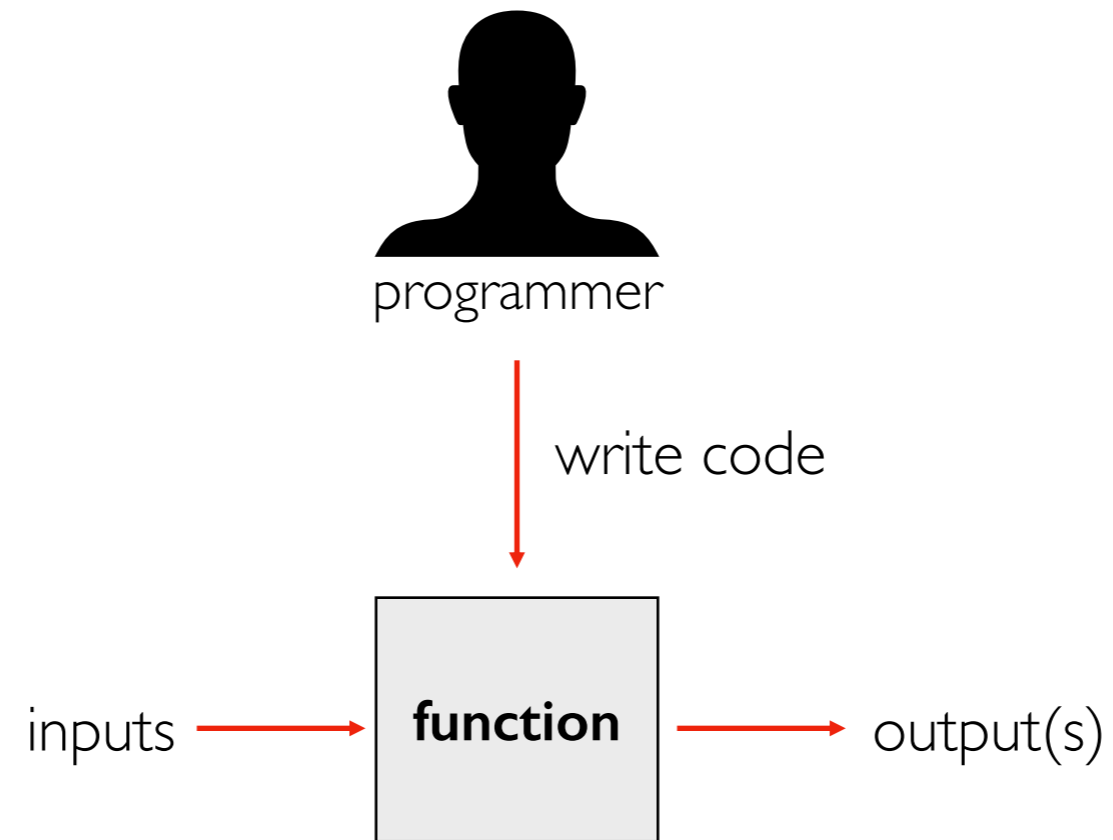


[320] Machine Learning: Intro

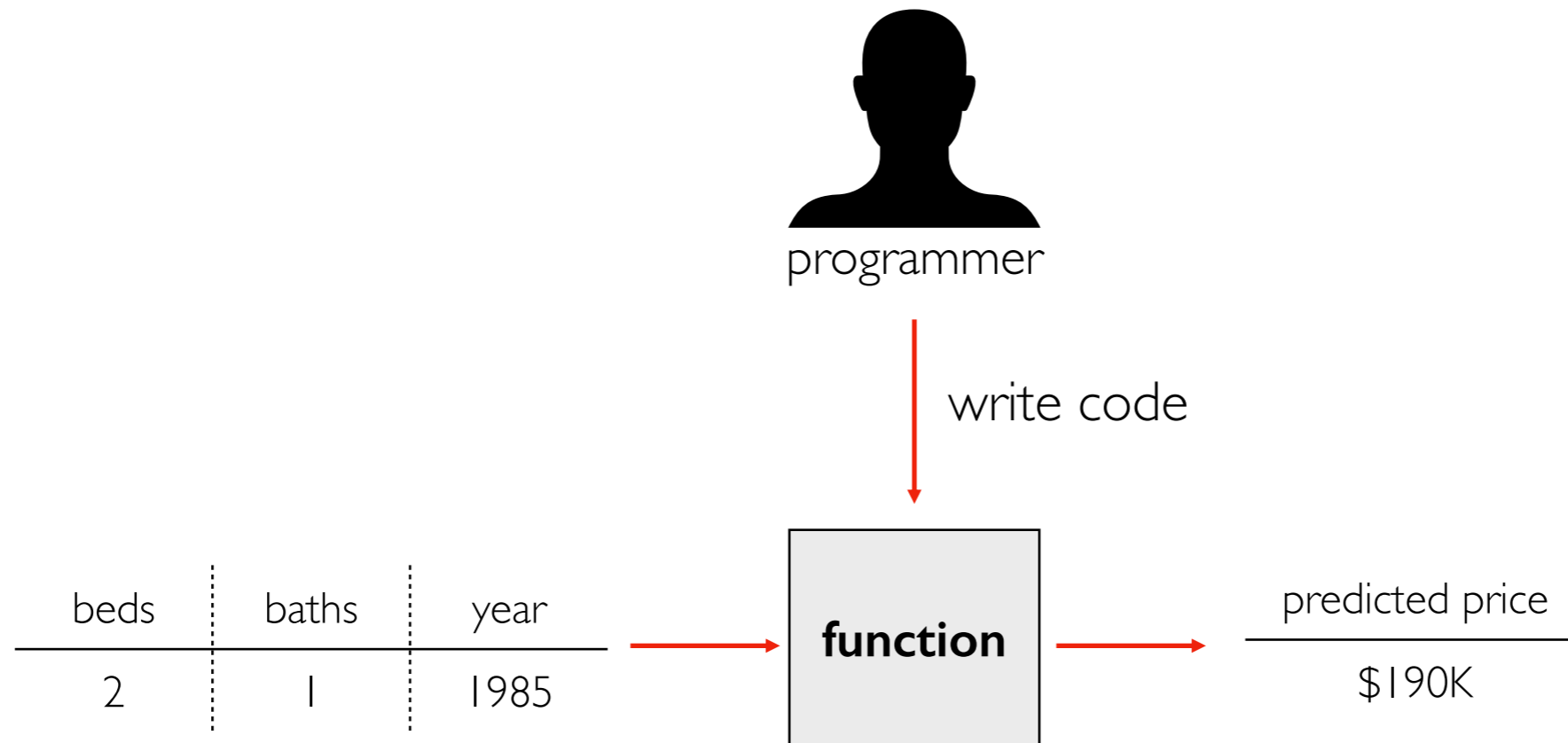
Yiyin Shen

Functions/Models

How do we make functions?

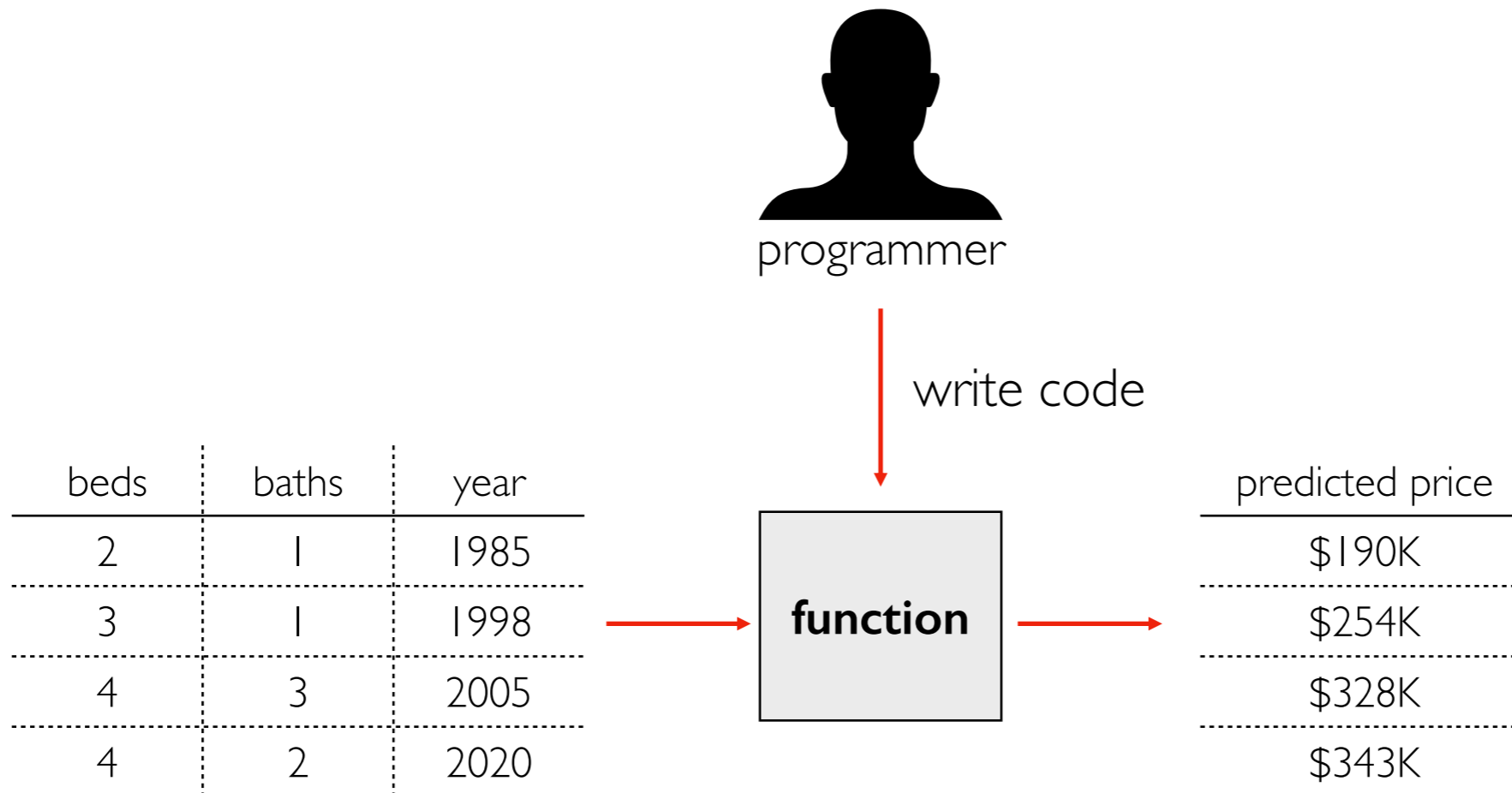


How do we make functions?



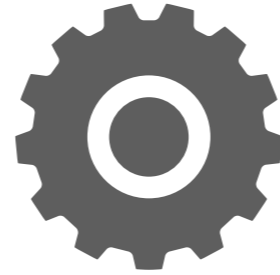
many functions are **models** that can be used to predict

How do we make functions?

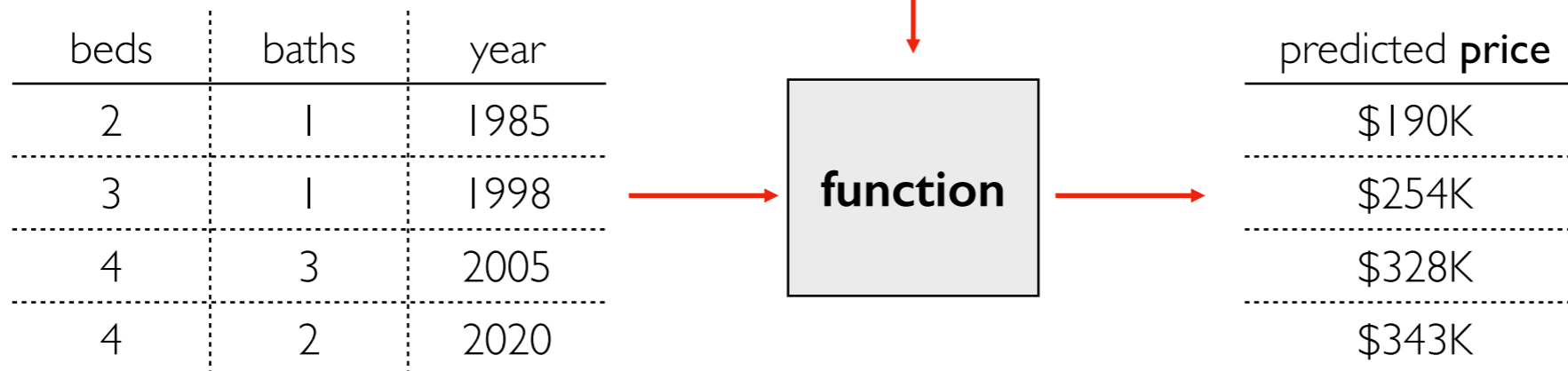


many functions are **models** that can be used to predict

How do we make functions?



Machine Learning Algorithm



many functions are **models** that can be used to predict

How do we make functions?

training data

| beds | baths | year | price |
|------|-------|------|--------|
| 1 | 1 | 1980 | \$140K |
| 3 | 1 | 1990 | \$240K |
| 3 | 4 | 2004 | \$295K |
| 4 | 3 | 2018 | \$350K |



Machine Learning Algorithm

live data

| beds | baths | year |
|------|-------|------|
| 2 | 1 | 1985 |
| 3 | 1 | 1998 |
| 4 | 3 | 2005 |
| 4 | 2 | 2020 |

train



predicted price

\$190K

\$254K

\$328K

\$343K

many functions are **models** that can be used to predict

How do we make functions?

training data

| beds | baths | year | price |
|------|-------|------|--------|
| 1 | 1 | 1980 | \$140K |
| 3 | 1 | 1990 | \$240K |
| 3 | 4 | 2004 | \$295K |
| 4 | 3 | 2018 | \$350K |



Machine Learning Algorithm

live data

| beds | baths | year |
|------|-------|------|
| 2 | 1 | 1985 |
| 3 | 1 | 1998 |
| 4 | 3 | 2005 |
| 4 | 2 | 2020 |

train

function

predicted price

\$190K

\$254K

\$328K

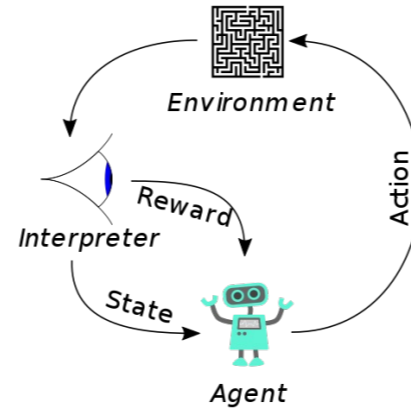
\$343K

this is an example of a **regression** model, which is a type of **supervised machine learning**, which is one of the 3 main categories of ML

Machine Learning

Reinforcement Learning

not covered in CS 320



https://en.wikipedia.org/wiki/Reinforcement_learning

Supervised Machine Learning

data is labeled, we know what we want to predict

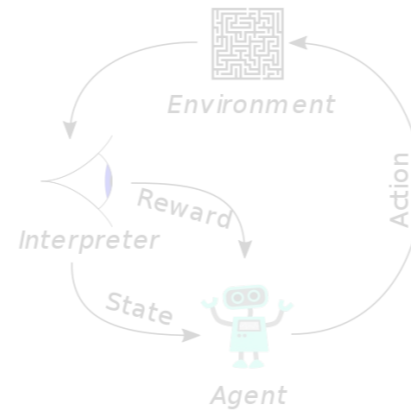
Unsupervised Machine Learning

data is unlabeled, we're just looking for patterns

Machine Learning

Reinforcement Learning

not covered in CS 320



https://en.wikipedia.org/wiki/Reinforcement_learning

Supervised Machine Learning

data is labeled, we know what we want to predict

Regression

predict a quantity

Classification

predict a category

Unsupervised Machine Learning

data is unlabeled, we're just looking for patterns

Clustering

place rows in groups

Decomposition

represent rows as combos of "component" rows

I. Regression (Supervised)

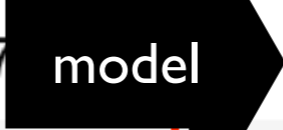
features

| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 |
| 1 | 50 | 13 | 7 | 67 | 79 | 25 |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 |
| 5 | 53 | 5 | 67 | 99 | 55 | ???? |
| 6 | 47 | 31 | 9 | 56 | 27 | ???? |
| 7 | 50 | 3 | 20 | 24 | 63 | ???? |
| 8 | 36 | 32 | 66 | 70 | 7 | ???? |
| 9 | 27 | 33 | 16 | 21 | 9 | ???? |

problem: can we predict an unknown **quantity** based on **features**?

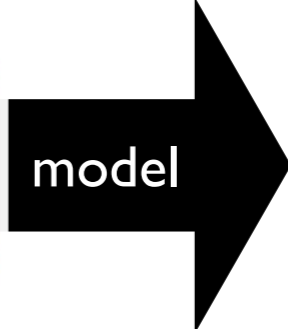
I. Regression (Supervised)

| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 |
| 1 | 50 | 13 | 7 | 67 | 67 | 25 |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 |
| 5 | 53 | 5 | 67 | 99 | 55 | ???? |
| 6 | 47 | 31 | 9 | 56 | 27 | ???? |
| 7 | 50 | 3 | 20 | 24 | 63 | ???? |
| 8 | 36 | 32 | 66 | 70 | 7 | ???? |
| 9 | 27 | 33 | 16 | 21 | 9 | ???? |



train: fit a model to the relationship between some label (y) and feature (x's) values

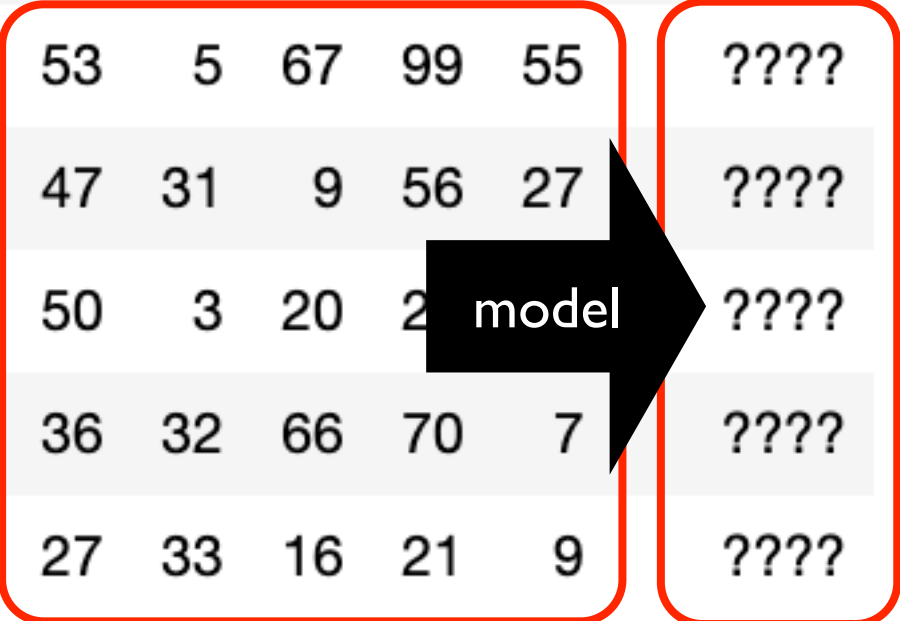
I. Regression (Supervised)

| | x0 | x1 | x2 | x3 | x4 | y (label) | |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|--|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 | |
| 1 | 50 | 13 | 7 | 67 | 79 | 25 | |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 | |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |  |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 | |
| 5 | 53 | 5 | 67 | 99 | 55 | ???? | 70 |
| 6 | 47 | 31 | 9 | 56 | 27 | ???? | 60 |
| 7 | 50 | 3 | 20 | 24 | 63 | ???? | |
| 8 | 36 | 32 | 66 | 70 | 7 | ???? | |
| 9 | 27 | 33 | 16 | 21 | 9 | ???? | |

test: make some predictions for known rows -- how close are we?

I. Regression (Supervised)

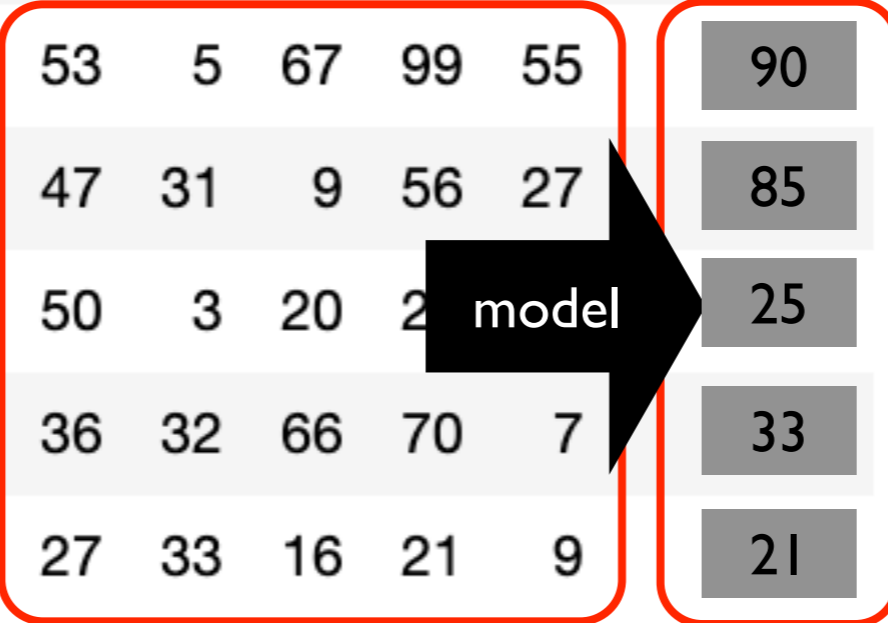
| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 |
| 1 | 50 | 13 | 7 | 67 | 79 | 25 |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 |
| 5 | 53 | 5 | 67 | 99 | 55 | ???? |
| 6 | 47 | 31 | 9 | 56 | 27 | ???? |
| 7 | 50 | 3 | 20 | 2 | 2 | ???? |
| 8 | 36 | 32 | 66 | 70 | 7 | ???? |
| 9 | 27 | 33 | 16 | 21 | 9 | ???? |



predict: estimate for actual unknowns

I. Regression (Supervised)

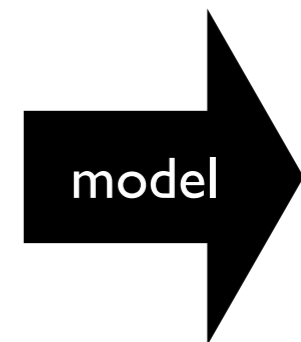
| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 |
| 1 | 50 | 13 | 7 | 67 | 79 | 25 |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 |
| 5 | 53 | 5 | 67 | 99 | 55 | 90 |
| 6 | 47 | 31 | 9 | 56 | 27 | 85 |
| 7 | 50 | 3 | 20 | 2 | 2 | 25 |
| 8 | 36 | 32 | 66 | 70 | 7 | 33 |
| 9 | 27 | 33 | 16 | 21 | 9 | 21 |



predict: estimate for actual unknowns

I. Regression (Supervised)

| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 5 |
| 1 | 50 | 13 | 7 | 67 | 79 | 25 |
| 2 | 56 | 12 | 5 | 15 | 90 | 44 |
| 3 | 89 | 70 | 85 | 49 | 68 | 72 |
| 4 | 36 | 93 | 52 | 33 | 14 | 59 |
| 5 | 53 | 5 | 67 | 99 | 55 | 90 |
| 6 | 47 | 31 | 9 | 56 | 27 | 85 |
| 7 | 50 | 3 | 20 | 24 | 63 | 25 |
| 8 | 36 | 32 | 66 | 70 | 7 | 33 |
| 9 | 27 | 33 | 16 | 21 | 9 | 21 |



interpret: what can we learn by looking directly at the model?

I. Regression (Supervised)

category features quantitative label

| | x0 | x1 | x2 | x3 | x4 | y (label) |
|---|----|-------|----|----------|----|-----------|
| 0 | 37 | green | 40 | triangle | 68 | 5 |
| 1 | 50 | green | 7 | circle | 79 | 25 |
| 2 | 56 | red | 5 | circle | 90 | 44 |
| 3 | 89 | blue | 85 | triangle | 68 | 72 |
| 4 | 36 | blue | 52 | square | 14 | 59 |
| 5 | 53 | green | 67 | triangle | 55 | ???? |
| 6 | 47 | blue | 9 | triangle | 27 | ???? |
| 7 | 50 | blue | 20 | circle | 63 | ???? |
| 8 | 36 | green | 66 | circle | 7 | ???? |
| 9 | 27 | red | 16 | circle | 9 | ???? |

a problem with some **category** features is still a regression as long as the lable is **quantitative**

2. Classification (Supervised)

category
label



| | x0 | x1 | x2 | x3 | x4 | y (label) |
|----------|-----------|-----------|-----------|-----------|-----------|------------------|
| 0 | 37 | green | 40 | triangle | 68 | orange |
| 1 | 50 | green | 7 | circle | 79 | pear |
| 2 | 56 | red | 5 | circle | 90 | pear |
| 3 | 89 | blue | 85 | triangle | 68 | apple |
| 4 | 36 | blue | 52 | square | 14 | pear |
| 5 | 53 | green | 67 | triangle | 55 | ???? |
| 6 | 47 | blue | 9 | triangle | 27 | ???? |
| 7 | 50 | blue | 20 | circle | 63 | ???? |
| 8 | 36 | green | 66 | circle | 7 | ???? |
| 9 | 27 | red | 16 | circle | 9 | ???? |

problem: can we predict an unknown **category**?

3. Clustering (Unsupervised)

no
label!



| | x0 | x1 | x2 | x3 | x4 |
|----------|-----------|-----------|-----------|-----------|-----------|
| 0 | 37 | 25 | 40 | 70 | 68 |
| 1 | 50 | 13 | 7 | 67 | 79 |
| 2 | 56 | 12 | 5 | 15 | 90 |
| 3 | 89 | 70 | 85 | 49 | 68 |
| 4 | 36 | 93 | 52 | 33 | 14 |
| 5 | 53 | 5 | 67 | 99 | 55 |
| 6 | 47 | 31 | 9 | 56 | 27 |
| 7 | 50 | 3 | 20 | 24 | 63 |
| 8 | 36 | 32 | 66 | 70 | 7 |
| 9 | 27 | 33 | 16 | 21 | 9 |

problem: can we organize data into groups of similar rows?

3. Clustering (Unsupervised)

the algorithm
decides groups

| | x0 | x1 | x2 | x3 | x4 | group |
|----------|-----------|-----------|-----------|-----------|-----------|--------------|
| 0 | 37 | 25 | 40 | 70 | 68 | 1 |
| 1 | 50 | 13 | 7 | 67 | 79 | 0 |
| 2 | 56 | 12 | 5 | 15 | 90 | 0 |
| 3 | 89 | 70 | 85 | 49 | 68 | 1 |
| 4 | 36 | 93 | 52 | 33 | 14 | 2 |
| 5 | 53 | 5 | 67 | 99 | 55 | 0 |
| 6 | 47 | 31 | 9 | 56 | 27 | 1 |
| 7 | 50 | 3 | 20 | 24 | 63 | 1 |
| 8 | 36 | 32 | 66 | 70 | 7 | 2 |
| 9 | 27 | 33 | 16 | 21 | 9 | 0 |

there is no official grouping to check the model against,
but a good grouping places similar rows together

4. Decomposition (Unsupervised)

| | x0 | x1 | x2 | x3 | x4 |
|----------|-----------|-----------|-----------|-----------|-----------|
| 0 | -11 | -7 | 3 | 20 | 20 |
| 1 | 2 | -19 | -30 | 17 | 31 |
| 2 | 8 | -20 | -32 | -35 | 42 |
| 3 | 41 | 38 | 48 | -1 | 20 |
| 4 | -12 | 61 | 15 | -17 | -34 |
| 5 | 5 | -27 | 30 | 49 | 7 |
| 6 | -1 | -1 | -28 | 6 | -21 |
| 7 | 2 | -29 | -17 | -26 | 15 |
| 8 | -12 | 0 | 29 | 20 | -41 |
| 9 | -21 | 1 | -21 | -29 | -39 |

4. Decomposition (Unsupervised)

original data

| | x0 | x1 | x2 | x3 | x4 |
|----------|-----------|-----------|-----------|-----------|-----------|
| 0 | -11 | -7 | 3 | 20 | 20 |
| 1 | 2 | -19 | -30 | 17 | 31 |
| 2 | 8 | -20 | -32 | -35 | 42 |
| 3 | 41 | 38 | 48 | -1 | 20 |
| 4 | -12 | 61 | 15 | -17 | -34 |
| 5 | 5 | -27 | 30 | 49 | 7 |
| 6 | -1 | -1 | -28 | 6 | -21 |
| 7 | 2 | -29 | -17 | -26 | 15 |
| 8 | -12 | 0 | 29 | 20 | -41 |
| 9 | -21 | 1 | -21 | -29 | -39 |

components

| | x0 | x1 | x2 | x3 | x4 |
|----------|-----------|-----------|-----------|-----------|-----------|
| 0 | -0.0 | 0.6 | 0.5 | 0.1 | -0.6 |
| 1 | 0.3 | -0.2 | 0.5 | 0.6 | 0.5 |
| 2 | 0.4 | 0.5 | 0.1 | -0.6 | 0.5 |

-11

21

-8

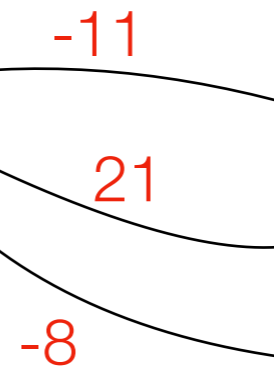
4. Decomposition (Unsupervised)

original data

| | x0 | x1 | x2 | x3 | x4 |
|---|-----|-----|-----|-----|-----|
| 0 | -11 | -7 | 3 | 20 | 20 |
| 1 | 2 | -19 | -30 | 17 | 31 |
| 2 | 8 | -20 | -32 | -35 | 42 |
| 3 | 41 | 38 | 48 | -1 | 20 |
| 4 | -12 | 61 | 15 | -17 | -34 |
| 5 | 5 | -27 | 30 | 49 | 7 |
| 6 | -1 | -1 | -28 | 6 | -21 |
| 7 | 2 | -29 | -17 | -26 | 15 |
| 8 | -12 | 0 | 29 | 20 | -41 |
| 9 | -21 | 1 | -21 | -29 | -39 |

components

| | x0 | x1 | x2 | x3 | x4 |
|---|------|------|-----|------|------|
| 0 | -0.0 | 0.6 | 0.5 | 0.1 | -0.6 |
| 1 | 0.3 | -0.2 | 0.5 | 0.6 | 0.5 |
| 2 | 0.4 | 0.5 | 0.1 | -0.6 | 0.5 |



weights

| | pc0 | pc1 | pc2 |
|---|-----|-----|-----|
| 0 | -11 | 21 | -8 |
| 1 | -43 | 12 | -6 |
| 2 | -58 | -14 | 30 |
| 3 | 36 | 41 | 53 |
| 4 | ... | ... | ... |

...

4. Decomposition (Unsupervised)

original data

| | x0 | x1 | x2 | x3 | x4 |
|---|-----|-----|-----|-----|-----|
| 0 | -11 | -7 | 3 | 20 | 20 |
| 1 | 2 | -19 | -30 | 17 | 31 |
| 2 | 8 | -20 | -32 | -35 | 42 |
| 3 | 41 | 38 | 48 | -1 | 20 |
| 4 | -12 | 61 | 15 | -17 | -34 |
| 5 | 5 | -27 | 30 | 49 | 7 |
| 6 | -1 | -1 | -28 | 6 | -21 |
| 7 | 2 | -29 | -17 | -26 | 15 |
| 8 | -12 | 0 | 29 | 20 | -41 |
| 9 | -21 | 1 | -21 | -29 | -39 |

components

| | x0 | x1 | x2 | x3 | x4 |
|---|------|------|-----|------|------|
| 0 | -0.0 | 0.6 | 0.5 | 0.1 | -0.6 |
| 1 | 0.3 | -0.2 | 0.5 | 0.6 | 0.5 |
| 2 | 0.4 | 0.5 | 0.1 | -0.6 | 0.5 |

-43

12

-6

weights

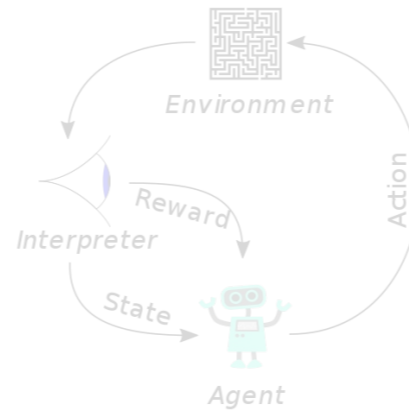
| | pc0 | pc1 | pc2 |
|---|-----|-----|-----|
| 0 | -11 | 21 | -8 |
| 1 | -43 | 12 | -6 |
| 2 | -58 | -14 | 30 |
| 3 | 36 | 41 | 53 |
| 4 | ... | ... | ... |

...

Machine Learning

Reinforcement Learning

not covered in CS 320



https://en.wikipedia.org/wiki/Reinforcement_learning

Supervised Machine Learning

data is labeled, we know what we want to predict

Regression

predict a quantity

Classification

predict a category

Unsupervised Machine Learning

data is unlabeled, we're just looking for patterns

Clustering

place rows in groups

Decomposition

represent rows as combos of "component" rows

this semester, we'll learn at least one technique in each of these four categories

1. Regression (Supervised)

+

2. Classification (Supervised)

```
linear_model.LogisticRegression([penalty, ...]) classification!  
linear_model.LogisticRegressionCV(*[, Cs, ...])  
linear_model.PassiveAggressiveClassifier(*  
linear_model.Perceptron(*[, penalty, alpha, ...])  
linear_model.RidgeClassifier([alpha, ...])  
linear_model.RidgeClassifierCV([alphas, ...])  
linear_model.SGDClassifier([loss, penalty, ...])
```

```
linear_model.LinearRegression(*[, ...])  
linear_model.Ridge([alpha, fit_intercept, ...])  
linear_model.RidgeCV([alphas, ...])  
linear_model.SGDRegressor([loss, penalty, ...])
```

```
svm.LinearSVC([penalty, loss, dual, tol, C, ...]) |  
svm.LinearSVR(*[, epsilon, tol, C, loss, ...]) |
```

```
tree.DecisionTreeClassifier  
tree.DecisionTreeRegressor  
tree.ExtraTreeClassifier  
tree.ExtraTreeRegressor
```

```
neighbors.KNeighborsClassifier([...])  
neighbors.KNeighborsRegressor([n_neighbors, ...])
```

3. Clustering (Unsupervised)

```
cluster.AffinityPropagation(*[, damping, ...])  
cluster.AgglomerativeClustering([...])  
cluster.Birch(*[, threshold, ...])  
cluster.DBSCAN([eps, min_samples, metric, ...])  
cluster.FeatureAgglomeration([n_clusters, ...])  
cluster.KMeans([n_clusters, init, n_init, ...])  
cluster.MinibatchKMeans([n_clusters, init, ...])  
cluster.MeanShift(*[, bandwidth, seeds, ...])  
cluster.OPTICS(*[, min_samples, max_eps, ...])  
cluster.SpectralClustering([n_clusters, ...])  
cluster.SpectralBiclustering([n_clusters, ...])  
cluster.SpectralCoclustering([n_clusters, ...])
```

4. Decomposition (Unsupervised)

```
decomposition.DictionaryLearning([...])  
decomposition.FactorAnalysis([n_components, ...])  
decomposition.FastICA([n_components, ...])  
decomposition.IncrementalPCA([n_components, ...])  
decomposition.KernelPCA([n_components, ...])  
decomposition.LatentDirichletAllocation([...])  
decomposition.MinibatchDictionaryLearning([...])  
decomposition.MinibatchSparsePCA([...])  
decomposition.NMF([n_components, init, ...])  
decomposition.PCA([n_components, copy, ...])  
decomposition.SparsePCA([n_components, ...])  
decomposition.SparseCoder(dictionary, *[, ...])  
decomposition.TruncatedSVD([n_components, ...])
```

Foundations: Modules and Math

Important Packages

We'll be learning the following to do ML and related calculations efficiently:

1

numpy

2

pytorch

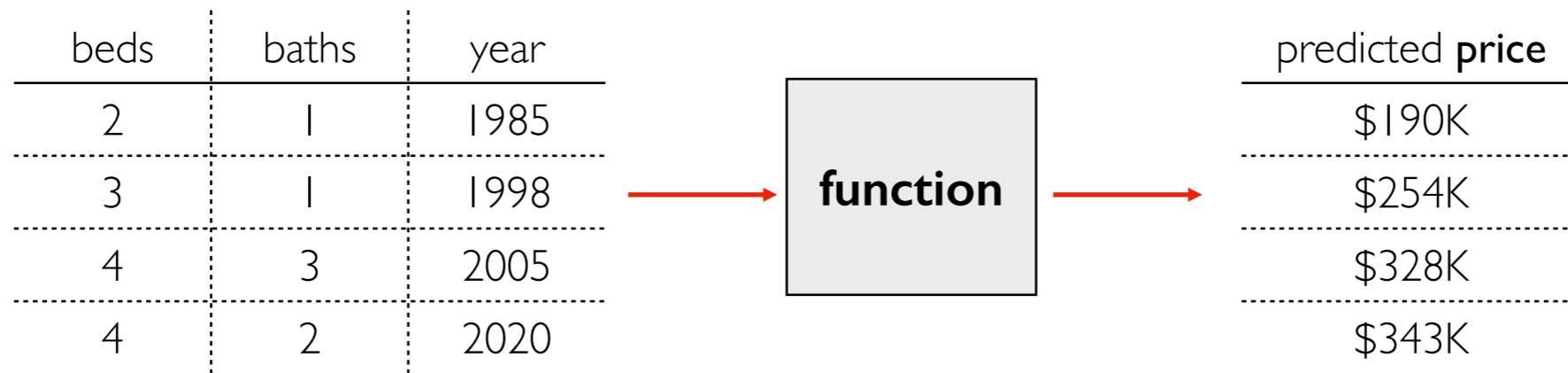
3

scikit-learn

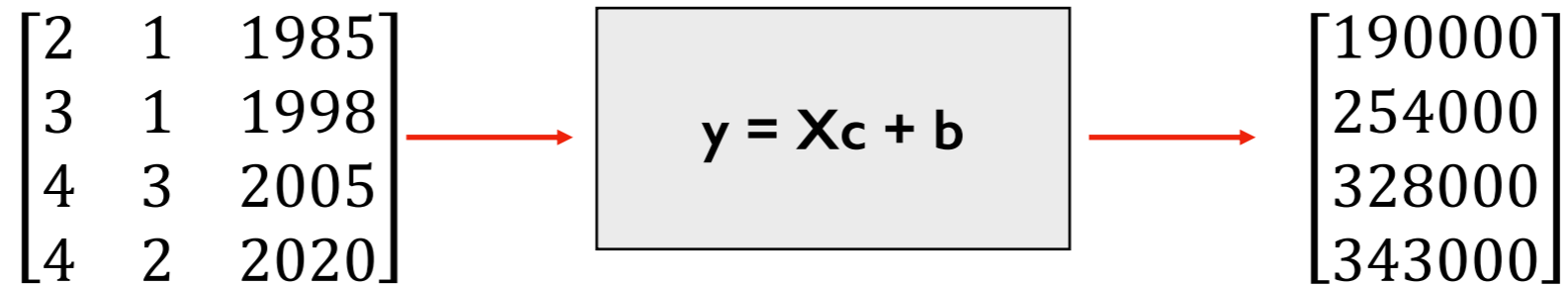
```
pip3 install numpy scikit-learn
```

```
pip3 install torch torchvision
```

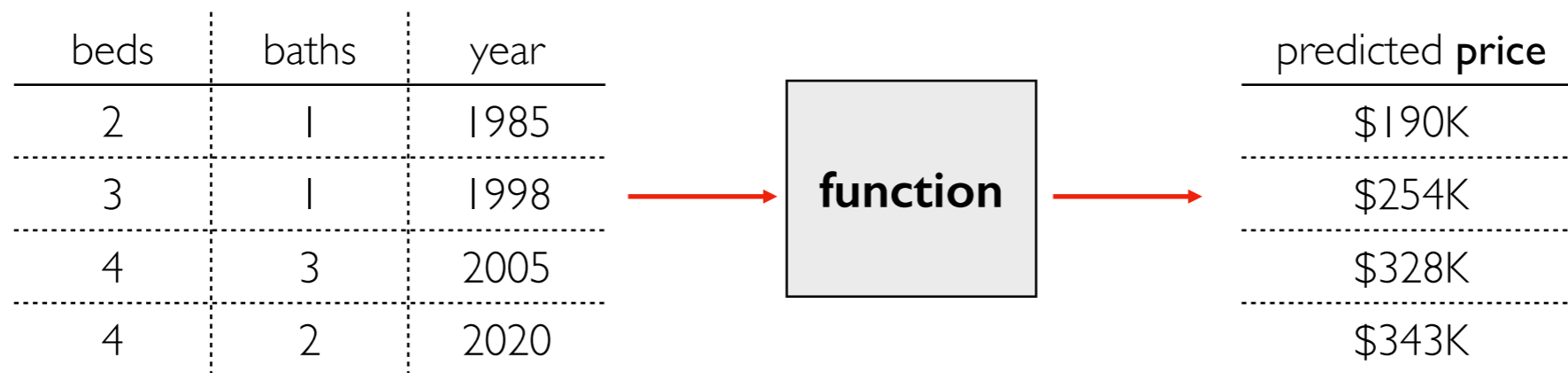
Linear Algebra



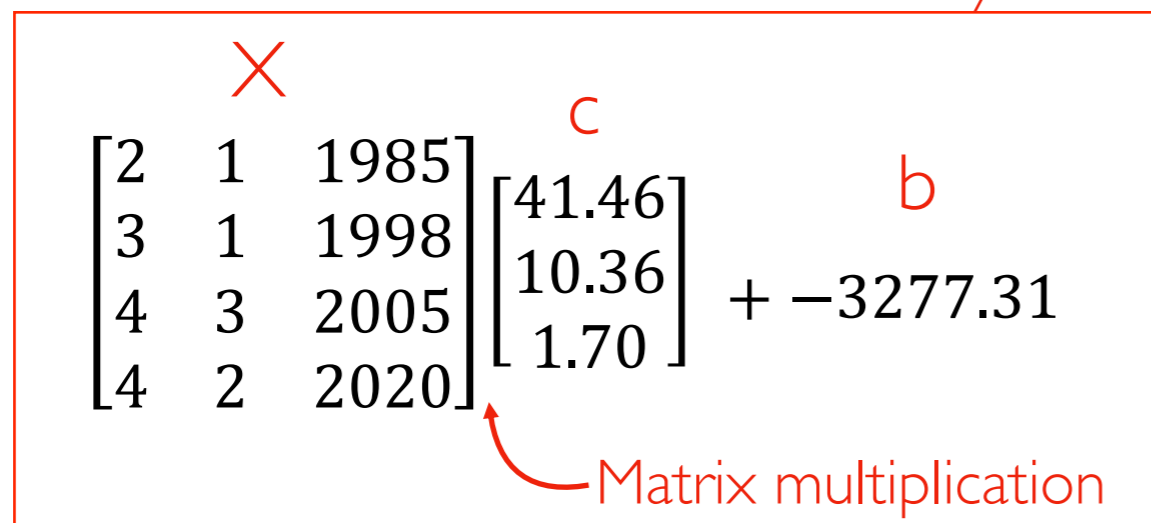
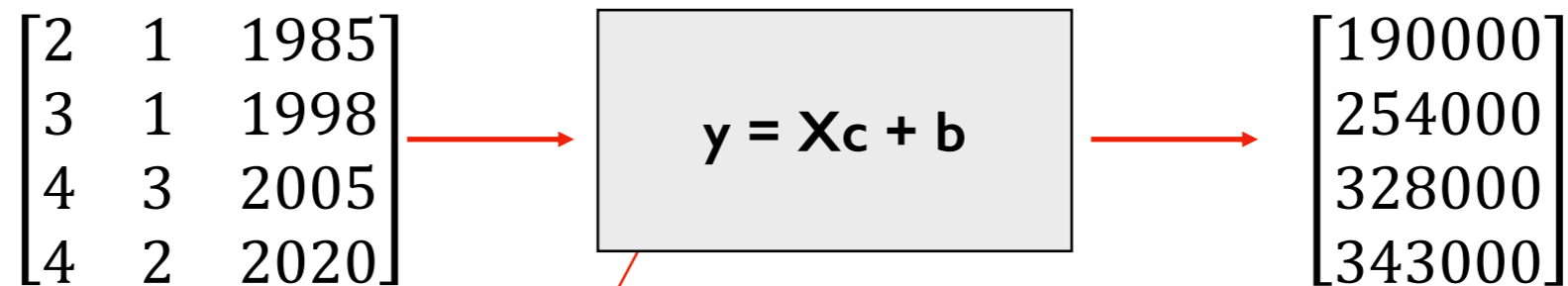
with matrices...



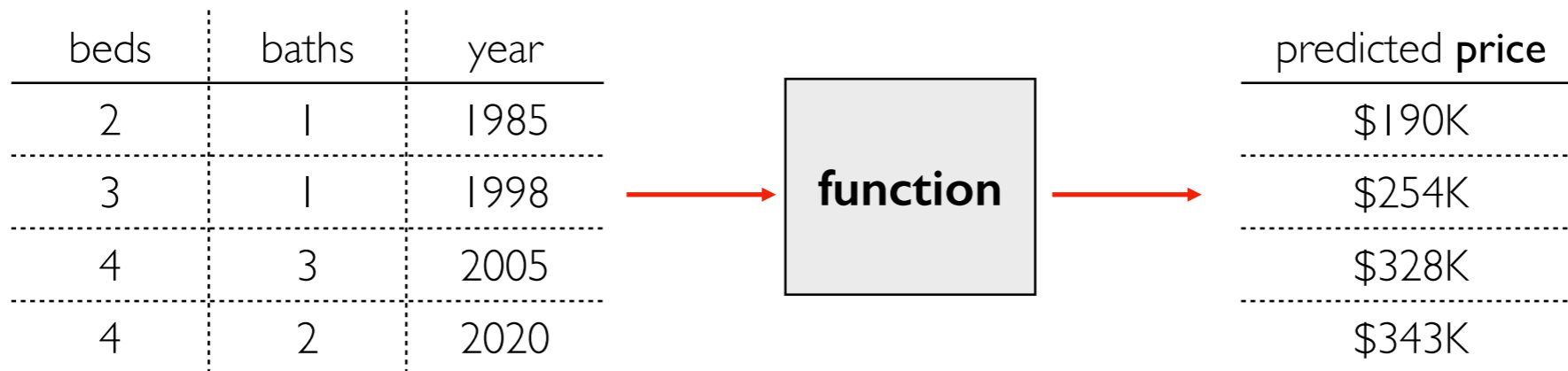
Linear Algebra



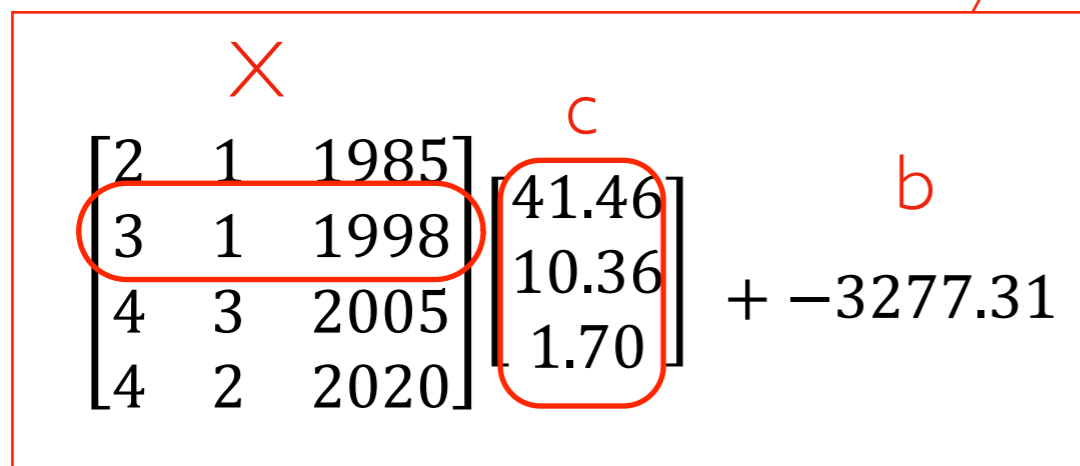
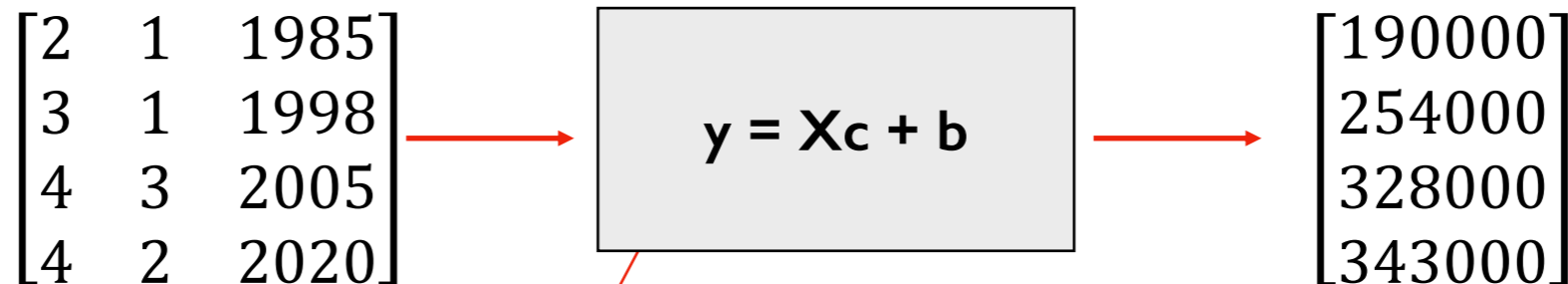
with matrices...



Linear Algebra

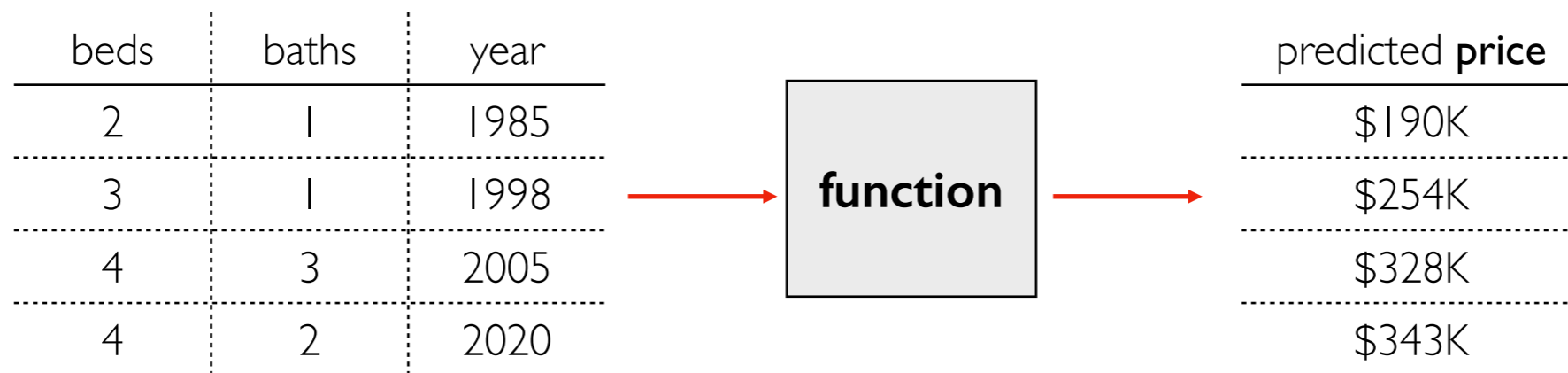


with matrices...



$$\begin{array}{ccccccc}
 \times & c & \times & c & \times & c & b \\
 3 * 41.46 + 1 * 10.36 + 1998 * 1.7 - 3277.31 & & & & & & \\
 & & & & & & = 254000 \\
 & & & & & & y
 \end{array}$$

Linear Algebra



with matrices...

note! Some resources will use **A** instead of **X** and **x** instead of **c**

$$\begin{bmatrix} 2 & 1 & 1985 \\ 3 & 1 & 1998 \\ 4 & 3 & 2005 \\ 4 & 2 & 2020 \end{bmatrix}$$

$$y = Xc + b$$

$$\begin{bmatrix} 190000 \\ 254000 \\ 328000 \\ 343000 \end{bmatrix}$$

$$\begin{matrix} X \\ \begin{bmatrix} 2 & 1 & 1985 \\ 3 & 1 & 1998 \\ 4 & 3 & 2005 \\ 4 & 2 & 2020 \end{bmatrix} \end{matrix} \begin{matrix} c \\ \begin{bmatrix} 41.46 \\ 10.36 \\ 1.70 \end{bmatrix} \end{matrix} + \begin{matrix} b \\ -3277.31 \end{matrix}$$

```
import numpy as np
X = df.values
y = X @ c + b
```

Linear Algebra

$$y = x^{**2} \quad \text{not linear}$$

$$y = x_0*4 + x_1*(-1) + x_2*0.5 + \dots + x_{10}*3 \quad \text{linear}$$

with matrices...

$$\begin{bmatrix} 2 & 1 & 1985 \\ 3 & 1 & 1998 \\ 4 & 3 & 2005 \\ 4 & 2 & 2020 \end{bmatrix} \longrightarrow \boxed{y = Xc + b} \longrightarrow \begin{bmatrix} 190000 \\ 254000 \\ 328000 \\ 343000 \end{bmatrix}$$

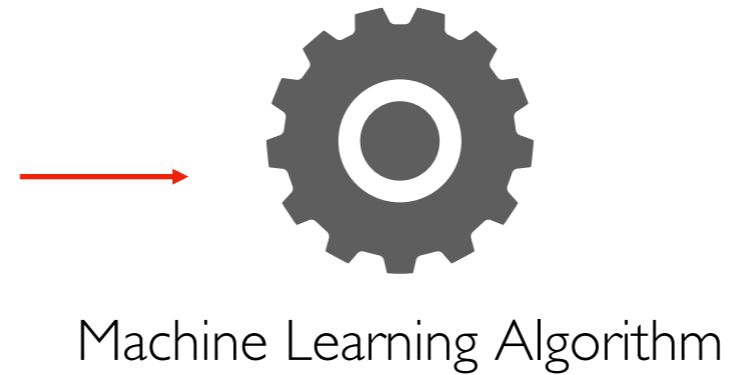
$$\begin{matrix} X & c & b \\ \begin{bmatrix} 2 & 1 & 1985 \\ 3 & 1 & 1998 \\ 4 & 3 & 2005 \\ 4 & 2 & 2020 \end{bmatrix} & \begin{bmatrix} 41.46 \\ 10.36 \\ 1.70 \end{bmatrix} & + -3277.31 \end{matrix}$$

```
import numpy as np
X = df.values
y = X @ c + b
```

Calculus: Minimizing Something

training data

| beds | baths | year | price |
|------|-------|------|--------|
| 1 | 1 | 1980 | \$140K |
| 3 | 1 | 1990 | \$240K |
| 3 | 4 | 2004 | \$295K |



train

$$y = Xc + b$$

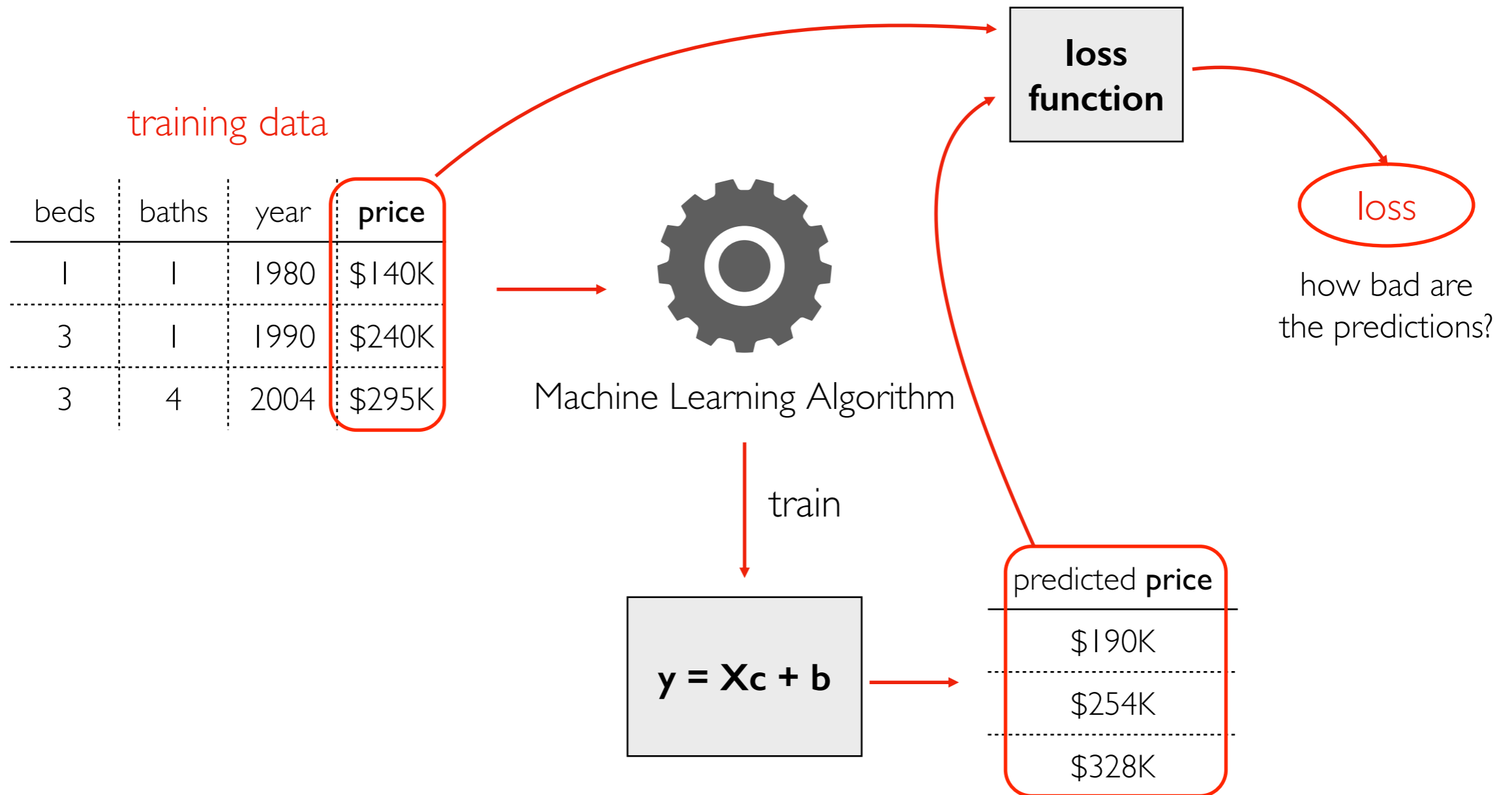
predicted price

\$190K

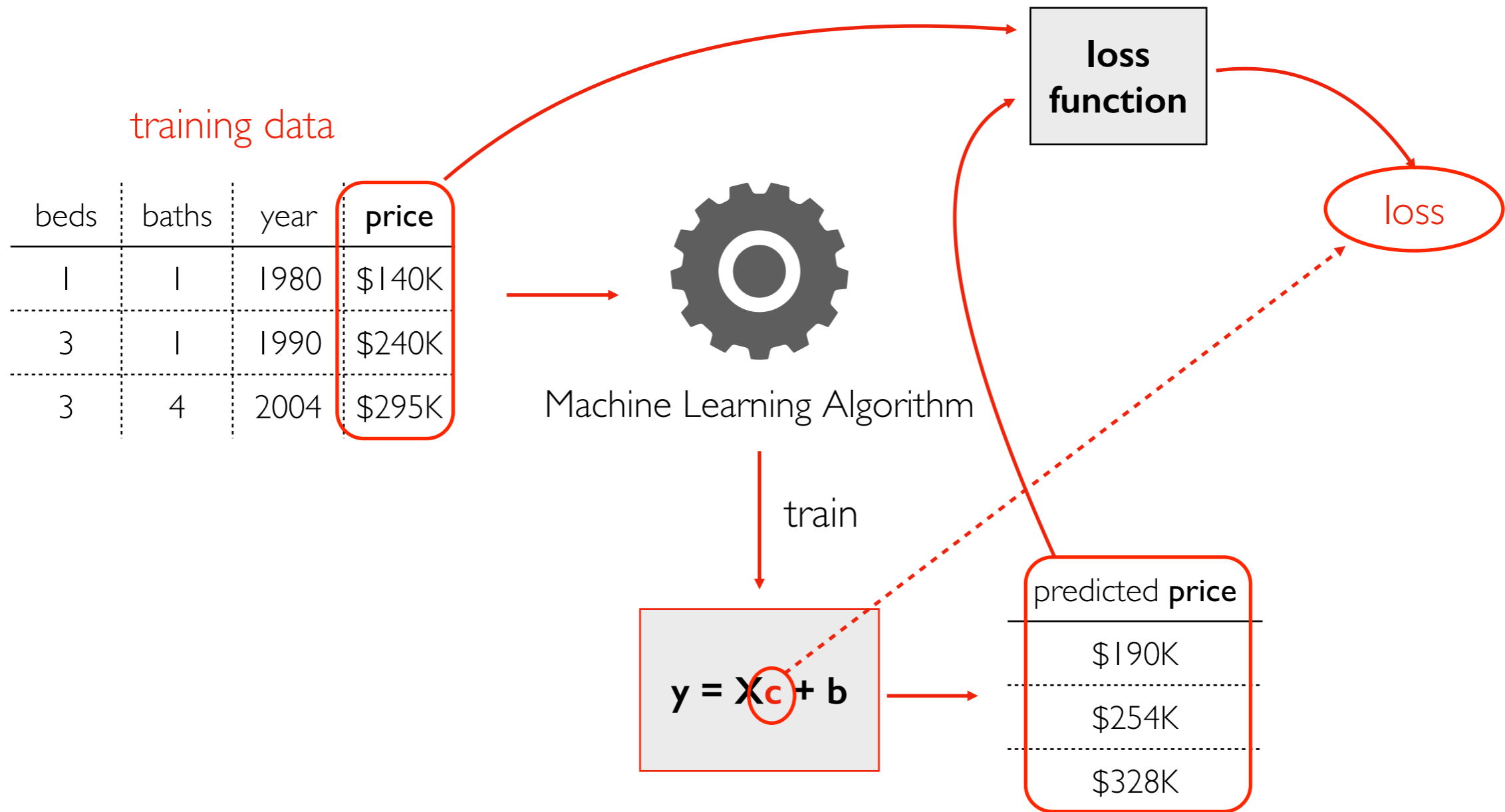
\$254K

\$328K

Calculus: Minimizing Something



Calculus: Minimizing Something

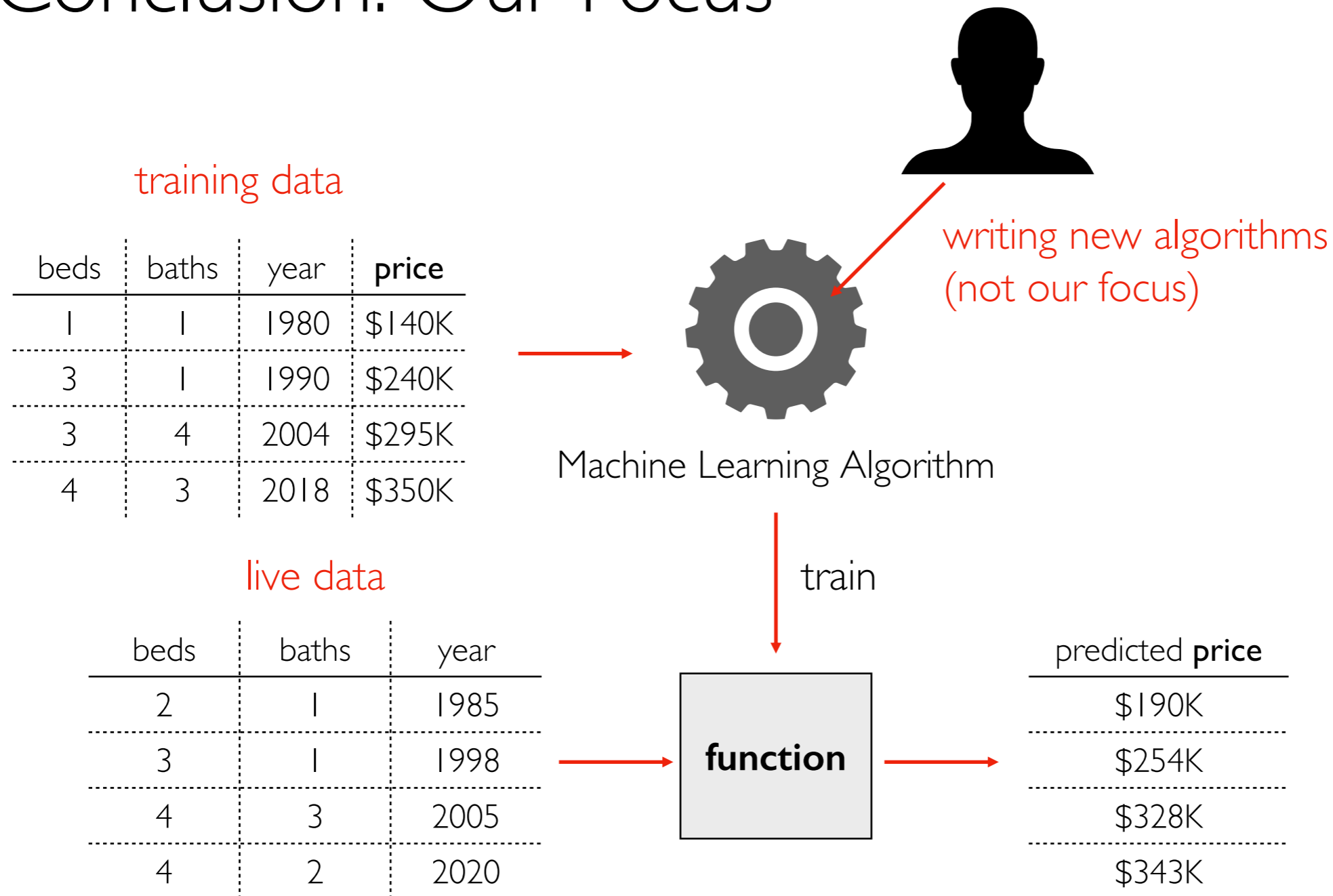


how do we optimize \mathbf{c} to minimize **loss**?
Important concepts: derivative, gradient

(pytorch can do this)

Conclusion: Developers vs. Users

Conclusion: Our Focus

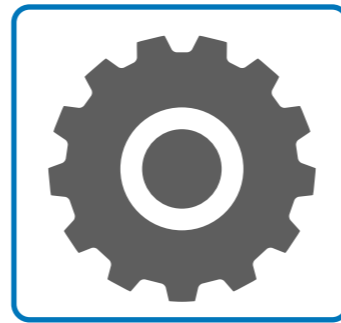


Conclusion: Our Focus

how can we clean this up?

training data

| beds | baths | year | price |
|------|-------|------|--------|
| 1 | 1 | 1980 | \$140K |
| 3 | 1 | 1990 | \$240K |
| 3 | 4 | 2004 | \$295K |
| 4 | 3 | 2018 | \$350K |



which algorithm (from sklearn?) should we pick, and how should we configure it?

Machine Learning Algorithm

is it working well?
(evaluation)

live data

| beds | baths | year |
|------|-------|------|
| 2 | 1 | 1985 |
| 3 | 1 | 1998 |
| 4 | 3 | 2005 |
| 4 | 2 | 2020 |

train

function

predicted price

\$190K

\$254K

\$328K

\$343K