Ineuron Test On 10/03/2024 by Pranjal Sharma

Python-

1. Ans Python-1-> iNeuron_Test/python_1.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

2. Ans Python-2-> iNeuron_Test/Py_2.py at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

3. Ans Python-3-> iNeuron_Test/py_3.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

4. Ans Python-4-> iNeuron_Test/Py_4.py at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

SQL

1. The query will return all runners who have never won a race. It accomplishes this by finding all runner IDs in the runners table that are not present in the winner_id column of the races table.
   We can use Not Exists instead of Not In because - The problem arises when dealing with NULL values in the winner_id column. If a race has no recorded winner (NULL), the NOT IN operator may exclude all runners because NULL doesn't match any specific value.
   The NOT EXISTS operator checks if a subquery returns any rows.This alternative approach handles NULL values appropriately, ensuring that runners with no recorded wins are included in the results.

   ```
   SELECT *
   FROM runners r
   WHERE NOT EXISTS (SELECT 1
               FROM races rc
               WHERE rc.winner_id = r.id);
   ```

2. We can use left join in this we have to select the value where is b id is null,

```
Code- > select a.id
          from  test_a as a left join text_b as b on a.id=b.id
           Where b.id is null;
```

3. query to to get the list of users who took the a training lesson more than once in the same day, grouped by user and training lesson,

```
SELECT u.user_id, u.username, td.training_id, td.training_date, COUNT(*) AS count
FROM users u
INNER JOIN training_details td ON u.user_id = td.user_id
GROUP BY u.user_id, training_id, training_date
HAVING COUNT(*) > 1
ORDER BY training_date DESC;
```

4.
```
SELECT M.Manager_Id, M.Emp_name as Manager, AVG(E.Salary) AS
Average_Salary_Under_Manager
FROM Employee E
INNER JOIN Employee M ON E.Manager_Id = M.Emp_Id
GROUP BY M.Manager_Id, M.Emp_name
HAVING M.Manager_Id IS NOT NULL;
```

Stats

1. Six Sigma is based on the idea that all business processes can be measured and optimized. The term Six Sigma originated in manufacturing as a means of quality control. Six Sigma quality is achieved when long-term defect levels are below 3.4 defects per million opportunities (DPMO).
   Eg- Let's take a pen company as an example. A Six Sigma approach could be applied to ensure consistent ink flow in pens. With a Six Sigma process, only 3.4 out of every million pens produced would have an ink flow issue. This significantly reduces the chances of a customer receiving a faulty pen and improves overall customer satisfaction.

   Six Sigma goes beyond this statistical definition. It encompasses a structured methodology for continuous improvement, aiming to eliminate defects and enhance process efficiency. It involves identifying areas for improvement, analyzing data to pinpoint the root causes of issues, and implementing solutions to minimize variations and achieve better outcomes.

2. Many types of data wouldn't follow a log-normal or Gaussian (normal) distribution. Here are a couple of examples:

- **Categorical Data:** This type of data consists of distinct categories and doesn't represent numerical values. For instance, blood type (A, B, AB, O) or shoe size (5, 6, 7, etc.) wouldn't be suited for either distribution.
- **Uniform Data:** This refers to data where all values within a specific range are equally probable. Imagine a random number generator set to pick numbers between 1 and 10. The distribution of these generated numbers would be uniform, where every number (1, 2, 3, etc.) has an equal chance of being picked. This wouldn't follow the bell curve of a normal distribution or the skewed shape of a log-normal one.
- **Data with Arbitrary Lower or Upper Bounds:** Both normal and log-normal distributions can extend infinitely on one or both sides. However, some data sets may have restrictions on their values. For example, response times to a survey can't be negative, so a normal distribution wouldn't be ideal. Likewise, if measuring the number of people attending a concert, it can't be zero or negative, making a log-normal distribution a poor fit.
- **Exponential Distribution:** You're spot on! The time between events in a Poisson process, like calls at a center, indeed often follows an exponential distribution.
- **Poisson Distribution:** The number of arrivals at a bus stop, like you mentioned, perfectly aligns with a Poisson distribution where events (arrivals) occur independently within a fixed interval.
- **Bimodal Distribution:** The distribution of heights with separate peaks for males and females is a classic example of a bimodal distribution.

3. The five-number summary in statistics is a set of five key values that provide a concise overview of how a dataset is spread out. It offers insights into the data's center, spread, and potential outliers, without delving into every single data point.

Here's a breakdown of the five numbers it includes:

- ❖ **Minimum:** This is the absolute lowest value in the dataset.
- ❖ **First Quartile (Q1):** This represents the 25th percentile. It signifies that 25% of the data points fall below this value.
- ❖ **Median (Q2):** This is the middle value when the data is arranged in ascending order. It represents the 50th percentile, where 50% of the data falls below it and 50% falls above it.

❖ **Third Quartile (Q3):** This represents the 75th percentile. It signifies that 75% of the data points fall below this value.
❖ **Maximum:** This is the absolute highest value in the dataset.

Code for example-> iNeuron_Test/Stat_3.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

4. Correlation is a statistical measure that describes the extent to which two variables change together. It indicates the strength and direction of a linear relationship between two variables. The correlation coefficient, typically denoted by "r," ranges from -1 to 1, where:

- 1 indicates a perfect positive linear relationship,
- 0 indicates no linear relationship,
- -1 indicates a perfect negative linear relationship.

Code -> iNeuron_Test/Stats_4.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

Machine Learning

1. Answer ML-1-> iNeuron_Test/ML_1.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

2. Ans ML-2-> iNeuron_Test/ML_3.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

3. Ans- ML-3-> iNeuron_Test/ML_3.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

Deep Learning

1. a)
Implementing Deep Learning (DL) in a real-world application involves several steps and considerations. Below is a general guide on how to go about it:

i) **Define the Problem**:

- Clearly articulate the problem you want to solve with DL. This could be image recognition, natural language processing, speech recognition, etc.

ii) **Data Collection and Preparation**:

- Gather relevant and diverse data for training, validation, and testing. The quality and quantity of your data greatly impact the performance of your model.
- Clean and preprocess the data, handling missing values, scaling features, and ensuring a balanced dataset if possible.

iii) **Choose a DL Framework:**

- Select a deep learning framework based on your familiarity, community support, and specific requirements. Popular frameworks include TensorFlow, PyTorch, Keras.

iv) **Model Selection**:

- Choose an appropriate DL architecture for your problem. Convolutional Neural Networks (CNNs) are common for image-related tasks, Recurrent Neural Networks (RNNs) for sequential data, and Transformers for natural language processing.

v) **Model Design and Architecture**:

- Design the neural network architecture based on your problem. Define the input and output layers, as well as the hidden layers. Experiment with different architectures and hyperparameters to optimize performance.

vi) **Training the Model:**

- Split your data into training, validation, and test sets. Train the model on the training set, adjusting weights and biases through backpropagation and optimization algorithms.
- Monitor the model's performance on the validation set to avoid overfitting. Adjust the model architecture or hyperparameters accordingly.

vii) **Evaluate and Fine-Tune**:

- Evaluate the model's performance on the test set to assess its generalization to new, unseen data.
- Fine-tune the model by adjusting hyperparameters, collecting more data, or refining the model architecture based on the performance.

viii) **Deployment**:

- Once satisfied with the model's performance, deploy it in a real-world environment. This could involve integration into a web application, mobile app, or other systems.
- Consider the scalability, efficiency, and real-time requirements of your deployment platform.

ix) **Monitoring and Maintenance:**

- Implement monitoring mechanisms to track the model's performance in the production environment. This includes tracking metrics, detecting drift, and retraining the model as needed.
- Stay updated on new data and trends to ensure the model remains relevant and effective.

b) **Activation Function in Artificial Neural Networks:**

An activation function is a crucial component in artificial neural networks (ANNs). It introduces non-linearity to the network, allowing it to learn complex relationships and patterns in the data. Without activation functions, a neural network would behave like a linear model, making it limited in its ability to capture intricate patterns and representations.

- ❖ Purpose of Activation Functions:
  - ➢ **Non-linearity**: Activation functions introduce non-linearities to the model, enabling it to learn and represent complex, non-linear relationships in the data.
  - ➢ **Enable Learning of Hierarchical Features**: Non-linear activation functions allow neural networks to learn hierarchical representations of features, which is essential for handling intricate patterns in data.
- ❖ Common Activation Functions:
  - ➢ Sigmoid: Used in the output layer for binary classification problems. It squashes the output between 0 and 1.
  - ➢ Hyperbolic Tangent (tanh): Similar to sigmoid but with a range between -1 and 1. It helps mitigate the vanishing gradient problem.
  - ➢ Rectified Linear Unit (ReLU): Widely used in hidden layers, ReLU introduces non-linearity by setting negative values to zero.
  - ➢ Leaky ReLU: A variant of ReLU that allows a small negative slope for negative values, addressing the "dying ReLU" problem.
  - ➢ Softmax: Used in the output layer for multi-class classification, providing a probability distribution over multiple classes.
- ❖ Issues without Activation Functions:
  - ➢ Without activation functions, the entire neural network would collapse into a linear model, making it incapable of learning complex patterns and hierarchical representations.
  - ➢ The model's expressiveness and capacity to model intricate relationships in the data would be severely limited.

Answer 2 DL-> iNeuron_Test/DL_2.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)

Answer-3 DL-> iNeuron_Test/DL_3.ipynb at main · Pranjal-sharma-SDE/iNeuron_Test (github.com)