In [4]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [5]:
```
df = pd.read_csv('/kaggle/input/udemy-courses/Course_info.csv', encoding = 'utf-8')
df.shape
```

Out[5]:
```
(209734, 20)
```

In [6]:
```
df.head()
```

Out[6]:

	id	title	is_paid	price	headline	num_subscribers	avg_rating	num_reviews
0	4715.0	Online Vegan Vegetarian Cooking School	True	24.99	Learn to cook delicious vegan recipes. Filmed ...	2231.0	3.75	134.0
1	1769.0	The Lean Startup Talk at Stanford E-Corner	False	0.00	Debunking Myths of Entrepreneurship A startup ...	26474.0	4.50	709.0
2	5664.0	How To Become a Vegan, Vegetarian, or Flexitarian	True	19.99	Get the tools you need for a lifestyle change ...	1713.0	4.40	41.0
3	7723.0	How to Train a Puppy	True	199.99	Train your puppy the right way with Dr. Ian Du...	4988.0	4.80	395.0
4	8157.0	Web Design from the Ground Up	True	159.99	Learn web design online: Everything you need t...	1266.0	4.75	38.0

In [7]:
```
df.head(5)
```

Out[7]:

	id	title	is_paid	price	headline	num_subscribers	avg_rating	num_reviews
0	4715.0	Online Vegan Vegetarian Cooking School	True	24.99	Learn to cook delicious vegan recipes. Filmed ...	2231.0	3.75	134.0
1	1769.0	The Lean Startup Talk at Stanford E-Corner	False	0.00	Debunking Myths of Entrepreneurship A startup ...	26474.0	4.50	709.0
2	5664.0	How To Become a Vegan, Vegetarian, or Flexitarian	True	19.99	Get the tools you need for a lifestyle change ...	1713.0	4.40	41.0
3	7723.0	How to Train a Puppy	True	199.99	Train your puppy the right way with Dr. Ian Du...	4988.0	4.80	395.0
4	8157.0	Web Design from the Ground Up	True	159.99	Learn web design online: Everything you need t...	1266.0	4.75	38.0

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209734 entries, 0 to 209733
Data columns (total 20 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   id                  209734 non-null  float64
 1   title               209734 non-null  object 
 2   is_paid             209734 non-null  bool   
 3   price               209734 non-null  float64
 4   headline            209707 non-null  object 
 5   num_subscribers     209734 non-null  float64
 6   avg_rating          209734 non-null  float64
 7   num_reviews         209734 non-null  float64
 8   num_comments        209734 non-null  float64
 9   num_lectures        209734 non-null  float64
 10  content_length_min  209734 non-null  float64
 11  published_time      209734 non-null  object 
 12  last_update_date    209597 non-null  object 
 13  category            209734 non-null  object 
 14  subcategory         209734 non-null  object 
 15  topic               208776 non-null  object 
 16  language            209734 non-null  object 
 17  course_url          209734 non-null  object 
 18  instructor_name     209729 non-null  object 
 19  instructor_url      209307 non-null  object 
dtypes: bool(1), float64(8), object(11)
memory usage: 30.6+ MB
```

In [9]:
```
pd.isnull(df)
```

Out[9]:

	id	title	is_paid	price	headline	num_subscribers	avg_rating	num_reviews	num_con
0	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	
209729	False	False	False	False	False	False	False	False	
209730	False	False	False	False	False	False	False	False	
209731	False	False	False	False	False	False	False	False	
209732	False	False	False	False	False	False	False	False	
209733	False	False	False	False	False	False	False	False	

209734 rows × 20 columns

In [10]:
```
pd.isnull(df).sum()
```

Out[10]:
```
id                     0
title                  0
is_paid                0
price                  0
headline              27
num_subscribers        0
avg_rating             0
num_reviews            0
num_comments           0
num_lectures           0
content_length_min     0
published_time         0
last_update_date     137
category               0
subcategory            0
topic                958
language               0
course_url             0
instructor_name        5
instructor_url       427
dtype: int64
```

In [11]:
```
df.shape
```

Out[11]:
```
(209734, 20)
```

In [12]:
```
df.dropna(inplace = True)
```

In [13]:
```
df.shape
```

Out[13]:
```
(208190, 20)
```

In [14]:
```
pd.isnull(df).sum()
```

Out[14]:
```
id                   0
title                0
is_paid              0
price                0
headline             0
num_subscribers      0
avg_rating           0
num_reviews          0
num_comments         0
num_lectures         0
content_length_min   0
published_time       0
last_update_date     0
category             0
subcategory          0
topic                0
language             0
course_url           0
instructor_name      0
instructor_url       0
dtype: int64
```

In [15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 208190 entries, 0 to 209733
Data columns (total 20 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   id                  208190 non-null  float64
 1   title               208190 non-null  object 
 2   is_paid             208190 non-null  bool   
 3   price               208190 non-null  float64
 4   headline            208190 non-null  object 
 5   num_subscribers     208190 non-null  float64
 6   avg_rating          208190 non-null  float64
 7   num_reviews         208190 non-null  float64
 8   num_comments        208190 non-null  float64
 9   num_lectures        208190 non-null  float64
 10  content_length_min  208190 non-null  float64
 11  published_time      208190 non-null  object 
 12  last_update_date    208190 non-null  object 
 13  category            208190 non-null  object 
 14  subcategory         208190 non-null  object 
 15  topic               208190 non-null  object 
 16  language            208190 non-null  object 
 17  course_url          208190 non-null  object 
 18  instructor_name     208190 non-null  object 
 19  instructor_url      208190 non-null  object 
dtypes: bool(1), float64(8), object(11)
memory usage: 32.0+ MB
```

In [16]:
```
df['avg_rating'] = df['avg_rating'].astype('int')
```

In [17]:
```
df['avg_rating'].dtypes
```

Out[17]:
```
dtype('int64')
```

In [18]:
```
df['num_reviews'] = df['num_reviews'].astype('int')
```

In [19]:
```
df['price'] = df['price'].astype('int')
```

In [20]:
```
df[' num_comments'] = df['num_comments'].astype('int')
```

In [21]:
```
df['content_length_min'] = df['content_length_min'].astype('int')
```

In [22]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 208190 entries, 0 to 209733
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   id                  208190 non-null  float64
 1   title               208190 non-null  object 
 2   is_paid             208190 non-null  bool   
 3   price               208190 non-null  int64  
 4   headline            208190 non-null  object 
 5   num_subscribers     208190 non-null  float64
 6   avg_rating          208190 non-null  int64  
 7   num_reviews         208190 non-null  int64  
 8   num_comments        208190 non-null  float64
 9   num_lectures        208190 non-null  float64
 10  content_length_min  208190 non-null  int64  
 11  published_time      208190 non-null  object 
 12  last_update_date    208190 non-null  object 
 13  category            208190 non-null  object 
 14  subcategory         208190 non-null  object 
 15  topic               208190 non-null  object 
 16  language            208190 non-null  object 
 17  course_url          208190 non-null  object 
 18  instructor_name     208190 non-null  object 
 19  instructor_url      208190 non-null  object 
 20   num_comments       208190 non-null  int64  
dtypes: bool(1), float64(4), int64(5), object(11)
memory usage: 33.6+ MB
```

In [23]:
```
df['num_subscribers'] = df['num_subscribers'].astype('int')
```

In [24]:
```
df['num_lectures'] = df['num_lectures'].astype('int')
```

In [25]:
```
df['num_comments']=df['num_comments'].astype('int')
```

In [26]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 208190 entries, 0 to 209733
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   id                  208190 non-null  float64
 1   title               208190 non-null  object 
 2   is_paid             208190 non-null  bool   
 3   price               208190 non-null  int64  
 4   headline            208190 non-null  object 
 5   num_subscribers     208190 non-null  int64  
 6   avg_rating          208190 non-null  int64  
 7   num_reviews         208190 non-null  int64  
 8   num_comments        208190 non-null  int64  
 9   num_lectures        208190 non-null  int64  
 10  content_length_min  208190 non-null  int64  
 11  published_time      208190 non-null  object 
 12  last_update_date    208190 non-null  object 
 13  category            208190 non-null  object 
 14  subcategory         208190 non-null  object 
 15  topic               208190 non-null  object 
 16  language            208190 non-null  object 
 17  course_url          208190 non-null  object 
 18  instructor_name     208190 non-null  object 
 19  instructor_url      208190 non-null  object 
 20   num_comments       208190 non-null  int64  
dtypes: bool(1), float64(1), int64(8), object(11)
memory usage: 33.6+ MB
```

In [27]:
```
df['num_comments']=df['num_comments'].astype('int')
```

In [28]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 208190 entries, 0 to 209733
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   id                  208190 non-null  float64
 1   title               208190 non-null  object 
 2   is_paid             208190 non-null  bool   
 3   price               208190 non-null  int64  
 4   headline            208190 non-null  object 
 5   num_subscribers     208190 non-null  int64  
 6   avg_rating          208190 non-null  int64  
 7   num_reviews         208190 non-null  int64  
 8   num_comments        208190 non-null  int64  
 9   num_lectures        208190 non-null  int64  
 10  content_length_min  208190 non-null  int64  
 11  published_time      208190 non-null  object 
 12  last_update_date    208190 non-null  object 
 13  category            208190 non-null  object 
 14  subcategory         208190 non-null  object 
 15  topic               208190 non-null  object 
 16  language            208190 non-null  object 
 17  course_url          208190 non-null  object 
 18  instructor_name     208190 non-null  object 
 19  instructor_url      208190 non-null  object 
 20   num_comments       208190 non-null  int64  
dtypes: bool(1), float64(1), int64(8), object(11)
memory usage: 33.6+ MB
```

In [29]:
```
df.columns
```

Out[29]:
```
Index(['id', 'title', 'is_paid', 'price', 'headline', 'num_subscribers',
       'avg_rating', 'num_reviews', 'num_comments', 'num_lectures',
       'content_length_min', 'published_time', 'last_update_date', 'category',
       'subcategory', 'topic', 'language', 'course_url', 'instructor_name',
       'instructor_url', ' num_comments'],
      dtype='object')
```

In [30]:
```
df.describe()
```

Out[30]:

	id	price	num_subscribers	avg_rating	num_reviews	num_commen
count	2.081900e+05	208190.000000	2.081900e+05	208190.000000	208190.000000	208190.0000
mean	3.013411e+06	80.966689	3.104826e+03	3.355829	245.377597	45.0677
std	1.340752e+06	117.339729	1.561931e+04	1.402331	2466.323175	356.9972
min	2.762000e+03	0.000000	0.000000e+00	0.000000	0.000000	0.0000
25%	1.949413e+06	19.000000	2.600000e+01	3.000000	3.000000	1.0000
50%	3.287052e+06	34.000000	2.070000e+02	4.000000	17.000000	5.0000
75%	4.184922e+06	99.000000	1.441000e+03	4.000000	75.000000	18.0000
max	4.914146e+06	999.000000	1.752364e+06	5.000000	436457.000000	39040.0000

In [31]:
```
df[['price','avg_rating','num_reviews','num_comments']].describe()
```

Out[31]:

	price	avg_rating	num_reviews	num_comments
count	208190.000000	208190.000000	208190.000000	208190.000000
mean	80.966689	3.355829	245.377597	45.067712
std	117.339729	1.402331	2466.323175	356.997245
min	0.000000	0.000000	0.000000	0.000000
25%	19.000000	3.000000	3.000000	1.000000
50%	34.000000	4.000000	17.000000	5.000000
75%	99.000000	4.000000	75.000000	18.000000
max	999.000000	5.000000	436457.000000	39040.000000

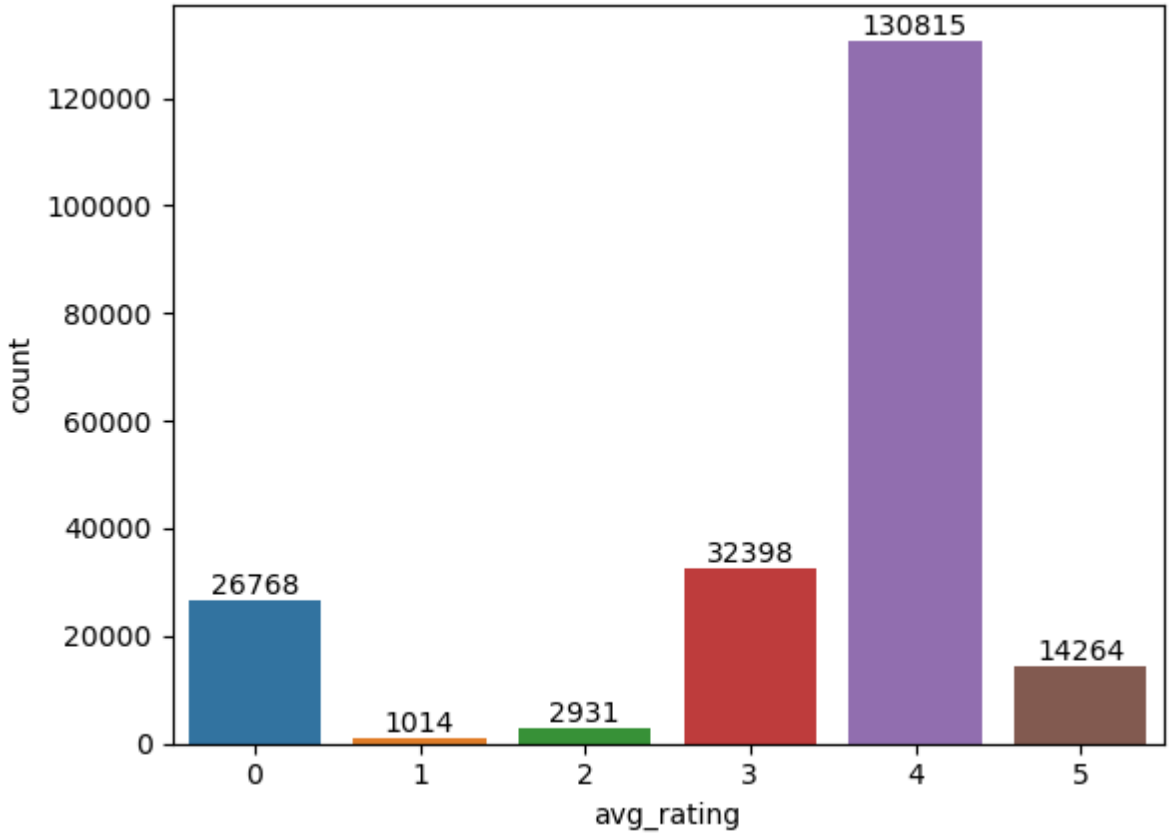
EXPLORATORY DATA ANALYSIS :

In [32]:
```
df.columns
```

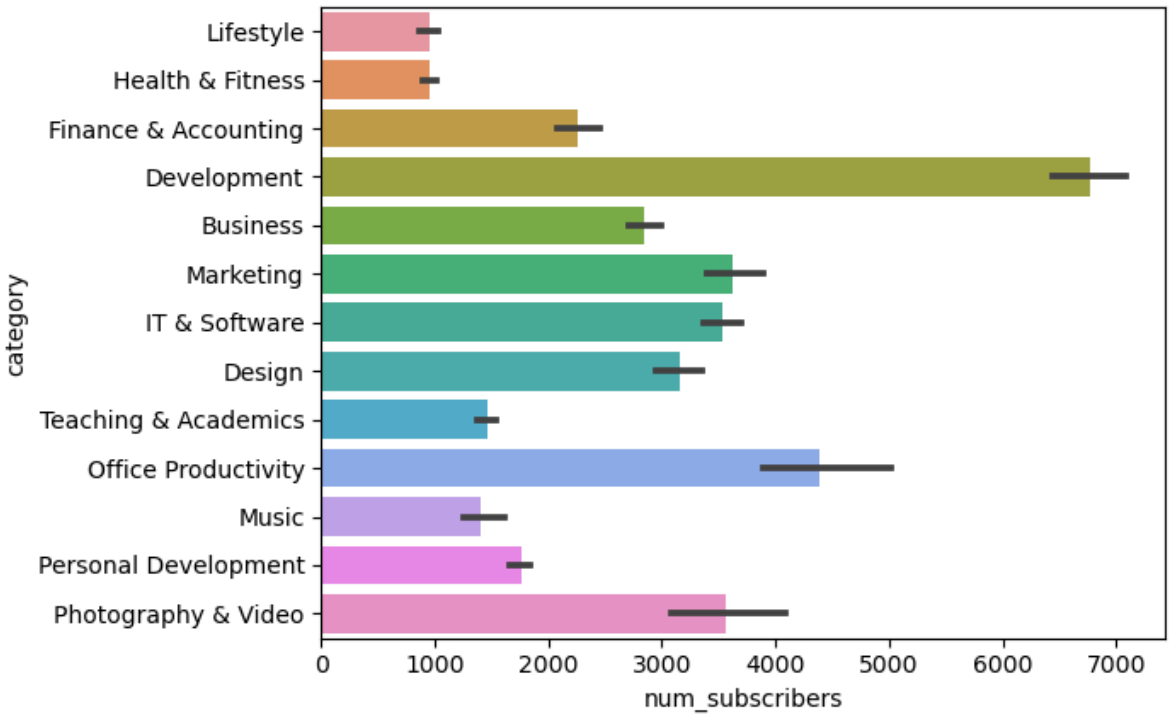
Out[32]:
```
Index(['id', 'title', 'is_paid', 'price', 'headline', 'num_subscribers',
       'avg_rating', 'num_reviews', 'num_comments', 'num_lectures',
       'content_length_min', 'published_time', 'last_update_date', 'category',
       'subcategory', 'topic', 'language', 'course_url', 'instructor_name',
       'instructor_url', ' num_comments'],
      dtype='object')
```

In [33]:
```
ax = sns.countplot(x ='avg_rating',data =df)

for bars in ax.containers:
```

```
    ax.bar_label(bars)
```



In [34]:
```
sns.barplot(data=df ,x = 'num_subscribers' , y='category')
sns.set(rc={'figure.figsize':(15,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```



In [35]:
```
df.columns
```

Out[35]:
```
Index(['id', 'title', 'is_paid', 'price', 'headline', 'num_subscribers',
       'avg_rating', 'num_reviews', 'num_comments', 'num_lectures',
       'content_length_min', 'published_time', 'last_update_date', 'category',
       'subcategory', 'topic', 'language', 'course_url', 'instructor_name',
       'instructor_url', ' num_comments'],
      dtype='object')
```

In [ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming I have the sales_state DataFrame with 'num_reviews' and 'category' colum

# Assuming the 'sales_state' DataFrame is already sorted and contains the top 20 da
top_20_data = sales_state.head(20)

sns.set(rc={'figure.figsize': (6, 4)})
sns.barplot(data=top_20_data, x='num_reviews', y='category')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[36], line 7
      2 import matplotlib.pyplot as plt
      4 # Assuming you have the sales_state DataFrame with 'num_reviews' and 'cate
gory' columns
      5
      6 # Assuming the 'sales_state' DataFrame is already sorted and contains the
top 20 data points
----> 7 top_20_data = sales_state.head(20)
      9 sns.set(rc={'figure.figsize': (6, 4)})
     10 sns.barplot(data=top_20_data, x='num_reviews', y='category')

NameError: name 'sales_state' is not defined
```

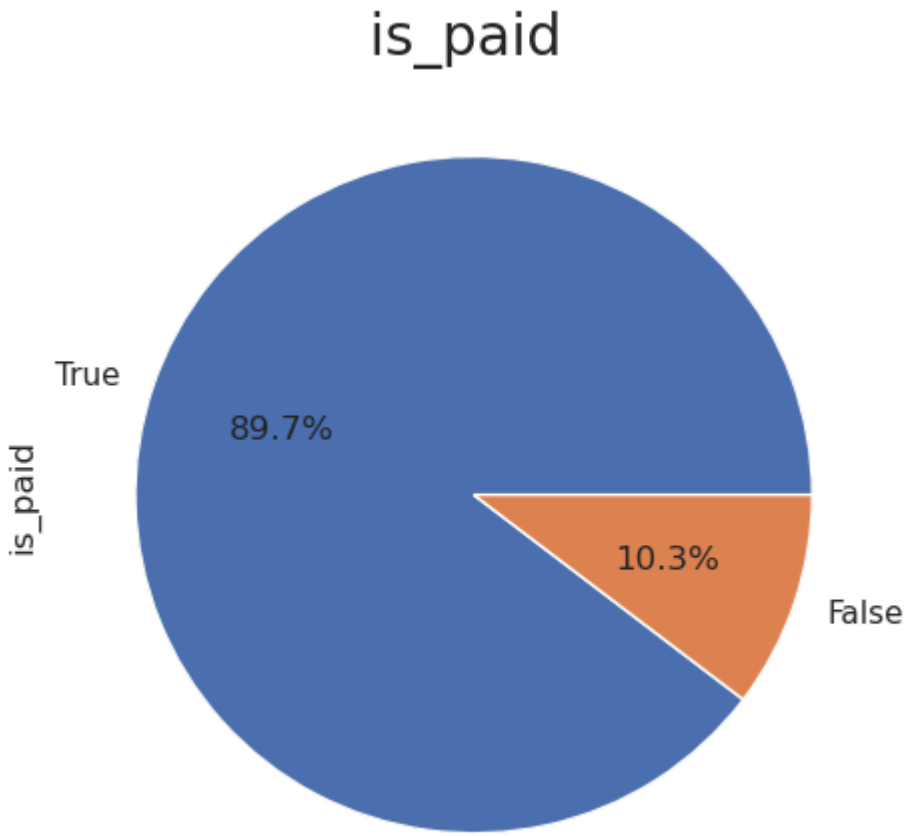
In [ ]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Loading the dataset from the CSV file
csv_path = 'path/to/your/dataset.csv'
df = pd.read_csv(csv_path).sort_values(by='price')

# Creating the bar plot
fig, ax1 = plt.subplots(figsize=(8, 4))
ax1.bar(df.index, df['num_subscribers'], color='blue', label='Subscribers')
ax1.set_xlabel('Courses')
ax1.set_ylabel('Number of Subscribers')
ax1.set_title('Courses vs. Number of Subscribers')
ax1.set_xticks(df.index)
ax1.set_xticklabels(df['title'], rotation=45, ha='right')
ax1.legend()

# Adding a secondary y-axis for price
ax2 = ax1.twinx()
ax2.plot(df.index, df['price'], color='red', marker='o', label='Price')
ax2.set_ylabel('Price')
ax2.legend()

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

In [1]:
```
df.columns()
```

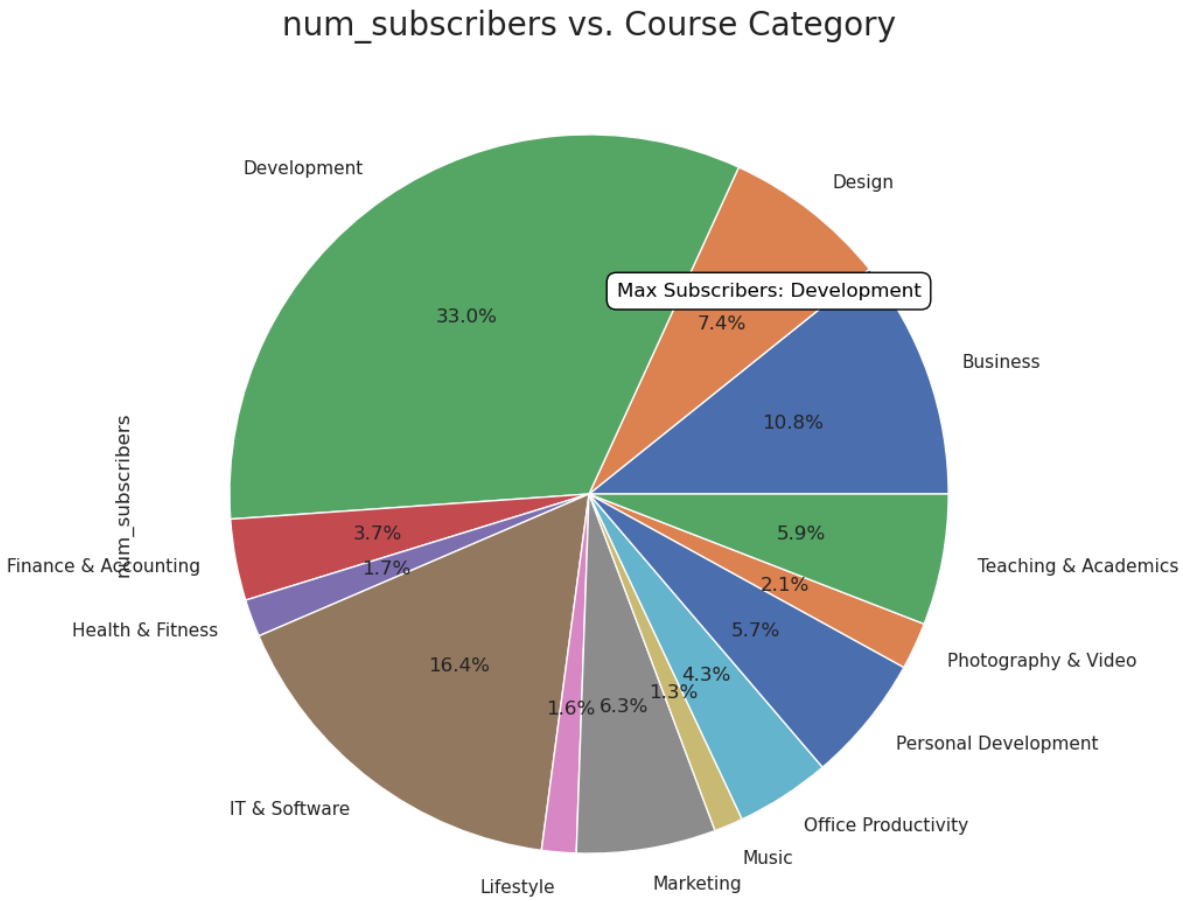
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 1
----> 1 df.columns()

NameError: name 'df' is not defined
```

In [40]:
```
plt.figure(figsize=(30,10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                    wspace=0.5, hspace=0.2)
plt.subplot(141)
plt.title('is_paid ',fontsize = 20)
df['is_paid'].value_counts().plot.pie(autopct="%1.1f%%")
```

Out[40]:
```
<Axes: title={'center': 'is_paid '}, ylabel='is_paid'>
```

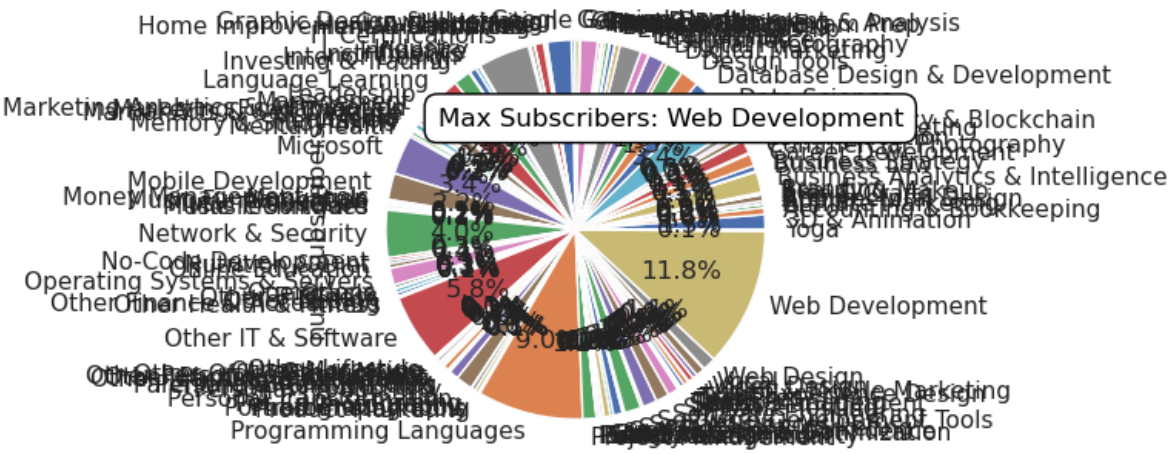


In [ ]:
```
category_subscribers = df.groupby('category')['num_subscribers'].sum()

# Find the category with the maximum subscribers
max_subscribers_category = category_subscribers.idxmax()


plt.figure(figsize=(10, 10))
plt.title('num_subscribers vs. Course Category', fontsize=20)
category_subscribers.plot.pie(autopct="%1.1f%%")

# Annotating the category with the maximum subscribers
plt.annotate(f"Max Subscribers: {max_subscribers_category}",
             xy=(0.5, 0.5), xytext=(0.5, 0.55),
             fontsize=12, ha="center", color="black",
             bbox=dict(facecolor='white', edgecolor='black', boxstyle='round,pad=0.

plt.show()
```



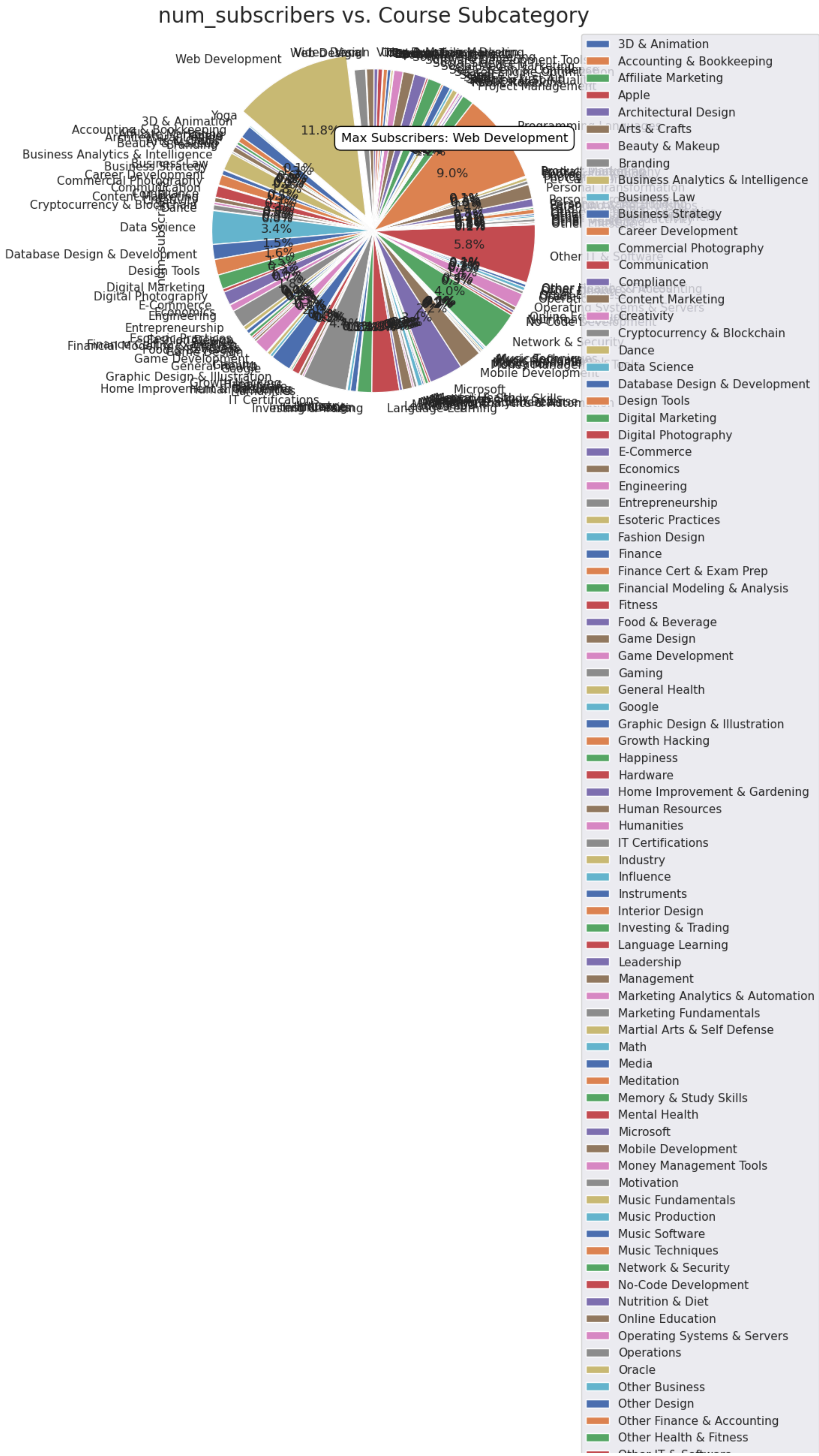
In [ ]:

```
category_subscribers = df.groupby('subcategory')['num_subscribers'].sum()

# Finding the category with the maximum subscribers
max_subscribers_category = category_subscribers.idxmax()


plt.figure(figsize=(7, 4))
plt.title('num_subscribers vs. Course Category', fontsize=20)
category_subscribers.plot.pie(autopct="%1.1f%%")

# Annotating the category with the maximum subscribers
plt.annotate(f"Max Subscribers: {max_subscribers_category}",
             xy=(0.5, 0.5), xytext=(0.5, 0.55),
             fontsize=12, ha="center", color="black",
             bbox=dict(facecolor='white', edgecolor='black', boxstyle='round,pad=0.

plt.show()
```



In [ ]:

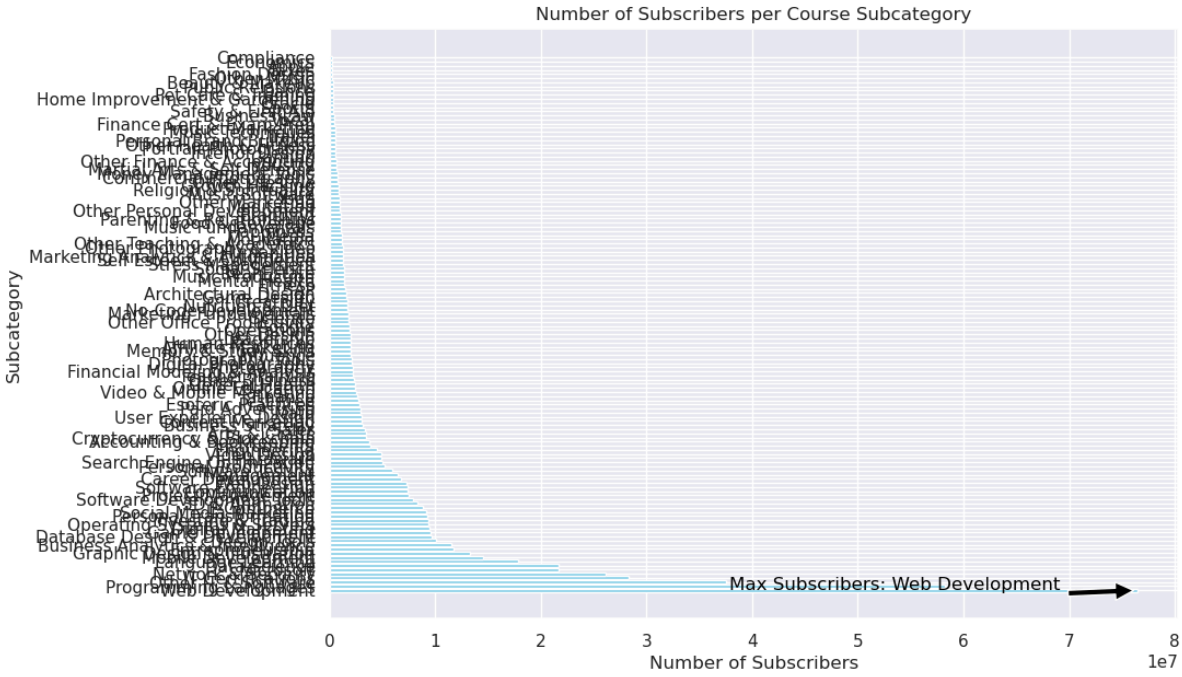
```
category_subscribers = df.groupby('subcategory')['num_subscribers'].sum()

# Finding the category with the maximum subscribers
max_subscribers_category = category_subscribers.idxmax()

# Creating a pie chart for the distribution of subscribers across course subcategor
plt.figure(figsize=(10, 7))

# Exploding the slice corresponding to the category with the maximum subscribers
explode = [0.1 if subcategory == max_subscribers_category else 0 for subcategory in

plt.title('num_subscribers vs. Course Subcategory', fontsize=20)
category_subscribers.plot.pie(autopct="%1.1f%%", explode=explode, startangle=140)

# Moving the legend outside the pie chart for clarity
plt.legend(labels=category_subscribers.index, loc="upper left", bbox_to_anchor=(1,

# Annotating the category with the maximum subscribers
plt.annotate(f"Max Subscribers: {max_subscribers_category}",
             xy=(0.5, 0.5), xytext=(0.5, 0.55),
             fontsize=12, ha="center", color="black",
             bbox=dict(facecolor='white', edgecolor='black', boxstyle='round,pad=0.

plt.show()
```



num_subscribers vs. Course Subcategory
Max Subscribers: Web Development
Web Development
11.8%
9.0%
5.8%
4.0%
3.4%
Data Science
Database Design & Development
Design Tools
Network & Security
Mobile Development
Microsoft
Language Learning
IT Certifications
Graphic Design & Illustration
3D & Animation
Accounting & Bookkeeping
Affiliate Marketing
Apple
Architectural Design
Arts & Crafts
Beauty & Makeup
Branding
Business Analytics & Intelligence
Business Law
Business Strategy
Career Development
Commercial Photography
Communication
Compliance
Content Marketing
Creativity
Cryptocurrency & Blockchain
Dance
Data Science
Database Design & Development
Design Tools
Digital Marketing
Digital Photography
E-Commerce
Economics
Engineering
Entrepreneurship
Esoteric Practices
Fashion Design
Finance
Finance Cert & Exam Prep
Financial Modeling & Analysis
Fitness
Food & Beverage
Game Design
Game Development
Gaming
General Health
Google
Graphic Design & Illustration
Growth Hacking
Happiness
Hardware
Home Improvement & Gardening
Human Resources
Humanities
IT Certifications
Industry
Influence
Instruments
Interior Design
Investing & Trading
Language Learning
Leadership
Management
Marketing Analytics & Automation
Marketing Fundamentals
Martial Arts & Self Defense
Math
Media
Meditation
Memory & Study Skills
Mental Health
Microsoft
Mobile Development
Money Management Tools
Motivation
Music Fundamentals
Music Production
Music Software
Music Techniques
Network & Security
No-Code Development
Nutrition & Diet
Online Education
Operating Systems & Servers
Operations
Oracle
Other Business
Other Design
Other Finance & Accounting
Other Health & Fitness

In [ ]:

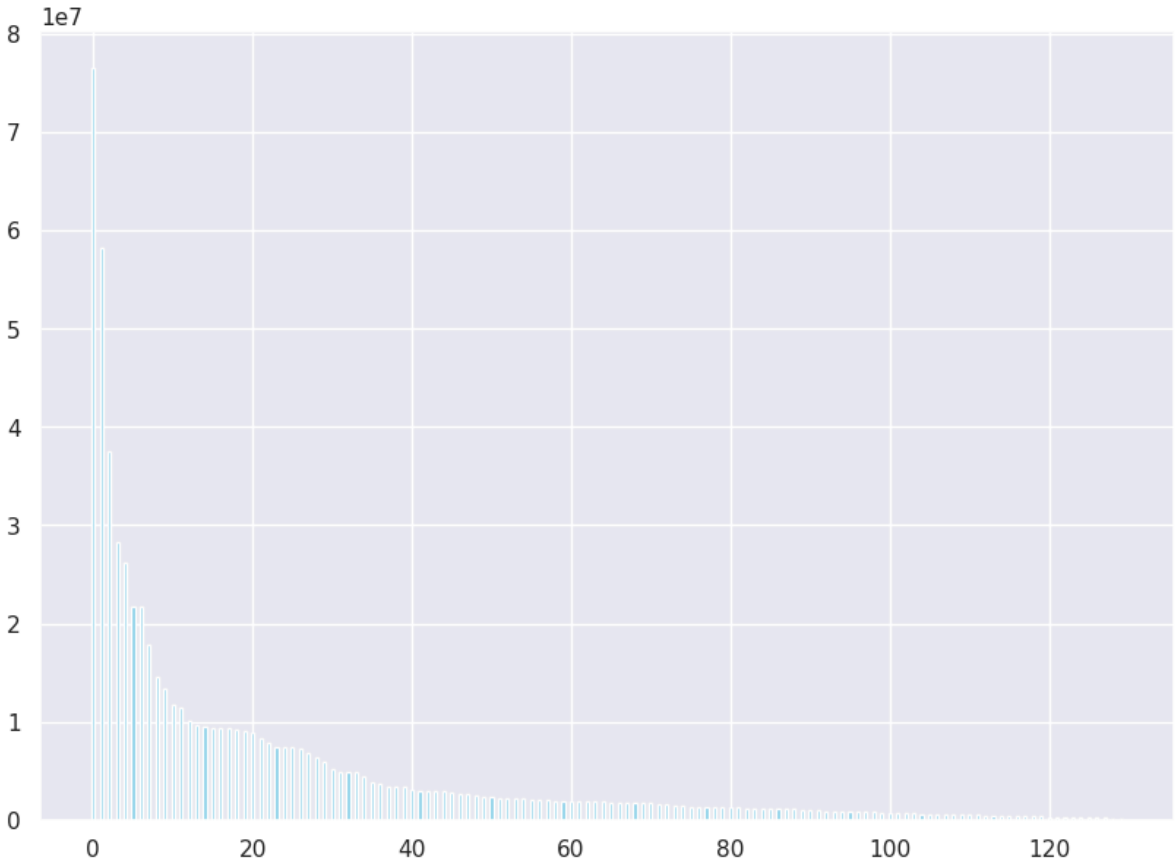
```
category_subscribers = df.groupby('subcategory')['num_subscribers'].sum()

# Finding the category with the maximum subscribers
max_subscribers_category = category_subscribers.idxmax()

# Sorting the data by subscribers for better visualization
sorted_subscribers = category_subscribers.sort_values(ascending=False)

plt.figure(figsize=(10, 7))
plt.barh(sorted_subscribers.index, sorted_subscribers.values, color='skyblue')
plt.xlabel('Number of Subscribers')
plt.ylabel('Subcategory')
plt.title('Number of Subscribers per Course Subcategory')

# Annotating the category with the maximum subscribers
plt.annotate(f"Max Subscribers: {max_subscribers_category}",
             xy=(sorted_subscribers[max_subscribers_category], max_subscribers_cate
             xytext=(0.7 * sorted_subscribers.max(), max_subscribers_category),
             fontsize=12, ha="center", color="black",
             arrowprops=dict(facecolor='black', shrink=0.05))

plt.show()
```



In [ ]:

```
# Calculating the total subscribers for each subcategory
category_subscribers = df.groupby('subcategory')['num_subscribers'].sum()

# Sorting the data by subscribers for better visualization
sorted_subscribers = category_subscribers.sort_values(ascending=False)

# Finding the category with the maximum subscribers
max_subscribers_category = sorted_subscribers.idxmax()

# Creating a figure and axis
plt.figure(figsize=(10, 7))
ax = plt.gca()

# Plotting the grouped bar chart
width = 0.4
ind = np.arange(len(sorted_subscribers))
bars = ax.bar(ind, sorted_subscribers.values, width, color='skyblue')

# Annotating the category with the maximum subscribers
for bar in bars:
    height = bar.get_height()
    if sorted_subscribers.index[bar.get_x()] == max_subscribers_category:
        ax.text(bar.get_x() + bar.get_width() / 2, height + 10, f"Max: {height}",
                ha='center', va='bottom', fontsize=10, color='black')

ax.set_xticks(ind)
ax.set_xticklabels(sorted_subscribers.index, rotation=45, ha='right')
ax.set_xlabel('Subcategory')
ax.set_ylabel('Number of Subscribers')
ax.set_title('Number of Subscribers per Course Subcategory')

plt.show()
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[46], line 22
     20 for bar in bars:
     21     height = bar.get_height()
---> 22     if sorted_subscribers.index[bar.get_x()] == max_subscribers_category:
     23         ax.text(bar.get_x() + bar.get_width() / 2, height + 10, f"Max: {he
ight}",
     24                 ha='center', va='bottom', fontsize=10, color='black')
     26 ax.set_xticks(ind)

File /opt/conda/lib/python3.10/site-packages/pandas/core/indexes/base.py:5320, in
Index.__getitem__(self, key)
   5317 if is_integer(key) or is_float(key):
   5318     # GH#44051 exclude bool, which would return a 2d ndarray
   5319     key = com.cast_scalar_indexer(key, warn_float=True)
-> 5320     return getitem(key)
   5322 if isinstance(key, slice):
   5323     # This case is separated from the conditional above to avoid
   5324     # pessimization com.is_bool_indexer and ndim checks.
   5325     result = getitem(key)

IndexError: only integers, slices (`:`), ellipsis (`...`), numpy.newaxis (`None`)
and integer or boolean arrays are valid indices
```

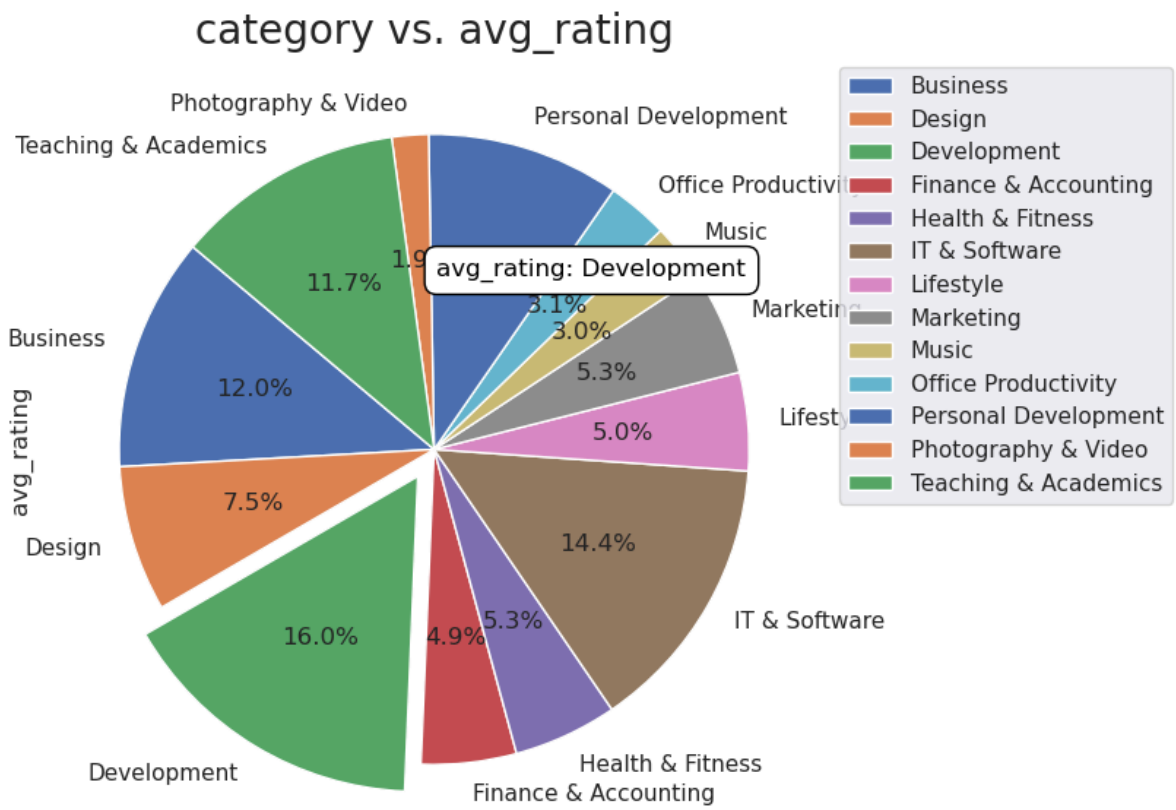


In [47]:
```
df.columns
```

Out[47]:
```
Index(['id', 'title', 'is_paid', 'price', 'headline', 'num_subscribers',
       'avg_rating', 'num_reviews', 'num_comments', 'num_lectures',
       'content_length_min', 'published_time', 'last_update_date', 'category',
       'subcategory', 'topic', 'language', 'course_url', 'instructor_name',
       'instructor_url', ' num_comments'],
      dtype='object')
```

In [49]:
```
category_subscribers = df.groupby('category')['avg_rating'].sum()

# Find the category with the maximum subscribers
```

```
max_subscribers_category = category_subscribers.idxmax()

# Create a pie chart for the distribution of subscribers across course subcategorie
plt.figure(figsize=(10, 7))

# Explode the slice corresponding to the category with the maximum subscribers
explode = [0.1 if subcategory == max_subscribers_category else 0 for subcategory ir

plt.title('category vs. avg_rating', fontsize=20)
category_subscribers.plot.pie(autopct="%1.1f%%", explode=explode, startangle=140)

# Move the legend outside the pie chart for clarity
plt.legend(labels=category_subscribers.index, loc="upper left", bbox_to_anchor=(1,

# Annotate the category with the maximum subscribers
plt.annotate(f"avg_rating: {max_subscribers_category}",
             xy=(0.5, 0.5), xytext=(0.5, 0.55),
             fontsize=12, ha="center", color="black",
             bbox=dict(facecolor='white', edgecolor='black', boxstyle='round,pad=0.

plt.show()
```



In [ ]:

```
category_subscribers = df.groupby('category')['price'].sum()

# Finding the category with the maximum subscribers
max_subscribers_category = category_subscribers.idxmax()

# Creating a pie chart for the distribution of subscribers across course subcategor
plt.figure(figsize=(10, 7))

# Exploding the slice corresponding to the category with the maximum subscribers
explode = [0.1 if subcategory == max_subscribers_category else 0 for subcategory ir

plt.title('category vs. price', fontsize=20)
category_subscribers.plot.pie(autopct="%1.1f%%", explode=explode, startangle=140)

# Moving the legend outside the pie chart for clarity
```

```
plt.legend(labels=category_subscribers.index, loc="upper left", bbox_to_anchor=(1,

# Annotating the category with the maximum subscribers
plt.annotate(f"avg_rating: {max_subscribers_category}",
             xy=(0.5, 0.5), xytext=(0.5, 0.55),
             fontsize=12, ha="center", color="black",
             bbox=dict(facecolor='white', edgecolor='black', boxstyle='round,pad=0.

plt.show()
```