1. Matrix Derivation:
$f : R^n \to R$
$\frac{\partial f(x)}{\partial x} = \begin{pmatrix} \partial f/x_1 \\ \partial f/x_2 \\ \dots \\ \partial f/x_n \end{pmatrix} \in R^n$
$f : R^{m\times n} \to R$
$\frac{\partial f(x)}{\partial x} = \begin{pmatrix} \partial f/x_{11} & \partial f/x_{12} & \dots & \partial f/x_{1n} \\ & \dots & \dots & \\ \partial f/x_{m1} & \partial f/x_{m2} & \dots & \partial f/x_{mn} \end{pmatrix} \in R^{m\times n}$

2. Convex:
$J(\alpha\theta_1 + (1-\alpha)\theta_2) \le \alpha J(\theta_1) + (1-\alpha)J(\theta_2)$
or in R, $J''(\theta) \ge 0, \forall\theta$

3. Linear Regression:
$\hat{\theta} = \arg\min_{\theta} \sum_i l(\theta^T\phi(x_i), y_i)$, l is l_2-norm^2
$\hat{\theta} = \arg\min_{\theta} \sum_i \|\theta^T\phi(x_i) - y_i\|_2^2$
Convex, let Derivitive $= 0 \Rightarrow \frac{\partial J(\theta)}{\partial\theta} = 0$
$\sum_i 2(\theta^T\phi(x_i) - y_i)\phi(x_i) = 0 \Rightarrow \Phi^T\Phi\theta = X^TY$
Cost time: $O(d^3 + d^2N)$ When not invertible: $n < d$
since $rk(\Phi^T\Phi) < rk(\Phi) < \min\{n, d\}$, when $n < d$ we have $rk(\Phi^T\Phi) = n < d$ then it is not-full rank.
Deal with Outliers:

   - remove detected Outliers
   - Robust Regression function: Huber loss Func. We write $\phi(x_i) - y_i$ as e, where δ is a hyperparameter MSE squares errors, outliers will distort the loss value significantly. Huber loss calculate l_1 norm on large values which gives more tolerance.

$$l_\delta(e) = \begin{cases} \frac{1}{2}e^2 & |e| \le \delta \\ \delta|e| - \frac{\delta^2}{2} & |e| \ge \delta \end{cases}$$

$$\frac{\partial l_\delta(e)}{\partial e} = \begin{cases} e & -\delta \le ele\delta \\ \delta & e > \delta \\ -\delta & e < \delta \end{cases}$$

Deal with Overfitting: overfitting might have large θ
$\arg\min_\theta \sum_{i=1}^N (\theta^T\phi(x_i) - y_i)^2 + \lambda\|\theta\|_2^2$
Closed Form: $(\Phi^T\Phi + \lambda I_d)\theta = \Phi^TY$
$\lambda \to 0$ same as original; $\lambda \to \infty, \hat{\theta} \to \vec{0}$
When selecting λ, the naive training on all data is not work since $\lambda = 0$ minimizes the overall error.

   - We need to Train λ_i on **training set** to minimize the cost function
   - Measure error on the **hold-out set** D^{ho} and find the λ that minimize $\epsilon^{ho} = \sum_{x_i, y_i \in D^{ho}} (y_i - (\theta_\lambda^*)^T x_i)^2$

4. Gradient Descent:
Start with $\theta^{(0)} \in R^d$
In each iteration, update θ until $\|\theta^{(k)} - \theta^{(k+1)}\| < \epsilon$

$$\theta^{(k+1)} = \theta^{(k)} - \rho\frac{\partial J(\theta)}{\partial\theta}|_{\theta^{(k)}}$$

$O(Nd)$ in each iteration.
Stochastic gradient descent: randomly select 1 data entry when computing derivative in each iteration.

5. K-fold cross validation
divide Data set to k equally large sets $\{D_1, D_2, \dots, D_k\} \in D$

   - For $\lambda \in \{\lambda_1, \lambda_2 \dots, \lambda_p\}$
     - For $i = 1, 2, \dots, k$
       * train on $\bigcup_{j\ne i} D^j$ and get $\theta_i^*(\lambda)$
       * compute validate error on $D^i \to \epsilon_i^{ho}(\lambda)$
     - compute average of $\{\epsilon_i^{ho}(\lambda)\}$: $\epsilon^{ho} = \frac{1}{k}\sum_{i=1}^k \epsilon^{ho}(\lambda)$
   - select $\lambda^* = \min_{\{\lambda_1, \lambda_2 \dots, \lambda_p\}} \epsilon^{ho}(\lambda)$

6. Probability review:
Condition: $P(X|Y) = \frac{P(X,Y)}{P(Y)} \Leftrightarrow P(X,Y) = P(X|Y)P(Y)$
Bayes law: $P(X|Y)P(Y) = P(Y|X)P(X)$
Chain rule of Probability: $P(x_1, x_2, \dots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\dots P(x_n|x_1, \dots, x_{n-1})$ Independent: $P(X,Y) = P(X)P(Y)$
Expectation: $E(f(X)) = \sum f(x)p(x)$ or $\int f(x)p(x)dx$
Given $X \perp\!\!\!\perp Y$, $E[XY] = E[X]E[Y]$
iid r.v: independent and identically destributed

7. Maximum Likelihood Estimation (MLE):
$\theta^* = \arg\max_\theta P_\theta(D)$ Under such θ^*, probability of observing the given dataset is maximum. $L(\theta) = P(D|\theta) = \prod P(x_i)$, maximize $L(\theta)$. Solve: $\frac{\partial log(L(\theta))}{\partial\theta}$

8. Logestic Regression:
$w\phi(x) > 0 \Rightarrow g(x) = 1, w\phi(x) \le 0 \Rightarrow g(x) = 0$ Then, we have the model:

$$P(y = 1|x) = \delta(w^T\phi(x)) = \frac{1}{1 + e^{-w^T\phi(x)}}$$

Training:
Train through MLE $\max_w P_w(D) = \max_w P_w(y_1|x_1)\cdots P_w(y_N|x_N)$
We can write: $P_w(y_i|x_i) = P_w(y_i = 1|x_1)^{y_i} P_w(y_i = 0|x_1)^{1-y_i}$
$L(w) = logP_w(D) = \sum_{i=1}^N log(\frac{1}{1+e^{-w^T\phi(x_i)}})^{y_i} + log(\frac{1}{1+e^{w^T\phi(x_i)}})^{1-y_i}$
$= \sum_{i=1}^N (y_i w^T\phi(x_i)) - log(1 + e^{w^T\phi(x_i)})$
equivilent to minimize $-L(w)$
$\Rightarrow \arg\min_w J(w)\sum_{i=1}^N[-y_i w^T\phi(x_i) + log(1 + e^{w^T\phi(x_i)})]$ take derivative: $\frac{\partial J(w)}{\partial w} = \sum_{i=1}^N -y_i\phi(x_i) + \phi(x_i)\frac{1}{1+e^{-w^T\phi(x)}}$
No closed form, use GD.

Deal with overfitting:
Do regularization on w.
$\min_w \bar{J}(w) = J(w) + \frac{\lambda}{2}\|w\|_2^2$
In GD, it is just $\frac{\partial\bar{J}(w)}{\partial w} = \sum_{i=1}^N \phi(x_i)(-y_i + \frac{1}{1+e^{-w^T\phi(x)}}) + \lambda w$