QuestCamp GCR assignment examples

Task 1: Web scraping

Task: Scrape a complex website (e.g., news site or e-commerce platform) using scrapy and beautifulsoup.
Focus: Data extraction, HTML parsing, handling pagination.
Key Skills: BeautifulSoup and Scrapy, HTTP requests, data structuring.

Helpful Links:
https://www.zenrows.com/blog/scrapy-python
https://medium.com/analytics-vidhya/webscraping-a-site-with-pagination-using-beautifulsoup-fa0a09804445

Task 2: Data analysis and visualization

Task: Analyse and visualise COVID-19 data from a public dataset, performing Exploratory data analysis (EDA) with descriptive statistics.
Focus: Your python code should include Data cleaning and preprocessing steps using pandas, analysis with pandas, and data visualisation with matplotlib. You should also write down a brief summary of your findings and interpretations.
Key Skills: Data manipulation, statistical analysis, chart creation

Useful links:
You should already have a good understanding of pandas. If need a recap, try 10 minutes to pandas: https://pandas.pydata.org/docs/user_guide/10min.html Or go through the cookbook: https://pandas.pydata.org/docs/user_guide/cookbook.html#cookbook

Pyplot tutorial : https://matplotlib.org/stable/tutorials/pyplot.html#sphx-glr-tutorials-pyplot-py
Or if you prefer video tutorial you can follow along with this one first:
https://www.youtube.com/watch?v=nzKy9GY12yo

Some Datasets you can choose from:
https://www.kaggle.com/datasets/allen-institute-for-ai/CORD-19-research-challenge
https://www.kaggle.com/datasets/imdevskp/corona-virus-report
https://www.kaggle.com/datasets/harshitstark/covid-19-global-statistics-dataset

Task 3: API integration

Task: Build an application that integrates with a random API from RapidAPI.
Focus: Working with REST APIs, data aggregation, error handling.
Key Skills: API authentication, JSON parsing, HTTP request handling.

- Submit a python file which does the following and ensure you have a basic understanding of the concepts mentioned

- Authenticates and makes HTTP requests to a chosen API. (if you don't know what HTTP request/response cycle is, look into it)
- Parses JSON responses and structures the data, using native Python
- Handles HTTP responses gracefully. (look into what status codes are and handle them accordingly)
- Displays processed data via print statements or simple visualizations (e.g., console tables, charts). use what you learnt in the previous section.
- Includes setup instructions and example usage in a README. (you should look into why this is important and also check the very basics of api documentation if unfamiliar with it)

Useful links:
Follow this whole tutorial once and read through it before building your own app with your desired API. https://rapidapi.com/blog/how-to-use-an-api-with-python/
Integrate API in 5 minutes https://www.youtube.com/watch?v=t1lbJvoPxwM
If you got this far then try building a basic flask app:
https://rapidapi.com/blog/python-restful-api-client

Some Ideas for api's you can integrate:
GitHub REST API (via RapidAPI or direct) you can fetch repositories, issues, or user stats
Hugging Face Inference API: Use pre-trained models for NLP tasks like sentiment analysis or summarization
Weather API Retrieve current weather or forecast data
Numbers API or Bored API can be used to generate trivia facts or random activities

Task 4: Natural Language Processing

Task: Build a Text Summarization Tool.
Create a simple extractive summarizer that takes an input text and returns the most important sentences.

Focus:
Understand tokenization and stop word removal.
Learn to use NLP libraries like spaCy or NLTK.
Apply text processing techniques for real-world use cases.

Helpful Learning Resources
1. Tokenization & Stopwords with NLTK
NLTK Tokenization Docs https://www.nltk.org/api/nltk.tokenize.html
NLTK Tutorial for Beginners https://www.youtube.com/watch?v=X2vAabgKiuM

2. Getting Started with spaCy
spaCy NLP Crash Course (freeCodeCamp, 1hr crash course)
https://www.youtube.com/watch?v=cgwDB1THUBY

Sample Data & TutorialsBBC News Articles Dataset:
Kaggle - BBC Articles https://www.kaggle.com/datasets/pariza/bbc-news-summary/data

Text Files for Practice:
Project Gutenberg (Free eBooks in plain text) https://www.gutenberg.org/

Sample Tutorial to follow
Text summarization with NLTK in python
https://www.kaggle.com/code/imkrkannan/text-summarization-with-nltk-in-python

Submission:
Write a Python script that:
Accepts a long text from a file or input.
Tokenizes it into sentences and words. (Dont forget to clean)
Removes stopwords.
Ranks sentences by importance. (you might want to score each sentence)
Returns a summary with top n sentences.

Bonus: you could connect it with your webscraping assignment, by scraping some articles and then summarizing them :)

Also attach screenshots of your output.

RUBRIC

Correct Use of NLP Tools
Using NLTK or spacy properly for tokenization and stopwords.
5 pts

Functional Summary Output
The code output a reasonable and readable summary
5 pts

Code Quality
The code should be modular, readable, and well-commented. It should be easily reusable rather than hardcoded for a particular usecase (& you would be expected to demonstrate the reusability)
5 pts

Sentence Ranking Logic
Implementing and being able to explain any basic scoring logic e.g. TF-IDF or another frequency based ranking.
5 pts

Data preprocessing.
You are expected to clean punctuation and numbers etc if they affect ranking.
5 pts

Module 8: Machine learning

Task: Build a movie recommendation system using scikit-learn.

Focus: Implementing collaborative filtering, handling input/output from users, evaluating model performance.

Submission:

Create a script that

- Loads a movie-ratings dataset .
- Implements user-based or item-based collaborative filtering .
- Handles user input, e.g. you can accept a user ID or input a movie+users rating for it, and find similar ones that user may like based on that info , you can customize the exact type of input expected from the user, based on your use case.
- Evaluate recommendations using metrics like RMSE or MAE.

- **Must:** Push your code to github (public repo), Add a README file with instructions to reproduce results and sample outputs, add screenshots in the readme , push to github and submit the github link.

Helpful links
https://realpython.com/build-recommendation-engine-collaborative-filtering/
this kaggle notebook can be a good starting point
https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system
Building a Movie Recommendation Engine in Python using Scikit-Learn:
https://medium.com/%40sumanadhikari/building-a-movie-recommendation-engine-using-scikit-learn-8dbb11c5aa4b

One of the many movies datasets you can find on kaggle:
https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata (you can look for more depending on what info you want to see about the movies)

Be ready to explain how you cleaned the dataset, and how did you compute similarity!

RUBRIC
Data preprocessing
Loading data, preprocessing, cleaning
5 pts

User based or item based collaborative filtering
implement either user-based or item-based collaborative filtering using scikit-learn. The logic behind similarity calculation (e.g. cosine similarity, k-NN) should be well-reasoned and correctly implemented.
10 pts

User input
Calculations based on user input, not hard coded
5 pts

Model evaluation
Evaluate using appropriate metrics like RMSE or MAE, and provide a brief interpretation of the results.
10 pts

Code quality
Well-structured, logically organized, and properly commented code. Variable/function names should be clear and meaningful
5 pts

Github and README

Submit a link to your github repo. A README file should be submitted as mentioned

5 pts