

IEMS 304 Lecture 5: Non-linear and Non-parametric Regression

Yiping Lu

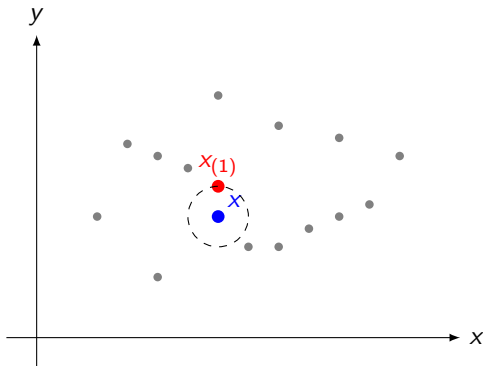
yiping.lu@northwestern.edu

*Industrial Engineering & Management Sciences
Northwestern University*



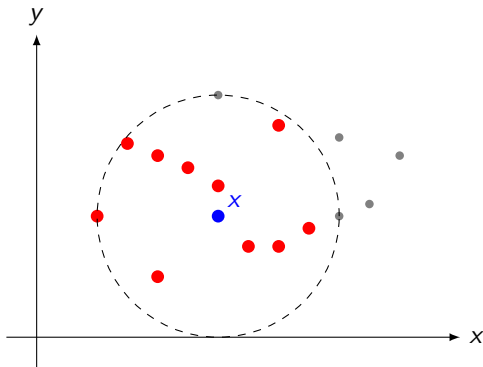
k-NN Regression ($k = 1$)

$$\hat{f}(x) = y_{(1)}$$



k-NN Regression ($k = 10$)

$$\hat{f}(x) = \frac{1}{10} \sum_{i=1}^{10} y_{(i)}$$



“A precise and universally acceptable definition of the term ‘nonparametric’ is not presently available. The viewpoint adopted in this handbook is that a statistical procedure is of a nonparametric type if it has properties which are satisfied to a reasonable approximation when some assumptions that are at least of a moderately general nature hold.”

– The Handbook of Nonparametric Statistics

Bias and Variance Trade-off

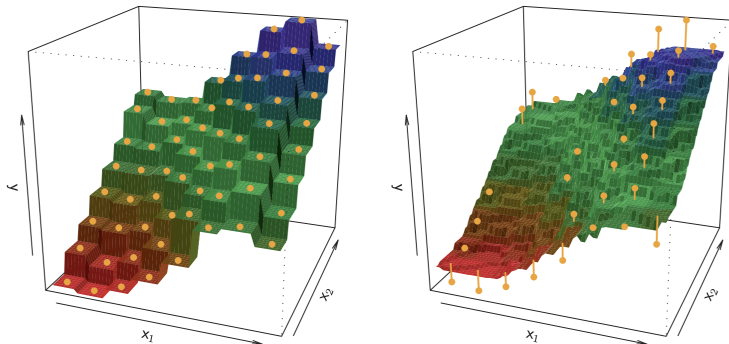
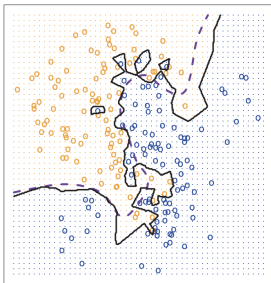


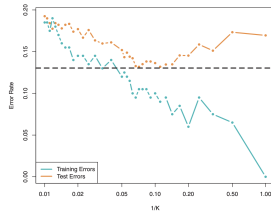
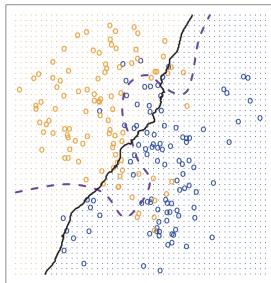
FIGURE 3.16. Plots of $\hat{f}(X)$ using KNN regression on a two-dimensional data set with 64 observations (orange dots). Left: $K = 1$ results in a rough step function fit. Right: $K = 9$ produces a much smoother fit.

Bias and Variance Trade-off

KNN: $K=1$

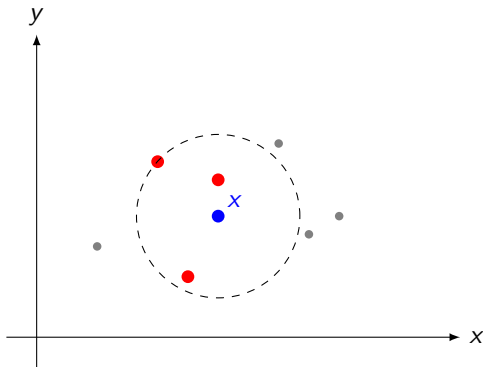


KNN: $K=100$



k-NN Regression with Limited Data (k=3)

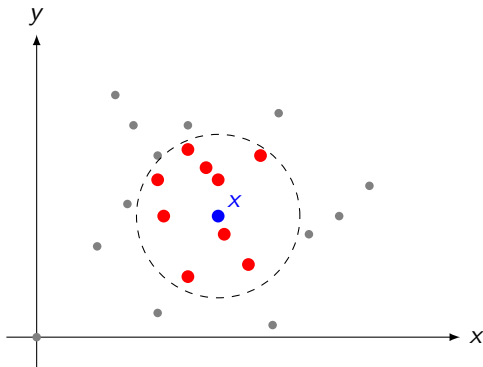
$$\hat{f}(x) = \frac{1}{3} \sum_{i=1}^3 y_{(i)}$$



k-NN Regression with More Data

Use the same size of neighborhood, now we have 10 data in the circle

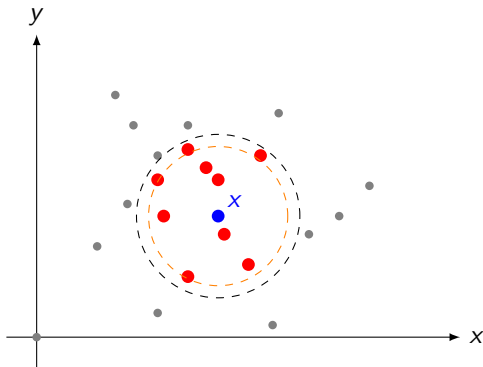
- ❑ How is bias changing? How is variance changing?
- ❑ How should we do bias-variance trade-off?



k-NN Regression with More Data

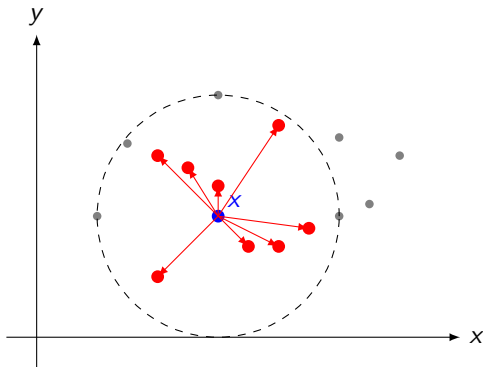
Use the same size of neighborhood, now we have 10 data in the circle

- ❑ How is bias changing? How is variance changing?
- ❑ How should we do bias-variance trade-off?

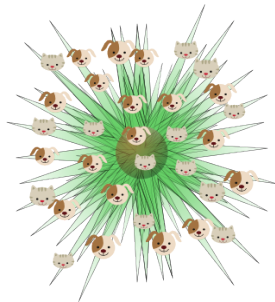
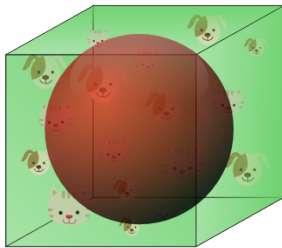
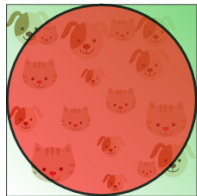


Local Kernel Smoothing: Nadaraya-Watson Estimator

$$\hat{f}(x) = \frac{\sum_{i=1}^n K_h(x - x_i) y_i}{\sum_{i=1}^n K_h(x - x_i)}$$



Curse of Dimensionality



Nonlinear Regression Models

Nonlinear Regression Model

A general form of nonlinear regression model is $Y_i = g(x_i; \beta) + \epsilon_i$, where

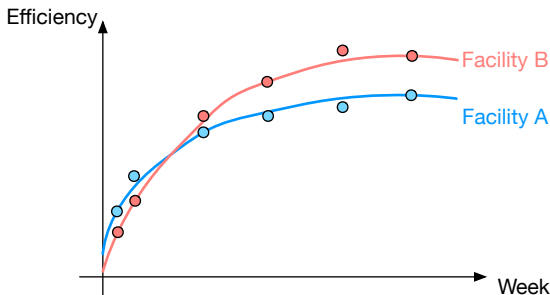
- Y_i : response for observation i ;
- x_i : vector of predictors for observation i ;
- β : vector of model parameters;
- $g(x_i; \beta)$: some parametric nonlinear function;
- ϵ_i : zero-mean random error for observation i .

We will see shortly that if the random errors are **Gaussian** and independent of x , the MLE of β is just nonlinear least squares.

Example of Manufacturing Learning Curve

- Two facilities operate with (different) efficiency as a function of time.
- We denote Y as the relative efficiency of operation. The predictor variables are

- $x_1 = \begin{cases} 1, & \text{facility B (modern)} \\ 0, & \text{facility A (old)} \end{cases};$
- $x_2 = \text{number of weeks.}$



Questions and Discussions

- For facility A, and the data looked like in the previous slide, how would you model it?
- Facilities A and B have different asymptotic efficiencies, how would you modify the model?
- If facilities A and B have different learning rates, how would you modify the model?
- If the objective was to determine if the two facilities have different asymptotic efficiencies, how could you do this?

Hint: Play with the model $Y = \beta_0 + \beta_3 \exp(\beta_2 x_2) + \epsilon$.

MLE for General Nonlinear Regression Model

Nonlinear model $Y_i = \underbrace{g(x_i; \beta)}_{:=\mu_i} + \epsilon_i$ with $\epsilon_i \sim N(0, \sigma^2)$.

Now we view x_i as deterministic, not random.

- Accordingly, the nonlinear model becomes $Y_i = \mu_i + \epsilon_i$.
- Marginal pdf of Y_i is $f(y_i; \beta, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(y_i - \mu_i)^2\right)$.

What is the Max-likelihood Estimator?

Maximizing Likelihood Function

Joint pdf (a.k.a. the likelihood function) of $Y_1, \dots, Y_n$ is

$$f(y; \beta, \sigma) = \frac{1}{(2\pi)^{n/2} \sigma^n} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu_i)^2\right).$$

We want to $\max_{\beta, \sigma} f(y; \beta, \sigma) = \max_{\beta, \sigma} \frac{1}{\sigma^n} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu_i)^2\right)$.

Some inspection suggests that for β , it suffices to

$$\min_{\beta} \sum_{i=1}^n (y_i - \mu_i)^2 = \min_{\beta} \sum_{i=1}^n (y_i - \mu_i)^2. \quad \text{log-likelihood}$$

That is, the MLE of β for the general nonlinear regression model with i.i.d. Gaussian errors (that are independent of x) is “nonlinear least squares”.

How to compute β ? Optimization!

Summary of Steps in General MLE

- ❑ Write out the form of the statistical model that you are using to represent the data.
- ❑ Find the marginal distribution of each individual observation Y_i (for regression problems the x_i 's are not treated as random, so you only need to find the marginal distribution of the Y_i 's given the x_i 's).
- ❑ From the marginal distributions in step (2), find the joint distribution $f(Y; \theta)$ of the entire set of data Y . Here θ denotes all the parameters.

If tractable, find an analytical expression for the θ that maximizes the likelihood $f(Y; \theta)$. Otherwise, use numerical optimization software to minimize $-\log f(Y; \theta)$.

- R has several built-in commands for nonlinear regression such as `nlm` and `nls` (a little buggier than `nlm`).
- For the manufacturing learning curve example, we read data in `MLC.csv`.
- The following code snippet is for nonlinear regression on `MLC.csv`.

```
MLC<-read.table("MLC.csv",sep=";",header=TRUE)
x1<-MLC$Location;x2<-MLC$Week;y<-MLC$Efficiency
fn <- function(p) {yhat<-p[1]+p[2]*x1+p[4]*exp(p[3]*x2); sum((y-yhat)^2)}
out<-nlm(fn,p=c(1,0,-.5,-.1),hessian=TRUE)
theta<-out$estimate #parameter estimates
```

Example: Gaussian Distribution with Learned Variance

The likelihood function of a Gaussian distribution is given by:

$$P(y_i | \mu(x_i), \sigma(x_i)^2) = \frac{1}{\sqrt{2\pi \sigma(x_i)^2}} \exp\left(-\frac{(y_i - \mu(x_i))^2}{2 \sigma(x_i)^2}\right)$$

$$\begin{aligned}\ell(\mu, \sigma^2) &= \sum_{i=1}^n \log P(y_i | \mu(x_i), \sigma(x_i)^2) \\&= \sum_{i=1}^n \left(-\frac{1}{2} \log(2\pi) - \frac{1}{2} \log(\sigma(x_i)^2) - \frac{(y_i - \mu(x_i))^2}{2\sigma(x_i)^2} \right) \\&= -\frac{n}{2} \ln(2\pi(x_i)) - \underbrace{\frac{n}{2} \ln(\sigma(x_i)^2)}_{\text{sparse regularization}} - \underbrace{\sum_{i=1}^n \frac{(y_i - \mu(x_i))^2}{2\sigma(x_i)^2}}_{\text{weighted } \ell_2 \text{ loss}}\end{aligned}$$

Another Example: Weibull Distribution

The likelihood function of a Weibull distribution is given by:

$$p_k(x|\lambda) = \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}$$

, where $1 > k > 0$ is the shape parameter and $\lambda > 0$ is the scale parameter.

$$\log p_k(y|\lambda(x)) = -(y/\lambda(x))^k - k \log \lambda(x) + \underbrace{\log(ky^{k-1})}_{\text{not dependent on the prediction } \lambda(x)}$$

Fact. $f(y, \lambda)$ attains its minimum at $\lambda = y$.

Non-parametric Statistical Inference

Statistical Uncertainty in Supervised Learning

- With nonlinear regression models, the formulae for assessing statistical uncertainty in linear regression (e.g., F -tests and t -tests for significance of predictors, SEs and CIs for parameters, PIs and CIs for new observations, etc.) do not apply directly.
 - Question: Why might we want to calculate SEs, CIs/PIs, do hypothesis tests, etc?
- For some nonlinear models, we can use approximate **asymptotic analytical results** valid for sufficiently large sample size n to assess statistical uncertainty.
- Fortunately, we have alternative **computational approaches** that apply to any nonlinear model:
 - **Cross-validation** for deciding which models are the best.
 - **Bootstrap resampling** (or **bootstrapping** for short) for SEs and CIs on the parameters and CIs and PIs on new observations.

Overview of Bootstrapping

Objective: Estimate the sampling distribution of $\hat{\theta}$ and quantities like $SE(\hat{\theta})$ that are derived from it.

- ❑ You are given a sample of data of size n observations.
- ❑ You have estimated some parameter(s) θ (call it $\hat{\theta}$).

Problem: Hypothetically, if we knew the form of the population distribution, we could consider using simulation to draw many random samples (each of size n) from the population and calculate a different $\hat{\theta}$ for each sample. We could construct a histogram of all the $\hat{\theta}$'s and take their sample standard deviation to be an estimate of $SE(\hat{\theta})$. But what if we do not know the form of the population distribution?

Illustration of Sampling from Known Distribution

AIM. estimate the mean of a Gaussian distribution and want to know the SE of the estimate.

- ❑ Generate say 10,000 samples, each of size $n = 20$, from an $N(5.3, 0.4^2)$ distribution.
- ❑ Calculate the averages $\{\bar{y}_{\text{sim}}^{(j)} : j = 1, \dots, 10000\}$ for the 10000 replicates.
- ❑ Take

$$\text{SE}(\bar{y}) \approx \sqrt{\frac{1}{10000 - 1} \sum_{j=1}^{10000} (\bar{y}_{\text{sim}}^{(j)} - \bar{y}_{\text{sim}})^2},$$

where $\bar{y}_{\text{sim}}$ is the average of $\bar{y}_{\text{sim}}^{(j)}$.

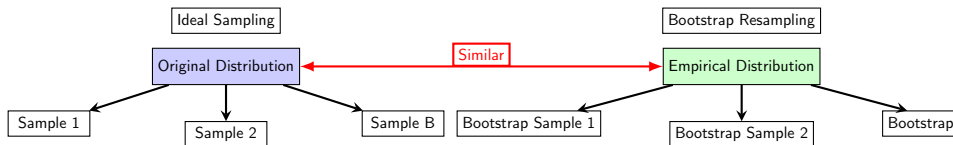
Idea: Bootstrap Sampling

However, Step:

Generate say 10,000 samples, each of size $n = 20$, from $\underbrace{N(5.3, 0.4^2)}_{\text{population}}$ is

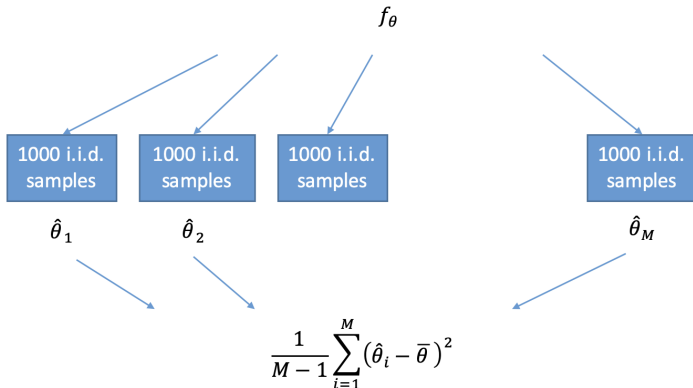
impossible!

Idea. Bootstrap Sampling

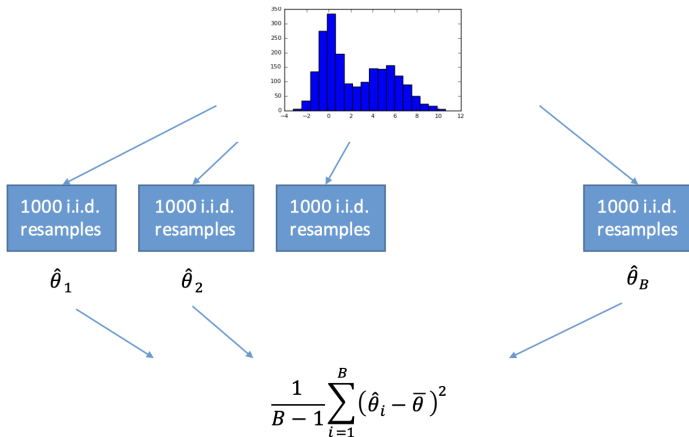


Idea:

If we know f_θ , we can generate new samples to recompute the statistic, and take the sample variance of these estimators



Idea: use the observed samples $z_1, \dots, z_n$ to generate n “new” samples, as if they come from



Bootstrapping Overview Cont'd

- ❑ The **bootstrap sampling approach**: Draw a “bootstrap” sample as a random sample of the same size n from the original sample of n observations (with replacement), and calculate a $\hat{\theta}$ for the bootstrap sample.
- ❑ Repeat a large number of times, each time drawing another bootstrap sample (of size n) and calculating another $\hat{\theta}$ for that sample.
- ❑ Then construct a histogram of all the $\hat{\theta}$'s, take their sample standard deviation to be an estimate of $SE(\hat{\theta})$, etc.

Why this works: Consider making a pretend population that consists of your original sample of n observations, copied over and over, an infinite number of times. Each bootstrap sample is equivalent to drawing a random sample of size n from this infinite pretend population.

Illustration of Bootstrapping

AIM. estimate the mean of an unknown distribution and want to know the SE of the estimate.

- ❑ Generate say 10,000 samples, each of size $n = 20$, from the given **observed data** (with replacement).
- ❑ Calculate the averages $\{\bar{y}^{(b)} : b = 1, \dots, 10000\}$ for the 10000 replicates. (We think of $\bar{y}^{(b)}$ just as the estimator $\hat{\theta}$.)
- ❑ Take

$$\text{SE}(\bar{y}) \approx \sqrt{\frac{1}{10000 - 1} \sum_{j=1}^{10000} (\bar{y}^{(b)} - \bar{y})^2},$$

where $\bar{y}$ is the average of $\bar{y}^{(b)}$.

Bootstrapping in Nonlinear Regression

- ❑ We have a sample of n observations $\{(y_i, x_i)\}_{i=1}^n$ of a response variable and a set of predictor variables.
- ❑ We fit a nonlinear regression model to the data to estimate a set of parameters θ .
- ❑ Let θ denote one of the parameters of interest and $\hat{\theta}$ its estimate.

Objective: Estimate the sampling distribution of $\hat{\theta}$, its standard error, a confidence interval for θ , etc.

Steps of the Bootstrap Procedure

- ❑ Generate a “bootstrap” sample (with replacement) of n observations from $\{(y_i, x_i)\}_{i=1}^n$. Denote the bootstrap sample by

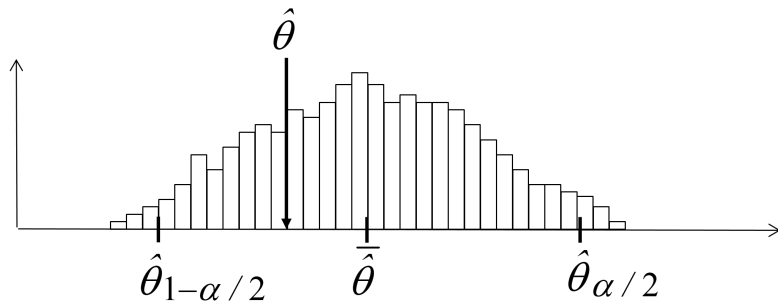
$$\{(y_i^{(b)}, x_i^{(b)})\}_{i=1}^n.$$

- ❑ Fit the same type of regression model (with the same set of parameters θ and parameter θ of special interest) to the bootstrapped sample. Denote the estimates for the bootstrapped sample by $\hat{\theta}^{(b)}$ and $\hat{\theta}^{(b)}$.
- ❑ Pick a large number B (e.g., $B = 10,000$), and repeat Steps (1) and (2) a total of B times, which produces

$$\{\hat{\theta}^{(b)}\}_{b=1}^B.$$

Steps of the Bootstrap Procedure Cont'd

- Construct a histogram of $\{\hat{\theta}^{(b)}\}_{b=1}^B$ and calculate
- $\bar{\hat{\theta}} = \frac{1}{B} \sum_{b=1}^B \hat{\theta}^{(b)}$: average of all bootstrapped estimates.
 - $SE(\hat{\theta}) = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{\theta}^{(b)} - \bar{\hat{\theta}})^2}$: standard error of $\hat{\theta}$.
 - $\hat{\theta}_{\alpha/2}$: upper $\alpha/2$ quantile.
 - $\hat{\theta}_{1-\alpha/2}$: lower $\alpha/2$ quantile.



- ❑ A crude $1 - \alpha$ confidence interval for θ is

$$\hat{\theta} - z_{\alpha/2} \cdot \text{SE}(\hat{\theta}) \leq \theta \leq \hat{\theta} + z_{\alpha/2} \cdot \text{SE}(\hat{\theta}).$$

- ❑ A better $1 - \alpha$ confidence interval for θ is

$$\hat{\theta} - (\hat{\theta}_{\alpha/2} - \hat{\theta}) \leq \theta \leq \hat{\theta} + (\hat{\theta} - \hat{\theta}_{1-\alpha/2}).$$

Conformal Prediction

AIM.

- ❑ Finite-sample coverage guarantees without distributional assumptions
- ❑ Converting a point prediction algorithm into a prediction set
 - ! **Input:** i.i.d. data pairs (X_i, Y_i) for $i = 1, \dots, n$
 - ! **Objective:** Construct a prediction band $\hat{C}_n(x)$ such that

$$P(Y_{n+1} \in \hat{C}_n(X_{n+1})) \geq 1 - \alpha$$

Note: Trivial solutions (Why?) exist, but the goal is to develop nontrivial, adaptive methods

Key Idea: Using Ranks and Quantiles

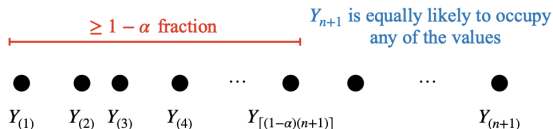
Observation. the rank of Y_{n+1} is uniformly distributed over the values $1, 2, \dots, n+1$. This means that

$$P\left(Y_{n+1} \text{ is among the } [(1-\alpha)(n+1)] \text{ smallest of } Y_1, \dots, Y_n\right) = 1-\alpha,$$

which is in turn equivalent to¹

$$P\left(Y_{n+1} \text{ is among the } (1-\alpha)(n+1) \text{ smallest of } Y_1, \dots, Y_n\right) \geq 1-\alpha.$$

Accordingly, by defining $q_n =$ the $[(1-\alpha)(n+1)]$ -th smallest of $Y_1, \dots, Y_n$, we have precisely achieved the desired property. via $Y_{n+1} \leq$ the $[(1-\alpha)(n+1)]$ -th order statistic of $Y_1, \dots, Y_n$.



Full Conformal Prediction

We have i.i.d. pairs $\{(X_t, Y_t)\}_{t=1}^n$, where $X_t \in \mathcal{X}$ and $Y_t \in \mathcal{Y}$. We want to construct a prediction set for Y_{n+1} given X_{n+1} . Let $\hat{f}_n$ be any regression predictor trained on

$$(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n).$$

Our goal is to achieve $(1 - \alpha)$ coverage, i.e.,

$$P(Y_{n+1} \in C_n(X_{n+1})) \geq 1 - \alpha.$$

Why the Naive procedure **Fails**?

- ❑ Compute the *training residuals* $\hat{g}_i = Y_i - \hat{f}_n(X_i)$, $i = 1, 2, \dots, n$.
- ❑ Let $\hat{q}_n$ be an estimate of a suitable quantile of the absolute residuals, for example the $(1 - \alpha)$ empirical quantile of

$$\{|\hat{g}_1|, |\hat{g}_2|, \dots, |\hat{g}_n|\}.$$

- ❑ Define the prediction set for a new point x as

$$C_n(x) = \left[\hat{f}_n(x) - \hat{q}_n, \hat{f}_n(x) + \hat{q}_n \right].$$

Full Conformal Prediction

We have i.i.d. pairs $\{(X_t, Y_t)\}_{t=1}^n$, where $X_t \in \mathcal{X}$ and $Y_t \in \mathcal{Y}$. We want to construct a prediction set for Y_{n+1} given X_{n+1} . Let $\hat{f}_n$ be any regression predictor trained on

$$(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n).$$

Our goal is to achieve $(1 - \alpha)$ coverage, i.e.,

$$P(Y_{n+1} \in C_n(X_{n+1})) \geq 1 - \alpha.$$

Full Conformal Prediction

- ❑ Compute the *training residuals* $\hat{g}_i = Y_i - \hat{f}_{-i,n}(X_i)$, $i = 1, 2, \dots, n$.
($-i$ means **delete i -th data** while training)
- ❑ Let $\hat{q}_n$ be an estimate of a suitable quantile of the absolute residuals, for example the $(1 - \alpha)$ empirical quantile of $\{|\hat{g}_1|, |\hat{g}_2|, \dots, |\hat{g}_n|\}$.
- ❑ Define the prediction set for a new point x as

$$C_n(x) = \left[\hat{f}_n(x) - \hat{q}_n, \hat{f}_n(x) + \hat{q}_n \right].$$

Split Conformal Prediction

Full Conformal Prediction is computationally intractable! (why?)

Key Idea. Data Split

- ❑ **Proper Training Set (D_1):** Fit the point predictor $\hat{f}_{n_1}(x)$
- ❑ **Calibration Set (D_2):** Compute residuals

$$R_i = |Y_i - \hat{f}_{n_1}(X_i)|, \quad i \in D_2$$

-
- Define quantile from calibration residuals:

$$q_{n_2} = \lceil (1 - \alpha)(n_2 + 1) \rceil\text{-th smallest residual}$$

- Prediction set:

$$\hat{C}_n(x) = [\hat{f}_{n_1}(x) - q_{n_2}, \hat{f}_{n_1}(x) + q_{n_2}]$$

- Guarantee: Ensures marginal coverage of at least $1 - \alpha$

Nonconformity Score

For a predictive model $\hat{f}$ and calibration data $\{(x_i, y_i)\}_{i=1}^{n_{\text{cal}}}$, define the nonconformity score as:

$$\alpha_i = |y_i - \hat{f}(x_i)|$$

Prediction Interval

Let $\hat{q}_{1-\alpha}$ be the $(1 - \alpha)$ -quantile of $\{\alpha_i\}_{i=1}^{n_{\text{cal}}}$. For a new input x_{n+1} , the prediction interval is given by:

$$\{y \in \mathbb{R} : |y - \hat{f}(x_{n+1})| \leq \hat{q}_{1-\alpha}\}$$

This interval guarantees that the true y falls inside with probability at least $1 - \alpha$.

Nonconformity Score

For a classification model, a common choice is:

$$\alpha_i = 1 - p(y_i | x_i)$$

where $p(y_i | x_i)$ is the predicted probability for the true class.

Prediction Set

For a new example x_{n+1} , the prediction set is defined as:

$$\Gamma(x_{n+1}) = \left\{ y \in \mathcal{Y} : \frac{\#\{i : \alpha_i \geq \alpha(y)\} + 1}{n_{\text{cal}} + 1} > \alpha \right\}$$

where $\alpha(y)$ is the nonconformity score computed if y were the true label.

Advantages and Limitations

Advantages

- **Finite-Sample Guarantees:** Ensures valid coverage without asymptotic approximations.
 - **Model-Agnostic:** Can be applied on top of any predictive model.
-

Limitations

- **Computational Cost:** Some methods can be computationally intensive, especially in the transductive setting.
- **Loose Confidence Interval**
- **Assumptions:** Relies on the exchangeability assumption which might not hold in all cases.