Analysis of the trends of post-secondary education in Singapore

PDS CA2

Done by: Lim Chuan Hao(1922264)
DIT/FT/1B/11

Datasets used

1. Enrolment - Secondary, By Level and Course (Dataset 1)
   a. https://data.gov.sg/dataset/enrolment-secondary-by-level
2. Enrolment - Pre-University, By Level and Course (Dataset 2)
   a. https://data.gov.sg/dataset/enrolment-pre-university-by-level
3. Polytechnics - Intake, Enrolment and Graduates by Course (Dataset 3)
   a. https://data.gov.sg/dataset/polytechnics-intake-enrolment-and-graduates-by-course
4. Universities - Intake, Enrolment and Graduates by Course (Dataset 4)
   a. https://data.gov.sg/dataset/universities-intake-enrolment-and-graduates-by-course
5. Graduate Employment Survey - NTU, NUS, SIT, SMU & SUTD (Dataset 5)
   a. https://data.gov.sg/dataset/graduate-employment-survey-ntu-nus-sit-smu-sutd

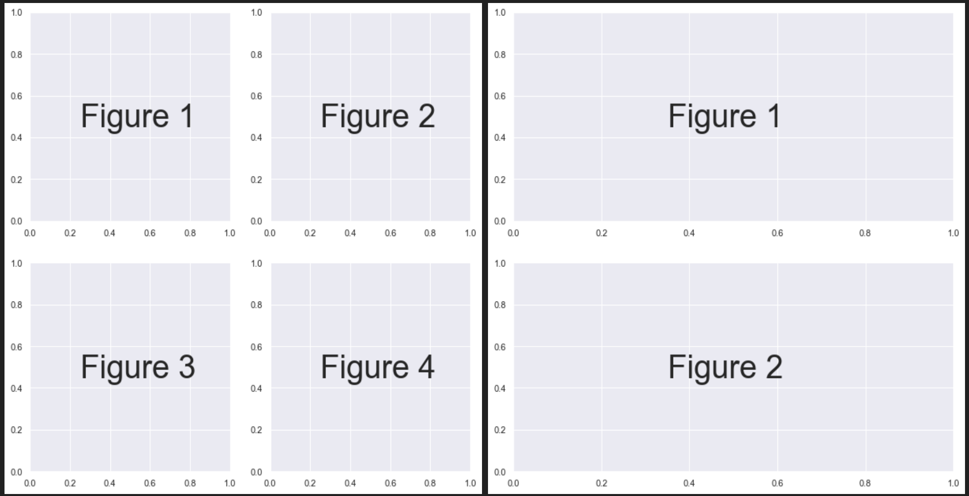
Sub-objectives explained

There are 2 main sub-objectives crafted to analyze the overall objective

1. Junior College (JC) or Polytechnic (Datasets 1, 2 and 3)
    a. Which is more popular?
    b. Which is harder to graduate?
2. Which field of study should I go into (Datasets 3, 4 and 5)
    a. Which field of study is more popular?
    b. Which field of study has better job prospects?

Some notes

I will be referring to the figures in the order as shown below



A more detailed explanation of the code and conclusion will be in the notebook.

Example shown below:

```
# Making sure the original dataset is not touched
post_sec_all = ori_post_sec_all

# Seem the data we thought would be int is actually a string
# We realise this is because there is - in some of the values of enrolment_preu
# Cleaning these values by setting them to 0
post_sec_all = post_sec_all.replace({'-': '0'})
# Changing the type back to int
post_sec_all = post_sec_all.astype({'enrolment_preu': 'float32'})
# Getting the mf_filter to get totals only
mf_filter = post_sec_all['sex'].isin(['MF'])
# Getting only JC 1 data
jc_1_filter = post_sec_all['level'].isin(['Junior College 1'])
jc_1_filter = jc_1_filter & mf_filter
all_jc_1 = post_sec_all[jc_1_filter]
# Getting Only JC 2 data
jc_2_filter = post_sec_all['level'].isin(['Junior College 2'])
jc_2_filter = jc_2_filter & mf_filter
all_jc_2 = post_sec_all[jc_2_filter]

# Setting up the JC intake first, then calculate the graduate %
# jc 1 grping
grp_jc_1 = all_jc_1.groupby(['year'])['enrolment_preu'].sum().reset_index()
grp_jc_1 = grp_jc_1.rename(columns={'enrolment_preu': 'jc_intake'})
# jc 2 grping
grp_jc_2 = all_jc_2.groupby(['year'])['enrolment_preu'].sum().reset_index()
grp_jc_2 = grp_jc_2.rename(columns={'enrolment_preu': 'promoted'})
# Getting the year they promoted from
grp_jc_2['promo_year'] = grp_jc_2['year'] - pd.offsets.DateOffset(years=1)
# Dropping year col
grp_jc_2 = grp_jc_2.drop(['year'], axis=1)
df_jc = pd.merge(grp_jc_1, grp_jc_2, how='inner', left_on=['year'], right_on=['promo_year'])
# Dropping dup promo_year
df_jc = df_jc.drop(['promo_year'], axis=1)
df_jc['promo_per'] = df_jc['promoted'] / df_jc['jc_intake']
df_jc = df_jc.drop(['promoted'], axis=1)
df_jc.head()
```

Sub-objective 1(Datasets 1, 2 and 3)

1. Junior College (JC) or Polytechnic (Datasets 1, 2 and 3)
    a. Which is more popular?
    b. Which is harder to graduate?

Some quick analysis of the datasets

1. Enrolment - Secondary, By Level and Course (Dataset 1)
    a. 5 Columns
        i. ['year', 'level', 'course', 'sex', 'enrolment_secondary']
    b. 1318 Rows
2. Enrolment - Pre-University, By Level and Course (Dataset 2)
    a. 5 Columns
        i. ['year', 'level', 'course', 'sex', 'enrolment_preu']
    b. 1166 Rows
3. Polytechnics - Intake, Enrolment and Graduates by Course (Dataset 3)
    a. 6 Columns
        i. ['year', 'sex', 'course', 'intake', 'enrolment', 'graduates']
    b. 336 Rows

An example of the output:

How does the raw data look like?

	year	level	course	sex	enrolment_secondary
0	1980-01-01	Secondary 1	Express	MF	45489
1	1980-01-01	Secondary 1	Express	F	22509
2	1980-01-01	Secondary 1	Normal (Acad)	MF	0
3	1980-01-01	Secondary 1	Normal (Acad)	F	0
4	1980-01-01	Secondary 1	Normal (Tech)	MF	0

Summary of the data at a glance

	year	level	course	sex	enrolment_secondary
count	1318	1318	1318	1318	1318.000000
unique	39	5	4	2	NaN
top	1980-01-01 00:00:00	Secondary 1	Normal (Acad)	MF	NaN
freq	34	310	390	659	NaN
first	1980-01-01 00:00:00	NaN	NaN	NaN	NaN
last	2018-01-01 00:00:00	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	NaN	8017.500000
std	NaN	NaN	NaN	NaN	8187.688607
min	NaN	NaN	NaN	NaN	0.000000
25%	NaN	NaN	NaN	NaN	1978.000000
50%	NaN	NaN	NaN	NaN	5624.000000
75%	NaN	NaN	NaN	NaN	12462.750000
max	NaN	NaN	NaN	NaN	45489.000000

```
More technical info of the dataset
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1318 entries, 0 to 1317
Data columns (total 5 columns):
year                   1318 non-null datetime64[ns]
level                  1318 non-null object
course                 1318 non-null object
sex                    1318 non-null object
enrolment_secondary    1318 non-null int64
dtypes: datetime64[ns](1), int64(1), object(3)
memory usage: 51.6+ KB
None
```

Data manipulation and Cleaning (Dataset 1)

1. Mainly for this dataset, I wanted to find out the number of eligible students that can move on to post-secondary education (JC or Poly)
    a. For this I mainly assumed the Normal (Academic) Secondary 4 and 5 and Express Secondary 4 students that graduated every year were able to move on to post-secondary education
    b. I filtered for the students and then further filtered it to only be taking the totals of male and females.
    c. Then I grouped by year and summed up the total graduates, to get the total number of students eligible for post-secondary education.
    d. Then I offset the year by 1 to make sure the previous year graduates were the total potential intake of next year

Limitations:

2. I assumed the total number of students that were eligible for post-secondary education as above
3. I also only assumed they could have gone for JC or Poly, there are other options I did not account for.

Data manipulation and Cleaning (Dataset 1)

```
# Displaying before
print('Before')
display(ori_sec_enrol.head())

# Making sure the original dataset is not touched
sec_enrol = ori_sec_enrol

# Filtering for only Secondary 4 express students and Secondary 5 students
sec_4_filter = sec_enrol['level'].isin(['Secondary 4'])
express_filter = sec_enrol['course'].isin(['Express', 'Normal (Acad)'])
sec_5_grad_filter = sec_enrol['level'].isin(['Secondary 5'])
sec_4_grad_filter = sec_4_filter & express_filter
# Only total filter
mf_filter = sec_enrol['sex'].isin(['MF'])
all_sec_grad_filter = sec_4_grad_filter | sec_5_grad_filter
all_sec_grad_filter = all_sec_grad_filter & mf_filter
all_sec_grad = sec_enrol[all_sec_grad_filter]
# Grouping them by year and getting total grads by years
df_sec_grad = all_sec_grad.groupby(['year'])['enrolment_secondary'].sum().reset_index()
# Getting enrolment_year
df_sec_grad['enrol_year'] = df_sec_grad['year'] + pd.offsets.DateOffset(years=1)
df_sec_grad = df_sec_grad.drop(['year'], axis=1)

# Showing the data after the manipulation
print('After')
display(df_sec_grad.head())
```

Before

	year	level	course	sex	enrolment_secondary
0	1980-01-01	Secondary 1	Express	MF	45489
1	1980-01-01	Secondary 1	Express	F	22509
2	1980-01-01	Secondary 1	Normal (Acad)	MF	0
3	1980-01-01	Secondary 1	Normal (Acad)	F	0
4	1980-01-01	Secondary 1	Normal (Tech)	MF	0

After

	enrolment_secondary	enrol_year
0	32925	1981-01-01
1	33931	1982-01-01
2	33938	1983-01-01
3	38023	1984-01-01
4	47677	1985-01-01

Data manipulation and Cleaning (Dataset 2)

1. I took the enrolled year 1 JC students as that year’s intake of students
2. I also took the current year 2 students enrolled and took them as having passed promos
    a. I then compared it to the previous year ‘intake’ to see the passing %
3. I will be using passing promo’s to compare with graduating Poly later on
4. Filtered similarly to dataset 1, took only total values and drop all null values

Data manipulation and Cleaning (Dataset 2)

```python
# Before
print('Before')
display(post_sec_all.head())

# Making sure the original dataset is not touched
post_sec_all = ori_post_sec_all

# Seem the data we thought would be int is actually a string
# We realise this is because there is - in some of the values of enrolment_preu
# Cleaning these values by setting them to 0
post_sec_all = post_sec_all.replace({'-': '0'})
# Changing the type back to int
post_sec_all = post_sec_all.astype({'enrolment_preu': 'float32'})
# Getting the mf_filter to get totals only
mf_filter = post_sec_all['sex'].isin(['MF'])
# Getting only JC 1 data
jc_1_filter = post_sec_all['level'].isin(['Junior College 1'])
jc_1_filter = jc_1_filter & mf_filter
all_jc_1 = post_sec_all[jc_1_filter]
# Getting Only JC 2 data
jc_2_filter = post_sec_all['level'].isin(['Junior College 2'])
jc_2_filter = jc_2_filter & mf_filter
all_jc_2 = post_sec_all[jc_2_filter]

# Setting up the JC intake first, then calculate the graduate %
# jc 1 grping
grp_jc_1 = all_jc_1.groupby(['year'])['enrolment_preu'].sum().reset_index()
grp_jc_1 = grp_jc_1.rename(columns={'enrolment_preu': 'jc_intake'})
# jc 2 grping
grp_jc_2 = all_jc_2.groupby(['year'])['enrolment_preu'].sum().reset_index()
grp_jc_2 = grp_jc_2.rename(columns={'enrolment_preu': 'promoted'})
# Getting the year they promoted from
grp_jc_2['promo_year'] = grp_jc_2['year'] - pd.offsets.DateOffset(years=1)
# Dropping year col
grp_jc_2 = grp_jc_2.drop(['year'], axis=1)
df_jc = pd.merge(grp_jc_1, grp_jc_2, how='inner', left_on=['year'], right_on=['promo_year'])
# Dropping dup promo_year
df_jc = df_jc.drop(['promo_year'], axis=1)
df_jc['promo_per'] = df_jc['promoted'] / df_jc['jc_intake']
df_jc = df_jc.drop(['promoted'], axis=1)

# After
print('After')
display(df_jc.head())
```

Before

	year	level	course	sex	enrolment_preu
0	1980-01-01	Junior College 1	Arts	MF	1158.0
1	1980-01-01	Junior College 1	Arts	F	903.0
2	1980-01-01	Junior College 1	Commerce	MF	1210.0
3	1980-01-01	Junior College 1	Commerce	F	995.0
4	1980-01-01	Junior College 1	Science	MF	3301.0

After

	year	jc_intake	promo_per
0	1980-01-01	5669.0	0.953255
1	1981-01-01	5323.0	1.136389
2	1982-01-01	5709.0	1.026449
3	1983-01-01	6510.0	1.061137
4	1984-01-01	7801.0	1.003205

Data manipulation and Cleaning (Dataset 3)

1. Same as the previous 2, I took only the intake and graduates
2. I also only filtered it to only have totals of both male and female
3. Cleaned up missing values to turn some columns from string back to floats
4. Then grouped by year to get intake and graduates by year
5. Assumed a normal graduate would graduate in 3 years, thus compared number of graduates to 3 years before

Data manipulation and Cleaning (Dataset 3)

```python
# Before
print('Before')
display(ori_poly_all)

# Making sure the original dataset is not touched
poly_all = ori_poly_all

# CLeaning up all missing values
for i in ['intake', 'enrolment', 'graduates']:
    poly_all[i] = poly_all[i].str.replace(',', '')
    poly_all[i] = poly_all[i].str.replace('-', '0')
# Changing the type back to int
poly_all = poly_all.astype({'intake': 'float32', 'enrolment': 'float32', 'graduates': 'float32'})
# Getting the only total filters of MF
mf_filter = poly_all['sex'].isin(['MF'])
# Filtering out for only totals
all_poly = poly_all[mf_filter]
# Grouping the data by year
grp_all_poly = all_poly.groupby(['year'])['intake', 'graduates'].sum().reset_index()

# Calculate when the graduates year intake year was
poly_grads = grp_all_poly[['year', 'graduates']].copy()
poly_grads['intake_year'] = poly_grads['year'] - pd.offsets.DateOffset(years=3)
poly_grads = poly_grads.drop(['year'], axis=1)

# Merging the dataset together
grp_all_poly = grp_all_poly.drop(['graduates'], axis=1)
df_poly = pd.merge(grp_all_poly, poly_grads, how='inner', left_on=['year'], right_on=['intake_year'])
df_poly = df_poly.drop(['intake_year'], axis=1)
df_poly['graduate_per'] = df_poly['graduates'] / df_poly['intake']
df_poly = df_poly.drop(['graduates'], axis=1)

# After
print('After')
display(df_poly.head())
```

Before

	year	sex	course	intake	enrolment	graduates
0	2005-01-01	MF	Applied Arts	1128	2593	550
1	2005-01-01	F	Applied Arts	687	1538	302
2	2005-01-01	MF	Architecture, Building & Real Estate	515	1466	425
3	2005-01-01	F	Architecture, Building & Real Estate	312	870	249
4	2005-01-01	MF	Business & Administration	3483	10143	3044
...	...	...	...	...	...	...
331	2018-01-01	F	Mass Communication	500	1420	437
332	2018-01-01	MF	Natural, Physical & Mathematical Sciences	1232	3794	1353
333	2018-01-01	F	Natural, Physical & Mathematical Sciences	773	2308	825
334	2018-01-01	MF	Services	1083	3279	1053
335	2018-01-01	F	Services	436	1436	541

336 rows × 6 columns

After

	year	intake	graduate_per
0	2005-01-01	20906.0	0.923993
1	2006-01-01	22276.0	0.907883
2	2007-01-01	23362.0	0.917944
3	2008-01-01	24838.0	0.922699
4	2009-01-01	25624.0	0.936583

Final dataframe used here

1. Mainly merged all the datasets above into one large dataframe
2. Mainly used the time as the matching key to merge
3. Carefully removed any columns not being used and also rename columns to not get confused

```python
# Renaming some columns to be more specific
df_poly = df_poly.rename(columns={'intake': 'poly_intake'})
df_sec_grad = df_sec_grad.rename(columns={'enrolment_secondary': 'total_sec_grads'})

# Making the main DF I would be using
df_jc_poly = pd.merge(df_sec_grad, df_jc, left_on=['enrol_year'], right_on=['year'], how='inner')
df_jc_poly = df_jc_poly.drop(['enrol_year'], axis=1)
df_jc_poly = pd.merge(df_jc_poly, df_poly, how='inner', on='year')

# Calculating ratio that went to jc or poly
df_jc_poly['jc_intake_per'] = df_jc_poly['jc_intake'] / df_jc_poly['total_sec_grads']
df_jc_poly['poly_intake_per'] = df_jc_poly['poly_intake'] / df_jc_poly['total_sec_grads']
df_jc_poly['year_str'] = df_jc_poly['year'].dt.strftime('%Y')
df_jc_poly
```

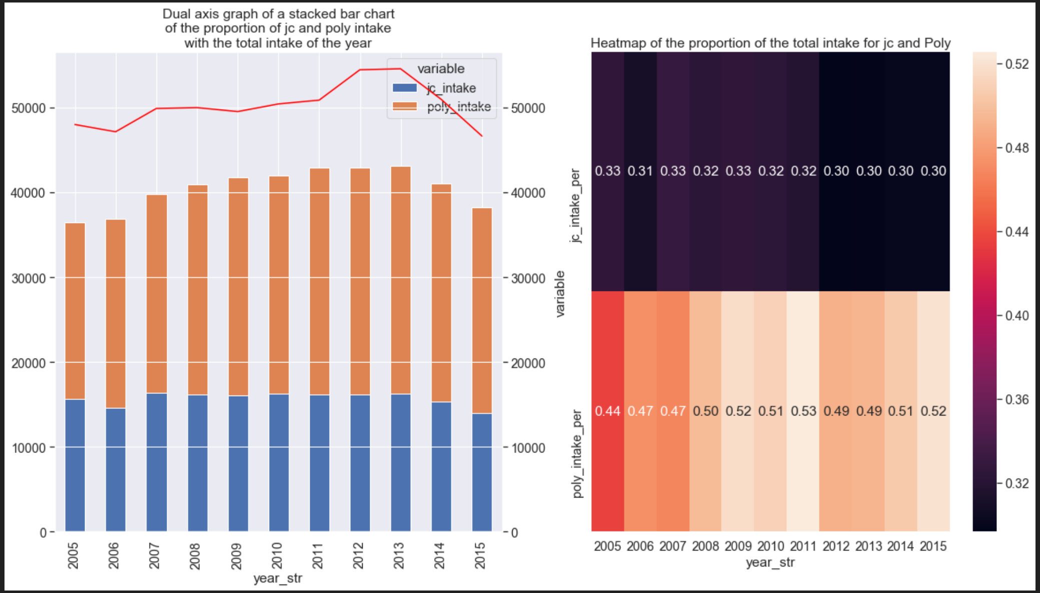
	total_sec_grads	year	jc_intake	promo_per	poly_intake	graduate_per	jc_intake_per	poly_intake_per	year_str
0	48015	2005-01-01	15616.0	0.949091	20906.0	0.923993	0.325232	0.435406	2005
1	47161	2006-01-01	14633.0	0.933780	22276.0	0.907883	0.310278	0.472339	2006
2	49908	2007-01-01	16435.0	0.904411	23362.0	0.917944	0.329306	0.468101	2007
3	50007	2008-01-01	16148.0	0.900855	24838.0	0.922699	0.322915	0.496690	2008
4	49542	2009-01-01	16121.0	0.913343	25624.0	0.936583	0.325401	0.517218	2009
5	50445	2010-01-01	16327.0	0.904698	25707.0	0.939705	0.323659	0.509605	2010
6	50891	2011-01-01	16195.0	0.905156	26737.0	0.924599	0.318229	0.525378	2011
7	54472	2012-01-01	16155.0	0.903807	26754.0	0.920647	0.296574	0.491151	2012
8	54592	2013-01-01	16261.0	0.916364	26879.0	0.933963	0.297864	0.492362	2013
9	50979	2014-01-01	15337.0	0.928082	25777.0	0.939209	0.300849	0.505640	2014
10	46654	2015-01-01	14043.0	0.934202	24251.0	0.932498	0.301003	0.519805	2015

Analyzing the graphs

1. Figure 1
    a. We can see that there is a larger majority of people going to poly instead of JC
    b. We can also see that the people choosing to go to Poly is increasing over the years
    c. However, note how after 2012, the total number of people that can go into post-secondary education has dropped, so with the total number of JC and Poly intake
2. Figure 2
    a. We can see here that by proportion, JC has been steadily decreasing in intake year by year.
    b. We can also see that in contrast, the proportion choosing to go to poly has been increasing.

Conclusions:

3. We can see clearly that Polytechnic is gaining popularity as more people are choosing Polytechnic over JC
4. However, the total number of students that can go into post-secondary education has been decreasing in recent years.

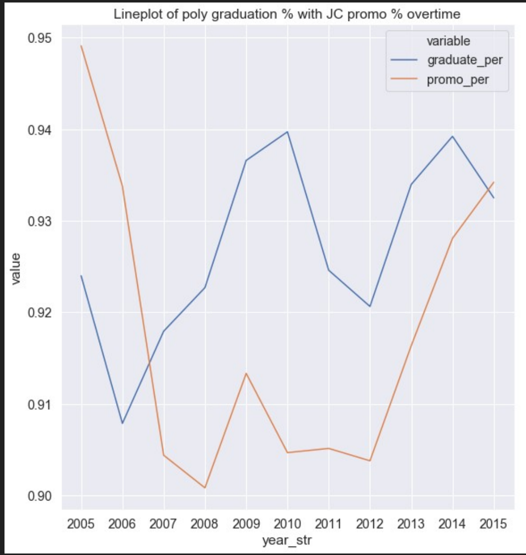


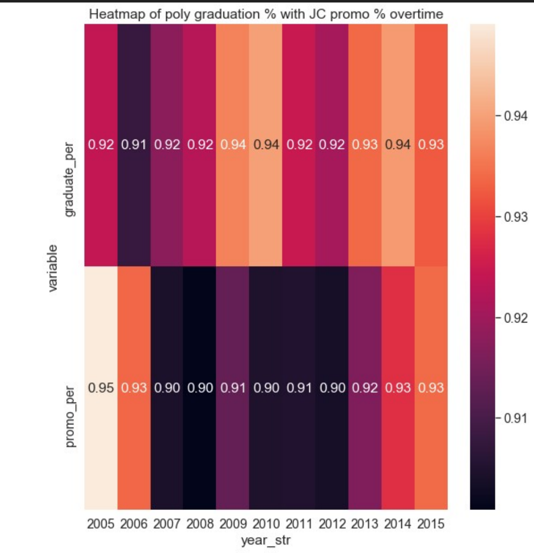
Analyzing the graphs

1. Figure 1
    a. We can see that mostly poly was much easier to graduate as compared to JC passing promos
    b. Note the axis as the actual difference is about 4% of difference
2. Figure 2
    a. Snae thing here as we can see the gradients gradually become the same tone of orange at the end

Conclusions:

3. We can see that at the start, poly was noticeably easier compared to JC by about 3%, however nowadays, we can see that the difficulty in either institute is about the same.





Final conclusions

Sub-Objective: Junior College (JC) or Polytechnic (Datasets 1, 2 and 3)

1. Which is more popular?
   a. We can see that poly is definitely more popular compared to JCs, both in terms of shear number as well as proportion.
   b. Both actually take up the majority of the students after they graduate from secondary school. (80% and above)
   c. Note the limitations
2. Which is harder to graduate?
   a. We can see that poly is slightly easier in the past to graduate, but nowadays it is about the same
   b. Note the limitations of comparing graduating poly to passing promos in JC

Sub-objective 2(Datasets 3, 4 and 5)

1. Which field of study should I go into (Datasets 3, 4 and 5)
    a. Which field of study is more popular?
    b. Which field of study has better job prospects?

Data manipulation and Cleaning (Dataset 3)

1. Firstly, I cleaned up all the missing values as before and took only the total of male and females
2. Then I mapped a new column of the course to their field of study
3. For this and subsequent datasets I have the following field of studies
   a. Arts
   b. Computing
   c. Engineering
   d. Medicine
   e. Science
   f. Business
   g. Law
   h. Education
4. I then grouped them by their field of study before getting the totals for things like intake

Data manipulation and Cleaning (Dataset 3)

```python
# Before
print('Before')
display(ori_poly_all)

# Making sure the original dataset is not touched
poly_all = ori_poly_all

# CLeaning up all missing values
for i in ['intake', 'enrolment', 'graduates']:
    poly_all[i] = poly_all[i].str.replace(',', '')
    poly_all[i] = poly_all[i].str.replace('-', '0')
# Changing the type back to int
poly_all = poly_all.astype({'intake': 'float32', 'enrolment': 'float32', 'graduates': 'float32'})
# Mapping to find the field based on course
course_field_map = {
'Applied Arts': 'arts',
'Architecture, Building & Real Estate': 'arts',
'Business & Administration': 'business',
'Education': 'education',
'Engineering Sciences': 'engineering',
'Health Sciences': 'medicine',
'Humanities & Social Sciences': 'arts',
'Information Technology': 'computing',
'Law': 'law',
'Mass Communication': 'business',
'Natural, Physical & Mathematical Sciences': 'science',
'Services': 'nil'}
poly_all['field'] = poly_all['course'].replace(course_field_map)
# Getting the only total filters of MF
mf_filter = poly_all['sex'].isin(['MF'])

# Getting only totals
df_poly = poly_all[mf_filter]
# Dropping all nil/na values
df_poly = df_poly.replace({'nil': None})
df_poly = df_poly.dropna()
# Dropping columns we will not be using
df_poly = df_poly.drop(['sex', 'course'], axis=1)
# Creating the df by grouping
df_poly = df_poly.groupby(['year', 'field'])['intake', 'enrolment', 'graduates'].sum().reset_index()
df_poly = df_poly.rename(columns={'intake': 'poly_intake', 'enrolment': 'poly_enrol', 'graduates': 'poly_graduates', 'field': 'poly_field'})
# Creating the year_str column
df_poly['year_str'] = df_poly['year'].dt.strftime('%Y')

# After
print('After')
df_poly.head()
```

Before

	year	sex	course	intake	enrolment	graduates
0	2005-01-01	MF	Applied Arts	1128	2593	550
1	2005-01-01	F	Applied Arts	687	1538	302
2	2005-01-01	MF	Architecture, Building & Real Estate	515	1466	425
3	2005-01-01	F	Architecture, Building & Real Estate	312	870	249
4	2005-01-01	MF	Business & Administration	3483	10143	3044
...	...	...	...	...	...	...
331	2018-01-01	F	Mass Communication	500	1420	437
332	2018-01-01	MF	Natural, Physical & Mathematical Sciences	1232	3794	1353
333	2018-01-01	F	Natural, Physical & Mathematical Sciences	773	2308	825
334	2018-01-01	MF	Services	1083	3279	1053
335	2018-01-01	F	Services	436	1436	541

336 rows × 6 columns

After

	year	poly_field	poly_intake	poly_enrol	poly_graduates	year_str
0	2005-01-01	arts	1724.0	4142.0	975.0	2005
1	2005-01-01	business	3931.0	11569.0	3463.0	2005
2	2005-01-01	computing	4122.0	11607.0	3356.0	2005
3	2005-01-01	education	189.0	484.0	111.0	2005
4	2005-01-01	engineering	7826.0	22462.0	6536.0	2005

Data manipulation and Cleaning (Dataset 4)

1. Same as dataset 3, cleaned the values and made columns like intake into floats
2. The mapped the course to their field of study
3. Only got the totals and grouped them by year and field of study
4. Also removed null fields that were not considered

Data manipulation and Cleaning (Dataset 4)

```python
# Before
print('Before')
display(ori_uni_all.head())

# Making sure the original dataset is not touched
uni_all = ori_uni_all

# Changing all missing or null values
for i in ['intake', 'enrolment', 'graduates']:
    uni_all[i] = uni_all[i].str.replace(',', '')
    uni_all[i] = uni_all[i].str.replace('-', '0')
# Changing the type back to int
uni_all = uni_all.astype({'intake': 'float32', 'enrolment': 'float32', 'graduates': 'float32'})
# Getting the only total filters of MF
mf_filter = uni_all['sex'].isin(['MF'])
# Mapping based on course
course_field_map = {
'Accountancy': 'business',
'Architecture, Building & Real Estate': 'arts',
'Business & Administration': 'business',
'Dentistry': 'medicine',
'Education': 'education',
'Engineering Sciences': 'engineering',
'Fine & Applied Arts': 'arts',
'Health Sciences': 'medicine',
'Humanities & Social Sciences': 'arts',
'Information Technology': 'computing',
'Law': 'law',
'Mass Communication': 'business',
'Medicine': 'medicine',
'Natural, Physical & Mathematical Sciences': 'science',
'Services': 'nil'}
uni_all['field'] = uni_all['course'].replace(course_field_map)

# Getting only totals
df_uni_all = uni_all[mf_filter]
# Dropping all nil/na values
df_uni_all = df_uni_all.replace({'nil': None})
df_uni_all = df_uni_all.dropna()
df_uni_all = df_uni_all.drop(['sex', 'course'], axis=1)
# Groupping to get df
df_uni_all = df_uni_all.groupby(['year', 'field'])['intake', 'enrolment', 'graduates'].sum().reset_index()
df_uni_all = df_uni_all.rename(columns={'intake': 'uni_intake', 'enrolment': 'uni_enrol', 'graduates': 'uni_graduates', 'field': 'uni_field'})
# Getting the year_str column
df_uni_all['year_str'] = df_uni_all['year'].dt.strftime('%Y')

# After
print('After')
df_uni_all.head()
```

Before

	year	sex	course	intake	enrolment	graduates
0	2005-01-01	MF	Accountancy	876	2561	706
1	2005-01-01	F	Accountancy	530	1732	495
2	2005-01-01	MF	Architecture, Building & Real Estate	299	1310	180
3	2005-01-01	F	Architecture, Building & Real Estate	175	786	106
4	2005-01-01	MF	Business & Administration	1545	5013	1256

After

	year	uni_field	uni_intake	uni_enrol	uni_graduates	year_str
0	2005-01-01	arts	2695.0	8408.0	1852.0	2005
1	2005-01-01	business	2593.0	8218.0	2108.0	2005
2	2005-01-01	computing	773.0	2356.0	562.0	2005
3	2005-01-01	education	0.0	0.0	0.0	2005
4	2005-01-01	engineering	4028.0	16599.0	3859.0	2005

Data manipulation and Cleaning (Dataset 5)

1. For this, I also had to clean the dataset, removing and replace null values
2. Then I was able to turn the columns like employment_rate_overall into floats
3. Also made a new field column based on the school degree was from
4. Then grouped by year and field, taking the mean of values

Data manipulation and Cleaning (Dataset 5)

```
for i in col_change:
    uni_grad[i] = uni_grad[i].str.replace(',', '')
    uni_grad[i] = uni_grad[i].str.replace('-', '0')
    uni_grad[i] = uni_grad[i].str.replace('na', '0')
# Changing the type back to int
col_change_map = dict(zip(col_change, ['float32']*len(col_change)))
uni_grad = uni_grad.astype(col_change_map)
# Mapping schools to the field of study
school_field_mapping = {
'College of Business (Nanyang Business School)': 'business',
'College of Engineering': 'engineering',
'College of Humanities, Arts & Social Sciences': 'arts',
'College of Sciences': 'science',
'National Institute of Education (NIE)': 'education',
'Faculty of Arts & Social Sciences': 'arts',
'NUS Business School': 'business',
'School of Computing': 'computing',
'Faculty of Dentistry': 'medicine',
'School of Design & Environment': 'arts',
'Faculty of Engineering': 'engineering',
'Faculty of Law': 'law',
'YLL School of Medicine': 'medicine',
'Yong Siew Toh Conservatory of Music': 'arts',
'Faculty of Science': 'science',
'School of Accountancy (4-years programme) *': 'business',
'School of Business (4-years programme) *': 'business',
'School of Economics (4-years programme) *': 'business',
'School of Information Systems (4-years programme) *': 'computing',
'School of Social Sciences (4-years programme) *': 'arts',
'School of Law (4-years programme) *': 'law',
'School of Accountancy (4-year programme) *': 'business',
'School of Business (4-year programme) *': 'business',
'School of Economics (4-year programme) *': 'business',
'School of Information Systems (4-year programme) *': 'computing',
'School of Social Sciences (4-year programme) *': 'arts',
'School of Law (4-year programme) *': 'law',
'DigiPen Institute of Technology': 'computing',
'The Glasgow School of Art': 'arts',
'Newcastle University': 'science',
'Technische Universitt Mnchen': 'science',
'The Culinary Institute of America': 'arts',
'Trinity College Dublin': 'science',
'University of Glasgow': 'computing',
'University of Manchester': 'science',
'University of Nevada, Las Vegas': 'business',
'Wheelock College': 'medicine',
'Multidisciplinary Program': 'nil',
'na': 'nil',
'Singapore Institute of Technology -Trinity College Dublin': 'computing',
'Sports Science and Management': 'arts',
```

Before

	year	university	school	degree	employment_rate_overall	employment_rate_ft_perm	basic_monthly_mean	basic_monthly_median	gross_monthly_mean	gross_monthly_median
0	2013-01-01	Nanyang Technological University	College of Business (Nanyang Business School)	Accountancy and Business	97.4	96.1	3701	3200	3727	3350
1	2013-01-01	Nanyang Technological University	College of Business (Nanyang Business School)	Accountancy (3-yr direct Honours Programme)	97.1	95.7	2850	2700	2938	2700
2	2013-01-01	Nanyang Technological University	College of Business (Nanyang Business School)	Business (3-yr direct Honours Programme)	90.9	85.7	3053	3000	3214	3000
3	2013-01-01	Nanyang Technological University	College of Business (Nanyang Business School)	Business and Computing	87.5	87.5	3557	3400	3615	3400
4	2013-01-01	Nanyang Technological University	College of Engineering	Aerospace Engineering	95.3	95.3	3494	3500	3536	3500

After

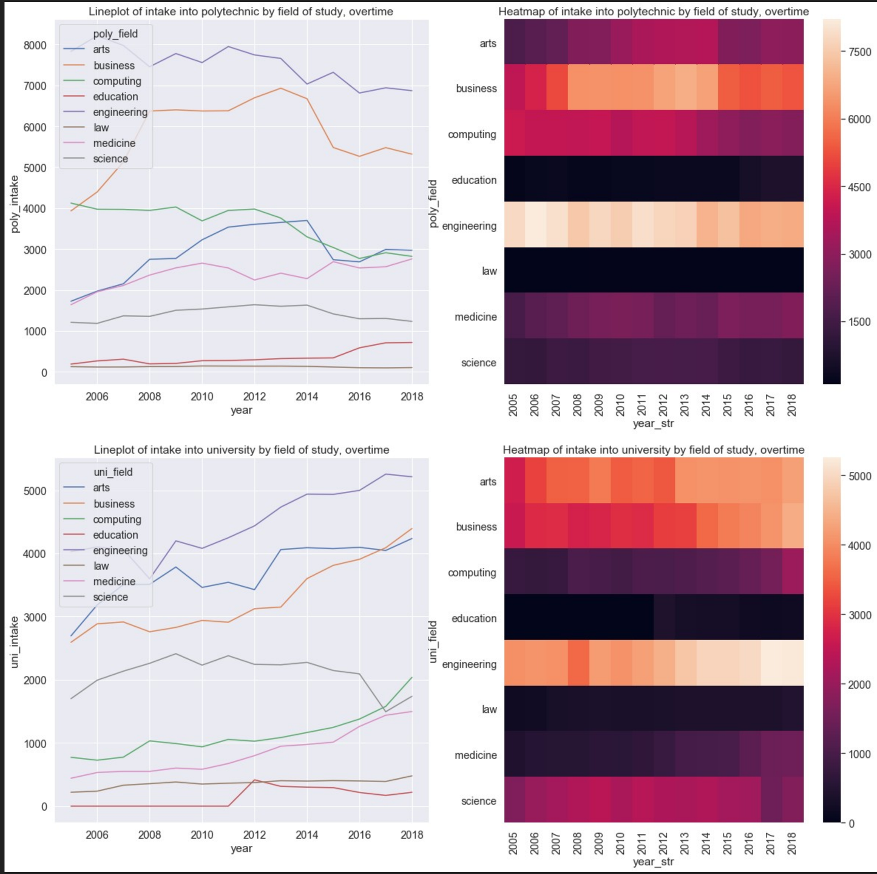
	year	field	employment_rate_overall	employment_rate_ft_perm	basic_monthly_mean	basic_monthly_median	gross_monthly_mean	gross_monthly_median	gross_mthly_25_percentile
0	2013-01-01	arts	76.538887	66.966667	2626.444336	2582.777832	2689.888916	2626.388916	2362.500000
1	2013-01-01	business	93.557144	90.885712	3316.428467	3044.142822	3424.071533	3121.000000	2832.142822
2	2013-01-01	computing	69.125000	63.587502	2625.125000	2434.375000	2662.750000	2457.500000	2265.625000
3	2013-01-01	education	100.000000	100.000000	3395.000000	3413.000000	3492.000000	3498.000000	3350.000000
4	2013-01-01	engineering	89.368179	85.445457	3163.409180	3056.818115	3251.136475	3126.590820	2959.681885

Analyzing the graphs

1. Figure 1 & 3
    a. We can see here that the trends for polytechnic and university by fields are quite different
    b. In polytechnic, engineering is the most popular followed by business
    c. But in University, its engineering followed by arts and business
    d. We can also see that the trends for different fields are also different.
2. Figure 2 & 3
    a. Gives a more detailed look, we can see by the gradient, that for poly, it is engineering followed by business then arts, computing and medicine
    b. Uni its, engineering then arts and business, followed by computing, medicine and science

Conclusions:

3. The more popular courses are engineering and business by far for both uni and polytechnic.
4. However it is hard to tell from the trends as for polytechnic, all of them are decreasing

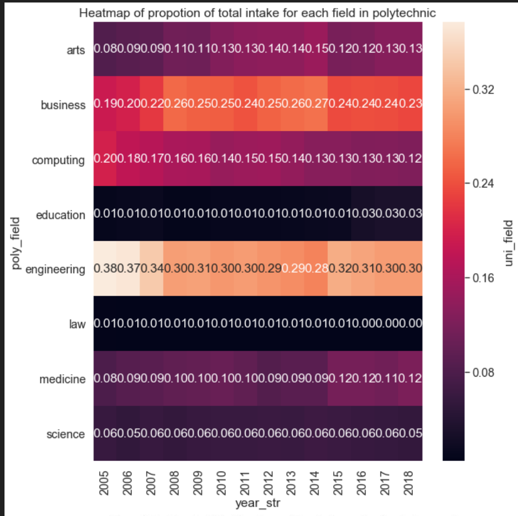


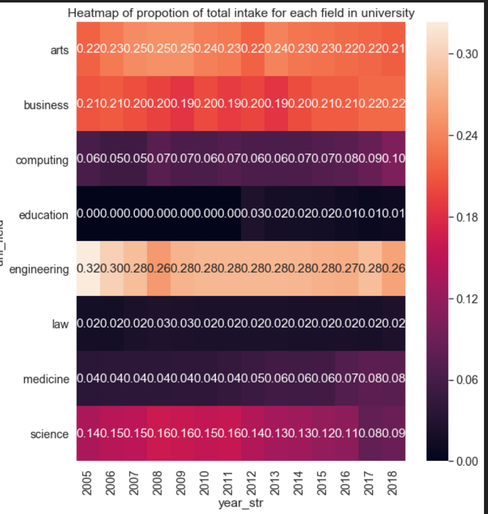
Analyzing the graphs

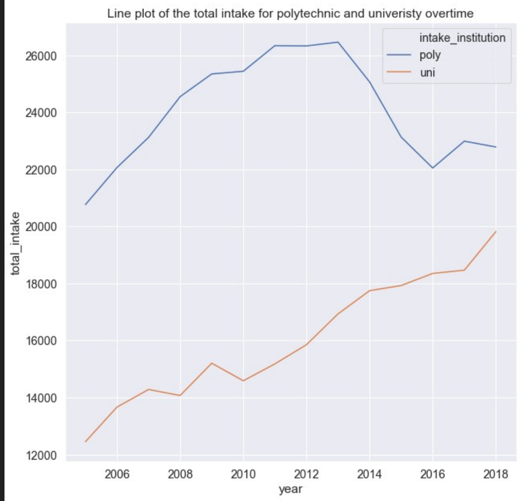
1. Figure 1 & 2
    a. We can see here that engineering is indeed the popular field of study
    b. However in uni, the popularity is actually decreasing as the proportion of intake is decreasing.
2. Figure 3
    a. We can also see that the overall increase in intake for uni and decrease of poly
    b. Thus even though we saw the downward trend for all fields of study in poly, it is probably due to this

Conclusion:

3. Thus we can conclude that engineering is the most popular field of study, however this may not tru in the future, with business and arts increasing in popularity

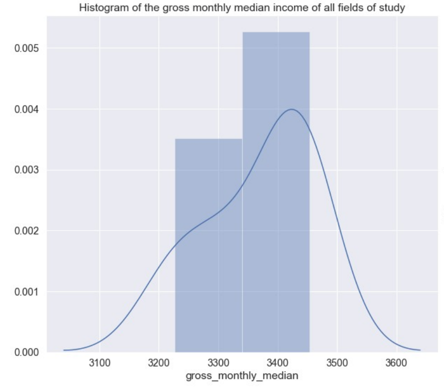


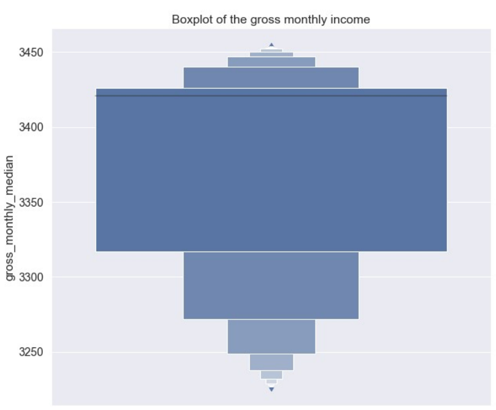


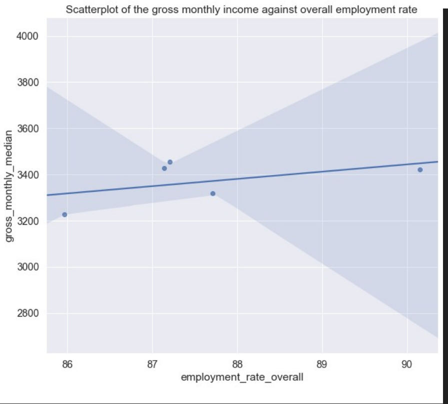


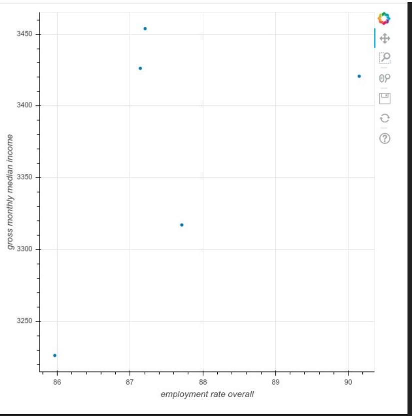
Analyzing the graphs

1. Figure 1, 2, 3 and 4
    a. Some simple plots, we can see that the gross monthly income is normally distributed
    b. The scatter plot shows some minor relationships
    c. The boxplot shows there is a large range and variation







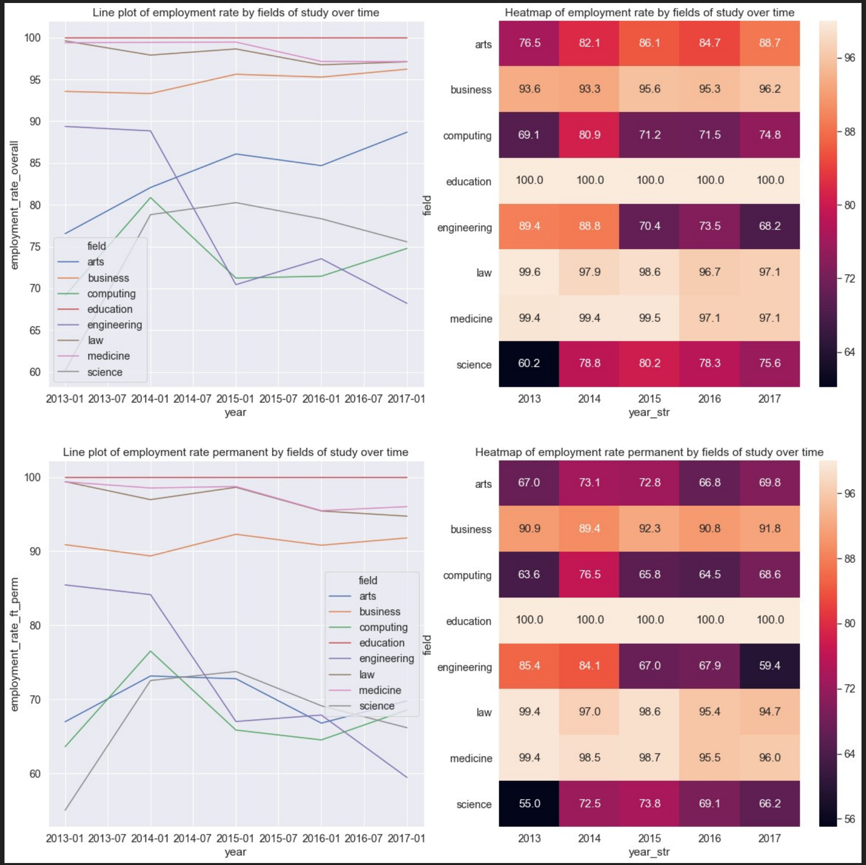


Analyzing the graphs

1. Figure 1 & 3
   a. We can clearly see that for some fields, the employment rate and permanent employment rate is different.
2. Figure 2 & 4
   a. We can see the most secure job is education
   b. Close seconds are law, medicine and business
   c. Although business permanent employment rate is much lower

Conclusion:

3. The field of study you take can significantly vary you employment rate
4. This also changes year after year

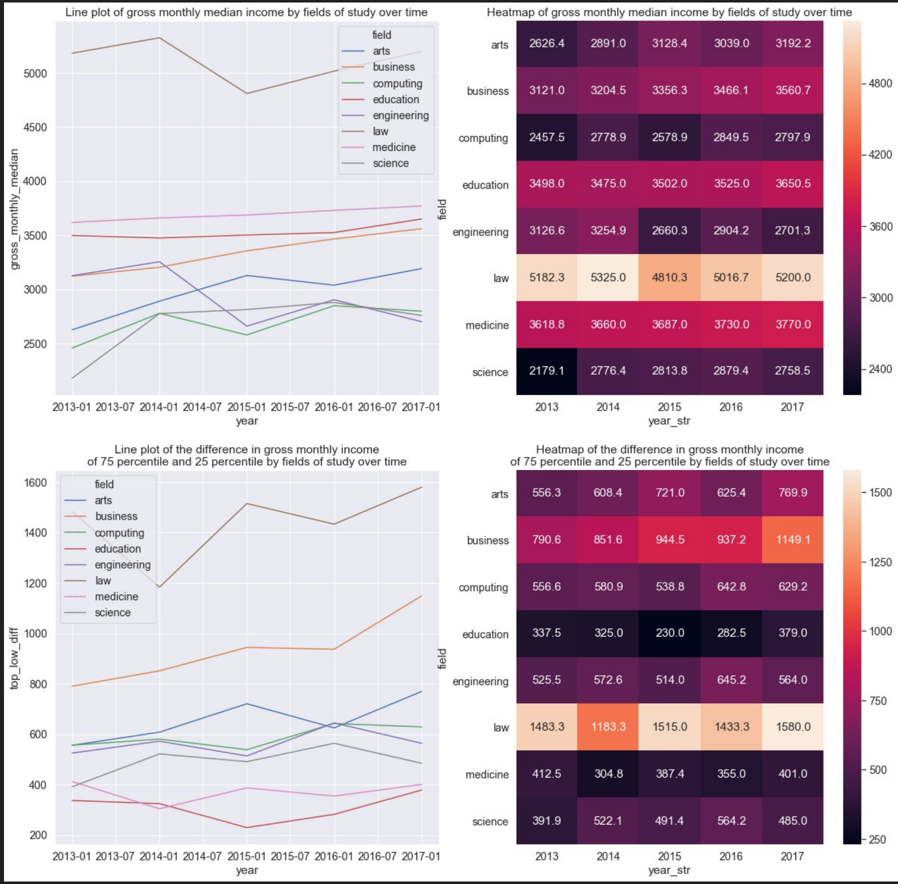


Analyzing the graphs

1. Figure 1 & 2
    a. We can see that the highest earner by far is law
    b. With medicine, education and business coming in together
2. Figure 3 & 4
    a. We can see there is also a large difference in the income of law students, meaning their pay is very much affected by their school grades
    b. Same with business to a smaller extent
    c. Lastly we can see this also affects the other fields, although not as much
    d. Smallest difference is in the education field

Conclusion:

3. Different fields of study also earn different amounts of income
4. School grades also decides how much more you can earn depending on your field. (Law)



Final conclusions

Sub-Objective: Which field of study should I go into (Datasets 3, 4 and 5)

1. Which field of study is more popular?
   a. Most popular is engineering then business
2. Which field of study has better job prospects?
   a. Depending on what you are aiming for, the most stable is education.
      i. University grades does not really affect your income
   b. Best income is law
      i. Deping on how good your grades are you could earn up to 1.6K more than the median