1. What is the curse of dimensionality? Explain how it can affect the performance of machine learning models. What techniques can be used to address this problem?

Ans 1)

The curse of dimensionality is that as the number of dimensions / features of the data increases, the complexity of the learning problem also increases.

It makes pattern recognition harder for ML models. In addition to this, it is more difficult to scale the model because of higher computational costs. Finding relevant features is also more time consuming because of the large number of features.

We can use dimensionality reduction techniques to address this problem. Performing Principal Component Analysis (PCA) reduces the dimensions of the data while retaining the important information.

2. What is normalisation in database design? Why is it important? Give an example of a situation where normalisation might be necessary.

Ans 2)

Normalization is a process that involves reducing a large database into smaller tables that are easier to manage, and establish relationships between them for lower access and update costs. It also ensures that each record is stored only once.

Normalization in database design is important for improving data integrity and reducing redundancies. It is also easier to maintain multiple small tables compared to a large one.

An example of a situation where normalization might be necessary is an e-commerce website. Initially, the database might store all customer information and order details in a single table. However, as the company scales, and the database grows, it could be difficult to manage all the data in a single table. To address this problem, the database could be normalized by breaking up the large table into several smaller tables, such as a customer table, product table, and an order table. Each table would have their own primary keys and relationships with other tables.

3. How do we run a python program in linux as a ‘systemctl’ service? Give us certain examples of such cases

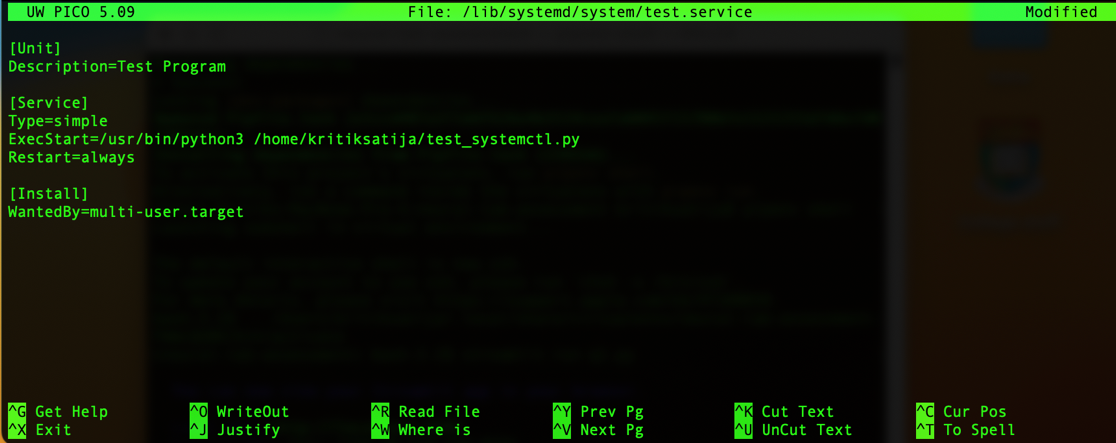
Ans 3)

To run a Python program in Linux as a ‘systemctl’ service, perform the following steps:-

1. Create a .service file in ‘/lib/systemd/system/’, say test.service. You can do that with the command: ‘sudo nano /lib/systemd/system/test.service’.

Thus, the path to the file will be ‘/lib/systemd/system/test.service’.

2. To ensure that the service would work, we need to specify ExecStart and Restart under [Service] in test.service. You can find an example code snippet below:-



The Python file I want to run is ‘test\_systemctl.py’ with path ‘/home/kritiksatija/test\_systemctl.py’.

3. Run the following command: ‘sudo systemctl daemon-reload’ to reload the system daemon

4. Run the following command: ‘sudo systemctl start test.service’ to start the service

5. You can check the status of the service by giving the command: ‘sudo systemctl status test.service’

Some examples where running a python program as a ‘systemctl’ service is useful:

1. ML models that need to be trained repeatedly can be run as a ‘systemctl’ service with a given schedule in the service file.

2. You can run a web server as soon as the system starts by scheduling it. This can easily be done by running the file as a ‘systemctl’ service.

4. With the dataset provided above, come up with faster and efficient techniques for fetching and processing the data from MongoDB to your local machine. Give us code examples what you’ve done and why’s the query faster (Code commenting)

There are a few ways to make the data fetching and processing faster from MongoDB to your local machine:-

1. Use table projection, which is the process of returning only the fields you need in a query. It reduces the time and space cost as we reduce the amount of data that needs to be transferred from MongoDB to your local machine. This could speed up the retrieval and processing steps.

2. Use indexes to speed up important queries. An index of 1 means that the field has values in ascending order, while -1 means it is in descending order.

Text

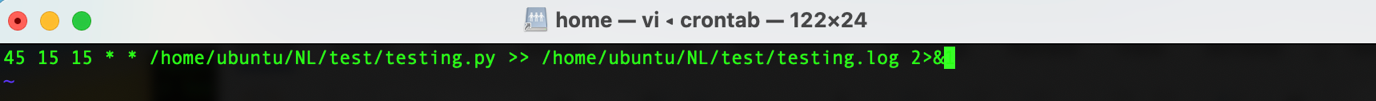
Description automatically generated

5. Create a cron job of a python file that needs to be run for every 15th of the month at 3:45PM GMT. It should execute from the startup and store the output in a log file. File name: testing.py in folder: /home/ubuntu/NL/test Add the code snippet to the document as well.

Ans 5)

Run the following command: ‘crontab -e”

Then enter the following command to create the cron job:



45 15 15 \* \* /home/ubuntu/NL/test/testing.py >> /home/ubuntu/NL/test/testing.log 2>&1

This assumes that the log file is called ‘testing.log’ and it’s path is ‘/home/ubuntu/testing/NL/test’. 2>&1 ensures that it will also redirect standard errors into the log file.