|  |
| --- |
|  |
|  |
|  |
|  |
|  |

A logo of a company

Description automatically generatedA close-up of a logo

Description automatically generated

**VIJITH S B**

**22CSR239**

**Assignment 1:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Year** | **Brand** | **Model** | **RAM (GB)** | **Storage (GB)** | **Processor** | **GPU** | **Battery Life (hrs)** | **Price (INR)** |
| 2019 | Dell | XPS 13 | 16 | 512 | Intel i7 | Intel Iris | 12 | 150000 |
| 2020 | Apple | Air m1 | 32 | 1024 | Apple M1 | Apple GPU | 20 | 220000 |
| 2018 | Lenovo | ThinkPad | 16 | 256 | Intel i5 | Intel UHD | 15 | 130000 |
| 2021 | HP | Spectre | 16 | 512 | Intel i7 | NVIDIA MX150 | 10 | 140000 |
| 2017 | Asus | Rog | 32 | 1024 | Ryzen 5 | RTX 4050 | 8 | 180000 |

* **Feature**:

Individual measurable properties (e.g., Year, Brand, Model, RAM, Storage, Processor, GPU, Battery Life) used as inputs to the model.

* **Label**:

The output variable that the model aims to predict (e.g., Price).

* **Outlier**:

A data point that deviates significantly from the rest of the data (e.g., if there was a laptop with a price of 500000 in this dataset).

* **Training Data**:

The dataset used to train the machine learning model, as given in the example above.

* **Test Data**:

A subset of the dataset used to assess the performance of the trained model.

* **Model**:

A mathematical representation of a real-world process, trained on data to make predictions.

* **Validation Data**:

A subset of the dataset used to tune the model's hyperparameters and prevent overfitting.

* **Hyperparameter**:

Parameters set before the learning process begins, controlling the behavior of the training algorithm.

* **Epoch**:

One complete pass through the entire training dataset.

* **Loss Function**:

A measure of how well the model's predictions match the actual data.

* **Learning Rate**:

A hyperparameter that controls how much the model's weights are adjusted with respect to the loss gradient.

* **Overfitting**:

When a model learns the training data too well, capturing noise and details that do not generalize to new data.

* **Underfitting**:

When a model is too simple to capture the underlying patterns in the data.

* **Regularization**:

A technique used to prevent overfitting by adding a penalty for larger coefficients to the loss function.

* **Cross-Validation**:

A technique for evaluating the performance of a model by dividing the data into multiple folds.

* **Feature Engineering**:

Creating new features or modifying existing ones to improve the performance of the model.

* **Dimensionality Reduction**:

Reducing the number of features in the dataset while retaining important information.

* **Bias**:

Error introduced by approximating a real-world problem by a simplified model.

* **Variance**:

The model's sensitivity to fluctuations in the training data.