Responsible → Sales Director
Increase customers base to reach 20% market share by FY end under Sales Director leadership
KPI naming standards
KPIs – start with symbols
Value of $
Number of #
Percentage of %
Value added by KPIs
Metrics - KPIs - KRIs - Analytics
Data visualization
Dos
Don'ts
Bar-charts
Pie charts
Line graph
3D graphs
Bullet graph
Dark backgrounds
Sparklines
Non-zero baseline
Small multiples
Overcrowded
Waterfall chart
Gridlines
KPI selection technique: Value flow analysis
Objective: Continuously improve skills through learning experience
Input
$ Training budget
# Training support staff
Process
# Training hours per employee
# Training courses organized
Output
% Employees trained
% Participant satisfaction with training experience
Outcome
% Staff meeting desired competency levels
# Skill level
KPI documentation
Name % Hospital bed occupancy rate
Definition
Measures the percentage of beds in the hospital that are occupied by patients, from overall number of hospital beds.
Calculation
Subordinate measures used for calculation
A = # Hospital beds occupied
B = # Hospital beds
Calculation formula (A/B)*100
Formula type Rate
Trend is good when Within range
Target
Threshold example
Red: < 80% ; >95%
Yellow: 80 - 85% ; 90 - 95%
Green: 85 - 90%
Value added by KPIs
Clarity
Paint a clear picture of strategy
Focus
Focus on what matters / requires attention
Improvement
Monitor progress towards the desired state
Metrics - KPIs - KRIs - Analytics
Metrics
Measurement
Key Performance Indicators
for Performance Assessment
Key Risk Indicators
for Risk Assessment
Analytics
Decision Making
KPI selection criteria
Relevant
Aligned with organizational strategy; Significant for the specific domain of services
Clearly defined
KPIs should be described by using clear and intelligible terms. Avoid the use of management jargon
Balanced
Quality / Quantity; Efficiency / Effectiveness; Subjectivity / Objectivity
KPI selection workshop input
Workshop
Primary sources
Front-line employees
Managers
Board
Suppliers
Customers
Expert advice
Internal
External
Secondary sources
Strategic development plans
Annual reports
Operational reports
smartKPIs.com
Competitor reports
Annual reports of other organizations

2.4 Tips

Running multiple batch files:

3 GENERAL PRINCIPLES

3.1 Chart types

- Single chart – Type
    - Line
    - Scatter
    - Bar/column
    - Polar
    - Gantt chart
    - Timeline
- Single chart – Simple vs. Compound
    - Simple chart (only line, only bars etc.)
        - See Figure 3.2
    - Compound chart (only line, only bars etc.)
        - See examples in Figure 3.1
- Multiple charts
    - See Figure 3.3
- Embedded charts
    - See examples in Figure 3.4

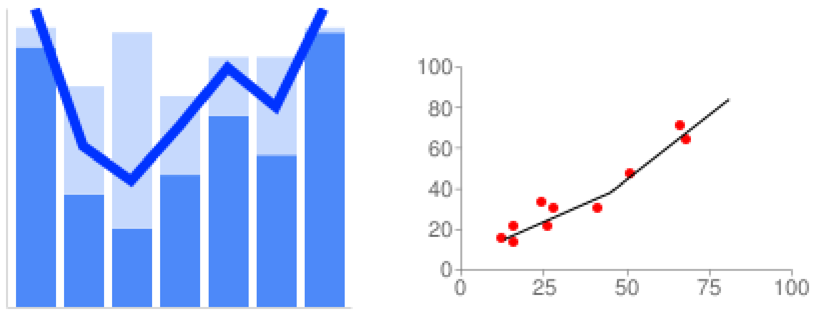


Figure 3.1 – Compound Charts

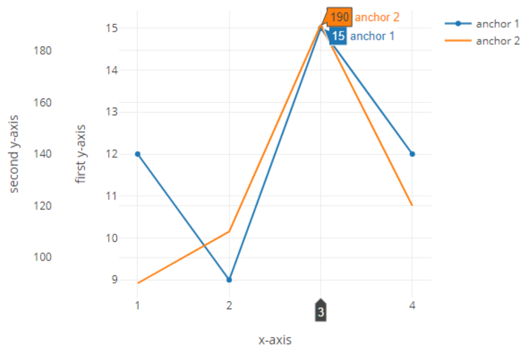


Figure 3.2 – Single Chart

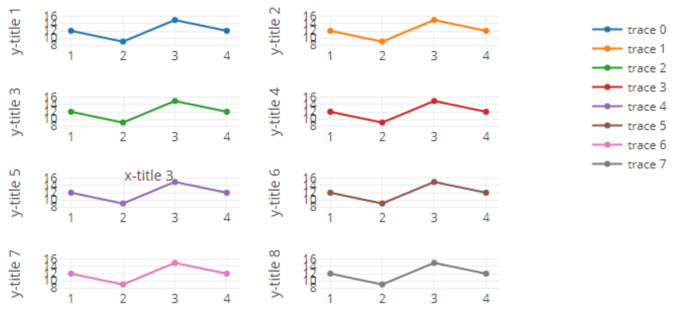


Figure 3.3 – Multiple Chart

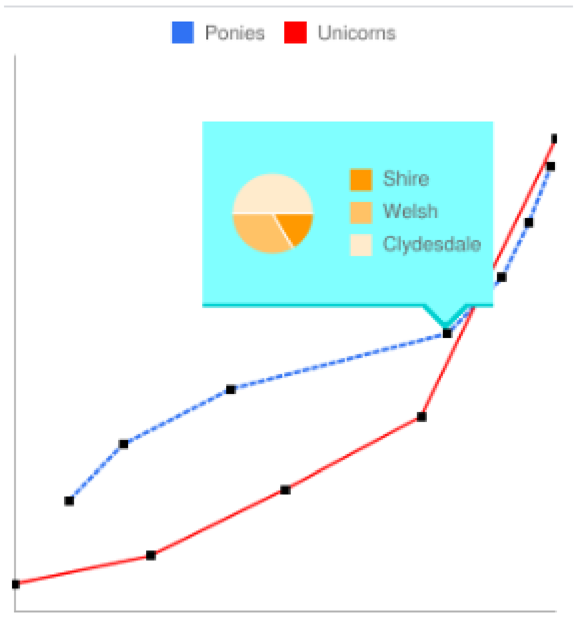
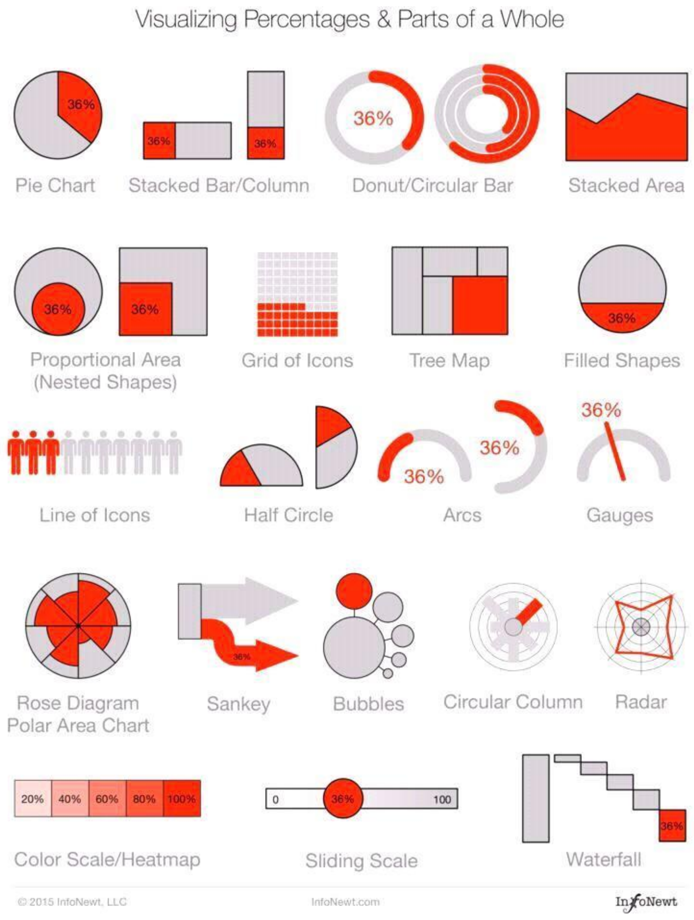


Figure 3.4 – Embedded Charts

https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/

3.2 Percentages



4 APPLICATION FEATURES

4.1 Data

4.1.1 Input Data Source

Gets data from database using a query

Timetrace data from database with a time range filter

Data can be procured using

- Table directly to get all records of table
- Using query file and arguments to get the data as required

4.1.2 Data Transformation

For utilizing unit conversion on data.
Currently uses scaling

Utilize transform with following:

- Shift
- Scale

4.1.3 Data Transformation – Parse Dates

(Should be in database document or refere to database document
Utilize pd_parse_dates_columns at the database definition level

```
pd_parse_dates_columns: [TIMESTAMP]
```

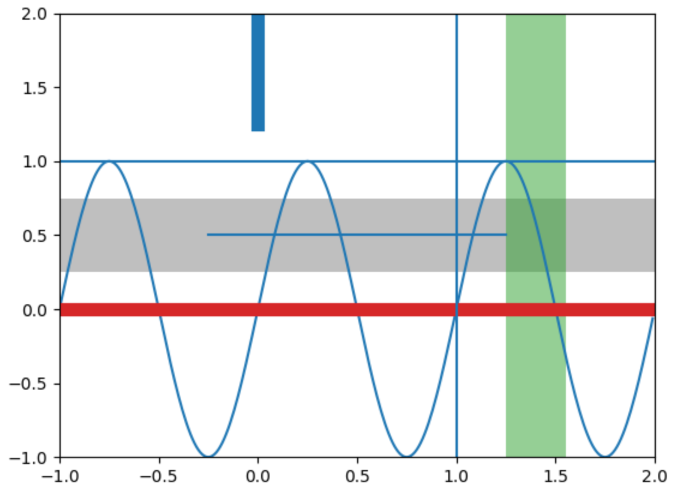
4.2 Single Charts

4.2.1 Line

Axline. To mark a line
Axvspan (To span using a color)

Draw a vertical, green, translucent rectangle from x = 1.25 to x = 1.55 that spans the yrange of the axes.

axvspan(1.25, 1.55, facecolor='g', alpha=0.5)



https://matplotlib.org/3.1.0/gallery/subplots_axes_and_figures/axhspan_demo.html#sphx-glr-gallery-subplots-axes-and-figures-axhspan-demo-py

For Datetime example:

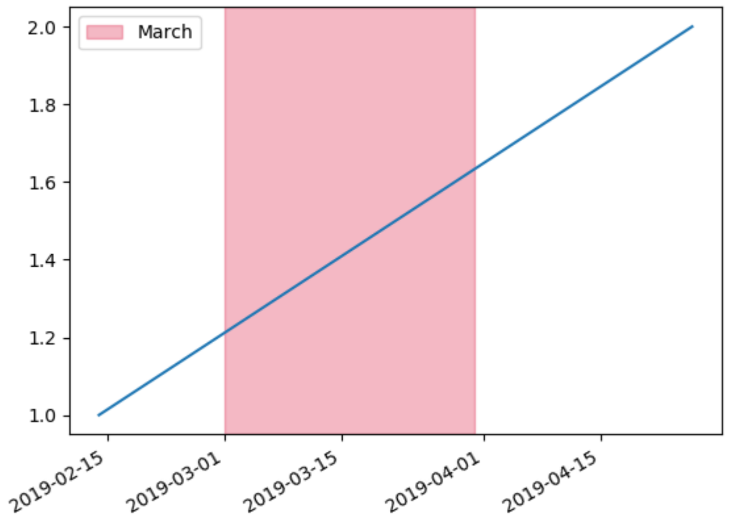
```
from datetime import datetime
import matplotlib.pyplot as plt
from matplotlib.dates import date2num

fig, ax = plt.subplots()

ax.plot([datetime(2019,2,14), datetime(2019,4,26)], [1,2])

ax.axvspan(date2num(datetime(2019,3,1)), date2num(datetime(2019,3,31)),
        label="March", color="crimson", alpha=0.3)

ax.legend()
fig.autofmt_xdate()
plt.show()
```



https://stackoverflow.com/questions/55866957/using-axvspan-for-date-ranges-in-matplotlib

4.2.2 Line with Makers

Customizing markers. Utilize below dictionary in plot_settings for each individual subplot.

```
marker:
  type: 'o'
  size: 3
  edge_color: None
```

Example code used is:

```
df.plot(kind=self.plt_settings['plt_kind'],marker =
self.plt_settings['marker']['type'], markersize=
self.plt_settings['marker']['size'],
        markerfacecolor= self.plt_settings['marker']['edge_color'])
```

https://stackoverflow.com/questions/8409095/matplotlib-set-markers-for-individual-points-on-a-line

4.2.3 Scatter

??

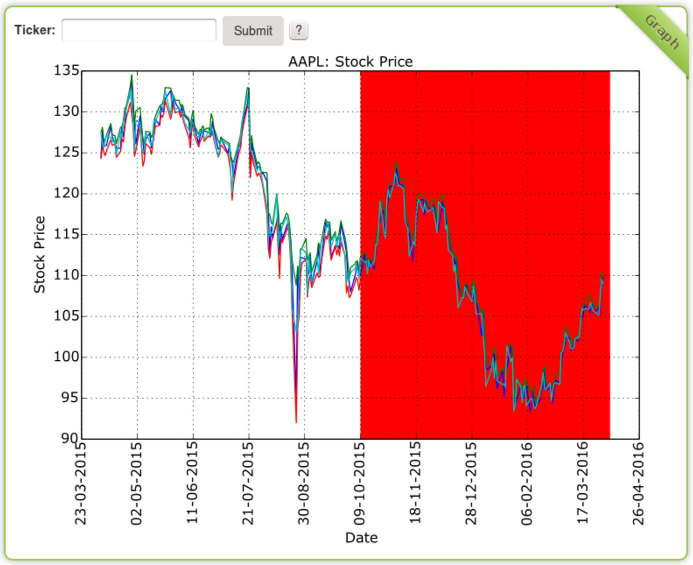
4.2.4 Timeline

Utilize Broken Horizontal Bar charts
https://matplotlib.org/devdocs/gallery/lines_bars_and_markers/broken_barh.html
https://stackoverflow.com/questions/47727339/plot-start-end-time-slots-matplotlib-python

http://www.blackarbs.com/blog/advanced-time-series-plots-in-python/1/6/2017

Format Date

Format Date this way for easy reading and reference.



https://matplotlib.org/gallery/recipes/common_date_problems.html

https://stackoverflow.com/questions/43968985/changing-the-formatting-of-a-datetime-axis-in-matplotlib/43969357

4.2.5 Gantt Chart

- Single chart – Type
    - Line
    - Scatter
    - Bar/column
    - Polar
    - Gantt chart

    - Timeline
- Single chart – Simple vs. Compound
    - Simple chart (only line, only bars etc.)
        - See Figure 3.2
    - Compound chart (only line, only bars etc.)
        - See examples in Figure 3.1
- Multiple charts
    - See Figure 3.3
- Embedded charts
    - See examples in Figure 3.4

4.2.6 Bar or column

These charts are not so well developed. Resort to library other than matplotlib to enable better plots.

4.2.7 Geographic Information System (GIS) Plots

The key packages available are:

- Arcgis
- Geopandas

References
https://developers.arcgis.com/python/guide/visualizing-data-with-the-spatial-dataframe/

https://www.earthdatascience.org/courses/scientists-guide-to-plotting-data-in-python/plot-spatial-data/customize-vector-plots/python-customize-map-legends-geopandas/

https://www.earthdatascience.org/courses/scientists-guide-to-plotting-data-in-python/plot-spatial-data/customize-vector-plots/

https://gis.stackexchange.com/questions/20276/is-the-arcgis-server-javascript-api-free-to-use?rq=1 ArcGIS conditions for use

Helpful tools:
Basemap
https://stackoverflow.com/questions/34979424/importing-mpl-toolkits-basemap-on-windows
pyproj
For converting coordinates

General References

https://www.maptools.com/tutorials/utm/quick_guide
http://www.earthpoint.us/Convert.aspx

4.2.8 Satellite Imagery

R has good packages
https://www.tylermw.com/a-step-by-step-guide-to-making-3d-maps-with-satellite-imagery-in-r/
https://github.com/tylermorganwall/rayshader

4.3 Multiple Subplots

An example code is given below:

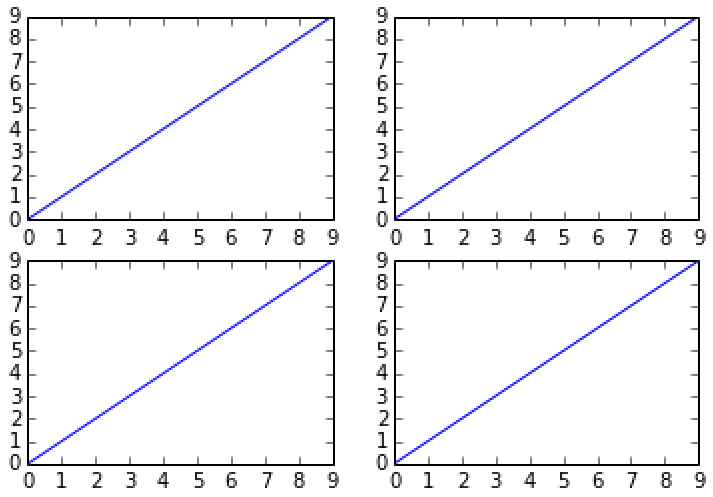
```
import matplotlib.pyplot as plt

x = range(10)
y = range(10)

fig, ax = plt.subplots(nrows=2, ncols=2)

for row in ax:
    for col in row:
        col.plot(x, y)

plt.show()
```



Order of difficulty:
Plotly < Matplotlib < Boheh

https://stackoverflow.com/questions/31726643/how-do-i-get-multiple-subplots-in-matplotlib
https://plot.ly/python/subplots/
https://stackoverflow.com/questions/26596901/bokehs-equivalent-to-matplotlib-subplots

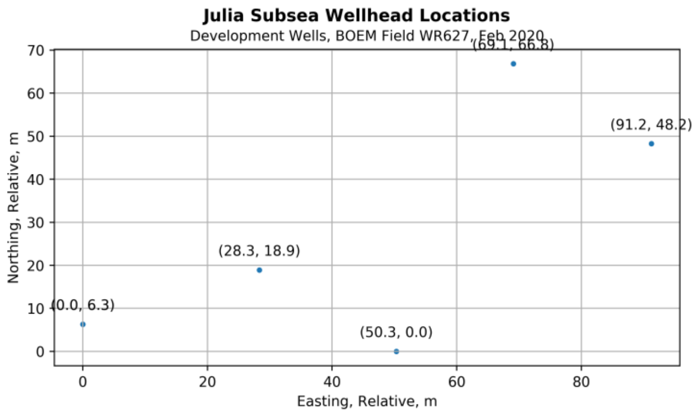
Setting Size of Figure
https://stackoverflow.com/questions/34028255/set-height-and-width-of-figure-created-with-plt-subplots-in-matplotlib/37405530
https://stackoverflow.com/questions/14770735/how-do-i-change-the-figure-size-with-subplots

4.4 General Features

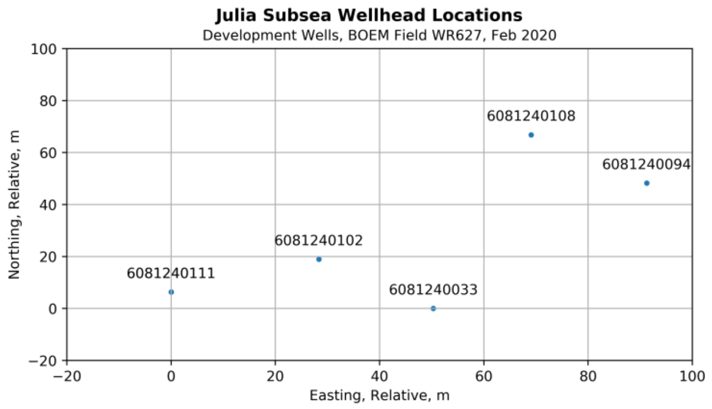
4.4.1 Annotate

If annotate is true and column is NULL. Then format is (x,y)
If

```
annotate:
  flag: True
  column: NULL
  # column: WELLAPI
```



```
annotate:
  flag: True
  # column: NULL
  column: WELLAPI
```



5 D3.JS

5.1 Overview

The data driven documents (D3) is a javascript library with the following salient features:

- Very flexible to code
- Very modular design
    - data (json)
    - Chart code (js),
    - Markup (html, css)
- Very popular library

The charts can be generated in multiple methods utilizing different libraries, section 5.2 However, the preferred method is to go the direct d3js route and maintaining their libraries directly – this gives good development strategy and stay abreast with the technologies.

There is a package dimple which is directly built on d3js with little code. On top of it, d3js can still be added for customization.

5.2 Possible Routes

The various routes possible and the tools available are summarized in this section.

- Direct D3 strategy
    - Utilize D3js using js and html
    - Prepare standard python code classes and functions to use

- Python developers are currently working on a D3JS module on pandas. Link is unknown
- An overall reference article:
    - https://stackoverflow.com/questions/12977517/python-equivalent-of-d3-js
- Python packages with timelines and brief summary
    - D3py
        - https://github.com/mikedewar/d3py
        - 2013
    - Vincent
        - 2018
        - https://github.com/wrobstory/vincent
    - Bokeh and plotly (active) gives html codes to be included
    - NVD3
        - Built on nvd3 and d3
        - https://github.com/areski/python-nvd3
        - 2019
    - Altair-viz

    - Built on vega
    - Actively maintained
    - Yet another package
  - Mpld3
    - Brings matplotlib to the browser
- Non-python packages
  - NVD3 : A very nice effort
    - built on top of D3
    - http://nvd3.org/examples/line.html
    - Supports simple examples, live coding broken into Chart code (js), data (json) and Markup (html, css)
      - http://nvd3.org/livecode/index.html#codemirrorNav

5.3 Dimple

http://dimplejs.org/examples_viewer.html?id=scatter_standard
A limited few starting graphs.
Not easily customizable.

5.4 Direct D3

Getting Started:
A short starter
https://alignedleft.com/tutorials/d3/

https://www.toptal.com/d3-js/towards-reusable-d3-js-charts

https://curran.github.io/dataviz-course-2018/
https://www.youtube.com/watch?v=rUnmw9fQEwg&list=PL9yYRbwpkykvOXrZumtZWbuaXWHvjD8gi&index=2

5.4.1 Key Concepts

The key concepts of D3js are as follows:
- Referencing D3
  - Use full version for development
  - The minified (min) version is compact for deployment
- Ability to chain functions
- Ability to select all data and assign them to a DOM (or multiple of DOM instances).
- Ability to access data using anonymous functions
  - Note the data variable, d can only be accessed by functions (anonymous or named)

    - For this function, D3 is looking for a variable name so it can pass the data into the argument. By convention variable name ‘d’ is default.
- Adding an action instance to a DOM will be overridden by the next action instance
- Power of data()
    - D3 Loops through the full length of dataset given in data() function
    - Each method beneath data() in the chain is executed while updating the context of d (i.e. the current datum in the loop)
    - The data index ‘i’ is also automatically populated
    - A limitation; if an existing dom is accessed using data() using a dataset array of length n, it creates n instances including the number of existing dom elements. Note this can a pitfall at times.
- Attr() is used to set an HTML attribute and value on an element.
    - An HTML attribute is any property/value pair .
    - For this, <img src="logo.png" width="100px" alt="logo" />
        - .attr(‘src’, ‘logo.png’)
        - .attr (‘width’, 100px)
        - .attr (‘alt’, ‘logo’)
    - To assign a generic styling class named “className”:
        - .attr (“class”, “className”)
    - To add or remove a class name:
        - Classed(“className”, true)
        - Classed(“className”, false)
- SVG Is a text-based image format and can be included in HTML
- Working with datetime
    - Convert the string to Date ojects
        - parseTime and
    - Use timescale (as needed)
    - Format Date objects as human friendly strings for UI
        - timefromat

5.4.2 Scatter Plot

Key Features

- Data as input
- Maker properties
    - Variable circle sizes by object property
    - Display data required (eg:)
- Mouse on and off
- Mouse click (Good for development?)
    - Show
    - Hide

- Others?

Key References:
Scott Murray Text Book
Simple plot
https://bl.ocks.org/d3noob/b3ff6ae1c120eea654b5
Simple plot with datapoints shown
https://bl.ocks.org/d3noob/38744a17f9c0141bcd04

- Few simple examples are
  - https://towardsdatascience.com/combining-python-and-d3-js-to-create-dynamic-visualization-applications-73c87a494396
  - https://www.analyticsvidhya.com/blog/2017/07/beginner-guide-build-data-visualisations-web-d3-js/
  - https://www.analyticsvidhya.com/blog/2017/08/visualizations-with-d3-js/

Future Features

-
- Develop ability to zoom
  - Zoomable development ideas:
  - https://observablehq.com/@d3/zoomable-scatterplot
-

5.4.3 Line Chart

Future Features

-
- Develop ability to zoom
  - Zoomable development ideas:
  - https://observablehq.com/@d3/zoomable-scatterplot
-

5.4.4 Bar Chart

Future Features

-
-

5.5 Colors

https://medium.com/@Elijah_Meeks/color-advice-for-data-visualization-with-d3-js-33b5adc41c90

https://observablehq.com/@d3/color-schemes
Category10
Accent
Dark2
Paired

https://www.tutorialspoint.com/d3js/d3js_colors_api.htm

5.6 Scalable Vector Graphics (SVG)

Scalable Vector Graphics (SVG) element:

- SVG Is a text-based image format and can be included
  - directly within any HTML document (or)
  - dynamically into the Document Object Model (DOM)
- Each SVG image is defined using markup code (similar to HTML)
- Sample SVG code
  - <svg width="50" height="50">
  - <circle cx="25" cy="25" r="22" fill="blue" stroke="gray" stroke-width="2"/>
  - </svg>
  - All svg properties are attributes and hence can be easily called from D3
- Getting a basic SVG working
  - Specify width and height
    - default units are px while supporting other units of em, pt, in, cm, mm
    -
- Coordinate system origin starts at top left corner
  - x is towards right; y towards bottom
  - SVG is a 2D graphic and there are no layers (or concept of depth)
- Typical shapes are:
  - circle, ellipse, line, text and path
  - Can be added at specified coordinates

  - Fill, stroke, stroke-width etc. are some salient properties
- Text can also be added.
  - Text will inherit the css-specified font styles (color, type, etc)
  - Can be added at a specified coordinate
- Colors and Transparency can be defined using rgba() format in fill command
  - rgba is rgb along with apha (transparency value between 0.0 and 1.0)
  - opacity can also be defined appropriately
  - Setting both alpha and opacity, the transperancies are multiplied
-
- Good practice is to where possible, reusable css code
  - should be added to a common .css file for resuse
  -

5.7 Resources

https://website.education.wisc.edu/~swu28/d3t/concept.html

3D Visualizations

D3 is not the best at 3D. With increased support for WebGL, there are now opportunities for 3D web experiences.

Three.js

https://threejs.org/examples/#webgl_geometry_spline_editor

6 PYTHON PYVIZ

6.1 Introduction and Installation

https://anaconda.org/pyviz/pyviz

https://towardsdatascience.com/pyviz-simplifying-the-data-visualisation-process-in-python-1b6d2cb728f1

https://www.anaconda.com/python-data-visualization-2018-why-so-many-libraries/

Scott Murray Book Code
https://github.com/scotthmurray/d3-book

6.1.1 Bokeh HeatMap

Bokeh is a Python library for interactive visualization that targets web browsers for representation.

The following code is used to read the data from CSV file.

```
import pandas
from bkcharts import HeatMap, output_file,show

df = pandas.read_csv('ESPExceptionsMaps.csv')
output_file('ESP.html', title= "ESP Heatmap")
p = HeatMap(df, title="ESP Heatmap", xlabel='ID', ylabel='CreatedTimeStamp', width=900, height=600)

show(p)
```

http://bokeh.pydata.org/en/latest/
http://bokeh.pydata.org/en/latest/docs/user_guide/categorical.html#userguide-categorical

http://bokeh.pydata.org/en/0.11.1/docs/installation.html

https://towardsdatascience.com/pyviz-simplifying-the-data-visualisation-process-in-python-1b6d2cb728f1

https://stackoverflow.com/questions/41282592/how-to-properly-create-a-heatmap-with-bokeh

6.2 Plotly

Creates super easy polar plots
https://plot.ly/python/polar-chart/

Exporting images in plotly
https://plot.ly/python/static-image-export/

6.3 Dash

Dash is a web Application framework which provides pure python abstraction around HTML, CSS and JavaScript.

Dash provides Python classes for all the visual components

- Dash_html_components
- Dash_html_components library has a component for every HTML tag. The html.H1(children='Hello Dash') component generates a <h1>Hello Dash</h1> HTML element in your application.

- Dash_core_components
- Dash_core_components describe higher-level components that are interactive and are generated with JavaScript, HTML, and CSS through the React.js library.
- Callback
- Events fired when input values change similar to Postback in .Net.
- Graphs/Plots
- Dash_core_components.Grpahs provides classes for data visualization.

Key Conclusions:

- A developer is indirectly writing html wrapped in python classes as explained above.
- For a Data Scientist, setting up an application in Dash will not be easy task. It will require some learning…
- Efforts required to build an interface containing user input(input fields ,dropdown ,radio button ,checkbox etc.), plots, filters will be same as setting up a .net web application.
- Setting up Database model, queries, writing back will require same effort as setting up a .net application.

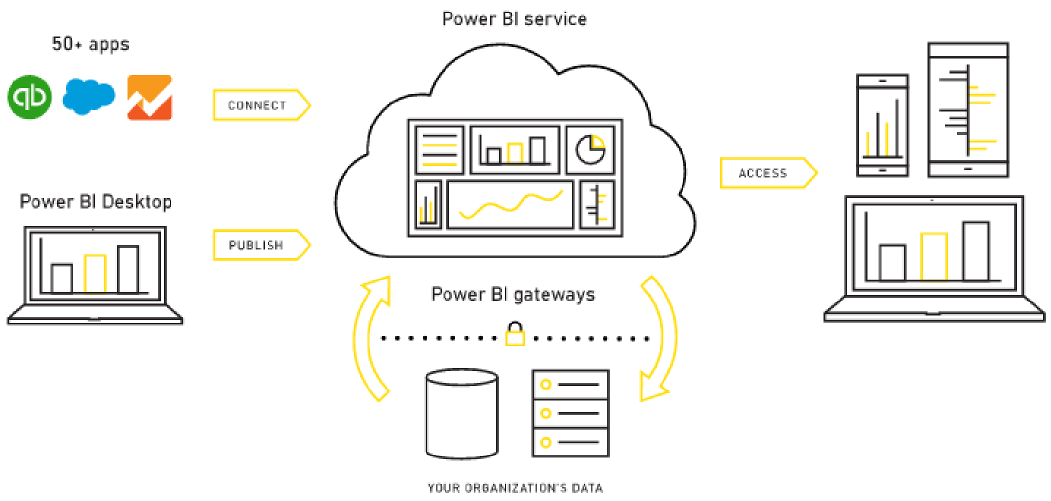
A great python resource to make good data science plots. Includes details of auto-formatting etc.
https://f0nzie.github.io/matplotlib-with-rmarkdown/advanced.html#change-of-axes

7 MS POWER BI

Power BI is a cloud-based business analytics service that enables anyone to visualize and analyse data with greater speed, efficiency, and understanding. It connects users to a broad range of data through easy-to-use dashboards, interactive reports, and compelling visualizations that bring data to life.

- Power BI dashboards provide a 360-degree view for business users with their most important metrics in one place, updated in real time, and available on all of their devices.
- And you can access your data and reports from anywhere with the Power BI Mobile apps, which update automatically with any changes to your data.
- Power BI Desktop is a feature-rich data mashup and report authoring tool.
- Combine data from disparate databases, files, and web services with visual tools that help you understand and fix data quality and formatting issues automatically.
- With the Power BI service, publish reports securely to your organization and setup automatic data refresh so everyone has the latest information.

- Power BI can unify all of your organization's data, whether in the cloud or on-premises. Using the Power BI gateways, you can connect SQL Server databases, Analysis Services models, and many other data sources to your same dashboards in Power BI.
- Power BI service were briefly shown in the below diagram,

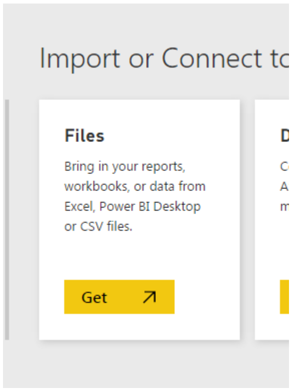


7.1.1 Getting Started with Power BI Service

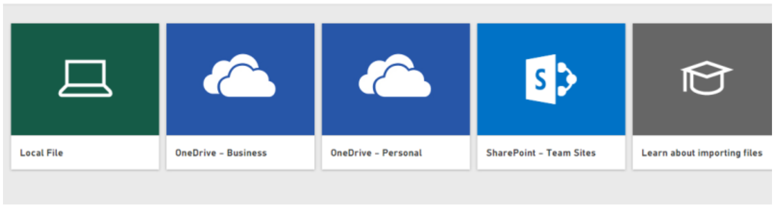
- Initially one has to Sign-up with the work address and not with the personal mail in the Microsoft Power BI for building reports, datasets and etc.
- Here we can Sign-up or Sign-in in the below link, https://powerbi.microsoft.com/en-us/
- When Service got Signed-up, you'll see a Welcome window on the screen.

7.1.2 Importing Files

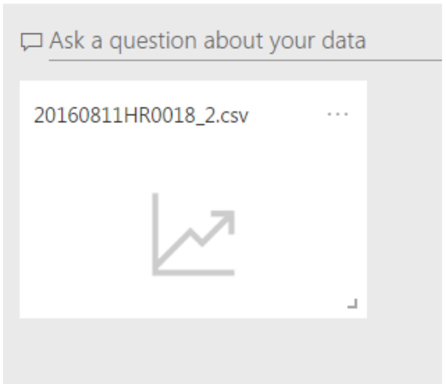
- On the Welcome screen, you will have a Get button in the Import or Connect to data block.


Import or Connect to
Files
Bring in your reports, workbooks, or data from Excel, Power BI Desktop or CSV files.
Get

- By clicking the button it directs to a page from where we need to get the file or data like from Local File or in any other cloud spaces.



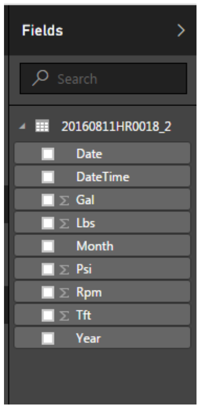
- As if we select a Local File option a file browse window is opened from where we need to select the file (.CSV, Excel, Text file, etc.)
- The selected file is displayed on the dashboard as below,



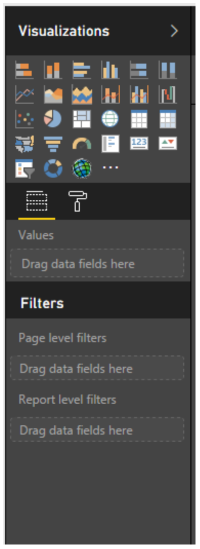
- We need to select the file for editing and adding visualizations to the data.

7.1.3 Editing the data from Imported File

- In My Workspace the imported file data is displayed under Fields section.



- The data in the file are displayed and we need to select the data for the graphs.
- The values and visualizations to be added for the data are shown under Visualizations section.



- Here we need to select the appropriate visuals for the data we uploaded.
- We can easily drag and drop from the fields to the Axis, Values and Filters.
- When the editing has done we can save the report by clicking the save option on the top navigation bar.
- A name should be provided to save the report.
- We can view the saved report in the Reports section in the side navigation bar.
- And the data is stored in the Datasets section in the side navigation bar.

7.1.4 Example 1 – Daily States

- Prepare data in a csv or excel file
- Load the data
- Transform the data

- Add data to a table
- Download the visualization template:

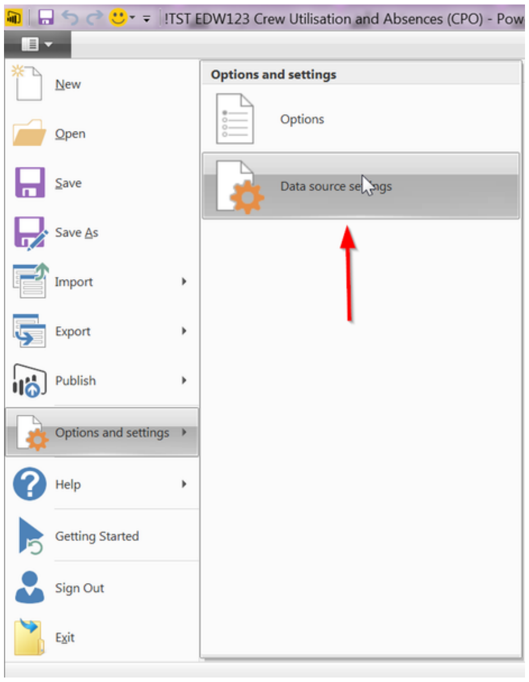
https://store.office.com/powerbiaddininstallpage.aspx?rs=en-US&assetid=WA104380818
- Select click on visualization
- Change the sum to "Do not summarize" in the end
- Change column titles
- COlorbrewer:
  - Use Set1 (Best combinations)
  - Paried can be used for risk maps (darker color means higher risk or over budget etc)

https://www.youtube.com/watch?v=1BHuy_vIyD0&t=41 Table Heat Map

7.1.5 Features

Change Data Source:

- Got to "Options and Settings"
- Got to "Data Source Settings"
- Right click to replace the underlying file
-



How to Print (PDF)

- Use online PowerBI version. This will help save with high resolution
- Using desktop app, only the snipping tool can be used
-

7.2 References

https://www.linkedin.com/pulse/7-best-data-visualization-tools-2017-bernard-marr

8 TECHNOLOGY DISCUSSIONS

8.1 Bokeh vs. Dash

I was using Bokeh till recently for some web plotting. After going through your email and reading further, Dash definitely looks futuristic with more capabilities and promise. As Dash is built on top of D3.JS, the final team (if using D3.JS) can still utilize or quickly build these visualizations.
https://blog.sicara.com/bokeh-dash-best-dashboard-framework-python-shiny-alternative-c5b576375f7f

I want to bring the following example work flow for discussion which we want to carry visualization all through the workflow to save time:
Data scientist (DataScience.calculation + DataScience.visualization with straightline plot) -> Cipher team (DataScience.calculation + Cipher.data provision + Temporary visualizations) -> Nexus/Service Delivery (DataScience.calculation + Cipher.data provision + Nexus.Visualization with smoothline plot)
a/ Visualization is not carried all through workflow.
b/ Technology utilized is different (Matplotlib – non interactive vs. D3.JS)

Whatever technology (Dash or D3.JS?) we choose, I suggest the following:
1/ Form standard visualization function libraries of Dash and d3js (simple charts, heatmap tables, …). We can grow this library
2/ Prepare a OXY Visualization cheatsheet so it will save all developers some (lot of) time and quick reference/refresh even for experienced folks. See example Bokeh cheatsheet attached.

8.2 PyViz (Big Data)

PyViz is a good tool to construct big data visualizations and codes.

https://www.youtube.com/watch?v=k27MJJLJNT4 PyViz Presentation

APPENDIX 1.0 - D3 REUSABLE CODE

Defining reusable d3 code is essential for repeat functions and data.

1.1 Bar Chart Example

Step 0: Simple starter code

```
    d3.select('body').append('svg')
        .attr('height', 300)
        .attr('width', 800)
        .selectAll('rect')
            .data(milesRun)
            .enter()
            .append('rect')
            .attr('y', function (d, i) { return i * 40 })
            .attr('height', 35)
            .attr('x', 0)
            .attr('width', function (d) { return d*100})
            .style('fill', 'steelblue');
```

Step 1: Define configuration variables

```
    var highTemperatures = [77, 71, 82, 87, 84, 78, 80, 84, 86, 72, 71, 68];
    var height = 300;
    var width = 800;
    var barPadding = 1;
    var barSpacing = height / highTemperatures.length;
    var barHeight = barSpacing - barPadding;
    var maxValue = d3.max(highTemperatures);
    var widthScale = width / maxValue;

    d3.select('body').append('svg')
            .attr('height', height)
            .attr('width', width)
            .selectAll('rect')
            .data(highTemperatures)
            .enter()
            .append('rect')
            .attr('y', function (d, i) { return i * barSpacing })
            .attr('height', barHeight)
            .attr('x', 0)
            .attr('width', function (d) { return d*widthScale})
            .style('fill', 'steelblue');
```

Step 2: Repeat through Functions

```
<!DOCTYPE html>
<html lang="en">
<script src="https://d3js.org/d3.v5.min.js"></script>
<script src="lib/d3_reusables.js"></script>
<head>
    <meta charset="UTF-8">
    <title>D3 dimple Tutorial</title>
</head>

<body>

<h1>2 Bar Charts with repeatable code</h1>

<div id="#runningHistory"></div>

<div id="#weatherHistory"></div>

<script type="text/javascript">
    var milesRun = [2, 5, 4, 1, 2, 6, 5];
    var runningOptions = {barPadding: 2};
    drawBarChart(document.getElementById("#runningHistory"), milesRun, runningOptions)
;

    var highTemperatures = [77, 71, 82, 87, 84, 78, 80, 84, 86, 72, 71, 68, 75, 73, 80
, 85, 86, 80];
    var weatherOptions = {fillColor: 'coral'};
    drawBarChart(document.getElementById("#weatherHistory"), highTemperatures, weather
Options);

</script>

</body>
</html>
```

```
function drawBarChart(dom, data, options) {
    var width = options.width || 800;
    var height = options.height || 200;
    var barPadding = options.barPadding || 1;
    var fillColor = options.fillColor || 'steelblue';

    var barSpacing = height / data.length;
    var barHeight = barSpacing - barPadding;
    var maxValue = d3.max(data);
    var widthScale = width / maxValue;

    d3.select(dom).append('svg')
            .attr('height', height)
            .attr('width', width)
            .selectAll('rect')
            .data(data)
            .enter()
            .append('rect')
            .attr('y', function (d, i) { return i * barSpacing })
            .attr('height', barHeight)
            .attr('x', 0)
            .attr('width', function (d) { return d*widthScale})
            .style('fill', fillColor);
    }
```

Other Variation Links

https://gist.github.com/ThomasBurleson/c23f4a14917ad54551f9

https://bl.ocks.org/hrecht/f84012ee860cb4da66331f18d588eee3 Adding X and Y Axis and labels
https://observablehq.com/@d3/bar-chart