Data Visualisation With Altair

Name: Mary Etokwudo **Email: etokwudomary@gmail.com**

SECTION 1: PROBLEM DEFINITION

VAST Challenge 2018 Mini Challenge 2

Introduction

VAST challenge is an annual international visual analytics competition. It provides a dataset and a few analysis questions.
Each entry needs to present:
1. The visual analytics tool they developed and
2. How they found the answers using the tool

Background

(All the people, places, groups, technologies, contained therein are fictitious.)

Mistford is a mid-size city to the southwest of the Boonsong Lekagul Wildlife Preserve. The city has a small industrial area with four light-manufacturing endeavours. Mistford and the wildlife preserve are struggling with the possible endangerment of the Rose-Crested Blue Pipit, a locally loved bird. The bird's nesting pairs seem to have decreased alarmingly.

An investigation last year (VAST challenge 2017) indicated that the Kasios Office Furniture, a Mistford manufacturing firm, may be linked to this.

Though there is no firm evidence. Now the company insists that they have done nothing wrong. It is time for more visual analytics investigation.

Dataset

Several years of water sensor readings from rivers and streams in the preserve. These samples were taken from different locations scattered throughout the area. It contains measurements of several chemicals of possible interest. The task is to investigate the sensor readings to find a possible link to the bird population deduction.

Analysis questions:

1. Describe trends and anomalies with respect to chemical contamination.
   i. Trends: changes over time and/or sensor site.
   ii. Anomalies: sudden change over time or one site significantly different from others.
2. Describe any data quality and uncertain issues, such as:
   i. missing data,
   ii. change in collection frequency, and.
   iii. unrealistic values (e.g. water temperature higher than 100 degrees).

SECTION 2: SOLUTION

NOTE: The visualisation tool used to analyse the dataset is [Vega-Altair](https://altair-viz.github.io/)

Data Preprocessing

Using pandas for data preprocessing, I converted the “sample date” variable to a datetime type and got the day, month and year variables from it. I created new columns for these variables.

	id	sample_date	day	month	year	location	measure	value
0	2221	1998-01-11	11	Jan	1998	Boonsri	Water temperature	2.00
1	2223	1998-01-11	11	Jan	1998	Boonsri	Dissolved oxygen	9.10
2	2227	1998-01-11	11	Jan	1998	Boonsri	Ammonium	0.33

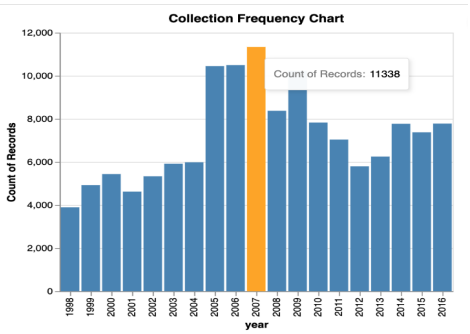
Data Quality

Change in Collection Frequency (Annual)

The chart below shows the collection frequency over the years.

```
# plotting a bar chart to see the frequency of s
frequncy_chart = alt.Chart(
    df, title='Collection Frequency Chart'
).mark_bar().encode(
    x='year:O',
    y='count(value)',
    tooltip=['count(value)']
)
frequncy_chart
```

```
# highlighting the highest frequency
frequncy_chart.encode(
    color = alt.condition(
        datum['year'] == 2007,
        alt.value('orange'),
        alt.value('steelblue')
    )
)
```



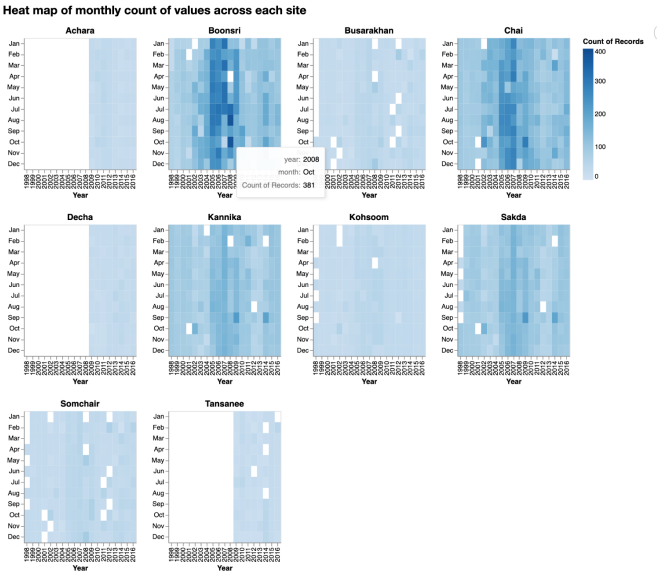
Insights:
The highest number of collections was in 2007 with a total of 11338 records. The least number of collections was in 1998 with a total of 3893 records.
The What: 1 ordinal attribute (year)
The Why:
Aim: To show the collection frequency over the years.
Actions: Summarise the count of records for each year and Discover the most count.
Targets: Distribution of the data over the years.
The How:
Bar charts are well suited for comparison of ordinal data.
Mark: Line
Channel:
x position: key (ordinal attribute)
y position: Quantity (quantitative attribute)
Length(for quantity) and Colour (for popout)

Missing Data and Monthly Frequency

The heatmap below is a calendar plot of the count of values for each month of every year. There are two aims of using this type of plot:

1. To show missing data (months with no records would be blank).
2. To show the volume of the month collection using colour saturation.

```
display(Markdown('### Heat map of monthly count of values across each site'))

# To plot heat maps across all locations

# splitting the 10 locations into 5 rows
row_1 = alt.concat()
row_2 = alt.concat()
row_3 = alt.concat()

for index, location in enumerate(locations):
    # generating a chart for each location
    chart = alt.Chart(df[df['location'] == location]).mark_rect().encode(
        x=alt.X('year:O', axis=alt.Axis(title='Year'), scale=alt.Scale(domain=year_list)
        y=alt.Y('month:N', axis=alt.Axis(title=None), scale=alt.Scale(domain=month_list)
        color=alt.Color(
            'count(value):Q', scale=alt.Scale(scheme='blues', domain=(0, 400))),
        tooltip=['year', 'month', 'count(value)']
    ).properties(width=170, height=200, title=location)

    # choosing which row to place the chart
    if index < 4:
        row_1 |= chart
    elif 4 <= index <= 7:
        row_2 |= chart
    else:
        row_3 |= chart

# vertically concatenating all the rows
alt.vconcat(row_1, row_2, row_3)
```



Insights:

From the dashboard above, the white cells or spaces indicate no readings. The coloured cells indicate readings, with the colour saturation representing the quantity of readings. The following insights can be drawn:

1. Monthly and yearly frequency:
    a. Readings were taken between 1998 and 2016 for Boonsri, Busarakhan, Chai, Kannika, Kohsoom, Sakda and Somchair.
    b. Readings were taken between 2009 and 2016 for Achara, Decha and Tansanee. No readings prior to 2009.
2. Missing Data: Within the years the readings were taken, there are missing readings for some months. For instance, Boonsri has missing readings for Jan, 2002.
3. Volume of collection: The darker cells show higher volumes than the lighter cells. The most collections were at Boonsri in October 2008.

The What: 2 quantitative attribute (year and value), 1 Categorical attribute (month)

The Why:

Aim: To show the collection frequency over the months for each location and, in the process, spot missing collections.
Actions: Summarise the count of records for each month.
Targets: Distribution of the data over the months and years.

The How:

Mark: Area
Channel: Colour Saturation, and Juxtapositioning of the charts for all locations.

Analysing the Measures

Since the number of measures (or chemical contaminants) is very large at 106, it is challenging to visualise all of them at once. I will be focusing on the most frequently collected measures and the measures with the highest values.

```
# Frequency of measures in the dataset
df.measure.value_counts(ascending=False).head(10)
```

```
Water temperature            5031
Nitrites                     4791
Ammonium                     4790
Nitrates                     4786
Orthophosphate-phosphorus    4782
Total phosphorus             4600
Dissolved oxygen             4531
Biochemical Oxygen           4488
Manganese                    4039
Chlorides                    3961
Name: measure, dtype: int64
```

```
daily_mean_values = df.groupby(
    ['measure', 'sample_date']).agg({'value': ['mean']})
daily_mean_values.columns = ['mean_value']
daily_mean_values= daily_mean_values.sort_values(
    by=['mean_value'], ascending=False)
daily_mean_values = daily_mean_values.reset_index()
daily_mean_values
```

	measure	sample_date	mean_value
0	Iron	2003-08-15	27299.0
1	Total coliforms	2009-01-15	16090.0
2	Total coliforms	2010-10-20	13000.0
3	Total coliforms	2009-11-20	4600.0
4	Total coliforms	2009-08-20	3300.0

Insights:

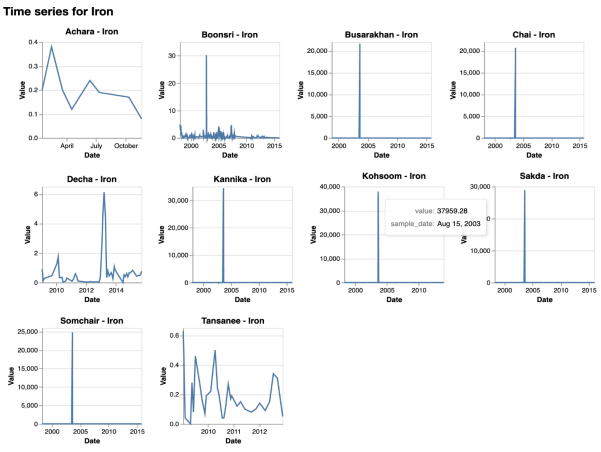
Looking at the table above, the maximum mean value is for Iron which is 27,299 on the 15th of August 2003. This means that, on that day, the average of all the Iron values collected across all locations is 27,299. This is a very high value but I cannot outrightly say if it is wrong or right as I do not know what the range of value is for other iron readings. Also, I expect that the next max values will be close to 27,299 and belong to Iron. Instead, there is a different measure Total coliforms with a mean value of 16,090. This value is about half of the first value. It is either there is no other Iron reading or the next Iron reading has a value far from the first, the reason why it did not make it to the top of the table. Further investigation is needed to understand the dataset for Iron.

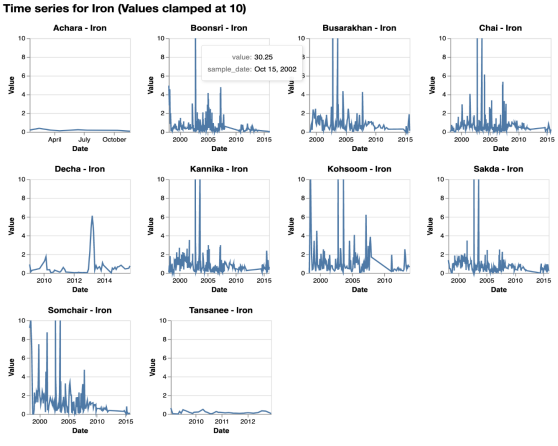
Trends & Anomalies

Plotting the time series for Iron across all the locations. The code snippet on the left below produces the chart on the left. The snippet on the right is used to limit and clamp the values on the y-axis. More details on this in the next paragraph.

```
# base chart with mark and encodings
base_chart = alt.Chart(
    df[df['measure'] == measure] # filter data by the measure
).mark_line().encode(
    x=alt.X('sample_date:T', axis=alt.Axis(title='Date', grid=False)),
    tooltip=['value', 'sample_date']
).properties(
    width = 150,
    height = 150
)
```

```
# generating a chart for each location
chart = base_chart.transform_filter(
    datum.location == location
).properties(
    title = title
)

if y_scale:   # add the y scale if given
    chart = chart.encode(
        y = alt.Y('value', axis=alt.Axis(title='Value'),
                  scale=alt.Scale(domain=y_scale, clamp=True))
    )
else:
    chart = chart.encode(
        y = alt.Y('value', axis=alt.Axis(title='Value'))
    )
```





The What: 1 quantitative attribute (value), 1 datetime attribute (sample date),
The Why:
Aim: To show the trend for a measure for each location while spotting anomalies and outliers.
Actions: Summarise the measure trend and discover anomalies.
Targets: Trend of measure values over the years.

The How:
Line graphs are well suited for trend assessments
Mark:
Line between marks
Channel:
Y: Lengths to express quantitative value
X: Ordered by datetime attribute

Insights:
From the visualisation on the left above, it can be observed that some locations have a shorter range of values compared to others.
Achara and Tansanee have values between 0 and 0.7.

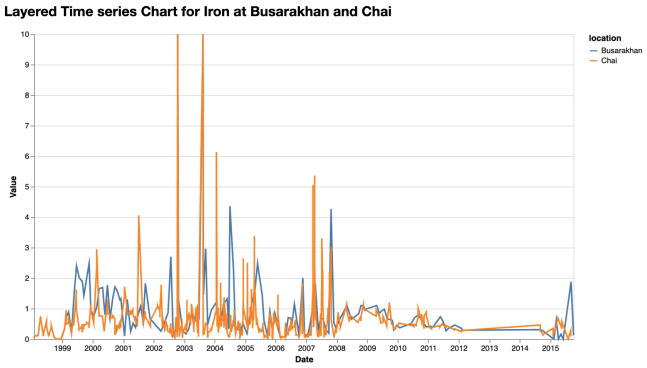
Noticeably, other locations have a sudden spike in value. This spike runs into tens of thousands for Busarakhan, Chai, Kannika, Kohsoom, Sakda and Somchair. Hovering over these spikes reveal that they happened on the same day, 15th of August 2003. Other sensors do not have readings for August 2003 except for Boonsri. The code snippet cell below reveals that Boonsri has no reading for 15th Aug 2003.

```
# Checking the dataset for Boonsri to see if there's a reading for 15, Aug 2003
df[(df['location'] == 'Boonsri') & (df['sample_date'] == pd.to_datetime('2003-08-15'))]
```

id	sample_date	day	month	year	location	measure	value

If all the samples collected on this day were done with the same sensor, there may be a sensor malfunction on that day which resulted in **unrealistic values** for Iron running into tens of thousands. To properly visualise the trends of Iron readings, it is necessary to remove (or hide) the effect of these high values by limiting the values on the y-axis to a lower number. Comparing the Iron readings for other locations, I have chosen to limit the y-axis to 10 as most of the other readings fall below 10. The visualisation on the right is the result of clamping the values at 10.
Furthermore, I noticed that Iron readings after 2008 are generally below 2 but are higher before 2008.
Also, Boonsri, Busarakhan, Chai, Kannika, Kohsoom and Sakda show similar trends over the years. Looking at the map provided, the locations nearest to the waste dumping site are Boonsori, Kohsoom, Busarakhan and Chai. For the measures that have been visualised so far, there is a similar trend for these locations. For the other locations, the data is not enough to conclude if there are similar trends.
The next chart below shows a similarity in the trends for Iron for Busarakhan and Chai when they are layered.

```
busarakhan_chart = alt.Chart(
    df[(df['measure'] == 'Iron') & (df['location'] == 'Busarakhan')]
).mark_line().encode(
    x=alt.X('sample_date:T', axis=alt.Axis(title='Date', grid=False)),
    y = alt.Y('value', axis=alt.Axis(title='Value'),
              scale=alt.Scale(domain=(0,10), clamp=True)),
    tooltip=['value', 'sample_date'],
    color='location:N'
).properties(
    width = 700,
    height = 400
)

chai_chart = alt.Chart(
    df[(df['measure'] == 'Iron') & (df['location'] == 'Chai')]
).mark_line(color='orange').encode(
    x=alt.X('sample_date:T', axis=alt.Axis(title='Date', grid=False)),
    y = alt.Y('value', axis=alt.Axis(title='Value'),
              scale=alt.Scale(domain=(0,10), clamp=True)),
    tooltip=['value', 'sample_date'],
    color='location:N'
).properties(
    width = 700,
    height = 400
)

busarakhan_chart + chai_chart
```



In addition to the analysis framework described in the previous page, the following marks and channels are also used in this chart.

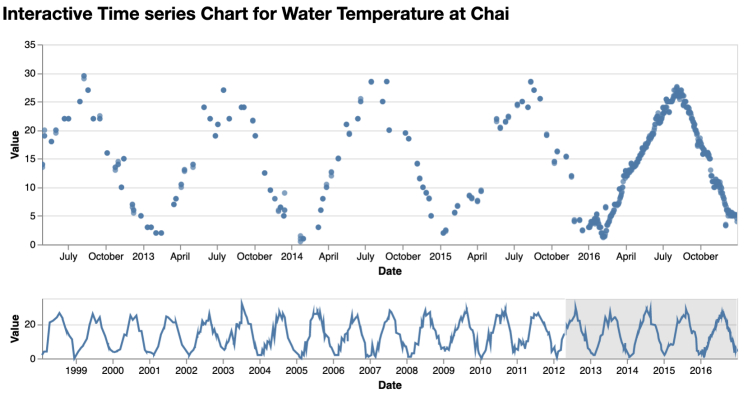
The Why:
Aim: To **compare** the **trends** of Iron for Chai and for Busarakhan and see if there's a **similarity**.

The How:
Channel:
Colour to differentiate the two locations

Time Series for Water Temperature at Chai

The time series below applies interval selection from Altair to enable the user to zoom in on time intervals of interest.

```
display(Markdown('### Interactive Time series Chart for Water Temperature at Chai'))

# Interval Selection brush
brush = alt.selection(type='interval', encodings=['x'])

upper = alt.Chart(df).mark_circle().encode(
    alt.X('sample_date:T', scale=alt.Scale(domain=brush),
          axis=alt.Axis(title='Date', grid=False)),
    y = alt.Y('value', axis=alt.Axis(title='Value')),
    tooltip=['value', 'sample_date']
).transform_filter(
    # filter data by the measure & location
    (datum.location == "Chai") & (datum.measure == "Water temperature")
).properties(
    width = 700,
    height = 200
)

lower = alt.Chart(df).mark_line().encode(
    x=alt.X('sample_date:T', axis=alt.Axis(title='Date', grid=False)),
    y = alt.Y('value', axis=alt.Axis(title='Value'))
).transform_filter(
    # filter data by the measure & location
    (datum.location == "Chai") & (datum.measure == "Water temperature")
).properties(
    width = 700,
    height=60
).add_selection(brush)

upper & lower
```



Insight:

The water temperature at Chai has a trend that is consistent all through the years. Temperature rises to around 30 degrees between June and July, then goes down slowly until the end of the year where it starts rising again. Also, this chart shows that the most collection was in the year 2016 (the upper chart makes use of dots to show how many records are).

In addition to the analysis framework in the previous page, the following marks and channels are also used in this chart.
The why: To **discover** the **trend** for Nitrates at Kohsoom and be able to zoom in and out of the time series.
The How:
Mark: Point and line
Channel: Interval **selection** to zoom in on the datetime

Unrealistic Values

Asides Iron, some other measures show values that can be seen as unrealistic when compared to other readings. This is visualised in the dashboard below where Manganese has the most remarkable spikes across different locations. The chart applies Altair selection for a dropdown selection of all the locations and a legend selection of the top 10 frequent measures.

Location-Measure Time Series Dashboard

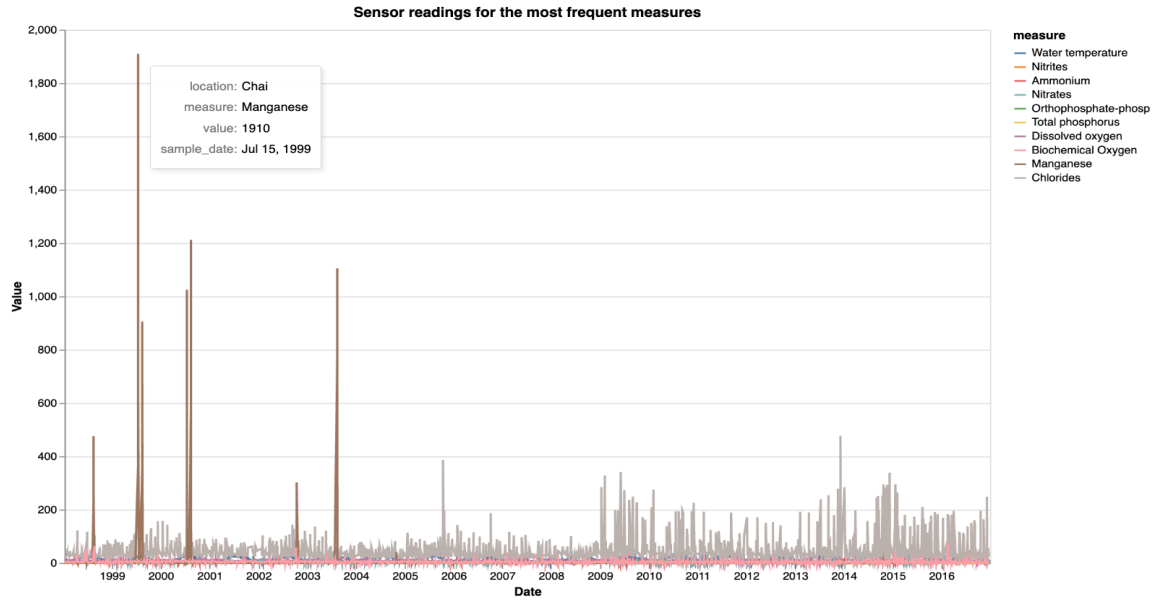
```
locations_dropdown = alt.binding_select(options=locations)
```

```
locations_selection = alt.selection_single(
    fields=['location'], bind=locations_dropdown, name='Select ')
```

```
measures_selection = alt.selection_multi(fields=['measure'], bind='legend')
```

```
# Filter the dataframe for only the measures of interest
interest_df = df[df['measure'].isin(top_10_by_count)]
```

```
alt.Chart(
    interest_df, title=("Sensor readings for the most frequent measures")
).mark_line().encode(
    x=alt.X('sample_date:T', axis=alt.Axis(title='Date', grid=False)),
    y=alt.Y('value', axis=alt.Axis(title='Value')),
    color=alt.Color('measure:N', scale=alt.Scale(domain=top_10_by_count)),
    tooltip=['location', 'measure', 'value', 'sample_date']
).add_selection(
    measures_selection,
    locations_selection
).transform_filter(
    measures_selection
).transform_filter(
    locations_selection
).properties(
    width=800,
    height=500
)
```



The What: 1 quantitative attribute (value), 1 datetime attribute (sample date), 2 categorical attribute (measure and location)

The Why:

Aim: To give the user the opportunity to select location from a drop down and select measure from the legend.

Actions: Summarise the measure trend and discover anomalies.

Targets: Trend of measure values over the years and for each location

The How:

Line graphs are well suited for trend assessments

Mark: Line

Channel:

Colour legend for the measures

2 selections and filters (dropdown and legend)