CSE 475: Statistical Methods in AI **Monsoon 2019**

SMAI-M-2019 6: Mathematical Foundations of ML - VI

Lecturer: C. V. Jawahar *Date: 19 Aug 2019*

6.31 Problem Space - VI

Recall that $(x_i, y_i), i = 1, ...n$ are independent pairs with the same distribution as (X, Y), and that $f()$ fit on this training set. We will look at the expected test error, conditional on $X = x$ for some arbitrary input x

6.31.1 Bias Variance Tradeoff

The expected test error in, also called the prediction error, for any machine learning algorithm can be broken down into three parts:

1. Bias Error
2. Variance Error
3. Irreducible Error

The third component, we may not be able to help. But we can adjust our problem formulation (say hyper parameters or model complexity) that can give us a solution that has lower error. From a conceptual level, we will see what are these errors at this stage. We will see how they practically appear in subsequent lectures.

We can never predict y_i from x_i with zero prediction error. Note that what we are interested is in predicting for the future. Even if have a magical function that capture the ideal relation underlying X and Y, i.e., the true function, $y_i = f(x_i)$ or $f(X) = E(Y|X)$, we would still incur some error, due to noise in practical situations. We call this as irreducible error. This error can not be reduced regardless of the algorithm we use. We are helpless. It is the error introduced from the modeling/formulation of the problem, and may be the result of factors like unknown variables etc.

What happens if our fitted function $f()$ belongs to a model class that is far from the true function $f()$? E.g., we choose to fit a linear model in a setting where the true relationship is far from linear, say quadratic? We will often refer to this as a wrong (or overcomplex) model. As a result of this design choice, we encounter an error. We call this as an estimation bias. and error due to bias.

What happens if our fitted (random) function $f()$ is itself quite variable? In other words, over different subsets/sampling of the training set, we end up constructing a substantially different functions $f()$? This is also bad. This is another source of error. We will call this estimation variance.

Bias Error

- Low Bias: Suggests less assumptions about the form of the target function.
- High-Bias: Suggests more assumptions about the form of the target function.

Variance Error

- Low Variance: Suggests small changes to the estimate of the target function with changes to the training dataset.
- High Variance: Suggests large changes to the estimate of the target function with changes to the training dataset.

There is a trade off here, but it need it is really never be one-to-one; i.e., in some cases, it can be worth sacrificing a little bit of bias to gain large decrease in variance, and in other cases,vice versa Typical trend: underfitting means high bias and low variance, overfitting means low bias but high variance.

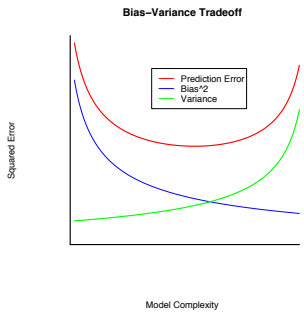

Figure 6.3: The tradeoff of Bias and Variance

6.32 Unifying View Points

Let us now look at the three different view points that we have for the classification.

- Generative view
- Discriminative view
- Distance based view

We start with a simple problem of binary classification with one varible (univariate gaussian with equal variance)

6.32.0.1 Generative/Probabilistic view

We look at the solution as decide class 1 if $P(\omega_1|x) > P(\omega_2|x)$ else class 2.

If we plot the class denisties as Gaussians, with mean as μ_1 and μ_2, we see that this classification makes all samples with $x < \theta$ as class 1 and others class 2. And

$$\theta = \frac{\mu_1 + \mu_2}{2}$$

This is the Bayesian optimal classifier.

6.32.0.2 Discriminative View

We had seen classifiers that say $\mathbf{w}^T\mathbf{x} > \theta$, then class 1 else class 2. Our classifier of $[1][x] > \theta$ is exactly the same.

We see that the simple linear classifier like $\mathbf{w}^T\mathbf{x} > \theta$ is identical to the Bayesian optimal ones under certain settings.

6.32.0.3 Distance bassed view

Assume we look at the classification problem as "classify x to class 1 if distance between x and μ_1 is smaller than that of distance between x and μ_2. This also reduces to the same classifier we discussed above.

In short, all three different looking paradigms yield the same results (under certain assumption).

6.32.1 Multivariate Gaussians

The arguments for the univariate will remain same even if move to Multivariate setting (say with identical covariance).

Q: Do you see a role for Mahalanobis distance?

6.33 Parameter Estimation

Consider the problem of estimating parameters given a data set. Why is this important? Note that we often have only discrete samples and not a distribution to start with. We can make assumption of the type of distribution. But the parameters could be unknown.

Problem Setting There is a training set $\mathcal{D} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_m}, y_m)\}$ of examples sampled i.i.d. according to $p(\mathbf{x}, y)$.

What is IID? This is a popular and valid assumption to make in most situations.

- **Independent:** Each example is sampled independently from the others.
- **Identically Distributed:** All examples are sampled from the same distribution.

Consider the situation when all the samples come from c different samples (like a multi class classification problem). The training set can be divided into $\mathcal{D}_1 \ldots, \mathcal{D}_c$ subsets, one correspond to each of these classes ($\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ contains i.i.d examples for target class y_i)

For any new example $\mathbf{x}$ (not in training set), we compute the posterior probability of the class given the example and the full training set $\mathcal{D}$ as

$$P(y_i|\mathbf{x}, \mathcal{D}) = \frac{p(\mathbf{x}|y_i, \mathcal{D})p(y_i|\mathcal{D})}{p(\mathbf{x}|\mathcal{D})}$$

Note that this is very similar to the Bayesian decision theory (compute posterior probability of class given example) except that parameters of distributions are unknown, and a training set $\mathcal{D}$ is provided instead.

We assume $\mathbf{x}$ is independent of $\mathcal{D}_j (j \neq i)$ given y_i and $\mathcal{D}_i$ with out additional knowledge, $p(y_i|\mathcal{D})$ can be computed as the fraction of examples with that class in the dataset the normalizing factor $p(\mathbf{x}|\mathcal{D})$ can be computed marginalizing $p(\mathbf{x}|y_i, \mathcal{D}_i)p(y_i|\mathcal{D})$ over possible classes.

6.33.1 Estimating θ

We must estimate class-dependent parameters θ_i for $p(\mathbf{x}|y_i, \mathcal{D}_i)$. Assumes parameters θ_i have fixed but unknown values, Values are computed as those maximizing the probability of the observed examples $\mathcal{D}_i$ (the training set for the class). Obtained values are used to compute probability for new examples

$$p(\mathbf{x}|y_i, \mathcal{D}_i) = p(x|\theta_i)$$

Bayesian Parameter Estimation: Assumes parameters θ_i are random variables with some known prior distribution. Observing examples turns prior distribution over parameters into aposteri distribution. Predictions for new examples are obtained integrating over all possible values for the parameters

$$p(\mathbf{x}|y_i, \mathcal{D}_i) = \int_{\theta_i} p(\mathbf{x}, \theta_i|y_i, \mathcal{D}_i)d\theta_i$$

6.33.2 Maximum a-posteriori Estimation

$$\theta_i^* = \arg\max_{\theta_i} p(\theta_i|\mathcal{D}_i, y_i) = \arg\max_{\theta_i} p(\mathcal{D}_i, y_i|\theta_i)p(\theta_i)$$

Assumes a prior distribution for the parameters $p(\theta_i)$ is available

6.33.3 Maximum Liklihood Estimation

$$\theta_i^* = \arg\max_{\theta_i} p(\mathcal{D}_i, y_i|\theta_i)$$

maximizes the likelihood of the parameters with respect to the training samples. Here, no assumption about prior distributions for parameters is made.

Each class y_i is treated independently: replace $y_i, \mathcal{D}_i \rightarrow \mathcal{D}$ for simplicity.

- A training data $\mathcal{D} = \mathbf{x}_1, \ldots, \mathbf{x}_N$ of i.i.d. examples for the target class is available.
- We assume the parameter vector θ has a fixed but unknown value.
- We estimate such value maximizing its likelihood with respect to the training data

$$\theta^* = \arg\max_{\theta} p(\mathcal{D}|\theta) = \arg\max_{\theta_n} \prod_{j=1}^{N} p(x_j|\theta)$$

- The joint probability over $\mathcal{D}$ decomposes into a product as examples are i.i.d (thus independent of each other given the distribution)

Maximum log-likelihood It is usually simpler to maximize the logarithm of the likelihood (monotonic)

$$\theta^* = \arg\max_{\theta} \ln p(\mathcal{D}|\theta) = \arg\max_{\theta} \sum_{j=1}^{n} \ln p(x_j|\theta)$$

Necessary conditions for the maximum can be obtained zeroing the gradient with respect to to θ

$$\nabla_\theta = 0$$

Note zeroing the gradient can be local or global maxima depending on the form of the distribution

6.34 Example: Gaussians

Now we take an example where we want to estimate the parameters of a distribution, given a set of samples. We know the univariate and multivariate Gausiians as: Univariate

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

and multivariate case

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}[x-\mu]^T\Sigma^{-1}[x-\mu]\right]$$

Note that the final expression for the parameter θ (in this case, mean and covariance) may be familiar to you. But the point to note is that, this leaves a Bayesian perspective of how parameters are estimated.

6.34.1 Unknown μ

Let us take a simple case, where covariance Σ is known and only mean μ is unknown.

The log-likelihood is

$$\sum_{j=1}^{n} \ln p(\mathbf{x}_j|\theta) - \sum_{j=1}^{n} \frac{-1}{2}[\mathbf{x}_j-\mu]^T\Sigma^{-1}[\mathbf{x}-\mu]$$

The gradient with respect to the mean is

$$\nabla \sum_{j=1}^{n} \ln p(\mathbf{x}_j|\theta) = \sum_{j=1}^{n} \Sigma^{-1}[\mathbf{x}_j-\mu]$$

Setting the gradient to zero gives

$$\sum_{j=1}^{n} \Sigma^{-1}[\mathbf{x}_j-\mu] = 0 \implies \mu^* = \frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_j$$

6.34.2 Unknown μ and σ^2

The log-likelihood is

$$\mathcal{L} = \sum_{j=1}^{n} -\frac{1}{2\sigma^2}(x_j-\mu)^2 - \frac{1}{2}\ln 2\pi\sigma^2$$

The gradient wrt to μ and σ^2 of L respectively are

$$\sum_{j=1}^{n} \frac{1}{\sigma^2}(x_j-\mu)$$

and

$$\sum_{j=1}^{n}\left(-\frac{1}{2\sigma^2} + \frac{(x_j-\mu)^2}{2\sigma^4}\right)$$

Setting gradients to zero lead to

$$\mu^* = \frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_j$$

and

$$\sum_{j=1}^{n}\frac{1}{2\sigma^2} = \sum_{j=1}^{n}\frac{(x_j-\mu)^2}{2\sigma^4}$$

$$\sum_{j=1}^{n}\sigma^2 = \sum_{j=1}^{n}(x_j-\mu)^2$$

$$\sigma^2 = \frac{1}{n}\sum_{j=1}^{n}(x_j-\mu)^2$$

6.34.3 Unknown μ and Σ

The log-likelihood is:

$$\sum_{j=1}^{n} -\frac{1}{2}[\mathbf{x}_j-\mu]^T\Sigma^{-1}[\mathbf{x}_j-\mu] - \frac{1}{2}\ln(2\pi)^d|\Sigma|$$

The maximum-likelihood estimates are:

$$\mu^* = \frac{1}{n}\sum_{j=1}^{n}\mathbf{x}_j$$

and

$$\Sigma = \frac{1}{n}\sum_{j=1}^{n}[\mathbf{x}_j-\mu^*][\mathbf{x}_j-\mu^*]$$

Maximum likelihood estimates for Gaussian parameters are simply their empirical estimates over the samples:

- Gaussian mean is the sample mean
- Gaussian covariance matrix is the mean of the sample covariances

Q: Verify the above equations/derivations