CS 419

Lecture 1: Introduction to Machine Learning

January 13, 2023

Lecturer: Abir De *Scribe: Priyansh, Shantanu, Vishruth*

Contents

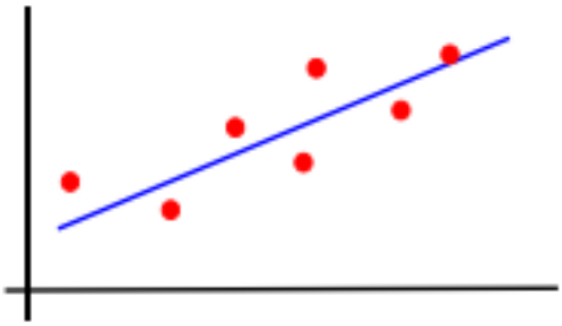
1 Machine Learning in general

1.1 Supervised Learning

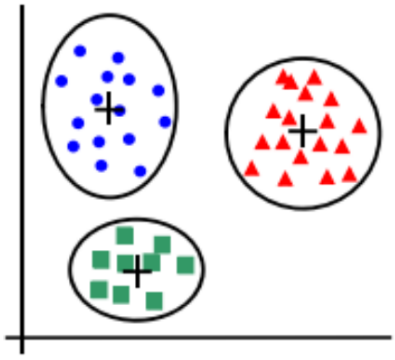
- Trained using labelled data
- Input data is provided to the model along with the output
- Goal is to train the model so that it can predict the output for new data
- Can be categorized in Classification & Regression problems
- Includes various algorithms such as Linear & Logistic Regression, Support Vector Machine, Decision tree etc.

1.2 Unsupervised Learning

- Trained using unlabeled data
- Only input data is provided to the model (no output provided)
- The goal is to find hidden patterns and useful insights from the unknown dataset
- Can be classified into Clustering and Association problems
- Includes various algorithms such as Clustering, K-Nearest Neighbours and Apriori algorithm



Regression



Clustering

1.3 Applications & examples

Task: Suppose we had a fruit basket & the task is to arrange the same type of fruits (Apples, Bananas, Cherries, Grapes) in one place.

Case 1:

- We already know Shape & Colour
- Train Data: Pre-classified data
- Goal: Learn from the pre-classified data and predict on new unclassified fruits
- This type of learning is called **Supervised Learning**

Case 2:

- We're seeing the fruits for the first time & hence know nothing about them
- To arrange the fruits, we can consider various characteristics of a fruit
- We can use colour and size to group them:
  - Red colour and big size: Apple
  - Red colour and small size: Cheery
  - Green colour and big Size: Banana
  - Green colour and small Size: Grape
- This type of learning is **Unsupervised Learning**

2 Canonical Learning Problems

2.1 Regression Supervised

Regression supervised is a type of machine learning where the goal is to predict a continuous outcome variable based on a set of input features. In this problem, the model is provided with labelled training data, where the input data is paired with a corresponding output value. The model learns to map the input data to the output values by recognizing patterns and identifying relationships between the inputs and outputs.

2.2 Classification Supervised

Classification supervised is a type of machine learning where the goal is to predict a categorical outcome variable based on a set of input features. Similar to regression supervised, the model is provided with labelled training data, where the input data is paired with a corresponding output label. The model learns to map the input data to the correct output label by recognizing patterns and identifying relationships between the inputs and labels.

2.3 Unsupervised Modelling of Data

Unsupervised modelling of data is a type of machine learning where the model is not provided with labelled data and is left to discover hidden patterns or structures in the input data on its own. The goal of this type of problem is to uncover underlying structures or relationships in the data rather than predicting a specific outcome variable. Examples of unsupervised learning include clustering and dimensionality reduction.

It is important to note that in supervised learning if the model is presented with input data that is quite different from the dataset it was trained on, it may give random or unreliable output, as it has not been exposed to that range of inputs. Therefore, it is essential to split the dataset appropriately and to use techniques such as cross-validation to evaluate the model's performance on unseen data.

Additionally, Reinforcement Learning is another type of machine learning in which the agent interacts with an environment to learn the best sequence of actions that maximizes the overall rewards.

3 What is Data?

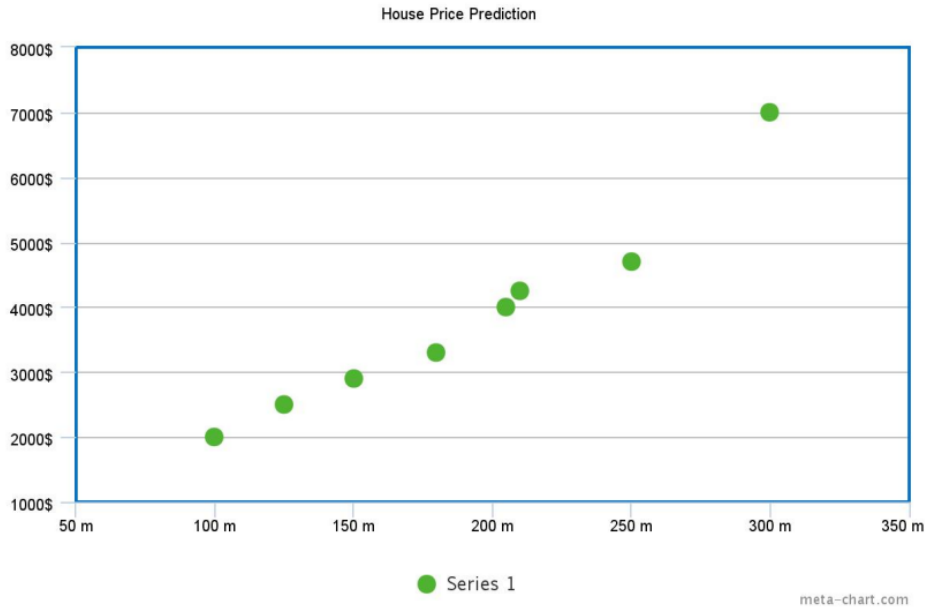
Data is the information about the problem we are solving using ML in quantized form. This data can be from any source, some examples are Prices of stock & stock indexes such as BSE or Nifty; Prices of houses, area & size of the house; Temperature of a place, latitude, longitude & time of year. The objective of ML is to predict or classify something using the given data. Hence, one or more parameters of the data must also represent the output of our program.

3.1 Noise in Data

Data in real-life problems are generally collected through surveys which may have random human errors. Hence most methods we will be using deals with expectations as they minimize the effect of error in our predictions. It is better to find outliers and clean data in the first step. This is known as **Data Cleansing**.

3.2 Example Dataset

Consider the variation of the cost of the house with the area of the house. We will use this data to find a pattern or curve for predicting the price for any value of an area.



House Purchase Data

4 How to predict?

4.1 Fitting a curve

Curve fitting is the process of constructing a curve or mathematical function that has the best fit to a series of data points, possibly subject to constraints. Thus, we need a criterion to compare two curves on a dataset. We describe an error function F(f, D) which takes a curve f and dataset D as input and returns a real number. Error function must be such that it can capture how worse is the model's prediction from the actual value.

4.2 Error Measurement

Consider the example below, where we have two curves on our dataset defined by blue (f_b) and red (f_r) lines, respectively. To find which is a better fit, use error measurement functions to find out how bad our model is. For this, we introduce the Loss Function and Cost Function. Loss Function is to capture the difference between the actual and predicted values for a single record, whereas Cost Functions aggregate the difference for the entire training dataset. The following is a problem of Linear Regression for which the most commonly used Loss Function is the Least Squared Error, and its cost function is called Mean Squared Error (MSE).

House Purchase Data Curve Fit

4.3 Minimizing Error

Assume we have a fixed loss function L. We should construct our model to have minimum error for this loss function. Let f_k be the curve we use to fit the data onto. Our aim would be to minimize the error E = $\sum_{i=1}^{N} L(y_i - f_k(x_i))$. For different f_k we would have different values of E. Thus, we would want to find that f_k, which leads to minimum E. So, for a given L, we have:

$$E = \min_{f_k} \sum_{i=1}^{N} L(y_i - f_k(x_i))$$

The choice of L is problem dependent. Above mentioned Least Squared Error is a type of loss function. It's one of the most widely used Loss Function & based on the method of Least Squares.

5 Method of Least Squares

Suppose we want to fit a function f through points (x_i, y_i), with minimum error. What kind of error should we try to minimize? We have a few options:

- $\sum |f(x_i) - y_i|$
- $\sum (f(x_i) - y_i)^2$
- $\max |f(x_i) - y_i|$

We want our error function to be continuous and differentiable. Further, by choosing squares, large errors get amplified more. Therefore we choose sum of squares to be our error function.

The sum of squares can also be thought of as a maximum likelihood estimator in a natural way: Finding the minimum of $\sum (f(x_i) - y_i)^2$ is the same as finding the maximum of

$$\prod e^{-(f(x_i)-y_i)^2}$$

That is, if we assume the errors $f(x_i) - y_i$ form a normal distribution around 0 (which is a practical assumption), then the method of least squares gives the same result as the maximum likelihood estimator.

6 Simulating Distributions using Uniform Samples

Suppose we have a way to draw samples from a uniform distribution in $[0, 1]$ (say using random.uniform() in Python). Is there a way to simulate random numbers in any other distribution? This can be done if the distribution is "good enough".

Let F be the cumulative distribution function of our required distribution. Assume first that F is strictly increasing. Note that $F(x) \in [0, 1]$ for all x. Note that for fixed $b \in [0, 1]$, and x drawn from a uniform distribution, we have $\mathbf{Pr}(x \leq b) = b$. Therefore $\mathbf{Pr}(x \leq F(y)) = F(y)$ for any y. Since F is strictly increasing, it is invertible, so $x \leq F(y)$ iff $F^{-1}(x) \leq y$. Therefore $\mathbf{Pr}(F^{-1}(x) \leq y) = F(y)$ for any fixed $y \in \mathbb{R}$. $F^{-1}(x)$ hence has the same cumulative distribution function F.

Hence the way to simulate the distribution is: Draw x uniformly in $[0, 1]$ and return $F^{-1}(x)$. This works as long as F is strictly increasing, and this is guaranteed as long as $F' \neq 0$ everywhere, i.e., the distribution takes all values and is not bounded.