

Foundation of Machine Learning (CSE4032)

Lecture 02: Overview of Supervised Learning (Part 2)

Dr. Kundan Kumar
Associate Professor
Department of ECE



Faculty of Engineering (ITER)
S'O'A Deemed to be University, Bhubaneswar, India-751030

Outline

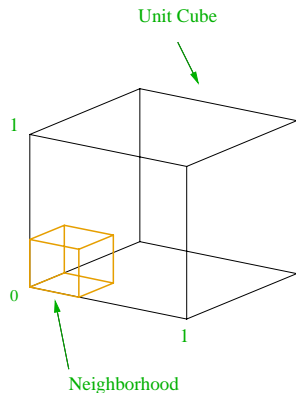
- 1 Introduction
- 2 Local Methods in High Dimensions
- 3 Structured Regression Models
- 4 Prediction Errors
- 5 Model Evaluation
- 6 References

Local Methods in High Dimensions

Introduction

- Two learning techniques for prediction
 - The stable but biased linear model
 - Less stable but apparently less biased class of k -NN estimates
- Curse of dimensionality
 - For reasonably large set of training data, we could always approximate the theoretically optimal conditional expectation by k -nearest-neighbor averaging; however, this approach breaks in high dimensionality called **Curse of Dimensionality**.

Curse of dimensionality



- Consider a p -dimensional unit hypercube.
- Suppose we send out a hypercubical neighborhood about a target point to capture a fraction r of the observations. Then

$$\text{Expected edge length} = e_p(r) = r^{1/p}$$

- In ten dimensions $e_{10}(0.01) = 0.63$ and $e_{10}(0.1) = 0.80$, while the entire range for each input is only 1.0.
- So to capture 1% or 10% of the data to form a local average, we must cover 63% or 80% of the range of each input variable. Such neighborhoods are no longer “local.”

Curse of dimensionality

- Reducing r dramatically does not help much either, since the fewer observations we average, the higher is the variance of our fit.
- Another consequence of the sparse sampling in high dimensions is that all sample points are close to an edge of the sample.
 - Consider N data points uniformly distributed in a p -dimensional unit ball centered at the origin.
 - Suppose we consider a nearest-neighbor estimate at the origin. The median distance from the origin to the closest data point is given by the expression

$$d(p, N) = \left(1 - \frac{1}{2}^{1/N}\right)^{1/p}$$

- A more complicated expression exists for the mean distance to the closest point.

Curse of dimensionality

- For $N = 500$, $p = 10$, $d(p, N) = 0.52$, more than halfway to the boundary. Hence most data points are closer to the boundary of the sample space than to any other data point.
- The reason that this presents a problem is that prediction is much more difficult near the edges of the training sample.
- The sampling density is proportional to $N^{1/p}$, where p is the dimension of the input space and N is the sample size. Thus, if $N_1 = 100$ represents a dense sample for a single input problem, then $N_{10} = 100^{10}$ is the sample size required for the same sampling density with 10 inputs. Thus in high dimensions all feasible training samples sparsely populate the input space.

Example

- Suppose we have 1000 training examples x_i generated uniformly on $[-1, 1]^p$. Assume that the true relationship between X and Y is

$$Y = f(X) = e^{-8\|X\|^2}$$

without any measurement error.

- We use the 1-nearest-neighbor rule to predict y_0 at the test-point $x_0 = 0$. Denote the training set by \mathcal{T} .
- We can compute the expected prediction error at x_0 for our procedure, averaging over all such samples of size 1000.

Example

- Since the problem is deterministic, this is the mean squared error (MSE) for estimating $f(x_0)$:

$$\begin{aligned}\text{MSE}(x_0) &= \mathbb{E}_{\mathcal{T}} [f(x_0) - \hat{y}_0]^2 \\ &= \mathbb{E}_{\mathcal{T}} [\hat{y}_0 - \mathbb{E}_{\mathcal{T}}(\hat{y}_0)]^2 + [\mathbb{E}_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2 \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)\end{aligned}$$

- We have broken down the MSE into two components that will become familiar as we proceed: variance and squared bias.
- Such a decomposition is always possible and often useful, and is known as the **bias–variance decomposition**.

Examples

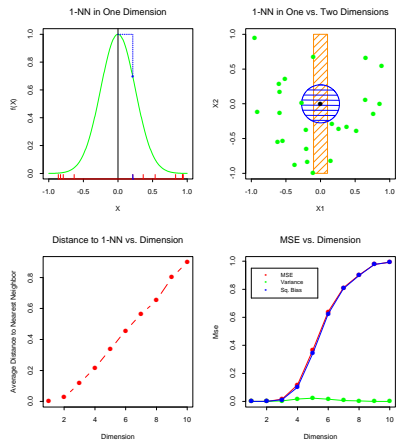


Figure: A simulation example, demonstrating the curse of dimensionality and its effect on MSE, bias and variance.

Structured Regression Models

Structured Regression Models

- We have seen that although nearest-neighbor and other local methods focus directly on estimating the function at a point, they face problems in high dimensions.
- They may also be inappropriate even in low dimensions in cases where more structured approaches can make more efficient use of the data.
- This section introduces classes of such structured approaches. Before we proceed, though, we discuss further the need for such classes.

Difficulty of the Problem

- Consider the RSS criterion for an arbitrary function f ,

$$\text{RSS}(f) = \sum_{i=1}^N (y_i - f(x_i))^2 \quad (1)$$

- Minimizing above equation leads to infinitely many solutions: any function \hat{f} passing through the training points (x_i, y_i) is a solution.
- Any particular solution chosen might be a poor predictor at test points different from the training points.
- If there are multiple observation pairs $x_i, y_{i\ell}$ $\ell = 1, \dots, N_i$ at each value of x_i , the risk is limited. In this case, the solutions pass through the average values of the $y_{i\ell}$ at each x_i .

Structured Regression Models

- In order to obtain useful results for finite N , we must restrict the eligible solutions to Eq. (1) to a smaller set of functions.
- How to decide on the nature of the restrictions is based on considerations outside of the data.
- These restrictions are sometimes encoded via the parametric representation of f , or may be built into the learning method itself, either implicitly or explicitly.
- Any restrictions imposed on f that lead to a unique solution to Eq. (1) do not really remove the ambiguity caused by the multiplicity of solutions.
- There are infinitely many possible restrictions, each leading to a unique solution, so the ambiguity has simply been transferred to the choice of constraint.

Structured Regression Models

- Any method that attempts to produce locally varying functions in small isotropic neighborhoods will run into problems in high dimensions—again the curse of dimensionality.
- And conversely, all methods that overcome the dimensionality problems have an associated—and often implicit or adaptive—metric for measuring neighborhoods.
- Which basically does not allow the neighborhood to be simultaneously small in all directions.

Classes of Restricted Estimators

- The variety of nonparametric regression techniques or learning methods fall into a number of different classes depending on the nature of the restrictions imposed.
- These classes are not distinct, and indeed some methods fall in several classes.
- Each of the classes has associated with it one or more parameters, sometimes appropriately called smoothing parameters, that control the effective size of the local neighborhood.
- Broadly, In this course we will discuss three classes.
 1. Roughness Penalty and Bayesian Methods
 2. Kernel Methods and Local Regression
 3. Basis Functions and Dictionary Methods

Roughness Penalty and Bayesian Methods

- Here the class of functions is controlled by explicitly penalizing $RSS(f)$ with a roughness penalty

$$PRSS(f; \lambda) = RSS(f) + \lambda J(f)$$

- The user-selected functional $J(f)$ will be large for functions f that vary too rapidly over small regions of input space. Penalty function, or regularization methods, express our prior belief that the type of functions we seek exhibit a certain type of smooth behavior, and indeed can usually be cast in a Bayesian framework.

Kernel Methods and Local Regression

- These methods can be thought of as explicitly providing estimates of the regression function of conditional expectation by specifying the nature of the local neighborhood, and of the class of regular functions fitted locally. The local neighborhood is specified by a kernel function $K_\lambda(x_0, x)$ which assigns weights to points x in a region around x_0 . In general we can define a local regression estimate of $f(x_0)$ as $f_{\hat{\theta}}(x_0)$, where $\hat{\theta}$ minimizes

$$RSS(f_\theta, x_0) = \sum_{i=1}^N K_\lambda(x_0, x_i) (y_i - f_\theta(x_i))^2$$

and f_θ is some parameterized function, such as a low-order polynomial.

Basis Functions and Dictionary Methods

- The model for f is a linear expansion of basis functions

$$f_{\theta}(x) = \sum_{m=1}^M \theta_m h_m(x)$$

where each of the h_m is a function of the input x , and the term linear here refers to the action of the parameters θ . Adaptively chosen basis function methods are also known as dictionary methods, where one has available a possibly infinite set or dictionary D of candidate basis functions from which to choose, and models are built up by employing some kind of search mechanism.

Expected prediction error for categorical variable

- Already, We have seen that how we can compute prediction error for quantitative output variables.
- What do we do when the output is a categorical variable G ?
- Our loss function can be represented by a $K \times K$ matrix L , where $K = \text{card}(G)$.
- L will be zero on the diagonal and non-negative elsewhere, where $L(k, l)$ is the price paid for classifying an observation belonging to class G_k as G_l .
- Most often we use the **zero-one loss function**, where all misclassifications are charged a single unit.

Expected prediction error for categorical variable

- With this 0 – 1 loss function the solution that minimizes the expected prediction error is

$$\hat{G}(X) = G_k \text{ if } \Pr(G_k | X = x) = \max_g \Pr(g | X = x)$$

- This reasonable solution is known as the **Bayes classifier**, and says that we classify to the most probable class, using the conditional (discrete) distribution $\Pr(G | X)$.
- The error rate of the Bayes classifier is called the **Bayes rate**.

Confusion Matrix

- For a two class-problem, a table of confusion (sometimes also called a confusion matrix), is a table with two rows and two columns that reports the number of
 - false positives (FP),
 - false negatives (FN),
 - true positives (TP), and
 - true negatives(TN)
- In statistical classification, a confusion matrix, also known as an error matrix.

		True condition	
		Condition positive	Condition negative
Predicted condition	Predicted condition positive	True positive	False positive, Type I error
	Predicted condition negative	False negative, Type II error	True negative

Performance Evaluation using confusion matrix

- True positive rate (TPR), also called Sensitivity
- False positive rate (FPR), also called Fall-out
- False negative rate (FNR), also called Miss rate
- True negative rate (TNR), also called Specificity

True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$
False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$

Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$

Performance Evaluation using confusion matrix

$\text{Prevalence} = \frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	$\text{Accuracy (ACC)} = \frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$	
$\text{Positive predictive value (PPV), Precision} = \frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	$\text{False discovery rate (FDR)} = \frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$	
$\text{False omission rate (FOR)} = \frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	$\text{Negative predictive value (NPV)} = \frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$	
$\text{Positive likelihood ratio (LR+)} = \frac{\text{TPR}}{\text{FPR}}$	$\text{Diagnostic odds ratio (DOR)} = \frac{\text{LR+}}{\text{LR-}}$	$\text{F}_1 \text{ score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
$\text{Negative likelihood ratio (LR-)} = \frac{\text{FNR}}{\text{TNR}}$		

Receiver Operating Characteristics

- If we use a parameter (e.g., a threshold) in our decision, the plot of TPR vs FPR for different values of the parameter is called the **receiver operating characteristic (ROC) curve**.
- The ROC curve is created by plotting the **true positive rate (TPR)** against the **false positive rate (FPR)** at various threshold settings.

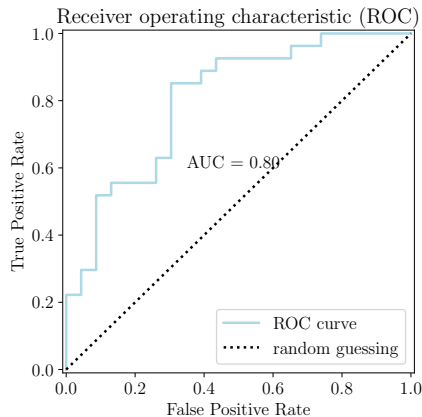
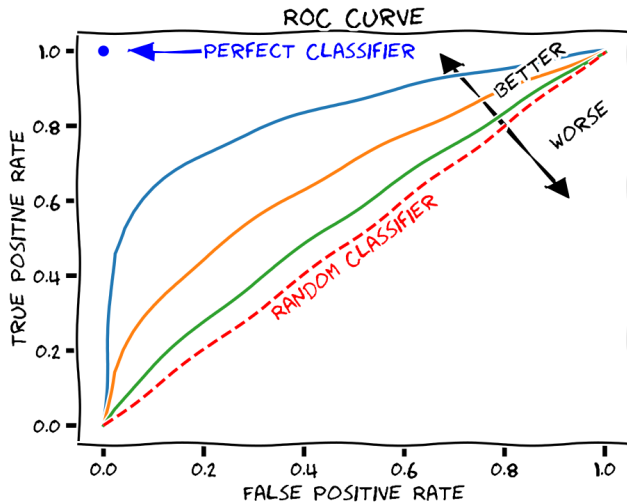
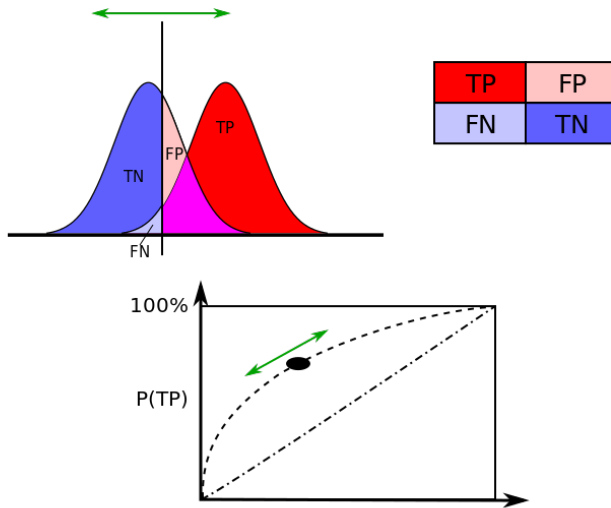


Figure: Example receiver operating characteristic (ROC) curves for different setting of the system

Receiver Operating Characteristics



Receiver Operating Characteristics



Cross-validation

- Cross-validation is a technique to measure predictive performance of a model.
- There are different way to perform this
 - Holdout Sample: Training and Test Data
 - Three-way split: Training, Validation, and Test Data
 - Random subsampling
 - K -fold Cross-Validation
 - Leave-One-Out Cross-Validation

Holdout Sample

- Data is split into two groups: Training set and Test set.
- The training set is used to train the learner.
- The test set is used to estimate the error rate of the trained model.
- Drawbacks
 - In a sparse data set, one may not have the luxury to set aside a reasonable portion of the data for testing.
 - Since it is a single repetition of the train-&-test experiment, the error estimate is not stable.
 - If we happen to have a 'bad' split, the estimate is not reliable.
- A typical split is 80% for the training data and 20% for test set.

Three-way Split

- The available data is partitioned into three sets: **training**, **validation** and **test set**. The prediction model is trained on the training set and is evaluated on the validation set.
- In general, validation set is used to set the model parameter optimal in the training process.
- Training and validation may be iterated a few times till a 'best' model is found. The final model is assessed using the test set.
- A typical split is 50% for the training data and 25% each for validation set and test set.
- With three-way split, the model selection and the true error rate computation can be carried out simultaneously. But error rate computation on a third independent part of the data, the test data, is required.
- **Unfortunately, data insufficiency often does not allow three-way split.**

Random subsampling

- The limitations of the holdout or three-way split can be overcome with a family of resampling methods at the expense of higher computational cost.
- Essentially cross-validation includes techniques to split the sample into multiple training and test data sets.
- Random subsampling performs K data splits of the entire sample. For each data split, a fixed number of observations is chosen without replacement from the sample and kept aside as the test data.
- The prediction model is fitted to the training data from scratch for each of the K splits and an estimate of prediction error is obtained from each test set.
- Let the estimated PE in i th test set be denoted by E_i .

$$\text{True error estimate} = \frac{1}{K} \sum_{i=1}^K E_i$$

K -fold Cross-Validation

- The original sample is randomly partitioned into K equal sized (or almost equal sized) subsamples.
- Of the K subsamples, a single subsample is retained as the test set for estimating the PE , and the remaining $K - 1$ subsamples are used as training data.
- The cross-validation process is then repeated K times (the folds), with each of the K subsamples used exactly once as the test set. The K error estimates from the folds can then be averaged to produce a single estimation.
- The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

K -fold Cross-Validation

- A common choice for K is 5 or 10.
- With a large number of folds (K large) the bias of the true error rate estimator is small but the variance will be large.
- The computational time may also be very large as well, depending on the complexity of the models under consideration.
- With a small number of folds the variance of the estimator will be small but the bias will be large. The estimate may be larger than the true error rate.
- In practice the choice of the number of folds depends on the size of the data set. For large data set, smaller K (e.g. 3) may yield quite accurate results. For sparse data sets, Leave-one-out (LOO or LOOCV) may need to be used.

Leave-One-Out Cross-Validation

- LOO is the degenerate case of K -fold cross-validation where $K = n$ for a sample of size n .
- That means n separate times, the prediction function is trained on all the data except for one point and a prediction is made for that point.
- As before the average error is computed and used to evaluate the model.
- The evaluation given by leave-one-out cross validation error is good, but sometimes it may be very expensive to compute.

Further Reading Recommendation



- Explore the following articles

<https://heartbeat.fritz.ai/>

[introduction-to-machine-learning-model-evaluation-fa859e1b2d7f](https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f)

https://www.saedsayad.com/model_evaluation.htm

References

-  The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition, Hastie, Tibshirani, and Friedman, Springer.
-  In Introduction to Statistical Learning with Application in R, Second Edition, James, Witten, Hastie, and Tibshirani, Springer.



Thank you!