Final Project in Data Analytics

Course Popularity on Udemy

Weilin Chen

Table of Contents

Use Case

MOOC (Massive Online Open Courses) platforms have taken the world of digital learning by storm. It has transformed the face of education and will continue to do so in the foreseeable future as technology evolves. Digital learning platforms have become an important channel for learning for not only individuals seeking continuous learning but also for companies seeking to update the skillset of their workforce.

My report aims to focus on an important KPI for the digital learning sector: the number of subscribers of a course. This is an essential metric both for the MOOC platforms seeking to evaluate their course offerings, for individuals choosing courses on these platforms, and for the future educators seeking to understand the needs of potential students. I will try to identity the relevant features that determine course popularity by focusing on one specific MOOC platform, Udemy. Furthermore, my data collection and analysis will serve as important stepping stones for the machine learning model that I will try to develop to help predict the popularity (number of subscribers) of planned or early stage courses on the Udemy platform.

Principle Findings of the Report

The main findings of this report are the following:

1. The top 3 course categories in terms of the number of subscribers are Development, Marketing, and Office Productivity. When it comes to rating, which is an essential metric due to its importance for the decision making of users in course selection, there is still much room for improvement: nearly half of the categories have an average rating less than 4 and no category has an average rating above 4.25. Moreover, the category IT & Software, which ranks high in popularity, performs particularly poorly in rating. This suggests that improving user feedback is a potential strategy of self-differentiation among competitors, both for IT & Software courses and in general.
2. The top 3 course subcategories in terms of the number of subscribers are Influence, Seach Engine Optimization, and Programming Languages. The average ratings are mostly good for the top 15 most popular course subcategories. Subcategories like Growth Hacking and Programming Languages, however, have high popularity and low ratings. Again, improving user feedback can be a potential strategy of self-differentiation in these two subcategories.
3. The data suggests that the courses with high potential are likely to lie in the fields of Programming Languages and Generative AI, thus indicating future trends for the field of digital learning.
4. Price elasticity is likely high in the digital learning landscape. While balancing price and quality remains an important task, low price tolerance suggests that strategic pricing remains important for staying competitive in the market.
5. The popular courses generally have a high ratio of good reviews (above 4). This highlights the importance of user feedback for staying competitive.

Data Collection

Data Sources and API use

The data used for this project are collected directly from Udemy with the use of the Udemy Affiliate API. It is a REST API that takes HTTP requests and responds to those requests in the JSON format.[1] The Udemy API allows for four types of API requests, for four types of information: a list of courses and information about them, course reviews (including ratings and sometimes comments by the reviewer), course curriculum, and course details. The course list API request allows for obtaining information about different courses in bulk whereas the other three types of API requests only allow for calls based on a specific course.

For this project I used the course list request, the course curriculum request, and the course reviews request. Since the information obtainable from the course details request overlaps with that of the course list request, I did not need to use the course details request for this project.

Methods and Process

My main collection tool was the course list API request which returns a list of courses and information about them. I adjusted the settings of the API call to maximize the amount of information (the number of courses) in the response, as well as ask for only the types of information needed (the future columns in my table). I created two Python functions to facilitate the process: one function for transforming the response of each API call from the JSON format to a Dataframe format with the specified columns, another function (which incorporates the first function) for making a series of API calls and returning the information in the dataframe format. I then proceeded to store the raw data in MySQL.

[1] https://www.udemy.com/developers/affiliate/

My data collection with the course list request proceeded in two phases. I first collected as many courses as possible without any specifications about the courses, then I collected courses by going through the list of primary categories that Udemy uses to label the courses. In total, 39,512 courses were collected.

With regard to the other two types of API requests, an important technical constraint was imposed by the fact that, unlike the course list request, which allows for obtaining information of a large quantity of courses with each request, the curriculum and course reviews requests only return information about one specific course. Given the time constraint and the necessity of proper API handling, it was therefore not possible to obtain reviews and curricula for all the courses retrieved with the course list request.

In the end I collected reviews and curricula for two specific groups of courses to enable comparative analyses later on. The first group is comprised of the 100 courses with the highest numbers of subscribers (ranging from about 180,000 to 1,800,000) in the data already collected. The second group is comprised of the first 100 courses with more than 10,000 subscribers. The idea is to compare the most popular courses with courses which are still popular and more moderately so (based on the distribution shown below of the numbers of subscribers across all courses, above 10,000 is still much above the 75 percentile). The two groups are stored together in the reviews and curricula tables.

```
count    3.951200e+04
mean     5.577677e+03
std      2.490657e+04
min      0.000000e+00
25%      7.800000e+01
50%      6.070000e+02
75%      3.179000e+03
max      1.837007e+06
Name: num_subscribers, dtype: float64
```

In the Course Reviews table, the number of reviews for the first group (100 000) is much larger than the number of reviews for the second group (9 190). This is due to the fact that the first group has many more subscribers and hence many more reviews.

Similar to the collection process with the course list request, I created two functions for using the reviews and the curriculum requests. I then stored the raw data in MySQL.

Data Cleaning and Data Preparation

The three tables are cleaned and prepared in turn:

Course list

- Dropped duplicated rows
- Dropped rows with null values
- Dropped the "discount" and "discount_price" columns which are not useful
- Changed the data type of the column "published_time" to datetime
- For the "price" column, removed currency sign, changed the value "free" to "0", and changed the datatype to float
- Changed the data type of "is_practice_test_course" to Boolean.
- Removed HTML tags from the "description column
- Changed blank values in the "level" column to "All Levels"
- Added "months_since_created" column (number of months since the course was published)
- Created "num_subscriberes_monthly" column ("num_subscribers" divided by "months_since_created")

<u>Course reviews</u>

- Changed the datatype of "created_time" column to datetime

<u>Curricula</u>

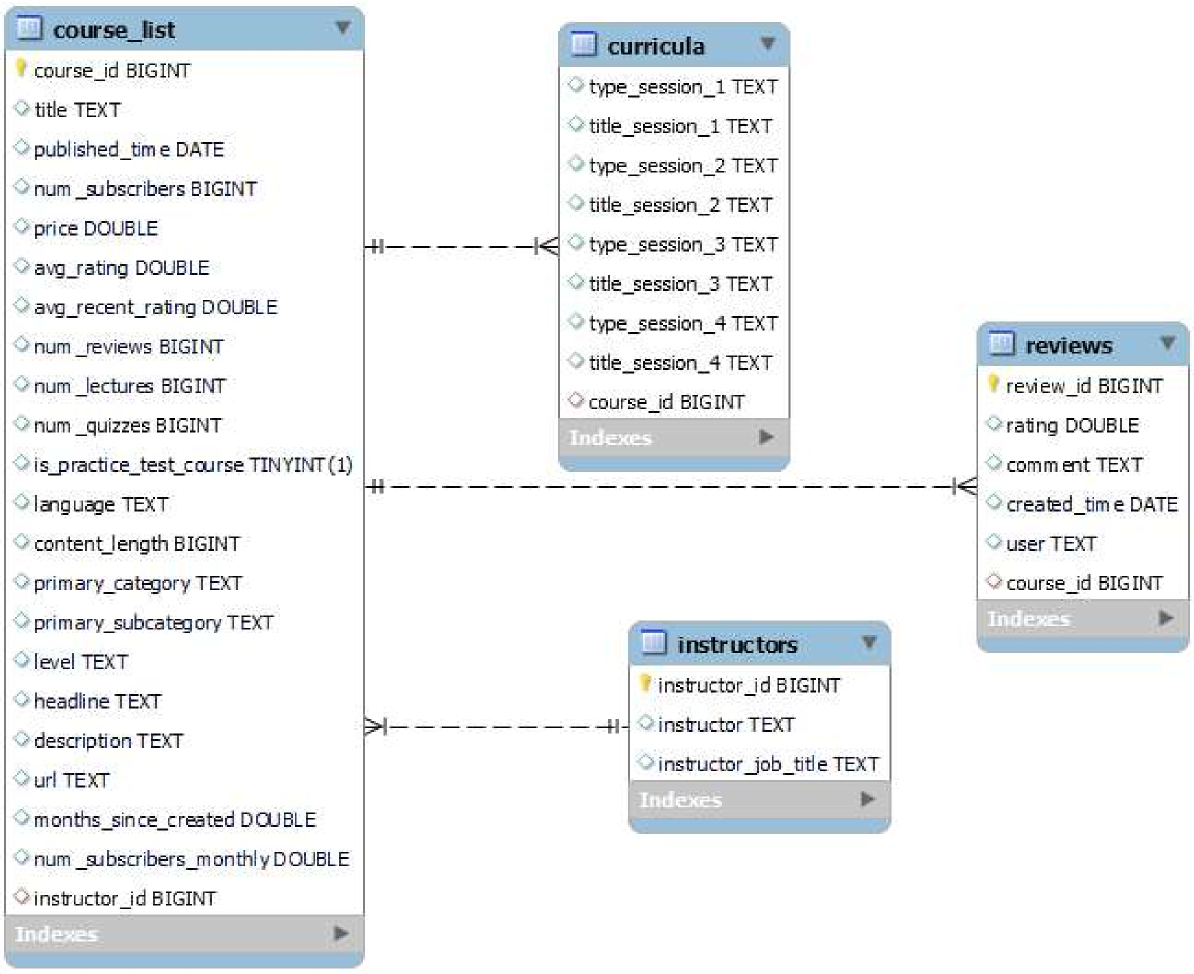
- No changes needed.

I also created a separate instructors table with a unique ID assigned to each instructor. I then merged this table with the Course List table and dropped the "instructor" and "instructor_job_title" columns from the Course List table. In the end, the Course List table contains only the "instructor_id" for information related to instructors. When it comes to database design, it is important to strike a balance between normalization and denormalization. While the data from the course list API request comes in a denormalized form, I believe that partly normalizing the data would increase the overall usability and flexibility of the entire dataset, given the fact that the same instructor can teach multiple classes and that more information about instructors can be easily added later on.

Data Storage and Creation of the Entity Relationship Diagram (ERD)

The four tables are then sent to MySQL for storage. I then created primary and foreign key constraints with the commands shown below, followed by the ERD:

```
alter table course_list add primary key (course_id);
alter table instructors add primary key (instructor_id);
alter table course_list add foreign key (instructor_id) references instructors(instructor_id);
alter table curricula add foreign key (course_id) references course_list(course_id);
alter table reviews add primary key (review_id);
alter table reviews add foreign key (course_id) references course_list(course_id);
```


course_list
course_id BIGINT
title TEXT
published_time DATE
num_subscribers BIGINT
price DOUBLE
avg_rating DOUBLE
avg_recent_rating DOUBLE
num_reviews BIGINT
num_lectures BIGINT
num_quizzes BIGINT
is_practice_test_course TINYINT(1)
language TEXT
content_length BIGINT
primary_category TEXT
primary_subcategory TEXT
level TEXT
headline TEXT
description TEXT
url TEXT
months_since_created DOUBLE
num_subscribers_monthly DOUBLE
instructor_id BIGINT
Indexes
curricula
type_session_1 TEXT
title_session_1 TEXT
type_session_2 TEXT
title_session_2 TEXT
type_session_3 TEXT
title_session_3 TEXT
type_session_4 TEXT
title_session_4 TEXT
course_id BIGINT
Indexes
reviews
review_id BIGINT
rating DOUBLE
comment TEXT
created_time DATE
user TEXT
course_id BIGINT
Indexes
instructors
instructor_id BIGINT
instructor TEXT
instructor_job_title TEXT
Indexes

The Dataset

The final dataset contains the following tables and columns:

Course List
(39512 rows)

1. course_id	Unique ID of the course
2. title	Title of the course
3. published_time	Time when the course was published
4. num_subscribers	Number of subscribers of the course
5. price	Price of the course
6. avg_rating	Average rating of the course
7. avg_recent_rating	Average recent rating of the course (period unknown)
8. num_reviews	Number of reviews given for the course
9. num_lectures	Number of lectures in the course
10. num_quizzes	Number of quizzes in the course
11. is_practice_test_course	Whether the course is a test preparation course
12. language	Language of the course
13. content_length	Length of course content
14. primary_category	The primary category to which the course belongs
15. primary_subcategory	The primary subcategory to which the course belongs
16. level	Level of the students targeted by the course
17. headline	Headline of the course
18. description	Brief description of the course
19. url	URL of the course webpage
20. months_since_created	Number of months since the course was created
21. num_subscribers_monthly	Number of subscribers per month
22. instructor_id	Unique ID assigned to each course instructor

Reviews

(109190 rows)

1. review_id	Unique ID of the review
2. rating	Rating given by the review
3. comment	Comment in the review
4. created_time	Time when the review was made
5. user	The user who gave the review
6. course_id	ID of the course reviewed

Curricula

(200 rows)

1. type_session_1	Session 1 type (a chapter or a lecture belonging to a chapter)
2. title_session_1	Session 1 title
3. type_session_2	Session 2 type
4. title_session_2	Session 2 title
5. type_session_3	Session 3 type
6. title_session_3	Session 3 title
7. type_session_4	Session 4 type
8. title_session_4	Session 4 title
9. course_id	ID of the course in question

Instructors

(19588 rows)

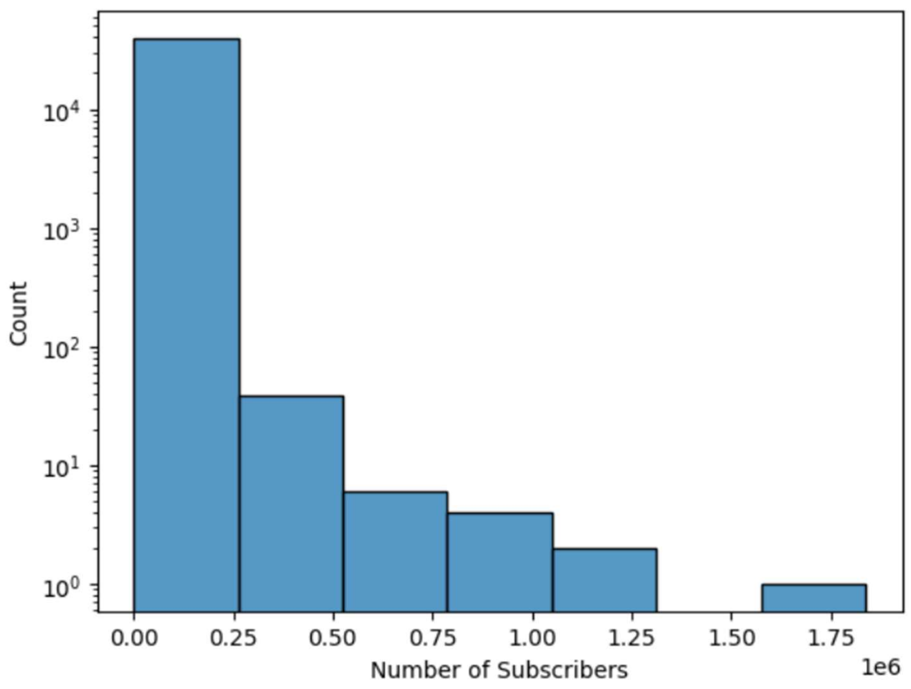
1. instructor_id	Unique ID assigned to each course instructor
2. instructor	Name of the instructor
3. instructor_job_title	Job title of the instructor

Database Type

The main reason for choosing MySQL rather than a NoSQL database for this project is the fact that the data collected is relational and needs to be stored in a tabular framework to facilitate the kind of analysis needed for this project. The moderate size of my data also means that it can be securely stored in MySQL.

Exploratory Data Analysis and Data Visualization

Since the metric that my project focuses on is the number of subscribers for a course, I began by plotting a histogram of the metric across all the courses. I plotted the chart on the logarithmic scale due to the highly uneven distribution. The result is a right skewed histogram, which is expected given the statistical summary chart on page 2 showing that the median is less than the mean.

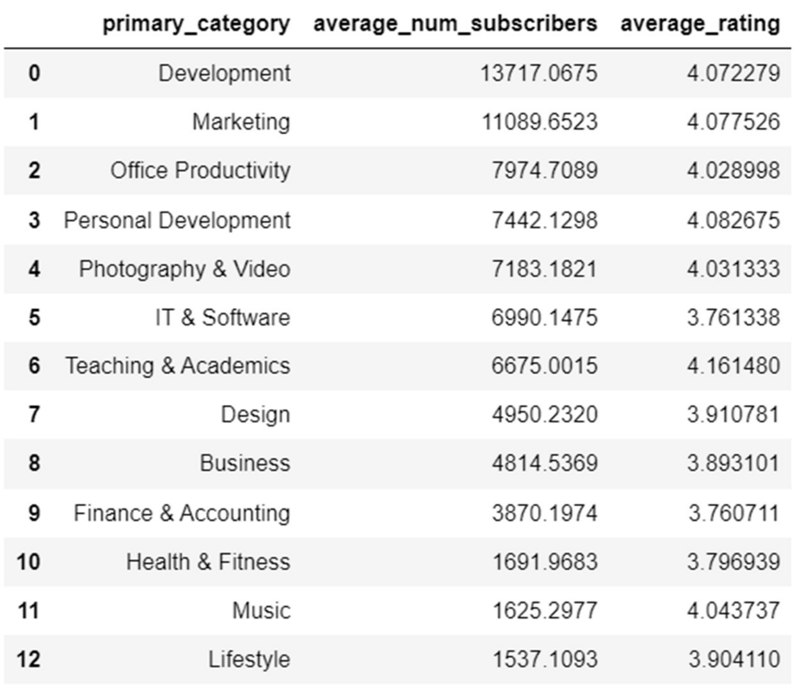


Number of Subscribers by Course Subject

One of the most important considerations, whether from the point of view of a MOOC platform like Udemy or someone thinking about starting an online course, concerns the themes of courses.

Udemy divides its course into 12 primary categories each of which is further divided into subcategories. I first created a table showing the primary categories ranked by the average number of subscribers, with the following SQL query:

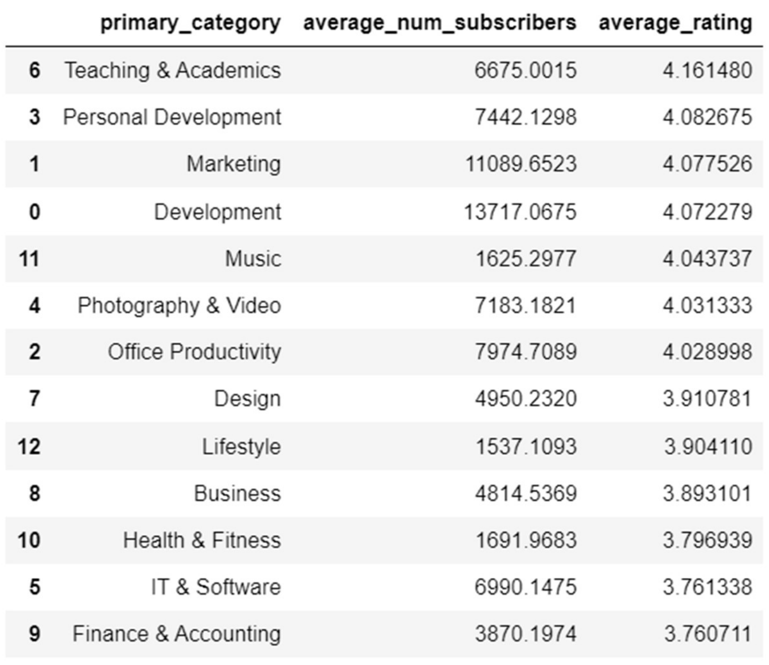
```
p_categories = pd.read_sql("select primary_category ,avg(num_subscribers) as
average_num_subscribers, avg(avg_rating) as average_rating from udemy_final.course_list group by
primary_category order by avg(num_subscribers) desc", engine)
```



	primary_category	average_num_subscribers	average_rating
0	Development	13717.0675	4.072279
1	Marketing	11089.6523	4.077526
2	Office Productivity	7974.7089	4.028998
3	Personal Development	7442.1298	4.082675
4	Photography & Video	7183.1821	4.031333
5	IT & Software	6990.1475	3.761338
6	Teaching & Academics	6675.0015	4.161480
7	Design	4950.2320	3.910781
8	Business	4814.5369	3.893101
9	Finance & Accounting	3870.1974	3.760711
10	Health & Fitness	1691.9683	3.796939
11	Music	1625.2977	4.043737
12	Lifestyle	1537.1093	3.904110

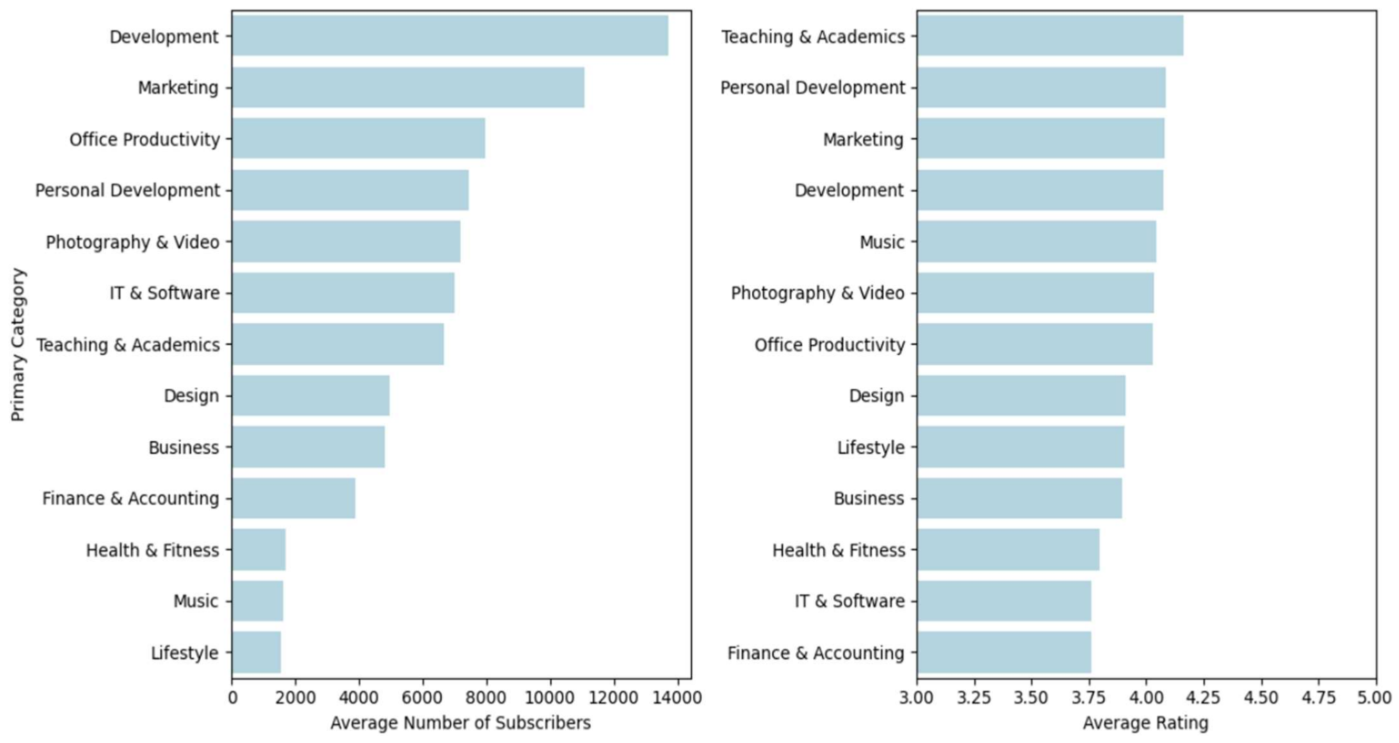
Then I sorted the table according to average rating with the following command:

```
p_categories_1 = p_categories.sort_values(by='average_rating', ascending=False)
```



	primary_category	average_num_subscribers	average_rating
6	Teaching & Academics	6675.0015	4.161480
3	Personal Development	7442.1298	4.082675
1	Marketing	11089.6523	4.077526
0	Development	13717.0675	4.072279
11	Music	1625.2977	4.043737
4	Photography & Video	7183.1821	4.031333
2	Office Productivity	7974.7089	4.028998
7	Design	4950.2320	3.910781
12	Lifestyle	1537.1093	3.904110
8	Business	4814.5369	3.893101
10	Health & Fitness	1691.9683	3.796939
5	IT & Software	6990.1475	3.761338
9	Finance & Accounting	3870.1974	3.760711

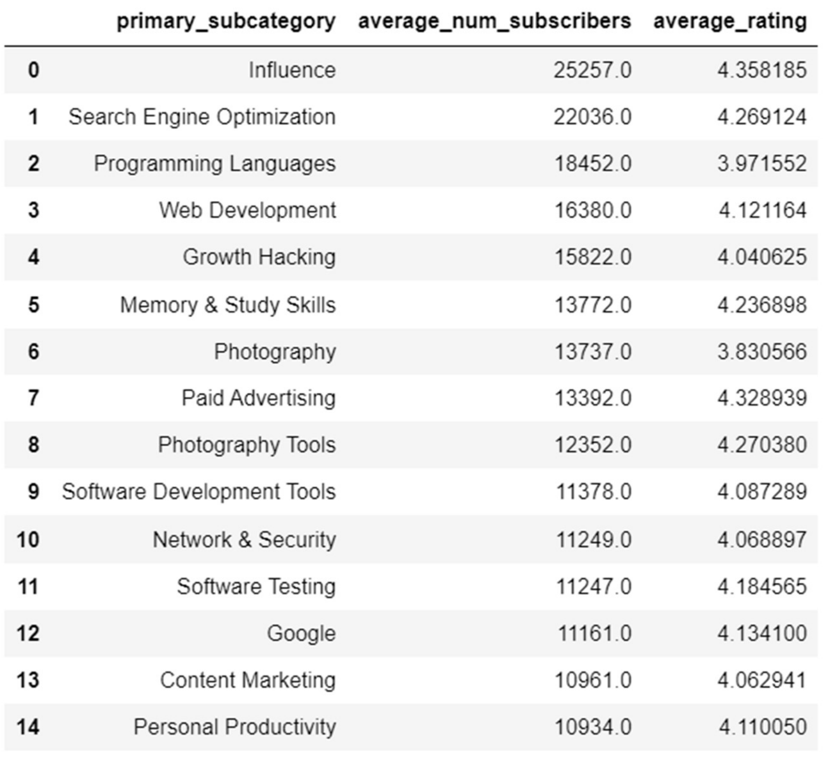
I then plotted the two tables to create two charts placed side by side (note: the scale of the x-axis of the chart on the right starts at 3.00):



It is interesting to note, from the visualizations above, that no category has yet attained an average rating of more than 4.25. Moreover, IT & Software, which is in the top half among the categories in terms of the number of subscribers, is almost at the bottom of the list when it comes to average rating. This observation suggests that a possible strategy for a course in IT & Software to differentiate itself from its potential competitors is to try to achieve a rating of more than 4.

I then move onto the primary subcategories. Since there are more than 100 subcategories in total, I selected only the top 15 in terms of the number of subscribers which I then sort by as the number of subscribers as well as by average rating:

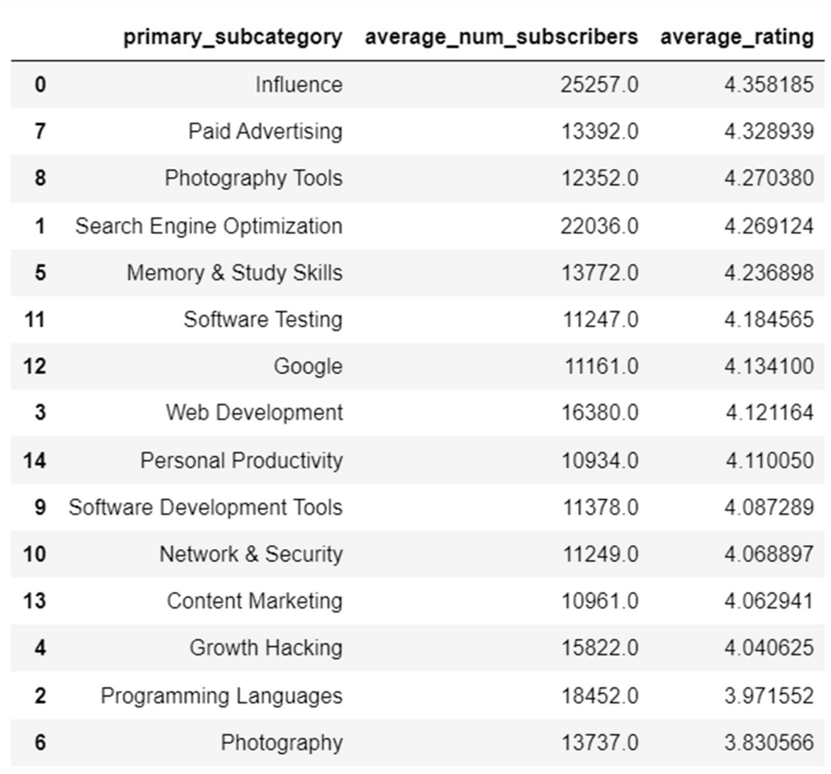
```
p_subcategories = pd.read_sql("select primary_subcategory ,round(avg(num_subscribers)) as
average_num_subscribers, avg(avg_rating) as average_rating from udemy_final.course_list group by
primary_subcategory order by avg(num_subscribers) desc limit 15", engine)
```



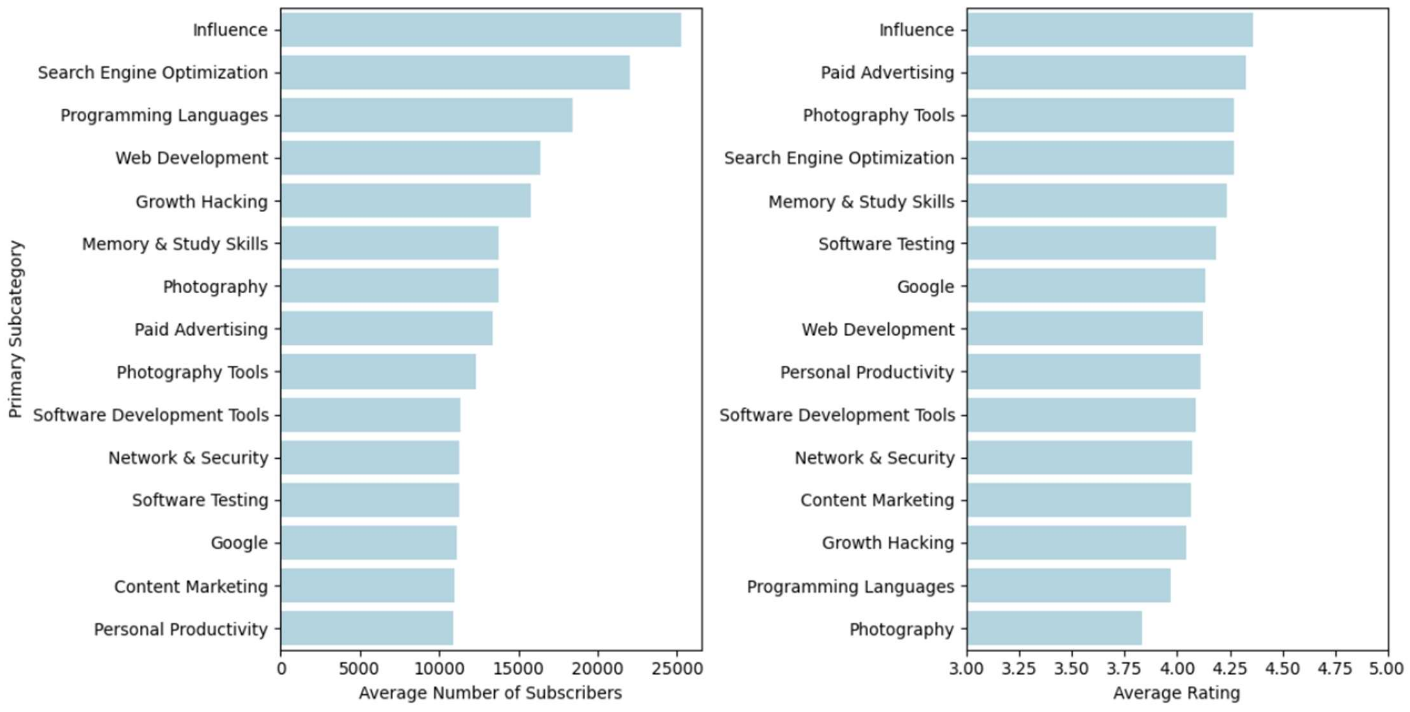
	primary_subcategory	average_num_subscribers	average_rating
0	Influence	25257.0	4.358185
1	Search Engine Optimization	22036.0	4.269124
2	Programming Languages	18452.0	3.971552
3	Web Development	16380.0	4.121164
4	Growth Hacking	15822.0	4.040625
5	Memory & Study Skills	13772.0	4.236898
6	Photography	13737.0	3.830566
7	Paid Advertising	13392.0	4.328939
8	Photography Tools	12352.0	4.270380
9	Software Development Tools	11378.0	4.087289
10	Network & Security	11249.0	4.068897
11	Software Testing	11247.0	4.184565
12	Google	11161.0	4.134100
13	Content Marketing	10961.0	4.062941
14	Personal Productivity	10934.0	4.110050

Then:

```
p_subcategories_1 = p_subcategories.sort_values(by='average_rating', ascending=False)
```



	primary_subcategory	average_num_subscribers	average_rating
0	Influence	25257.0	4.358185
7	Paid Advertising	13392.0	4.328939
8	Photography Tools	12352.0	4.270380
1	Search Engine Optimization	22036.0	4.269124
5	Memory & Study Skills	13772.0	4.236898
11	Software Testing	11247.0	4.184565
12	Google	11161.0	4.134100
3	Web Development	16380.0	4.121164
14	Personal Productivity	10934.0	4.110050
9	Software Development Tools	11378.0	4.087289
10	Network & Security	11249.0	4.068897
13	Content Marketing	10961.0	4.062941
4	Growth Hacking	15822.0	4.040625
2	Programming Languages	18452.0	3.971552
6	Photography	13737.0	3.830566



Four subcategories (Influence, Paid Advertising, Photography Tools, Search Engine Optimization) have ratings above 4.25. Growth Hacking, and especially Programming Languages, differ sharply across the two metrics (better than average popularity with lower than average rating).

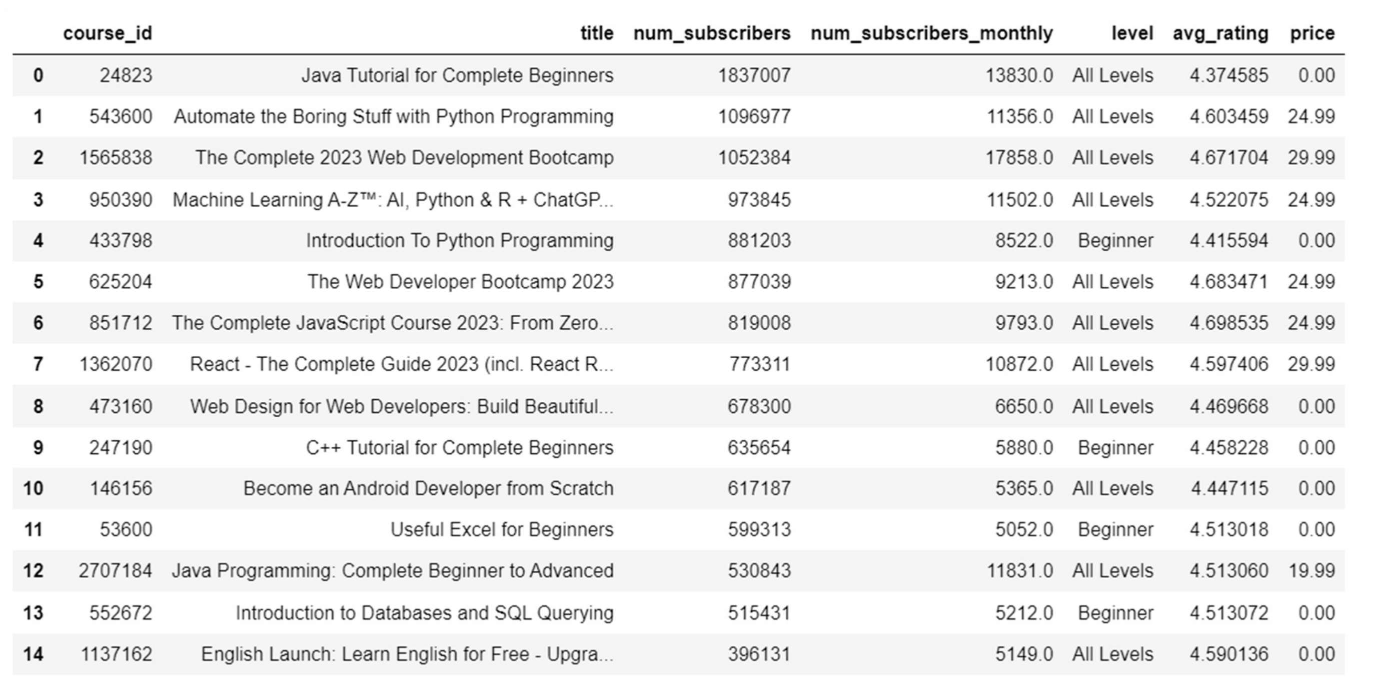
Again, this could inform decision making about possible ways for a course in these two subcategories to differentiate itself.

Identification of Courses with High Potential

An important question faced by any actor in the digital education sector concerns emerging trends, such as what courses are likely to be successful in the future. Judgments concerning what courses have high potential are crucial for the making of decisions such as what courses to start or which courses to target for marketing efforts. I took the following steps to identity courses with high potential:

I first selected the top 15 courses in terms of the number of subscribers:

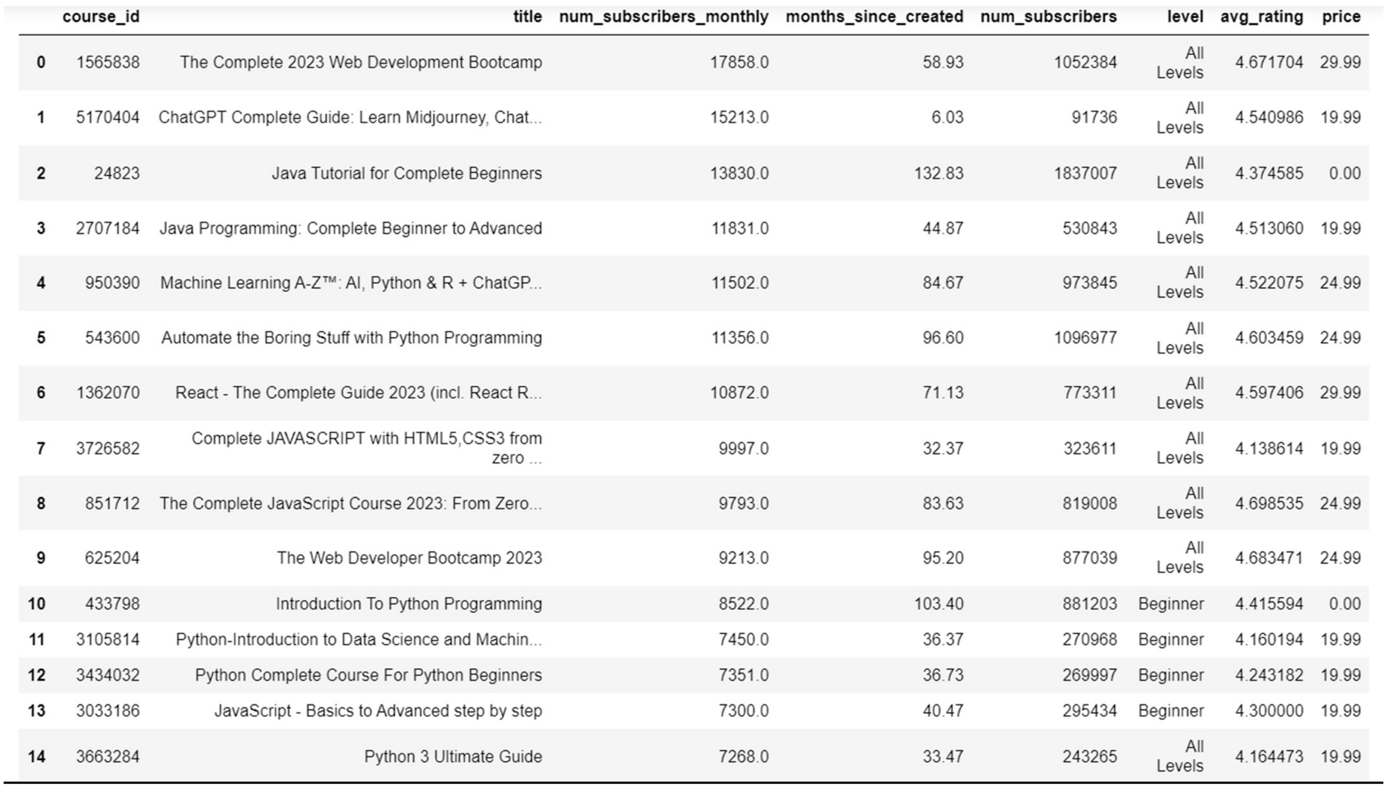
```
total_subscribers_rank = pd.read_sql("select course_id, title ,num_subscribers,
num_subscribers_monthly, level, avg_rating from udemy_final.course_list order by num_subscribers
desc limit 15", engine)
```



	course_id	title	num_subscribers	num_subscribers_monthly	level	avg_rating	price
0	24823	Java Tutorial for Complete Beginners	1837007	13830.0	All Levels	4.374585	0.00
1	543600	Automate the Boring Stuff with Python Programming	1096977	11356.0	All Levels	4.603459	24.99
2	1565838	The Complete 2023 Web Development Bootcamp	1052384	17858.0	All Levels	4.671704	29.99
3	950390	Machine Learning A-Z™: AI, Python & R + ChatGP...	973845	11502.0	All Levels	4.522075	24.99
4	433798	Introduction To Python Programming	881203	8522.0	Beginner	4.415594	0.00
5	625204	The Web Developer Bootcamp 2023	877039	9213.0	All Levels	4.683471	24.99
6	851712	The Complete JavaScript Course 2023: From Zero...	819008	9793.0	All Levels	4.698535	24.99
7	1362070	React - The Complete Guide 2023 (incl. React R...	773311	10872.0	All Levels	4.597406	29.99
8	473160	Web Design for Web Developers: Build Beautiful...	678300	6650.0	All Levels	4.469668	0.00
9	247190	C++ Tutorial for Complete Beginners	635654	5880.0	Beginner	4.458228	0.00
10	146156	Become an Android Developer from Scratch	617187	5365.0	All Levels	4.447115	0.00
11	53600	Useful Excel for Beginners	599313	5052.0	Beginner	4.513018	0.00
12	2707184	Java Programming: Complete Beginner to Advanced	530843	11831.0	All Levels	4.513060	19.99
13	552672	Introduction to Databases and SQL Querying	515431	5212.0	Beginner	4.513072	0.00
14	1137162	English Launch: Learn English for Free - Upgra...	396131	5149.0	All Levels	4.590136	0.00

Then I selected, from a pool of courses which are at least 3 months old (to rule out possible ephemerality), the top 15 courses in terms of the average number of subscribers by month (num_subscribers divided by months_since_created):

monthly_subscribers_rank = pd.read_sql("select course_id, title ,num_subscribers_monthly, months_since_created, num_subscribers, level, avg_rating, price from udemy_final.course_list where months_since_created >= 3 order by num_subscribers_monthly desc limit 15", engine)

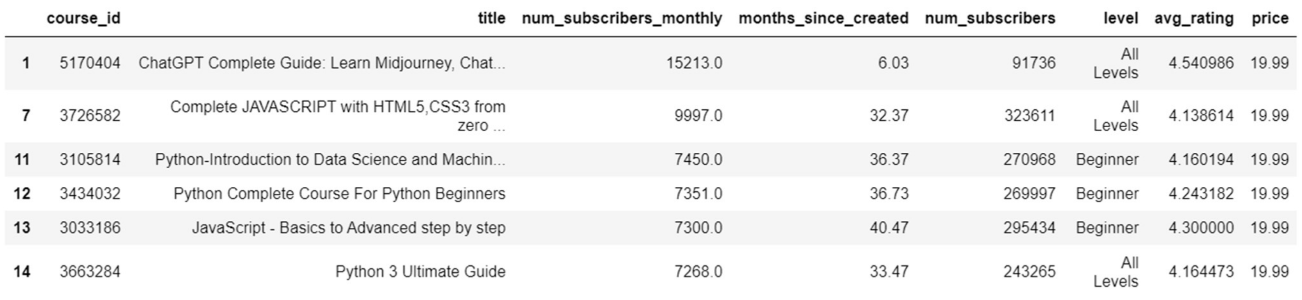


	course_id	title	num_subscribers_monthly	months_since_created	num_subscribers	level	avg_rating	price
0	1565838	The Complete 2023 Web Development Bootcamp	17858.0	58.93	1052384	All Levels	4.671704	29.99
1	5170404	ChatGPT Complete Guide: Learn Midjourney, Chat...	15213.0	6.03	91736	All Levels	4.540986	19.99
2	24823	Java Tutorial for Complete Beginners	13830.0	132.83	1837007	All Levels	4.374585	0.00
3	2707184	Java Programming: Complete Beginner to Advanced	11831.0	44.87	530843	All Levels	4.513060	19.99
4	950390	Machine Learning A-Z™: AI, Python & R + ChatGP...	11502.0	84.67	973845	All Levels	4.522075	24.99
5	543600	Automate the Boring Stuff with Python Programming	11356.0	96.60	1096977	All Levels	4.603459	24.99
6	1362070	React - The Complete Guide 2023 (incl. React R...	10872.0	71.13	773311	All Levels	4.597406	29.99
7	3726582	Complete JAVASCRIPT with HTML5,CSS3 from zero ...	9997.0	32.37	323611	All Levels	4.138614	19.99
8	851712	The Complete JavaScript Course 2023: From Zero...	9793.0	83.63	819008	All Levels	4.698535	24.99
9	625204	The Web Developer Bootcamp 2023	9213.0	95.20	877039	All Levels	4.683471	24.99
10	433798	Introduction To Python Programming	8522.0	103.40	881203	Beginner	4.415594	0.00
11	3105814	Python-Introduction to Data Science and Machin...	7450.0	36.37	270968	Beginner	4.160194	19.99
12	3434032	Python Complete Course For Python Beginners	7351.0	36.73	269997	Beginner	4.243182	19.99
13	3033186	JavaScript - Basics to Advanced step by step	7300.0	40.47	295434	Beginner	4.300000	19.99
14	3663284	Python 3 Ultimate Guide	7268.0	33.47	243265	All Levels	4.164473	19.99

Then I selected those courses which are in the second table but which are *not* in the first table, namely courses which have a very high number of subscribers per month and which will likely reach the top ranks in terms of total number of subscribers given more time:

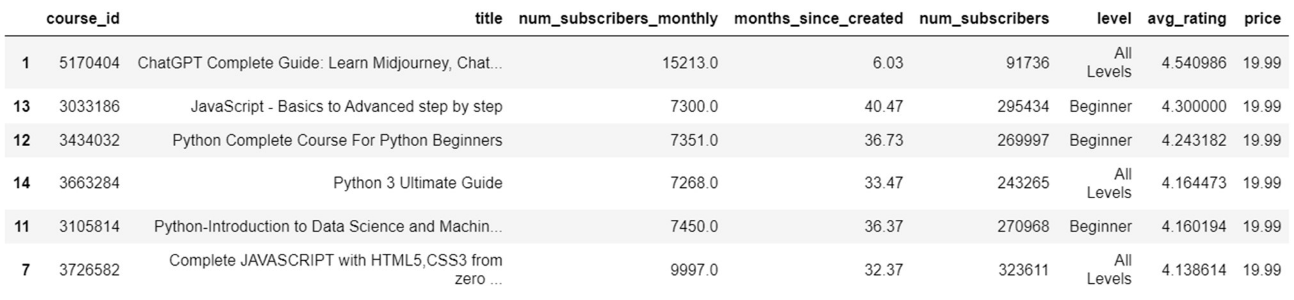
ids_potential = set(monthly_subscribers_rank['course_id'])-(set(monthly_subscribers_rank['course_id'])&set(total_subscribers_rank['course_id']))

courses_with_potential = monthly_subscribers_rank[monthly_subscribers_rank['course_id'].isin(ids_potential)]



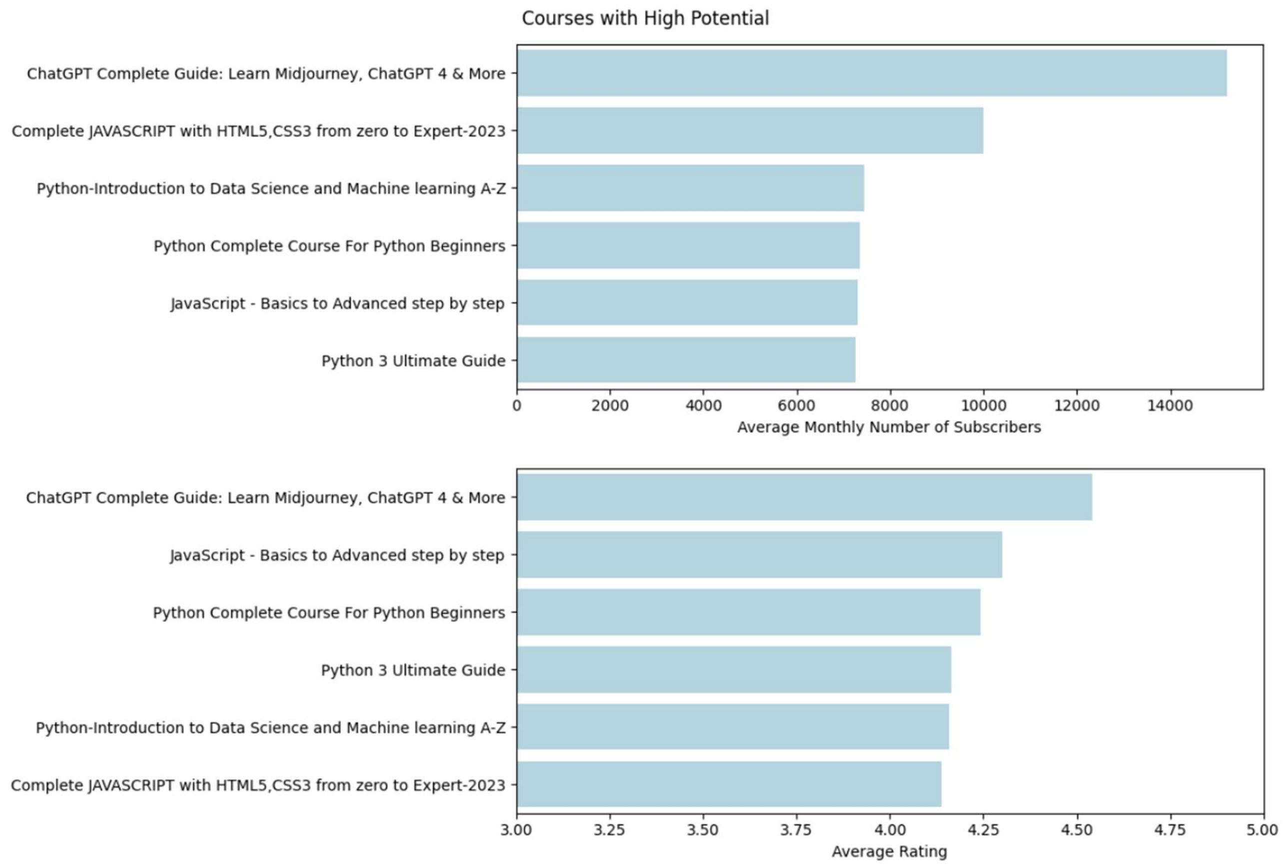
	course_id	title	num_subscribers_monthly	months_since_created	num_subscribers	level	avg_rating	price
1	5170404	ChatGPT Complete Guide: Learn Midjourney, Chat...	15213.0	6.03	91736	All Levels	4.540986	19.99
7	3726582	Complete JAVASCRIPT with HTML5,CSS3 from zero ...	9997.0	32.37	323611	All Levels	4.138614	19.99
11	3105814	Python-Introduction to Data Science and Machin...	7450.0	36.37	270968	Beginner	4.160194	19.99
12	3434032	Python Complete Course For Python Beginners	7351.0	36.73	269997	Beginner	4.243182	19.99
13	3033186	JavaScript - Basics to Advanced step by step	7300.0	40.47	295434	Beginner	4.300000	19.99
14	3663284	Python 3 Ultimate Guide	7268.0	33.47	243265	All Levels	4.164473	19.99

And ranked by average rating:



	course_id	title	num_subscribers_monthly	months_since_created	num_subscribers	level	avg_rating	price
1	5170404	ChatGPT Complete Guide: Learn Midjourney, Chat...	15213.0	6.03	91736	All Levels	4.540986	19.99
13	3033186	JavaScript - Basics to Advanced step by step	7300.0	40.47	295434	Beginner	4.300000	19.99
12	3434032	Python Complete Course For Python Beginners	7351.0	36.73	269997	Beginner	4.243182	19.99
14	3663284	Python 3 Ultimate Guide	7268.0	33.47	243265	All Levels	4.164473	19.99
11	3105814	Python-Introduction to Data Science and Machin...	7450.0	36.37	270968	Beginner	4.160194	19.99
7	3726582	Complete JAVASCRIPT with HTML5,CSS3 from zero ...	9997.0	32.37	323611	All Levels	4.138614	19.99

I then plotted both tables:

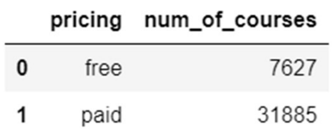


It is interesting to note that all the popular courses, whether by total number of subscribers or by the numbers of subscribers per month, have low entry requirements since they target students of either the beginner level or of all levels. Moreover, none of the courses with potential I identified are free though they are on the lowest price level (19.99 euros) in the dataset.

Pricing Strategy

Another important consideration is pricing. To find out the distribution of paid and free courses:

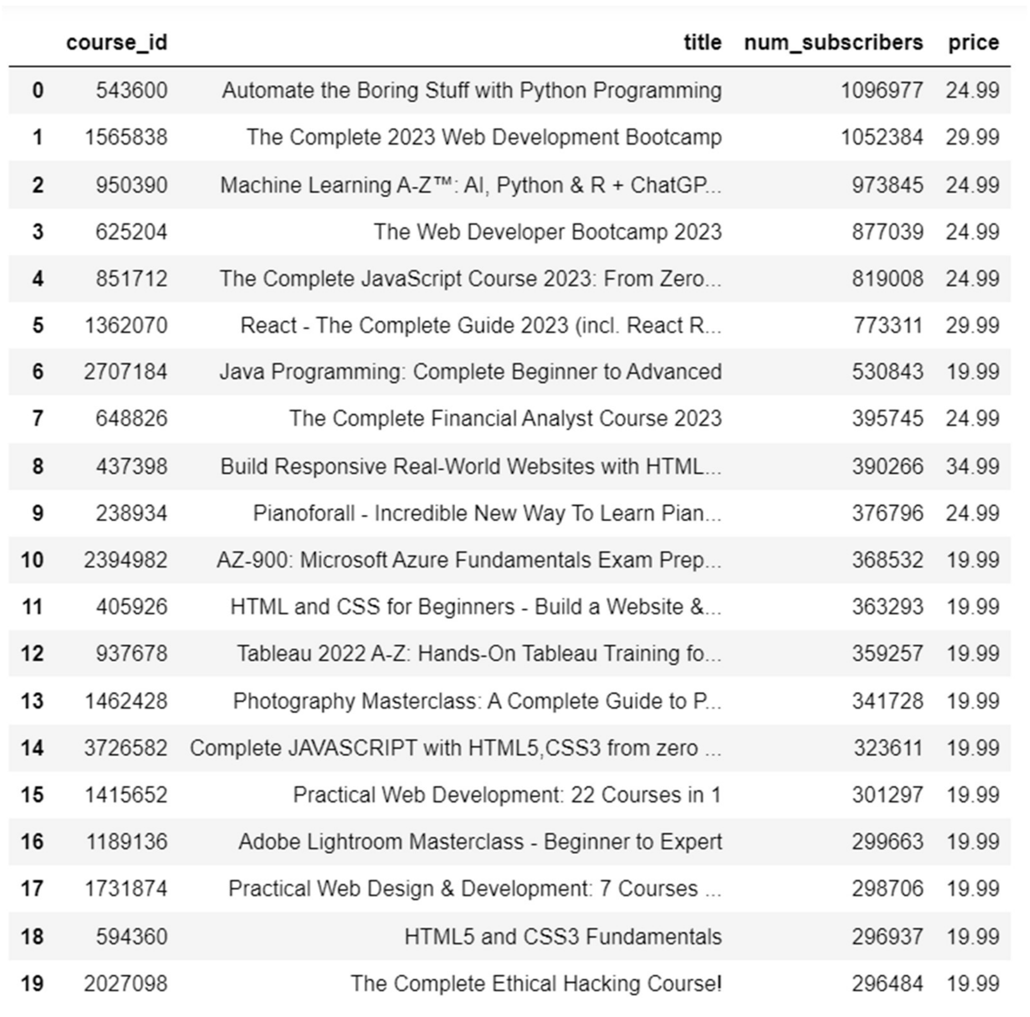
```
paid_or_free = pd.read_sql("select case when price > 0 then 'paid' else 'free' end as pricing, count(1)
as num_of_courses from udemy_final.course_list group by pricing", engine)
```



	pricing	num_of_courses
0	free	7627
1	paid	31885

Close to 20% of the courses are free. I then made a table of the top 20 paid courses in terms of the number of subscribers and their prices:

```
top_paid_courses = pd.read_sql("select course_id, title ,num_subscribers, price from
udemy_final.course_list where price > 0 order by num_subscribers desc limit 20", engine)
```



	course_id	title	num_subscribers	price
0	543600	Automate the Boring Stuff with Python Programming	1096977	24.99
1	1565838	The Complete 2023 Web Development Bootcamp	1052384	29.99
2	950390	Machine Learning A-Z™: AI, Python & R + ChatGP...	973845	24.99
3	625204	The Web Developer Bootcamp 2023	877039	24.99
4	851712	The Complete JavaScript Course 2023: From Zero...	819008	24.99
5	1362070	React - The Complete Guide 2023 (incl. React R...	773311	29.99
6	2707184	Java Programming: Complete Beginner to Advanced	530843	19.99
7	648826	The Complete Financial Analyst Course 2023	395745	24.99
8	437398	Build Responsive Real-World Websites with HTML...	390266	34.99
9	238934	Pianoforall - Incredible New Way To Learn Pian...	376796	24.99
10	2394982	AZ-900: Microsoft Azure Fundamentals Exam Prep...	368532	19.99
11	405926	HTML and CSS for Beginners - Build a Website &...	363293	19.99
12	937678	Tableau 2022 A-Z: Hands-On Tableau Training fo...	359257	19.99
13	1462428	Photography Masterclass: A Complete Guide to P...	341728	19.99
14	3726582	Complete JAVASCRIPT with HTML5,CSS3 from zero ...	323611	19.99
15	1415652	Practical Web Development: 22 Courses in 1	301297	19.99
16	1189136	Adobe Lightroom Masterclass - Beginner to Expert	299663	19.99
17	1731874	Practical Web Design & Development: 7 Courses ...	298706	19.99
18	594360	HTML5 and CSS3 Fundamentals	296937	19.99
19	2027098	The Complete Ethical Hacking Course!	296484	19.99

Here is the statistical summary of prices across all courses:

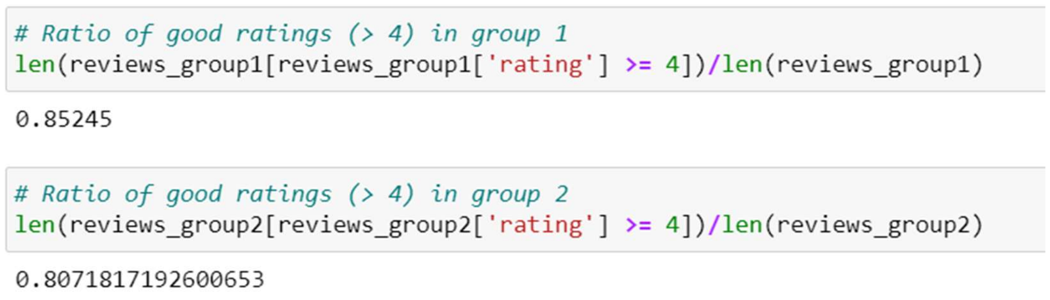
```
count    39512.000000
mean        18.797103
std         20.202196
min          0.000000
25%         19.990000
50%         19.990000
75%         19.990000
max        219.990000
Name: price, dtype: float64
```

As mentioned earlier, the lowest price among paid courses in the dataset is 19.99 euros. The fact that the average price is below that suggests that price tolerance is likely to be low among Udemy users. This can be seen from the fact that more than half of the most popular paid courses on the table on the previous page are in the low price range (less than 20). The potentially high price elasticity should make one hesitate before choosing to raise the price of a course. That being said, the table also shows that it is possible to charge a moderately higher price (such as above 25) and have still have a high number of subscribers.

Comparative Analysis of Reviews

Aside from the information about ratings in the Course List table, I also have rating information the Course Reviews table with courses belonging to two groups: the most popular courses and the first 100 courses with more than 10 000 subscribers. I would like to compare the proportions of good ratings of the two groups. I first obtain from the Course Reviews table in SQL the reviews for both groups before dividing them into two separate dataframes. Then I calculate the proportion of ratings above 4 in each group:

```
reviews_both = pd.read_sql('''select r.course_id, cl.title ,cl.num_subscribers, r.rating from
udemy_final.reviews r
inner join udemy_final.course_list cl
on r.course_id = cl.course_id
order by num_subscribers desc''', engine)
```



```
# Ratio of good ratings (> 4) in group 1
len(reviews_group1[reviews_group1['rating'] >= 4])/len(reviews_group1)
```

```
0.85245
```

```
# Ratio of good ratings (> 4) in group 2
len(reviews_group2[reviews_group2['rating'] >= 4])/len(reviews_group2)
```

```
0.8071817192600653
```

Both groups have a high percentage of good ratings in the most recent reviews on the Udemy platform. The first group has a higher ratio of good recent reviews, which is interesting given the fact the first group also has a much higher number of reviews (100 000 for the first group and 9190 for second).