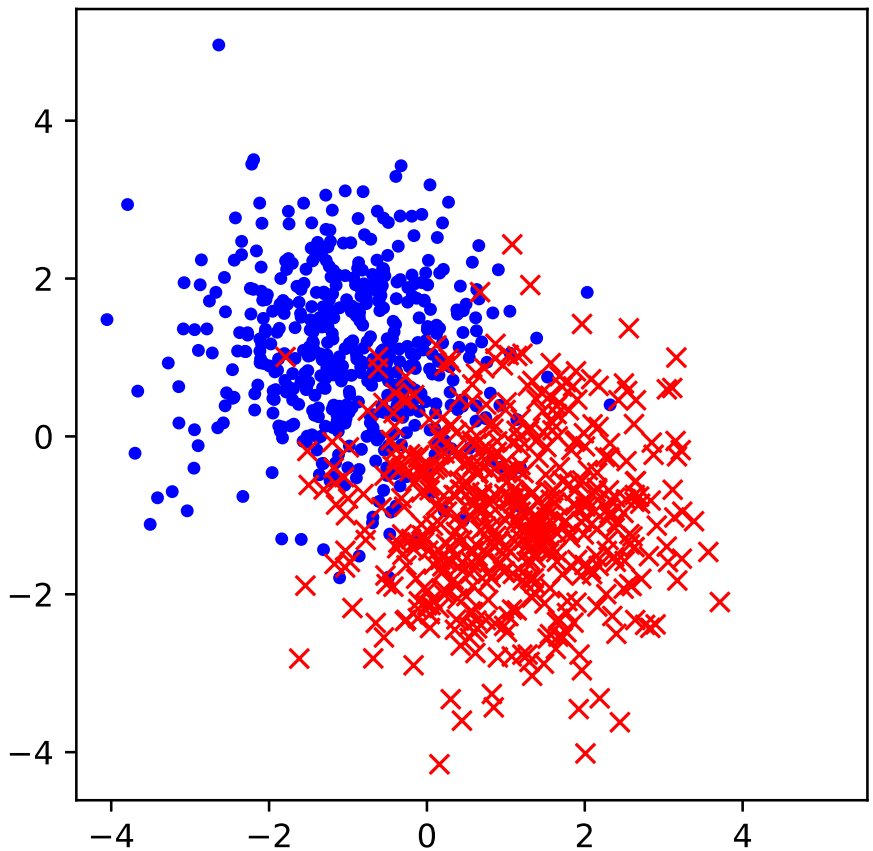
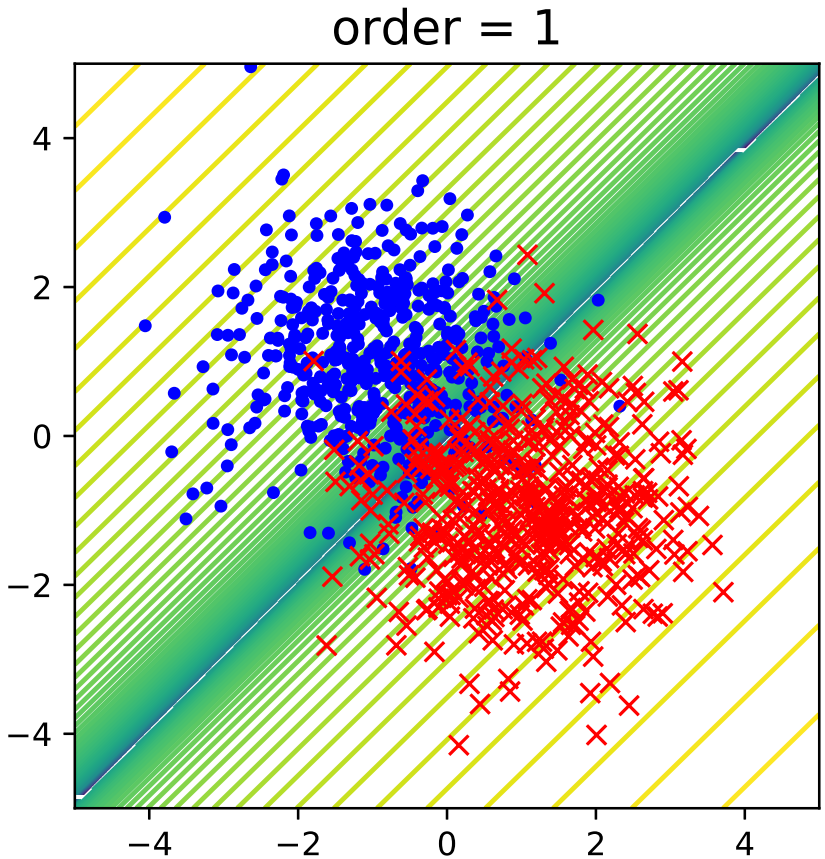
14. Generalization

- Overfitting
- Variance, bias, generalization
- Cross validation

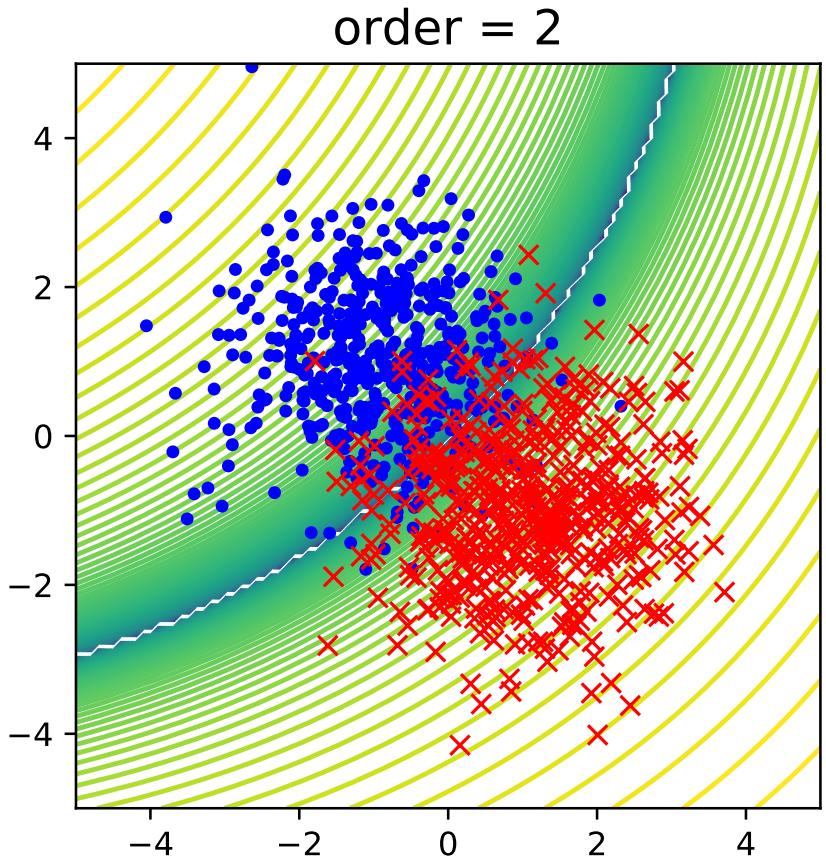
Linear classifier, polynomial basis functions



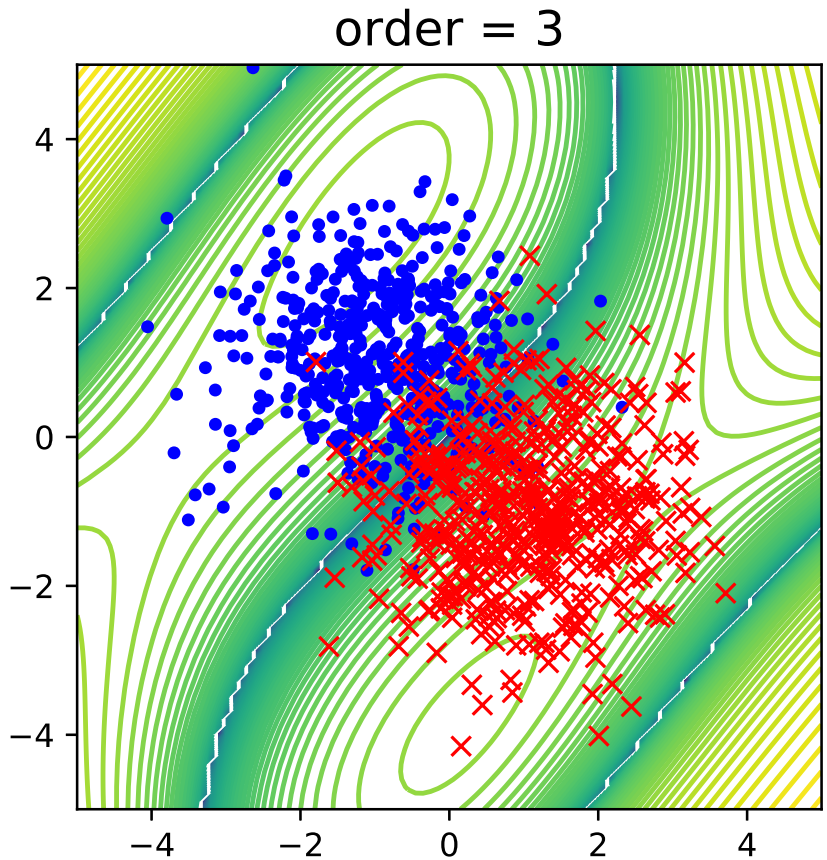
Linear classifier, polynomial basis functions



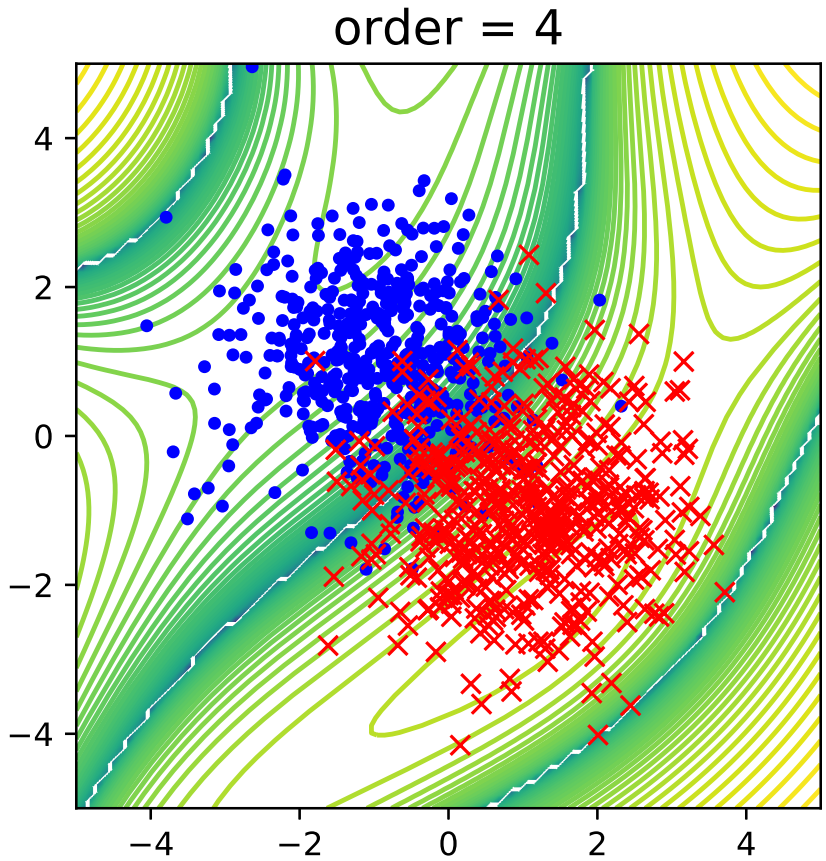
Linear classifier, polynomial basis functions



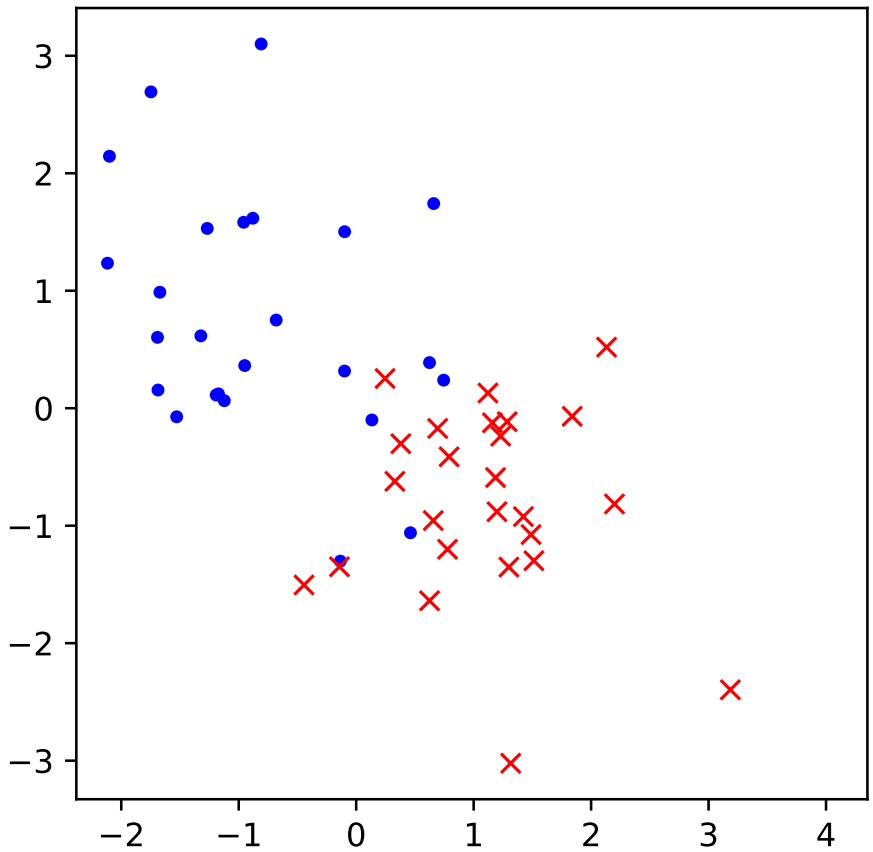
Linear classifier, polynomial basis functions



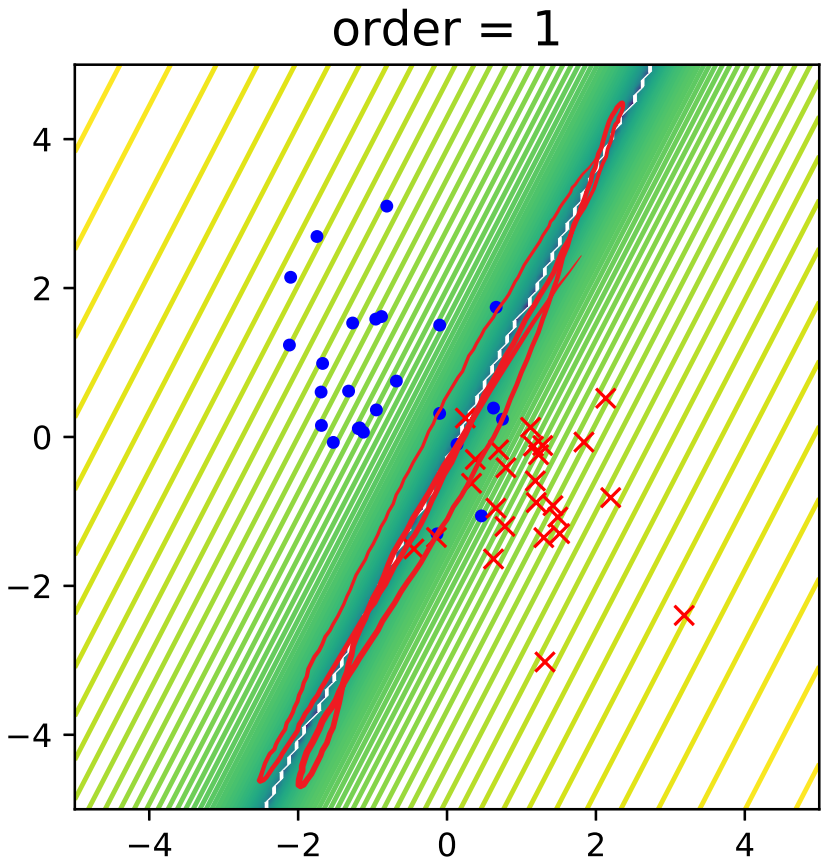
Linear classifier, polynomial basis functions



Linear classifier, polynomial basis functions

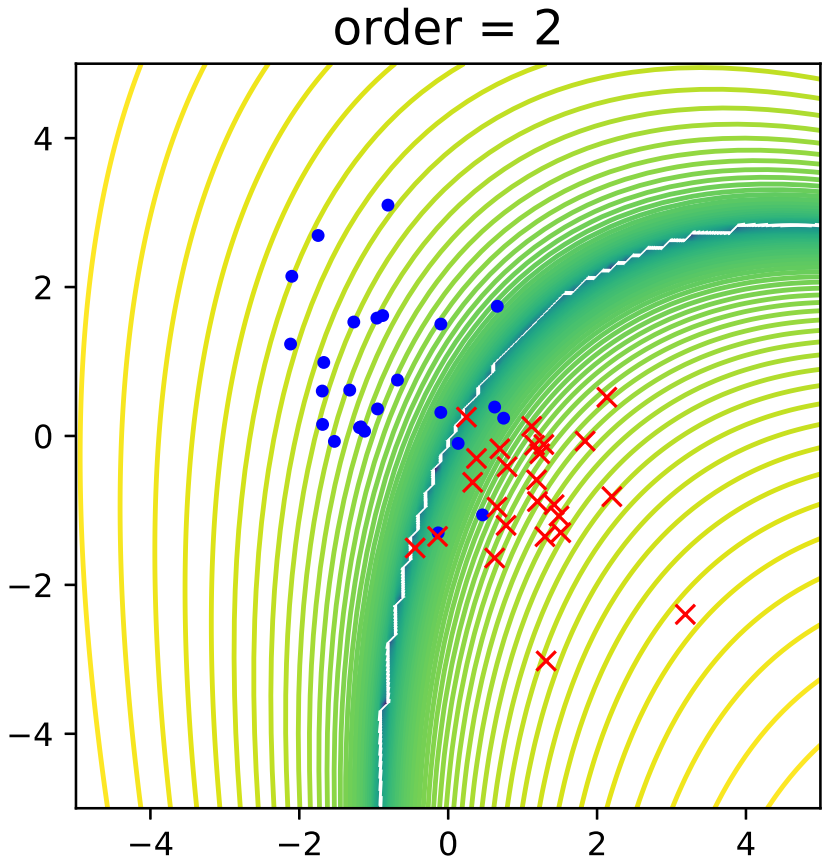


Linear classifier, polynomial basis functions

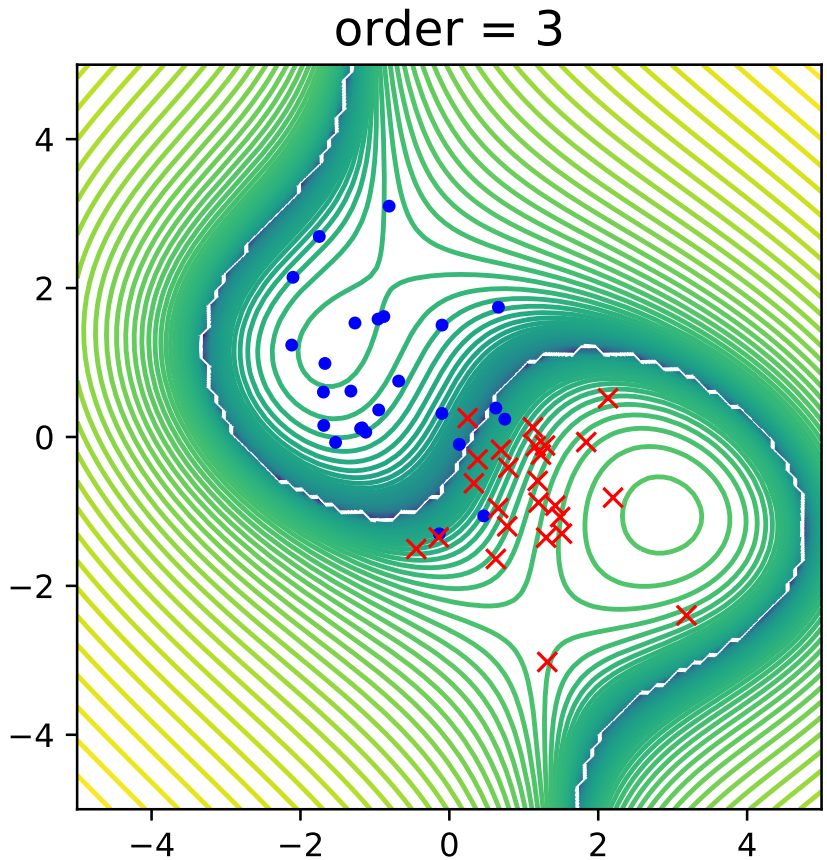


$E_D[h(x) - f_D(x)]$

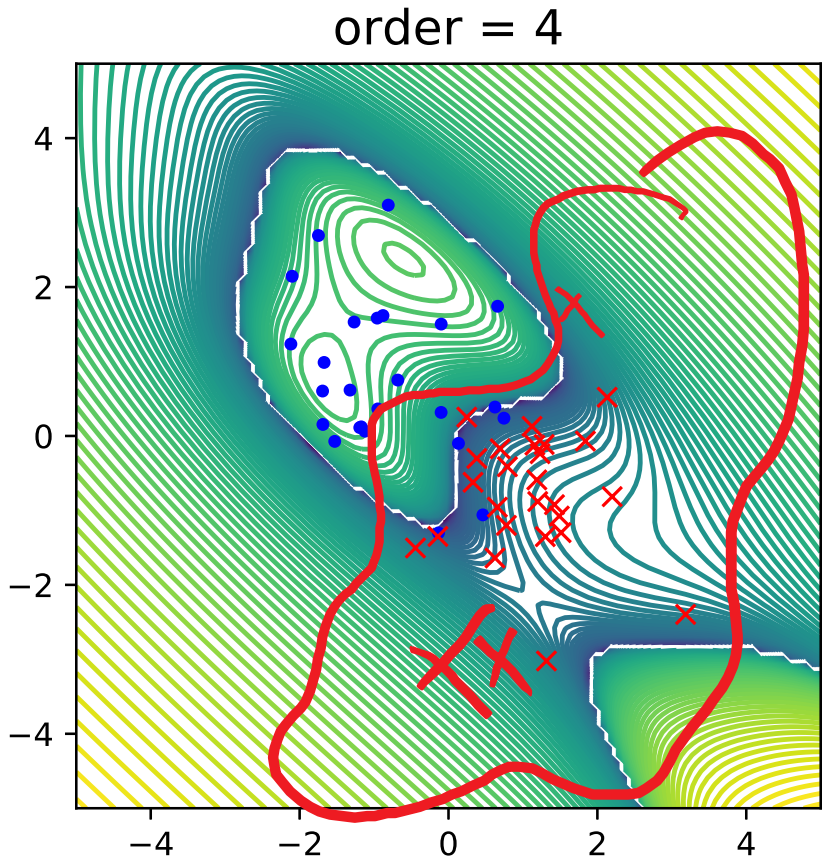
Linear classifier, polynomial basis functions



Linear classifier, polynomial basis functions

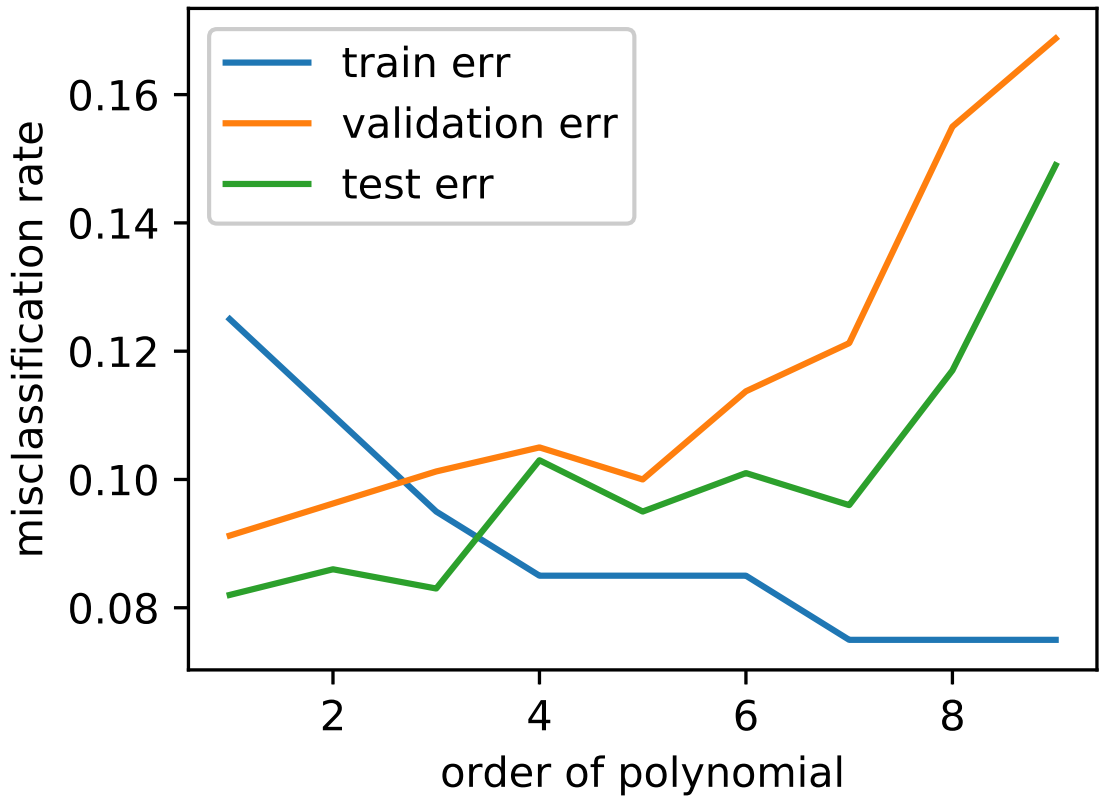


Linear classifier, polynomial basis functions



$\hat{h}_\theta(x)$

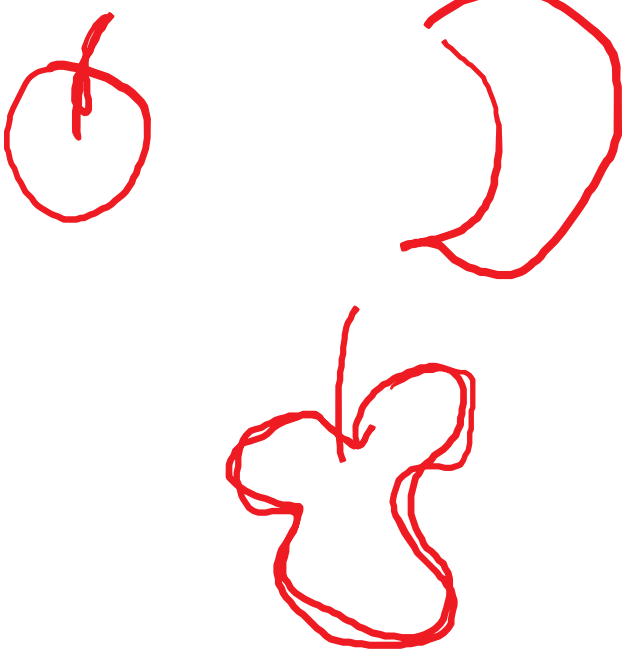
Overfitting



What is overfitting?

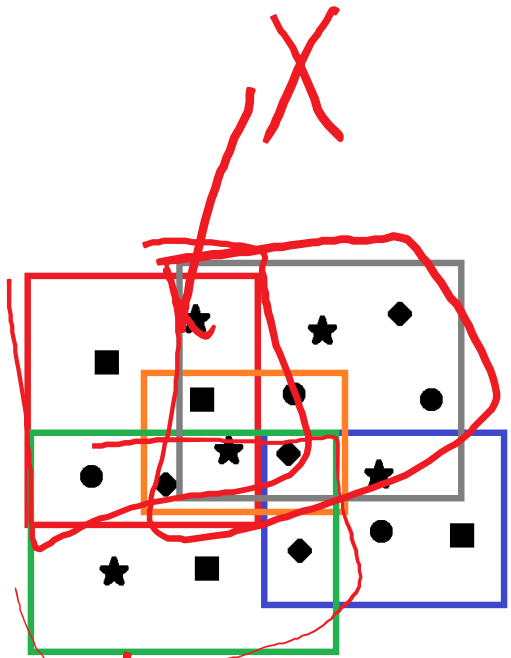
"perfect learn"

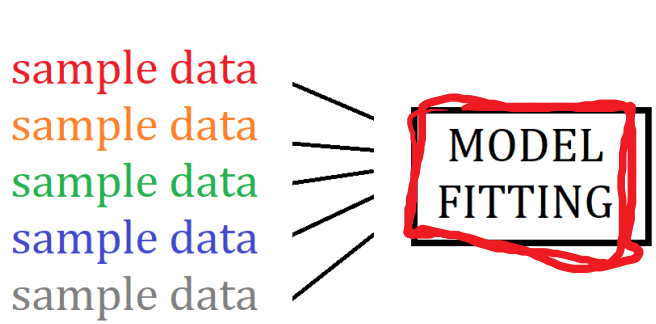
- train ↓ 0, test doesn't
- model has "learned" noise



"out of training distribution error"

Model as random variable





Each data sample can lead to a different model

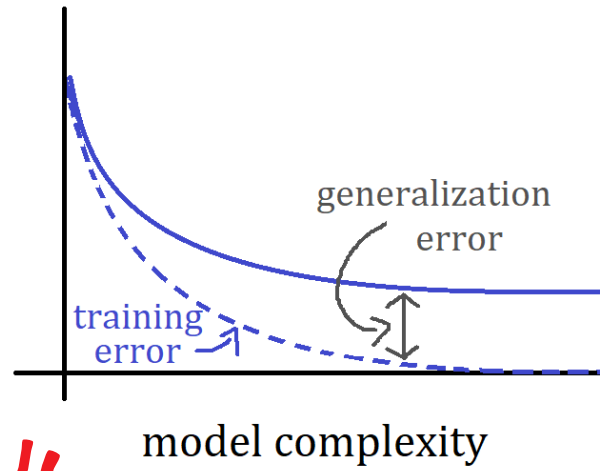
Training error vs generalization error

- Empirical risk

$$\mathcal{E}_{\mathcal{D}} := \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \mathcal{L}(\text{pred}_{\mathcal{D}}(x), \text{label}(x))$$

- Expected risk

$$\mathcal{E}^* := \mathbb{E}_x[\mathcal{L}(\text{pred}_{\mathcal{D}}(x), \text{label}(x))]$$



- Then

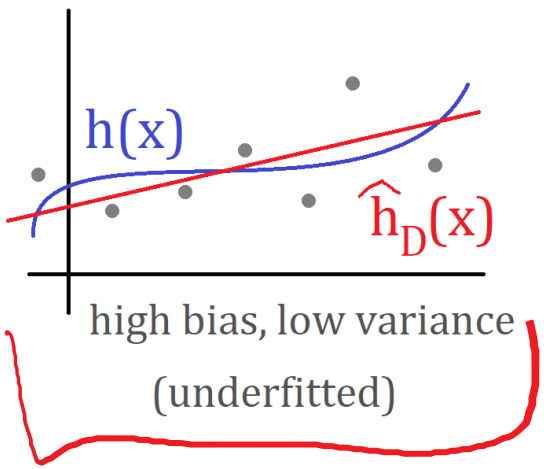
$$\mathcal{E}^* = \underbrace{\mathcal{E}_{\mathcal{D}}}_{\text{training error}} + \underbrace{\mathcal{E}^* - \mathcal{E}_{\mathcal{D}}}_{\text{generation error}}$$

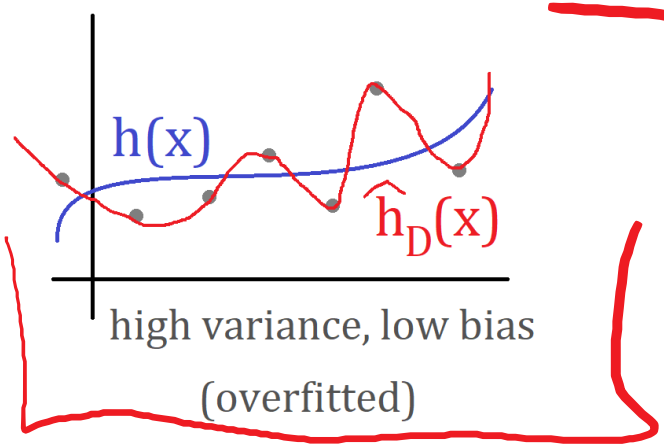
(duh!)

Bias Variance tradeoff

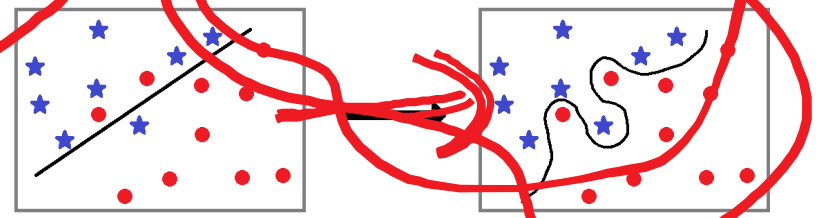
- True response h, trained model $\hat{h}_{\mathcal{D}}$
- Mean squared error of model on new sample x:

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

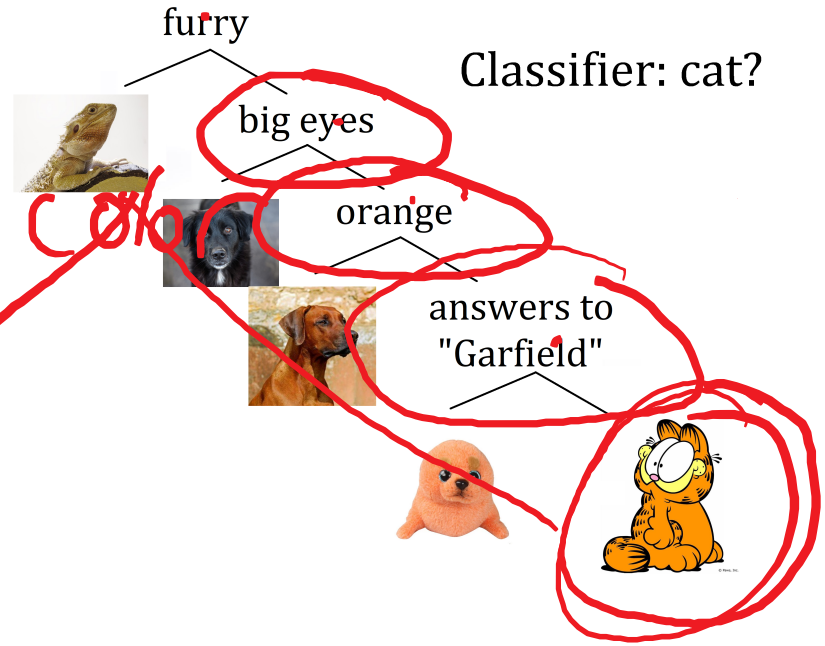




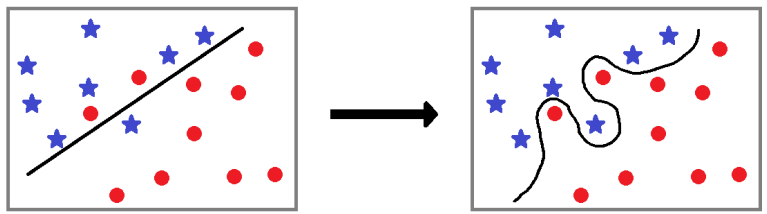
Example: Decision trees



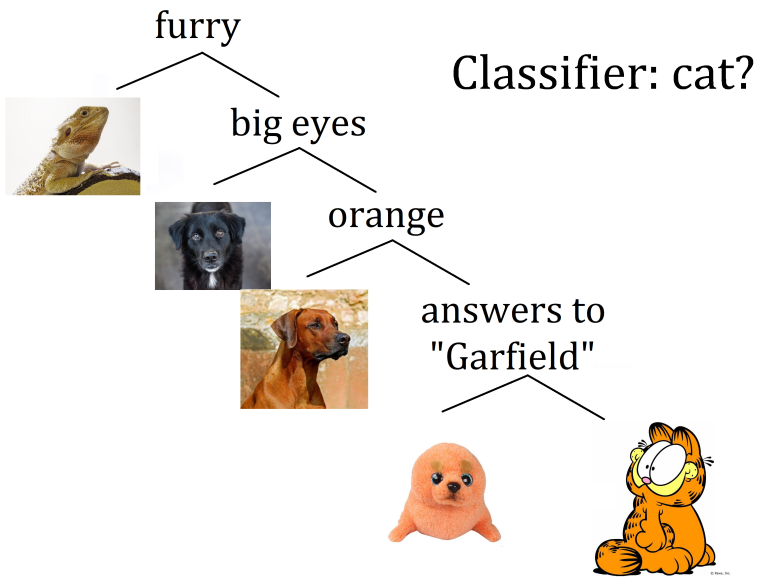
- model complexity?
- depth of tree?
- # samples $\ll$ # features?
  - Can linear models overfit?



Example: Decision trees



- model complexity?
- depth of tree?
- # samples $\ll$ # features?
  - Can linear models overfit?
    Yes! if there are far more features than samples!



Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

unbiased

$$\mathbb{E}[\hat{\theta}] = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \mathbb{E}[\theta]$$

Example revisited: Point estimation

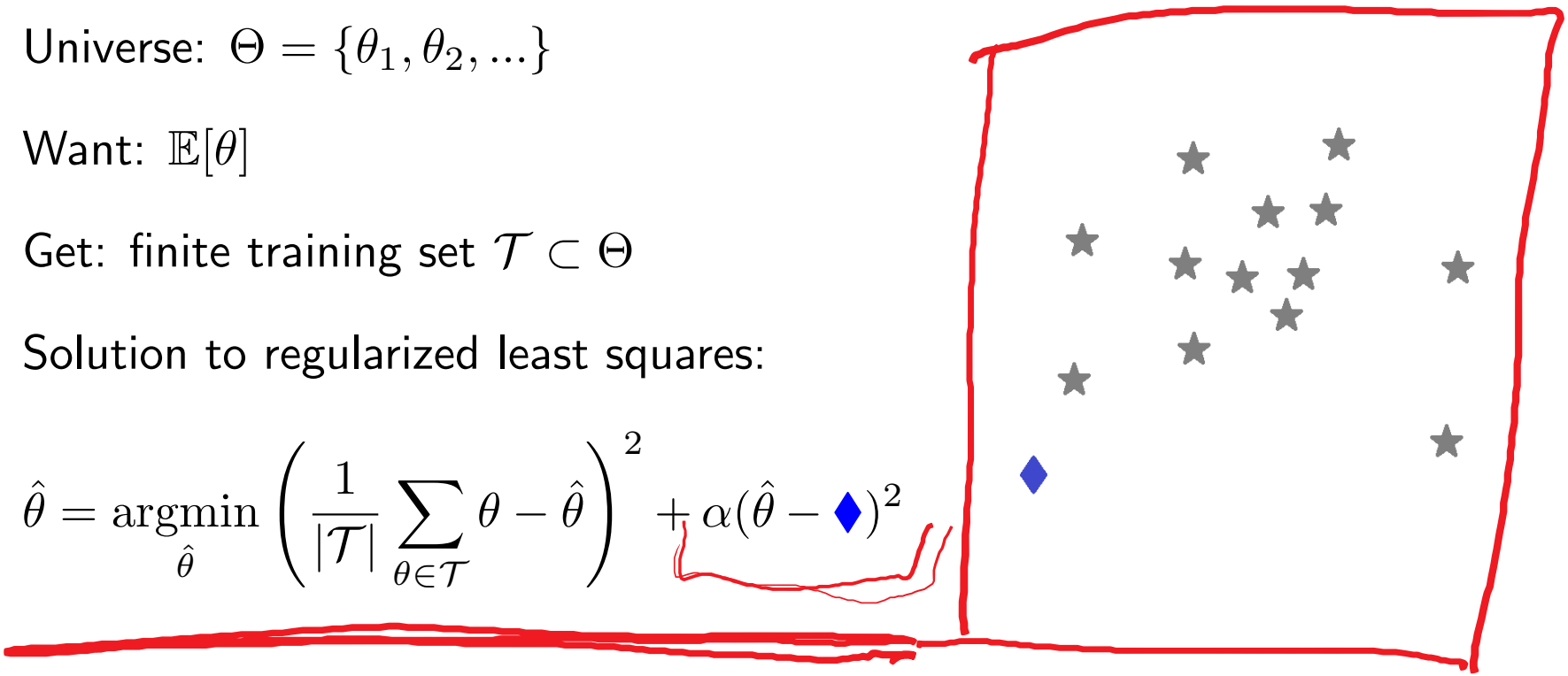
- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Sample estimate:

$$\hat{\theta} = \frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta$$

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

$$\hat{\theta} = \underset{\hat{\theta}}{\operatorname{argmin}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$



Higher bias, lower variance

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

$$\hat{\theta} = \underset{\hat{\theta}}{\operatorname{argmin}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$

Higher bias, lower variance

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

$$\hat{\theta} = \operatorname*{argmin}_{\hat{\theta}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$

Higher bias, lower variance

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

$$\hat{\theta} = \operatorname*{argmin}_{\hat{\theta}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$

Higher bias, lower variance

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

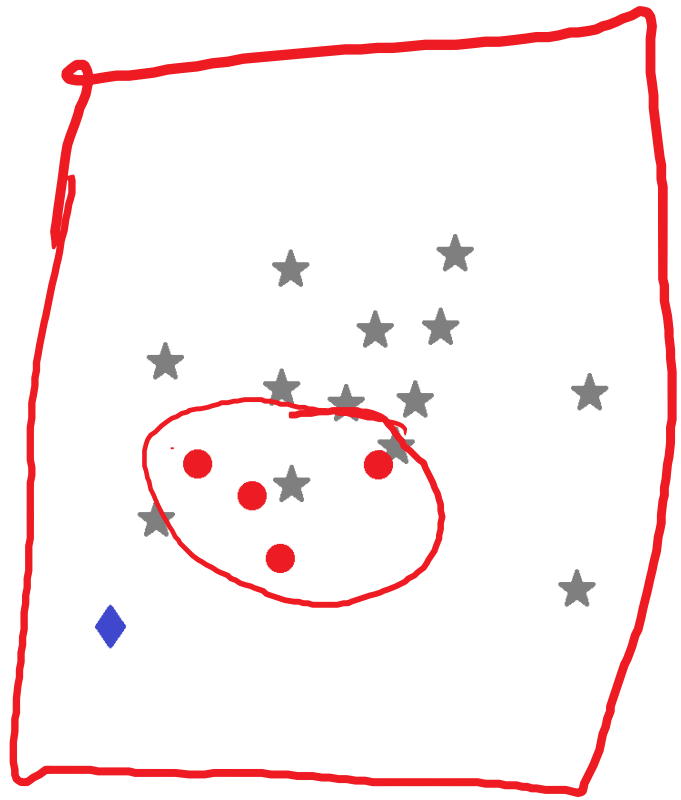
$$\hat{\theta} = \operatorname*{argmin}_{\hat{\theta}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$

Higher bias, lower variance

Example revisited: Point estimation

- Universe: $\Theta = \{\theta_1, \theta_2, ...\}$
- Want: $\mathbb{E}[\theta]$
- Get: finite training set $\mathcal{T} \subset \Theta$
- Solution to regularized least squares:

$$\hat{\theta} = \operatorname*{argmin}_{\hat{\theta}} \left(\frac{1}{|\mathcal{T}|} \sum_{\theta \in \mathcal{T}} \theta - \hat{\theta} \right)^2 + \alpha(\hat{\theta} - \blacklozenge)^2$$



Higher bias, lower variance

Ways to combat: Get more data!

- This is the best solution, but usually impossible/difficult in practice
- Generalization error: Assuming $\mathcal{D}$ sampled uniformly

$$\mathcal{L}_{\mathcal{D}} \overset{|\mathcal{D}|\to+\infty}{\to} \mathcal{L}^*$$

- Variance:

$$\mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(x)^2] - \left(\mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(x)]\right)^2 = \left(\mathbb{E}_{\mathcal{D}} \underbrace{\left[\hat{h}_{\mathcal{D}}(x) - \mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(x)]\right]}_{\to 0 \text{ as } |\mathcal{D}|\to+\infty}\right)^2$$

- Predictor approaches best possible answer given model complexity
- In practice: increase model complexity with more data, to drive bias $\to 0$

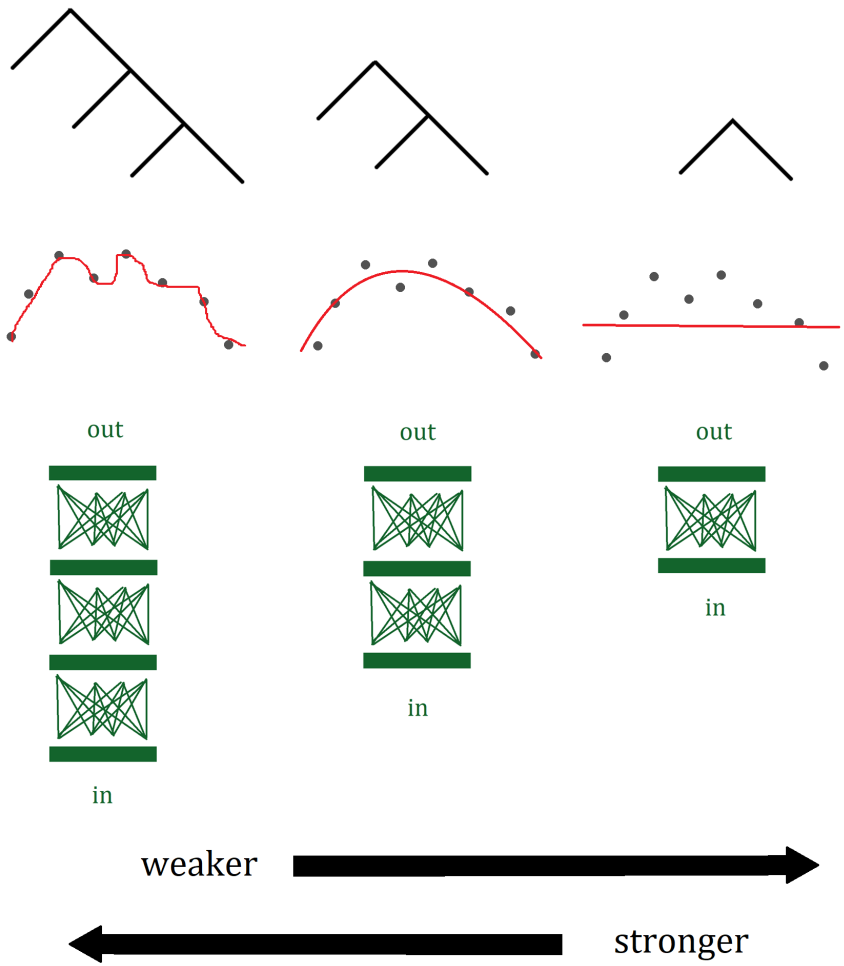
Ways to combat: Regularization

- Goal: reduce $\mathbb{E}_{x,y}\mathcal{L}_\theta(\hat{f}(x), y)$
- In practice, minimize regularized empirical risk

$$\underset{\theta}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_\theta(f(x_i), y_i) + \rho\mathcal{R}(\theta)$$

- $\mathcal{R}(\theta) = \|\theta\|_2^2$ draws θ closer to 0 and improves conditioning
- $\mathcal{R}(\theta) = \|\theta\|_1$ promotes sparse θ

Ways to combat: Weakening the learner



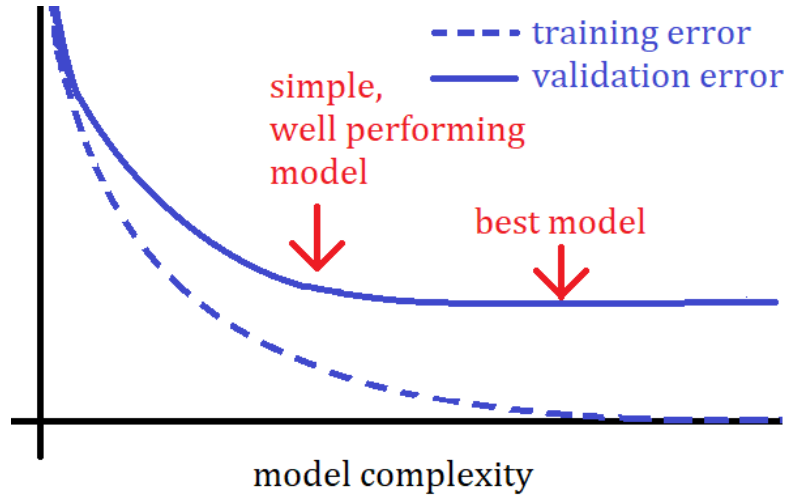
- Weaker model = larger bias, smaller variance
- Hyperparameters
  - depth of tree
  - width of neural network
  - order of polynomial
  - ...

Cross validation: Picking hyperparameters

1. Split training data $\mathcal{D} = \mathcal{D}_T \cup \mathcal{D}_V$
2. Sweep hyperparameters, train each model on $\mathcal{D}_T$
3. Evaluate each model on $\mathcal{D}_V$, pick simplest model with best validation score

Cross validation: picking hyperparameters

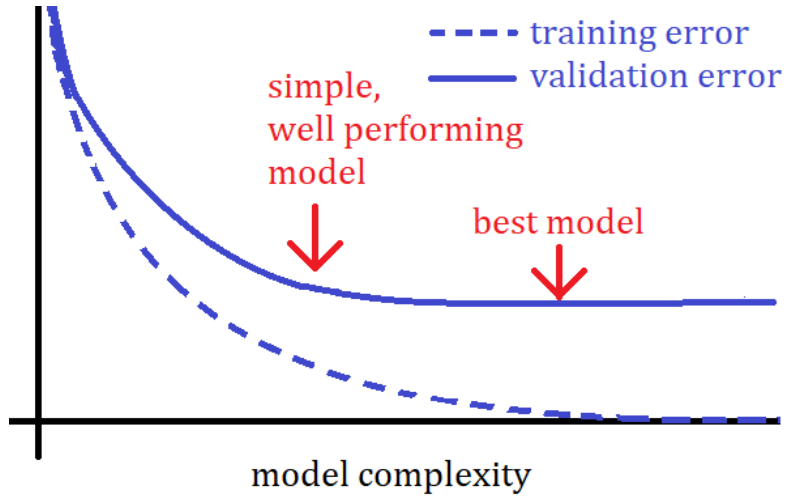
1. Split training data $\mathcal{D} = \mathcal{D}_T \cup \mathcal{D}_V$
2. Sweep hyperparameters, train each model on $\mathcal{D}_T$
3. Evaluate each model on $\mathcal{D}_V$, pick simplest model with best validation score



- "K-fold cross validation, 75 /25 split" =
  repeat 1-3 K times with $\mathcal{D} = \underbrace{\mathcal{D}_T}_{75\%} \cup \underbrace{\mathcal{D}_V}_{25\%}$, randomly partitioned
- Key: never use test set in cross validation! (why?)

Cross validation: picking hyperparameters

1. Split training data $\mathcal{D} = \mathcal{D}_T \cup \mathcal{D}_V$
2. Sweep hyperparameters, train each model on $\mathcal{D}_T$
3. Evaluate each model on $\mathcal{D}_V$, pick simplest model with best validation score



- "K-fold cross validation, 75 /25 split" =
  repeat 1-3 K times with $\mathcal{D} = \underbrace{\mathcal{D}_T}_{75\%} \cup \underbrace{\mathcal{D}_V}_{25\%}$, randomly partitioned
- Key: never use test set in cross validation! (why?)
  Hyperparameters too can be overfitted!

K-fold cross validation

- Divide the data into K disjoint subsets
  - Train on union of $K-1$ subsets
  - Test on the left-out set
  - Repeat K times
- Leave one out CV (LOOCV) is K-fold CV with $K = \#$ of data points

Summary

- Training error: how well your model fit data you saw
- Generalization error: how consistent your model behaves on unseen data
- Balancing act

$$\text{Total error} = \text{training error} + \text{generalization error}$$

- Ways to improve generalization error
  - More data
  - Weakening the model
  - Cross validation