Logistic Regression Part 1

13 July 2023 01:24 AM

Linear Regression

Find the best fit Hyper-plane

Simple Linear Regression (SLR)

Optimization Equation of SLR

$$\rightarrow m^*, c^* = \underset{m,c}{\arg\min} \left\{ \underbrace{\frac{\Sigma}{n}\left[y_i - (mx_i + c)\right]^2}_{f(m,c)} \right\}$$

Multiple Linear Regression (MLR)

Optimization Equation of MLR

$$\rightarrow \vec{w}^*, w_0^* = \underset{\vec{w}, w_0}{\arg\min} \left\{ \frac{\Sigma}{n}\left[y_i - (w^T x_i + c)\right]^2 \right\}$$

$\vec{w} = \begin{bmatrix} \ \end{bmatrix}$ column vector $\quad w_0 \rightarrow$ Scalar

These optimization equation can be solved using Gradient Descent

Gradient Descent for SLR

→ Randomly initialise m_i & c_i

→ Compute m_{i+1} & c_{i+1} such that they are closer to m^* & c^* respectively.

$$\left.\begin{aligned} m_{i+1} &= m_i - \eta\left[\frac{\partial f(m,c)}{\partial m}\right]_{m_i} \\ c_{i+1} &= c_i - \underbrace{\eta}_{\text{Hyperparameter of Gradient Descent}}\left[\frac{\partial f(m,c)}{\partial m}\right]_{c_i} \end{aligned}\right\}$$

Update Equation of Gradient Descent

Gradient Descent for MLR

→ Randomly initialise $\vec{w}_i$ & $(w_0)_i$

→ Compute $\vec{w}_{i+1}$, & $(w_0)_{i+1}$ such that they are closer $\vec{w}^*$ and w_0^* respectively

$$\left\{\begin{aligned} \vec{w}_{i+1} &= \vec{w}_i - \eta\left[\frac{\partial f(\vec{w}, w_0)}{\partial \vec{w}}\right]_{\vec{w}_i} \\ (w_0)_{i+1} &= (w_0)_i - \eta\left[\frac{\partial f(\vec{w}, w_0)}{\partial w_0}\right]_{(w_0)_i} \end{aligned}\right.$$

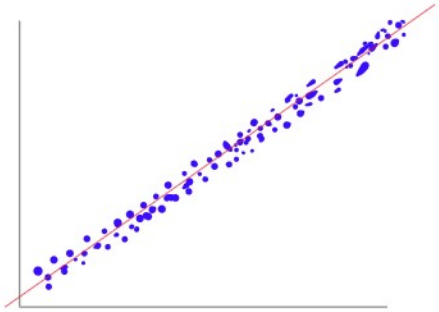
Repeat step 2 untill convergence

Linear Regression

Type: Supervised Learning

Task: Regression

How; By finding a $\underbrace{\text{line}}_{\substack{m,c \\ \text{model}}}$ that best fits the $\underbrace{\text{Data}}_{(D_n)}$



⇒ Best fit line = Minimizing the error done by the line

$$\rightarrow D_n = \left\{ (x_i, y_i)_{i=1}^n \;\middle|\; x_i \in R^{d-1}, y_i \in R \right\}$$

Logistic Regression

Type: Supervised Learning

Task: Classification (Binary Classification)

How; By finding a line that best separates the positives from negatives in the given data

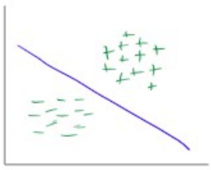
K-NN classifier, Naïve Bayes Classifier, Decision Tree Classifier

↓

They can perform both Binary classification as well as multi-class classification.

On the other hand, Logistic Regression by default can only perform Binary Classification

In order to perform multi-class classification using logistic Regression, some modifications are required.



$$\rightarrow D_n = \left\{ (x_i, y_i)_{i=1}^n \;\middle|\; x_i \in R^{d-1}, y_i \in \{-1, +1\} \right\}$$

$$m^*, c^* = \underset{m,c}{\arg\max} \left\{ \text{correctly classified Points} \right\}$$

→ Best Separator Line = Maximizing the number of correctly classified points.

How?

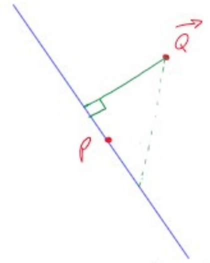
Tasks:

1.) Mathematical correct representation of "Correctly Classified Points"

2.) Convert maximization to minimization → Then Solve using Gradient Descent.

Mathematical Foundations

Q How to compute distance of a point from a line?



$y = w^T x + w_0$

distance of point P from line = 0

when talking about the distance of Point Q from line, we need the shortest distance.

* Also the point and the line should have same number of dimensions for maths/formulas to work out.

$$\vec{Q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_d \end{bmatrix}$$

$$\boxed{dist_o = \frac{w_1 q_1 + w_2 q_2 + \dots + w_d q_d + w_0}{|w|}}$$

Length of a vector

$$\boxed{dist = \frac{\sum_{i=1}^{d} w_i q_i + w_0}{\sqrt{\sum_{i=1}^{d} w_i^2}}}$$

Norm of a Line / PLANE / Hyper plane

General Equation of a Hyper-plane in

$2D \rightarrow \boxed{w_1 x_1 + w_2 x_2 + w_0} = 0$

$3D \rightarrow \boxed{w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0} = 0$

$\vdots$

$dD \rightarrow \boxed{w_1 x_1 + w_2 x_2 + \dots + w_d x_d + w_0} = 0$

$$\vec{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ w_0 \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ 1 \end{bmatrix}$$

$$\boxed{\vec{w} \cdot \vec{x} = 0}$$

1. $\boxed{\sum_{i=0}^{d} w_i x_i = 0}$

2. $\boxed{w^T x = 0}$

3. $\boxed{|\vec{w}||\vec{x}| \cos\theta_{\vec{w},\vec{x}} = 0}$

The dot product of $\vec{w}, \vec{x} = 0$ when the angle between the vectors is 90.

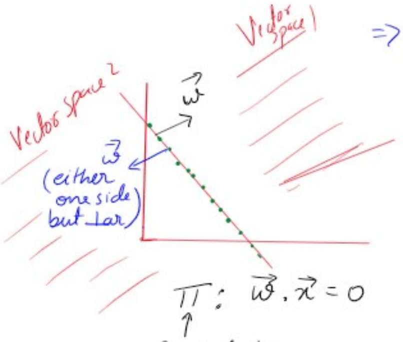
Using (3) $\quad \vec{w} \cdot \vec{x} = |\vec{w}|\,|\vec{x}| \cos\theta_{\vec{w},\vec{x}}$

$\theta_{\vec{w},\vec{x}} = 90$

$\Rightarrow \vec{w} \cdot \vec{x} = |\vec{w}|\,|\vec{x}| \cos 90 \qquad \{\cos 90 = \sin 0 = 0\}$

$\Rightarrow \vec{w} \cdot \vec{x} = 0$

$\Rightarrow \vec{w}$ & $\vec{x}$ are perpendicular



$\Pi : \vec{w} \cdot \vec{x} = 0$

Symbol to represent a Hyperplane

Observation

↳ $\vec{w}$ is going to be perpendicular / orthogonal to Π

↳ $\vec{w}$ is the NORM of a Hyperplane.

↳ $\vec{w}$ helps in identifying the Direction in which the hyperplane is facing

finding the distance between Point $\vec{P}$ and Π {i.e. Hyperplane}

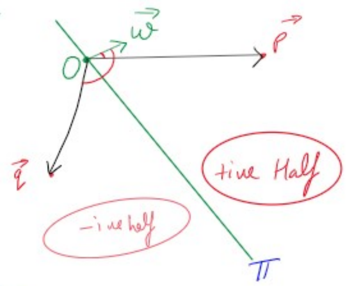
$$\boxed{dist = \frac{\vec{w} \cdot \vec{P}}{|\vec{w}|}}$$

$$\boxed{dist_\circ = \frac{w_1 P_1 + w_2 P_2 + \dots + w_d P_d + w_0}{|\vec{w}|}}$$

$$dist = \frac{\cancel{|\vec{w}|}\,|\vec{P}| \cos\theta_{\vec{w},\vec{P}}}{\cancel{|\vec{w}|}}$$

$$\boxed{dist = \frac{\sum_{i=1}^{d} w_i P_i + w_0}{\sqrt{\sum_{i=1}^{d} w_i^2}}}$$

$$\Rightarrow \boxed{dist = |\vec{P}| \cos\theta_{\vec{w},\vec{P}}}$$

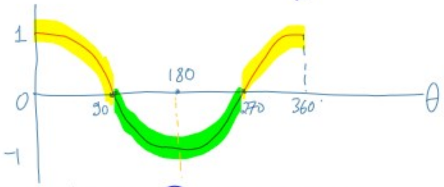


→ $\vec{w}$ is the norm of Π and $\vec{w} \perp \Pi$

→ $\vec{w}$ divides the entire d-dimensional vector spaces into two halves (half spaces), one on the upper side and one on lower.

* +ive Half → Point lies on the side where the hyperplane is facing
-ive Half → Point lies on the opposite side to where the hyperplane is facing.

To find where do point $\vec{P}$ & $\vec{Q}$ lie on which side of hyperplane.



→ $0 \le \theta \le 90$ (or) $270 \le \theta \le 360$
$\Rightarrow \cos\theta$ is +ive

→ $90° < \theta < 270° \Rightarrow \cos\theta$ is -ive

Compute → $dist(\vec{P}, \Pi) = \frac{\vec{w} \cdot \vec{P}}{|\vec{w}|} = \frac{\cancel{|\vec{w}|}\,|\vec{P}| \cos\theta_{\vec{w},\vec{P}}}{\cancel{|\vec{w}|}} = \underbrace{|\vec{P}|}_{\text{always +ive}} \underbrace{\cos\theta_{\vec{w},\vec{P}}}_{\text{+ive}} = \text{+ive}$

$dist(\vec{Q}, \Pi) = \frac{\vec{w} \cdot \vec{Q}}{|\vec{w}|} = \underbrace{|\vec{Q}|}_{\text{always +ive}} \underbrace{\cos\theta_{\vec{w},\vec{Q}}}_{\text{-ive}} = \text{-ive}.$

* dist. can be
→ +ive → if $\vec{P}$ lies in the same of direction of Π
→ -ive → if $\vec{P}$ lies on the opposite direction of Π

Task → Find a $\underbrace{\text{hyperplane}}_{\vec{w},\, w_0}$ which $\underbrace{\text{best separates}}_{\text{maximize the no. of correctly classified points.}}$ +ives from −ives.

$$\boxed{act_i \times pred_i}$$

Case A	Case B	Case C	Case D
act = +1	act = +1	act = −1	act = −1
pred = +ive	pred = −ive	pred = −ive	pred = +ive
↓ correct	↓ misclass.	↓ correct	↓ misclass.

mathematically,

$$\vec{w}^*, w_0^* = \underset{\vec{w},\, w_0}{\arg\max} \left\{ act_i \times pred_i \right\}$$

$$\boxed{\vec{w}^*, w_0^* = \underset{\vec{w},\, w_0}{\arg\max} \left\{ \underbrace{y_i \times (w^T x_i + w_0)}_{\text{Continuous \& Concave Cost } f^n} \right\}}$$

optimization equation of Logistic Regression

Continuous & Concave Cost f^n } aka $\boxed{\text{signed distance}}$ → The sign helps in determining whether the point is correctly classified or not.

∵ it is an optimization equation → we use Gradient Ascent

$$\boxed{\vec{w}_{i+1} = \vec{w}_i + \eta \left[\frac{\partial f(\vec{w}, w_0)}{\partial w} \right]_{w_i}}$$

∵ the cost f^n will tell whether the classification is correct or not for each datapoint, we perform summation :

$$\boxed{\vec{w}^*, w_0^* = \underset{\vec{w},\, w_0}{\arg\max} \left\{ \sum_{i=1}^{n} \left[y_i \times (w^T x_i + w_0) \right] \right\}}$$

Eg.

y-train	Line-1	Act × pred₁	Line-2	Act × pred₂
+1	−1	−1	+1	+1
−1	−1	+1	−1	+1
+1	+1	+1	+1	+1
+1	+1	+1	+1	+1
−1	−1	+1	−1	+1
−1	+1	−1	−1	+1
		$\Sigma = 2$		$\Sigma = 6$

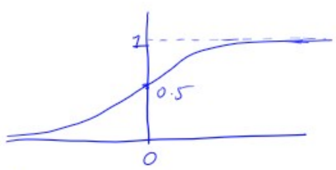
{ If signed dist → +ive → Correct Class.
→ −ive → Incorrect Class. }

The above optimization equation can't be used incase you have outliers.

So in order to tackle the problem of signed distance ($-\infty \le dist \le +\infty$), we use $\underbrace{\text{Sigmoid function}}_{\text{Helps in reducing the impact of outliers.}}$ ([0, 1])

Benefits of Sigmoid function ($\sigma(x)$) :

1.) Reduce outlier impact on the model



2.) Range $0 \le \sigma(x) \le 1$

3.) Can be interpreted as probability $\sigma(x) = \dfrac{1}{1+e^{-x}} = \dfrac{1}{1+e^{-[y_i \times (w^T x_i + w_0)]}} = \dfrac{1}{1+\exp\left\{-[y_i \times (w^T x_i + w_0)]\right\}}$