$(X\underline{w})^T = \underline{w}^T X^T$

Probabilistic Perspective

Vikas Dhiman

Tuesday 7th October, 2025

1 Optimization perspective is not enough

So far in this course, we stuck to an optimization perspective on Machine Learning. We identified the following steps:

1. Collecting a training dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$
2. Selecting a model $\hat{\mathbf{y}} = f(\mathbf{x}; \mathbf{w})$ with weights (also called the parameters) $\mathbf{w}$.
3. Selecting a loss function $l(\mathbf{y}_i, \hat{\mathbf{y}}_i)$
4. Training: Minimizing the total or average loss function to find the optimal weights

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} l(\mathbf{y}_i, \hat{\mathbf{y}}_i; \mathbf{w}) \tag{1}$$

$$= \arg \min_{\mathbf{w}} \underbrace{\frac{1}{n} \sum_{i=1}^{n} l(\mathbf{y}_i, f(\mathbf{x}_i; \mathbf{w}))}_{L(\mathcal{D}; \mathbf{w})} \tag{2}$$

[resume]For a new input $\mathbf{x}^* \notin \mathcal{D}$, use $\hat{\mathbf{y}}^* = f(\mathbf{x}^*; \mathbf{w}^*)$ as the prediction.

For linear models (or other "simple" models) this perspective works fine. However, in general it does not.

Question Imagine you are designing a pedestrian detector (input: image, output: a box around all pedestrians in the image) for an autonomous car (or a face detector for your ring-camera based security system). You collect a training dataset of pedestrians in Iceland and then try to deploy it in India.

1. How well is your pedestrian detector likely to work? Very good, very bad, okayish.
- Why?

- How would describe the reason mathematically?

Answer

The pedestrian detector is likely to be very bad to okayish.

Because pedestrians in Iceland are likely to look different from pedestrians in India. Evidence from facial recognition.

The real world data is called test data. Mathematically, all the samples in both training data and test data must be identically distributed. In other words, they must have the same probability distribution. Let $P(X, Y)$ be the true but unknown probability distribution. All samples, in the training data and the test data must come from the same distribution, $(\mathbf{x}_i, \mathbf{y}_i) \sim P(X, Y)$ and $(\mathbf{x}^*, \mathbf{y}^*) \sim P(X, Y)$.

From a Machine Learning engineer perspective, you can try to get close to this ideal by carefully collecting your training data so that matches the test data as faithfully as possible.

1.1 Independent and Identically distributed

A random variable Z_1 is said to be independent from random variable Z_2, denoted as $Z_1 \perp Z_2$ if either of the three equivalent conditions are met:

1. $P(Z_1, Z_2) = P(Z_1)P(Z_2)$
2. $P(Z_1|Z_2) = P(Z_1)$
3. $P(Z_2|Z_1) = P(Z_2)$

In typical machine learning, we want the data samples to be Independent and Identically Distributed, in short the i.i.d assumption.

1.2 Maximum Likelihood estimation

The optimization perspective can be related to maximum likelihood estimation under the iid (Independent and Indentically distributed) assumpotion.

Let the $\mathbf{x}_i$ and y_i be random vectors for all i. Model the probability distribution as a negative log of the loss function:

$$P((\mathbf{x}_i, y_i)|\mathbf{W}) = \frac{1}{Z} \exp(-l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{3}$$

If the samples are IID, then we can write the probability of the entire dataset as products of sample probabilities

$$P(\mathcal{D}|\mathbf{W}) = \prod_{i=1}^{n} P((\mathbf{x}_i, y_i)|\mathbf{W}) \tag{4}$$

$$P(\mathcal{D}|\mathbf{W}) = \prod_{i=1}^{n} \frac{1}{Z} \exp(-l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{5}$$

A product of exponents is the summation of their powers,

$$P(\mathcal{D}|\mathbf{W}) = \frac{1}{Z} \exp(-\sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{6}$$

Denote

$$L(\mathcal{D}; \mathbf{W}) = \sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i; \mathbf{W}). \tag{7}$$

Minimizing the loss is same as maximizing the probability $P(\mathcal{D}|\mathbf{W})$.

1.2.1 No free lunch theorem

ML: Data → function

In the era of so much machine learning hype it is important to look back at David Wolpert's No free lunch theorems. ECE590 students must read Wolpert's 1996 paper "The Lack of A Priori Distinctions Between Learning Algorithms" and mark the terms, symbols and concepts that do not make sense.

$$\underset{f}{\operatorname{argmin}} \sum_{\mathcal{D}} \| y_i - f(x_i) \|_2$$

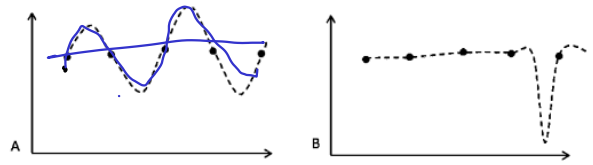


Figure 3. The problem of underdetermination: a sample of data can be described by quite different models (in this case function A and function B).

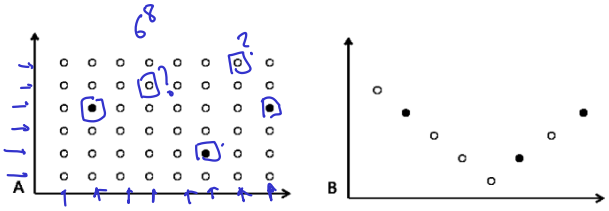


Figure 2. The No Free Lunch Theorem. The information collected so far will not say anything about the values of the function in other regions (case A). For a given subclass of functions, i.e., convex (case B), those algorithms that can take advantage of this structure will perform better than others.

Machine learning theory

$\sum_{i=1}^{n} z_i P(z_i)$ $\quad \frac{1}{n} \sum_{i=1}^{n} z_i$

1.3 Expectation vs Mean

Given i.i.d. random variables $Z_1, Z_2, ...Z_n$ with distribution $P(.)$, write the mathematical definition for the mean and that of the expectation. What is the difference?

Question: What is the diference between Empirical risk and Expected risk?

Mean — Training — True prob distribution — Total loss — Average loss

Answer: Empirical risk is the average loss over the dataset while Expected risk is the expectation of the loss when the inputs and outputs are treated as random variables.

For example, if a dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with $\mathbf{x}_i \in R^n$ and $y_i \in R$ is given and the model $f(\mathbf{x}_i; \mathbf{W})$ predicts labels $\hat{h}_i = f(\mathbf{x}_i; \mathbf{W})$. Then the empirical risk is the average loss over the dataset samples,

$$R_{\mathcal{D}}(\mathbf{W}) = \frac{1}{n} \sum_{i=1}^{n} loss(\hat{y}_i, y_i). \quad (8)$$

On the other hand to compute Expected risk, we assume that the dataset $\mathcal{D}$ was independently sampled (IID assumption) from a distribution $P_{X,Y}$ where the input $\mathbf{x}_i$ is an instance of random variable X and y_i is an instance of random variable Y.

$$R(P_{X,Y}, \mathbf{W}) = E_{X,Y}[loss(\hat{Y}, Y)], \quad (9)$$

where $\hat{Y} = f(X; \mathbf{W})$.

If X and Y are continous random variables, then the expectation is the weighted integraal over all possible values of values of X and Y weighted by the joint probability density function (PDF) of X and Y.

$$R(P_{X,Y}, \mathbf{W}) = E_{X,Y}[loss(\hat{Y}, Y)] = \int_{x \in \Omega_X} \int_{y \in \Omega_Y} loss(\hat{y}, y) f_{X,Y}(x, y) dx dy, \quad (10)$$

where $\hat{y} = f(x; \mathbf{W})$ and Ω_X and Ω_Y is the sample space for X and Y.

1.3.1 Generalization gap bounds in Machine Learning

There are many generalization gap bounds in machine learning, often studied under Machine Learning theory; Probably Approximatly Correct (PAC) theory. Mohri 2012, Foundations of Machine Learning is an excellent book to learn more about it.

If the loss $l(.,.)$ is between [0, 1], then for any $\delta > 0$ with Probability at least $1 - \delta$, we have

$$\max_{\mathbf{W} \in \mathcal{W}} R(P_{X,Y}, \mathbf{W}) - R_{\mathcal{D}}(\mathbf{W}) \leq 2R_m(l) + \sqrt{\frac{-\ln(\delta)}{2n}} \quad (11)$$

where Rademacher Complexity is defined as

Definition 3.1 **Empirical Rademacher complexity**
Let G be a family of functions mapping from Z to $[a, b]$ and $S = (z_1, \ldots, z_m)$ a fixed sample of size m with elements in Z. Then, the empirical Rademacher complexity *of G with respect to the sample S is defined as:*

$$\widehat{\mathfrak{R}}_S(G) = \mathop{\mathrm{E}}_{\boldsymbol{\sigma}} \left[\sup_{g \in G} \frac{1}{m} \sum_{i=1}^{m} \sigma_i g(z_i) \right], \quad (3.1)$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)^\top$, with σ_is independent uniform random variables taking values in $\{-1, +1\}$.[1] The random variables σ_i are called Rademacher variables.

Definition 3.2 **Rademacher complexity**
Let D denote the distribution according to which samples are drawn. For any integer $m \geq 1$, the Rademacher complexity *of G is the expectation of the empirical Rademacher complexity over all samples of size m drawn according to D:*

$$\mathfrak{R}_m(G) = \mathop{\mathrm{E}}_{S \sim D^m}[\widehat{\mathfrak{R}}_S(G)]. \tag{3.2}$$

1.3.2 Bayes Rule

Question: Given two random variables X and Y specify the relationship between $P(X|Y)$ and $P(Y|X)$ using Bayes rule?

Answer

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{12}$$

Question: Using CIFAR 10 dataset $\mathcal{D}$ you want to find your neural network with weights $\mathbf{W}$. Define Prior, Likelihood and Posterior in terms of the weights $\mathbf{W}$ and data $\mathcal{D}$. Write Bayes rule to determin posterior from likehood and prior.

Answer:

- Prior is $P(\mathbf{W})$ i.e. the probability of weights $\mathbf{W}$ before we have seen the dataset.
- Likelihood is $P(\mathcal{D}|\mathbf{W})$ i.e. the probability of observing the dataset once if we pick a particular choice of weighs $\mathbf{W}$.
- Posterior is $P(\mathbf{W}|\mathcal{D})$ i.e the probability of chosing weights given the dataset $\mathcal{D}$.

By Bayes rule, we have:

$$P(\mathbf{W}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{W})P(\mathbf{W})}{P(\mathcal{D})} \tag{13}$$

Question: How can regularization can be interpretted as an application of the Bayes theorem?

Given the dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, a model $\hat{y}_i = f(\mathbf{x}_i; \mathbf{W})$, a regularizer $R(\mathbf{W})$ and a loss function $l(y_i, \hat{y}_i)$, show that the following optimization problem can be interpreted as maximum-a-posteriori estimation. In the process show that for the interpretation, we need the IID (independently, identically distributed) assumption over the dataset. List any other assumptions that you need for the interpretation.

$$\mathbf{W}^* = \arg \min_{\mathbf{W}} \sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i; \mathbf{W})) + \lambda R(\mathbf{W}), \tag{14}$$

where λ is some positive constant that balances between the loss function and the regularizer.

Advantages of Stochastic Gradient Descent over GD

SGD

while $\left|\nabla_w L(B; \underline{w})\right| \geq 1e\text{-}4$:

for B in D :
batch Data

$B = D$

$$\underline{w}_{t+1} = \underline{w}_t - \alpha_t \sum_B \nabla_{\underline{w}} l(\hat{y}_i, y_i; \underline{w})$$

Training, Validation, Test

Regularization

GD minimizes Empirical Risk $\sum_D \nabla_w l(\hat{y}_i, y_i; w)$

SGD makes a compromize between making progress to minimize Empirical Risk and making fast progress

Expected Risk

$B \sim D \sim \mathbb{P}_{x,y}$

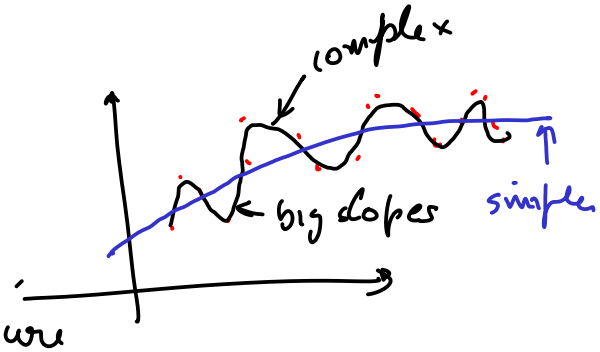
SGD inherently regularizes is a regularizing strategy

Understanding Deep Learning 2023-25

Regularizing / Regularizers

Occam's Razor

"simplest"



→ Neural Network : smaller architecture

→ Smaller weight magnitude ⟷ Sparse weight magnitude

more smoothness
smaller derivative / slope

$$\hat{y} = \underline{w}^T \underline{x}$$

$$\frac{\partial \hat{y}}{\partial \underline{x}} = \underline{w}^T \leftarrow \text{big weight} = \text{big slope}$$

hyperparameter

$$L(D; \underline{w}) = \frac{1}{n}\sum_{i=1}^{n} l(\hat{y}_i, y_i; \underline{w}) + \lambda \|\underline{w}\|_2^2$$

Empirical risk 0.~1 0.5

Regularization term 0.5

L_2-regularization
Weight decay

L_1-regularization leads to sparser weights

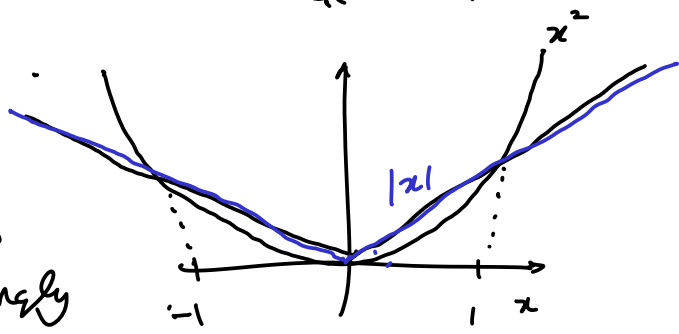
$l_2(x) = x^2$

$l_1(x) = |x|$

$$\min_x \; f(x) + x^2$$

$$\min_x \; f(x) + |x| \leftarrow \text{force } |x| \text{ to zero more strongly than } x^2$$

$|x|^{1/2}$



$x = 0.01$

$|x| = 0.01$

$x^2 = 0.0001$

Regularization is optimization perspective

Probabilistic perspective : Maximum-a-posteriori (MAP) vs Maximum likelihood estimation (MLE)

Bayes Theorem

$$P(D, \underline{w}) = P(D|\underline{w})\,P(\underline{w}) = P(\underline{w}|D)\,P(D)$$

Data, Weights

$$\Rightarrow \boxed{P(\underline{w}|D) = \frac{P(D|\underline{w})\,P(\underline{w})}{P(D)}}$$

likelihood, Prior, Evidence

Posterior distribution

$$\boxed{P(H|D) = \frac{P(D|H)\,P(H)}{P(D)}}$$

MLE:

$$\max_{\underline{w}} P(D|w)$$

MAP:

$$\max_{\underline{w}} P(\underline{w}|D)$$

$$= \max_{\underline{w}} \frac{P(D|\underline{w})\,P(\underline{w})}{P(D)}$$

$$= \max_{\underline{w}} P(D|w)\,P(\underline{w})$$

$$P(D|w) = \prod_{i=1}^{n} P((\underline{x}_i, y_i)|\underline{w}) \quad \Big| \text{ i.i.d. assumption}$$

$$= \prod_{i=1}^{n} \frac{1}{Z_i} \exp\left(-\ell(y_i; f(\underline{x}_i; w))\right) \quad \Big| \text{ modeling choic}$$

$$P(\underline{w}) = \frac{1}{Z_0} \exp(-\lambda \|w\|^2) \quad \text{Gaussian prior} \quad \Big| \quad \text{Interpreting regularization as a Bayesian Prior}$$

$$P(\underline{w}|D) = \frac{1}{\pi Z_0} \exp\left(-\sum_{i=1}^{n} \ell(y_i; f(\underline{x}_i; \underline{w})) - \lambda \|w\|^2\right)$$

maximum-a-posteriori (MAP)

L_1 - regularization

Answer: Let the $\mathbf{x}_i$ and y_i be random vectors for all i. Model the probability distribution as a negative log of the loss function:

$$P((\mathbf{x}_i, y_i)|\mathbf{W}) = \frac{1}{Z}\exp(-l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{15}$$

If the samples are IID, then we can write the probability of the entire dataset as products of sample probabilities

$$P(\mathcal{D}|\mathbf{W}) = \prod_{i=1}^{n} P((\mathbf{x}_i, y_i)|\mathbf{W}) \tag{16}$$

$$P(\mathcal{D}|\mathbf{W}) = \prod_{i=1}^{n} \frac{1}{Z}\exp(-l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{17}$$

A product of exponents is the summation of their powers,

$$P(\mathcal{D}|\mathbf{W}) = \frac{1}{Z}\exp(-\sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i; \mathbf{W})). \tag{18}$$

Denote

$$L(\mathcal{D}; \mathbf{W}) = \sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i; \mathbf{W}). \tag{19}$$

The original optimization problem can be written as:

$$\mathbf{W}^* = \arg \min_{\mathbf{W}} L(\mathcal{D}; \mathbf{W}) + \lambda R(\mathbf{W}) \tag{20}$$

Taking negative exponent on both sides turns the problem into a maximization problem because $\exp(-y)$ is a monotonically decreasing function.

$$\mathbf{W}^* = \arg \max_{\mathbf{W}} \exp(-L(\mathcal{D}; \mathbf{W}))\exp(-\lambda R(\mathbf{W})) \tag{21}$$

The first term is the same as maximizing the likelihood $P(\mathcal{D}|\mathbf{W})$. If we interpret the second term as a prior:

$$P(\mathbf{W}) = \frac{1}{Z'}\exp(-\lambda R(\mathbf{W})), \tag{22}$$

then we can rewrite the original optimization problem as

$$\mathbf{W}^* = \arg \max_{\mathbf{W}} P(\mathcal{D}|\mathbf{W})P(\mathbf{W}) \tag{23}$$

By Bayes theorem $P(\mathcal{D}|\mathbf{W})P(\mathbf{W}) = P(\mathbf{W}|\mathcal{D})P(\mathcal{D})$, hence we can write the optimization problem as maximizing the posterior

$$\mathbf{W}^* = \arg \max_{\mathbf{W}} P(\mathbf{W}|\mathcal{D})P(\mathcal{D}). \tag{24}$$

We can ignore the evidence term $P(\mathcal{D})$, because it is independent of $\mathbf{W}$ the optimization variable. The original problem reduces to maximizing the posterior, hence maximum a posteriori:

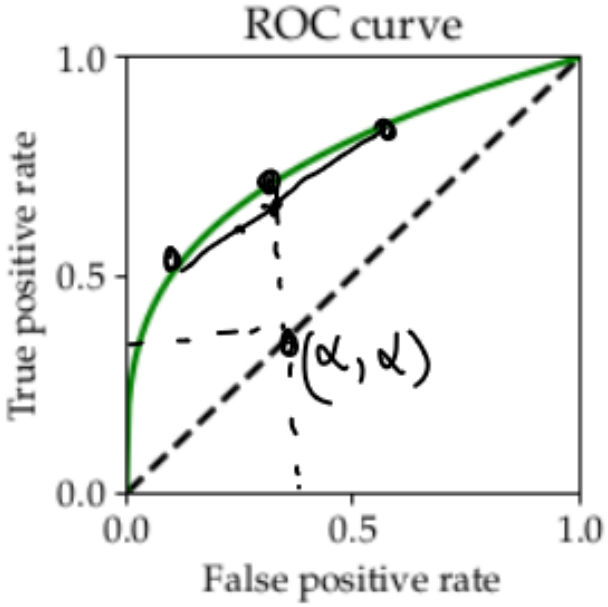
$$\mathbf{W}^* = \arg \max_{\mathbf{W}} P(\mathbf{W}|\mathcal{D}) \tag{25}$$

Question: You are doing 0-1 binary classification on the MNIST hand digit classification task. Let detecting the digit 1 be considered the positive class. Using this example, define the terms False negative, False positive, Accuracy, Precision, recall, F1-score.

Answer:

- False negative: is the case when the predictor classifies the digit as negative i.e. 0 but it is actually 1.
- False positive: is the case when the predictor classifies the digit as positive i.e. 1 but it is actually 0.
- Accuracy: Total correct classifictions/Total samples .
- Precision: True positives / Predicted positives.
- Recall: True positives / All actual positives.

Question: What is a ROC (Receiver Operating characteristics) for a binary classifier? Argue why it lies above the diagonal for any reasonable classifier? Also, argue why it must be concave?



Answer:

ROC curve is plot between False positive rate (FPR) $P(\hat{Y}(X) = 1|Y = 0)$ and True positive rate (TPR) $P(\hat{Y}(X) = 1|Y = 1)$.

ROC curve is always *lies above the diagonal* because you can construct a trivial classifier that predicts the positive class with probability $\alpha \in (0, 1)$, i.e.

$P(\hat{Y}=1)=\alpha$ without even looking at the input data X. This trivial classifier will form the diagonal of the ROC curve with both FPR and TPR being equal to α. Any classifier that works better than this trivial classifier will lie above the diagonal.

Why ROC curve is a concave curve

Suppose you are given hyperparameter values on the ROC curve of a classifier such that:

1. The first classifier 1 has true positive rate (TPR) is $P(\hat{Y}_1(X)=1|Y=1)=TPR(f_1)$ at given FPR $P(\hat{Y}_1(X)=1|Y=0)=f_1$.
2. Similarly second classifier 2 has $FPR=P(\hat{Y}_2(X)=1|Y=0)=f_2$, assume it's TPR is $TPR(f_2)$.

If you pick the classifier f_1 randomly with probability $P(\hat{Y}=\hat{Y}_1)=\alpha$ and the other one with probability $P(\hat{Y}=\hat{Y}_2)=1-\alpha$, and take the output of the picked classifier.

$$P(\hat{Y}(X)=1|Y=1)=P(\hat{Y}(X)=1|\hat{Y}=\hat{Y}_1,Y=1)P(\hat{Y}=\hat{Y}_1)+P(\hat{Y}(X)=1|\hat{Y}=\hat{Y}_2,Y=1)P(\hat{Y}=\hat{Y}_2) \tag{26}$$

$$=P(\hat{Y}(X)=1|\hat{Y}=\hat{Y}_1,Y=1)\alpha+P(\hat{Y}(X)=1|\hat{Y}=\hat{Y}_2,Y=1)(1-\alpha) \tag{27}$$

$$=P(\hat{Y}_1(X)=1|Y=1)\alpha+P(\hat{Y}_2(X)=1|Y=1)(1-\alpha) \tag{28}$$

Then you have FPR of this new classifier as $FPR=P(\hat{Y}_1(X)=1|Y=0)\alpha+P(\hat{Y}_2(X)=1|Y=0)(1-\alpha)=f_1\alpha+f_2(1-\alpha)$. If you get a better TPR than this randomized classifier, the ROC curve will be concave $TPR^*(\alpha f_1+(1-\alpha)f_2)>\alpha TPR(f_1)+(1-\alpha)TPR(f_2)$.

The definition of concavity: A function g is concave if for all points y_1, y_2 and all $\alpha\in(0,1)$ the following condition is true, $g(y_1\alpha+y_2(1-\alpha)>g(y_1)\alpha+g(y_2)(1-\alpha)$.

Question: What is the likelihood ratio test for a binary classifier? Is it better or worse than Posterior ratio test?

Answer: For a binary classifier, likelihood is the probability of observing the evidence $\mathbf{x}$ given a hypothesis class $y\in\{0,1\}$,

$$\text{Likelihood}(\mathbf{x},y)=P(\mathbf{x}|y) \tag{29}$$

Likelihood ratio test gives us a classifier that predicts the positive class $y=1$ if the likelihood ratio exceeds a given threshold η,

$$\hat{y}(\mathbf{x})=\begin{cases}1 & \text{if } \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)}\geq\eta\\ 0 & \text{otherwise}\end{cases} \tag{30}$$

Posterior is the probability of hypothesis class y given an observed evidence $\mathbf{x}$,

$$\text{Posterior}(\mathbf{x}, y) = P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} \tag{31}$$

Let the posterior ratio test be,

$$\hat{y}(\mathbf{x}) = \begin{cases} 1 & \text{if } \frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})} \geq \eta_p \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

Posterior ratio test and likelihood ratio test are equivalent, because the condition for posterior ratio test can be written as likelihood ratio test,

$$\frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})} \geq \eta_p \tag{33}$$

$$\implies \frac{P(\mathbf{x}|y=1)P(y=1)}{P(\mathbf{x}|y=0)P(y=0)} \geq \eta_p \tag{34}$$

$$\implies \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0))} \geq \frac{P(y=0)}{P(y=1)}\eta_p. \tag{35}$$

The last condition is the same condition in the likelihood ratio test with $\eta = \frac{P(y=0)}{P(y=1)}\eta_p$.

1.4 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

Upload the generated zip file to the gradescope autograder

```
# Save your notebook first, then run this cell to export your submission.
grader.export(run_tests=True)
```