What is Machine Learning?

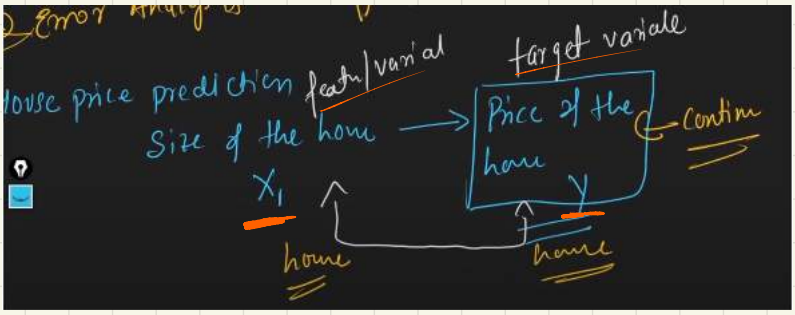
Computer programs that uses algorithms to analyze data and make intelligent predictions based on the data without being explicitly programmed.

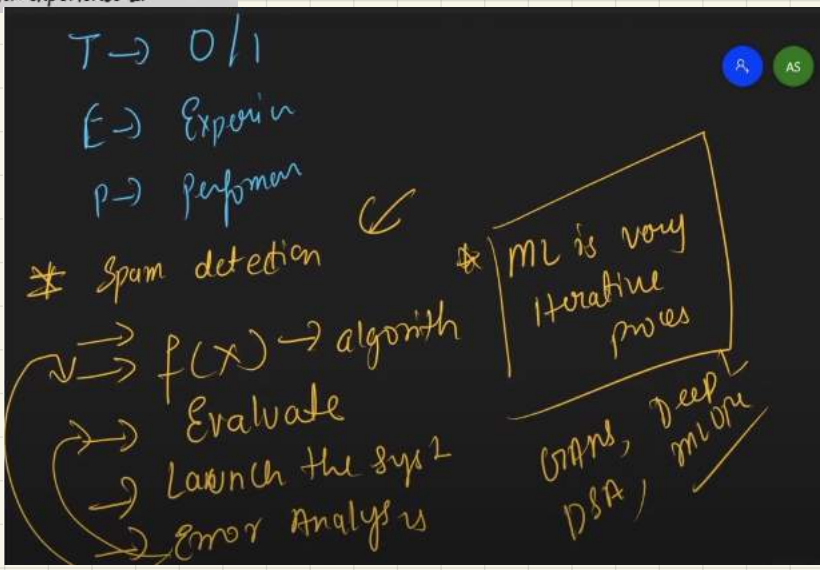
Slightly more formal definition:-

- Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.

  Arthur Samuel, 1959

- A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.





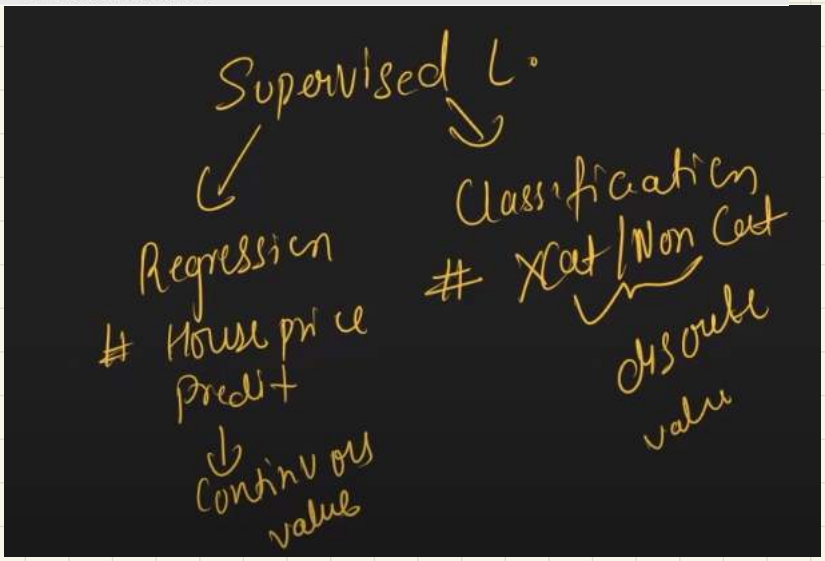
Supervised Learning Problems

Regression:- for continuous data

- A person's height: could be any value (within the range of human heights), not just certain fixed heights.

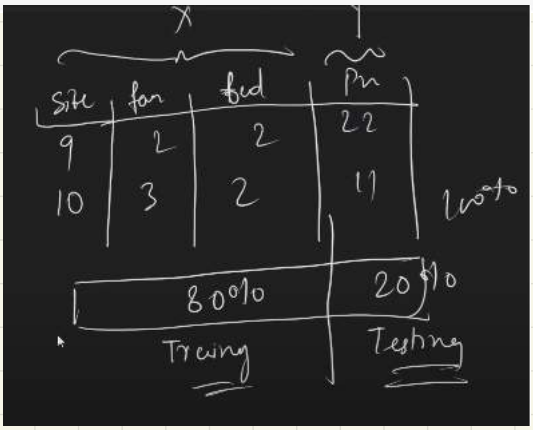
Classification:- for categorical data

- Discrete data is counted.



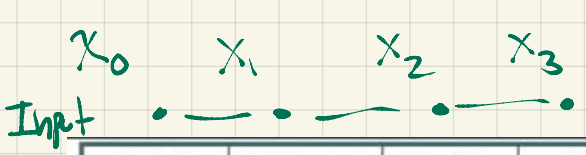
Dividing Our Data

We divide our data into two sets, Training and Testing sets the reason is very simple, we want to test the model after we are done, so we make a test set for evaluating our model.



Supervised Vs. Unsupervised

- Input Ft. x & Output y j
- there is some kind of relationship between x_n & y_n
- x ~> Independt Ft. & y is dependt Ft.



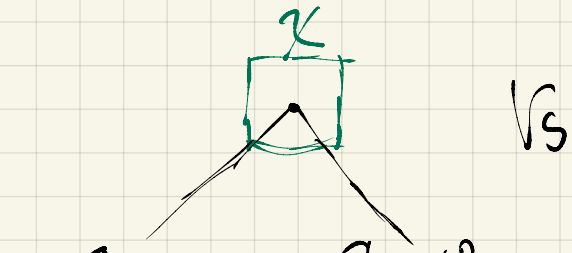
y target Var

Output Val

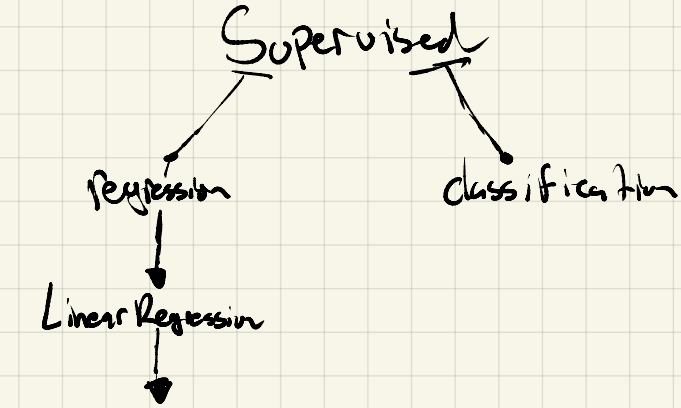
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Dependent Feature

Independent Feature

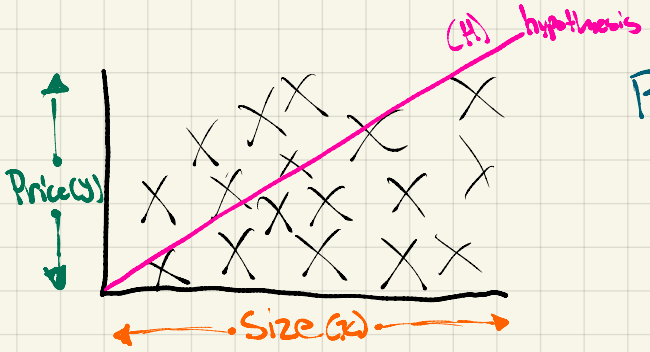


Linear Regression



y ~ in Con. Value

#priceOfHouse ~ Cont.



Predicting $ of House [based on Size]

X → Size

y → Price

Hypothesis Func.

$\theta_0 \cdot x_0 \dots \theta_n \cdot x_n$

$\theta_0 + \theta_1 + \theta_2 + \dots + \theta_n$

$x_0 + x_1 + x_2 \dots + x_n$

Features

Biased term
y-int
Base Case [0]?

$\theta_0 \rightsquigarrow$ Implements x's ft.

$\theta_0 = 2$

$\theta_1 = 7$

$\theta_2 = 3$

Weighted features.

$(\theta_0 \cdot x_0) + (\theta_1 \cdot x_1) + (\theta_2 \cdot x_2)$

$2 \quad 7 \cdot ft_1 \quad 3 \cdot ft_2 = y$

- Best ft. Weight == Best Prediction
- θ_0 = Implements x's ft.

- $x \rightsquigarrow x_n \rightsquigarrow x_{nth}$
- Only learn how to get θ
- If θ is bad then

the Func. Hypothesis is bad

$\therefore$

$f(x) = \theta_0 \cdot x_0 \dots \theta_n \cdot x_n$

Vectorization

memo?

The difference between vectorization and loop unrolling: Consider the following very simple loop that adds the elements of two arrays and stores the results to a third array.

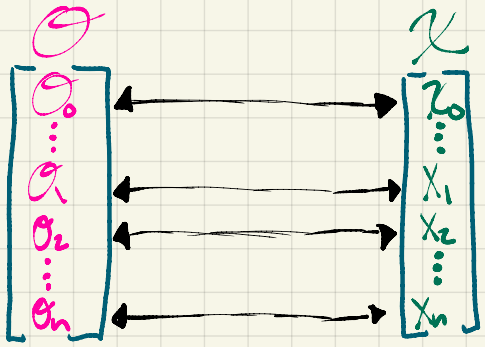
```
for (int i=0; i<16; ++i)
    C[i] = A[i] + B[i];
```

Unrolling this loop would transform it into something like this:

```
for (int i=0; i<16; i+=4) {
    C[i]   = A[i]   + B[i];
    C[i+1] = A[i+1] + B[i+1];
    C[i+2] = A[i+2] + B[i+2];
    C[i+3] = A[i+3] + B[i+3];
}
```

Vectorizing it, on the other hand, produces something like this:

```
for (int i=0; i<16; i+=4)
    addFourThingsAtOnceAndStoreResult(&C[i], &A[i], &B[i]);
```

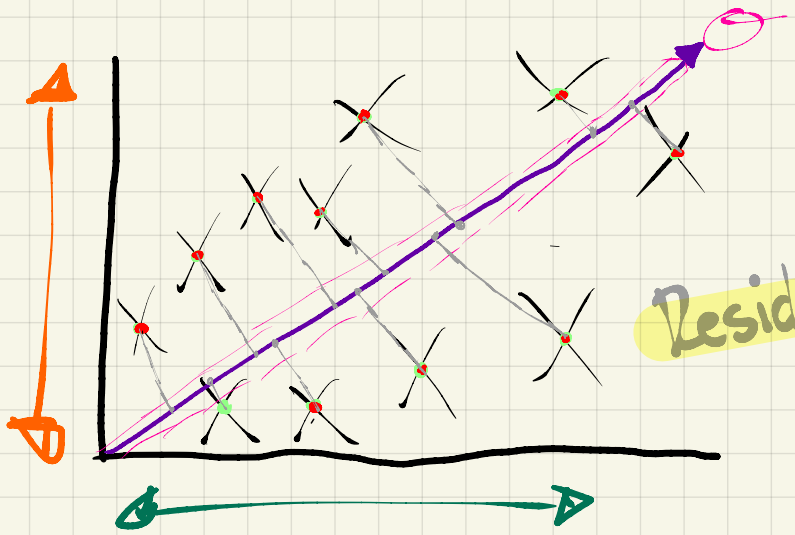


$y = \theta \cdot x$

Vectorized form

Code : np.dot(θ, x)

Cost function y? Math



Actual Val = •

Predict ==↗

Residuals Predict − Actual Val

$$\hat{y} = f(x) = \theta \cdot x$$

$$\text{Func}(\theta) = \frac{1}{m} + \sum_{i=1}^{m} (\hat{y} - y)$$

$\frac{1}{m}$ ← Actual Val / Points

- Higher Function Worse Model
- Less Function worse Model
- $\hat{y}$ → Predicted Val.

$$\text{Func}(\theta) = \frac{1}{m} + \sum_{i=1}^{m} (\theta^{T} x^{(i)} - y^{(i)})$$

Func(θ) ← BaseCase; $\frac{1}{m}$ ← idk; θ^T → Transpose; $x^{(i)}$ (H) → Predict; $y^{(i)}$ → Actual Val

Transpose of a Matrix

$$\begin{bmatrix} 3 & 4 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 4 & 1 \end{bmatrix}$$

Input Matrix — Transpose Matrix

$$\text{if}\left(\frac{1}{m} + \sum_{i=1}^{m} (P-A)^2\right) == \text{RME}$$

$$\text{if}\left(\sqrt{\frac{1}{m} + \sum_{i=1}^{m} (P-A)^2}\right)$$

⋮

RMSE

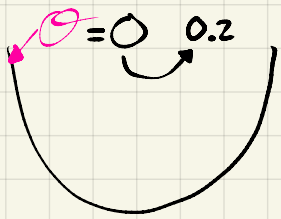
Root · mean · error · Sqrt()

Optimal Theta (θ)

$\frac{\partial}{\partial}$ = Partial Der'

- Gradient Desent Algorithm
- Optimized - memo?
- Partial Derivitives

$\frac{f(x)}{f'(x)}$ $f(\theta)$



Partial derivatives of that function are:

with respect to x

$\frac{\partial f}{\partial x}\ x^4 y^3 = 4x^3 y^3$

We take y as a constant. Then we obtain the derivative of x^4.

with respect to y

$\frac{\partial f}{\partial y}\ x^4 y^3 = x^4 3y^2$

We take x as a constant. Then we obtain the derivative of y^3.

*Note: a ∂ , del symbol is mostly used for multivariable functions, aka when we find partial derivatives and not just d which is used when we find a "usual" derivative.

or a slightly another example:

Let's take a function:

$f(x, y) = x^3 y$

It has a form of:

The partial derivatives with respect to x and to y would be:

$\frac{\partial}{\partial x}(x^3 y) = 3x^2 y$

$\frac{\partial}{\partial y}(x^3 y) = x^3$

$$\frac{d}{\partial} J(\theta) = \frac{\partial}{m} + \sum_{i=1}^{m} (\hat{y} - y)^2$$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial} J(\theta)$$

$$\theta_0$$

$$\nabla J(\theta) = \begin{bmatrix} \frac{\partial}{\partial} J(\theta_0) \\ \frac{\partial}{\partial} J(\theta_1) \\ \frac{\partial}{\partial} J(\theta_2) \end{bmatrix}$$

$$\theta_j := \theta - \alpha \frac{\partial}{m} \cdot X^T (X\theta - y)$$