CSE 242: Machine Learning

2022 Fall

Razvan Marinescu
Computer Science and Engineering

Welcome

- Instructor: Razvan Marinescu
  - ramarine@ucsc.edu
  - Class times: Tue & Thu: 9:50am – 11:25am
  - In-person class (zoom available, but only if absolutely necessary – check canvas for link)

- Two TAs: Fatemeh Elyasi & Swati Jindal
- NEW Two tutors/graders! (still in hiring process)

- Lectures & office hours:
  - Keep your conversations respectful and polite.
  - Attendancy is required, active class participation will count towards your final grade.
  - Lectures will be video-recorded and made available after the class
  - Office hours:
    - Fatimeh: Tue 1-3pm
    - Swati: Thu 12-2pm
    - Razvan: Tue 4-6pm

- Lab sessions:
  - Hosted by Fatemeh & Swati
  - Time: One of the office hours will be used, **TBD**
  - First three weeks: intro to jupyter notebook and assignment submission

- Tutors:
  - Grade your assignments
  - Host office hours to address questions related to assignments

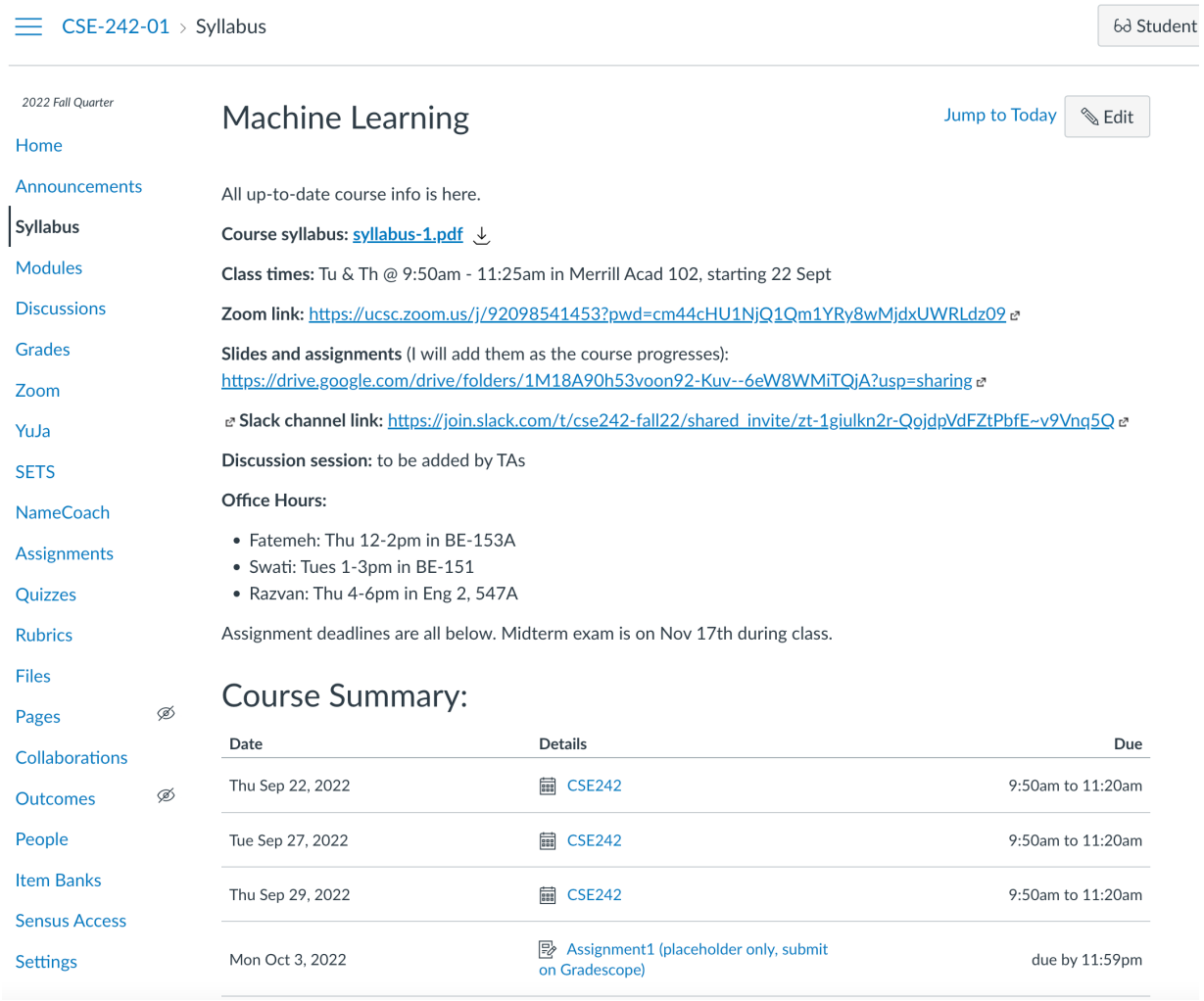
Meeting online

- Online class
  - The live lecture time will be a mixture of lectures, online discussions, meeting our teaching staffs, questions and answering

Web Info

- Emailing the instructor might ***not*** be the fastest way to get a response
- Sign up on Piazza (access code=CSe242) at
  - https://piazza.com/ucsc/fall2022/cse242
  - Use Piazza for all questions
- We will also enable Slack for asking questions
- Assignment submission on Gradescope:
- Make sure you access Canvas, Piazza and UCSC Email regularly

All up-to-date links to slides, assignments, zoom, etc … **are on Canvas**



Your course evaluation

- Midterm exam (30%): will be conducted online via Zoom
- Assignments (5 x 12% = 60%):
  - 5 assignments contribute equally to your final grade, each might consist a writing component and a programming component.
  - Will be posted on Canvas, you submit your copies on Gradescope.

```
Assignment 1: get you started
```

- Participation (10%): lectures, class exercises, discussions, attending invited talks, posting answers on piazza, etc

This course is not about

- Database
- Data mining
- Data processing
- Data visualization
- Coding a deep neural network
- Build a conversational robot

Planned Topics (tentative)

Week 1

- Basic concepts of machine learning

Week 2

- Probability reviews (MAP, MLE)
- Linear algebra (PCA)

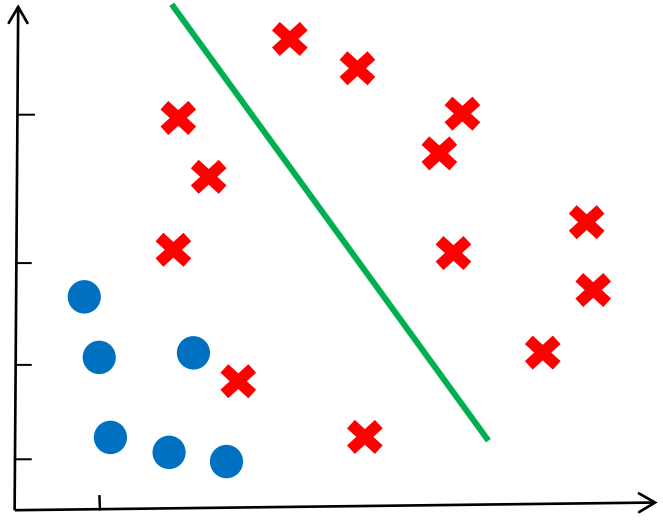
Planned Topics

Week 3 (Regression and Optimization)

- Linear regression, Logistic regression
- Risk minimization, SGD

Week 4 (Linear Models)

- Perceptron
- Support Vector Machine



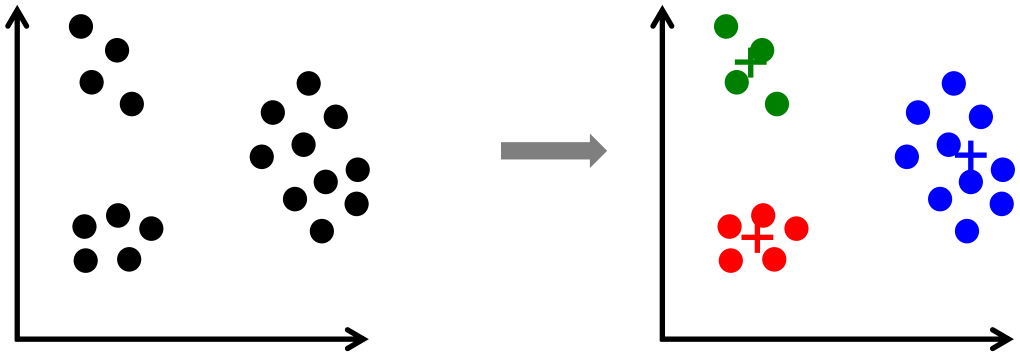
Planned Topics

Week 5 (light touch on other ML models)

- Kernel
- KNN, Naïve Bayes

Week 6

- Decision Tree
- Clustering
- EM



Planned Topics

Week 7

- Boosting
- Special topics

Week 8

- Reinforcement learning

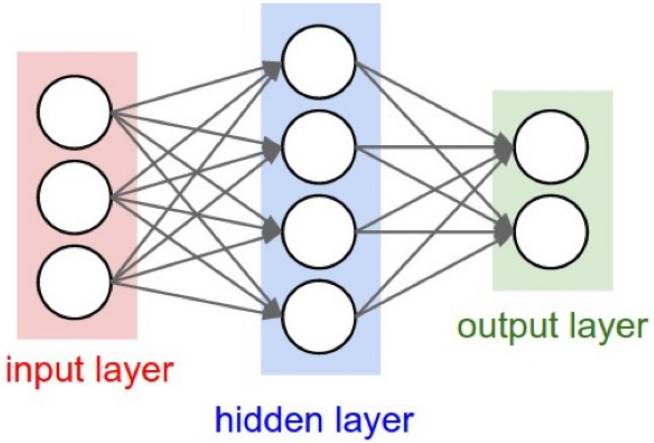
Planned Topics

Week 9

- Guest lecture
- **17th Nov: Midterm exam (during the lecture)**
- DRC students, email me in advance if you need special accommodations

Week 10

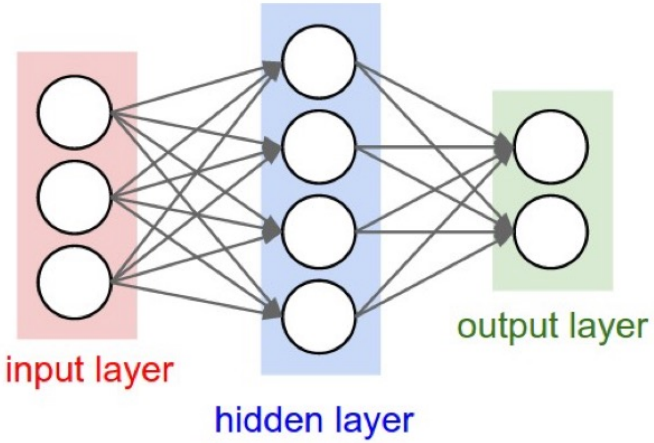
- Neural Networks
- Thanksgiving



Planned Topics

Week 11

- Generative Adversarial Networks
- Special Topics (my research on Bayesian GANs, or read a recent research paper)



A Brief Introduction

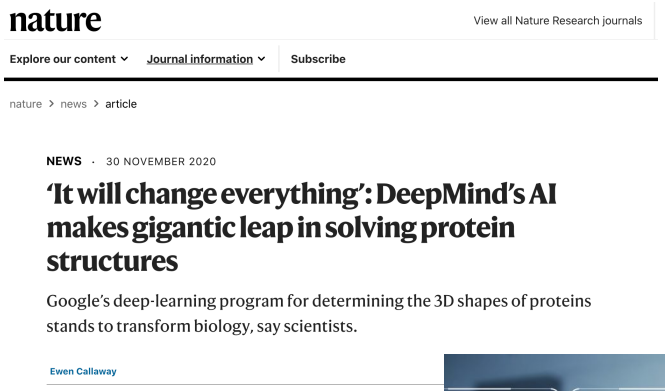
Annoucements

- We will use both Piazza and Slack for questions and discussions.
- Assignment 1 will be released later today. You will have two weeks to complete.

Classroom policy

- Quick and clarification questions are welcomed
- Vague questions ("I don't understand"; "can you repeat everything again") **=>** Office hour, online platforms, and tutoring sessions
- Frequent questions **=>** Office hour, online platforms, and tutoring sessions (think of the lecutres are serving a much broader audience than yourself)
- All other logistical questions **=>** Office hour, piazza, slack, email

Machine Learning


nature

View all Nature Research journals

Explore our content · Journal information · Subscribe

nature › news › article

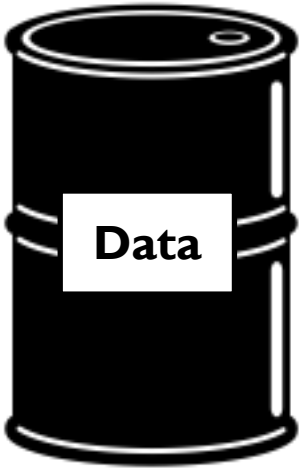
NEWS · 30 NOVEMBER 2020

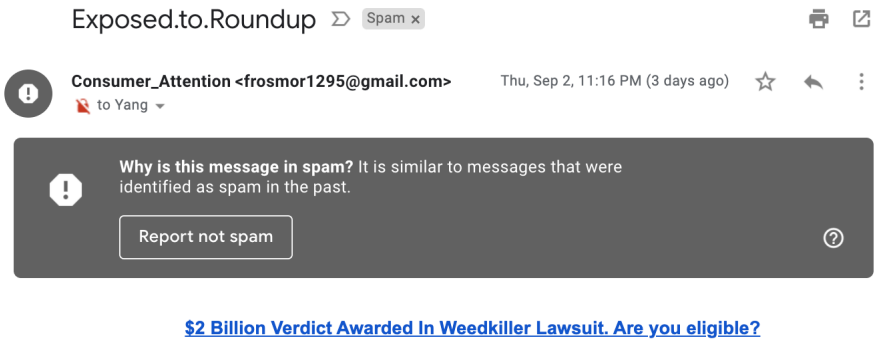
'It will change everything': DeepMind's AI makes gigantic leap in solving protein structures

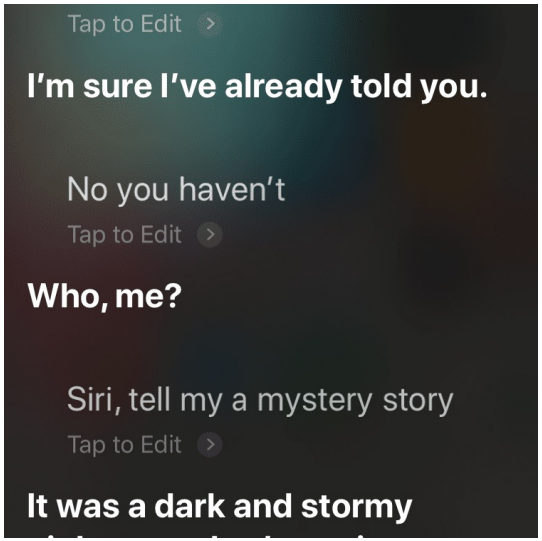
Google's deep-learning program for determining the 3D shapes of proteins stands to transform biology, say scientists.

Ewen Callaway

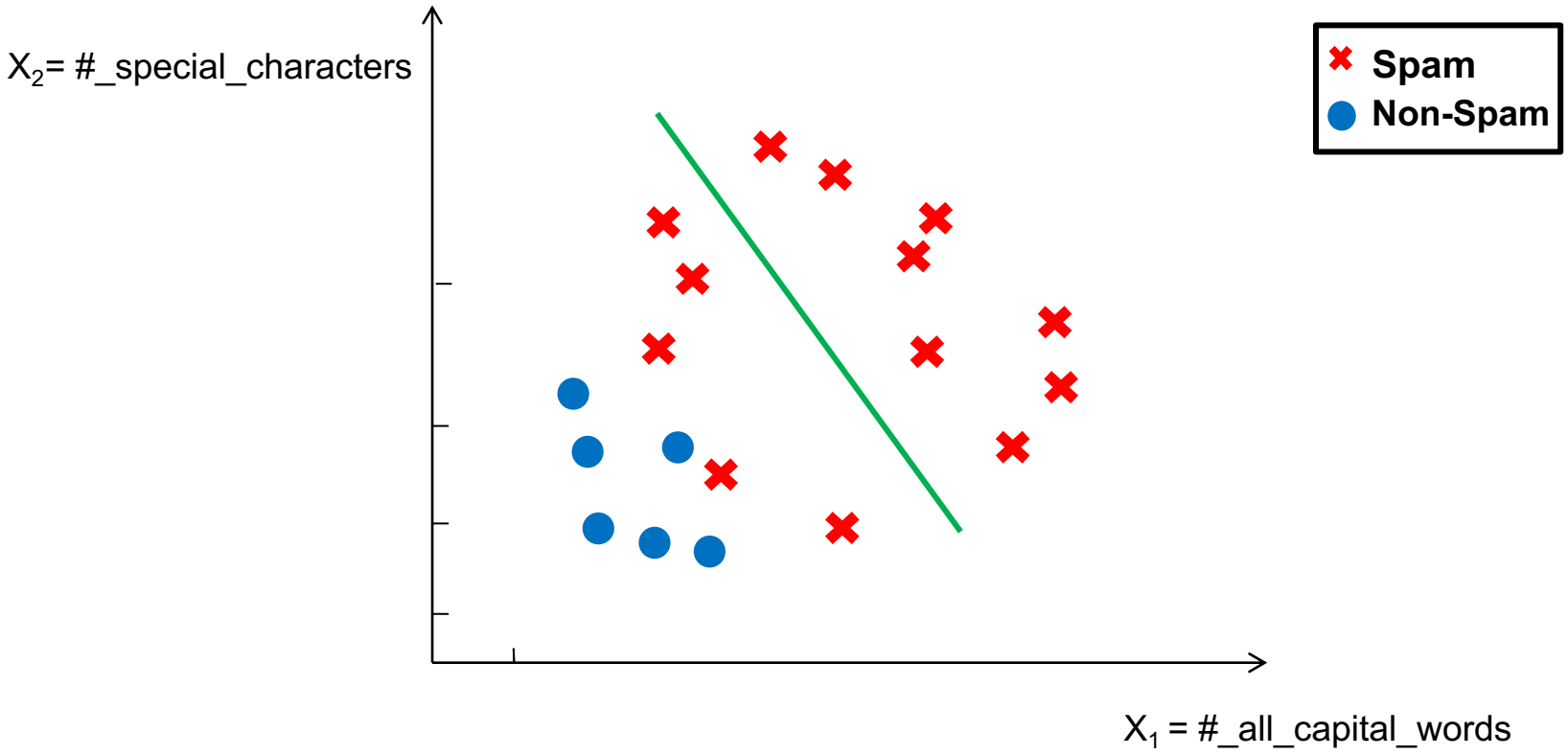
What is data?


Data


Exposed.to.Roundup
Spam
Consumer_Attention <frosmor1295@gmail.com>
Thu, Sep 2, 11:16 PM (3 days ago)
to Yang
Why is this message in spam? It is similar to messages that were identified as spam in the past.
Report not spam
$2 Billion Verdict Awarded In Weedkiller Lawsuit. Are you eligible?


Tap to Edit
I'm sure I've already told you.
No you haven't
Tap to Edit
Who, me?
Siri, tell my a mystery story
Tap to Edit
It was a dark and stormy

Representing Data



Representing Data

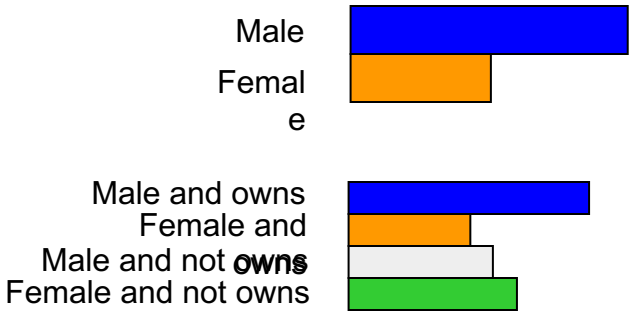
Data

Feature X **=** [
`most_recent_payment,`
`months_piad_in_full_in_last_6_months,`
`annual_salary,`
`#_loans]`

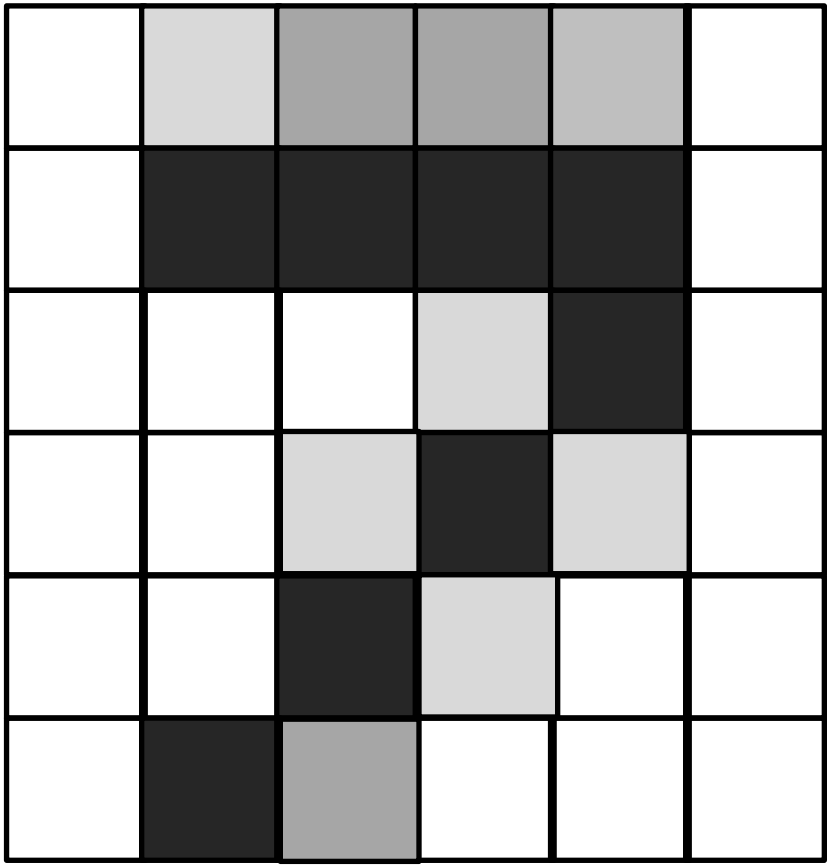
Label Y **=** [`1,0`] (Qualified)

Representing Data

id	Gender	Owns a car
1	Male	yes
2	Female	yes
3	Female	no
4	Male	yes
5	Female	yes
6	Male	no
	⋮	



Representing Data



0	100	150	150	100	0
0	255	255	255	255	0
0	0	0	100	255	0
0	0	100	255	100	0
0	0	255	100	0	0
0	255	150	0	0	0

Representing Data

0	100	150	150	100	0
0	255	255	255	255	0
0	0	0	100	255	0
0	0	100	255	100	0
0	0	255	100	0	0
0	255	150	0	0	0

```
[ 0, 100, 150, …, 0
  0, 255, 255, …, 0
  …
  0, 255, 150, 0, 0, 0 ]
```

X = `[0, 100, 150, …, 0,`
`0, 255, …, 0,…,`
`0, 255, 150, 0, 0, 0]`

Y = `[0, …, 1,…, 0]`

8th

Representing Data

0	100	150	150	100	0
0	255	255	255	255	0
0	0	0	100	255	0
0	0	100	255	100	0
0	0	255	100	0	0
0	255	150	0	0	0

$\boldsymbol{X}$ = [0, 100, 150, …, 0,
0, 255, …, 0,…,
0, 255, 150, 0, 0, 0]

Is this the best way to represent the data?

We will explore other options.

Types of data

Kind	*Order*	*Scale*	*Tendency*	*Dispersion*	*Shape*
Categorical	×	×	mode	n/a	n/a
Ordinal	√	×	median	quantiles	n/a
Quantitative	√	√	mean	range, interquartile range, variance, standard deviation	skewness, kurtosis

Statistics – calculations on the features

Supervised Batch Learning

- Assume (unknown) distribution over "things" (data)
- Things have measurable ***attributes*** or ***features***
- Get ***instances*** (feature vectors) ***X*** by drawing things from distribution and recording observations.
- Teacher ***labels*** instances making ***examples*** (***X, Y)***
- Set of labeled examples is the ***training set*** or ***sample***

Supervised Batch Learning

- Create ***hypothesis*** (rule or function) from sample
- Hypothesis predicts labels of *new* random instances, evaluated using a ***loss function*** (e.g. number of mistakes)

Supervised Learning (cont.)

- **Classification**: labels are nominal (unordered set, e.g. {ham, spam} {democrat, republican, indep.})
- **Binary Classification**
- **Regression**: labels are numeric (e.g. price of house)
- **Ranking** problems (order a set of objects)

Supervised Learning (cont.)

- **Classification**: labels are nominal (unordered set, e.g. {ham, spam} {democrat, republican, indep.}, {0,1,2,…,9})

- **Regression**: labels are numeric (e.g. price of house)

→ $10m

→ $100,000

→ $50,000

Supervised Learning (cont.)

- **Ranking** problems (order a set of objects)

Why "Learn" ?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience (inference in statistics)
- To reason about data $\mathbf{X}$
- There is no need to "learn" to calculate payroll

Why "Learn" ?

- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition, object detection)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted or customized to particular cases (or users)

What We Talk About When We Talk About "Learning"

- Learning general models from a set of particular examples
- **To generate representation g(X) (often unsupervised)**
- **Or a prediction f(X) (often supervised)**

What We Talk About When We Talk About "Learning"

- *Data* is cheap and abundant (data warehouses, data marts); *knowledge* is expensive and scarce.
- Example in retail: Customer transactions to consumer behavior:

  People who bought "Da Vinci Code" also bought "The Five People You Meet in Heaven" (www.amazon.com)

- Build a model that is *a good and useful approximation* to the data.

What is Machine Learning?

- Training feature **X** + labels **Y (sometimes missing)**
  => programs for labeling data

- Optimize a performance criterion using example data or past experience.

- Role of Statistics: Inference from a sample

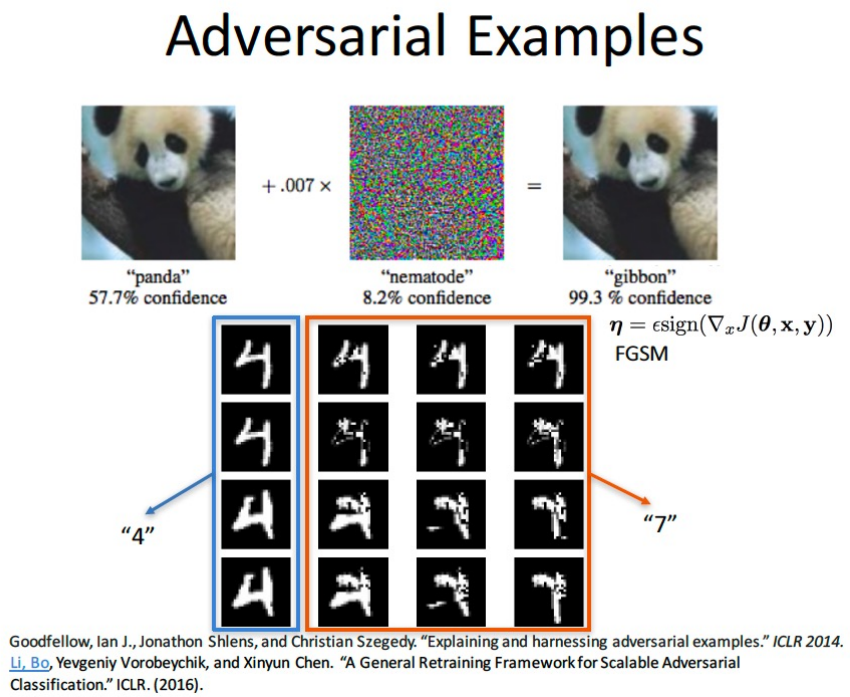
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Representing and evaluating the model for inference

Statistical Machine learning is not:

- Cognitive science (how people think/learn)
- Teaching computers to think

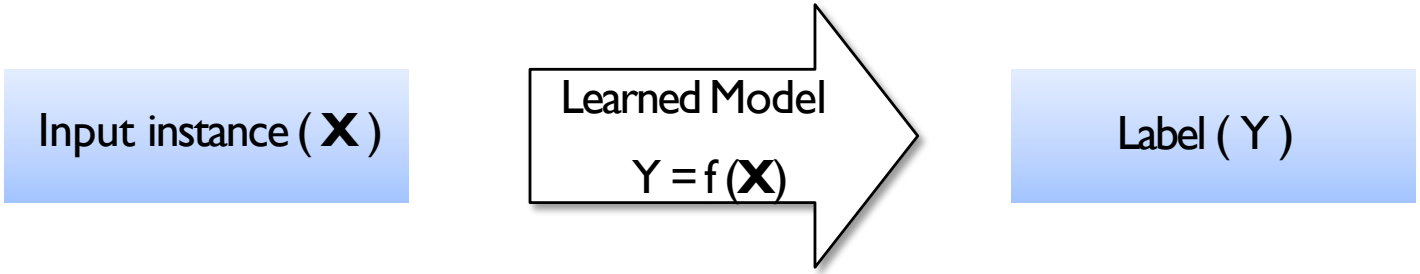
But is related to:

- Statistics
- Data Mining – Knowledge Discovery
- Control theory
- part of AI, but not "traditional" AI



Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *ICLR 2014.*
Li, Bo, Yevgeniy Vorobeychik, and Xinyun Chen. "A General Retraining Framework for Scalable Adversarial Classification." ICLR. (2016).

How do we learn?



- The hypothesis, f(x), is defined by parameters
  - For hypothesis = quadrangle, the parameters are its 4 corners
- How to learn the parameters?
  - Optimize a loss function

Loss function

- Learn parameters by minimizing Loss function -- L(y, y')
  - Loss function measures error of predictions y' (on train set)
  - Loss functions tells you how good f(x) or g(x) is
  - Different for different ML algorithms
  - E.g. of classification loss: L(y,y')=0 if y=y' and L(y,y')=1 o/w
  - E.g. of regression loss: $(y-y')^2$
  - Will see more examples in future

- Minimization is done using an optimization algorithm like gradient descent

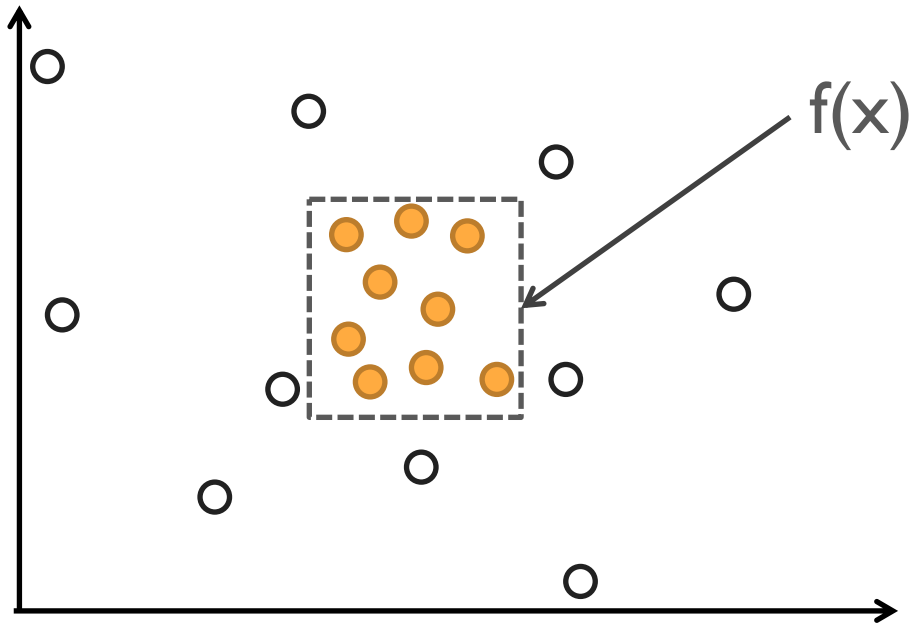
Why learning works?

- Learning is an ill-posed problem:
  If we assume nothing else, any label *Y* could be right for an unseen ***X***

Why learning works?

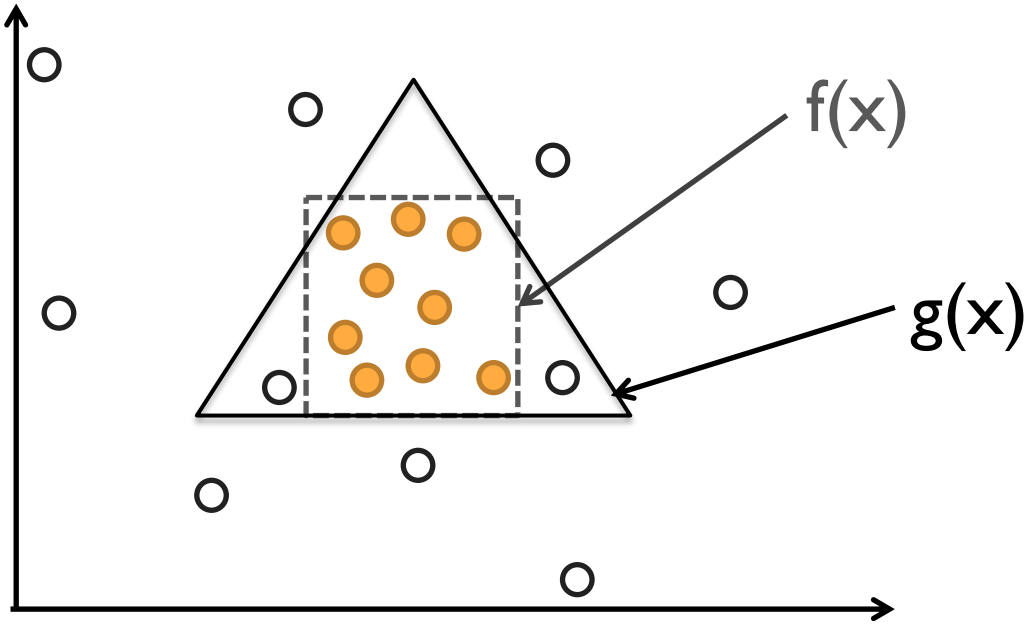
- Need an ***inductive bias*** limiting possible connections between $\boldsymbol{X}, Y$
- Often assume some kind of simplicity (e.g. linearity, neural networks) based on domain knowledge
- Bayesian approach: put *prior* on rules, and balance prior with evidence (data)

Choosing hypothesis space



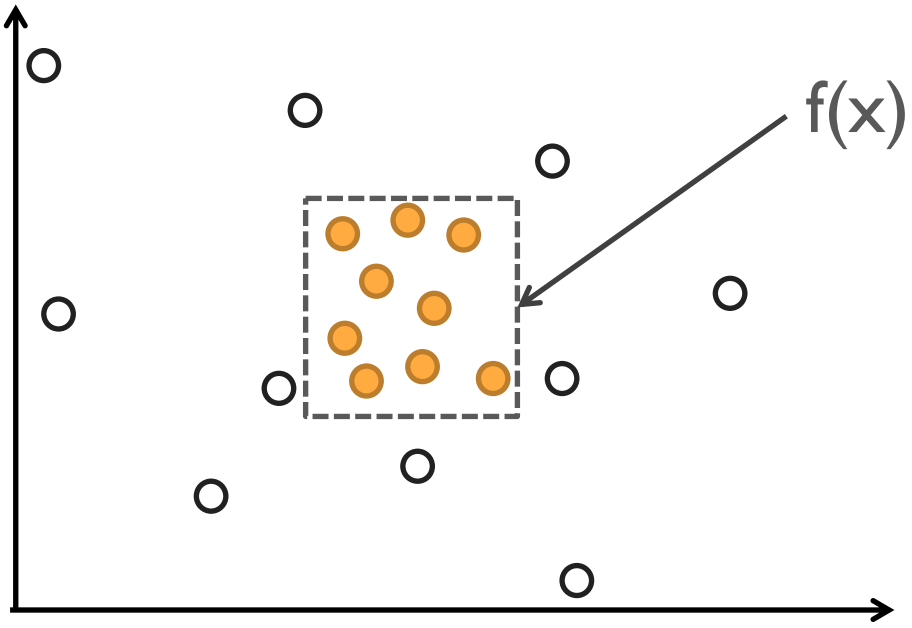
Choosing hypothesis space

- If we choose a hypothesis space that is too simple
  - Fewer parameters to learn, but less powerful
  - Model has less variance: fewer changes with changing training data
  - Model has more bias: makes more assumptions



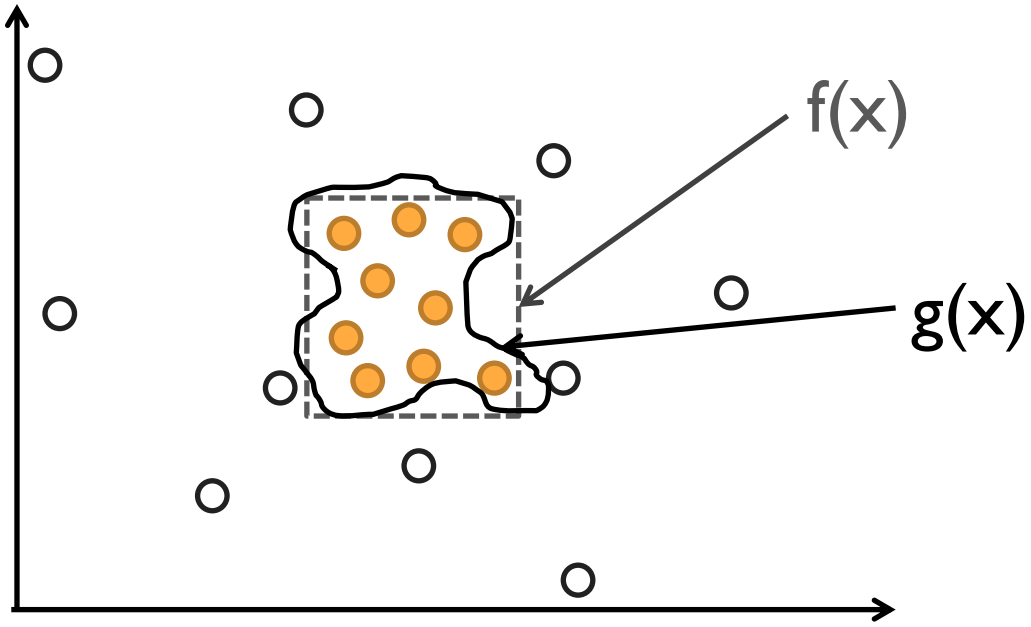
Choosing hypothesis space

- If we choose a hypothesis space that is too complex



Choosing hypothesis space

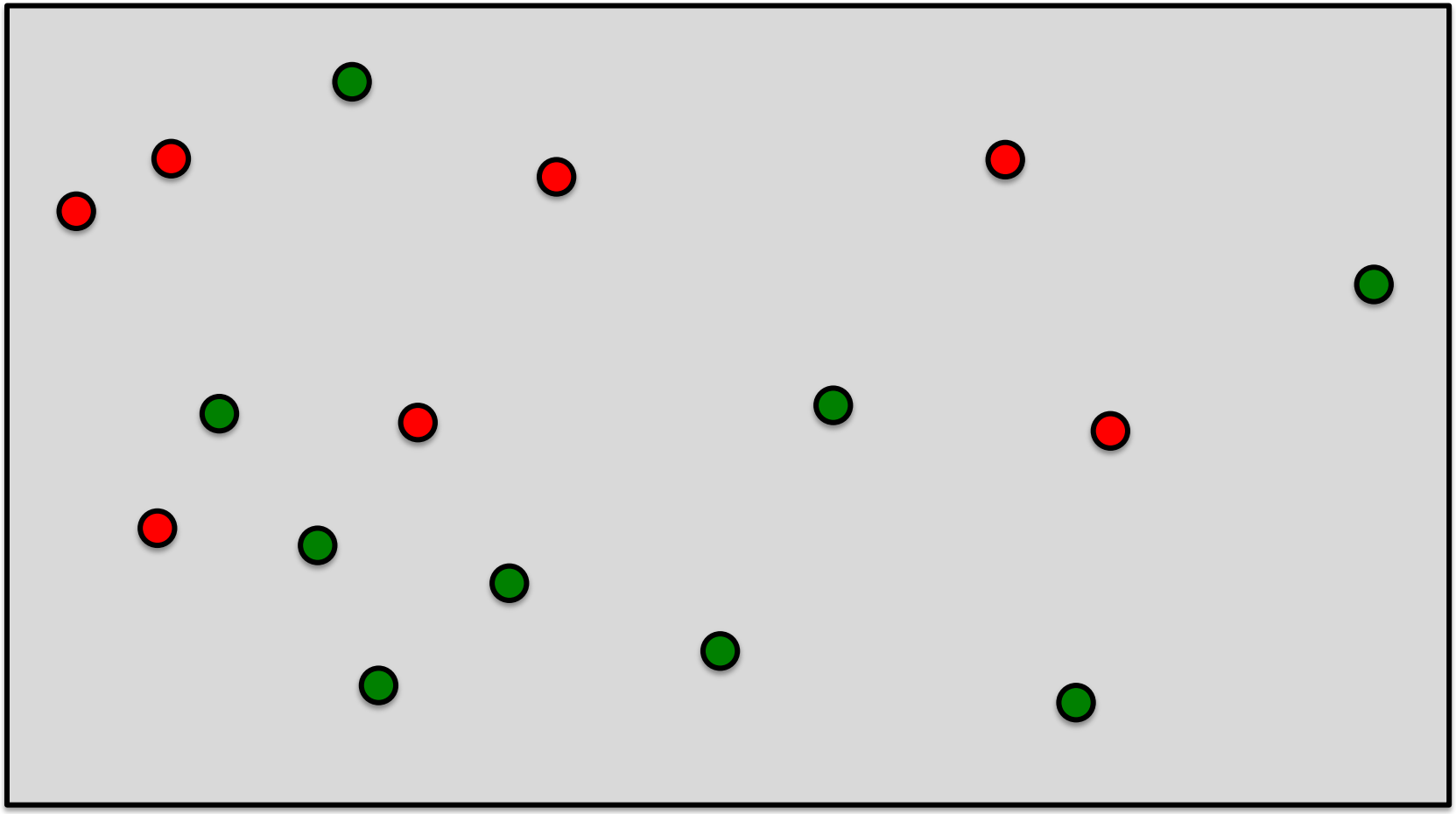
- If we choose a hypothesis space that is too complex
  - More parameters to learn but more powerful
  - Model has more variance and less bias
- This is called bias-variance tradeoff



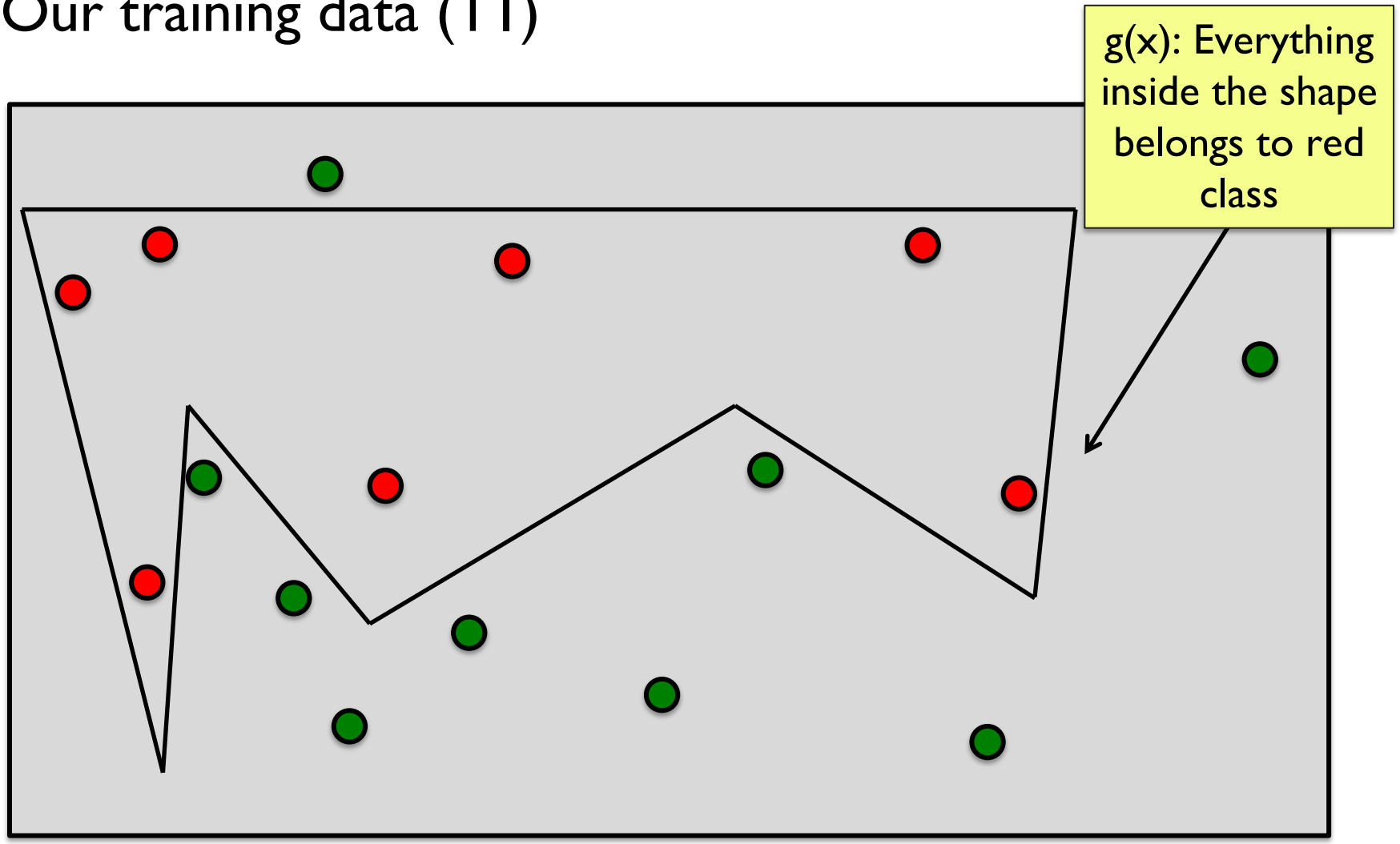
More on bias and variance

- What do we mean when we say a model from a complex model family has high variance?

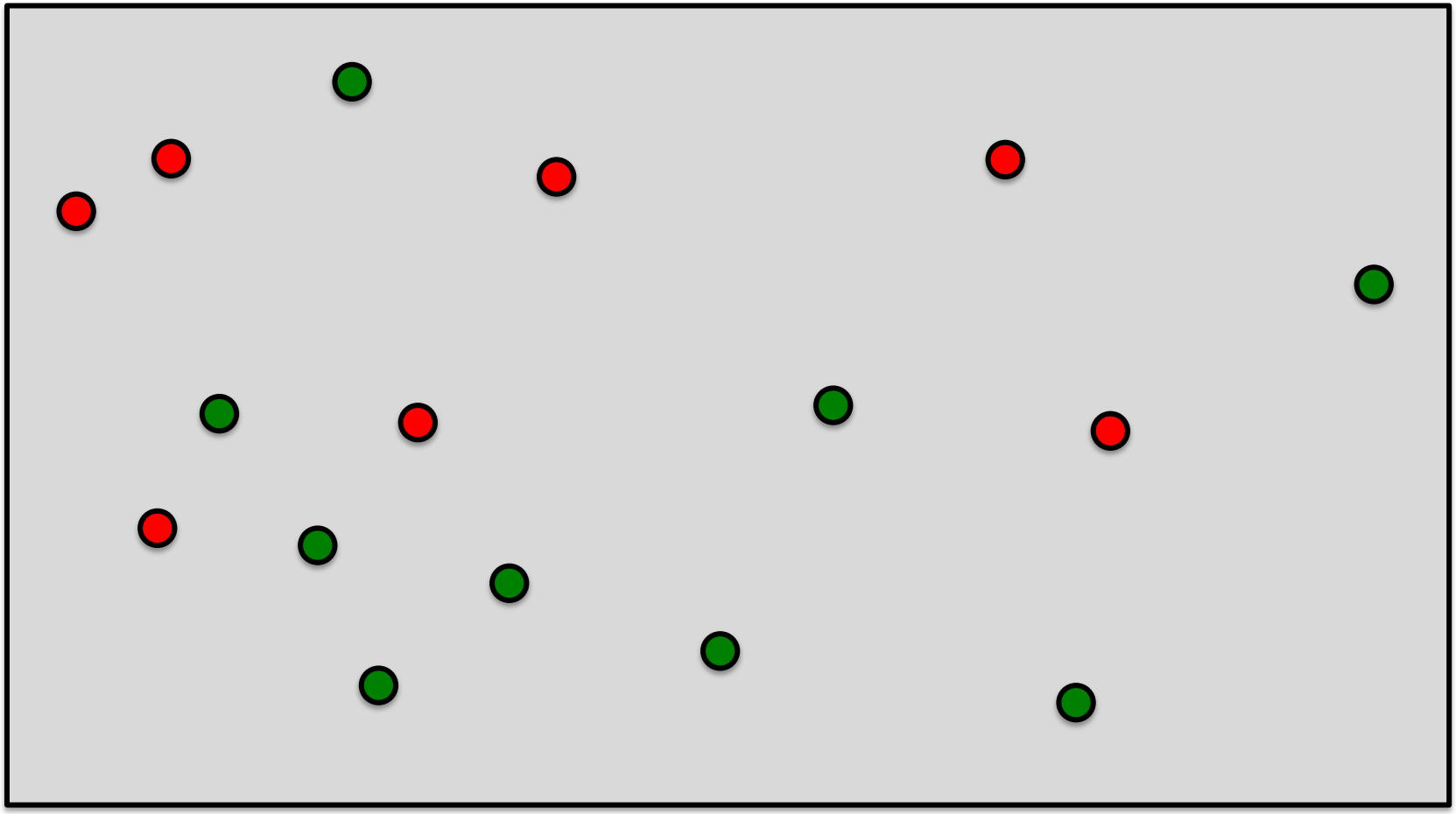
Our training data (T1)



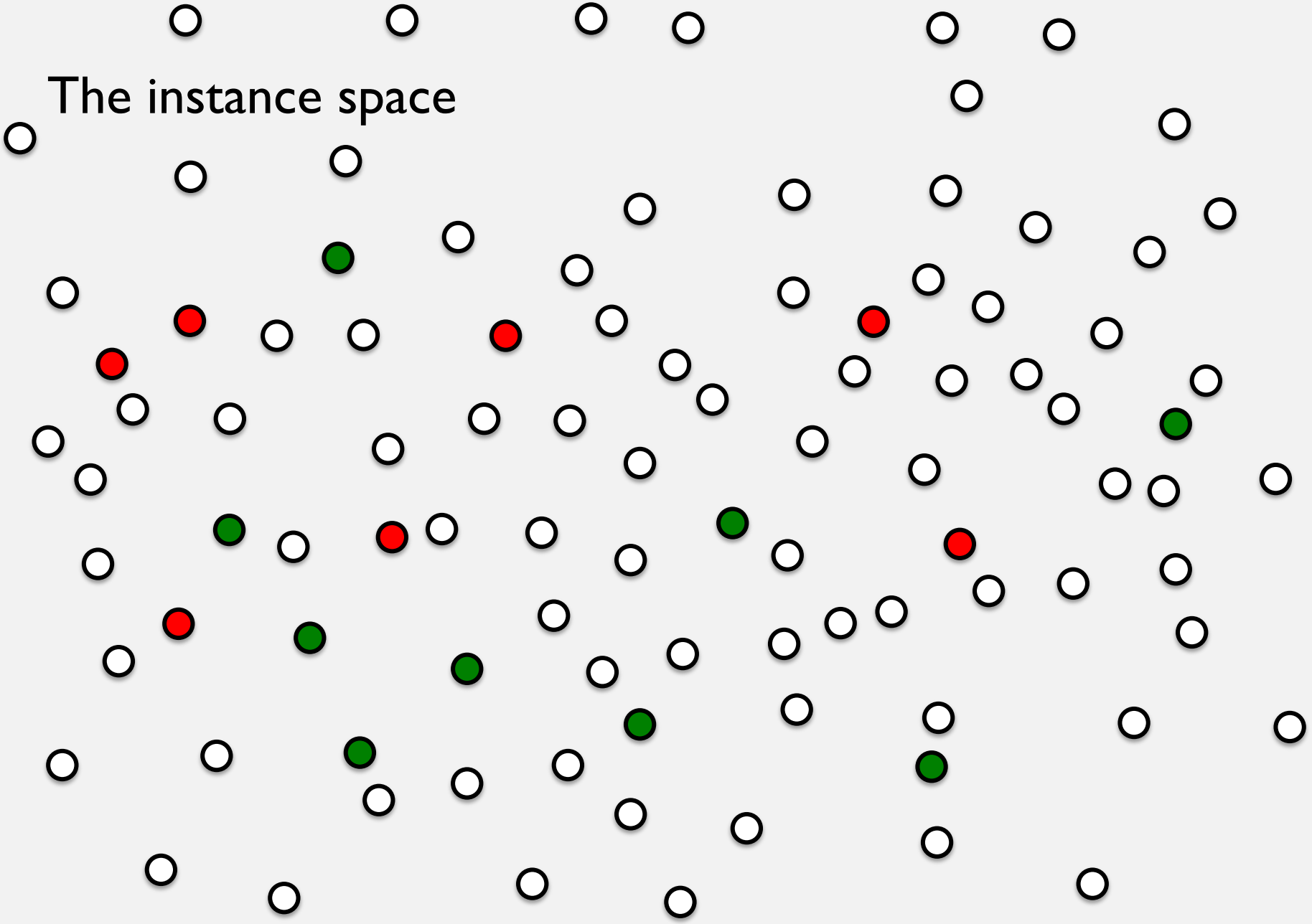
Our training data (T1)



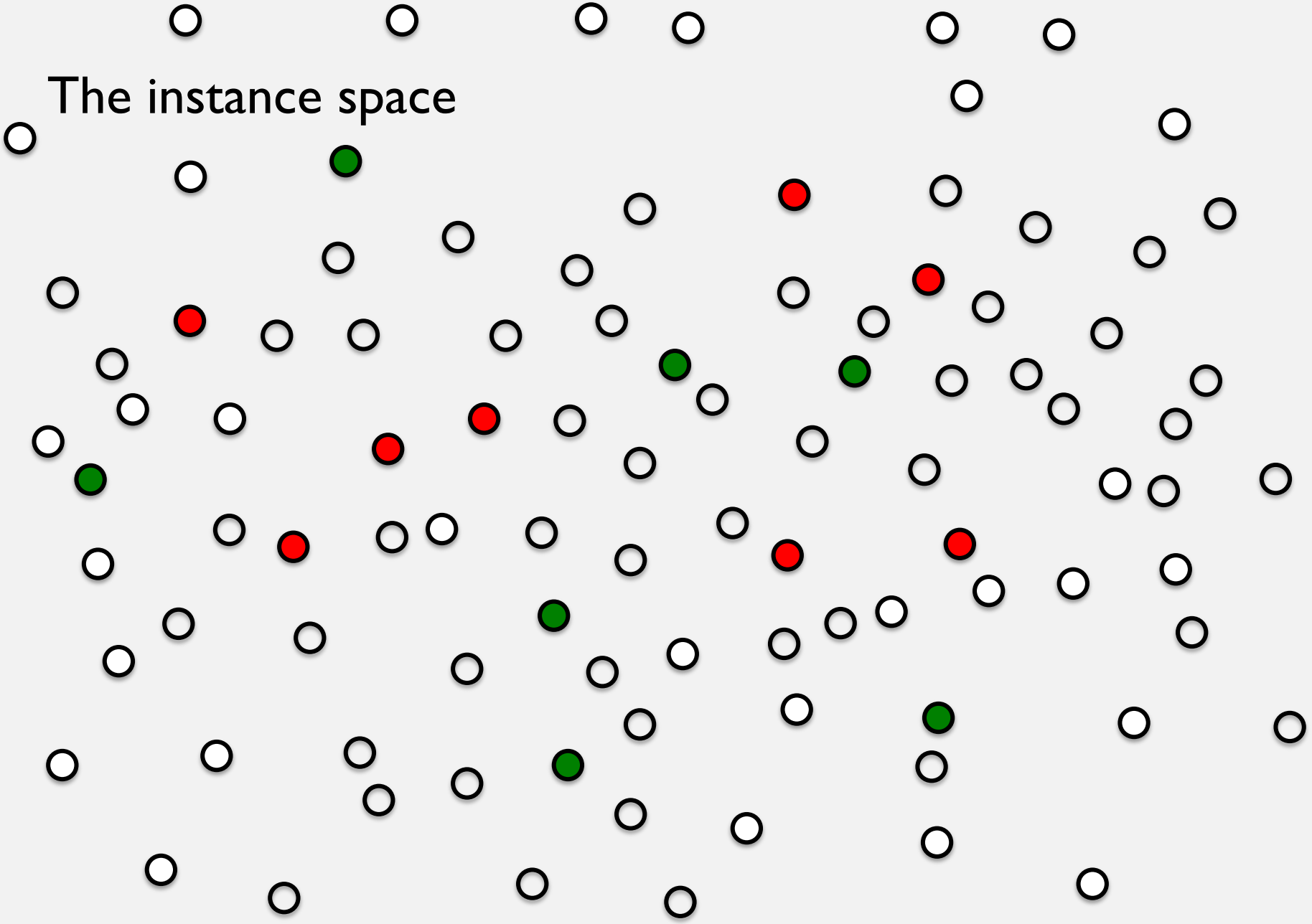
Our training data (T1)



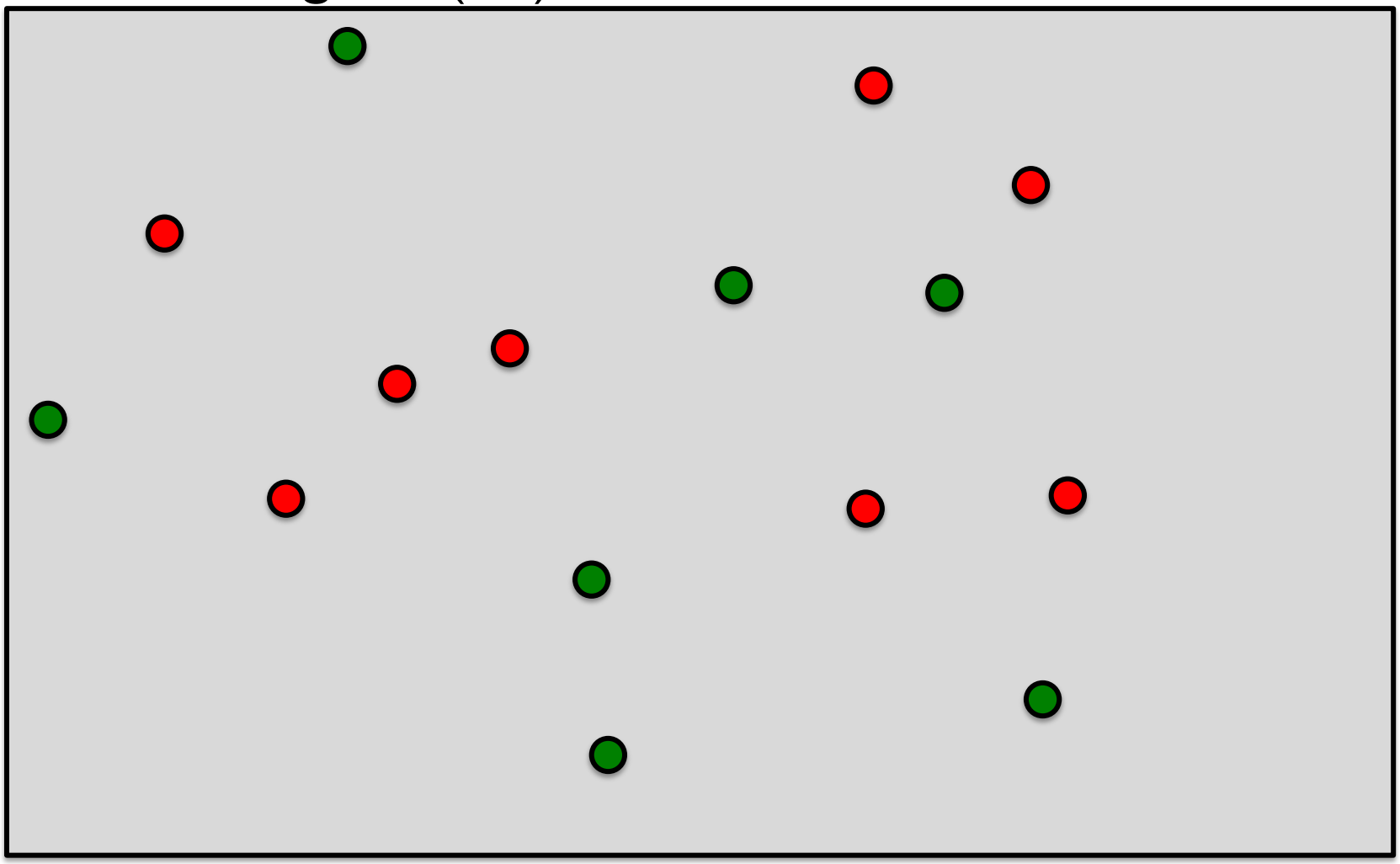
The instance space



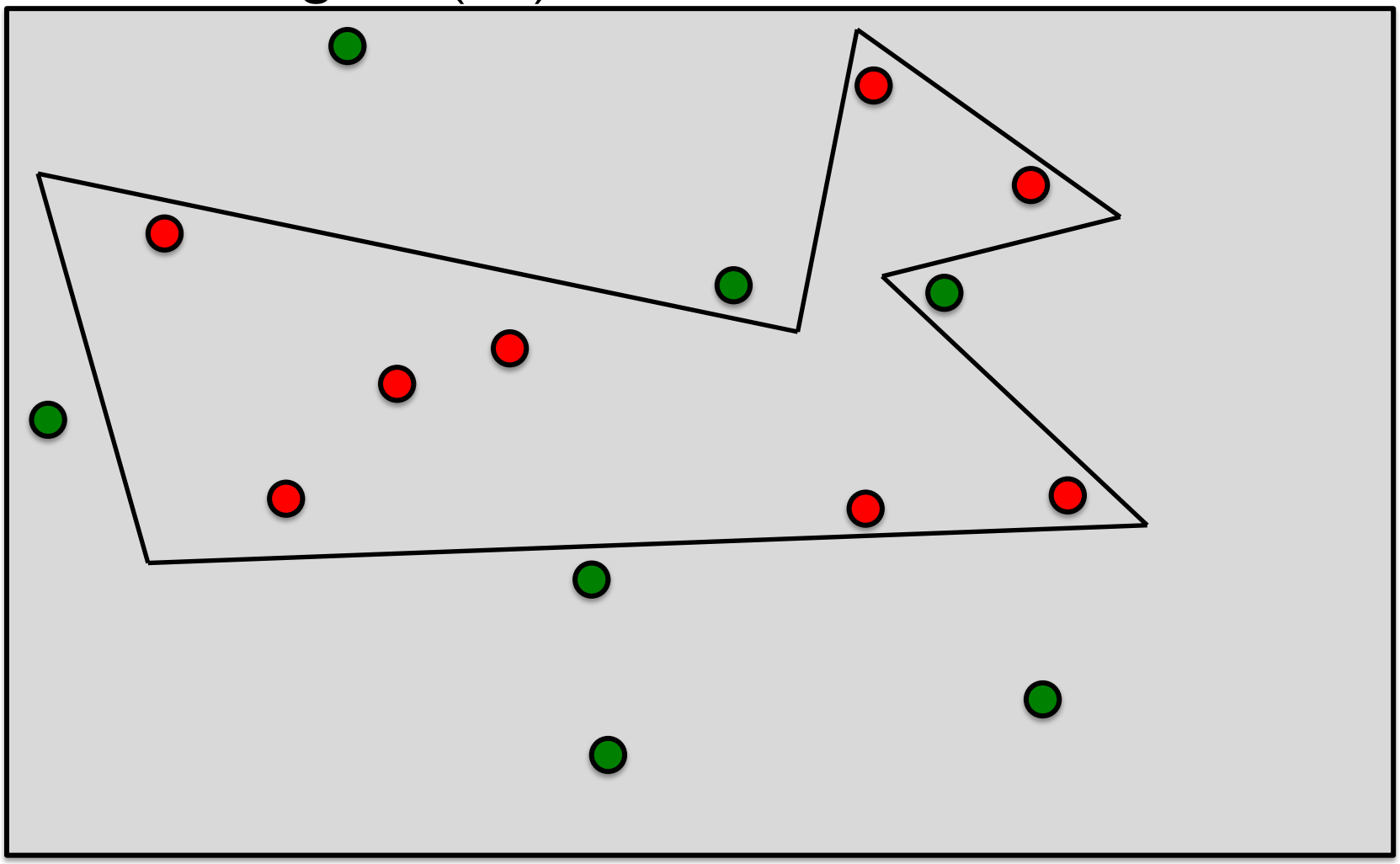
The instance space



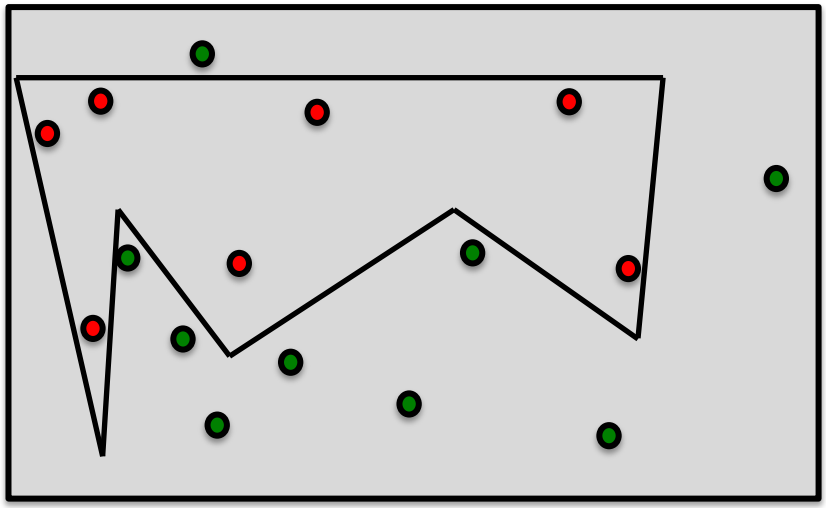
Our Training Set (T2)



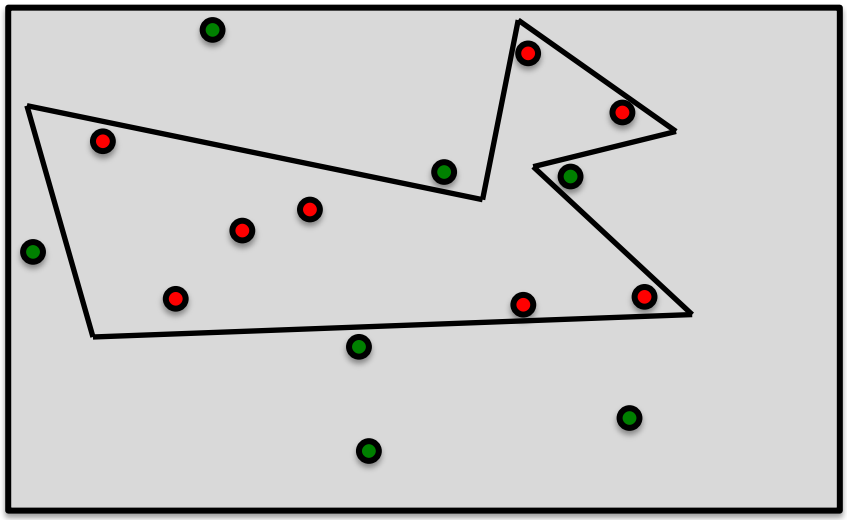
Our Training Set (T2)



High Variance



Training Set T1



Training Set T2

- T1 and T2 were different samples of the same instance space
- In principle, they represent examples for the same problem
- So, I should have learned almost identical hypothesis for both which would try to approximate the "true" decision boundary
- However, the hypotheses learned are very different, in spite of belonging to the same model family. In other words, our hypothesis changes a lot with different Ts. i.e. it has high variance

Inductive Bias

- What do we mean when we say a model has a high inductive bias?

Inductive Bias

- What do we mean when we say a model has a high inductive bias?
  - The model makes a lot of assumptions about the data.
  - Assumptions due to model family chosen *a-priori*

Bias-variance trade-off

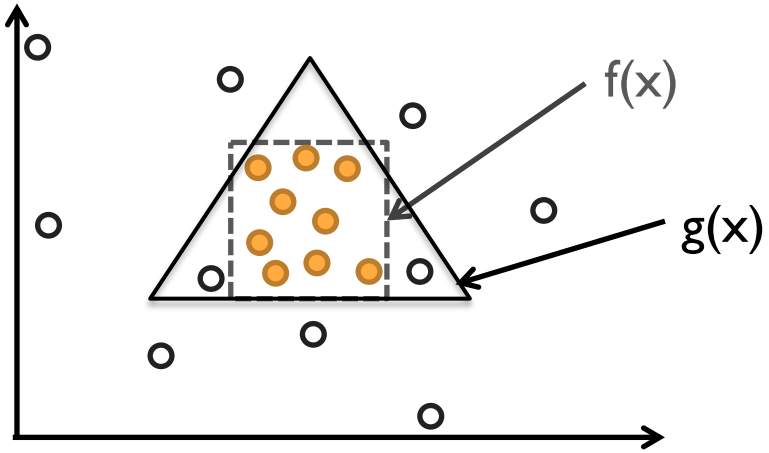
- Quality of a classifier can be decomposed into variance & bias
- Variance represents estimation error (limitations due to data)
- Bias represents approximation error (limitation of the model family)

error(g) = f(bias, variance)

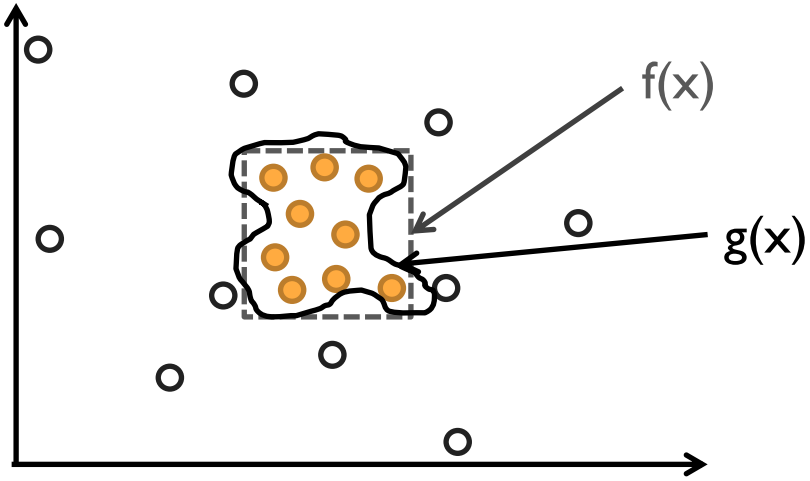
We will see this more formally
for Linear Regression

Bias-variance trade-off

- Quality of a classifier can be decomposed into variance & bias
- Variance represents **estimation error** (limitations due to data)
- Bias represents **approximation error** (limitation of the model family)



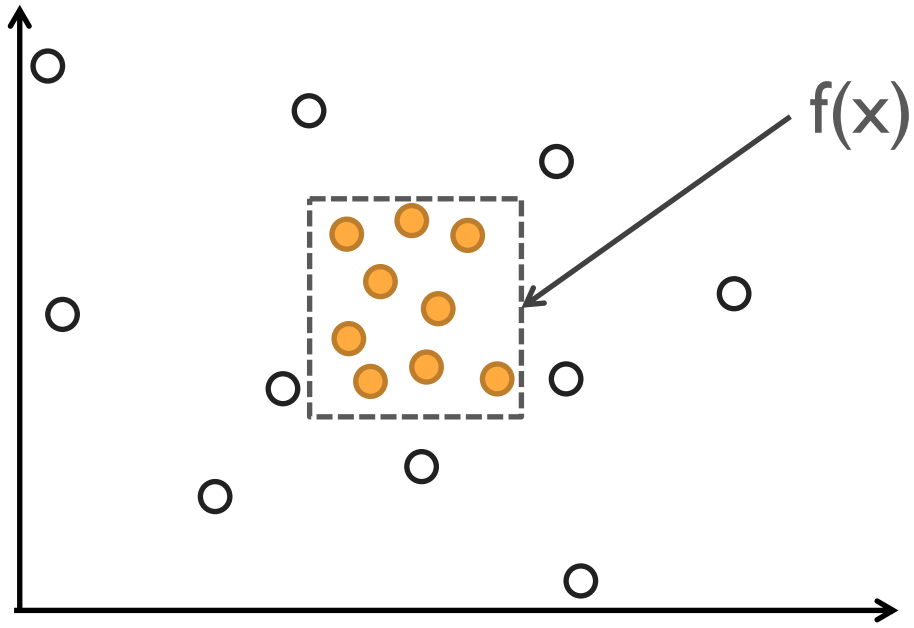
Low variance, high bias



High variance, low bias

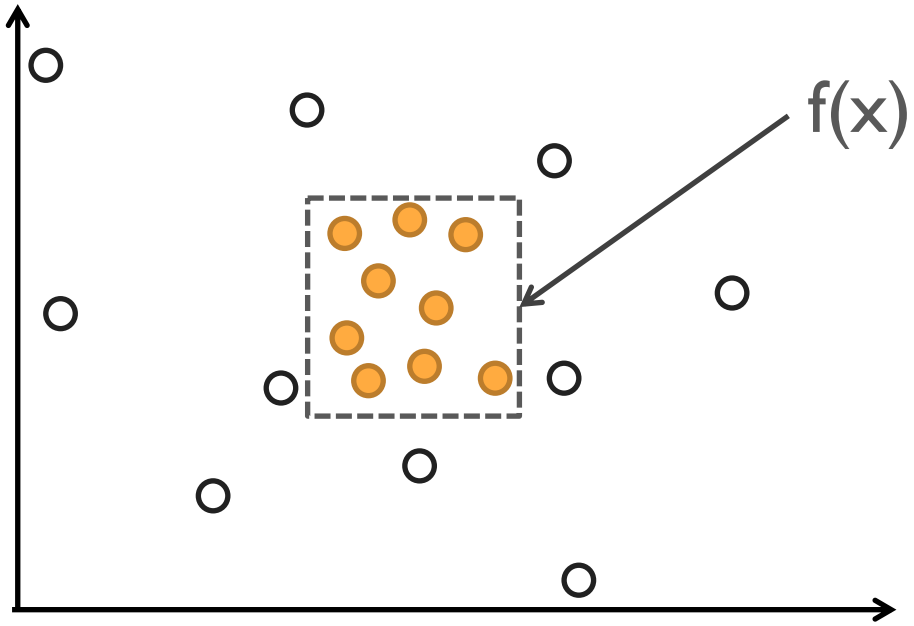
Over-fitting and Under-fitting

- Is it better to have a more complex model?



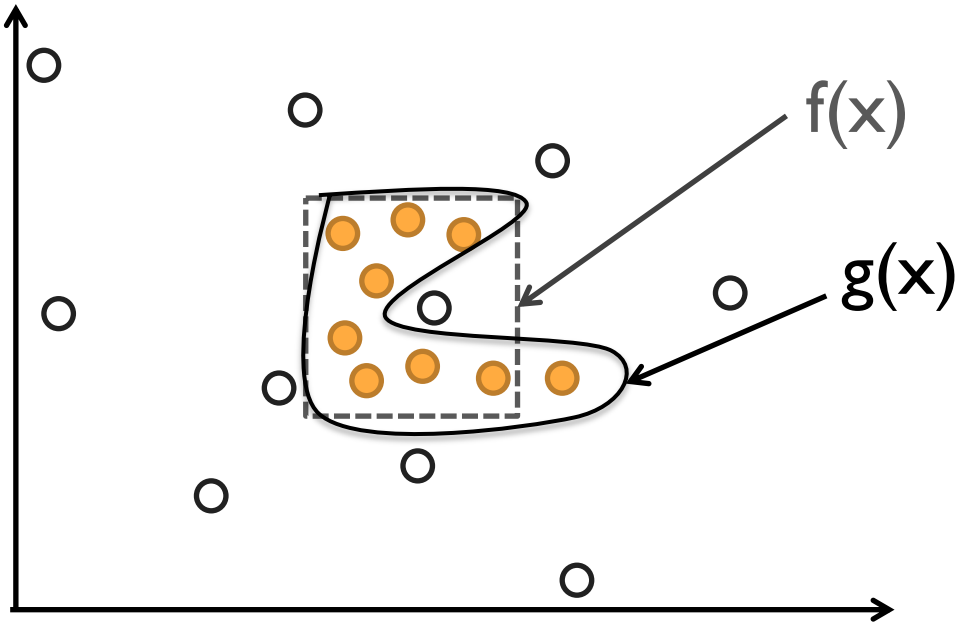
Over-fitting and Under-fitting

- Is it better to have a more complex model?
- No, follow the principle of Occam's Razor (simpler explanations are more plausible)



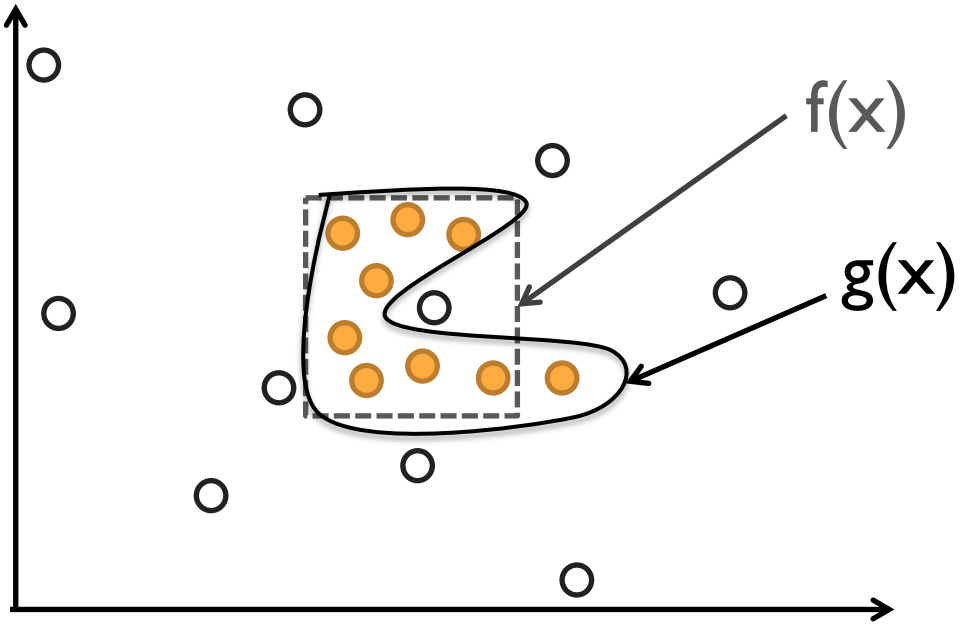
Over-fitting and Under-fitting

- Is it better to have a more complex model?
- Real world data is noisy
- You might over-fit



Over-fitting and Under-fitting

- Is it better to have a more complex model?
- real world data is noisy
- You might over-fit
- Too simple models, on the other hand, under-fit

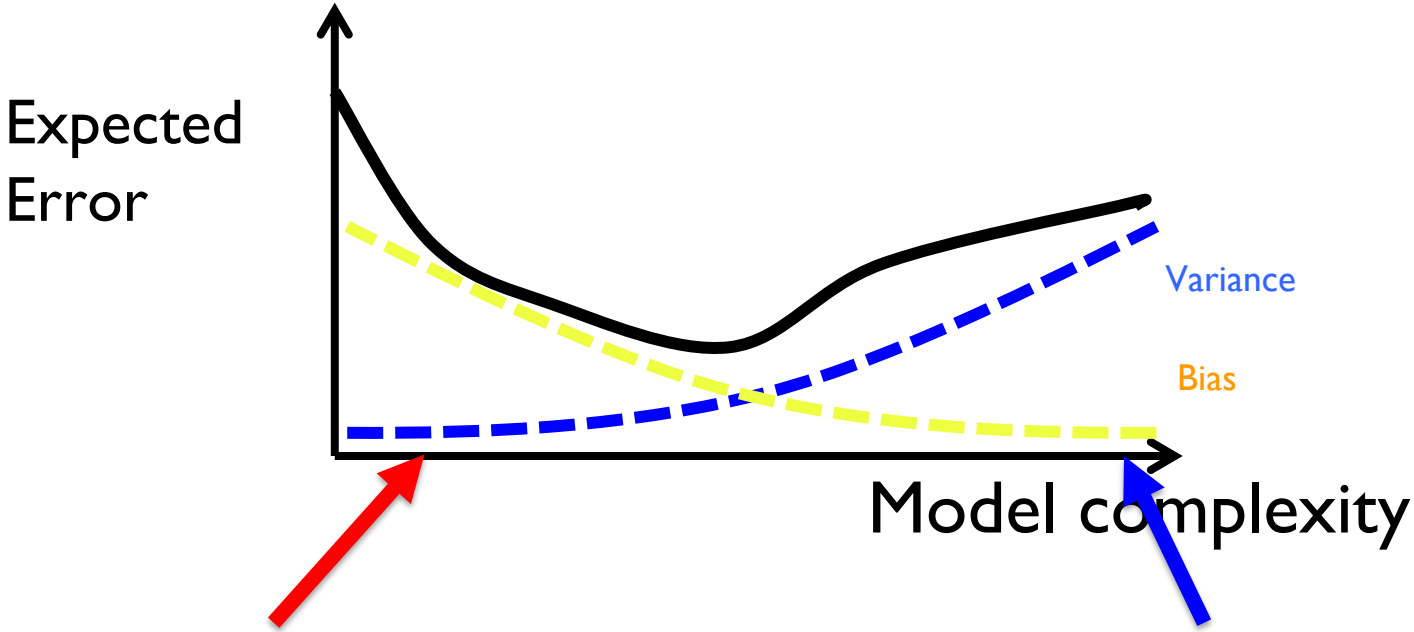


Notes on Over-fitting and Under-fitting

- Overfitting results in models that are more complex than necessary: after learning knowledge they "tend to learn noise"
- More complex models tend to have more complicated decision boundaries and tend to be more sensitive to noise, and missing examples
- Underfitting doesn't represent data well enough

We want to neither overfit nor underfit

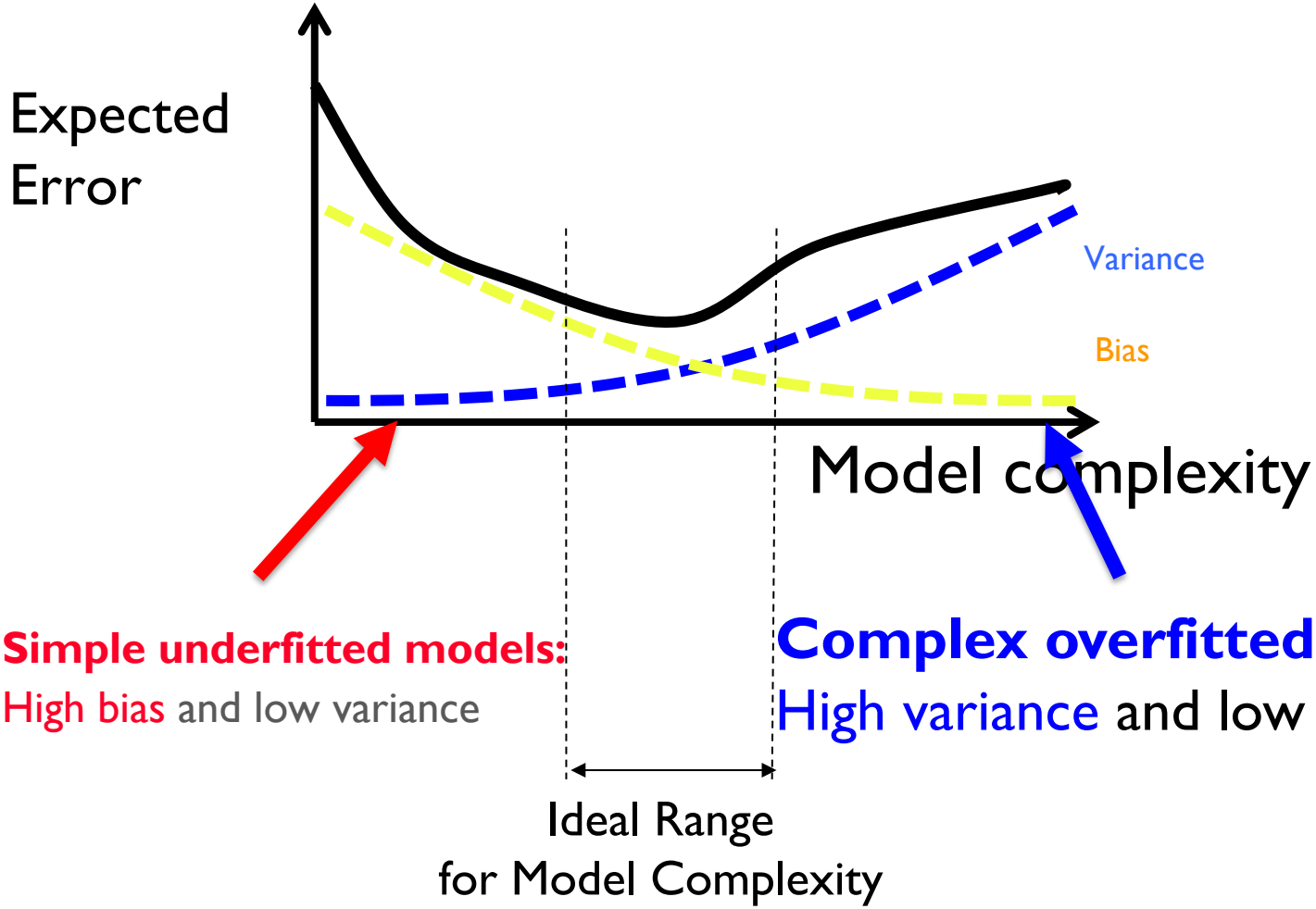
Model complexity



- **Simple underfitted models:**
  High bias and low variance

Complex overfitted models:
High variance and low bias

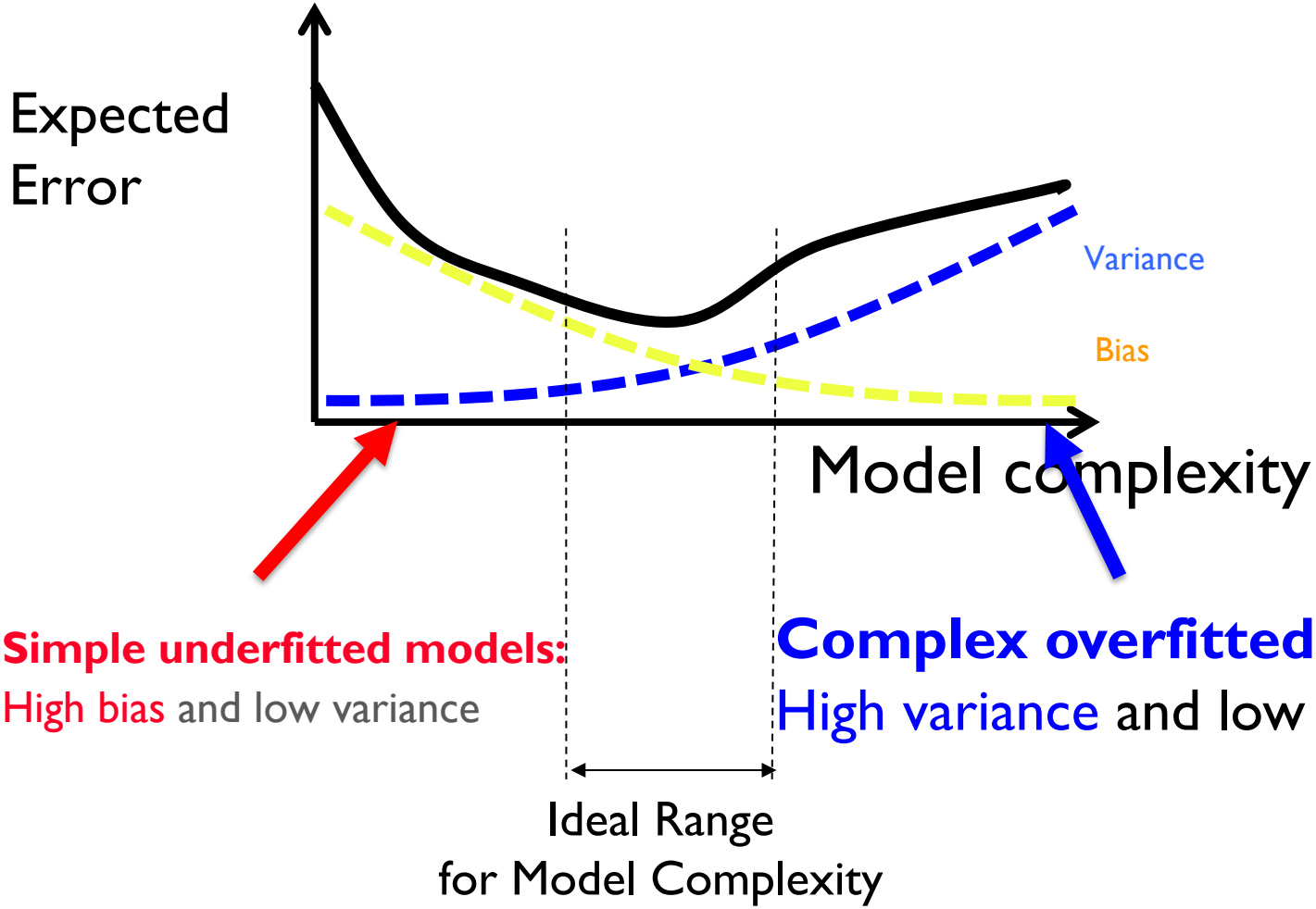
Model complexity



- **Simple underfitted models:**
  High bias and low variance

Complex overfitted models:
High variance and low bias

Model complexity



- **Simple underfitted models:**
  High bias and low variance

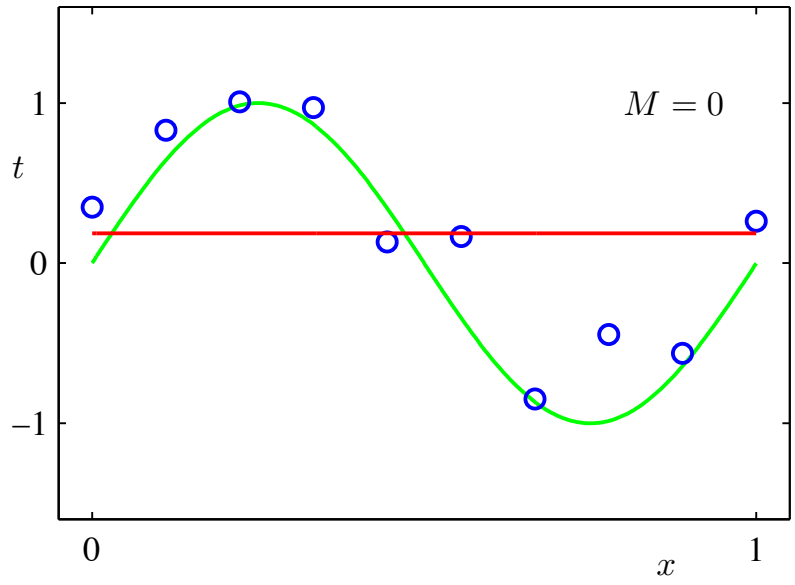
Complex overfitted models:
High variance and low bias

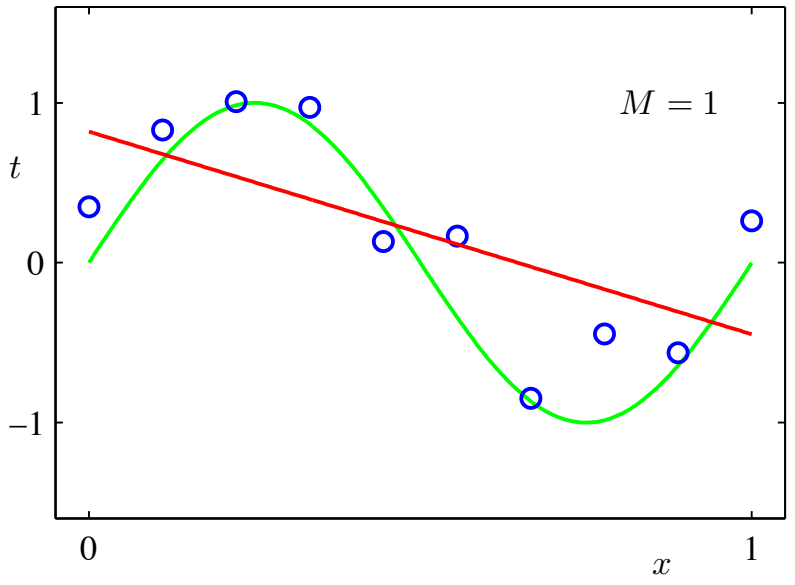
We need to find the "ideal range"

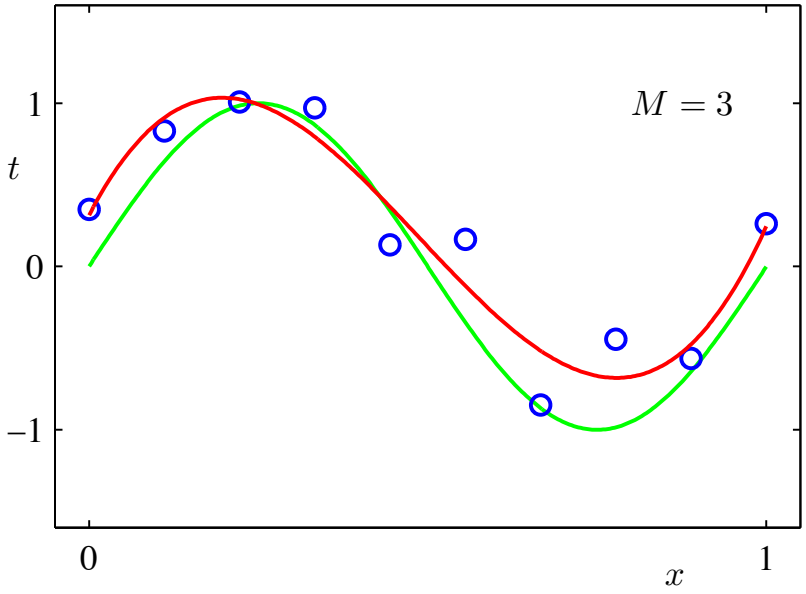
Overfitting and Underfitting

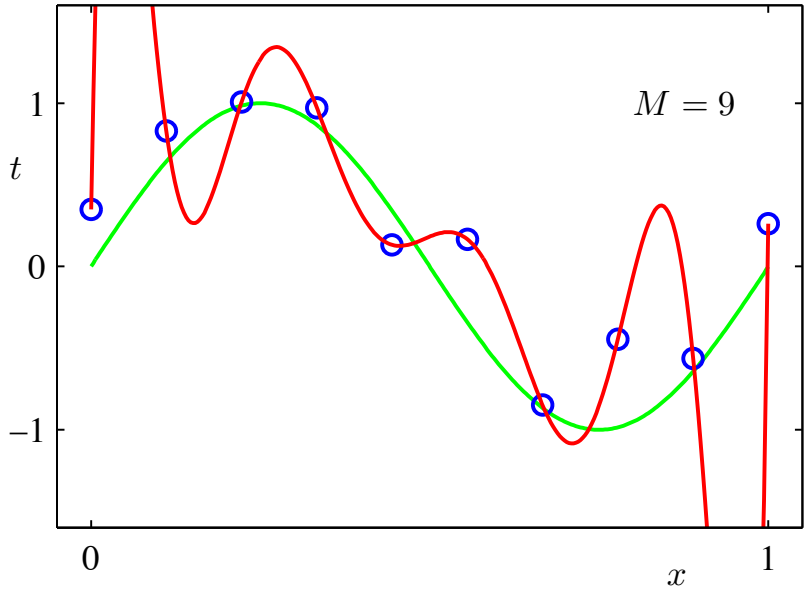
- **Overfitting** happens when the hypothesis is too complex for the "truth"
- **Underfitting** happens when the hypothesis is too simple.

Bishop fig 1.4









How to find the "ideal" range?

- Cannot find it using Training error
- Training performance is often a poor indicator of generalization
  - Keep improving train accuracy => overfit
  - Stop too early => underfit
  - Both are bad!
- Generalization is what we really care about in ML
- Test performance is a good indicator of generalization
- Testing performance is not accessible during training
- Keep a small fraction (10-20%) of the data as held-out development set

Imperfect measurements

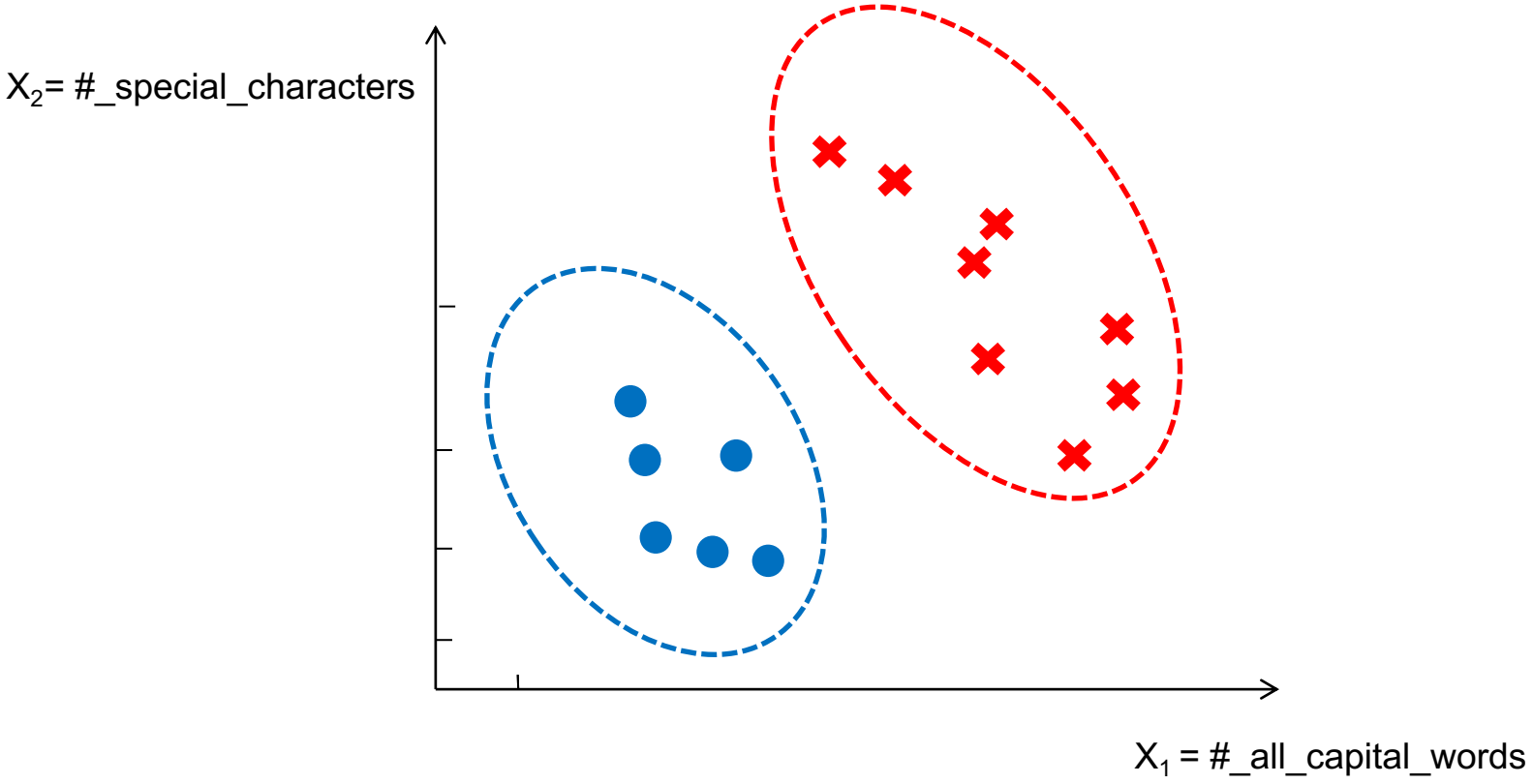
- Unmeasured Features
- Attribute noise (random or systemic)
- Label noise (random or systemic)
- Inductive bias errors may look like noise

Other kinds of supervised learning

- **Reinforcement learning -** learning a policy for influencing or reacting to environment
  - Game playing/robot in a maze, etc.
  - No supervised output, but delayed rewards
- **On-line learning**: predict on each instance in turn
- **Semi-supervised learning** uses both labeled and unlabeled data
- **Active learning** – request labels for particular instances

Unsupervised Learning

- Learning "what normally happens" in ***X***, no labels *Y*
- Clustering: Grouping similar instances



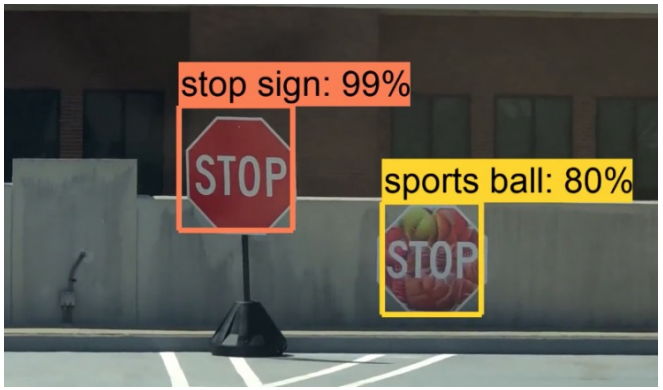
Unsupervised Learning

- Example applications
  - Segmentation in customer relationship mgmt
  - Image compression: Color quantization
  - Identifying unusual Airplane landings
  - Autoencoding – learn the "features"

```
Assignment 4: applying unsupervised
learning to solve supervised learning
problem
```

Other topics

- Security
- Privacy
- Bias



`Assignment 5: explore data bias`

Evaluation: Measuring Predictive Performance

Measuring Predictive Performance

- Various evaluation measures exist in literature that can evaluate predictive performance
- Most popular for classification:
  - Accuracy and Error Rate
  - Precision Recall and F - measure

Accuracy and Error Rate

$$\text{accuracy} = \frac{\#\text{ correct predictions}}{\#\text{ test instances}}$$

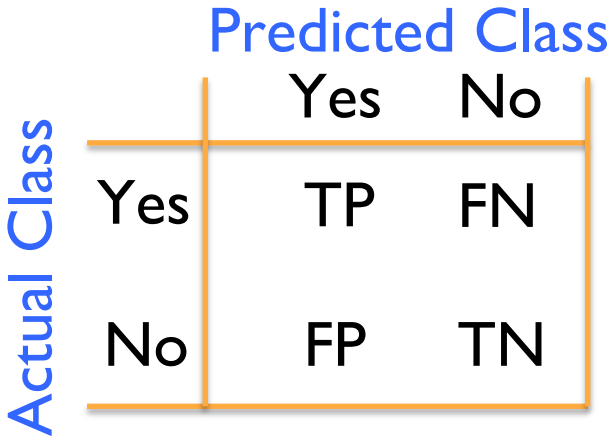
$$\text{error} = 1 - \text{accuracy} = \frac{\#\text{ incorrect predictions}}{\#\text{ test instances}}$$

Useful in "X vs Y" type of problems: both classes are equally important

Examples?

Confusion Matrix

- Given a dataset of positive instances and negative instances:



TP = True Positives
TN = True Negatives
FP = False Positives
FN = False Negatives

Example

	Predicted: Yes	Predicted: No
Actual: Yes	TP = 100	FN=5
Actual: No	FP = 10	TN = 50

Total number of instances in the corpus, N = ??

Total number of predicted yes's = ??
Total number of predicted no's = ??

Total number of actual yes's = ??
Total number of actual no's = ??

Example

N = 165	Predicted: Yes	Predicted: No	Total
Actual: Yes	TP = 100	FN=5	105
Actual: No	FP = 10	TN = 50	60
Total	110	55	

Total number of instances in the corpus, N = 165

Total number of predicted yes's = 110
Total number of predicted no's = 55

Total number of actual yes's = 105
Total number of actual no's = 60

Example

N = 165	Predicted: Yes	Predicted: No	Total
Actual: Yes	TP = 100	FN=5	105
Actual: No	FP = 10	TN = 50	60
Total	110	55	

Accuracy = ??

Example

N = 165	Predicted: Yes	Predicted: No	Total
Actual: Yes	TP = 100	FN=5	105
Actual: No	FP = 10	TN = 50	60
Total	110	55	

Accuracy = (TP+TN)/N = (100+50)/165 = 0.91

Precision, Recall and F-measure

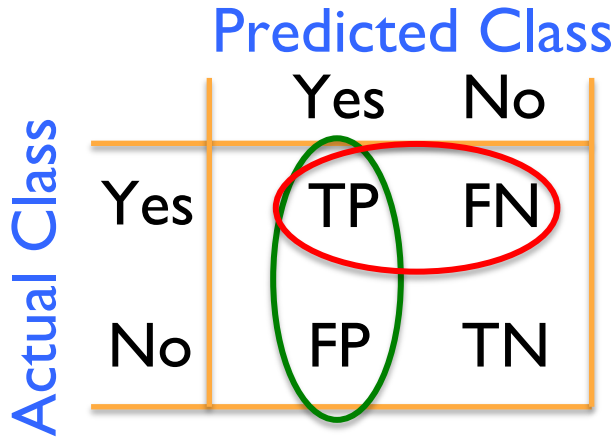
- Accuracy is useful in "X versus Y" type of problems

$$\text{accuracy} = \frac{\#\ \text{correct predictions}}{\#\ \text{test instances}}$$

- What about "X versus non-X"type of problems: one of the classes is more interesting
  - E.g. spam versus not-spam
  - E.g. information retrieval: Retrieve all relevant documents from a given list of documents

Precision, Recall and Accuracy

- Given a dataset of P positive instances and N negative instances:



$$\text{accuracy} = \frac{TP + TN}{P + N}$$

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

Precision measures what fraction of the retrieved documents were relevant

Recall measures what fraction of the relevant records were retrieved

F measure

$$\text{F1 measure} = \frac{2 * P * R}{P + R}$$

Inverse relationship between Precision and Recall

Example

N = 165	Predicted: Yes	Predicted: No	Total
Actual: Yes	TP = 100	FN=5	105
Actual: No	FP = 10	TN = 50	60
Total	110	55	

Accuracy = (TP+TN)/N = (100+50)/165 = 0.91

Precision = ??
Recall = ??
F1 score = ??

Example

N = 165	Predicted: Yes	Predicted: No	Total
Actual: Yes	TP = 100	FN=5	105
Actual: No	FP = 10	TN = 50	60
Total	110	55	

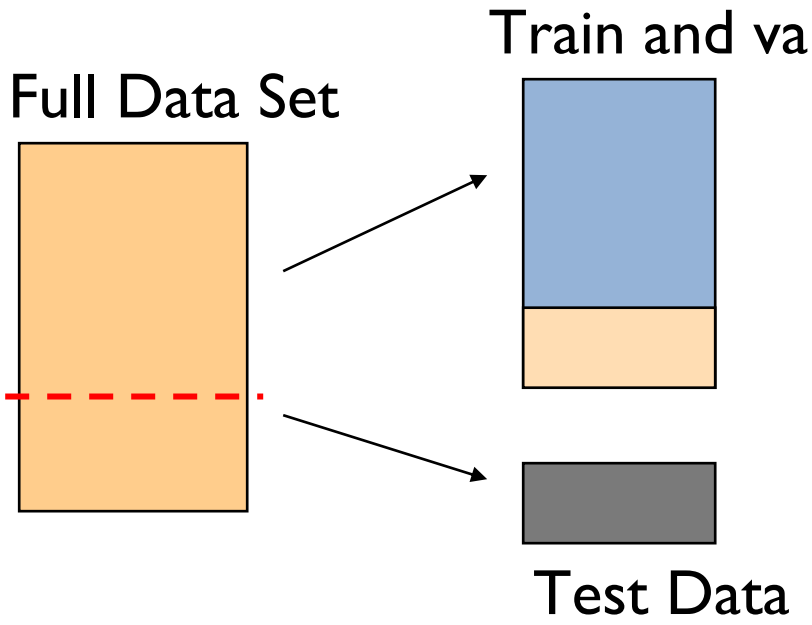
Accuracy = (TP+TN)/N = (100+50)/165 = 0.91

Precision = TP/(TP + FP) = 100/110 = 0.91
Recall = TP /(TP + FN) = 100/105 = 0.95
F1 score = 2*0.91*0.95/(0.91+0.95) = 0.93

How to report performance?

- Train data: data used to build the model

  Use held-out validation/dev set if needed
- Test data: new data, not used in the training process



Usually 80-20 split

Idea:
Train a model on the "training data"...

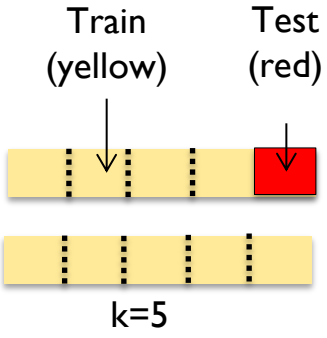
...AND THEN test model's performance on the test data

** In perfect world, you wouldn't have to hold out a test set, but instead have access to a human evaluator

k-fold cross validation

- Why just choose one particular "split" of the data as test set?
  - In principle, we should do this multiple times since performance may be different for each split
    - Run k iterations
    - In each iteration, hold out a different portion of the data as test set
  - This would improve robustness of the reported result

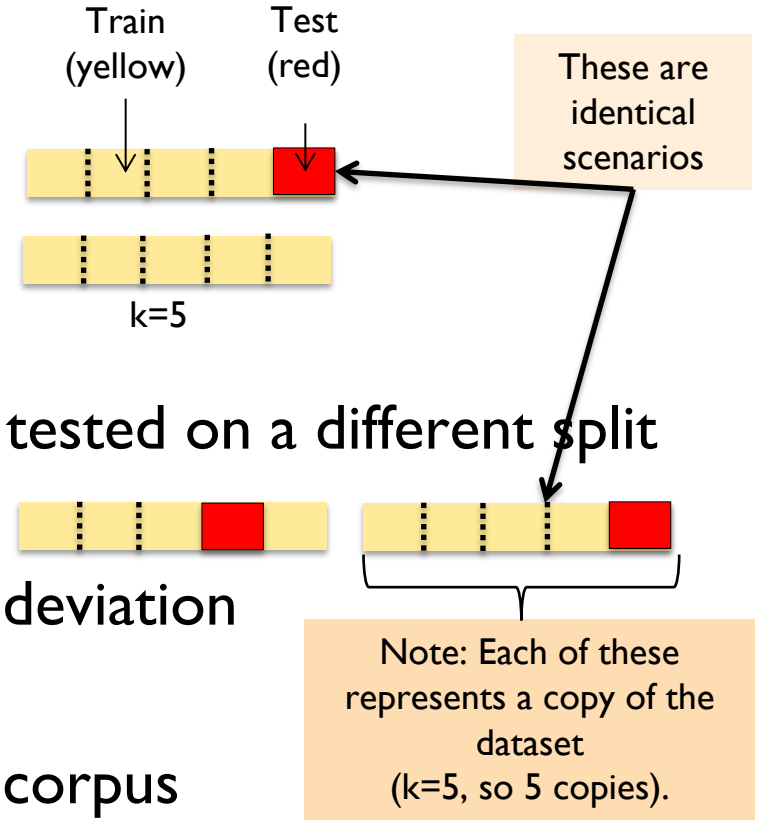
k-fold cross validation



- Instead of testing on only one split:
- Split data into k equal-sized parts:
- Train and test k **different** classifiers
- Each of the k classifier is trained and tested on a different split

- Report average accuracy & standard deviation of the accuracies across all k splits
- Leave-one-out(LOO) CV: k= size of corpus

k-fold cross validation



- Instead of testing on only one split:
- Split data into k equal-sized parts:
- Train and test k **different** classifiers
- Each of the k classifier is trained and tested on a different split

- Report average accuracy & standard deviation of the accuracies across all k splits
- Leave-one-out(LOO) CV: k= size of corpus