Linear Regression

Problem Definition

Given a dataset $D = \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \ldots, (\boldsymbol{x_m}, y_m)\}$, and each instance $\boldsymbol{x_i}$ is described by d properties $\boldsymbol{x_i} = (\boldsymbol{x_{i1}}; \boldsymbol{x_{i2}}; \ldots; \boldsymbol{x_{id}})$, $y_i \in \mathbb{R}$. Linear regression try to learn a linear model with parameters $(\boldsymbol{w}, b)$:

$$f(\boldsymbol{x_i}) = \boldsymbol{w}^T \boldsymbol{x_i} + b, f(\boldsymbol{x_i}) \simeq y_i, \tag{1}$$

where $w = (w_1; w_2; \ldots; w_d)$.

If we define the loss function as Mean Square Error (MSE), then the Eq. 1 can be solved as:

$$\arg\min_{(\boldsymbol{w},b)} \frac{1}{m} \sum_{i=1}^{m} (f(\boldsymbol{x_i}) - y_i)^2 = \arg\min_{(\boldsymbol{w},b)} \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \boldsymbol{x_i} + b - y_i)^2, \tag{2}$$

If we define $E_{(\boldsymbol{w},b)} = \sum_{i=1}^{m} (y_i - \boldsymbol{w}^T \boldsymbol{x_i} - b)^2$, we can derive the partial derivatives toward $\boldsymbol{w}$ and b:

$$\frac{\partial E_{(\boldsymbol{w},b)}}{\partial \boldsymbol{w}} = \frac{2}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \boldsymbol{x_i} + b - y_i) \boldsymbol{x_i}, \quad \frac{\partial E_{(\boldsymbol{w},b)}}{\partial b} = \frac{2}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \boldsymbol{x_i} + b - y_i). \tag{3}$$

Then we can derive the close-form solution for $\boldsymbol{w}$ and b:

$$\boldsymbol{w} = \left(\sum_{i=1}^{m} \boldsymbol{x_i} \boldsymbol{x_i}^T - m \bar{\boldsymbol{x}} \bar{\boldsymbol{x}}^T \right)^{-1} \left(\sum_{i=1}^{m} y_i \boldsymbol{x_i} - m \bar{y} \bar{\boldsymbol{x}} \right), \tag{4}$$

where $\bar{\boldsymbol{x}}$ is the mean of $\boldsymbol{x_i}$ and y is the mean of y_i.

If we incorporate b into $\boldsymbol{w}$ and rewrite $\mathbf{X}$ as the following:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1d} & 1 \\ x_{21} & x_{22} & \ldots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \ldots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1^T & 1 \\ \boldsymbol{x}_2^T & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_m^T & 1 \end{pmatrix}$$

if $\mathbf{X}^T \mathbf{X}$ is invertible, we can derive the following solutions:

$$\boldsymbol{w} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}.$$

However, in practice we want $\mathbf{X}^T \mathbf{X}$ is well-conditioned so that the Condition number $k(\mathbf{X}^T \mathbf{X})$ should not be too large, otherwise there is numerical instability.

> For matrix $\mathbf{A}$, $k(A) = \sigma_{max}/\sigma_{min}$. σ_{max} is the largest singular value for matrix $\mathbf{A}$ and σ_{min} is the smallest singular value for matrix $\mathbf{A}$. Ridge regression can be used to mitigate the issue. For ridge regression, the loss function is :
>
> $$\sum_{i=1}^{m}(y_i - \boldsymbol{w}^T\boldsymbol{x_i} - b)^2 + \alpha\sum_{i=1}^{d}(w_i)^2,$$
>
> where α is the coefficient to penalize the $\boldsymbol{w}$. The close form solution for ridge regression is:
>
> $$\boldsymbol{w} = \left(\mathbf{X}^T\mathbf{X} + \alpha\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}.$$

> If there are noise ϵ in the data set, to use the linear regression, some assumptions must be hold: such as the noise are independent across samples and all noise have zero mean and shared variance.

Log-Linear Regression

Also, we can use the **log-linear regression** if we want to let $e^{\boldsymbol{w}^T\boldsymbol{x}+b}$ approximate y:

$$\ln y = \boldsymbol{w}^T\boldsymbol{x} + b.$$

Logistic Regression

If we want to introduce the Linear Regression to do binary classification task, we can use the unit-step function:

$$y = \begin{cases} 1, & \boldsymbol{w}^T\boldsymbol{x} + b > 0; \\ 0.5, & \boldsymbol{w}^T\boldsymbol{x} + b = 0; \\ 0, & \boldsymbol{w}^T\boldsymbol{x} + b < 0. \end{cases}$$

However, the unit-step function is not a continuous function, and we can use the logisitic function as the surrogate function:

$$y = \frac{1}{1 + e^{-(\boldsymbol{w}^T\boldsymbol{x}+b)}},$$

then we can derive the log odds (or logit) as:

$$\ln\frac{y}{1-y} = \boldsymbol{w}^T\boldsymbol{x} + b.$$

The characteristics of Logistic Regression:

- No assumption about the distribution of data, so that can avoid the inaccurate assumption about data distribution
- Can not only generate class for the sample, but also the probability, which can be extend to other areas, such as uncertainty quantitation.

- The log-likelihood function is a convex functions of arbitrary order. (prove?)

Logistic Distribution

Let X as continuous random samples and follow the logistic distribution as the following PDF and CDF.

$$F(x) = P(X \leq x) = \frac{1}{1 + e^{-(x-\mu)/\gamma}}$$

,

and

$$f(x) = F'(x) = \frac{e^{-(x-\mu)/\gamma}}{\gamma(1 + e^{-(x-\mu)/\gamma})^2},$$

where μ is the position parameter and $\gamma > 0$ is the shape parameter.

Use the maximum likelihood mathod to obtain the solution

We can use the maximum likelihood method (MLE) to solve $\boldsymbol{w}$ and b. For each sample $(\boldsymbol{x}, y)$, where $y \in \{0, 1\}$, the probability is given by the Bernoulli distribution, parameterized by the output of the sigmoid function, $\pi(\boldsymbol{x})$.

$$\pi(\boldsymbol{x}) = p(y = 1|\boldsymbol{x}) = \frac{e^{\boldsymbol{w}^T\boldsymbol{x}+b}}{1 + e^{\boldsymbol{w}^T\boldsymbol{x}+b}}$$

$$p(y = 0|\boldsymbol{x}) = \frac{1}{1 + e^{\boldsymbol{w}^T\boldsymbol{x}+b}}$$

If $p(y = 1|\boldsymbol{x})$ is set as $\pi(x)$, then $p(y = 1|\boldsymbol{x}) = 1 - \pi(x)$. The likelihood function is:

$$\prod_{i=1}^{N} [\pi(x_i)]^{y_i} [1 - \pi(x_i)]^{1-y_i}, \tag{5}$$

then the log likelihood function is:

$$\begin{aligned} L(\boldsymbol{w}, b) &= \sum_{i=1}^{N} [y_i \ln \pi(x_i) + (1 - y_i) \ln (1 - \pi(x_i))] \\ &= \sum_{i=1}^{N} \left[y_i \ln \frac{\pi(x_i)}{1 - \pi(x_i)} + \ln (1 - \pi(x_i)) \right] \\ &= \sum_{i=1}^{N} \left[y_i(\boldsymbol{w}^T\boldsymbol{x}_i) - \ln \left(1 + e^{\left(\boldsymbol{w}^T\boldsymbol{x}+b\right)} \right) \right]. \end{aligned}$$

There is no analytic solution for the log likelihood function, but as it is a convex function of arbitrary order, we can use the gradient descend method or Newton method to iteratively find the solution.

Maximum Entropy Model

In Maximum Entropy Model, when learn the probability model, the maximum entropy model is the best model among all the possible models. Usually constraints are used to identify the set of probability models. Therefore, the Maximum Entropy Model can be viewed as the selection of the maximum entropy model under the constraints.

Suppose probability distribution for a discrete variable X is $P(X)$, then the entropy is:

$$H(P) = -\sum_x P(x)\log P(x),$$

$0 \leq H(P) \leq log|X|$. Only when the distribution of X is the uniform distribution, the $H(X)$ has the largest value.

The definition of Maximum Entropy Model

Suppose the classification model is a conditional probability distribution $P(Y|X), X \in \chi \subseteq \mathbf{R}^n$ stands for the input, $Y \in \mathcal{Y}$, $\mathcal{X}, \mathcal{Y}$ stand for the set of input and output.

Given the dataset $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ with N samples, we can define the empirical distribution function $\tilde{P}(X, Y)$ and $\tilde{P}(X)$ as:

$$\tilde{P}(X = x, Y = y) = \frac{v(X = x, Y = y)}{N} \tag{6}$$

$$\tilde{P}(X = x) = \frac{v(X = x)}{N} \tag{7}$$

$v(\cdot)$ stands for the occurence count in the train sample. We use $0 - 1$ feature fucntion $f(x, y)$ to describe the relationship between x and y:

$$f(x, y) = \begin{cases} 1, & x \text{ and } y \text{ satisfy the requirement;} \\ 0, & \text{otherwise;} \end{cases}$$

We can use $E_{\tilde{P}}(f)$ to represent the expectation of $f(x, y)$ over emperical distribution $\tilde{P}(X, Y)$: $E_{\tilde{P}}(f) = \sum_{x,y} \tilde{P}(x, y) f(x, y)$ and the expectation of model $P(Y|X)$ and emperical distribution $\tilde{P}(X)$: $E_P(f) = \sum_{x,y} \tilde{P}(x) P(y|x) f(x, y)$. If the model can well fit the train dataset, we can suppose the equal between these two expectation $E_{\tilde{P}}(f) = E_P(f)$. This is the formular for the constraint condition for the learned model. If there are n feature function $f_i(x, y)$, there are n constraints.

Definition: Suppose the set of models that satisfy all the constraints is:

$$\mathcal{C} \equiv \{P \in \mathcal{P} | E_P(f_i) = E_{\tilde{P}}(f_i), \quad i = 1, 2, \cdots, n\}$$

the conditional entropy defined on the conditional probability distribution $P(Y|X)$ is:

$$H(P) = -\sum_{x,y} \tilde{P}(x) P(y|x) \log P(y|x), \tag{8}$$

then the model has largest conditional entropy in the set $\mathcal{C}$ is called the Maximum Entropy Model.

The Maximum Entropy Model is the constrained optimization problem:

$$\begin{aligned} \max_{P \in \mathbf{C}} H(P) &= -\sum_{x,y} \tilde{P}(x)\, P(y|x) \log P(y|x). \\ \text{s.t.} \quad E_P(f_i) &= E_{\tilde{P}}(f_i), \quad i = 1, 2, \cdots, n \\ \sum_y P(y|x) &= 1 \end{aligned}$$

We can use the Lagrange Multiplier Method and duality can solve the problem.

> The maximum of duality is equal to the MLE of Maximum Entropy Model

> The conditional entropy is defined as $H(Y|X) = -\sum_{x,y} P(X, Y) \log P(y|x) = \sum_x P(x)\, H(Y|X = x) = -\sum_x P(x) \sum_y P(y|x) \log P(y|x)$.

LDA

The idea of LDA is to project the samples to a line $\boldsymbol{w}^T \boldsymbol{x} = 0$, so that instances in the same class should be as close as possbile, but instances in the different classes should be as far as possible. When inference, a test sample is projected to the line and then decide the class based on the location.
Define $X_i, \mu_i \in \mathbb{R}^d, \Sigma_i \in \mathbb{R}^{d \times d}$ stands for the set of samples $x_i \in \mathbb{R}^d$ in the $i \in \{0, 1\}$ class, the mean and convariance.

If we project these value so the line, then the means are projected as $\boldsymbol{w}^T \boldsymbol{\mu_0} \in \mathbb{R}$ and $\boldsymbol{w}^T \boldsymbol{\mu_1} \in \mathbb{R}$, and the convariances are projected as $\boldsymbol{w}^T \Sigma_0 \boldsymbol{w} \in \mathbb{R}$ and
$\boldsymbol{w}^T \Sigma_1 \boldsymbol{w} \in \mathbb{R}$. In order to ensure that after projection, the instances in the same class are closer, we can minimize the convariances after projection $\boldsymbol{w}^T \Sigma_0 \boldsymbol{w} + \boldsymbol{w}^T \Sigma_1 \boldsymbol{w}$. Also, we can maximize the mean of different classes to enlarge the distances of instances, e.g. maximize $||\boldsymbol{w}^T \boldsymbol{\mu_0} - \boldsymbol{w}^T \boldsymbol{\mu_1}||_2^2$. We can maximimze the following objective function:

$$J = \frac{||\boldsymbol{w}^T \boldsymbol{\mu_0} - \boldsymbol{w}^T \boldsymbol{\mu_1}||_2^2}{\boldsymbol{w}^T \Sigma_0 \boldsymbol{w} + \boldsymbol{w}^T \Sigma_1 \boldsymbol{w}} = \frac{\boldsymbol{w}^T (\boldsymbol{\mu_0} - \boldsymbol{\mu_1})(\boldsymbol{\mu_0} - \boldsymbol{\mu_1})^T \boldsymbol{w}}{\boldsymbol{w}^T (\Sigma_0 + \Sigma_1) \boldsymbol{w}} \tag{9}$$

We define within-class scatter matrix $S_w = \Sigma_0 + \Sigma_1$ and between-class scatter matrix $S_b = (\boldsymbol{\mu_0} - \boldsymbol{\mu_1})(\boldsymbol{\mu_0} - \boldsymbol{\mu_1})^T$, the above objective function can be rewritten as the generalized Rayleigh quotient about S_b and S_w:

$$J = \frac{\boldsymbol{w}^T S_b \boldsymbol{w}}{\boldsymbol{w}^T S_w \boldsymbol{w}}.$$

Let $\boldsymbol{w}^T S_w \boldsymbol{w} = 1$, then the objecitve function is formulated as:

$$\min_{\boldsymbol{w}} -\boldsymbol{w}^T S_b \boldsymbol{w}$$
$$\text{s.t.} \quad \boldsymbol{w}^T S_w \boldsymbol{w} = 1.$$

After applying the Method of Lagrange Multipliers and calculate the gradient, we can obtain the following equation:

$$S_b \boldsymbol{w} = \lambda S_w \boldsymbol{w},$$

where λ is the Lagrange Multiplier. Since the direction of $S_b \boldsymbol{w}$ is same as the $\boldsymbol{\mu_0} - \boldsymbol{\mu_1}$, we can set $S_w \boldsymbol{w} = \alpha(\boldsymbol{\mu_0} - \boldsymbol{\mu_1})$, we can derive the solution as:

$$\boldsymbol{w} = S_w^{-1}(\boldsymbol{\mu_0} - \boldsymbol{\mu_1}).$$

When calculating, to ensure the numerical stability, we first do the singular decomposition to $S_w = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$, then we can get $\boldsymbol{w} = \mathrm{V}\Sigma^{-1}\mathrm{U}^{\mathrm{T}}(\boldsymbol{\mu_0} - \boldsymbol{\mu_1})$.

LDA can be explained from the Bayesian theory and can be proved that when two classes share the same priors and meet the Gaussian distribution and the covariances are same, LDA can achieve the optimal classification performance. LDA can be extend to multi classes and also be applied to the supervised dimension-reduction technique.

> Grad matrix XX^T stands for the sample-to-sample similarity, and Scatter matrix X^TX stands for the unnormalized convariance.

Multi-class classification

In order to classify N classes, we can adopt OvO (One vs. One), OvR (One vs. Rest) and MvM(Many vs. Many). OvO needs to train $N(N-1)/2$ classifiers and OvR needs to train N classifier with more memory and computation cost. MvM treats several classes as the postive and several as the negative class. A typical technique is called ECOC (Error Correcting Output Codes):

- Encoding: divide the N classes M times, each time some classes are labelled as postive and some are negative, we can train M classifiers;
- Decoding: classify the test sample with M classifiers and generate a code for the prediction. Compare the code with the code of each class and return the smallest distance as the final prediction result.

ECOC has the tolerance and correction ability to the encoding errors. And the longer the encoding length, the more powerful of correction but with high computation and storage cost.

Questions

Linear Regression

- Q1: Derive the closed-form solution for the ordinary least squares (OLS) estimator.

- Q2: Explain when this solution may not exist or may be unstable.
- Q3: What are the key assumptions of the linear regression model? Describe what happens when the assumption doesn't hold (such as correlations between features).
- Q4: Explain how Ridge and Lasso regression modify the loss function. What are the main differences in their effects on the model parameters and feature selection?
- Q5: Whether linear regression can be used in non-linear situation?

Logistic Regression

- Q1: Starting from the Bernoulli likelihood, derive the log-likelihood function for logistic regression and explain why there is no closed-form solution.
- Q2: Describe how gradient descent (or stochastic gradient descent) is used to fit a logistic regression model. Why is the sigmoid function important for the gradient's stability?
- Q3: Why the loss function (Negative Log-Likelihood) is convex?
- Q4: Why MSE is not a good loss function for LR?
- Q5: How is the decision boundary determined in logistic regression? Can it be nonlinear? Explain how and when.

Answer

Q1

We consider a supervised learning problem with:

- (n) samples and (d) features.
- Design matrix $\mathbf{X} \in \mathbb{R}^{n \times (d+1)}$
  Each row $\mathbf{x}_i^T = [x_{i1}, \ldots, x_{id}, 1]$ (the last term is bias).
- Target vector $\mathbf{y} \in \mathbb{R}^n$.
- Parameters $\boldsymbol{\beta} = [w_1, \ldots, w_d, b]^T \in \mathbb{R}^{d+1}$.

The general form of linear regression is:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1d} & 1 \\ x_{21} & x_{22} & \ldots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nd} & 1 \end{pmatrix}$$

$$\boldsymbol{\beta} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{pmatrix}$$

The least squares objective function is :

$$L(\boldsymbol{\beta}) = ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}).$$

Compute gradient w.r.t $\boldsymbol{\beta}$:

$$\nabla_{\boldsymbol{\beta}} L = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}$$

Set the gradient to zero:

$$\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\mathbf{y}$$

Therefore, the solution by ordinary least squares (OLS) is:

$$\boxed{\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}}$$

Assumption:

1. Invertibility. The formula requires X to have full column rank so that $\mathbf{X}^T\mathbf{X}$ is nonsingular. Otherwise, use Moore-Penrose pseudoinverse.
2. Statistical properties: $\boldsymbol{\epsilon}$ have zero mean, and are uncorrelated and homoscedastic (same variance). $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator. If $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$:

   $$p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\beta}, \sigma^2) = (2\pi\sigma^2)^{-\frac{n}{2}} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2\right).$$

   Maximizing this w.r.t. $\boldsymbol{\beta}$ is equivalent to minimizing $L(\boldsymbol{\beta})$,
   So **OLS = MLE**.
   Estimator distribution $\hat{\boldsymbol{\beta}} \sim \mathcal{N}(\boldsymbol{\beta}, \sigma^2(\mathbf{X}^T\mathbf{X})^{-1})$.

Q2

In practice we want $\mathbf{X}^T\mathbf{X}$ is well-conditioned so that the Condition Number $k(\mathbf{X}^T\mathbf{X})$ should not be too large, otherwise there is numerical instability.

> For matrix $\mathbf{A}$, $k(A) = \sigma_{max}/\sigma_{min}$. σ_{max} is the largest singular value for matrix $\mathbf{A}$ and σ_{min} is the smallest singular value for matrix $\mathbf{A}$.

--- SCENARIO A: ILL-CONDITIONED MODEL (High Correlation) ---

Condition Number: 3.51e+07

Original Coefficients: [2.06 15.06 -8.07]

Perturbed Coefficients: [2.08 17.53 -10.55]

--- SCENARIO B: WELL-CONDITIONED MODEL (Low Correlation) ---

Condition Number: 1.54e+02

Original Coefficients: [1.94 3.02 4.]

Perturbed Coefficients: [1.99 3.01 4.]

Q3

The one of main assumptions is the linear relationship between the X and y. For the residuals ϵ, they should be independent and have zero means and the constant variance. If not, we can use the weighted least squares (WLS), or step-wise regression.

If the multicollinearity exists, we can use the regularization (e.g., Ridge):

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}.$$

Q4

Let's look at the cost function, which is the Sum of the Squared Errors (SSE) plus the regularization term.

Lasso (L1 regularization):

$L_{\text{lasso}} = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1.$

The L1 term is **non-differentiable**; use **subgradient**:

$$\partial|w_j| = \begin{cases} +1, & w_j > 0, \\ -1, & w_j < 0 \end{cases}$$

Optimization is usually done via **coordinate descent** or **proximal methods**. The penalty's contribution to the gradient is a constant value. This constant push is what eventually forces the weight to become exactly zero.

Ridge (L2 regularization):

$$L_{\text{ridge}} = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2.$$

Gradient:

$$\nabla_{\boldsymbol{\beta}} L_{\text{ridge}} = 2\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) + 2\lambda\boldsymbol{\beta}.$$

Closed-form solution:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T\mathbf{y}.$$

Q5

Notice: both $\mathbf{w}$ and $\mathbf{x}$ are column vectors ($\mathbf{w}, \mathbf{x} \in \mathbf{R}^d$), b and y are scalars.

Also, we can use the **log-linear regression** if we want to let $e^{\mathbf{w}^T\mathbf{x}+b}$ approximate y:

$$\ln y = \mathbf{w}^T\mathbf{x} + b.$$

Linear regression can be used in non-linear situations by transforming the features. For example, by creating polynomial features (e.g., $\mathbf{x}^2, \mathbf{x}^3$), the model remains linear in its parameters but can fit a non-linear curve to the data. This is called generalized linear model (GLM).

Logistic Regression

Q1

For each sample $(\mathbf{x}, y)$, where $y \in \{0, 1\}$, the probability is given by the Bernoulli distribution, parameterized by the output of the sigmoid function, $\pi(\mathbf{x})$.

$$\pi(\mathbf{x}) = p(y = 1|\mathbf{x}) = \frac{e^{\mathbf{w}^T\mathbf{x}+b}}{1 + e^{\mathbf{w}^T\mathbf{x}+b}}$$

$$p(y = 0|\mathbf{x}) = \frac{1}{1 + e^{\mathbf{w}^T\mathbf{x}+b}}$$

The log-likelihood for all (n) samples:

$$\ell(\mathbf{w}) = \sum_{i=1}^{n} \left[y_i \log \pi(\mathbf{x}_i) + (1 - y_i) \log(1 - \pi(\mathbf{x}_i)) \right].$$

There is no analytic solution for the log likelihood function, but as it is a convex function, we can use the gradient descend method or Newton method to iteratively find the solution.

Q2

The most common cost function for logistic regression is the **Negative Log-Likelihood** (also called Log Loss or Binary Cross-Entropy):

$$J(\mathbf{w}) = -\ell(\mathbf{w}) = -\sum_{i=1}^{n} \left[y_i \log \pi(\mathbf{x}_i) + (1 - y_i) \log(1 - \pi(\mathbf{x}_i)) \right].$$

$$\nabla_{\mathbf{w}} J = \frac{1}{N} \sum_{i=1}^{N} (\pi(\mathbf{x}_i) - y_i)\, \mathbf{x}_i.$$

The update rule is:

$$\mathbf{w} := \mathbf{w} - \alpha \nabla_{\mathbf{w}} J.$$

The gradient is propotional to the error. If $\pi - \mathbf{y}$ is large, $\mathbf{w}$ will update values fast, otherwise it will update slowly.

The derivative of the sigmoid function $\sigma(z)$ with respect to its input z is $\sigma'(z) = \sigma(z)(1 - \sigma(z))$. It is bounded as the maximum value is 0.25, so that it can prevent exploding gradients.

Q3

Calculate the First Derivative:

$$\nabla_{\mathbf{w}} J = \frac{1}{N} \sum_{i=1}^{N} (\pi(\mathbf{x}_i) - y_i)\, \mathbf{x}_i.$$

Calculate the Second Derivative (Hessian):

$$H = \nabla(\nabla J(\mathbf{w})^T) = \nabla \left(\frac{1}{N} \sum_{i=1}^{N} (\pi(\mathbf{x}_i) - y_i)\ \mathbf{x}_i \right) = \frac{1}{N} \sum_{i=1}^{N} (\pi_i(1 - \pi_i)\mathbf{x}_i)\mathbf{x}_i^T$$

Let us assume any non-zero vector $\mathbf{z} \in \mathbb{R}^d$, then

$$\mathbf{z}^T H \mathbf{z} = \mathbf{z}^T \frac{1}{N} \sum_{i=1}^{N} (\pi_i(1 - \pi_i)\mathbf{x}_i)\mathbf{x}_i^T \mathbf{z} = \frac{1}{N} \sum_{i=1}^{N} \pi_i(1 - \pi_i)\mathbf{z}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{z} = \frac{1}{N} \sum_{i=1}^{N} \pi_i(1 - \pi_i)(\mathbf{z}^T \mathbf{x}_i)^2 \geq 0$$

So the loss function (Negative Log-Likelihood) is convex.

Q4

Reason 1: the vanishing gradient problem.

$$J = (y - \hat{y})^2$$

$$\frac{\partial J}{\partial \hat{y}} = -2(y - \hat{y})$$

$$\frac{\partial \hat{y}}{\partial w_j} = \pi(x_i)(1 - \pi(x_i))\, x_j = \hat{y}(1 - \hat{y})\, x_j$$

so

$$\frac{\partial J}{\partial w_j} = -2(y - \hat{y})\hat{y}(1 - \hat{y})\, x_j$$

when $y = 1, \hat{y} = 0.01$, the final gradient is about -0.0196 x_j, causing the gradient to vanish.

If we use the NLL, the loss is about -0.99 x_j, which is large.

Reason 2: no global minimal (not convex)

The computation is complex, so we can set $x = 1$ to make it simple. The second derivative is:

$$\frac{\partial J}{\partial w_j^2} = 2(\pi(w) - (1 - \pi(w)))^2 - 2(y - \pi(w))\pi(w)(1 - \pi(w))(1 - 2\pi(w))$$

If $y = 1, \pi(w) = 0.2$, the final value is -0.1024. The second derivative is negative, the function is non-convex.

Q5

$$p(y = 1|\mathbf{x}) = \frac{e^{\mathbf{w}^T\mathbf{x}+b}}{1 + e^{\mathbf{w}^T\mathbf{x}+b}}$$
$$p(y = 0|\mathbf{x}) = \frac{1}{1 + e^{\mathbf{w}^T\mathbf{x}+b}}$$
$$\frac{p(y = 1|\mathbf{x})}{p(y = 0|\mathbf{x})} = e^{\mathbf{w}^T\mathbf{x}+b} \geq 1$$
$$\mathbf{w}^T\mathbf{x} + b \geq 0.$$

This is the equation of a hyperplane. So, a standard logistic regression model, by its very definition, learns a linear decision boundary.

However, we can create new, more complex features from the original ones to create a nonlinear boundary. For example we can create a new set of features : $[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_1\mathbf{x}_2]$. Even the model is still a linear model because it's linear with respect to its new, expanded feature set, but the decision boundary is not linear anymore.