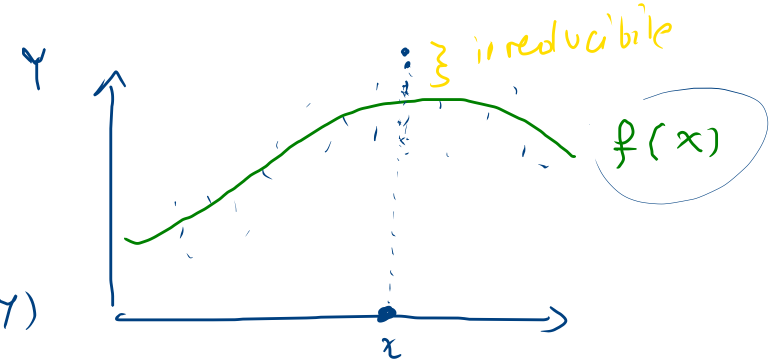
Problem 2

$$Y = f(X) + \varepsilon$$

$\varepsilon \sim N(0, \sigma^2)$ $\quad \underbrace{(x_1, y_1), \dots, (x_n, y_n)}_{\text{training}} \perp\!\!\!\perp \underset{\text{test}}{(X, Y)}$



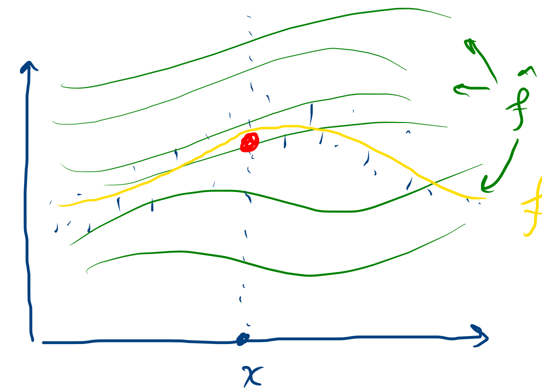
$$E[(Y - \hat{f}(X))^2 \mid X = x]$$

$$= E[(f(X) + \varepsilon - \hat{f}(X))^2 \mid X = x]$$

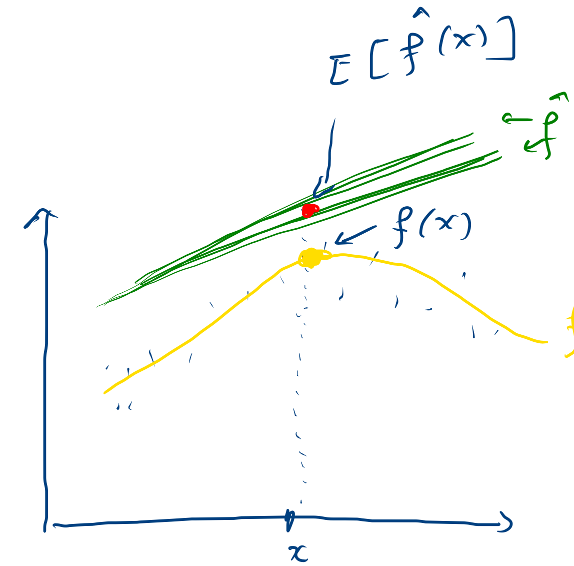
$$= E[(f(x) + \varepsilon - \hat{f}(x))^2]$$

$$= \underbrace{E[\varepsilon^2]}_{A} + \underbrace{E[(f(x) - \hat{f}(x))^2]}_{B}$$

$$+ \underbrace{2\, E[(f(x) - \hat{f}(x)) \cdot \varepsilon]}_{C} = A + B + C$$



hi variance
low bias



hi bias
low variance

$A = E[\varepsilon^2]$ irreducible error

$$C = 2 \cdot E[(f(x) - \hat{f}(x)) \cdot \varepsilon] = 2 \cdot E[f(x) - \hat{f}(x)]\, \overset{0}{\overset{\shortparallel}{E[\varepsilon]}} = 0$$

fixed

$(X_1, Y_1), \dots, (X_n, Y_n) \perp\!\!\!\perp (X, Y)$

TRAIN. $\left[\begin{array}{l} Y_1 = f(X_1) + \varepsilon_1 \\ Y_2 = f(X_2) + \varepsilon_2 \\ \vdots \\ Y_n = f(X_n) + \varepsilon_n \end{array}\right\} \to \hat{f}$

TEST $[\, Y = f(X) + \varepsilon$

Recall: $W \perp\!\!\!\perp Z \Rightarrow E[W \cdot Z] = E[W] \cdot E[Z]$

$$B = E[(f(x) - \hat{f}(x))^2] = E[(f(x) - E[\hat{f}(x)] + E[\hat{f}(x)] - \hat{f}(x))^2]$$

$$= E[(\underset{\text{fixed}}{f(x)} - \underbrace{E[\hat{f}(x)]}_{\text{fixed}})^2] + E[(\underset{\text{random}}{\hat{f}(x)} - \underbrace{E[\hat{f}(x)]}_{\text{fixed}})^2]$$

$$+ 2 \cdot E\{(\underset{\text{fixed}}{f(x)} - \underbrace{E[\hat{f}(x)]}_{\text{fixed}})(E[\hat{f}(x)] - \hat{f}(x))\}$$

$$= \underbrace{(f(x) - E[\hat{f}(x)])^2}_{\text{bias}^2} + \underbrace{E[(\hat{f}(x) - E[\hat{f}(x)])^2]}_{\text{variance}}$$

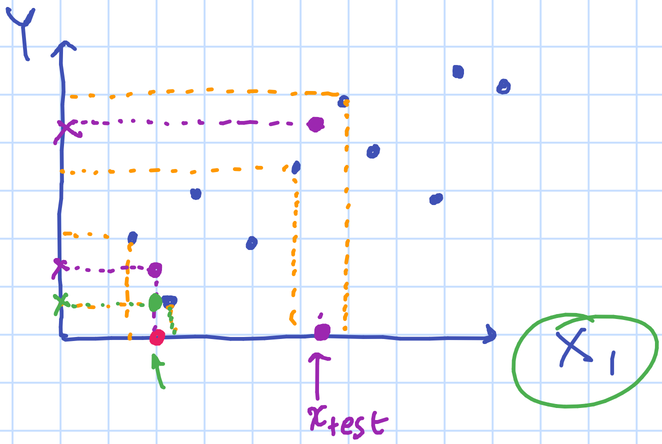
$$+ 2(f(x) - E[\hat{f}(x)]) \cdot \underbrace{E[E[\hat{f}(x)] - \hat{f}(x)]}_{E[\hat{f}(x)] - E[\hat{f}(x)] = 0}$$

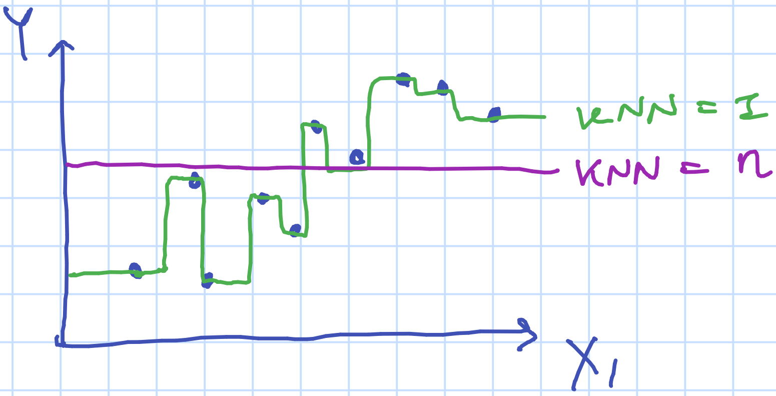
Recall: W with $E[W]$

$$Var(W) = E[(W - E[W])^2]$$

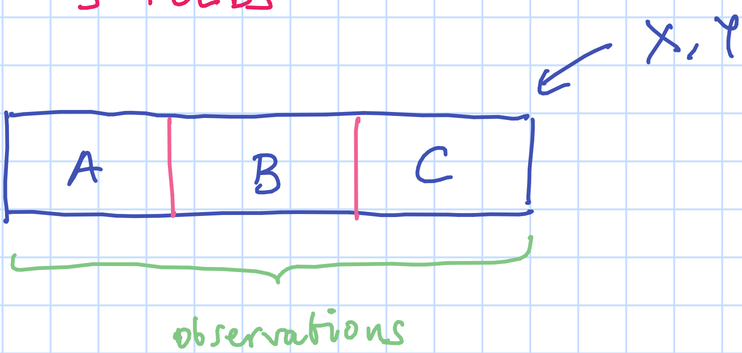
PRACTICAL 3

KNN - REGRESSION





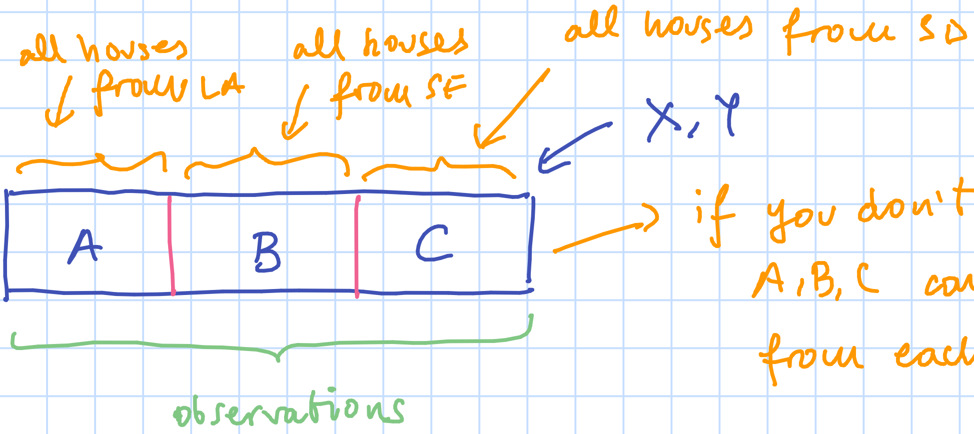
CV - 3 FOLDS



1. knn= KNNeigh (n_neigh=1); knn.fit (B, C) ; predict (A) ; mse (A)

2. knn= KNNeigh (n_neigh=1); knn.fit (A, C) ; predict(B); mse (B)

3. knn= KNNeigh (n_neigh=1); knn.fit (A, B) ; predict(C); mse (C)



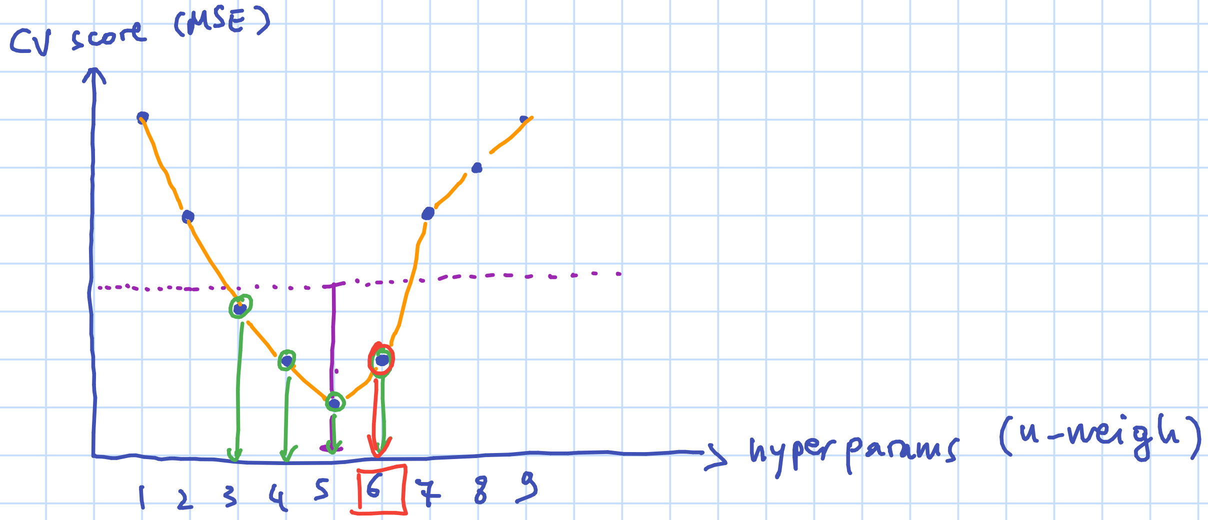
if you don't shuffle, A, B, C can be very different from each other

1. knn= KNNeigh(n_neigh=1); knn.fit(B, C); predict(A); mse(A) LA

2. knn= KNNeigh(n_neigh=1); knn.fit(A, C); predict(B); mse(B) SF

3. knn= KNNeigh(n_neigh=1); knn.fit(A, B); predict(C); mse(C) SD

1-SE RULE



choose hyperparam corresponding to most parsimonious model