Using HoloViz to Create an Interactive Dashboard

Introduction

As data becomes more copious in today's world, it would follow that building an understanding of it is necessary to further our ... As data scientist it is not only our job to identify insights and nuances within the data, but also communicate this information to those not as technically prowess. As such creating interactive dashboards and providing detailed explanations into their inner workings is a huge part of the role. Furthermore, visualising data can also lead to discoveries not available by merely performing statistical tests (think Simpson's Paradox or the Anscombe's Quartet).

In this article, I will utilise Panel, hvPlot and HoloViews, component of the Holoviz suite of interactive visualisation libraries that can you can use to build dashboards capable of demonstrating the different facets of your dataset. The library is capable of being integrated into Jupyter notebooks or ran as an independent Python script.

Please refer to the associated notebook for more details on the implementation

The HoloViz Suite in a Nutshell

The HoloViz suite is a collection of libraries developed by Anaconda, Inc. and the PyViz community to simplify interactive visualizations in Python. All the libraries are open-sourced, regularly updated and maintained by the community. Their packages provide high-level APIs for workflows like creating dashboards, plotting geographical information, and even agent based framework for automated data retrieval using LLMs.

A key advantage is it’s extensibility. It’s ecosystem provides a wide variety of API that integrate with one another to create a common plane for workflows. They also have 3rd party extensions to common plotting libraries (e.g seaborn, matplotlib, plotly) for their visualisation libraries for deeper functionality.

Another advantage is it’s declarative nature. The simplifies the programming flow since it requires all components to be defined and called in order, making it easier to pickup. It also has native support for Jupyter notebooks, allowing renders to be shown inline, making it very data science friendly.

Considering all these, it makes the HoloViz a good starting point to explore interactive visualisations, but it’s wide ecosystem also means that more detailed plots can be created if necessary.

Some other popular dashboarding options include `plotly` + `dash` which is a JS-backed rendering, or `Altair` which is based on `Vega`

It does suffer from some limitations which are explained at the end of this article.

The Setup

If you have `poetry` installed you can install install dependencies directly after cloning the project:

```
poetry install
```

Otherwise you can use the `requirements.txt` inside the repo using `pip` after creating a virtual environment:

```
pip install -r requirements.txt
```

The Data

We will be exploring graduate employment details from some Universities in Singapore. The Graduate Employment Survey (GES) is jointly conducted by NTU, NUS, SMU, SIT (from 2014), SUTD (from 2015) and SUSS (from 2018) annually to survey the employment conditions of graduates about six months after their final examinations. This information is published by the Ministry of Education (MOE) to provide prospective students with timely and comparable data to assist them in making informed decision about the course decisions.

The dataset contains details of the University that offers the program, the degree, employment rates and monthly salary details as summary statistics. A more details breakdown of columns and notations can be found online at the source.

Source

The Dashboard

The general structure for a HoloViz dashboard is to first declare the widgets you want to use, creates the plots, then finally combine them into a comprehensive dashboard. Since the they recommend a declarative approach to programming, we have to create these components in order.

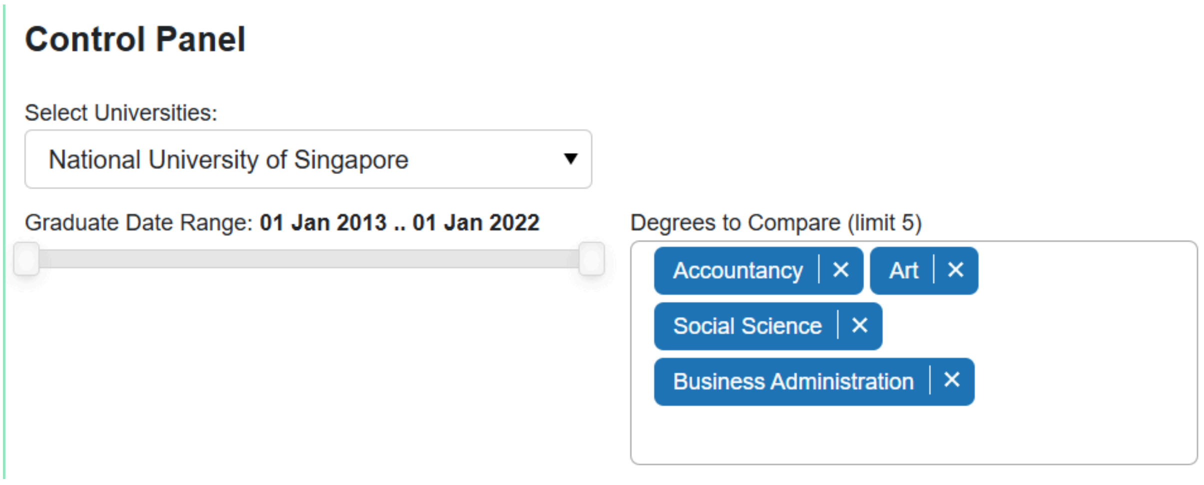
Control Panel

To create the control panel, we select the widgets we would like to use and store them in variables. These variables will later be used as parameters to the plots that will adjust based on what is selected by the user. Here, we create 3 widgets:

- `Select` — to select the University to filter on
- `DateRangeSlider` — to select the years to observe from
- `Multichoice` — to select the degrees of interest

The program flow can be see in the snippet below

```
# Create the select widget
uni_select = pn.widgets.Select(
  name='Select Universities:',
  value='National University of Singapore',
  options=universities,
)
```


Control Panel
Select Universities:
National University of Singapore
Graduate Date Range: 01 Jan 2013 .. 01 Jan 2022
Degrees to Compare (limit 5)
Accountancy
Art
Social Science
Business Administration

All the widgets created and displayed

Plot 1: Scatter + Line Plot

We start with the first plot. Since the data is changing dynamically when users make changes, two actions need to be done: (1) Filter the data (2) Create the plot.

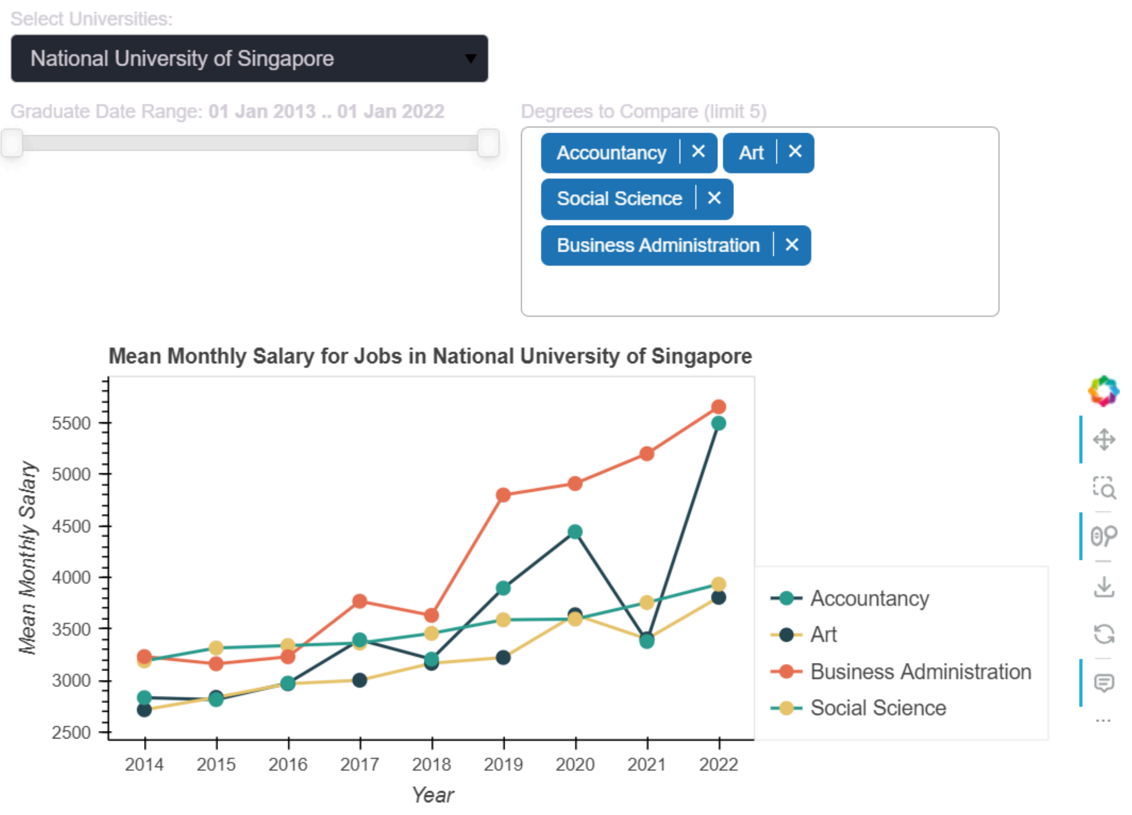
A scatter and line plot that showcases the changing mean monthly salary for fresh graduates across the different years. This plot is good for showcasing the historical changes for different categorical groups of data. Since our data is a time-series, it would be good to observe the change over time.

The plot should be able to filter according to school and degree. This will allow the user to filter and compare the job acquisition performance of the same degree across different schools.

```
@pn.depends(uni_select, year_slider, degree_multi_select) # Defines the widg
                                                           # variabl
def scatter_line_plot(uni_select, year_slider, degree_multi_select):
  # ---------------- STEP 1: DATA FILTERING ----------------
 df1 = df.copy()

 # Change the year details to string type for more consistent comparison
 start_year, end_year = int(year_slider[0].year), int(year_slider[1].year)
 year_range = [str(start_year + i) for i in range(1, int(end_year - start_year + 1))
```

```
df1['year'] = df1['year'].dt.strftime("%Y")

# Prepare the data for the plot
df1 = df1[
  (df1['university'] == uni_select) &
  (df1['year'].isin(year_range)) &
  (df1['degree'].isin(degree_multi_select)) &
  (df1['advanced'] == 0) # automatically filter out advanced further studies
]
df1 = df1.sort_values('year')

# Pivot the dataframe to match hvplot format
df2 = df1.pivot(index='year', columns='degree', values='basic_monthly_mear
cols_for_line = list(df2.columns.drop('year'))

  # ---------------- STEP 2: PLOTTING ----------------
# Plot the line graph
scatter_plot = df1.hvplot.scatter(
  x='year',
  y='basic_monthly_mean',
  by='degree',
  size=70,
  hover_cols=['year', 'basic_monthly_mean'],
  title=f'Mean Monthly Salary for Jobs in {uni_select}',
  xlabel='Year',
  ylabel='Mean Monthly Salary',
  color=COLOURS[:len(degree_multi_select)],
  legend='top'
)

line_plot = df2.hvplot.line(
  x='year',
  y=cols_for_line,
  by='degree',
  color=COLOURS[:len(degree_multi_select)]
)

# Combine the scatter plot and line plot together
```

```
plot1 = line_plot * scatter_plot

return plot1
```


Select Universities:
National University of Singapore
Graduate Date Range: 01 Jan 2013 .. 01 Jan 2022
Degrees to Compare (limit 5)
Accountancy
Art
Social Science
Business Administration
Mean Monthly Salary for Jobs in National University of Singapore
Mean Monthly Salary
Year
Accountancy
Art
Business Administration
Social Science

Scatter + Line Plot

One observation is that it's difficult to control the assigned colour for each category when they are separated into different plots i.e scatter and line. As seen above, the dots and line are not always the same colour. This happens because `hvplot` assigns colour based on the order of categories in the original dataframe, which is structured differently for both the scatter and line plot, therefore the colours are off.

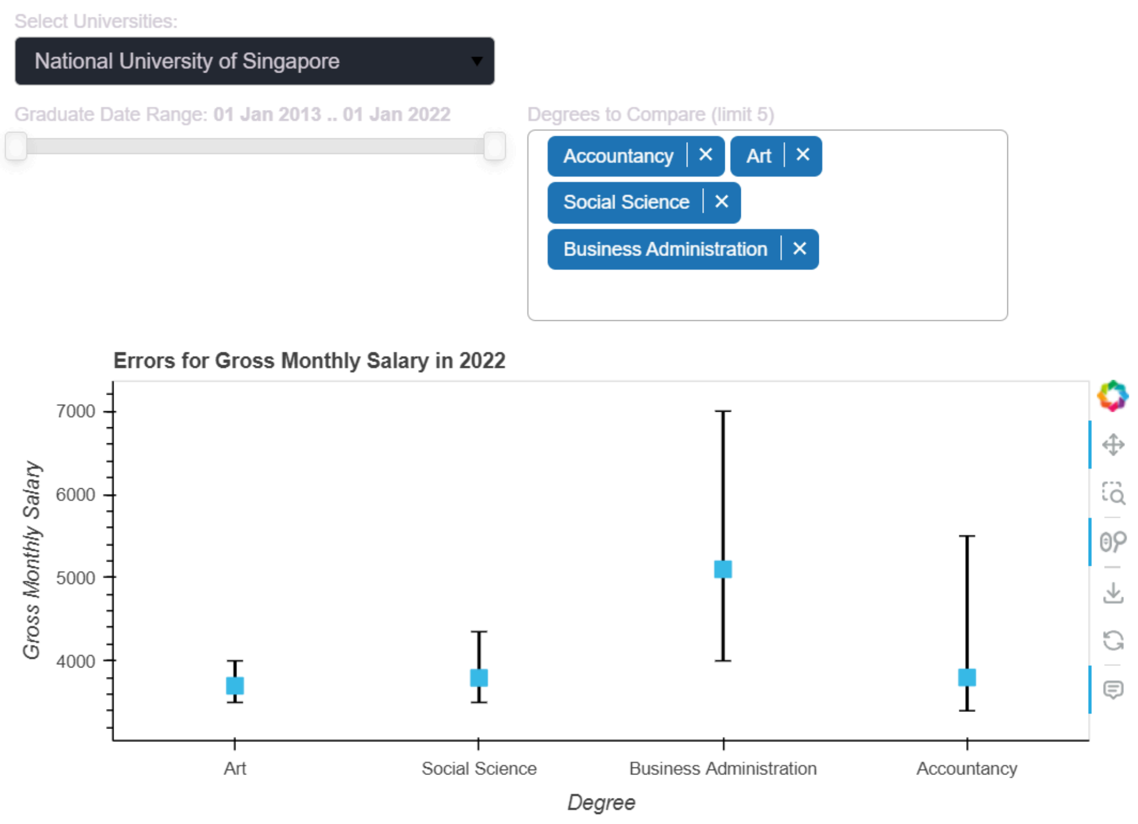
But from here we can see that the mean monthly salary for each degree has been increasing over time since 2014. Of note, Business Administration has seen the largest increase, while Accountancy experienced a sharp drop in 2021, but rebounded significantly in 2022.

Plot 2: Error Bars Plot

The error bars plot shows the 25th, median and 75th percentile for each degree, it is a simplified boxplot (here we are limited by the availability of data). This plot is good for showing the distribution of data across multiple categories. Showing a single median value might not provide a clear enough description of how wide the variation is.

It should allow a filter on the university and degree. This allows the user to have a better understanding of the distribution of the Gross Salary for each degree.

Refer to notebook for full code snippet


Select Universities: National University of Singapore

Graduate Date Range: 01 Jan 2013 .. 01 Jan 2022

Degrees to Compare (limit 5): Accountancy, Art, Social Science, Business Administration

Error Bars Plot

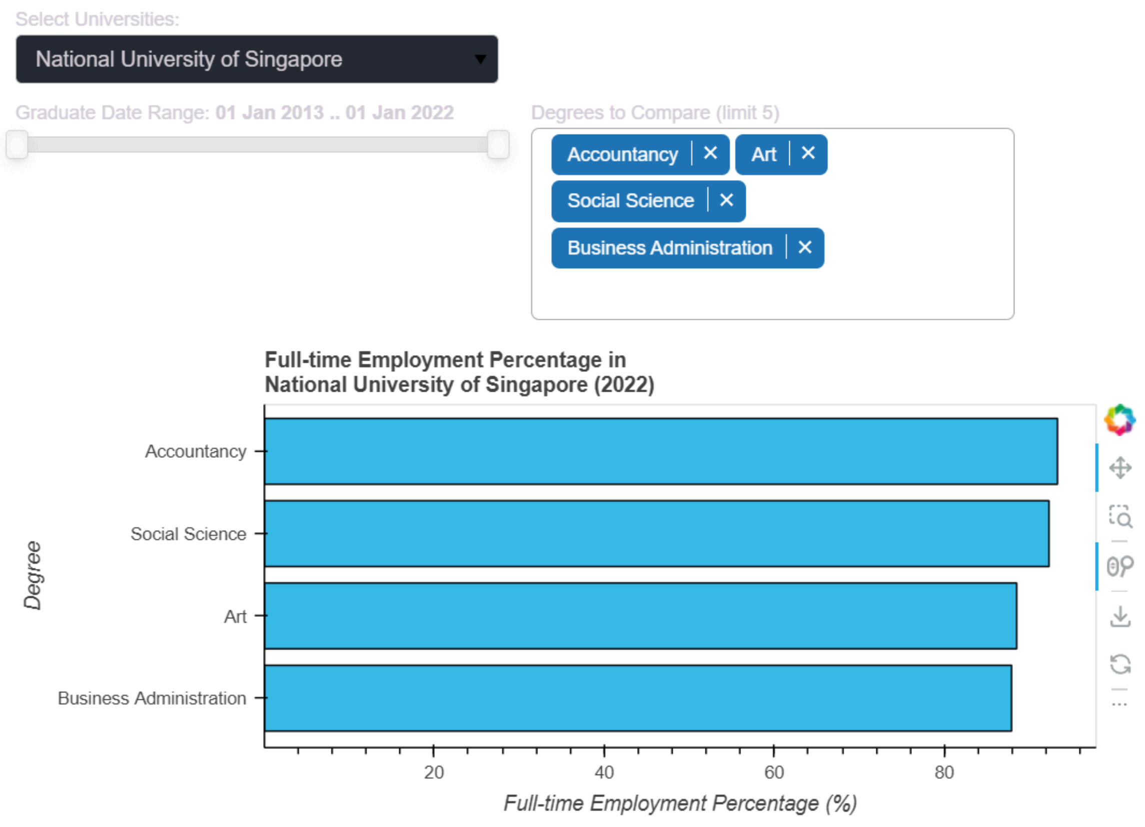
Even though Business Administration has the highest median Gross Monthly Salary amongst the listed degrees, the 75th percentile for Accountancy seems to overlap with the median value, which suggests that the higher paying Accountancy roles only hit about the median salary. Meanwhile, Art and Social Science have less variation in the salary offered to fresh graduates.

Plot 3: Bar Graph Plot

Next we'll build a bar graph to compare the employment rates between the various degrees in the **latest year** selected. The bar graph is a good way to compare numerical values side by side for each category, since it provides a common axis to compare them against.

The visualisation should filter on the university and degree. This will allow users to compare the various employment rates between each degrees without having to dive deeply and individually sort values.

Refer to notebook for full code snippet



Bar Graph Plot

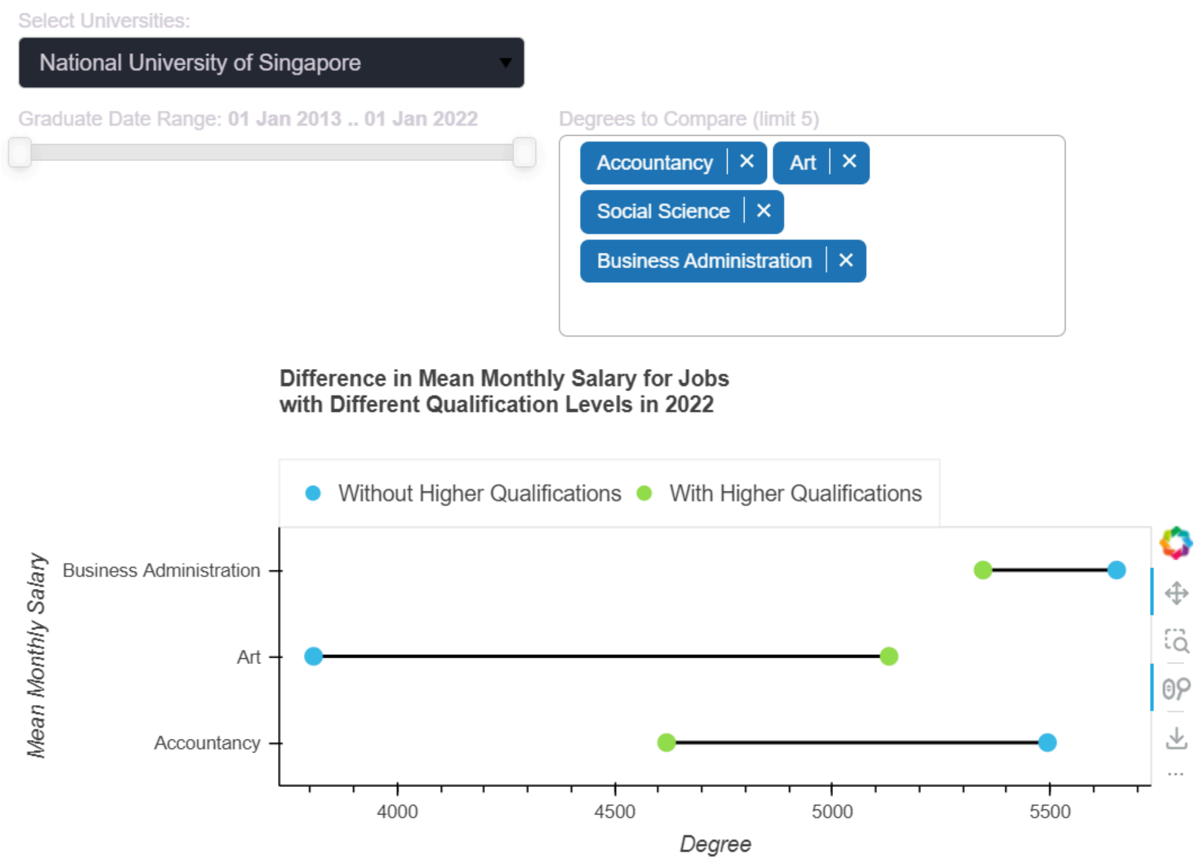
From this, although Business Administration had the highest median salary, it has the lowest Full-time Employment Percentage compared to the other degrees. Higher pay means lower employment since it costs more for the company to higher. But the overall employment rate is still relatively high >85% for each degree.

Plot 4: Dumbbell Plot

The dumbbell plot is used to emphasie gaps in values between 2 groups of data. In this case, we use it to showcase the difference between degrees that are honours degrees or cum laude achievement. Here we observe the mean salary difference.

This allows users to directly compare those degrees that have different tiers.

Refer to notebook for full code snippet


Select Universities:
National University of Singapore
Graduate Date Range: 01 Jan 2013 .. 01 Jan 2022
Degrees to Compare (limit 5)
Accountancy
Art
Social Science
Business Administration
Difference in Mean Monthly Salary for Jobs with Different Qualification Levels in 2022
● Without Higher Qualifications ● With Higher Qualifications
Mean Monthly Salary
Business Administration
Art
Accountancy
4000
4500
5000
5500
Degree

Dumbbell Plot

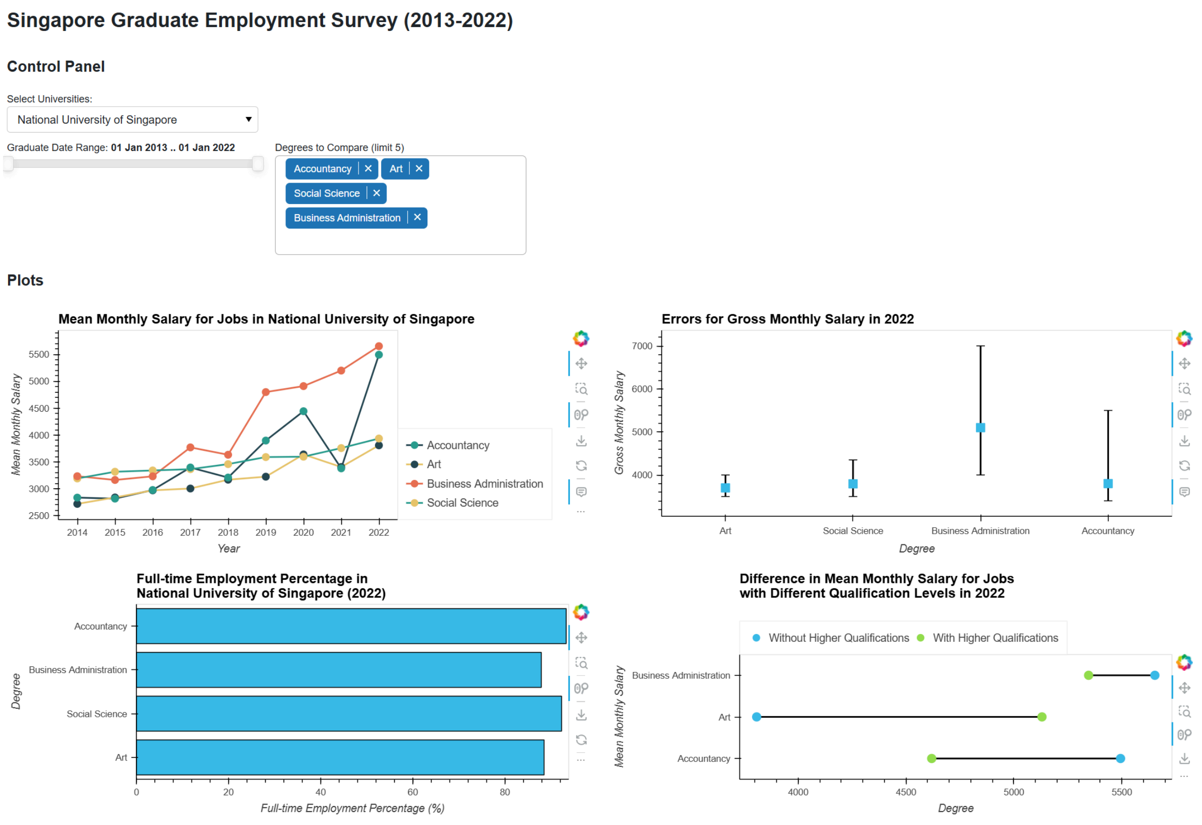
Here Social Science is not included since it doesn't have an honours program or data for it's cum laude graduates. However, funnily enough the degrees with higher qualifications have lower Mean Monthly Salary (Business Administration and Accountancy)

Putting it All Together

Finally, we combine all the dashboards we have created into 1 large viewing plane. This simplifies the functionality for users to view all information with a single control panel.

The combined dashboard is formatted in a “grid” format. Organisation of the widget components and plots can be done using the `Row` and `Column` components in Panel.

```
layout = pn.Column(
  pn.Row("# Singapore Graduate Employment Survey (2013-2022)"),
  pn.Row("## Control Panel"),
  pn.Row(uni_select),
  pn.Row(year_slider, degree_multi_select),
  pn.Row("## Plots"),
  pn.Row(scatter_line_plot, error_bars_plot),
  pn.Row(bar_graphs_plot, dumbbell_plot)
)

layout
```


Singapore Graduate Employment Survey (2013-2022)

Control Panel

Select Universities: National University of Singapore

Graduate Date Range: 01 Jan 2013 .. 01 Jan 2022

Degrees to Compare (limit 5): Accountancy ✕, Art ✕, Social Science ✕, Business Administration ✕

Plots

Mean Monthly Salary for Jobs in National University of Singapore
Mean Monthly Salary
Year
Accountancy
Art
Business Administration
Social Science
Errors for Gross Monthly Salary in 2022
Gross Monthly Salary
Degree
Art
Social Science
Business Administration
Accountancy
Full-time Employment Percentage in National University of Singapore (2022)
Degree
Full-time Employment Percentage (%)
Difference in Mean Monthly Salary for Jobs with Different Qualification Levels in 2022
Without Higher Qualifications
With Higher Qualifications
Mean Monthly Salary
Degree

Completed dashboard

Each plot provides details about the graduate salary for each degree, with respect to the school that offers it. The Scatter + Line plot provides the historical information on salary performance, which although leaves it slightly isolated from the rest of the plots provides key insights on the degrees performance over time. The remaining plot focus on the latest information (latest year) for each degree and how they stack up against each other.

Providing comparisons on employability, dsitribution of salaries and whether a higher qualification will land you higher pay.

All in all, this dashboard serves as a good view for prospective students to compare degrees that they are interested in for any of the available universities in Singapore.

Limitations

🚨 **ALERT: There is a `UnknownReferenceError` that pops up every time the date values are changed. This is a known issue that has not been resolved when working with different date steps. Please refresh the notebook if the error logs get too long**

The greatest value in the HoloViews ecosystem of packages is that they are very extensible with many connecting visualisation libraries both internally and externally (e.g seaborn, matplotlib). It also has a low skill floor, making it very easy to pickup and get going. However, a major weakness is in it's customisation options for both widgets and plots using their functional API.

For example, when creating the scatter plot, I was unable to apply two levels of categorical filters (colour and shape) to the plot, forcing me to limit each degree to a single University. For widgets, I am unable to control the step for the `DateRangeFilter` to step for each year since the `step` parameter is only limited to integer values.

Furthermore, there are many error messages that need to be suppressed to create a clean dashboard.

A consolidated list of limitations for this implementation of the dashboard:

- Unable to compare between degrees from different Universities
- Data cleaning limitations - some degrees are the same but have different names
- Limited to 5 degrees due to space constraint
- Limited to 1 year for some plots due to the plots nature

Future Works

Suggestions to improve dashboard

- Provide an additional filter layer to compare between Universities
- Invest more time into data cleaning efforts to ensure better comparisons
- `hvplot` has other library extensions that can potentially be used for more detailed plots