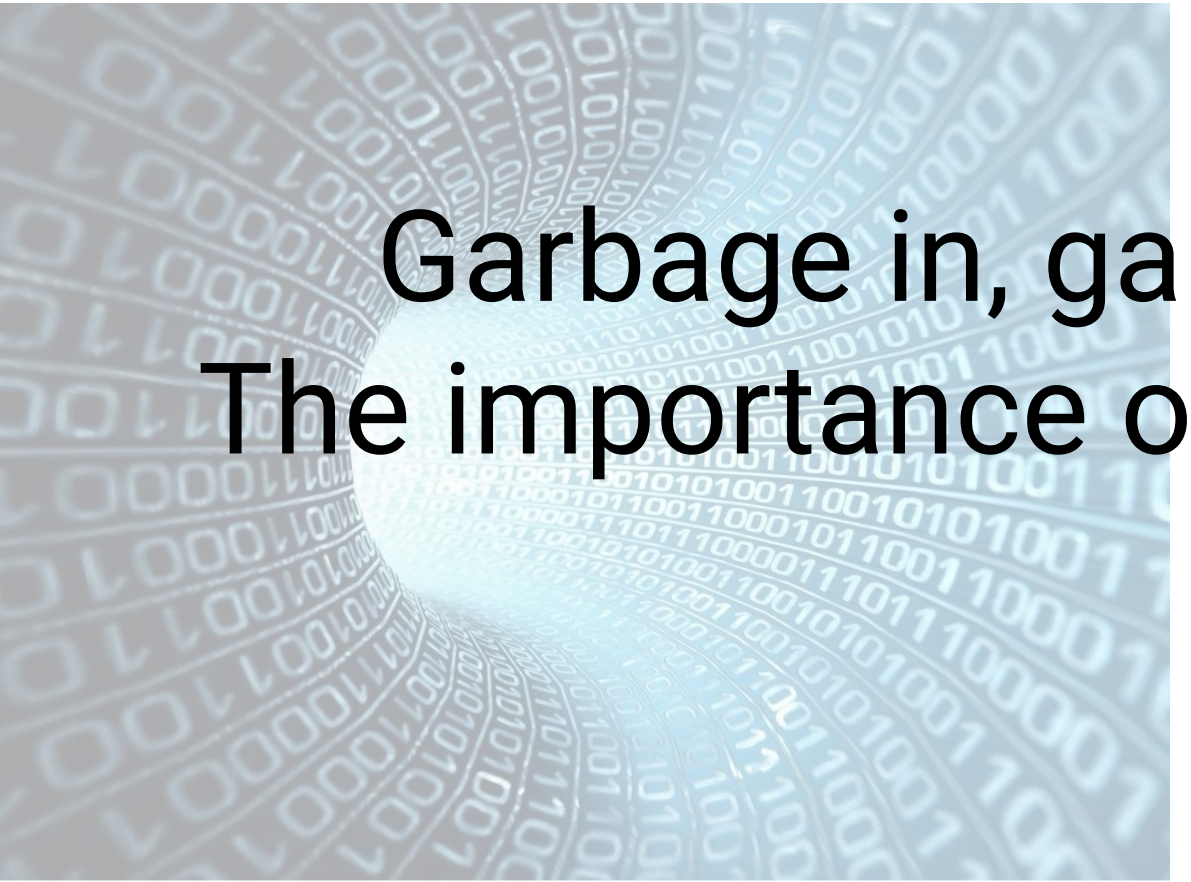
Class core values

1. Be **respect**ful to yourself and others
2. Be **confident** and believe in yourself
3. Always do your **best**
4. Be **cooperative**
5. Be **creative**
6. Have **fun**
7. Be **patient** with yourself while you learn
8. Don’t be shy to **ask “stupid” questions**
9. Be **inclusive** and **accepting**



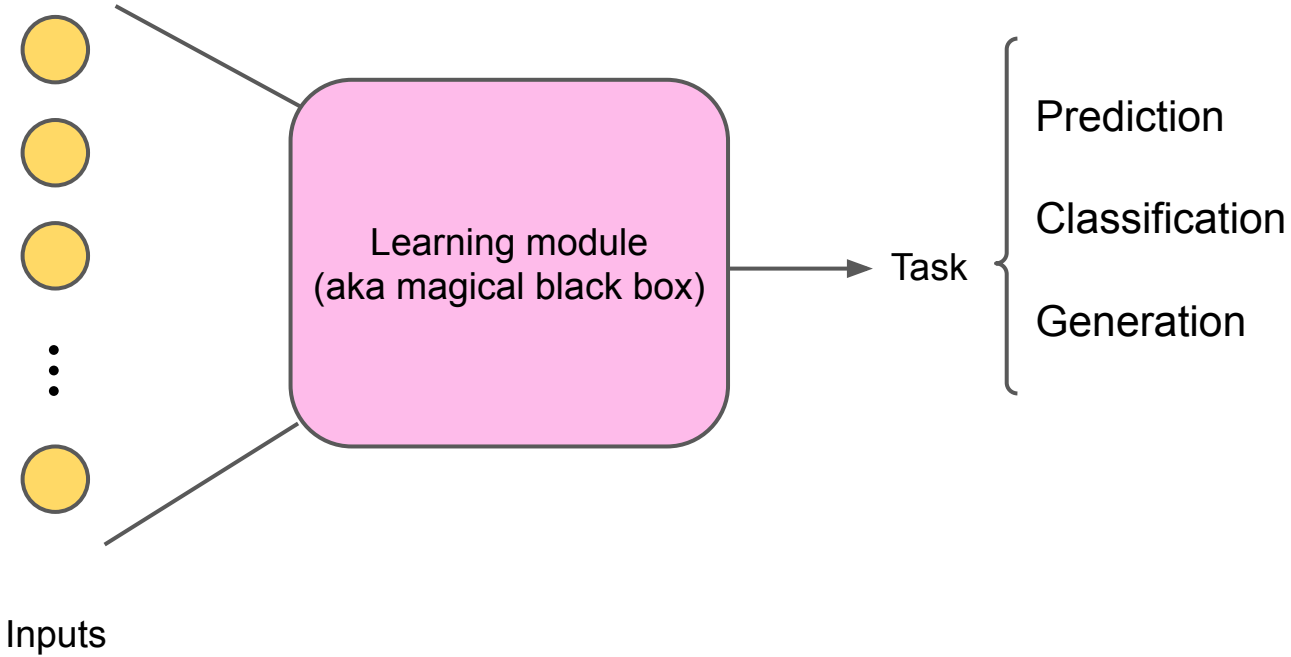
Week 1, Lecture 2

Garbage in, garbage out – The importance of input data

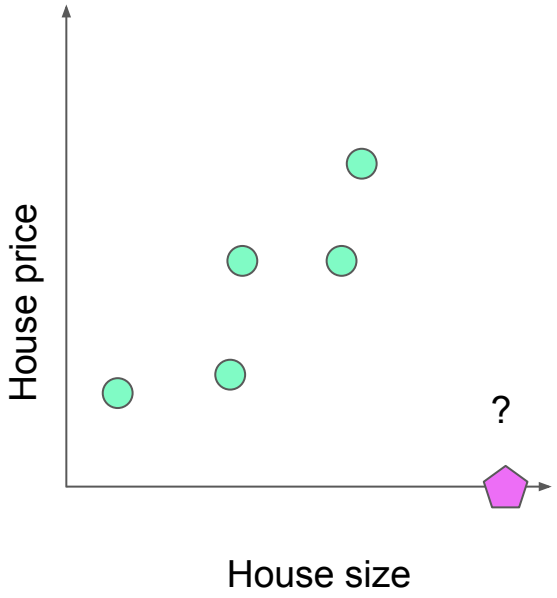
Learning Objectives

1. Apply simple statistical tools to find main features of the data
2. Describe the importance of data preparation for inputs
3. Apply python tools to prepare and clean data
4. Describe the importance of distribution in data collection and train/test split
5. Perform train/test/split on data
6. Describe some of the databases for obtaining protein data

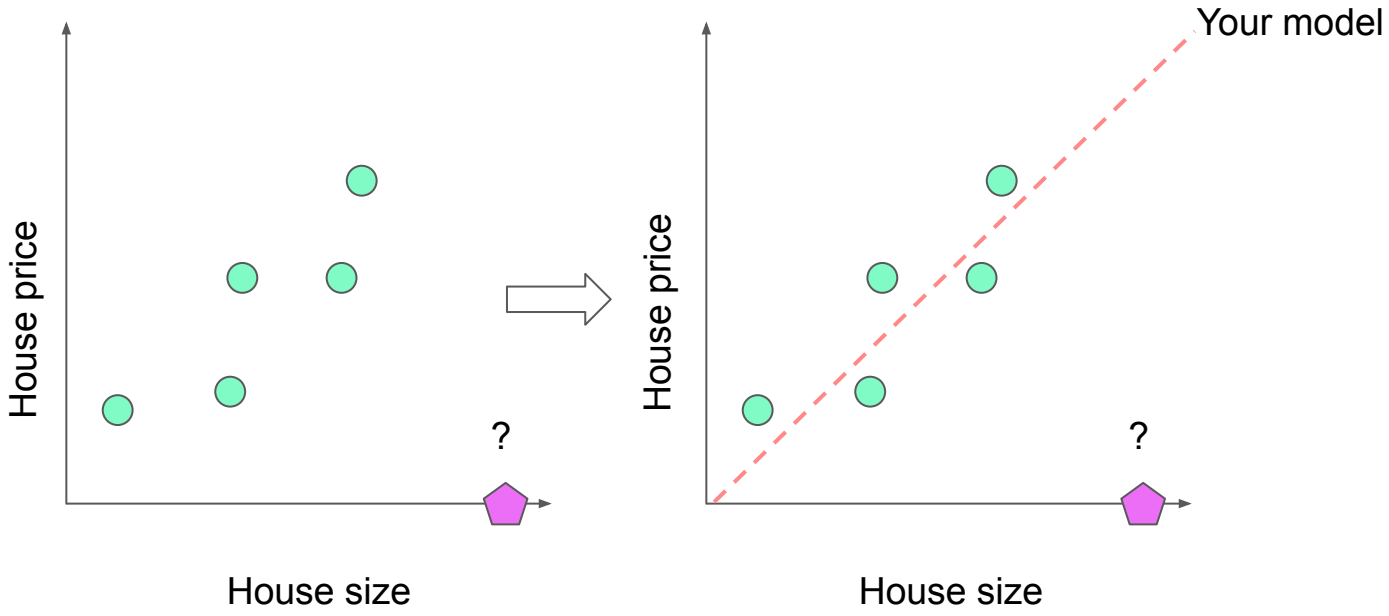
The basic components of a learning system



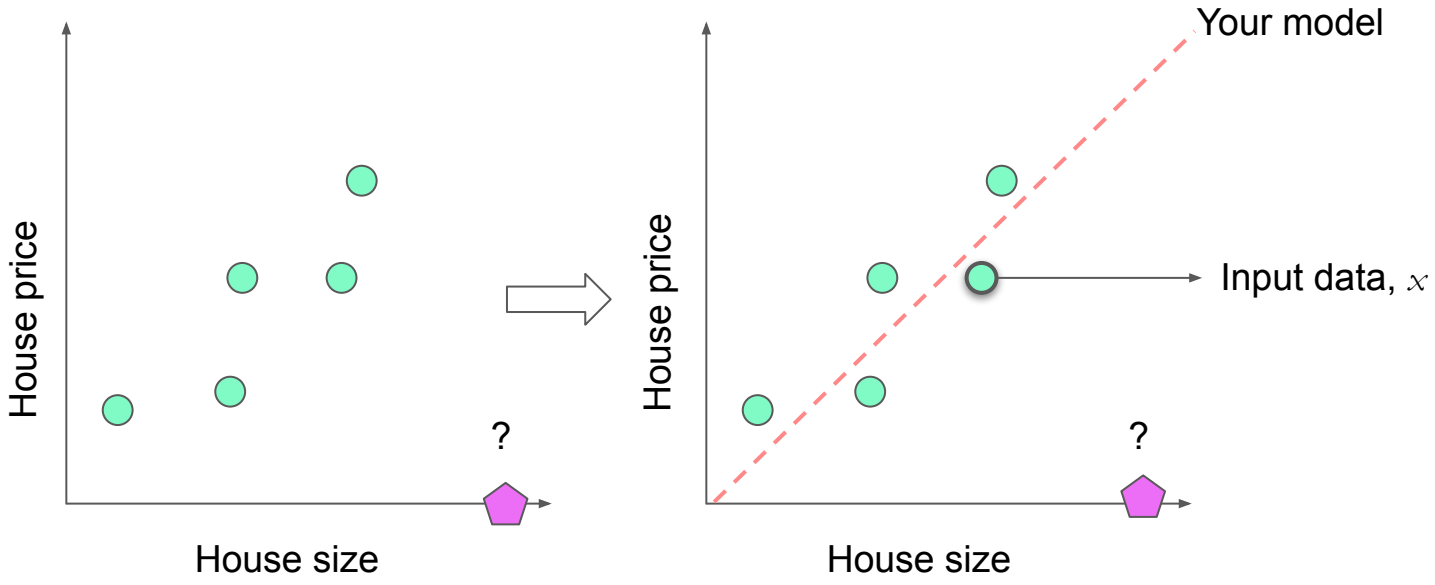
A simple learning case



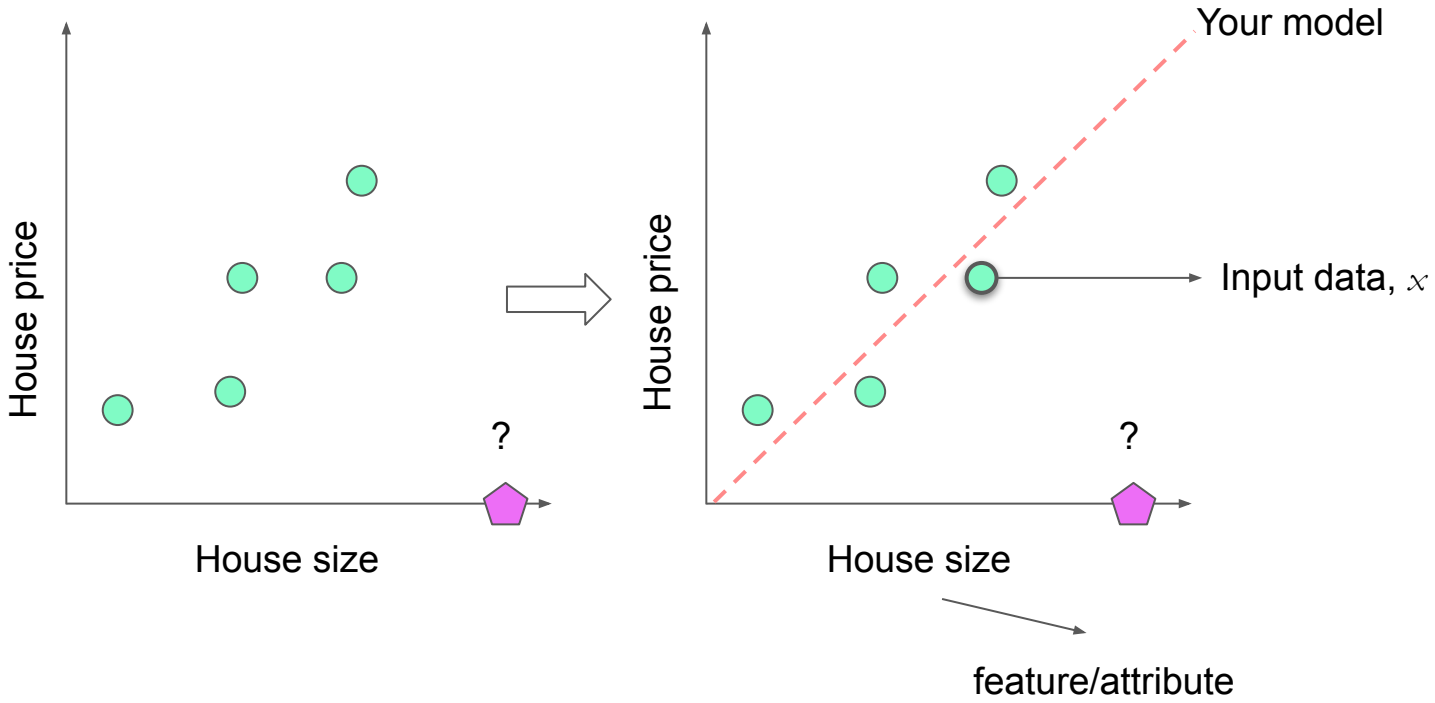
A simple learning case



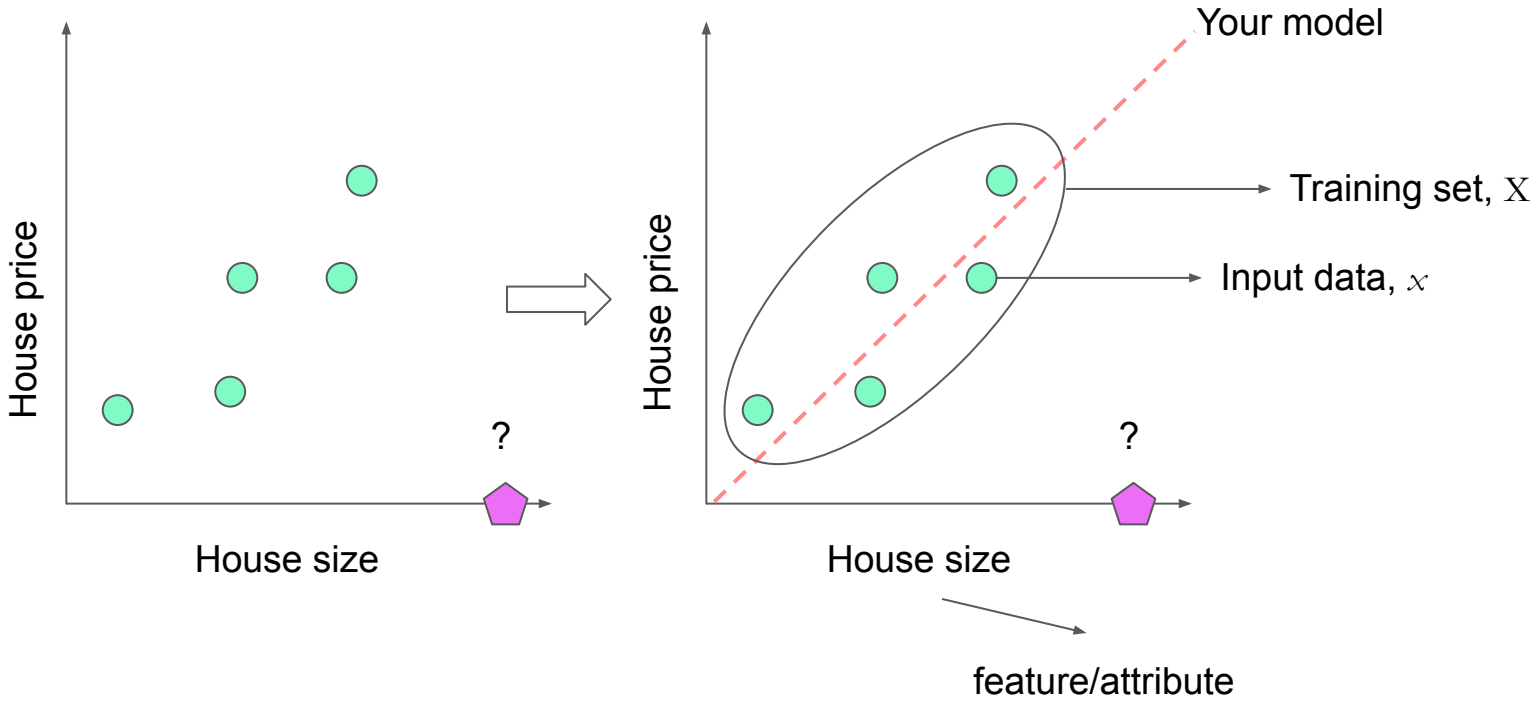
A simple learning case



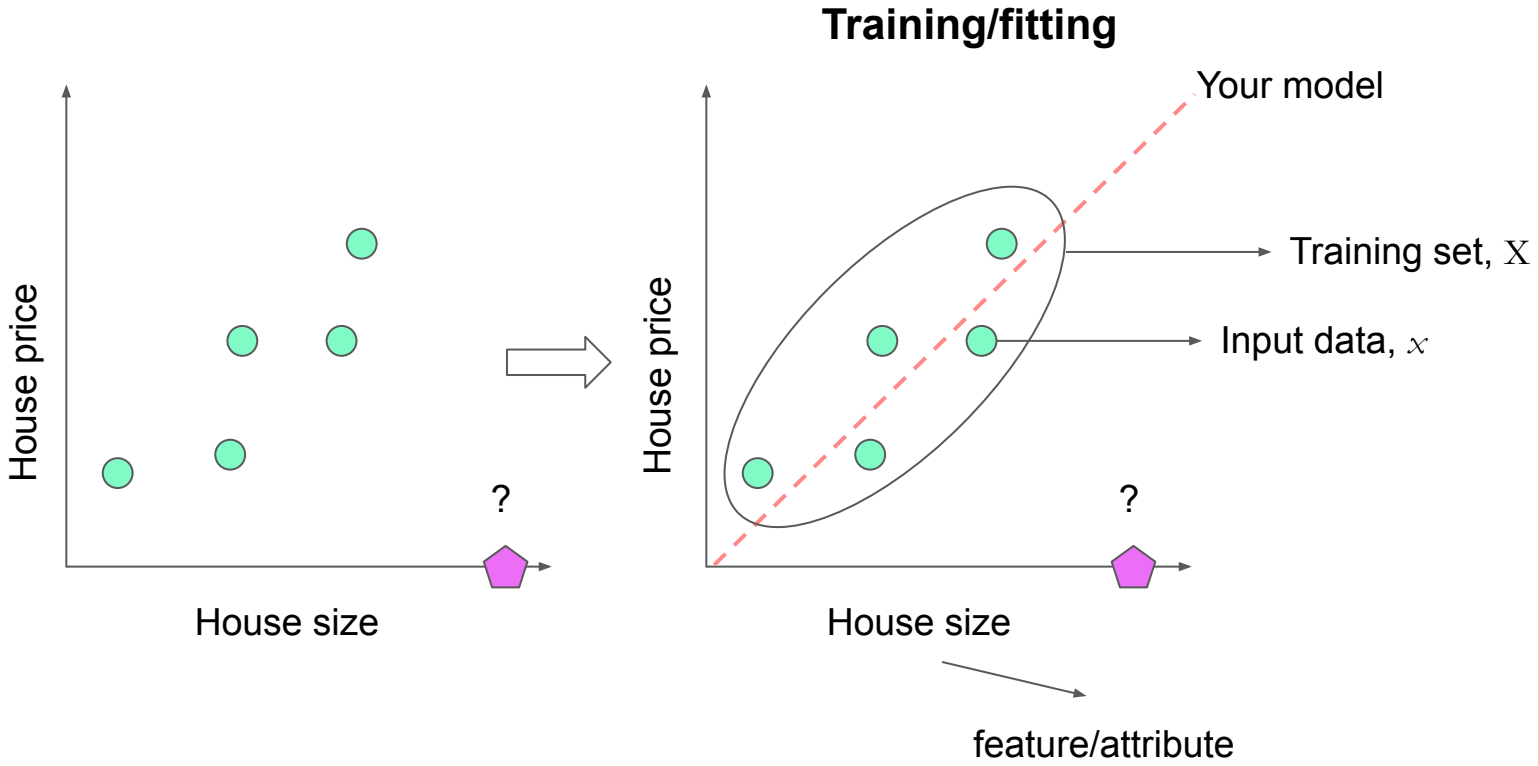
A simple learning case



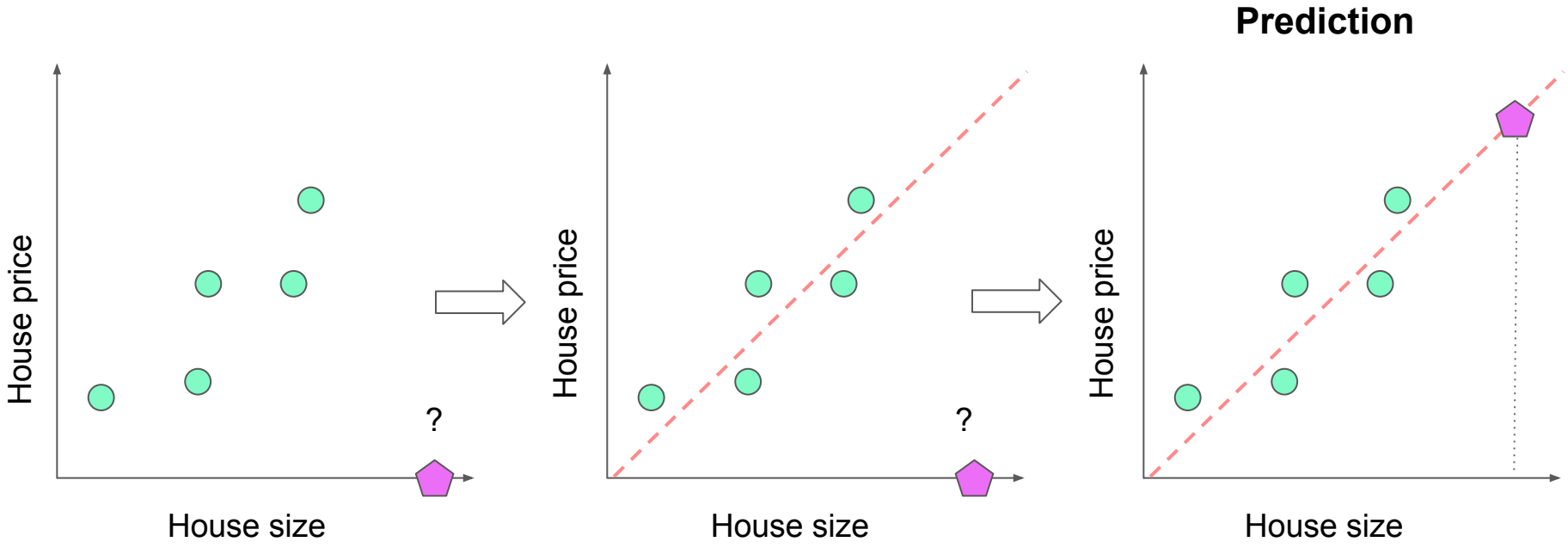
A simple learning case



A simple learning case



A simple learning case

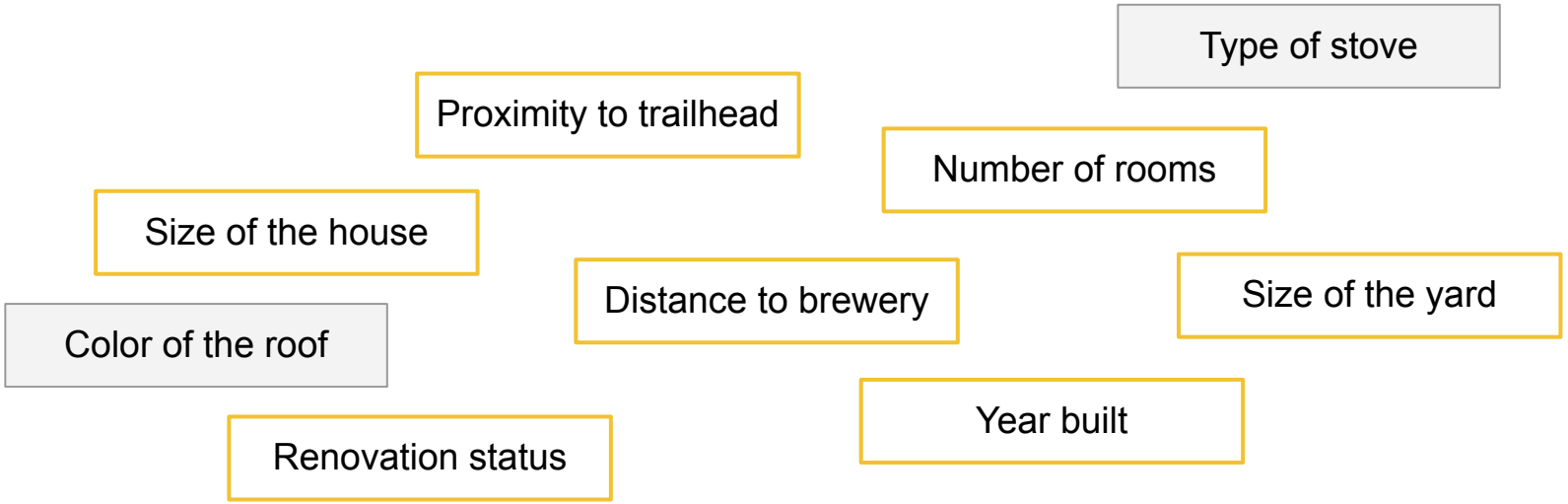


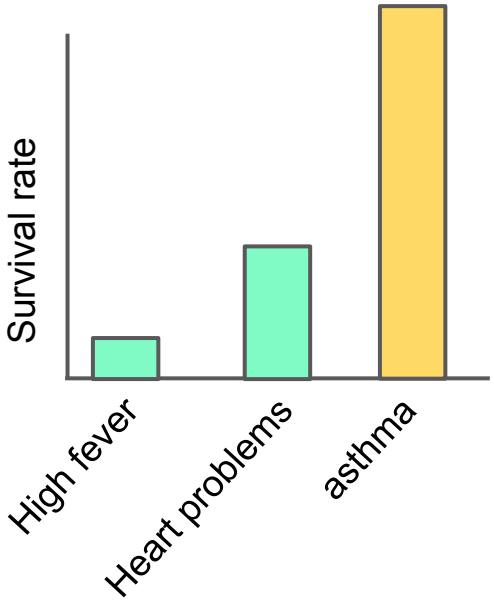
Finding important features

Housing price prediction in Eugene

Finding important features

Housing price prediction in Eugene





Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission

Rich Caruana
Microsoft Research
rcaruana@microsoft.com

Yin Lou
LinkedIn Corporation
ylou@linkedin.com

Johannes Gehrke
Microsoft
johannes@microsoft.com

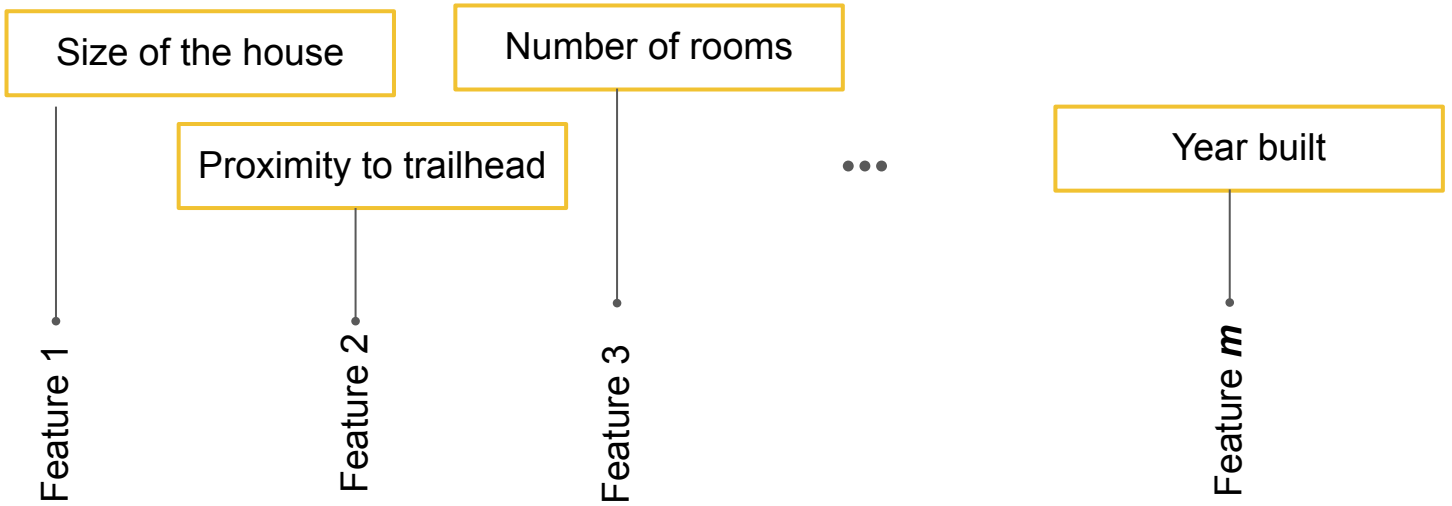
Paul Koch
Microsoft Research
paulkoch@microsoft.com

Marc Sturm
NewYork-Presbyterian Hospital
mas9161@nyp.org

Noémie Elhadad
Columbia University
noemie.elhadad@columbia.edu

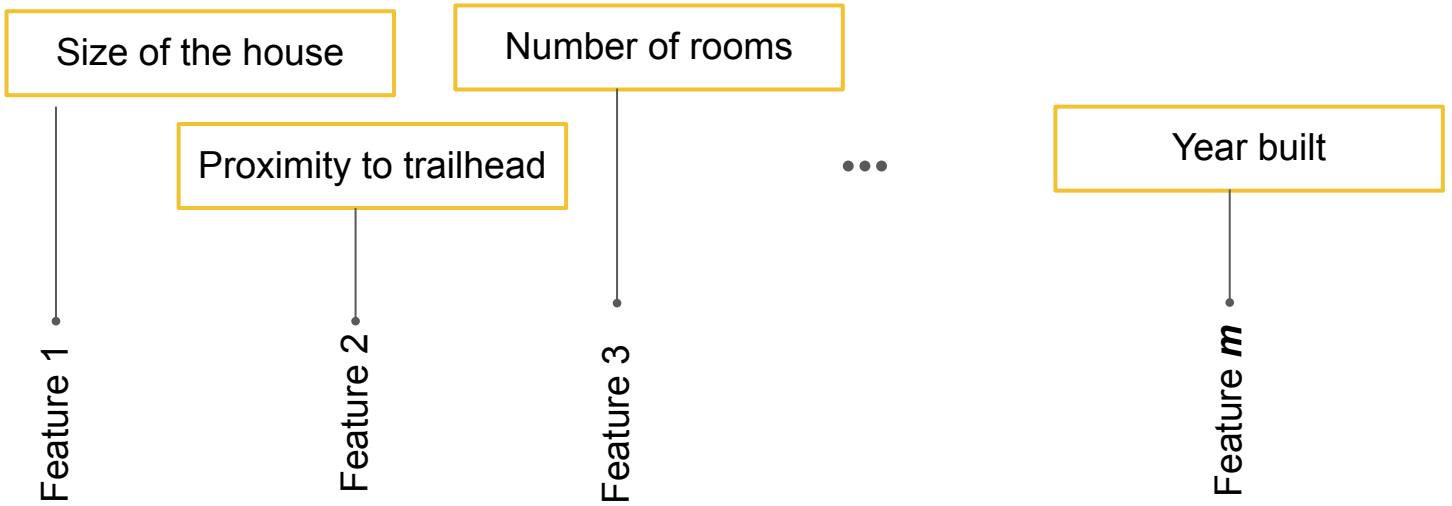
Preparing input data

Housing price prediction in Eugene



Preparing input data

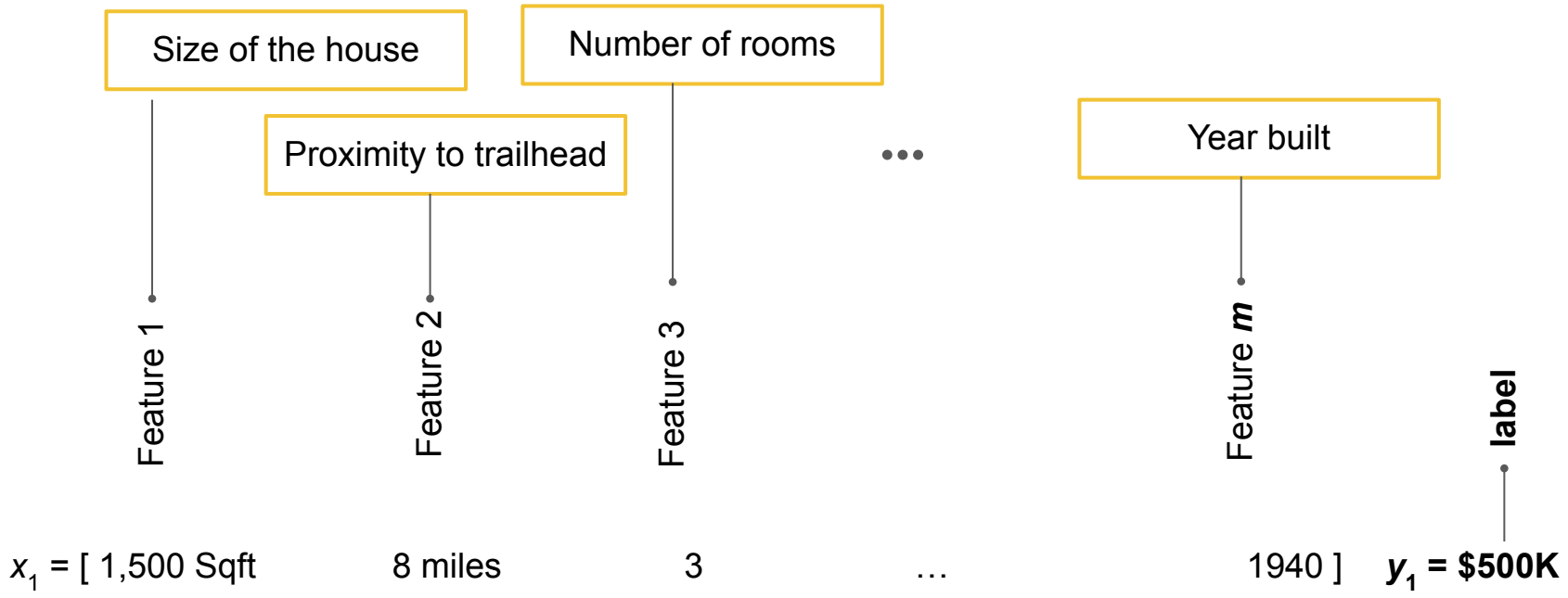
Housing price prediction in Eugene



x_1 = [1,500 Sqft 8 miles 3 … 1940]

Preparing input data

Housing price prediction in Eugene



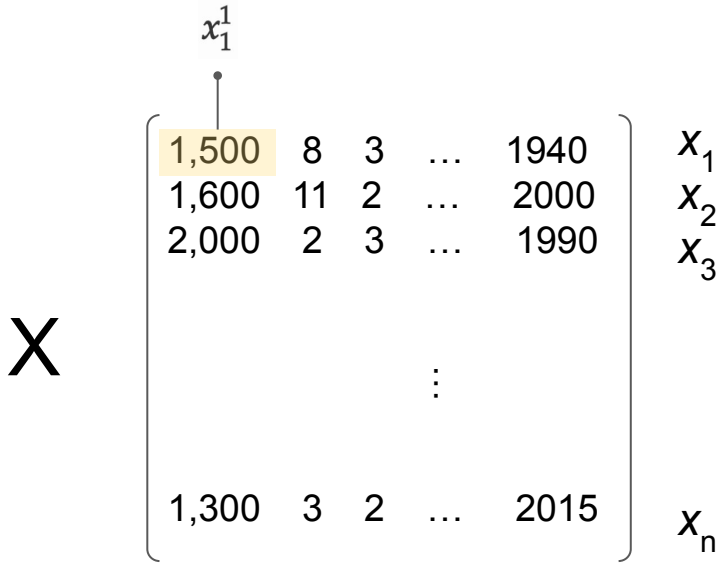
Preparing input data

Housing price prediction in Eugene

$$X = \begin{pmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & 11 & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & & \vdots & \\ 1{,}300 & 3 & 2 & \dots & 2015 \end{pmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ \\ \\ x_n \end{matrix}$$

Preparing input data

Housing price prediction in Eugene



Preparing input data

Housing price prediction in Eugene

$$X = \begin{bmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & 11 & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & \vdots & & \\ 1{,}300 & 3 & 2 & \dots & 2015 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ \\ \\ x_n \end{matrix} \qquad Y = \begin{bmatrix} 500{,}000 \\ 650{,}000 \\ 450{,}000 \\ \vdots \\ 650{,}000 \end{bmatrix}$$

Non-numeric data needs to be turned into numbers

Brand new

Fully renovated

Partially renovated

Not remodeled

Non-numeric data needs to be turned into numbers – numerical categories

Brand new	1
Fully renovated	2
Partially renovated	3
Not remodeled	4

Non-numeric data needs to be turned into numbers – one-hot encoding

Brand new	1	1000
Fully renovated	2	0100
Partially renovated	3	0010
Not remodeled	4	0001

Data normalization

# rooms:	1, 2, 3, 4,5
Size:	500 - 5,000

Data normalization

rooms: 1, 2, 3, 4,5

Size: 500 - 5,000

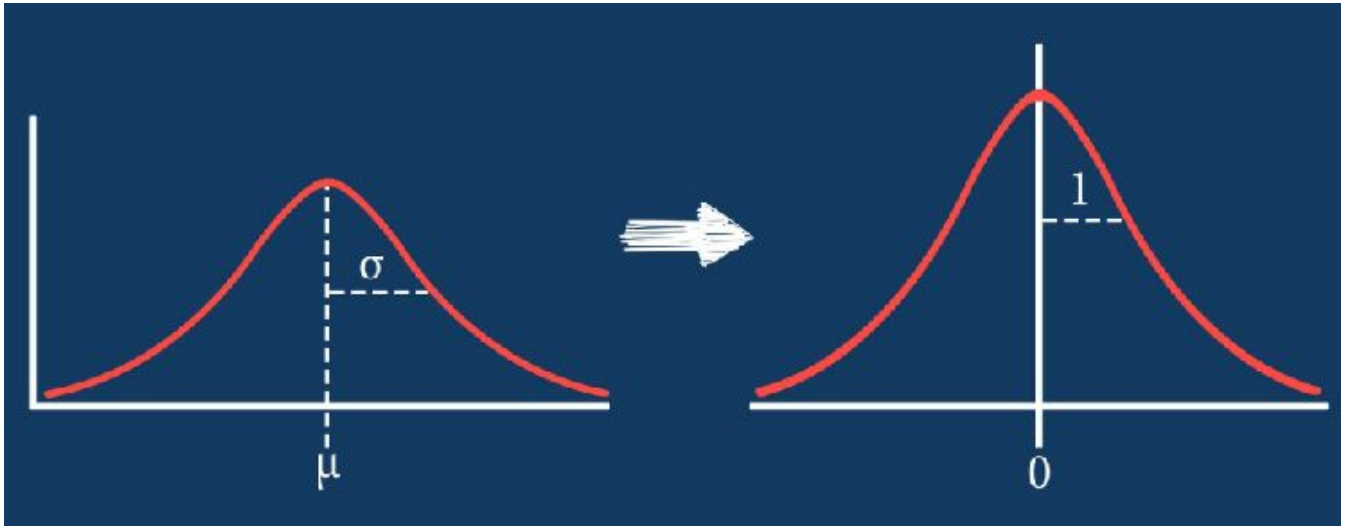
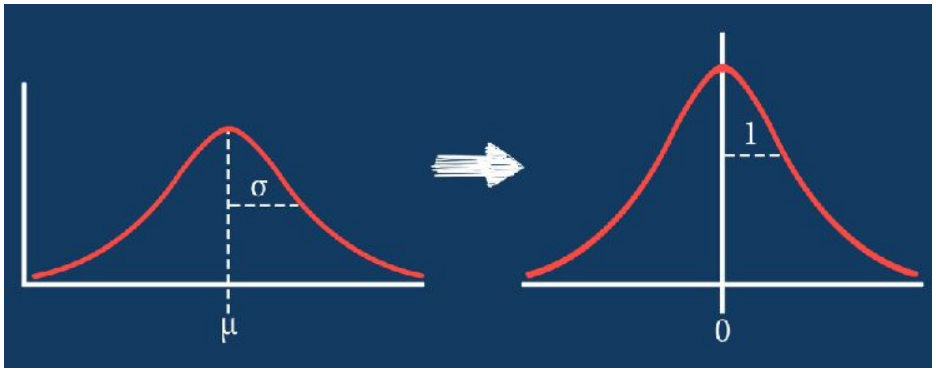


Image from towards ai: how, when, and why should you normalize/standardize/rescale your data

Data normalization

rooms: 1, 2, 3, 4,5

Size: 500 - 5,000



Min-max normalization
Decimal scaling

Image from towards ai: how, when, and why should you normalize/standardize/rescale your data

Missing data

$$X = \begin{pmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & \mathbf{NA} & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & & \vdots & \\ 1{,}300 & 3 & 2 & \dots & \mathbf{NA} \end{pmatrix}$$

Missing data

$$X = \begin{pmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & \mathbf{NA} & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & & \vdots & \\ 1{,}300 & 3 & 2 & \dots & \mathbf{NA} \end{pmatrix}$$

1. Removing inputs with missing features

Missing data

$$X = \begin{pmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & \mathbf{NA} & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & & \vdots & \\ 1{,}300 & 3 & 2 & \dots & \mathbf{NA} \end{pmatrix}$$

1. Removing inputs with missing features
2. Substitute with the mean

Missing data

$$X = \begin{bmatrix} 1{,}500 & 8 & 3 & \dots & 1940 \\ 1{,}600 & \mathbf{NA} & 2 & \dots & 2000 \\ 2{,}000 & 2 & 3 & \dots & 1990 \\ & & & \vdots & \\ 1{,}300 & 3 & 2 & \dots & \mathbf{NA} \end{bmatrix}$$

1. Removing inputs with missing features
2. Substitute with the mean
3. Substitute with the most frequent

Types of input data for proteins

1. Simple input

protein$_1$	25 kDa	pI=7.5	310 residues	…	2.5 hr half-life	Stability$_1$
protein$_2$	10 kDa	pI=4	50 residues	…	10 hr half-life	Stability$_2$
protein$_3$	100 kDa	pI=8	1200 residues	…	2 hr half-life	Stability$_3$
…						

Finding proper data is highly task dependent

Classification

Yes/No or multi-class?
Are there labeled data available?
How much data is out there?
Can you add labels?

Public datasets can be a great place

Classification

Yes/No or multi-class?
Are there labeled data available?
How much data is out there?
Can you add labels?

Places to look into:

- Datasets from other papers
- Competitions that curate a dataset
- Datasets that are not fully labeled, but can easily be modified

Where to find information on proteins?

1. PDB (https://www.rcsb.org/)
   a. X-ray structures
   b. Cryo-EM structures
   c. NMR structures

Where to find information on proteins?

1. PDB (https://www.rcsb.org/)

2. AlphaFold Protein structure database (https://alphafold.ebi.ac.uk/)
   a. Predicted structures
   b. Contains many disordered regions
   c. Not fully pruned for oligomers

Where to find information on proteins?

1. PDB (https://www.rcsb.org/)

2. AlphaFold Protein structure database (https://alphafold.ebi.ac.uk/)

3. UniRef (https://www.uniprot.org/help/uniref)
   a. 100 million protein sequences
   b. 0.5% are manually annotated
   c. Have clusters of % identity (UniRef50)

Where to find information on proteins?

1. PDB (https://www.rcsb.org/)
2. AlphaFold Protein structure database (https://alphafold.ebi.ac.uk/)
3. UniRef (https://www.uniprot.org/help/uniref)
4. Pfam (http://pfam.xfam.org/)
   a. Protein family domains
   b. 19,632 families and clans

Specialized databases

1. OAS (http://opig.stats.ox.ac.uk/webapps/oas/) for antibodies
2. ProteinNet (https://github.com/aqlaboratory/proteinnet) for protein structures
3. APD3 (https://aps.unmc.edu/) for antimicrobial peptides
4. Disprot (https://disprot.org/) for disordered proteins
5. …

Collecting data is a challenging task

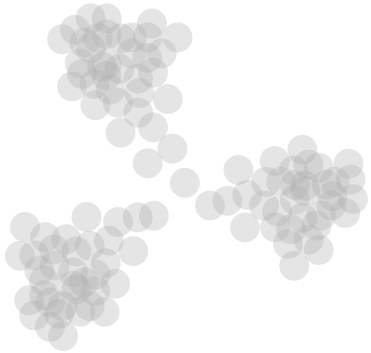
Classification

Yes/No or multi-class?
Are there labeled data available?
How much data is out there?
Can you add labels?

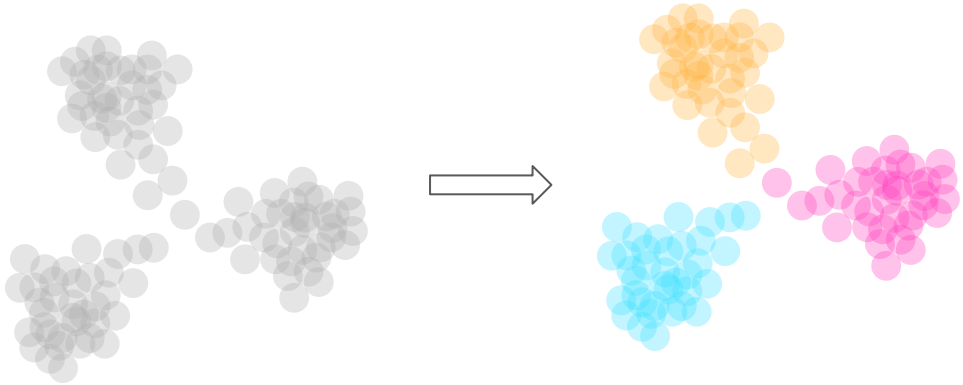
Think about:

- Time required for collection
- Storage
- Human error
- Automation

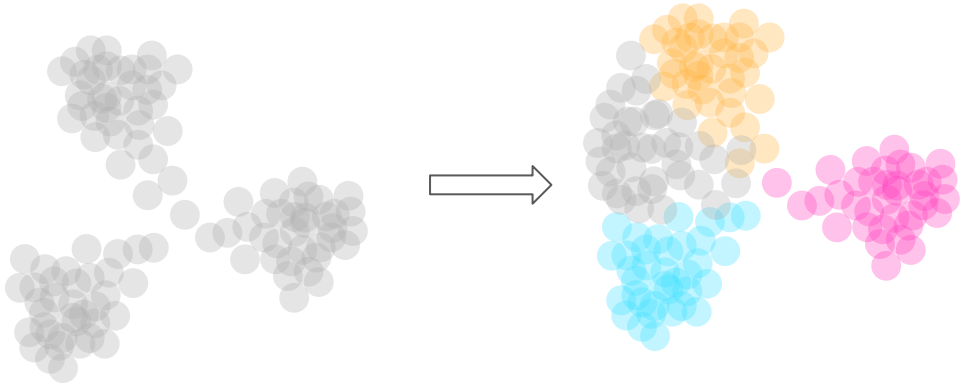
Truly representing the real distribution



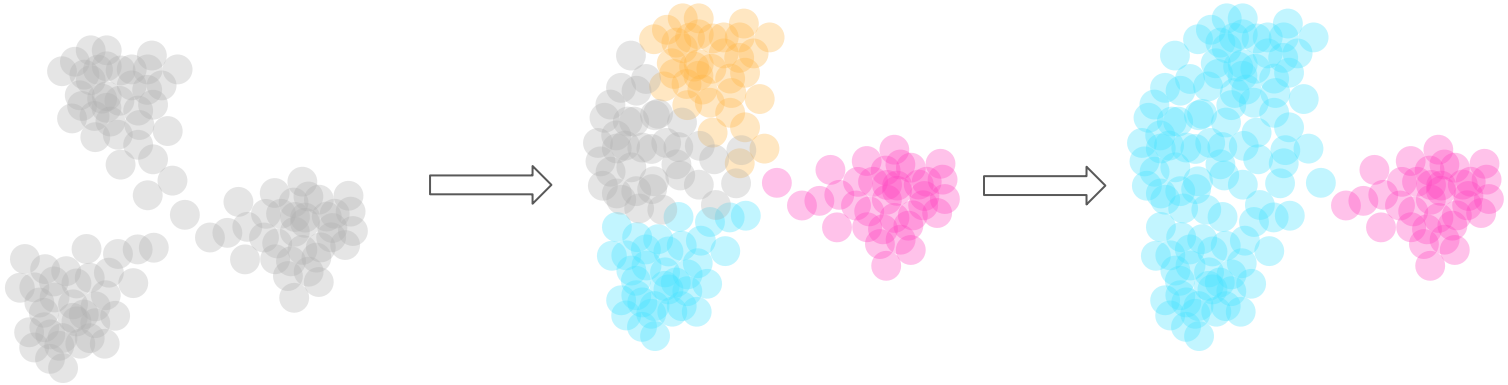
Truly representing the real distribution



Truly representing the real distribution



Truly representing the real distribution



Representation matters!

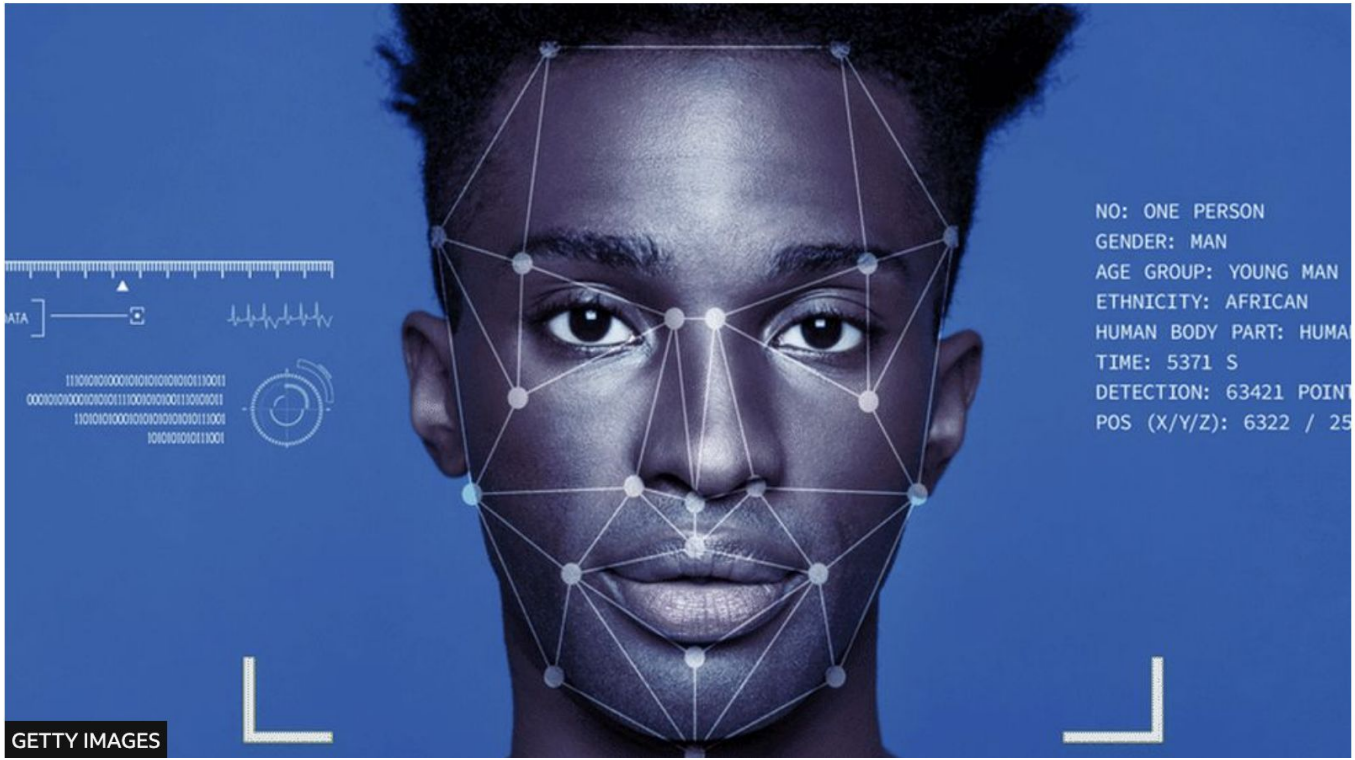


Image from BBC: Facial recognition fails on race

All data is flawed; some data is useful

1. **Human errors**: Check random subsets to make sure labeling can be trusted

All data is flawed; some data is useful

1. **Human errors**: Check random subsets to make sure labeling can be trusted
2. **Technical errors during file transfer:** Check the size of downloaded files

All data is flawed; some data is useful

1. **Human errors**: Check random subsets to make sure labeling can be trusted
2. **Technical errors during file transfer:** Check the size of downloaded files
3. **Missing values:** Replace as explained

All data is flawed; some data is useful

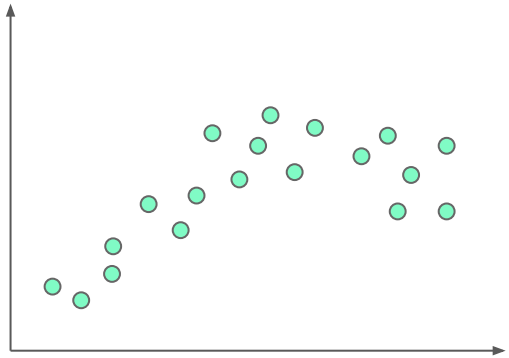
1. **Human errors**: Check random subsets to make sure labeling can be trusted
2. **Technical errors during file transfer:** Check the size of downloaded files
3. **Missing values:** Replace as explained
4. **Data matching the task:** Think about what you need and make sure you have it.

Train/test/validation split

Training set is used to train the algorithm

Training set

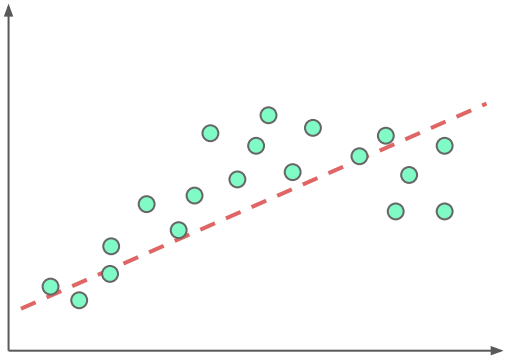
Data that is used by the learning algorithm to learn the hidden patterns in the data



Too simple models can’t learn complex patterns

Training set

Data that is used by the learning algorithm to learn the hidden patterns in the data

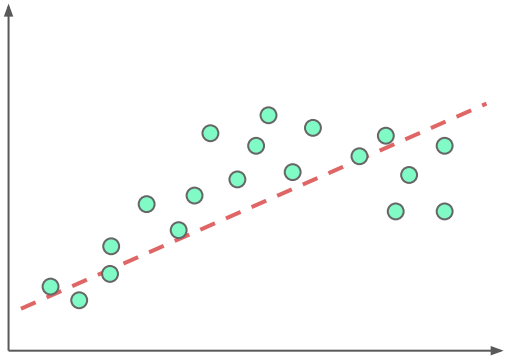


Model is underfitting

A more complex model is the solution to underfitting problems

Training set

Data that is used by the learning algorithm to learn the hidden patterns in the data



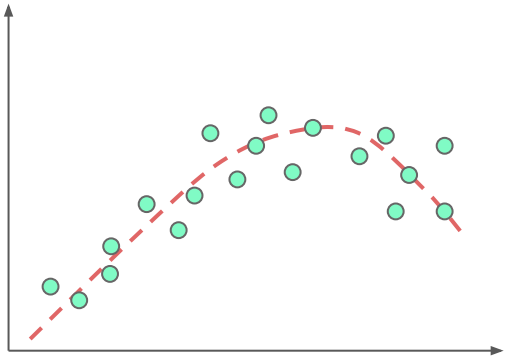
Model is underfitting

High loss value
Poor performance

A more complex model is the solution to underfitting problems

Training set

Data that is used by the learning algorithm to learn the hidden patterns in the data



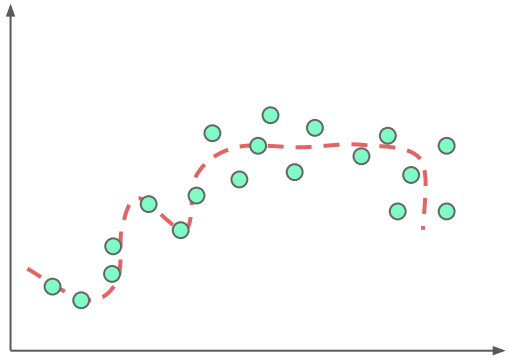
Model is a good fit

Predicted ~ Real
Good performance

Too complex models memorize the training set, resulting in overfitting

Training set

Data that is used by the learning algorithm to learn the hidden patterns in the data



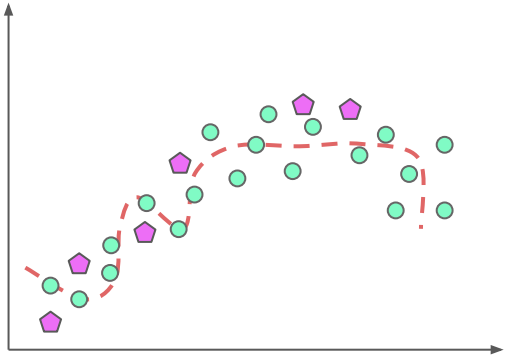
Model is a overfitting

An overfit model performs poorly on a test set

Training set

Test set

Separate set of data to be used to evaluate model AFTER training



Model is a overfitting

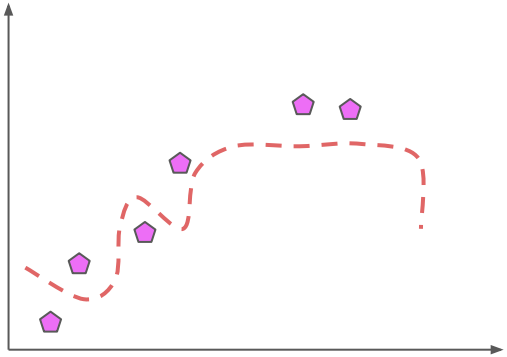
Performance on test set <<
Performance on training set

An overfit model performs poorly on a test set

Training set

Test set

Separate set of data to be used to evaluate model AFTER training



Model is a overfitting

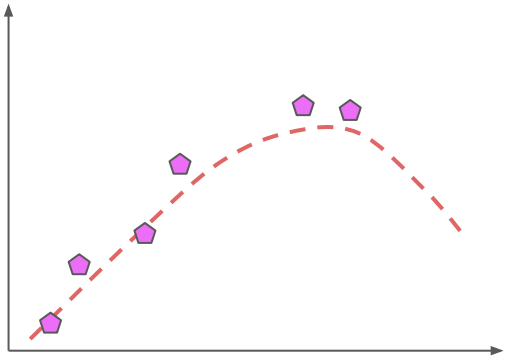
Performance on test set <<
Performance on training set

Simpler models or more data are solutions to overfitting

Training set

Test set

Separate set of data to be used to evaluate model AFTER training



Model is a overfitting

Performance on test set <<
Performance on training set

Validation set can help prevent overfitting

Training set

Test set

Validation set

Set of data separate from training set to evaluate the model during training

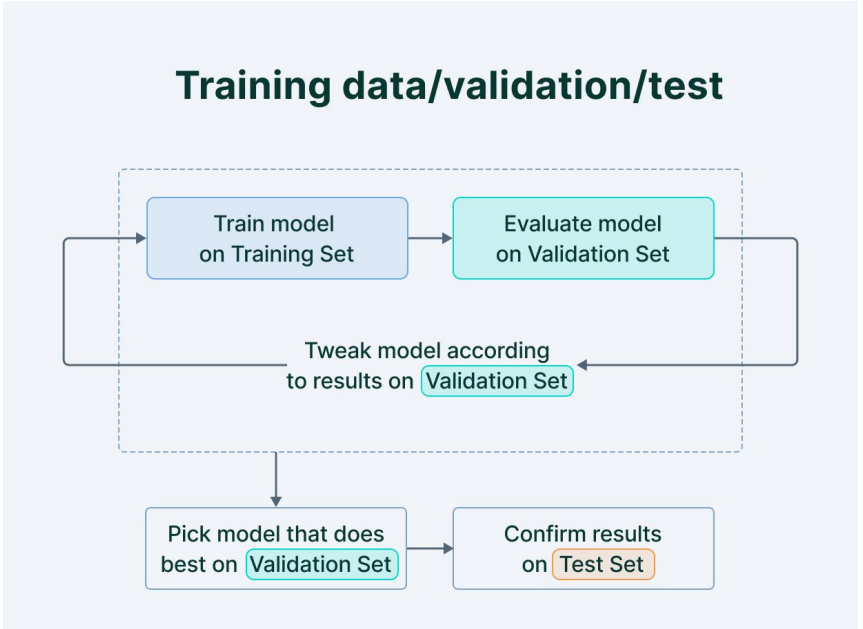


Image from V7 labs blog

Cross-validation can help when we don't have lots of data

Training set

Test set

Validation set

Set of data separate from training set to evaluate the model during training

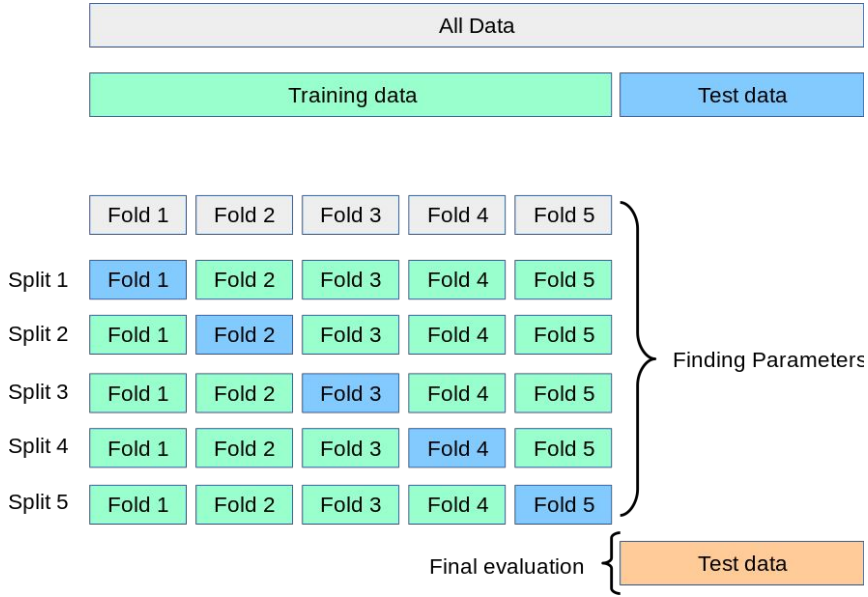


Image from scikit learn: cross-validation

What's a good split?

1. Make sure you have enough training data
2. Check to see if cross-validation is an option
3. A complex model with many features and parameters benefit from having more training+validation set
4. Too small a test set will result in high variations in performance

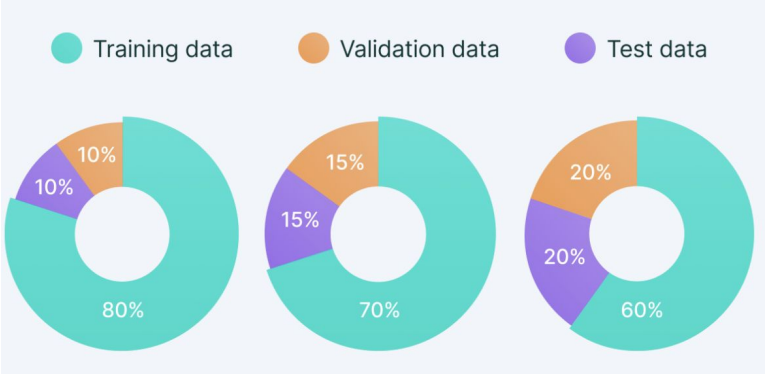
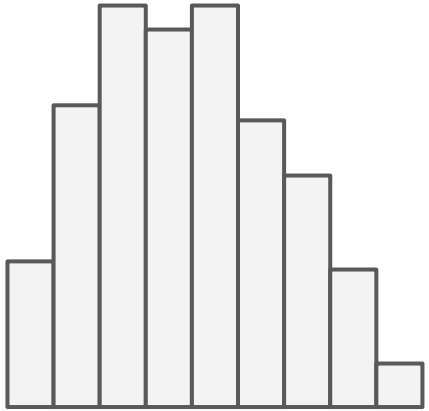
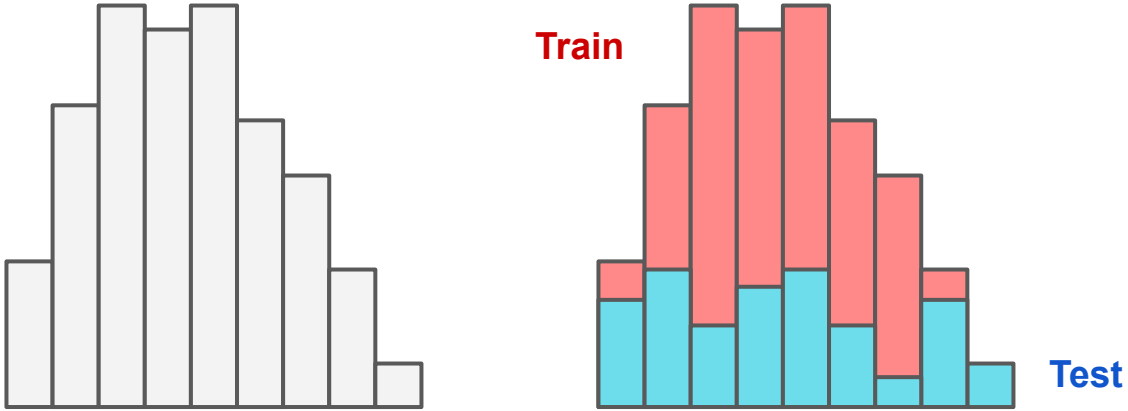


Image from V7 labs blog

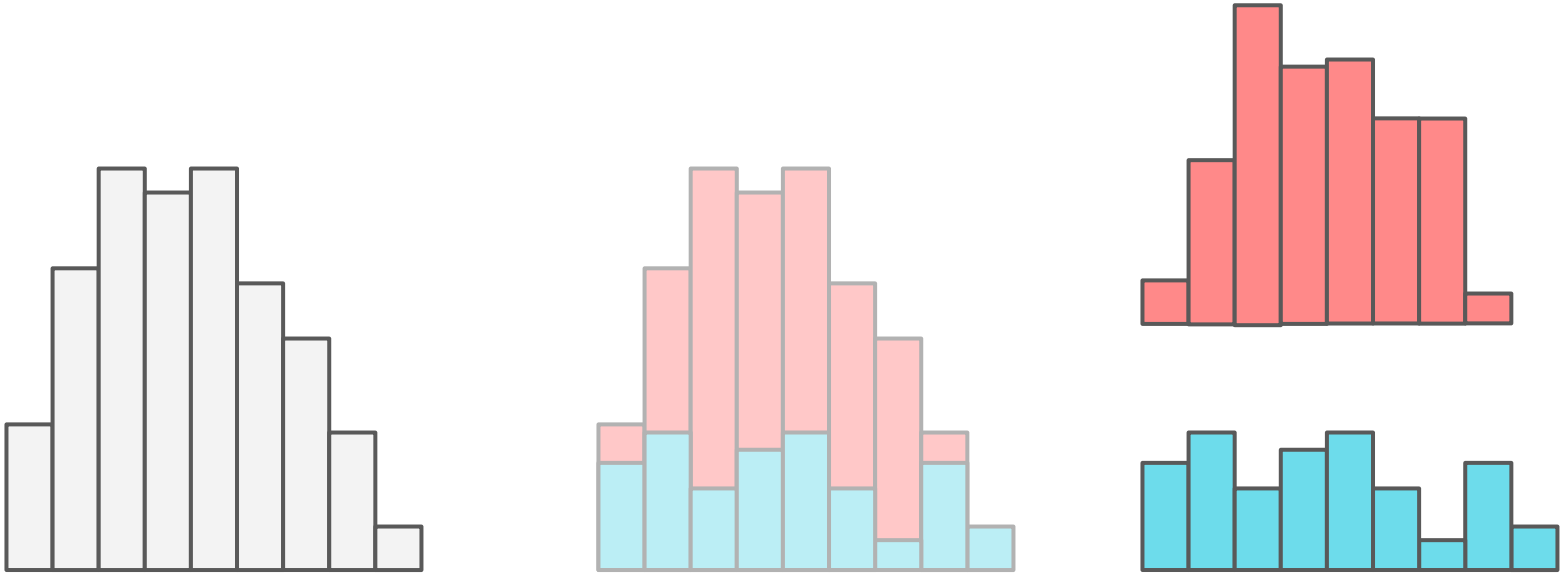
Keep the distribution of data in mind



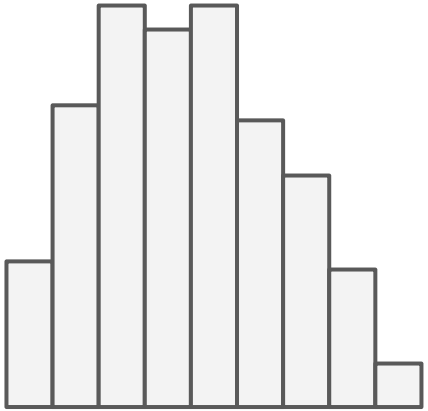
Keep the distribution of data in mind

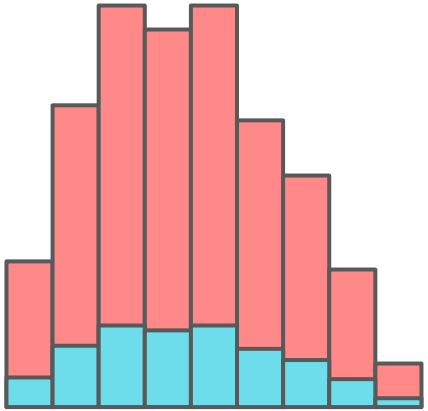


Keep the distribution of data in mind

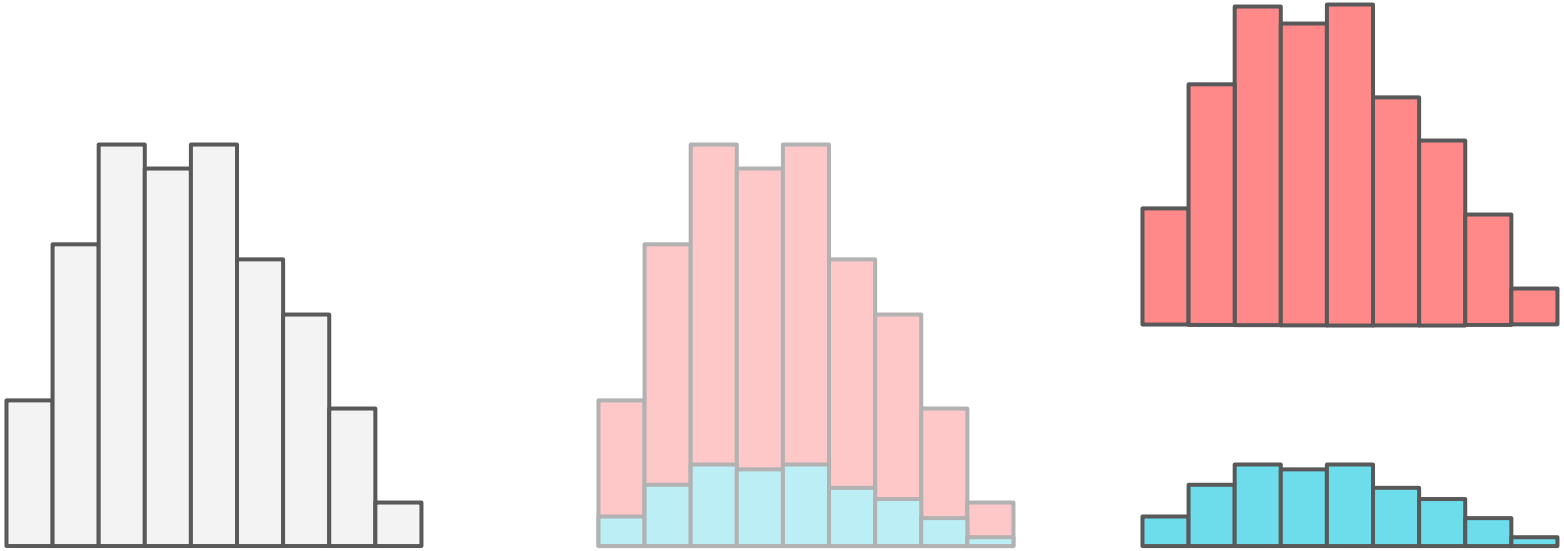


Keep the distribution of data in mind





Keep the distribution of data in mind



Next lecture:

Simple methods can take you a long way!

Mt Hood national park by Dale Nibbe, Unsplash

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids

p_1 MGLTDILGFNREFDILAV…SPLFG s_1

p_2 MLKPTRVNMSERCGHITDENVCSR…TLVRF s_2

p_3 MIKRTVIHGRDFRWNYTSPL…GMNSWQ s_3

…

Features: charge, pKa, size, functional groups, hydrogen bond status, …

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)

p_1 MGLTDILGFNREFDILAV…SPLFG s_1

p_2 MLKPTRVNMSERCGHITDENVCSR…TLVRF s_2

p_3 MIKRTVIHGRDFRWNYTSPL…GMNSWQ s_3

…

Features: charge, pKa, size, functional groups, hydrogen bond status, …

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image

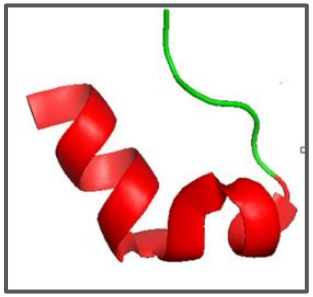


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image

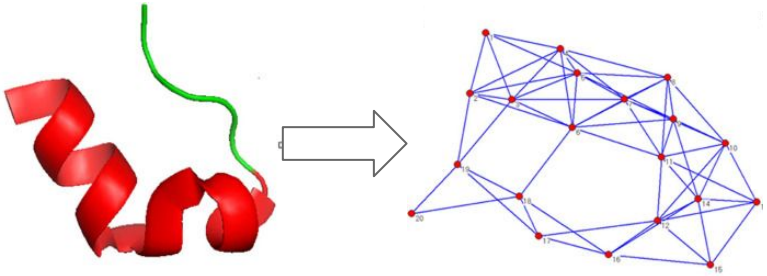


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image

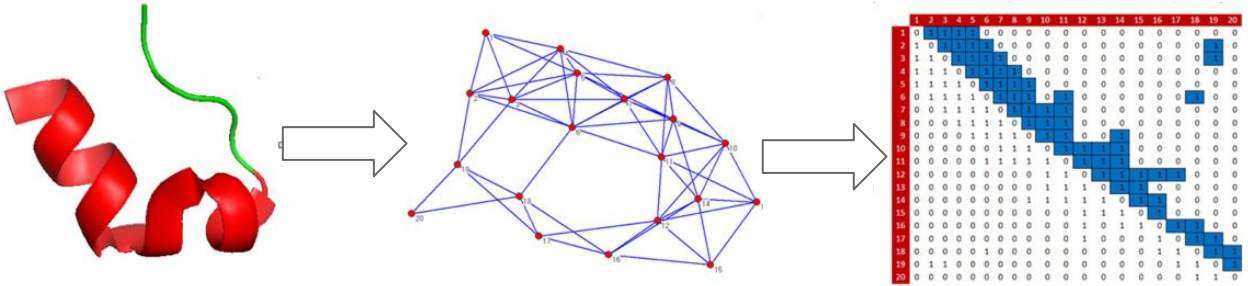


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image

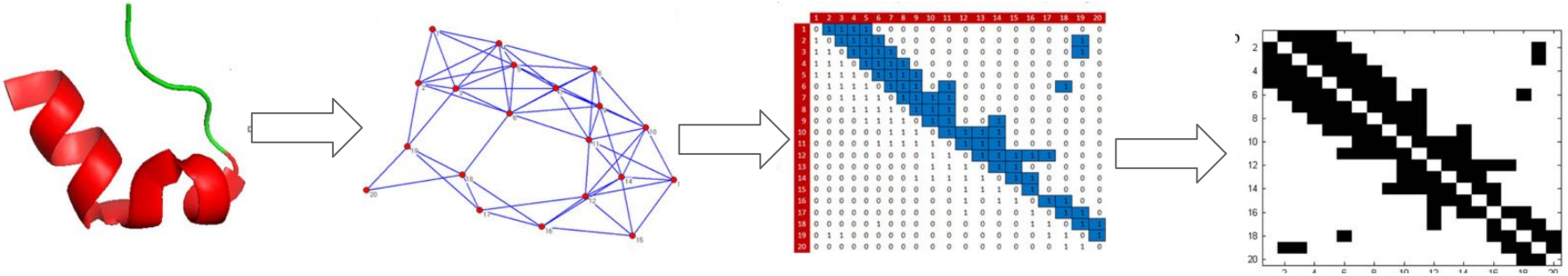


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image — Convolutional neural nets

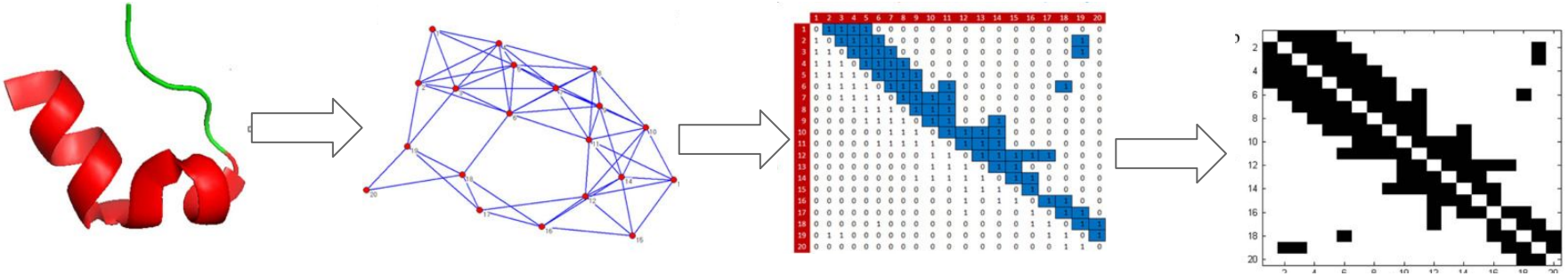


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image — Convolutional neural nets
4. Graphs

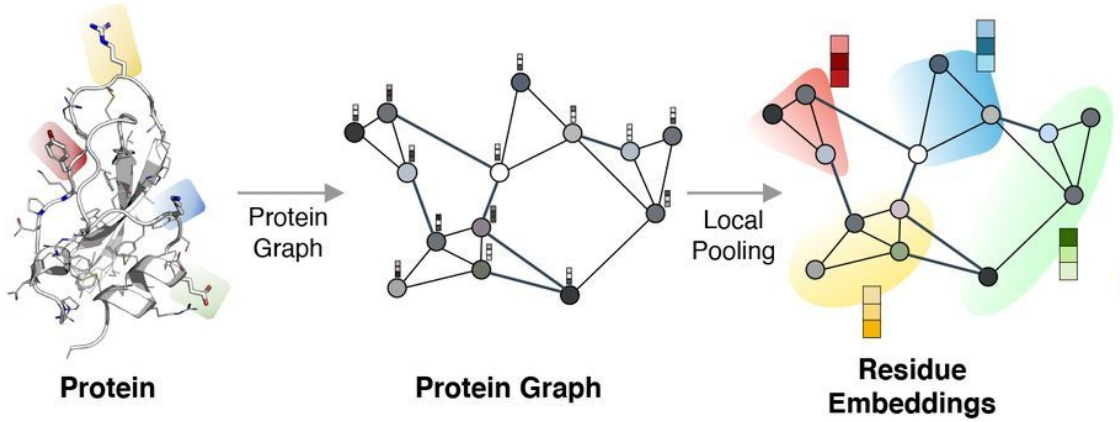


Image from bioRxiv 2020, ProteinGNC

Types of input data for proteins

1.	Simple input	SVM, Random Forest, dense neural net
2.	String of amino acids	Natural language processing (RNN, LSTM, Transformers)
3.	2D image	Convolutional neural nets
4.	Graphs	Graph convolutional neural nets

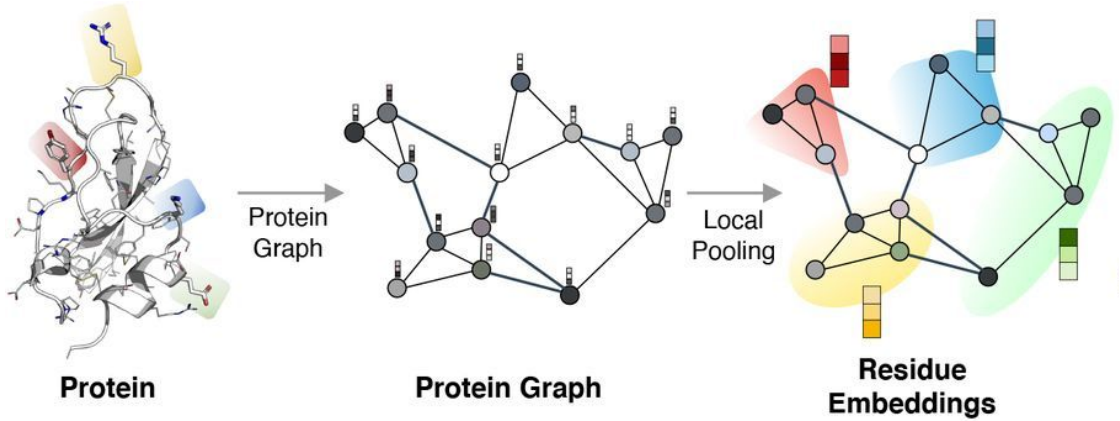
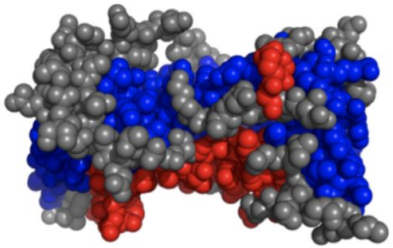


Image from bioRxiv 2020, ProteinGNC

Types of input data for proteins

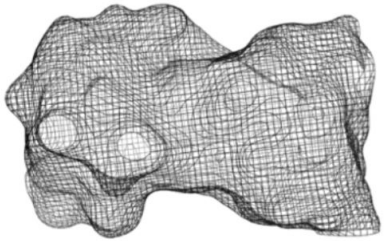
1. Simple input — SVM, Random Forest, dense neural net
2. String of amino acids — Natural language processing (RNN, LSTM, Transformers)
3. 2D image — Convolutional neural nets
4. Graphs — Graph convolutional neural nets
5. 3D objects

Set of balls / Point cloud



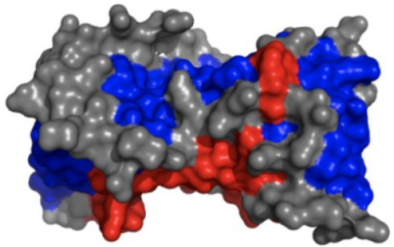
classical statistical potentials; Eismann et al. 2020

Gaussian clouds



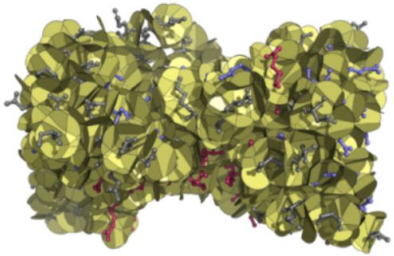
Derevyanko et al. Bioinformatics 2018; Pages et al. Bioinformatics 2019;

Molecular surface



Olechnovic & Venclovas, Proteins 2017; Correia, Bronstein et al. Nat Met 2020

3D tessellation



Igashov et al. Bioinformatics 2021; Olechnovic et al. Proteins 2021