This project is based on the dataset for the Udemy Data Analytics Project published on Medium. A member of the data team at Udemy had worked on the project with excel and visualised with PowerBI but I will be using python and PowerBI.

The scope of the project covers only four course categories: `Business Finance`, `Graphics Design`, `Musical Instruments` and `Web Development`.

The aim is to explore the number of courses for each subject area, the number of subscribers, how well the courses were rated and how much revenue is being generated etc. For each subject category, we will also identify the key words for the most best rated and most sought after courses..

We will find answers to the following questions and more:

- Which subject area has the highest and lowest number of published courses?
- Which subject area has the highest number of subscribers?
- What levels of courses are free?
- What words are common among the courses with high ratings and number of subscriptions?
- How well were the courses rated by subscribers?
- For each subject category, what year were most courses published?
- Which suject area contributes the most to the companies revenue?

Import Packages and Load Data

In [71]:
```
"""
Import the packages we need for our analysis.
"""

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import string, re, nltk
from nltk.corpus import stopwords
from wordcloud import wordcloud, STOPWORDS, ImageColorGenerator
from datetime import datetime
import plotly.express as px
import warnings
```

Business, Music, Design, Web Development Data

Using the read_csv built-in function, load the datasets onto jupyter notebook.

In [72]:
```
"""
Load the four different datasets into dataframes.
"""

df_buz = pd.read_csv('Business Courses.csv') #Business Finance
df_gfx = pd.read_csv('Design Courses.csv') #Graphics
df_mus = pd.read_csv('Music Courses.csv') #Musical Instrument
df_dev = pd.read_csv('Web development Courses.csv') #Web Development

#join the dataframes
df = pd.concat([df_buz, df_gfx, df_mus, df_dev])

#sample the first rows of the dataframe
print('\n')
print("This combined dataset has {} rows and {} columns".format(df.shape[0], df.shape[1]) )
df.head(2)
```

```
This combined dataset has 3681 rows and 12 columns
```

Out[72]:

	course_id	course_title	url	price	num_subscribers	num_reviews	num_lectures	level	Rating	content_duration	published_timestamp	subject
0	49798.0	Bitcoin or How I Learned to Stop Worrying and ...	https://www.udemy.com/bitcoin-or-how-i-learned...	0.0	65576.0	936.0	24.0	All Levels	0.56	8.0	2013-04-20T02:25:22Z	Business Finance
1	48841.0	Accounting in 60 Minutes - A Brief Introduction	https://www.udemy.com/accounting-in-60-minutes...	0.0	56659.0	4397.0	16.0	Beginner Level	0.95	1.5	2013-04-07T21:39:25Z	Business Finance

In [73]:
```
#summary statistics of the combined dataframe
df.describe()
```

Out[73]:

	course_id	price	num_subscribers	num_reviews	num_lectures	Rating	content_duration
count	3.676000e+03	3676.000000	3676.000000	3676.000000	3676.000000	3677.000000	3676.000000
mean	6.757535e+05	66.115343	3199.260881	156.309848	40.129761	0.610889	4.096137
std	3.431304e+05	61.056073	9486.582966	935.674518	50.398507	0.334244	6.054948
min	8.324000e+03	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.074740e+05	20.000000	112.000000	4.000000	15.000000	0.280000	1.000000
50%	6.876920e+05	45.000000	912.500000	18.000000	25.000000	0.760000	2.000000
75%	9.608140e+05	95.000000	2558.000000	67.000000	46.000000	0.930000	4.500000
max	1.282064e+06	200.000000	268923.000000	27445.000000	779.000000	1.000000	78.500000

Data Wrangling

In [74]:
```
#check data types and missing values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3681 entries, 0 to 1204
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   course_id            3676 non-null   float64
 1   course_title         3676 non-null   object 
 2   url                  3676 non-null   object 
 3   price                3676 non-null   float64
 4   num_subscribers      3676 non-null   float64
 5   num_reviews          3676 non-null   float64
 6   num_lectures         3676 non-null   float64
 7   level                3676 non-null   object 
 8   Rating               3677 non-null   float64
 9   content_duration     3676 non-null   float64
 10  published_timestamp  3676 non-null   object 
 11  subject              3677 non-null   object 
dtypes: float64(7), object(5)
memory usage: 373.9+ KB
```

In [75]:
```
#check for and drop duplicates
df[df.duplicated()]

#drop duplicates in the course id column
df.drop_duplicates(subset=['course_id'],inplace=True)
```

In [76]:
```
#drop rows with missing values
df.dropna(inplace=True)
```

In [79]:
```
#confirm there is no more any null values
df.isnull().sum()
```

Out[79]:
```
course_id              0
course_title           0
url                    0
price                  0
num_subscribers        0
num_reviews            0
num_lectures           0
level                  0
Rating                 0
content_duration       0
published_timestamp    0
subject                0
dtype: int64
```

In [81]:
```
"""
We do not need all the attributes for our analysis; therefore we choose only the
relevant ones to study and drop the rest.
"""
df.drop(columns = ['url','num_reviews', 'course_id', 'num_lectures','content_duration'],inplace=True)
```

We wouldn't be needing the columns below for our analysis; hence we shall drop them. We will change the column data types to the most suitable.

In [85]:
```
#cast datatype as int

df['num_subscribers'] = df['num_subscribers'].astype('int64')

#confirm change
df.head(2)
```

Out[85]:

	course_title	price	num_subscribers	level	Rating	published_timestamp	subject
0	Bitcoin or How I Learned to Stop Worrying and ...	0.0	65576	All Levels	0.56	2013-04-20T02:25:22Z	Business Finance
1	Accounting in 60 Minutes - A Brief Introduction	0.0	56659	Beginner Level	0.95	2013-04-07T21:39:25Z	Business Finance

In [92]:
```
#categorize courses as either free or paid
conditions = [
    (df['price'] == 0) ,
    (df['price'] > 0)]

values = ['free' , 'Paid']
df['price_group'] = np.select(conditions,values)

df['price_group'].value_counts()
```

Out[92]:
```
Paid    3362
free     310
Name: price_group, dtype: int64
```

Below we will group the ratings into bins to allow for some easy referencing.

In [96]:
```
"""
We want to create a categorical variable for the `Rating` to enable us track how the courses where rated.
"""
conditions1 = [
    (df['Rating'] == 0),
    (df['Rating'] > 0) & (df['Rating'] < 0.2),
    (df['Rating'] >= 0.2) & (df['Rating'] < 0.4),
    (df['Rating'] >= 0.4) & (df['Rating'] < 0.6),
    (df['Rating'] >= 0.6) & (df['Rating'] < 0.8),
    (df['Rating'] >= 0.8)
]
values1 = [0,1,2,3,4,5]

df['star_Rating'] = np.select(conditions1,values1)
df['star_Rating'].value_counts()
```

Out[96]:
```
5    1543
1     735
4     610
2     428
3     338
0      18
Name: star_Rating, dtype: int64
```

In [98]:
```
df['level'].value_counts()
```

Out[98]:
```
All Levels            1925
Beginner Level        1268
Intermediate Level     421
Expert Level            58
Name: level, dtype: int64
```

In [102…
```
"""
Change naming of the course levels into a more relatable nomenclature.
"""
levels = ({'All Levels':'General' , 'Beginner Level':'Beginner' , 'Intermediate Level':'Intermediate' ,
            'Expert Level':'Expert'})
df['level'] = df['level'].replace(levels)
df['level'].value_counts()
```

Out[102]:
```
General         1925
Beginner        1268
Intermediate     421
Expert            58
Name: level, dtype: int64
```

Let us extract the date only object from the datetime supplied; we wont be needing the time object for this analysis.We will then create the `year` column.

In [105…
```
#extract date only from datetime object
df['published'] = df['published_timestamp'].str.split('T').str[0]
```

In [113…
```
df.drop(columns=['published_timestamp'], inplace=True)
```

In [117…
```
#create column for the year the course was published
df['year'] = (df['published'].str.split('-').str[0]).astype(int)
df['year'].value_counts().sort_values(ascending=False)
```

Out[117]:
```
2016    1204
2015    1014
2017     713
2014     490
2013     201
2012      45
2011       5
Name: year, dtype: int64
```

Let us clean the text columns a bit.

In [119…
```
"""
Let us create a regex function to clean the text column off:
- urls
- punctuation
- special characters
"""

def clean(text):
    text = str(text).title()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text) #remove url
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text) #remove punctuations
    text = re.sub('\n', '', text)
    text = re.sub('[0-9]', '', text)
    text = re.sub('<.*?>+', '', text)
    return text

#apply the function created
df['course_title'] = df['course_title'].apply(clean)
```

In [120…
```
#clean off punctautions from subject column

df['subject']=df['subject'].str.split(': ').str[-1].str.lstrip()
df['subject'].value_counts()
```

Out[120]:
```
Web Development        1199
Business Finance       1191
Musical Instruments     680
Graphic Design          602
Name: subject, dtype: int64
```

Let us create the `Revenue` column by multiplying the `num_subscribers` by `price` of the course.

In [121…
```
df['revenue'] = df['num_subscribers'] * df['price']
```

In [123…
```
#separate the component dataframes
df_biz = df.query('subject == "Business Finance"')
df_gfx = df.query('subject == "Graphic Design"')
df_mus = df.query('subject == "Musical Instruments"')
df_dev = df.query('subject == "Web Development"')

print('There are {} {} courses'.format(df_biz.shape[0], df_biz.subject[0]))
print('There are {} {} courses'.format(df_gfx.shape[0], df_gfx.subject[0]))
print('There are {} {} courses'.format(df_mus.shape[0], df_mus.subject[0]))
print('There are {} {} courses'.format(df_dev.shape[0], df_dev.subject[0]))
```

```
There are 1191 Business Finance courses
There are 602 Graphic Design courses
There are 680 Musical Instruments courses
There are 1199 Web Development courses
```

Data Cleaning Steps:

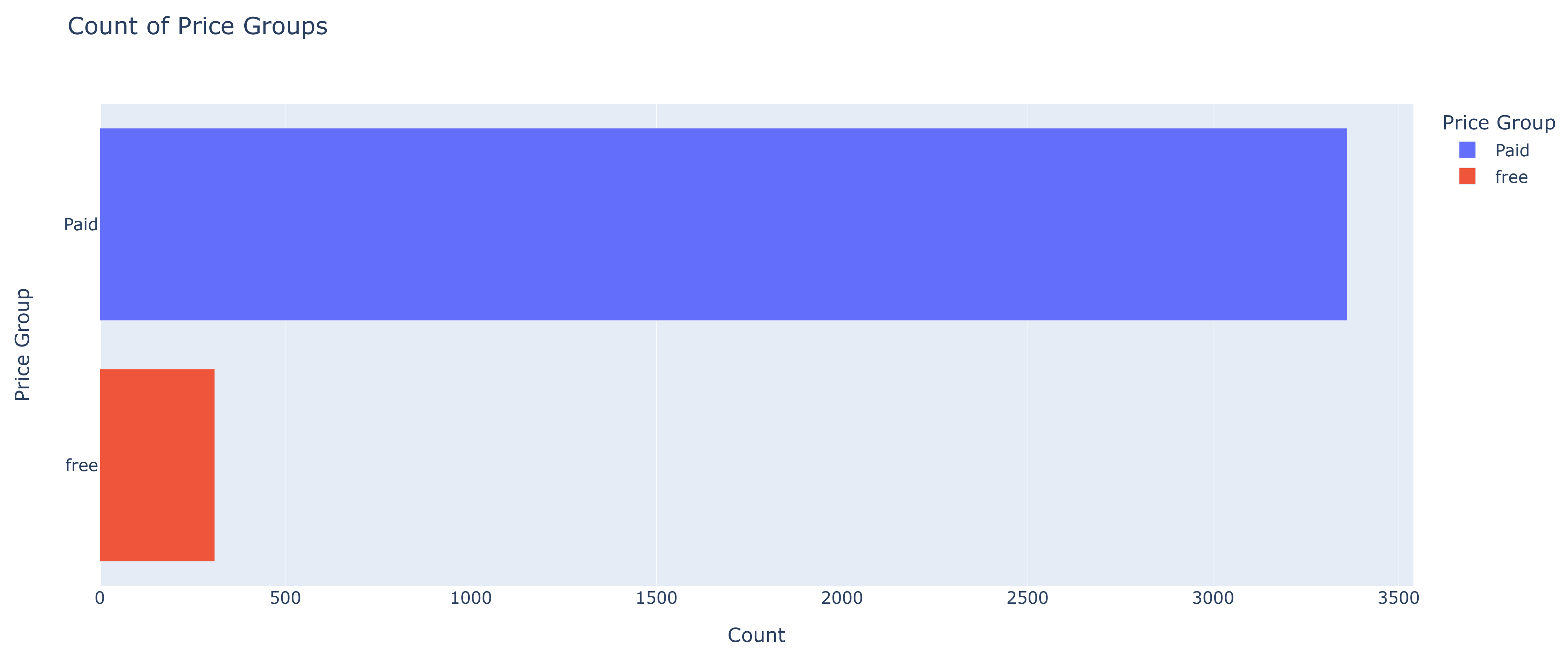
- Checked datatypes and missing values.
- Dropped null values from the dataframe.
- Created a column to categorize the courses into free and paid.
- Extracted and stored the date only data from the datetime column and created the `year` column.
- Created a function with regex to clean up the text columns.
- Dropped off columns that are not necesary for the analysis.
- Created the `revenue` column.

Exploratory Data Analysis

From the pie chart below, *Web Development* leads with **1203** courses while *Graphics Designs* at **602** courses is the least.

In [162…

```
result = df['price_group'].value_counts().reset_index()
result.columns = ['price_group', 'count']

fig = px.bar(result, x='count', y='price_group', orientation='h',
             labels={'count': 'Count', 'price_group': 'Price Group'},
             title='Count of Price Groups',
             color='price_group')

fig.show()
```

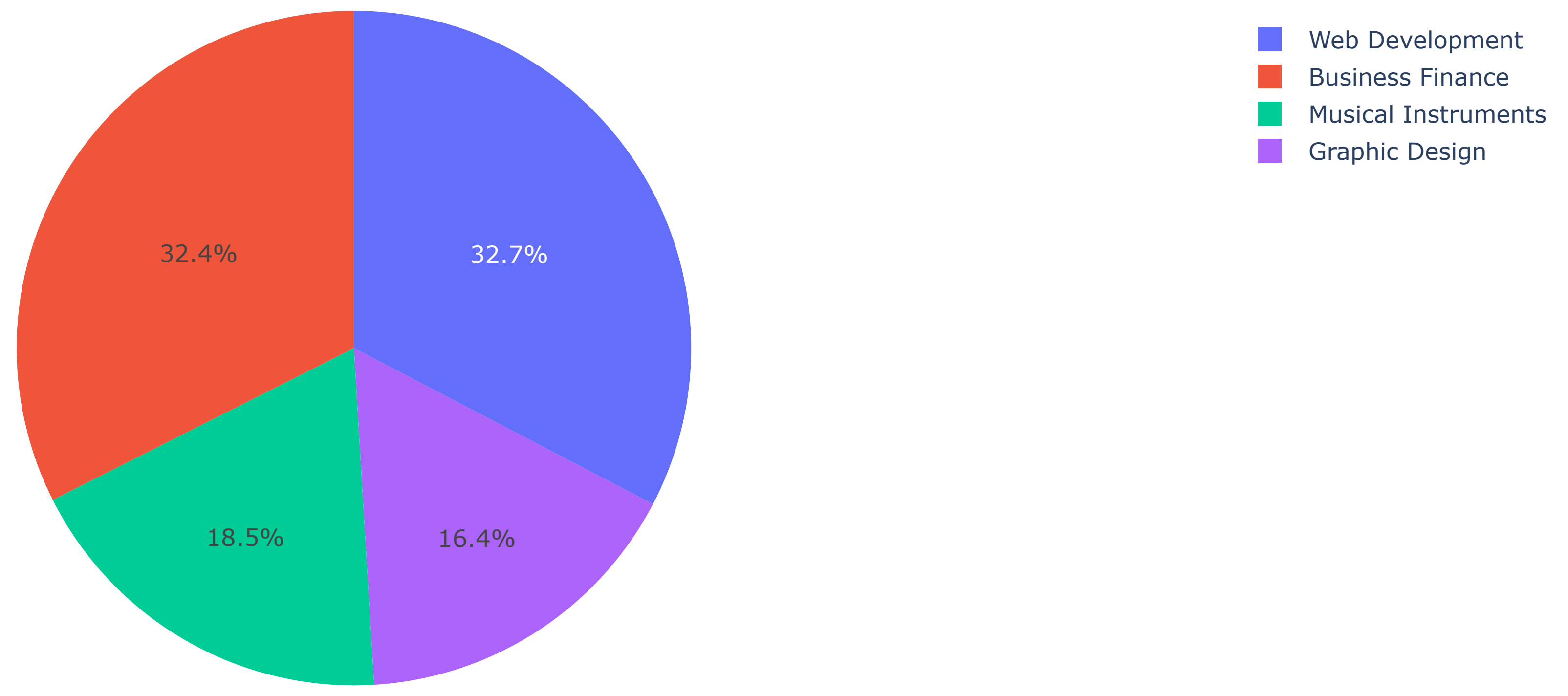


```
In [133…  df['subject'].value_counts()
```

```
Out[133]:  Web Development        1199
           Business Finance       1191
           Musical Instruments     680
           Graphic Design          602
           Name: subject, dtype: int64
```

```
In [164…  result = df['subject'].value_counts().reset_index()
          result.columns = ['subject', 'count']

          fig = px.pie(result, names='subject', values='count',
                       title='Distribution of Subjects',
                       labels={'count': 'Count'})

          fig.show()
```

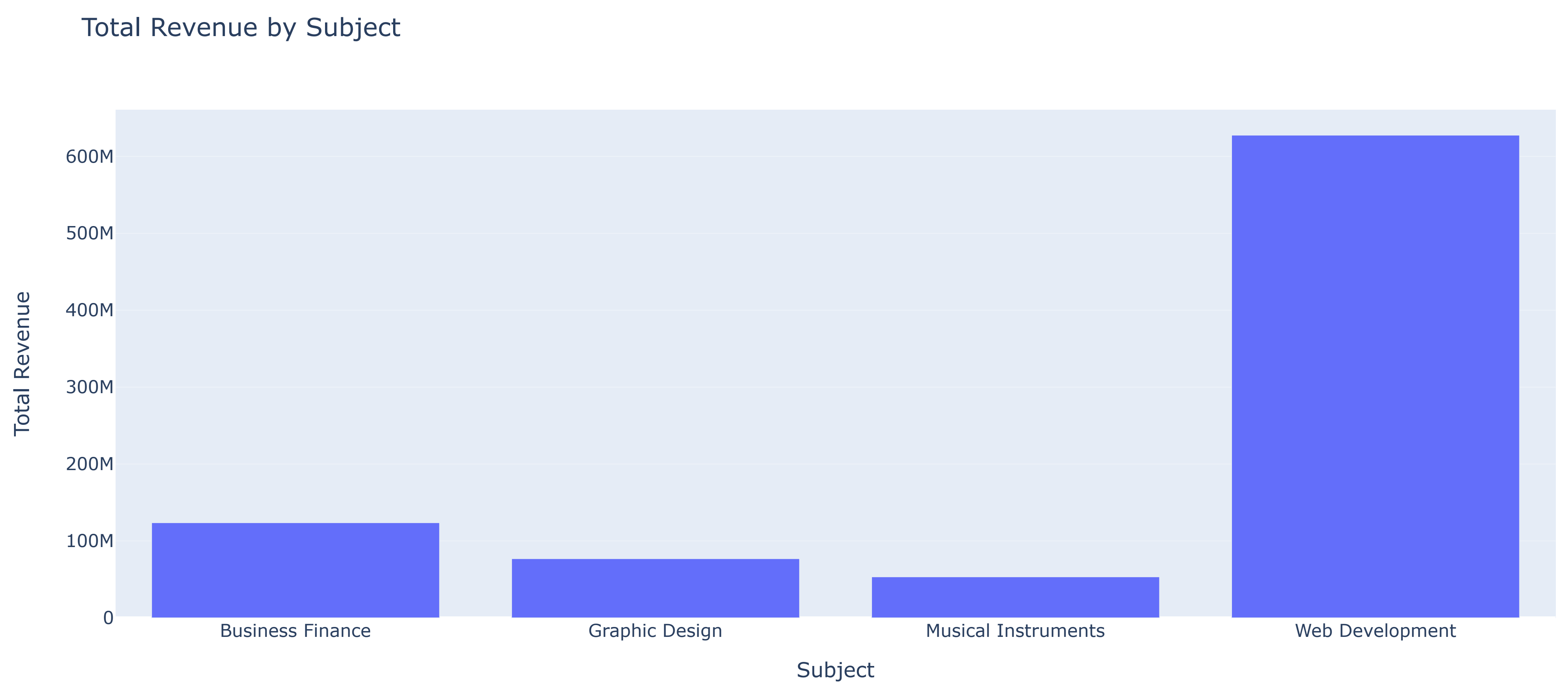
Distribution of Subjects



```
In [141…  df.groupby('subject')['revenue'].sum().sort_values(ascending=False)
```

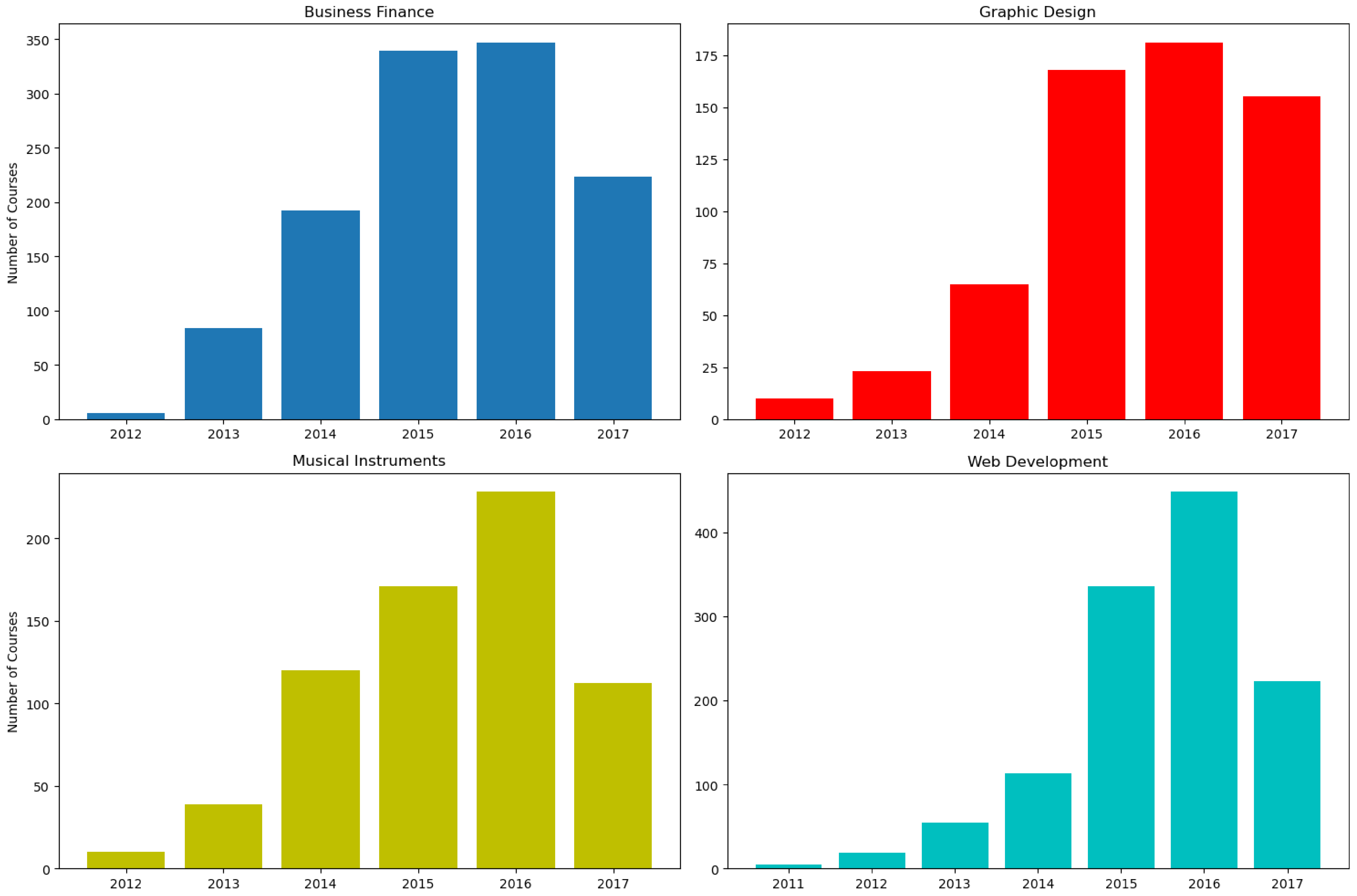
```
Out[141]: subject
          Web Development        627597400.0
          Business Finance       123735315.0
          Graphic Design          76983170.0
          Musical Instruments     53359055.0
          Name: revenue, dtype: float64
```

```
In [144…  result = df.groupby('subject')['revenue'].sum().reset_index()

          fig = px.bar(result, x='subject', y='revenue',
                       labels={'revenue': 'Total Revenue', 'subject': 'Subject'},
                       title='Total Revenue by Subject')

          fig.show()
```

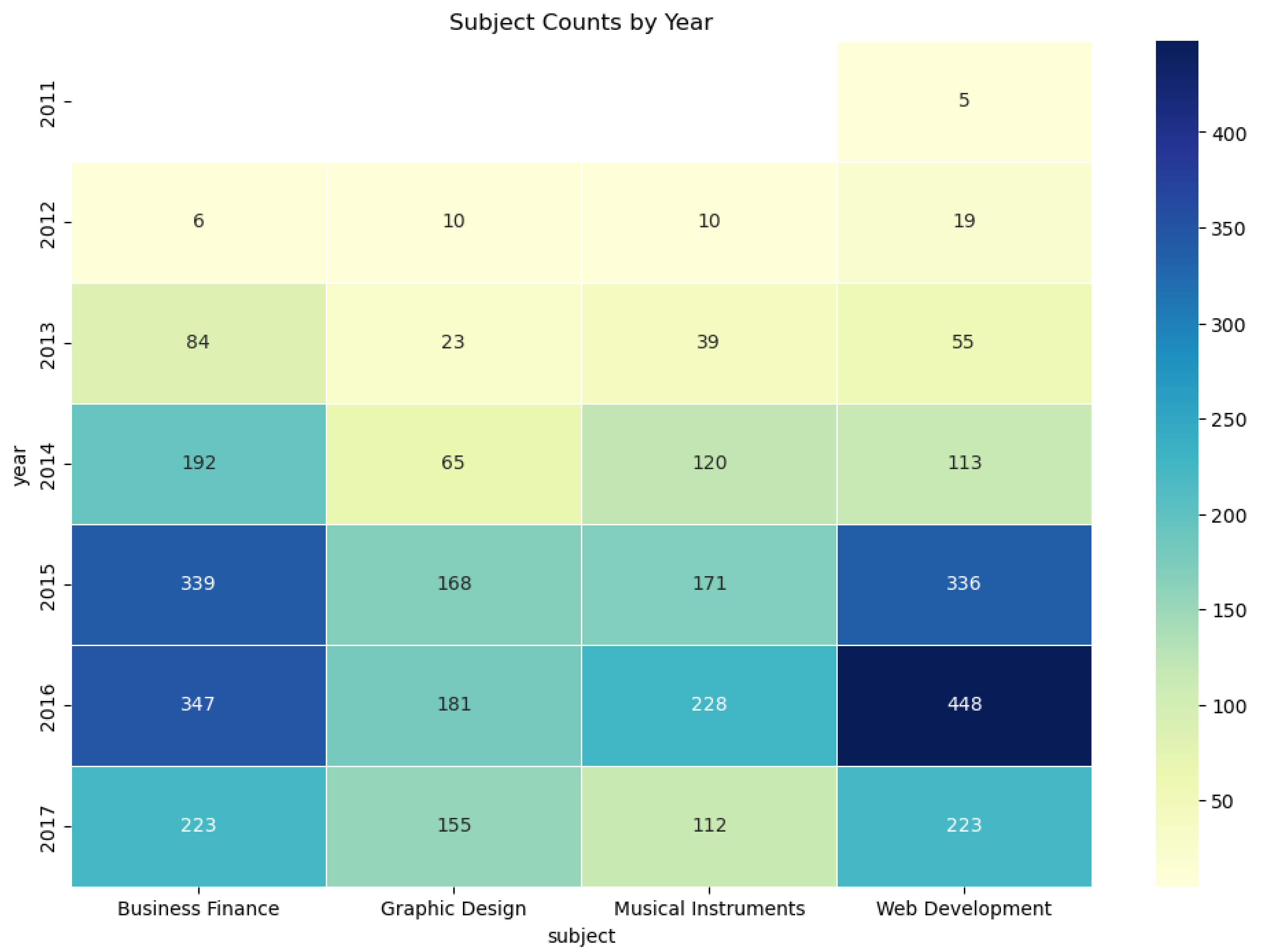


In [166…

```
fig, ((ax0, ax1), (ax2, ax3)) = plt.subplots(2,2 ,figsize=(15,10))

labels = [0,1,2,3,4,5]

ax0.bar(df_biz.year.value_counts().index,df_biz.year.value_counts().values)
ax0.set_title('Business Finance')
ax0.set_ylabel('Number of Courses')

ax1.bar(df_gfx.year.value_counts().index,df_gfx.year.value_counts().values,color='r')
ax1.set_title('Graphic Design')

ax2.bar(df_mus.year.value_counts().index,df_mus.year.value_counts().values,color='y')
ax2.set_title('Musical Instruments')
ax2.set_ylabel('Number of Courses')

ax3.bar(df_dev.year.value_counts().index,df_dev.year.value_counts().values, color='c')
ax3.set_title('Web Development')

fig.tight_layout()
plt.show()
```

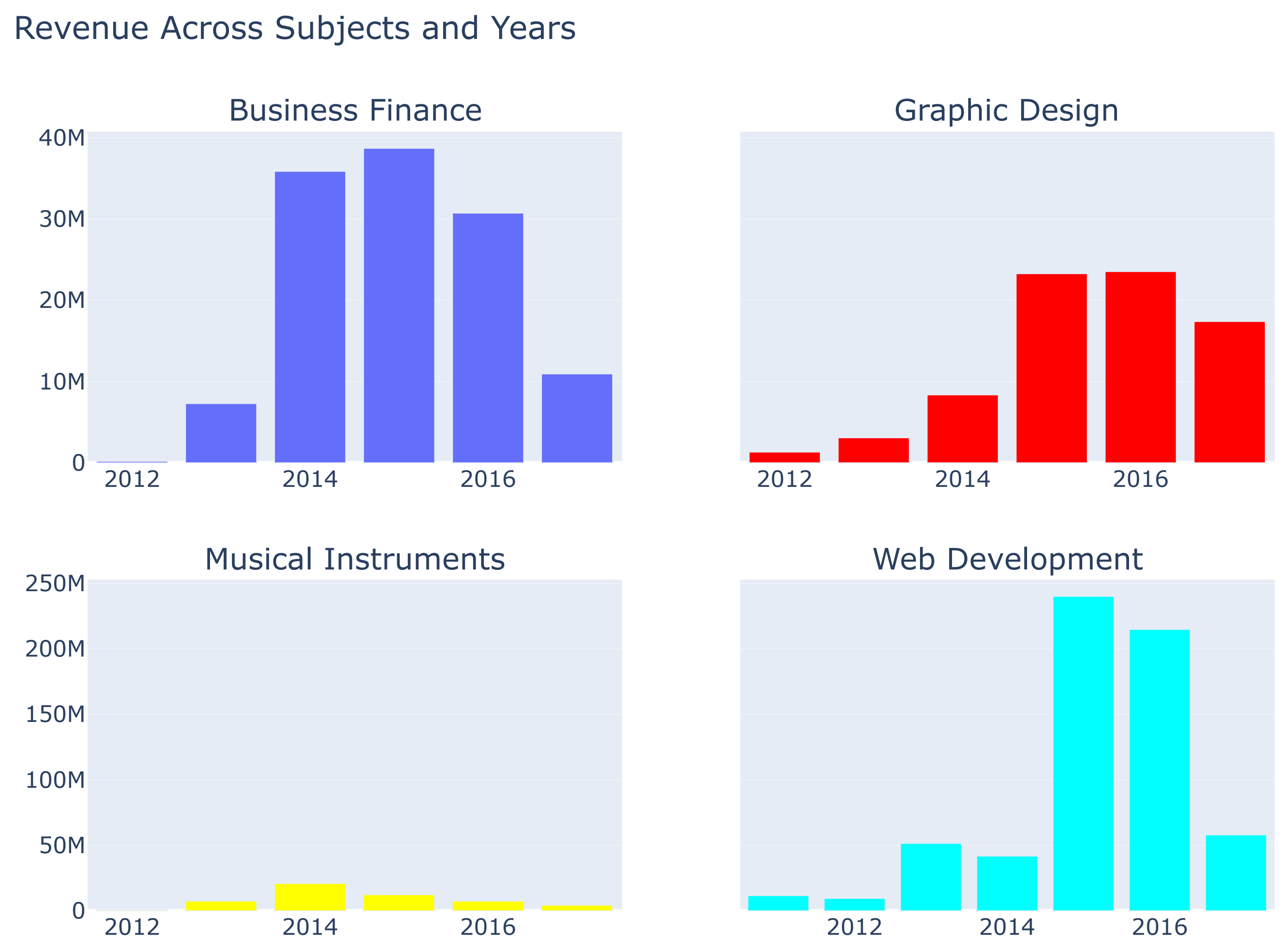


```
In [165…  result = df.groupby('year')['subject'].value_counts().reset_index(name='count')
          heatmap_data = result.pivot(index='year', columns='subject', values='count')

          plt.figure(figsize=(12, 8))
          sns.heatmap(heatmap_data, annot=True, cmap='YlGnBu', fmt='g', linewidths=.6)
          plt.title('Subject Counts by Year')
          plt.show()
```



In [168…

```python
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Create subplot grid
fig = make_subplots(rows=2, cols=2, subplot_titles=['Business Finance', 'Graphic Design', 'Musical Instruments', 'Web Development'],
                    shared_yaxes=True, horizontal_spacing=0.1, vertical_spacing=0.15)

# Plot for Business Finance
fig.add_trace(go.Bar(x=df_biz.groupby('year')['revenue'].sum().index, y=df_biz.groupby('year')['revenue'].sum(), name='Business Finance'), row=1, col=1)

# Plot for Graphic Design
fig.add_trace(go.Bar(x=df_gfx.groupby('year')['revenue'].sum().index, y=df_gfx.groupby('year')['revenue'].sum(), name='Graphic Design', marker_color='red'), row=1, col=2)

# Plot for Musical Instruments
fig.add_trace(go.Bar(x=df_mus.groupby('year')['revenue'].sum().index, y=df_mus.groupby('year')['revenue'].sum(), name='Musical Instruments', marker_color='yellow'), row=2, col=1)

# Plot for Web Development
fig.add_trace(go.Bar(x=df_dev.groupby('year')['revenue'].sum().index, y=df_dev.groupby('year')['revenue'].sum(), name='Web Development', marker_color='cyan'), row=2, col=2)

# Update layout
fig.update_layout(height=600, width=800, title_text='Revenue Across Subjects and Years', showlegend=False)

# Show the plot
fig.show()
```



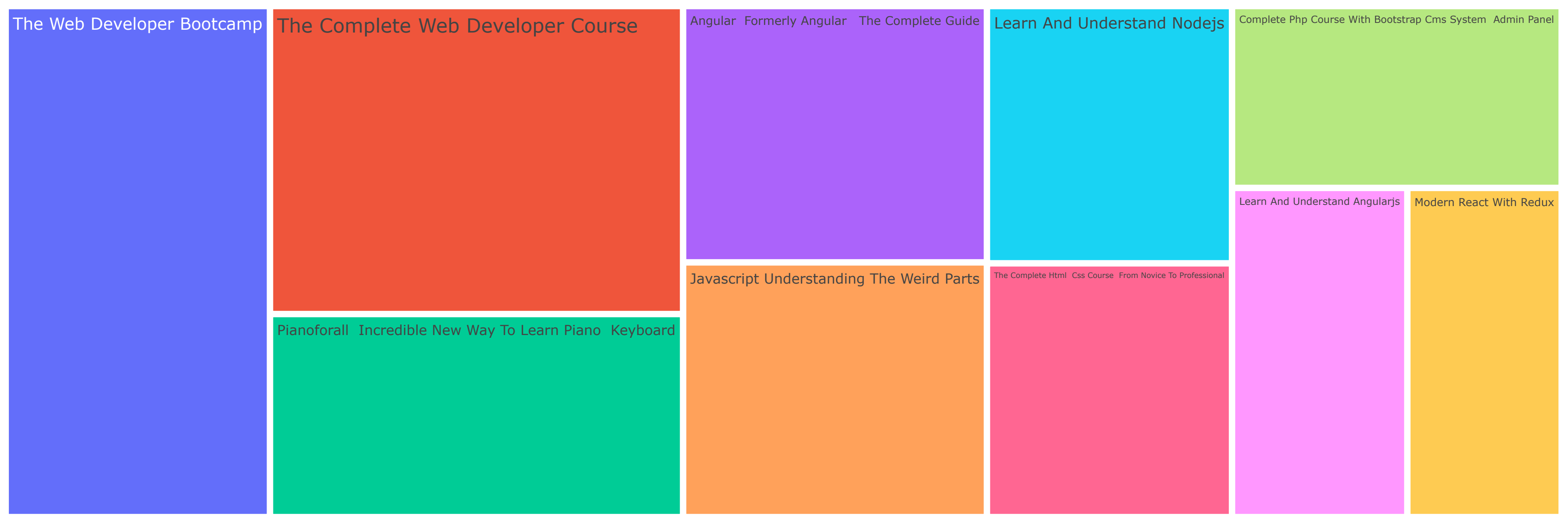
In [183…
```
df.groupby('subject')[['price', 'num_subscribers', 'Rating', 'star_Rating', 'revenue']].agg(['min', 'max', 'mean'])
```

Out[183]:

	price			num_subscribers			Rating			star_Rating			revenue		
	min	max	mean	min	max	mean	min	max	mean	min	max	mean	min	max	mean
subject															
Business Finance	0.0	200.0	68.694374	0	65576	1569.026868	0.00	1.00	0.690353	0	5	3.878254	0.0	4773795.0	103891.952141
Graphic Design	0.0	200.0	57.890365	0	53851	1766.026578	0.01	0.99	0.730382	1	5	3.978405	0.0	7257600.0	127879.019934
Musical Instruments	0.0	200.0	49.558824	0	101154	1245.130882	0.00	1.00	0.308912	0	5	2.039706	0.0	15099800.0	78469.198529
Web Development	0.0	200.0	77.035029	19	268923	6619.922435	0.00	1.00	0.642127	0	5	3.635530	0.0	24316800.0	523434.028357

In [255…
```
# Top 10 courses by revenue
top_courses = df[['course_title', 'revenue']].sort_values('revenue', ascending=False).head(10)

# Create a treemap
fig = px.treemap(top_courses, path=['course_title'], values='revenue', title='Top 10 Courses by Revenue')

# Show the plot
fig.show()
```

Top 10 Courses by Revenue



Comments

- The dataset is limited to only four subject areas while leaving out a ton of interesting areas including Data Science, Cyber Security, Cloud Computing, Digital Marketing.
- The dataset is not updated to the current year, 2022. It would have been nice to see how digital learning grew in the Covid-19 era.
- There are no demographic data on the subscribers and the duration of their learning of the chosen subject.
- The rating of the courses didnt provide details of the comments made by the subscribers.
- It was *not* expressely stated what the unit of the course duration was - whether hours or minutes.
- It was not explained what the scale of the `Rating` was; does a rating of 0.0 mean there was no rating at all or the course received the least score.
- There is no data on the content creators; it would have been great to know which tutors make the most subscribed and best rated contents.

Conclusions

- `Web Development` and `Graphics Designs` have the highest and lowest number of published courses respectively.
- Subscribers are more interested in `Web Development` courses and least in `Musical Instruments` courses.
- `Graphics Design` courses followed by `Business Finance` received the best ratings.
- `Web Development` courses are more expensive, attractive more subscribers; hence generate the highest revenue.
- The % of the courses that are free are more in expert level.
- For courses with high rating and subscriptions, they are courses on Accounting, Forex, Stock in Business Finance; Photoshop, Adobe Illustrator in Graphics Design; Piano and Guitar in Musical Instruments and HTML, CSS, building a Website in Web Development.
- The courses received high ratings across board.
- The year 2016 has highest number of courses published across the different subjects.
- For revenue generation, 2014 was the big year for `Musical Instruments`, 2015 for `Business Finance` and `Web Development` while `Graphics Design` made the most revenue in 2016.