# ML Supervised Learning 5

by ambedkar@IISc

- Probabilistic view of linear regression
- Logistic regression
- Hyperplane based classifiers and perceptron

# Rewind

# What we learning so far?

- Bayes Decision Theory
- Some foundational aspects of Machine learning and Generalizing capacity
- Linear Regression
- Regularization (very important)
- Gradient Descent

# Probabilistic View of Linear Regression

# Maximum Likelihood Estimation

Let $X = x_1, x_2, \ldots, x_N$, where $x_n \in \mathbb{R}^D$ be some data that is generated from $x_n \sim P(x|\theta)$

- **Recall**: In the statistical approach to machine learning, we assume that there is an underlying probability distribution from which the data is sampled.
- Hence $\theta$ denotes the parameters of the distribution.
- For example $x_n \sim \mathcal{N}(x|\mu, \sigma)$. That is $\theta = (\mu, \sigma)$.

**Assumption**: The data in X is generated i.i.d. (independent and identically distributed). This is very important assumption and we see this very often.

**Aim**: Learn $\theta$ given the data $X = x_1, x_2, \ldots, x_N$.

# Diversion: Some Probability

- We say two random variables $X$, $Y$ are identical that means that their probability distributions are the same
  - If two Gaussian random variables are same only if their means and variance (covariance matrices) are same.
- We say two random variables $X$, $Y$ are independent if

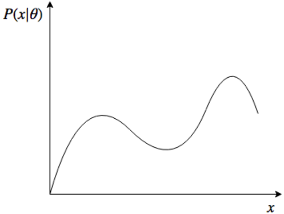
$$P(X, Y) = P(X)P(Y)$$

# Maximum Likelihood Estimation (contd...)

- Given $X = x_1, x_2, \ldots, x_N$, and $x_n \sim P(x|\theta)$
  - Learn $P$ so that likelihood of $x_1, x_2, \ldots, x_N$ are sampled from $P$ is maximum.
  - Equivalently learn or estimate $\theta$ so that likelihood of $x_1, x_2, \ldots, x_N$ are sampled from $P$ is maximum.
- By the iid assumption

$$P(X|\theta) = P(x_1, x_2, \ldots, x_N|\theta) = \prod_{n=1}^{N} P(x_n|\theta)$$

- $P(X|\theta)$ is the likelihood.



# Maximum Likelihood Estimation (contd...)

How do we estimate $\theta$ given the data $X$.

$$\Downarrow$$

Find value of $\theta$ that makes observed data most probable.

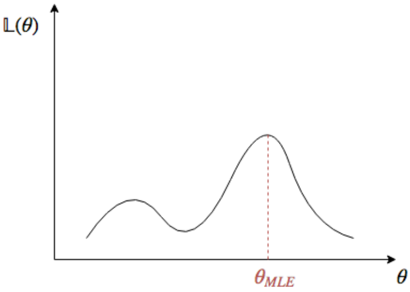
$$\Downarrow$$

Find $\theta$ that maximizes likelihood function

$$\mathcal{L} = \log P(X|\theta) = \sum_{n=1}^{N} \log P(x_n|\theta)$$

# Maximum Likelihood Estimation (contd...)

$$\theta^*_{\mathsf{MLE}} = \arg\max_{\theta} \mathcal{L}(\theta) = \arg\max_{\theta} \sum_{n=1}^{N} \log P(x_n|\theta)$$



**Example**:

Suppose $X_n$ is a binary random variable. Suppose it follows Bernoulli distribution
i.e. $P(x|\theta) = \theta^x (1-\theta)^{1-x}$

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \log P(x_n|\theta) = \sum_{n=1}^{N} x_n \log \theta + (1-x_n)\log(1-\theta)$$

$$\begin{aligned}\frac{\partial \mathcal{L}(\theta)}{\partial \theta} &= \frac{1}{\theta}\sum_{n=1}^{N} x_n + \frac{1}{1-\theta}\sum_{n=1}^{N}(1-x_n) \\ &= \frac{1}{\theta}\sum_{n=1}^{N} x_n + \frac{1}{1-\theta}\left(N - \sum_{n=1}^{N} x_n\right)\end{aligned}$$

$$\implies \theta^*_{MLE} = \frac{\sum_{n=1}^{N} x_n}{N}$$

[ In a coin tossing experiment, it is just a fraction of heads ]

# Maximum a Posteriori Estimate

We will have a prior on parameter $\theta$ i.e. $P(\theta)$

Yes $\theta$ is no more a mere number, it is a Random Variable.

- One can have knowledge on $\theta$
- It acts as a regularizer (We will see later)

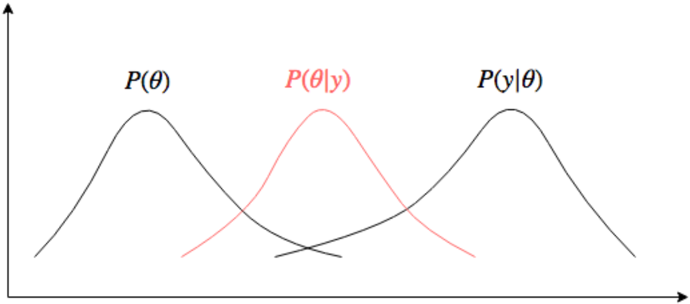
**Bayes Rule:**

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

$$\begin{aligned} P(\theta|X) &: \text{Posterior} \\ P(X|\theta) &: \text{Likelihood} \\ P(\theta) &: \text{Prior} \\ P(X) &: \text{Evidence} \end{aligned}$$

**Bayes Rule:**

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$



# Maximum a Posteriori Estimate (contd. . . )

$$
\begin{aligned}
\theta^*_{MAP} &= \arg\max_\theta P(\theta|x) \\
&= \arg\max_\theta \log P(x|\theta) + \log P(\theta) \\
&= \arg\max_\theta \sum_{n=1}^{N} \log P(x_n|\theta) + \log P(\theta)
\end{aligned}
$$

Note that when $P(\theta)$ is a uniform distribution, it reduces to MLE.

# Linear Regression : Probabilistic Setting

Each response is generated by a linear model + Gaussian noise

$$Y = W^\intercal X + E$$

That is, given N training samples $\{(x_n, y_n)_{n=1}^{N}\}$ i.i.d. $x_n \in \mathbb{R}^D$ and $y_n \in \mathbb{R}$

- $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$
- $y_n \sim \mathcal{N}(w^\intercal x_n, \sigma^2)$

$$\begin{aligned} \implies P(Y|X,W) &= \mathcal{N}(y|w^\intercal x, \sigma^2) \\ &= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y - w^\intercal x)^2}{2\sigma^2}\right) \end{aligned}$$

## Log Likelihood

$$
\begin{aligned}
\log \mathcal{L}(w) &= \log P(\mathcal{D}|w) = \log P(y|X, W) \\
&= \log \prod_{n=1}^{N} P(y_n|x_n, w) \\
&= \sum_{n=1}^{N} \log P(y_n|x_n, w) \\
&= \sum_{n=1}^{N} \left[ -\frac{1}{2} \log(2\pi\sigma^2) - \frac{(y_n - w^\intercal x_n)^2}{2\sigma^2} \right]
\end{aligned}
$$

## Linear Regression : ML Estimation (contd. . . )

$$
\begin{aligned}
w^*_{\mathsf{MLE}} &= \arg\max_w -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - w^\intercal x_n)^2 \\
&= \arg\min_w \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - w^\intercal x_n)^2
\end{aligned}
$$

i.e. ML Estimation in the case of Gaussian environment $\equiv$ Least square objective for regression

- Here we introduce prior on the parameter $w$. This will lead to regularization of model.
  - Remember we treat parameters as Random Variables in MAP.
- $\mathrm{P(w)} = \mathcal{N}(\mathrm{w}|\underbrace{0}_{\text{Mean}}, \underbrace{\lambda^{-1}I}_{\text{Variance}})$
- We have multivariate Gaussian

$$\mathcal{N}(x:\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}}\exp\Big(-\frac{1}{2}(x-\mu)^\intercal\Sigma^{-1}(X-\mu)\Big)$$
$$= \frac{1}{\sqrt{(2\pi)^{\frac{D}{2}}(\frac{1}{\lambda})^{\frac{D}{2}}}}\exp\Big(-\frac{\lambda}{2}w^\intercal w\Big)$$

- log posterior probability

$$\begin{aligned}\log(w|\mathcal{D}) &= \log \frac{P(\mathcal{D}|w)P(w)}{P(\mathcal{D})} \\ &= \log P(w) + \log P(w|\mathcal{D}) - \log P(\mathcal{D})\end{aligned}$$

- 

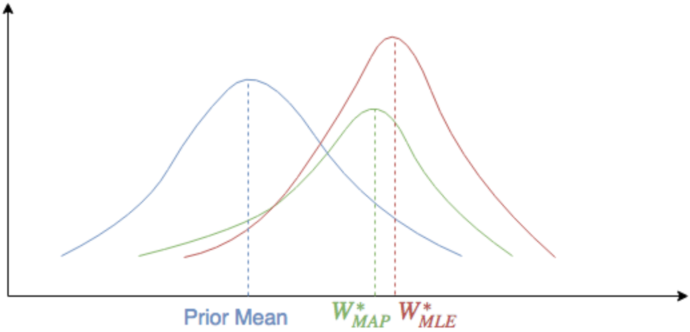
$$\begin{aligned}w^*_{\mathsf{MAP}} &= \arg\max_w \log P(w|\mathcal{D}) \\ &= \arg\max_w \left\{ \log P(w) + \log P(\mathcal{D}|w) + \log P(\mathcal{D}) \right\} \\ &= \arg\max_w \left\{ \log P(w) + \log P(\mathcal{D}|w) \right\}\end{aligned}$$

## Linear Regression : MAP Estimate (contd. . . )

$$
\begin{aligned}
W^*_{MAP} &= \arg\max_w \log P(w|\mathcal{D}) \\
&= \arg\max_w \Big[ -\frac{D}{2}\log 2\pi - \frac{\lambda}{2} w^T w \\
&\qquad + \sum_{n=1}^{N} \left( -\frac{1}{2}\log(2\pi\sigma^2) - \frac{(x_n - w^T x_n)^2}{2\sigma^2} \right) \Big] \\
&= \arg\max_w \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - w^T x_n)^2 + \frac{\lambda}{2} w^T w
\end{aligned}
$$

MAP estimate in the case of Gaussian environment $\equiv$ Least square objective with $L_2$ regularization.

# MLE vs MAP




MAP estimate shrinks the estimate of w towards the prior.

# Optimization is the Key

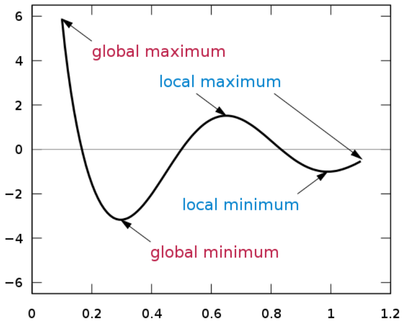
- Almost all problems in machine learning leads to optimization problems
- The following two factors decides the fate of any method:
  - What kind of optimization problem that we are led to
  - What are all optimization methods that are available to us
- There are several methods that are available for optimization, among these gradient descent methods are most popular

## Gradient Descent methods are Used in ....

- Linear Regression
- Logistic Regression
  - It is just classification, but instead of labels it gives us class probability
- Support Vector Machines
- Neural Networks
  - The backbone of neural networks is Back-propagation algorithm

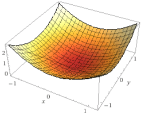
# Example of an objective

- Most often, we do not even have functional form of the objective.
  - Given $x$, we can only compute $f(x)$
  - Sometime this may involve a simulating a system
  - Computing each $f(x)$ can be time consuming

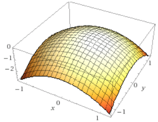



- This becomes even more difficult as $x$ is a $D$-dimensional vector and $D$ is very large

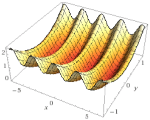
# Multivariate Functions



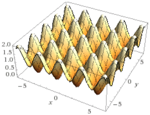
***(a)*** $f(x, y) = x^2 + y^2$



***(b)*** $f(x, y) = -x^2 + y^2$

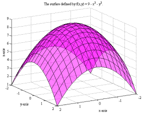


***(c)*** $f(x, y) = \cos^2(x) + y^2$

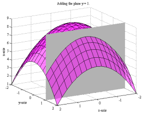


***(d)*** $f(x, y) = \cos^2(x) + \cos^2(y)$

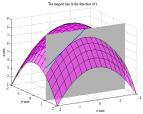
# Partial Derivatives




***(a)*** *Surface given by* $f(x, y) = 9 - \frac{x}{2} - \frac{y}{2}$

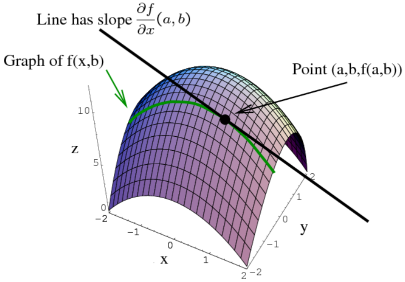



***(b)*** *Plane* $y = 1$




***(c)*** $f(x, 1) = 8 - \frac{x}{2}$ *denotes a curve, and* $f'(x) = -2x$ *denotes derivative (or slope) of that curve*

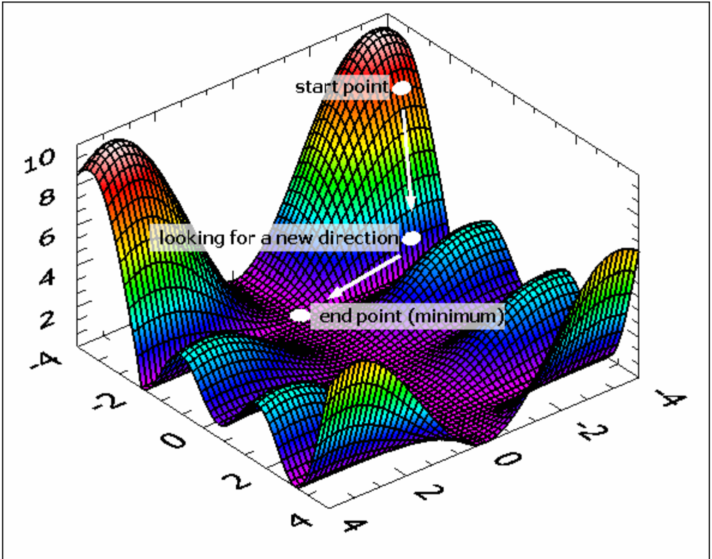
# Partial Derivatives (contd. . . )



# Idea of Gradient Descent Algorithm

- Start at some random point (of course final results will depend on this)
- Take steps based on the gradient vector of the current position till convergence
  - Gradient vector give us direction and rate of fastest increase any point
  - Any point $x$ if the gradient is nonzero, then the direction of gradient is the direction in which the function most quickly from $x$
  - The magnitude of gradient is the rate of increase in that direction

# Idea of Gradient Descent Algorithm[1]




[1] Credits for all the images in this sections goes to Michailidis and Maiden