Exploratory Data Analysis on Online Course Enrollment Data

Estimated time needed: **45** minutes

Before we keep our heads down building a cool personalized online courses recommender system, we always need to take a look at data first since it is the essence of any machine learning system. We need to explore what kind of data we will need and what such data would look like.

You have already learned it is important to perform initial investigations on the data and how to perform exploratory data analysis to find preliminary insights such as data patterns and to check assumptions with the help of summary statistics and graphical representations.

Now, let's apply your EDA skills to online courses related datasets such as course titles/genres and course enrollments.

Objectives

After completing this lab you will be able to:

- Identify keywords in course titles using a WordCloud
- Calculate the summary statistics and visualizations of the online course content dataset
- Determine popular course genres
- Calculate the summary statistics and create visualizations of the online course enrollment dataset
- Identify courses with the greatest number of enrolled students

---

Prepare and setup the lab environment

At the beginning, we need to install two required Python packages:

In [1]:
```
!pip install seaborn
!pip install wordcloud
!pip install pandas
!pip install matplotlib
```

```
Collecting seaborn
  Downloading seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
Collecting numpy!=1.24.0,>=1.20 (from seaborn)
  Downloading numpy-2.4.1-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_6
4.whl.metadata (6.6 kB)
Collecting pandas>=1.2 (from seaborn)
  Downloading pandas-2.3.3-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_6
4.whl.metadata (91 kB)
Collecting matplotlib!=3.6.1,>=3.4 (from seaborn)
  Downloading matplotlib-3.10.8-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x
86_64.whl.metadata (52 kB)
Collecting contourpy>=1.0.1 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading contourpy-1.3.3-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x8
6_64.whl.metadata (5.5 kB)
Collecting cycler>=0.10 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading fonttools-4.61.1-cp312-cp312-manylinux1_x86_64.manylinux2014_x86_6
4.manylinux_2_17_x86_64.manylinux_2_5_x86_64.whl.metadata (114 kB)
Collecting kiwisolver>=1.3.1 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading kiwisolver-1.4.9-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x8
6_64.whl.metadata (6.3 kB)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-
packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.2)
Collecting pillow>=8 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading pillow-12.1.0-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_
64.whl.metadata (8.8 kB)
Collecting pyparsing>=3 (from matplotlib!=3.6.1,>=3.4->seaborn)
  Downloading pyparsing-3.3.1-py3-none-any.whl.metadata (5.6 kB)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/
site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.12/site-pac
kages (from pandas>=1.2->seaborn) (2024.2)
Collecting tzdata>=2022.7 (from pandas>=1.2->seaborn)
  Downloading tzdata-2025.3-py2.py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.17.0)
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
Downloading matplotlib-3.10.8-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86
_64.whl (8.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.7/8.7 MB 125.8 MB/s eta 0:00:00
Downloading numpy-2.4.1-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_64.w
hl (16.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 16.4/16.4 MB 164.7 MB/s eta 0:00:00
Downloading pandas-2.3.3-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.
whl (12.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.4/12.4 MB 166.3 MB/s eta 0:00:00
Downloading contourpy-1.3.3-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_
64.whl (362 kB)
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Downloading fonttools-4.61.1-cp312-cp312-manylinux1_x86_64.manylinux2014_x86_64.m
anylinux_2_17_x86_64.manylinux_2_5_x86_64.whl (5.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.0/5.0 MB 111.2 MB/s eta 0:00:00
Downloading kiwisolver-1.4.9-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86_
64.whl (1.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 89.7 MB/s eta 0:00:00
Downloading pillow-12.1.0-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_64
.whl (7.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.0/7.0 MB 161.7 MB/s eta 0:00:00
Downloading pyparsing-3.3.1-py3-none-any.whl (121 kB)
```

```
Downloading tzdata-2025.3-py2.py3-none-any.whl (348 kB)
Installing collected packages: tzdata, pyparsing, pillow, numpy, kiwisolver, font
tools, cycler, pandas, contourpy, matplotlib, seaborn
Successfully installed contourpy-1.3.3 cycler-0.12.1 fonttools-4.61.1 kiwisolver-
1.4.9 matplotlib-3.10.8 numpy-2.4.1 pandas-2.3.3 pillow-12.1.0 pyparsing-3.3.1 se
aborn-0.13.2 tzdata-2025.3
Collecting wordcloud
  Downloading wordcloud-1.9.5-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86
_64.manylinux_2_28_x86_64.whl.metadata (3.4 kB)
Requirement already satisfied: numpy>=1.19 in /opt/conda/lib/python3.12/site-pack
ages (from wordcloud) (2.4.1)
Requirement already satisfied: pillow in /opt/conda/lib/python3.12/site-packages
(from wordcloud) (12.1.0)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.12/site-packa
ges (from wordcloud) (3.10.8)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.12/site
-packages (from matplotlib->wordcloud) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.12/site-pac
kages (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.12/sit
e-packages (from matplotlib->wordcloud) (4.61.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/conda/lib/python3.12/sit
e-packages (from matplotlib->wordcloud) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-
packages (from matplotlib->wordcloud) (24.2)
Requirement already satisfied: pyparsing>=3 in /opt/conda/lib/python3.12/site-pac
kages (from matplotlib->wordcloud) (3.3.1)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/
site-packages (from matplotlib->wordcloud) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.7->matplotlib->wordcloud) (1.17.0)
Downloading wordcloud-1.9.5-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86_6
4.manylinux_2_28_x86_64.whl (555 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 555.2/555.2 kB 24.1 MB/s eta 0:00:00
Installing collected packages: wordcloud
Successfully installed wordcloud-1.9.5
Requirement already satisfied: pandas in /opt/conda/lib/python3.12/site-packages
(2.3.3)
Requirement already satisfied: numpy>=1.26.0 in /opt/conda/lib/python3.12/site-pa
ckages (from pandas) (2.4.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python3.1
2/site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.12/site-pac
kages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.12/site-p
ackages (from pandas) (2025.3)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.8.2->pandas) (1.17.0)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.12/site-packa
ges (3.10.8)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.12/site
-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.12/site-pac
kages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.12/sit
e-packages (from matplotlib) (4.61.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/conda/lib/python3.12/sit
e-packages (from matplotlib) (1.4.9)
Requirement already satisfied: numpy>=1.23 in /opt/conda/lib/python3.12/site-pack
ages (from matplotlib) (2.4.1)
```

```
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-
packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /opt/conda/lib/python3.12/site-packag
es (from matplotlib) (12.1.0)
Requirement already satisfied: pyparsing>=3 in /opt/conda/lib/python3.12/site-pac
kages (from matplotlib) (3.3.1)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/
site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.7->matplotlib) (1.17.0)
```

and import necessary class/methods in the packages

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

%matplotlib inline
```

In [3]:
```
# also set a random state
rs = 123
```

Load and explore the dataset

First, let's load the datasets as `Pandas` dataframes and start some basic exploratory data analysis tasks on them.

In [4]:
```
# Point to the datasets stored on the cloud
course_genre_url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain
ratings_url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clou
```

In [5]:
```
course_df = pd.read_csv(course_genre_url)
ratings_df = pd.read_csv(ratings_url)
```

We will start by exploring the course genre dataset. Let's first check its columns/features name:

In [7]:
```
course_df.columns
```

Out[7]:
```
Index(['COURSE_ID', 'TITLE', 'Database', 'Python', 'CloudComputing',
       'DataAnalysis', 'Containers', 'MachineLearning', 'ComputerVision',
       'DataScience', 'BigData', 'Chatbot', 'R', 'BackendDev', 'FrontendDev',
       'Blockchain'],
      dtype='object')
```

We can see from the above cell output that the columns represent some metadata about a course such as its id, title, and genres. We can see that the course genres are some popular topics related to machine learning, databases, app developments, etc.

We can also check how many unique courses we have in this dataset:

In [8]:
```
course_df.shape[0]
```

Out[8]: 307

Then we can take a quick look at its header rows

In [9]:
```
course_df.head()
```

Out[9]:

	COURSE_ID	TITLE	Database	Python	CloudComputing	DataAnalysis	Cont
0	ML0201EN	robots are coming build iot apps with watson ...	0	0	0	0	
1	ML0122EN	accelerating deep learning with gpu	0	1	0	0	
2	GPXX0ZG0EN	consuming restful services using the reactive ...	0	0	0	0	
3	RP0105EN	analyzing big data in r using apache spark	1	0	0	1	
4	GPXX0Z2PEN	containerizing packaging and running a sprin...	0	0	0	0	

In [10]:
```
course_df.dtypes
```

Out[10]:
```
COURSE_ID          object
TITLE              object
Database            int64
Python              int64
CloudComputing      int64
DataAnalysis        int64
Containers          int64
MachineLearning     int64
ComputerVision      int64
DataScience         int64
BigData             int64
Chatbot             int64
R                   int64
BackendDev          int64
FrontendDev         int64
Blockchain          int64
dtype: object
```

As we can see, the `COURSE_ID` and `TITLE` are `str` datatypes and all the course genres are binary/int datatypes. Any genre column with value 1 means the course is associated with the course genre while 0 means the course is not.

For example, the following course `accelerating deep learning with gpu` is associated with genres `Python`, `MachineLearning`, and `DataScience`

In [11]:
```
course_df.iloc[1, ]
```

Out[11]:
```
COURSE_ID                                      ML0122EN
TITLE              accelerating deep learning with gpu
Database                                             0
Python                                               1
CloudComputing                                       0
DataAnalysis                                         0
Containers                                           0
MachineLearning                                      1
ComputerVision                                       0
DataScience                                          1
BigData                                              0
Chatbot                                              0
R                                                    0
BackendDev                                           0
FrontendDev                                          0
Blockchain                                           0
Name: 1, dtype: object
```

Next, let's focus on the course title column by trying to determine keywords in the titles. Those keywords in the titles may give us a more intuitive summary of what kind of courses we have in the dataset.

Plot a Word Cloud from Course Titles

First, let's join all the title values into one string, then we can conveniently get a wordcloud from the big string:

In [12]:
```
titles = " ".join(title for title in course_df['TITLE'].astype(str))
```

In [13]:
```
titles
```

Out[13]: 'robots are coming build iot apps with watson swift and node red acceleratin
g deep learning with gpu consuming restful services using the reactive jax rs c
lient analyzing big data in r using apache spark containerizing packaging and
running a spring boot application cloud native security conference data securi
ty data science bootcamp with r for university proffesors learn how to use dock
er containers for iterative development scorm test course create your first mon
godb database testing microservices with the arquillian managed container cloud
pak for integration essentials watson analytics for social media data science b
ootcamp with python for university professors advance create a cryptocurrency
trading algorithm in python data ai essentials securing java microservices wi
th eclipse microprofile json web token microprofile jwt enabling distributed
tracing in microservices with zipkin sql access for hadoop ybrid cloud conferen
ce ai pipelines lab dataops methodology data ai jumpstart your journey intr
oduction to open source end to end data science on cloudpak for data ai for eve
ryone master the basics serverless computing using cloud functions developer
i predicting customer satisfaction hybrid cloud conference serverless lab valid
ating constraints for javabeans in java microservices getting started with the
data apache spark makers build predicting financial performance of a company d
b2 fundamentals i using clustering methods for investment portfolio analysis ph
p web application on a lamp stack fundamentals of javascript through rock paper
scissors using r with databases using r with databases spark fundamentals ii da
ta science in insurance basic statistical analysis apply end to end security
to a cloud application spark overview for scala analytics testing a microprofil
e or jakarta ee application using microshed testing with an open liberty docker
container r 101 text analytics 101 how to build watson ai and swift apis and ma
ke money tmp data science bootcamp insurance risk assessment with montecarlo me
thod using apache spark statistics 201 statistics 301 build swift mobile apps
with watson ai services text analysis introduction to machine learning with sou
nd using hbase for real time access to your big data watson analytics 101 insur
ance business modelling and basic actuarial calculations accessing hadoop data
using hive beyond the basics istio and ibm cloud kubernetes service accelerati
ng deep learning with gpu text analytics 101 text analytics at scale data scien
ce bootcamp with python machine learning with apache systemml action classifica
tion task based on internet firewall logs container kubernetes essentials wit
h ibm cloud data science in health care advanced prognostication using neural
networks advanced machine deep learning for spam classification task visual d
ata analysis in banking secure analysis of credit card dataset data science in
health care advanced machine learning classification network traffic anomaly d
etection intrusion detection task getting started with node js getting starte
d with mysql command line an introduction to ibm cloud satellite introduction t
o quantum computing launch an ai hotdog detector as a serverless python app exp
loratory data analysis eda with pandas in banking text analytics 101 create t
ables and load data in mysql using phpmyadmin relational model concepts predict
ive modeling fundamentals i build an iot blockchain network for a supply chain
blockchain essentials getting started with postgresql command line ibm blockcha
in foundation developer keys and constraints in mysql exploring spark s graphx
data science in health care basic statistical analysis create tables and load
data in postgresql using pgadmin monitoring the metrics of java microservices u
sing eclipse microprofile metrics configuring microservices running in kubernet
es game playing ai with swift for tensorflow s4tf enabling cross origin resou
rce sharing cors in a restful java microservice getting started with db2 on c
loud apache pig 101 developing distributed applications using zookeeper control
ling hadoop jobs using oozie moving data into hadoop mapreduce and yarn deploy
a web server using python and ibm cloud engine acknowledging messages using mic
roprofile reactive messaging build a personal movie recommender with django man
aging and injecting dependencies into java microservices using contexts and dep
endency injection cdi spark fundamentals i deploying microservices to kuberne
tes getting started with open liberty consuming restful java microservices asyn
chronously using eclipse microprofile rest client statistics 101 analyzing big
data with a spreadsheet ui openrefine 101 solr 101 deep learning with tensorflo

w building fault tolerant microservices with the fallback annotation consuming restful java microservices with template interfaces using eclipse microprofile rest client build chatbots with watson assistant node red basics to bots normalization keys constraints in relational database ibm cloud essentials getting started with microservices with istio and ibm cloud kubernetes service playing tictactoe with reinforcement learning and openai gym deploying a microservice to openshift by using a kubernetes operator data analysis demos data journalism first steps skills and tools mathematical optimization for business problems db2 academic training db2 fundamentals ii scalable web applications on kubernetes data analysis using r 101 nosql and dbaas 101 kubernetes operators advanced project deploy a serverless app for image processing data science career talks data science with open data data science bootcamp with python for university professors bitcoin 101 data science hands on with open source tools data science methodology creating asynchronous java microservices using microprofile reactive messaging build a smart search form with algolia documenting restful apis using microprofile openapi reactive architecture distributed messaging patterns data science in agriculture land use classification reactive architecture cqrs event sourcing deep learning 101 reactive architecture reactive microservices reactive architecture domain driven design building robots with tjbot building cloud native and multicloud applications big data 101 modernizing java ee applications data science in agriculture prognostication using by neural network docker essentials a developer introduction accelerating deep learning with gpus reactive architecture building scalable systems build your own chatbots data science in agriculture basic statistical analysis and geo visualisation machine learning with python build your own chatbot machine learning dimensionality reduction machine learning with python consuming a restful java web service using json b and json p an introduction to ibm cloud for financial services data analysis with python testing reactive java microservices using microshed testing framework checking the health of java microservices by using kubernetes readiness and liveness probes scala 101 checking the health of java microservices by using eclipse microprofile health check train a hotdog image recognition model with python externalizing configuration for java microservices using eclipse microprofile config integrating restful services with a reactive system using the cql shell to execute keyspace operations in cassandra performing table and crud operations with cassandra how to create and publish guided projects and hands on labs building a simple restful java microservice using jax rs and json b hybrid cloud conference backend services for containers transform photos to sketches and paintings with opencv quick introduction to a b testing introduction to data science reactive architecture introduction to reactive systems data privacy fundamentals consuming a restful java web service with angularjs implement consumer driven contract testing for java microservices using the pact framework hadoop 101 introduction to cloud classification of yelp reviews using sentiment analysis spark mllib ibm cloud essentials v3 performing database operations in the cloudant dashboard working with databases in ibm cloudant python for data science introduction to containers kubernetes and openshift v2 simplifying data pipelines with apache kafka sql and relational databases 101 medical appointment data analysis data visualization with r implementing a graphql microservice using microprofile api to query and update data from multiple services consuming a restful java web service with reactjs building and testing a java web application with maven and open liberty building a hypermedia driven restful java microservice using hypermedia as the engine of application state hateoas data visualization with python digital analytics regression machine learning with r containerizing and running java microservices in docker containers deep learning with tensorflow r for data science accessing and persisting data in microservices using java persistence api jpa data science in health care basic prognostication and geo visualization enabling distributed tracing in java microservices using eclipse microprofile opentracing and the jaeger tracing system building cloud native and multicloud applications deploy an ai powered discord bot with a voice views in postgresql streaming updates from a microprofile reactive messaging microservice using server sent events sse consuming a restful

```
java web service with angular deep learning with tensorflow data science with s
cala relational database systems business intelligence and data warehousing nos
ql systems sql for data science \r\ndistributed computing with spark sql sql fo
r data science capstone project database management essentials using databases
with python process data from dirty to clean database architecture  scale  and
nosql with elasticsearch the nature of data and relational database design pyth
on scripting  files  inheritance  and databases relational database support for
data warehouses introduction to structured query language  sql  crash course on
python programming for everybody  getting started with python  python basics py
thon programming  a concise introduction introduction to python programming pyt
hon and statistics for financial analysis applied machine learning in python in
troduction to data science in python data analysis using python introduction to
cloud computing cloud computing basics  cloud 101  cloud computing foundations
cloud computing concepts  part 1 fundamentals of cloud computing cloud computin
g concepts  part 2 cloud computing applications  part 1  cloud systems and infr
astructure cloud computing applications  part 2  big data and applications in t
he cloud introduction to data analytics excel basics for data analysis introduc
tion to data analysis introduction to predictive modeling data analysis using p
ython data analysis with r programming data analysis with python excel fundamen
tals for data analysis exploratory data analysis for machine learning introduct
ion to data analysis using excel big data analysis  hive  spark sql  dataframes
and graphframes cloud virtualization  containers and apis alibaba cloud native
solutions and container service docker  basics machine learning machine learnin
g for all introduction to machine learning  language processing applied machine
learning in python build  train  and deploy ml pipelines using bert introductio
n to machine learning in production machine learning data lifecycle in producti
on deploying machine learning models in production exploratory data analysis fo
r machine learning advanced computer vision with tensorflow deep learning appli
cations for computer vision deep learning in computer vision computer vision ba
sics fundamentals of digital image and video processing introduction to tensorf
low for artificial intelligence  machine learning  and deep learning convolutio
nal neural networks in tensorflow introduction to data science in python a cras
h course in data science data science in real life data science fundamentals fo
r data analysts executive data science capstone introduction to big data big da
ta modeling and management systems machine learning with big data big data  ca
pstone project big data essentials  hdfs  mapreduce and spark rdd foundations f
or big data analysis with sql analyzing big data with sql fundamentals of big d
ata hadoop platform and application framework working with big data natural lan
guage processing with attention models natural language processing with sequenc
e models natural language processing with probabilistic models r programming da
ta analysis with r programming getting started with data visualization in r int
roduction to probability and data with r using r for regression and machine lea
rning in investment  data visualization in r with ggplot2 the r programming env
ironment html  css  and javascript for web developers javascript basics javascr
ipt  jquery  and json programming foundations with javascript  html and css fro
nt end web development with react introduction to web development interactivity
with javascript and jquery'
```

We also want to filter common stop words and some less meaningful words:

In [14]:
```
# English Stopwords
stopwords = set(STOPWORDS)
stopwords.update(["getting started", "using", "enabling", "template", "universit
```

Then, we create a `WordCloud` object and generate `wordcloud` from the titles.

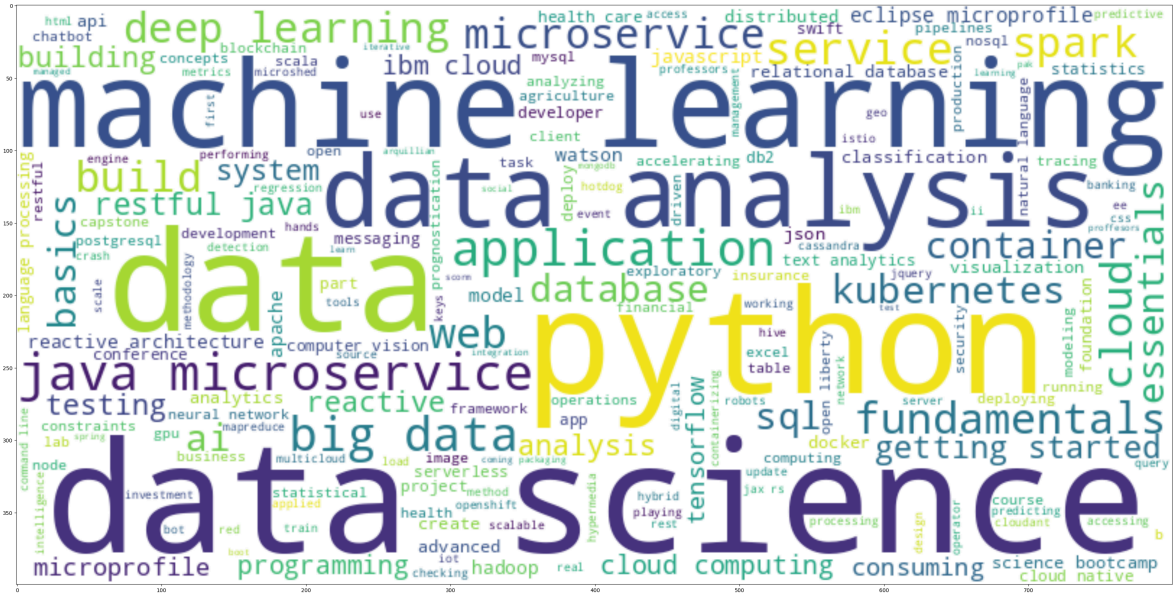
In [15]:
```
wordcloud = WordCloud(stopwords=stopwords, background_color="white", width=800,
```

In [16]:
```
wordcloud.generate(titles)
```

Out[16]:
```
<wordcloud.wordcloud.WordCloud at 0x728441841e50>
```

We can use `plt.imshow()` method to visualize the generated wordcloud:

In [17]:
```
# Disable axis display
plt.axis("off")
# Create a new figure with a specified size
plt.figure(figsize=(40,20))
# Adjust the layout to ensure tight spacing
plt.tight_layout(pad=0)
# Display the word cloud image with bilinear interpolation
plt.imshow(wordcloud, interpolation='bilinear')
# Show the plot
plt.show()
```



As we can see from the `wordcloud`, there are many popular IT related keywords such as python, data science, machine learning, big data, ai, tensorflow, container, cloud, etc. By looking at these keywords, we should have a general understanding that the courses in the dataset are focused on demanding IT skills.

Next, you need to perform some more detailed analysis on the course dataset.

TASK: Analyze Course Genres

First, you can try to find out which courses may be of interest you. For example, what are the all machine learning related courses?

TODO: Find all courses with genre `MachineLearning` == 1

In [23]:
```
# WRITE YOUR CODE HERE

ml_courses = course_df['MachineLearning']==1
print(ml_courses)
```

```
0      False
1       True
2      False
3      False
4      False
       ...
302    False
303    False
304    False
305    False
306    False
Name: MachineLearning, Length: 307, dtype: bool
```

▶ Click here for Hints

Similarly, you can try to find out what are the scalable machine learning courses?

TODO: Find all courses with genres `MachineLearning` == 1 and `BigData` == 1

In [25]:
```
# WRITE YOUR CODE HERE
ml_courses = course_df['BigData']==1
print(ml_courses)
```

```
0      False
1      False
2      False
3       True
4      False
       ...
302    False
303    False
304    False
305    False
306    False
Name: BigData, Length: 307, dtype: bool
```

▶ Click here for Hints

In [26]:
```
genres = course_df.columns[2:]
genres
```

Out[26]:
```
Index(['Database', 'Python', 'CloudComputing', 'DataAnalysis', 'Containers',
       'MachineLearning', 'ComputerVision', 'DataScience', 'BigData',
       'Chatbot', 'R', 'BackendDev', 'FrontendDev', 'Blockchain'],
      dtype='object')
```

Given all the course genres, now let's calculate the course count for each genre. e.g., there are 69 courses that belong to the `MachineLearning` genre and 23 courses that belong to the `Python` genre. We also want to sort the genre count list to find the most popular course genres.

TODO: Use the Pandas dataframe sum() and sort_values() methods to genreate a sorted course count per genre. You may also implement the task with different solutions

In [41]:
```
## WRITE YOUR CODE HERE
genre_sums = course_df[genres].sum()
genre_df = pd.DataFrame(genre_sums,columns=['Count'])
genre_df = genre_df.sort_values(by="Count",ascending=False)
print(genre_df)
```

```
                Count
BackendDev         78
MachineLearning    69
Database           60
DataAnalysis       56
DataScience        38
CloudComputing     37
BigData            36
Python             28
FrontendDev        19
Containers         17
R                  17
ComputerVision     10
Chatbot             4
Blockchain          4
```

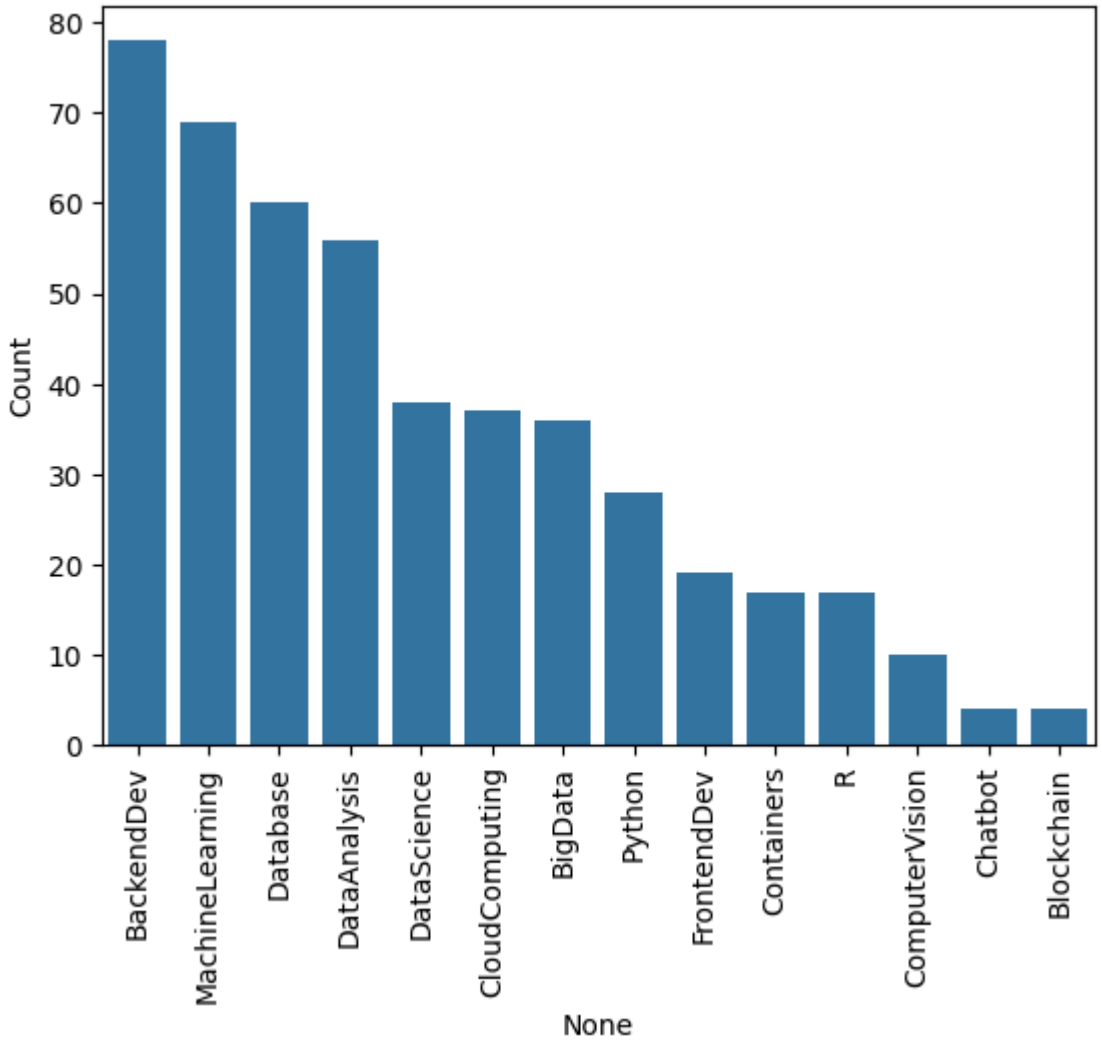
▶ Click here for Hints

We can also visualize course genre counts using a bar chart:

TODO: Use seaborn barplot or other plot methods to plot course genre counts using a barchart. The x-axis is the course genre and the y-axis is the course count per genre.

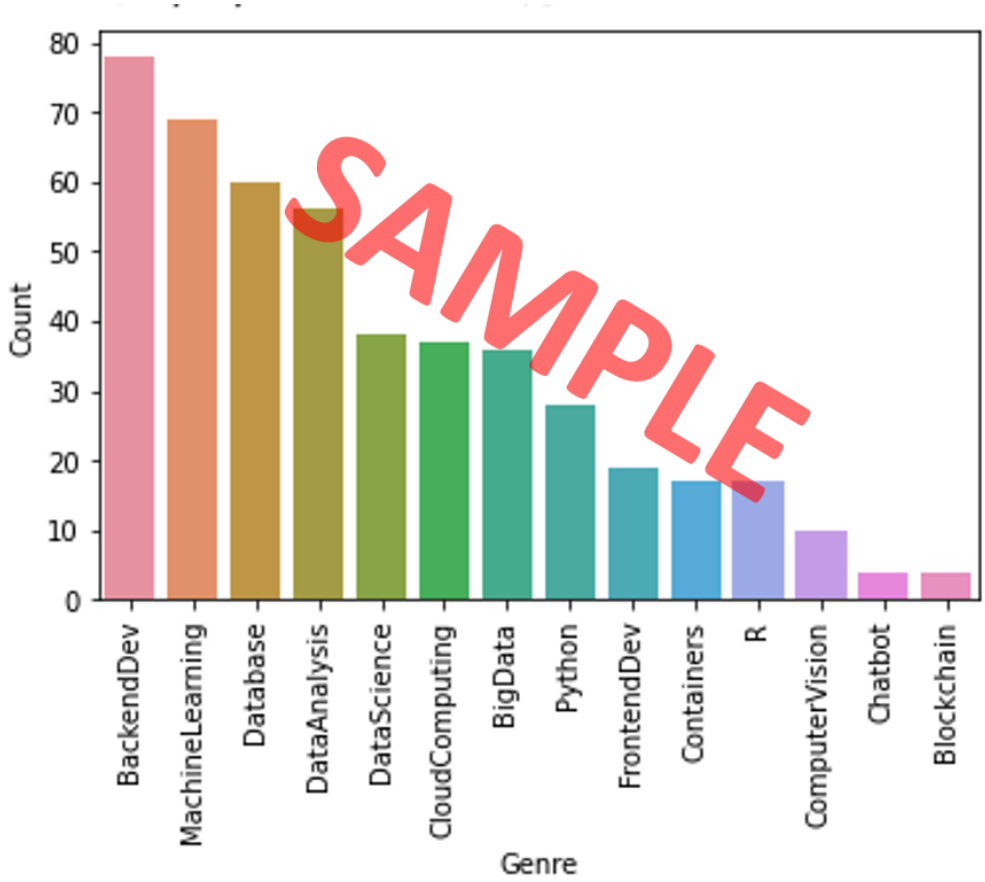
In [48]:
```
# WRITE YOUR CODE HERE
plot = sns.barplot(x=genre_df.index,y='Count', data=genre_df)
plot.set_xticklabels(plot.get_xticklabels(),rotation = 90)
plt.show()
```

```
/tmp/ipykernel_300/3204562845.py:3: UserWarning: set_ticklabels() should only be
used with a fixed number of ticks, i.e. after set_ticks() or using a FixedLocato
r.
  plot.set_xticklabels(plot.get_xticklabels(),rotation = 90)
```



▶ Click here for Hints

Your course genre barchart may look like the following:



Now, you should have some solid understanding about all course metadata including the keywords in titles and popular course genres. Next, we will switch gears to exploring course learners related datasets.

TASK: Analyze Course Enrollments

Let's first take a look at the course enrollments dataset.

In [49]:
```
ratings_df.head()
```

Out[49]:

	user	item	rating
0	1889878	CC0101EN	5
1	1342067	CL0101EN	3
2	1990814	ML0120ENv3	5
3	380098	BD0211EN	5
4	779563	DS0101EN	3

This dataset contains three colums, `user` representing a unique user id, `item` representing a course id, and `rating` representing the ratings given by the user.

In [50]:
```
ratings_df['rating'].unique()
```

Out[50]:
```
array([5, 3, 4])
```

The **rating** column consists of three potential values:

- A rating of `5` signifies that users who have enrolled in the course find it excellent and have given it the highest rating, thus recommending it to other learners.
- A rating of `4`, indicates that the enrolled users perceive the course as good and will recommend to the other learners,but suggest minor improvements.
- A rating of `3` indicates that enrolled users find the course below expectations and need significant modifications.

Let's see how many ratings we have in the dataset:

In [51]:
```
ratings_df.shape[0]
```

Out[51]:
```
233306
```

We have 233306 enrollments. In fact, each user is likely to interact with multiple items so let's find the rating counts for each user:

TODO: Apply Pandas' groupby() and size() method on the user column to aggregate the rating count for each user, then report the total number of users after aggregation.

In [55]:
```
# WRITE YOUR CODE HERE
ratings_df.groupby('user').size()
```

Out[55]:
```
user
2          61
4          44
5          59
7           1
8           3
          ..
2102054     8
2102356     1
2102680    11
2102983     1
2103039     1
Length: 33901, dtype: int64
```

▶ Click here for Hints

After the aggregation, you should get a new dataframe showing the rating count for each user. For example, user 4 rated 44 items and user 2 rated 61 items. Next, let's try to get some summary statistics and visualizations from the user ratings count dataframe.

TODO: Use describe() to report the statistics summary of the user enrollments.

In [56]:
```
## WRITE YOUR CODE HERE
ratings_df.describe()
```

Out[56]:

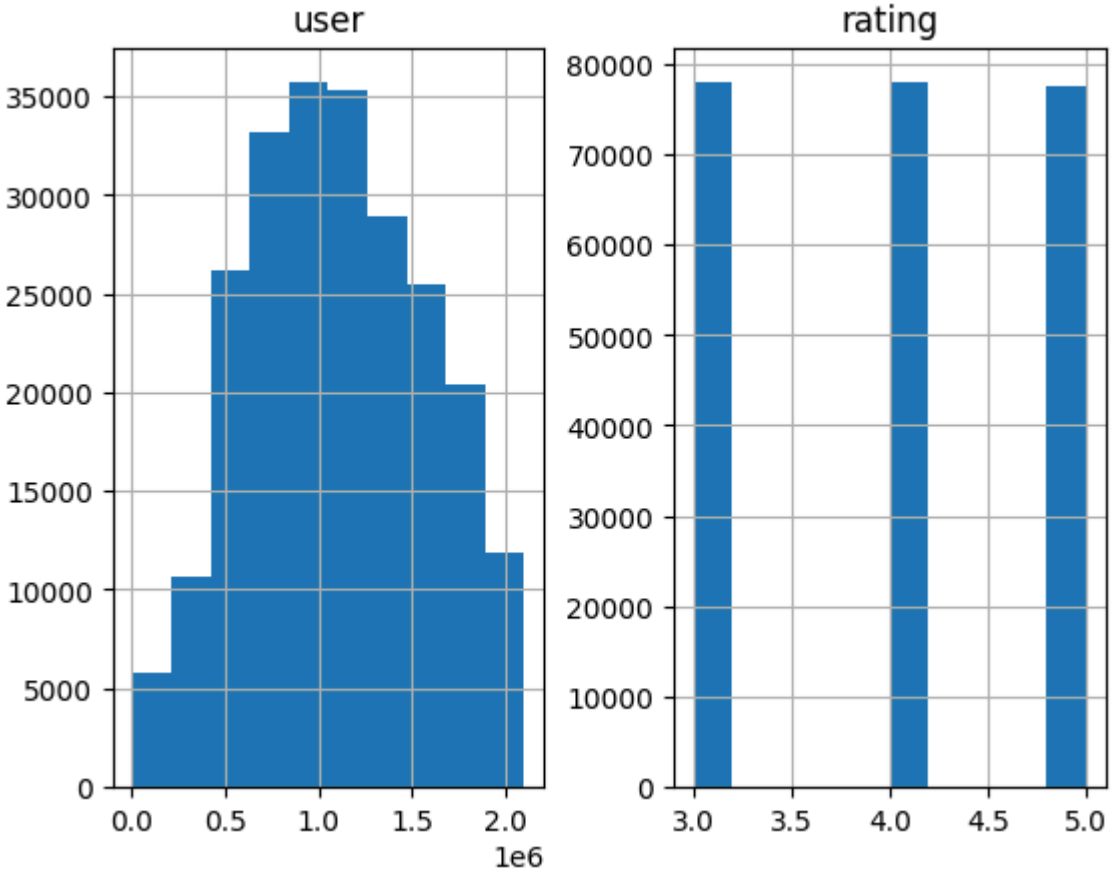
	user	rating
count	2.333060e+05	233306.000000
mean	1.099162e+06	3.998448
std	4.771661e+05	0.816058
min	2.000000e+00	3.000000
25%	7.210400e+05	3.000000
50%	1.080061e+06	4.000000
75%	1.466616e+06	5.000000
max	2.103039e+06	5.000000

We can also get a histogram showing the enrollment distributions, e.g., how many users rated just 1 item or how many rated 10 items, etc.

TODO: Plot the histogram of user rating counts.

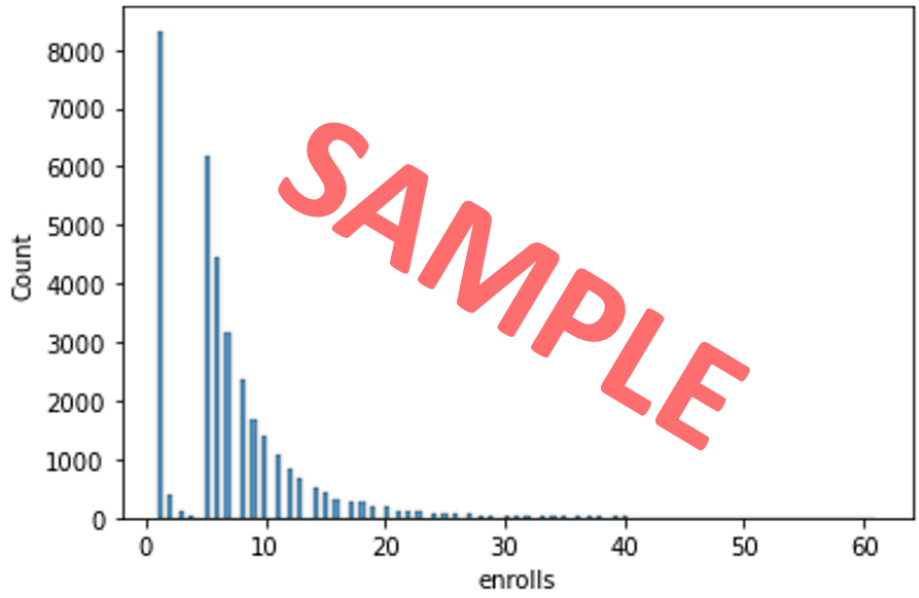
In [57]:
```
# WRITE YOUR CODE HERE
ratings_df.hist()
```

Out[57]:
```
array([[<Axes: title={'center': 'user'}>,
        <Axes: title={'center': 'rating'}>]], dtype=object)
```



▶ Click here for Hints

Your user enrollment histogram may look like the following:



Task: Find the Top-20 Most Popular Courses

Now we know how many items each user rated. Let's see the most popular 20 courses, i.e., items with the most rating counts.

TODO: Use Pandas groupby() and size() methods on the item column to aggregate the rating count for each item, then use the sort_values() method to sort the course enrollment count, and use the slice method to get the top 20 courses. You may also implement this task with different solutions

In [64]:
```
# WRITE YOUR CODE HERE
ratings = ratings_df.groupby(['item']).size().reset_index()
ratings.columns=['course', 'Ratings']
ratings.sort_values(by='Ratings', ascending=False).reset_index(drop=True)
```

Out[64]:

	course	Ratings
0	PY0101EN	14936
1	DS0101EN	14477
2	BD0101EN	13291
3	BD0111EN	10599
4	DA0101EN	8303
...	...	...
121	GPXX0QR3EN	1
122	DX0108EN	1
123	DX0106EN	1
124	ST0301EN	1
125	ST0201EN	1

126 rows × 2 columns

▶ Click here for Hints

Now you may only see the item IDs which do not indicate what kind of courses they are. To make it more clear, we need to join the course titles in the course metadata dataset (`course_df`) so that we can identify what the most popular courses are immediately:

TODO: Use Pandas merge() method to join the course_df (contains the course title column).

In [67]:
```
# WRITE YOUR CODE HERE
merged = pd.merge(ratings,course_df[['COURSE_ID', 'TITLE']],
                  how='left',
                  left_on='course',
                  right_on='COURSE_ID')
merged[['TITLE', 'Ratings']]
```

Out[67]:

	TITLE	Ratings
0	game playing ai with swift for tensorflow s4tf	383
1	blockchain essentials	6719
2	ibm blockchain foundation developer	2866
3	build an iot blockchain network for a supply c...	775
4	big data 101	13291
...	...	...
121	getting started with the data apache spark ma...	347
122	tmp data science bootcamp	35
123	data science bootcamp with python	2
124	watson analytics 101	2247
125	watson analytics for social media	56

126 rows × 2 columns

▶ Click here for Hints

Your most popular 20 courses list may look like the following:

	TITLE	Enrolls
0	python for data science	14936
1	introduction to data science	14477
2	big data 101	13291
3	hadoop 101	10599
4	data analysis with python	8303
5	data science methodology	7719
6	machine learning with python	7644
7	spark fundamentals i	7551
8	data science hands on with open source tools	7199
9	blockchain essentials	6719
10	data visualization with python	6709
11	deep learning 101	6323
12	build your own chatbot	5512
13	r for data science	5237
14	statistics 101	5015
15	introduction to cloud	4983
16	docker essentials a developer introduction	4480
17	sql and relational databases 101	3697
18	mapreduce and yarn	3670
19	data privacy fundamentals	3624

It seems that those popular courses take a huge amount of total ratings. Let's find the exact percentage.

In [68]:
```
# Get the total course enrollments again
total = ratings_df.shape[0]
total
```

Out[68]: 233306

In [69]:
```
top = 0
```

TODO: Get the percentage of the top-20 course enrollments.

In [78]:
```
# WRITE YOUR CODE HERE
top_courses = merged.sort_values(
    by='Ratings',
    ascending=False
).head(20)
top = sum(top_courses['Ratings'].values)
```

▶ Click here for Hints

In [79]:
```
print(f"Percentage of the top course enrollments {round((top * 100)/total, 2)}%"
```

```
Percentage of the top course enrollments 63.3%
```

Summary

Congratulations, you have completed the EDA lab! In this lab, you performed exploratory data analysis on the course metadata and course enrollments datasets and obtained some preliminary understanding.

As you know, these raw datasets can not be used to build recommender systems directly. Later, you will learn and practice how to process them and extract features for machine learning tasks.

Authors

Yan Luo

Other Contributors

Artem Arutyunov

toggle##

toggle|Date

toggle|-|-|-|-|

toggle|2021-10-25|1.0|Yan|Created