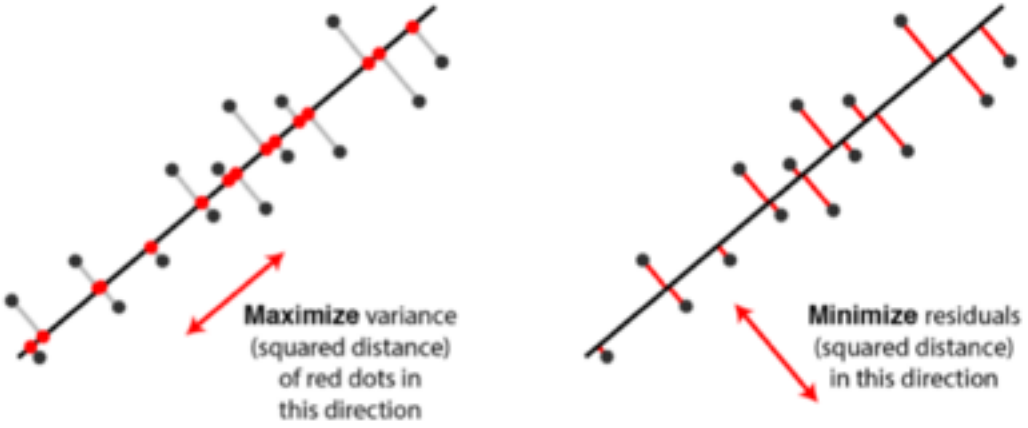
How can I interpret my ML system outputs?

Are my interpretations reasonable?

This course so far

How can one predict some value, find structure from data?



Two equivalent views of principal component analysis.

Let’s consider LASSO

We have come up with some sparse w for predicting y from x

How can we interpret the coefficients? Relative

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is
- # baths have a mean of 2 → w_i is

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is
- # baths have a mean of 2 → w_i is

How to fix?

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is
- # baths have a mean of 2 → w_i is

How to fix?

Normalize features to have mean and unit variance (z-score)

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is $/sqft
- # baths have a mean of 2 → w_i is

How to fix?

Normalize features to have mean and unit variance (z-score)

How can we interpret the coefficients? Relative

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i > \hat{w}_j$ means feature i is more important than feature j

FALSE

What if features have different scales?

- Sqft have a mean of 2500 → w_i is $/sqft
- # baths have a mean of 2 → w_i is $/bath

How to fix?

Normalize features to have mean and unit variance (z-score)

How can we interpret the coefficients? Zeros

How can we interpret the coefficients? Zeros

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2 + \lambda \|w\|_1$$ with

normalized data

Claim: $\hat{w}_i = 0$ means feature i has no predictive power for y

How can we interpret the coefficients? Zeros

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2 + \lambda \|w\|_1$$ with

normalized data

Claim: $\hat{w}_i = 0$ means feature i has no predictive power for y

FALSE

How can we interpret the coefficients? Zeros

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2 + \lambda \|w\|_1$$ with

normalized data

Claim: $\hat{w}_i = 0$ means feature i has no predictive power for y

FALSE

Feature i might be correlated with y but redundant with feature j

How can we interpret the coefficients? Zeros

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2 + \lambda \|w\|_1$$ with

normalized data

Claim: $\hat{w}_i = 0$ means feature i has no predictive power for y

FALSE

Feature i might be correlated with y but redundant with feature j
e.g. area in square feet and area in square meters

How can we interpret the coefficients? Distributional changes

How can we interpret the coefficients? Distributional changes

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i = 90{,}000$ and the ith feature = #fireplaces. If I add 10 more fireplaces, I can expect to sell my house for $900,000 more!

How can we interpret the coefficients? Distributional changes

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i = 90{,}000$ and the ith feature = #fireplaces. If I add 10 more fireplaces, I can expect to sell my house for $900,000 more!

FALSE

How can we interpret the coefficients? Distributional changes

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i = 90{,}000$ and the ith feature = #fireplaces. If I add 10 more fireplaces, I can expect to sell my house for $900,000 more!

FALSE

(1) Distribution shift: we've never seen a house with 10 fireplaces in the training data, so we don't have accurate predictions for this range. It's OOD: out of distribution

How can we interpret the coefficients? Distributional changes

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i = 90{,}000$ and the ith feature = #fireplaces. If I add 10 more fireplaces, I can expect to sell my house for $900,000 more!

FALSE

(1) Distribution shift: we've never seen a house with 10 fireplaces in the training data, so we don't have accurate predictions for this range. It's OOD: out of distribution

(2) Correlation ≠ causation

How can we interpret the coefficients? Distributional changes

Consider a linear model

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \left(y_i - w^\top x_i\right)^2$$

Claim: $\hat{w}_i = 90{,}000$ and the ith feature = #fireplaces. If I add 10 more fireplaces, I can expect to sell my house for $900,000 more!

FALSE

(1) Distribution shift: we've never seen a house with 10 fireplaces in the training data, so we don't have accurate predictions for this range. It's OOD: out of distribution

(2) Correlation ≠ causation

(3) Linear model, non-linear relationship

Generalization

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on hospital 4
```

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on hospital 4
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in a new hospital.

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on hospital 4
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in a new hospital.

FALSE

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on hospital 4
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in a new hospital.

FALSE

Distribution shift: new hospital could have different equipment, demographics, staff training, location…

Generalization

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in the **same hospitals.**

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on
hospitals 1, 2, 3
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in the **same hospitals.**

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on
hospitals 1, 2, 3
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in the **same hospitals.**

FALSE

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

```
# Train on hospitals 1, 2, 3 → Test on
hospitals 1, 2, 3
```

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in the **same hospitals.**

FALSE

Distribution shift: temporal shift. Conditions can change in our hospital by the time we deploy our system! e.g. new training procedures, new staff, new equipment, etc.

Generalization

Say we've trained a model to interpret medical x-rays using data from a few hospitals. We randomly split the data into train/validation/test splits.

Train on hospitals 1, 2, 3 → Test on hospitals 1, 2, 3

Claim: The test set performance is always a good indicator of how our model will do if we deploy this model in the **same hospitals.**

FALSE

Distribution shift: temporal shift. Conditions can change in our hospital by the time we deploy our system! e.g. new training procedures, new staff, new equipment, etc.

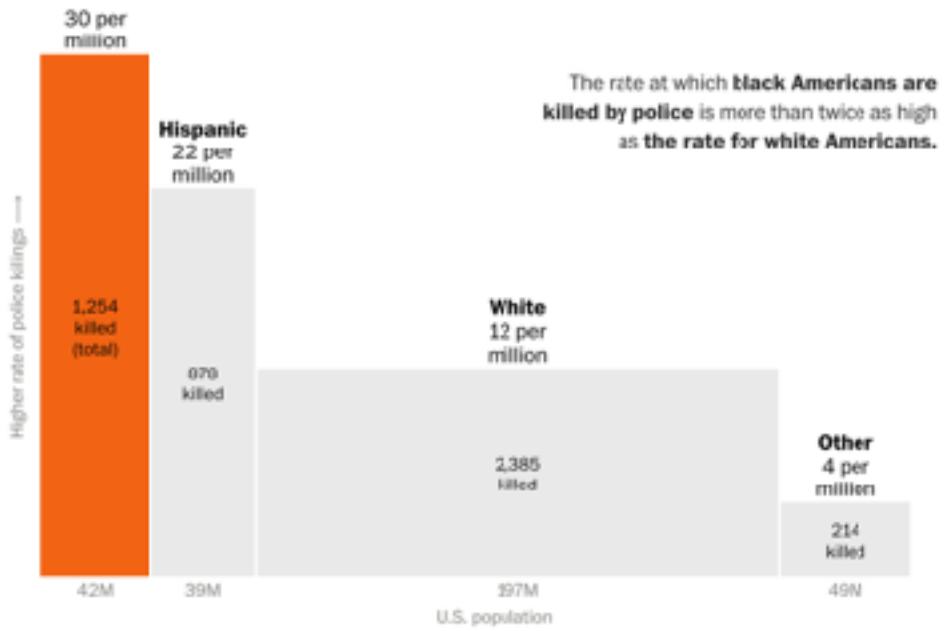
My particular hobby horse: the world is non-stationary!

Now, let’s talk about high-stakes decision making

Some sobering statistics

Many, many people are killed by police every year (10,429 in the past 9 years)

> half of which aren't reported to the FBI (reporting is voluntary)



Source: an excellent Washington Post interactive database

Design choices

Model Class

Features to collect

Loss function

Regularization

Optimization Method

Constraints

…

Design choices affect your model

Model Class

Features to collect

Loss function

Regularization

Optimization Method

Constraints

…

These will all affect what model you find, and how well and when that structure will generalize

… and that can have consequences "beyond" ML

Model Class

Features to collect

Loss function

Regularization

Optimization Method

Constraints

…

These will all affect what model you find, and how well and when that structure will generalize

Different models will have different kinds of errors, predictions, and failure modes

Consequence #1: Good predictions for whom?

The US Criminal Justice System

The US Criminal Justice System

This is a description of how things (largely) are and have been, not how they ought to be.

The US Criminal Justice System

This is a description of how things (largely) are and have been, not how they ought to be.

Any use of ML in the criminal justice system is just a tiny part of the larger CJ ecosystem.

The US Criminal Justice System

This is a description of how things (largely) are and have been, not how they ought to be.

Any use of ML in the criminal justice system is just a tiny part of the larger CJ ecosystem.

Moreover, “reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

The US Criminal Justice System

The US Criminal Justice System

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

The US Criminal Justice System

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

The US Criminal Justice System

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

The US Criminal Justice System

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

The US Criminal Justice System

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

The US Criminal Justice System

(Lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

-> A sentencing hearing determines the length of prison time, fines, …

The US Criminal Justice System

(Lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

-> A sentencing hearing determines the length of prison time, fines, …

-> With “good behavior”, people in prison can be considered for parole

What does the CJ process have to do with ML?

(Lots and lots and lots and lots of ~~possibly~~ biased processes, including racialized choices of where to police, what behaviors are most dangerous, …)

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

-> A sentencing hearing determines the length of prison time, fines, …

-> With “good behavior”, people in prison can be considered for parole

What does the CJ process have to do with ML?

Predictive policing: determine (statistically) where to send police

-> A person is charged with a crime

-> A judge determines whether to detain them or release them on bail until their trial

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

-> A sentencing hearing determines the length of prison time, fines, …

-> With “good behavior”, people in prison can be considered for parole

What does the CJ process have to do with ML?

Predictive policing: determine (statistically) where to send police

-> A person is charged with a crime

Predict probability of reappearing, being charged with another crime if on parole, …

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

-> A sentencing hearing determines the length of prison time, fines, …

-> With “good behavior”, people in prison can be considered for parole

What does the CJ process have to do with ML?

Predictive policing: determine (statistically) where to send police

-> A person is charged with a crime

Predict probability of reappearing, being charged with another crime if on parole, …

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

Predict risk of being charged with another crime, use in sentencing

-> With “good behavior”, people in prison can be considered for parole

What does the CJ process have to do with ML?

Predictive policing: determine (statistically) where to send police

-> A person is charged with a crime

Predict probability of reappearing, being charged with another crime if on parole, …

-> (Many, many months or years pass)

-> The trial determines whether the justice system considers a person guilty or innocent of charges

Predict risk of being charged with another crime, use in sentencing

Predict risk of being charged with another crime, use in parole decisions

The US Criminal Justice System

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

Largely, it corresponds to either being:

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

Largely, it corresponds to either being:

- **arrested**

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

Largely, it corresponds to either being:

- **arrested**
- **charged**

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

Largely, it corresponds to either being:

- **arrested**
- **charged**
- **or being found guilty of a crime**

The US Criminal Justice System

“reoffend” or “recidivate” are very, very, very far from precise terms in their use (in this lecture, and in industry).

Largely, it corresponds to either being:

- **arrested**
- **charged**
- **or being found guilty of a crime**

usually one which is violent.

Can such predictions be fair with respect to race?

Can such predictions be fair with respect to race?

Attempt #1: don't use race as a feature

- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race

- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

Can such predictions be fair with respect to race?

Attempt #1: don't use race as a feature

- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race

- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

To accurately predict which people are most likely commit a crime if released, so they can be released

Can such predictions be fair with respect to race?

Attempt #1: don't use race as a feature

- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race

- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

To accurately predict which people are most likely commit a crime if released, so they can be released

Can such predictions be fair with respect to race?

Attempt #1: don't use race as a feature

- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race

- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

To accurately predict which people are most likely commit a crime if released, so they can be released

… where these predictions should be accurate? similar? for people of all races.

Can such predictions be fair with respect to race?

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

Attempt #1: don't use race as a feature
- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race
- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

To accurately predict which people are most likely commit a crime if released, so they can be released

… where these predictions should be accurate? similar? for people of all races.

Can such predictions be fair with respect to race?

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

Attempt #1: don't use race as a feature
- Issue: race is strongly correlated with other features

Attempt #2: Remove any feature correlated with race
- Issue: (nearly) all features correlated with race

"Fairness through unawareness"

What exactly is the goal of this?

To accurately predict which people are most likely commit a crime if released, so they can be released

Concern: not all errors are equally costly.

… where these predictions should be accurate? similar? for people of all races.

Can such predictions be fair with respect to race?

Can such predictions be fair with respect to race?

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

Can such predictions be fair with respect to race?

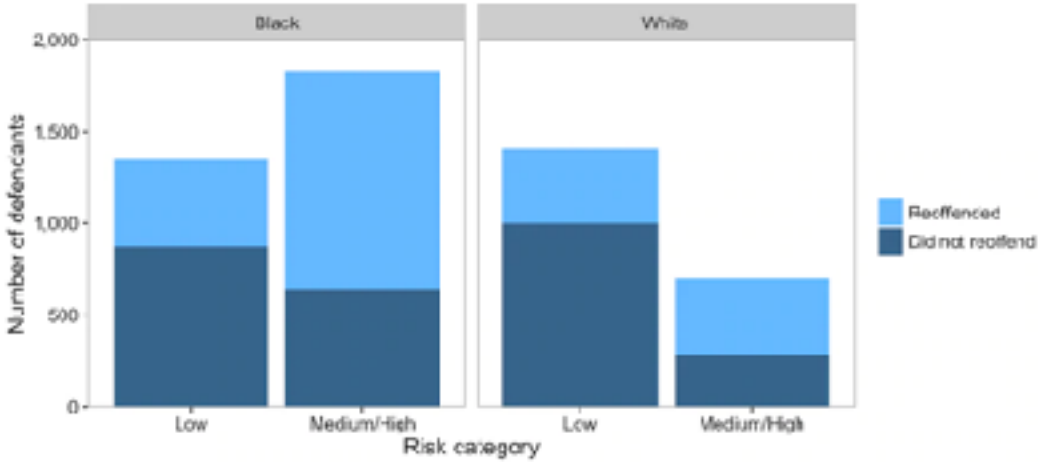
Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

Source: Washington Post

Can such predictions be fair with respect to race?

NORTHPOINTE

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

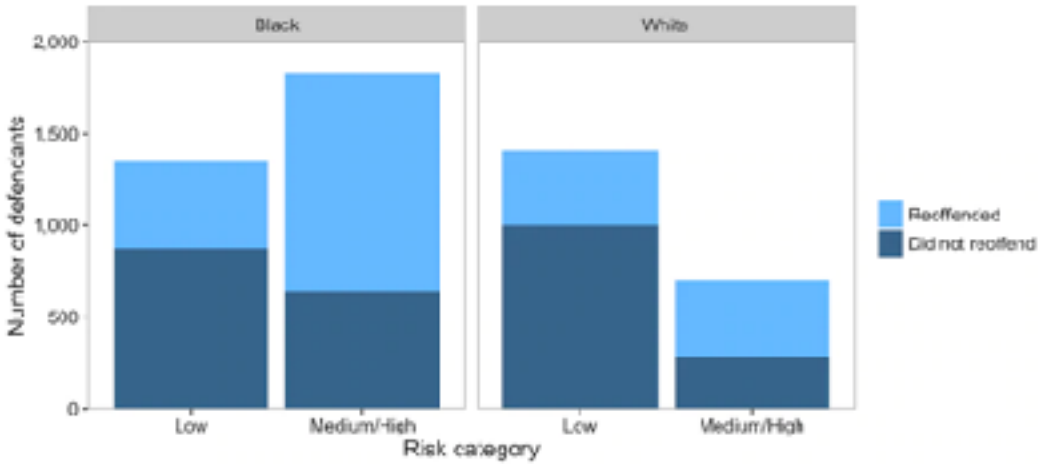


Source: Washington Post

Can such predictions be fair with respect to race?

NORTHPOINTE

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale



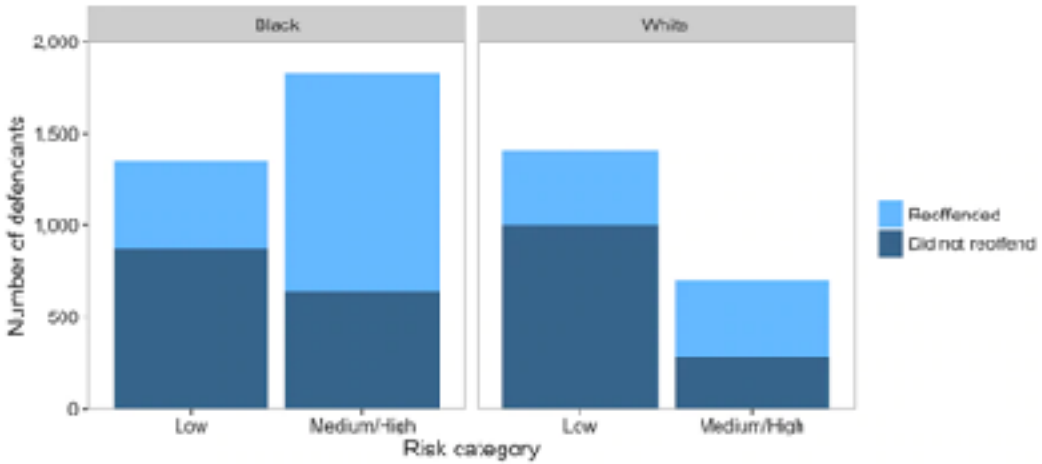
$$\mathbb{P}[\text{does not reoffend}|\text{predicted low risk, white}] < \mathbb{P}[\text{does not reoffend}|\text{predicted low risk, black}]$$

Source: Washington Post

Can such predictions be fair with respect to race?

NORTHPOINTE

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale



$$\mathbb{P}[\text{does not reoffend}|\text{predicted low risk, white}] < \mathbb{P}[\text{does not reoffend}|\text{predicted low risk, black}]$$

False positive rate for black defendants higher than for white defendants.

Source: Washington Post

Can such predictions be fair with respect to race?

Calibration of scores w.r.t. race

Can such predictions be fair with respect to race?

Calibration of scores w.r.t. race

Source: Washington Post

Can such predictions be fair with respect to race?

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale

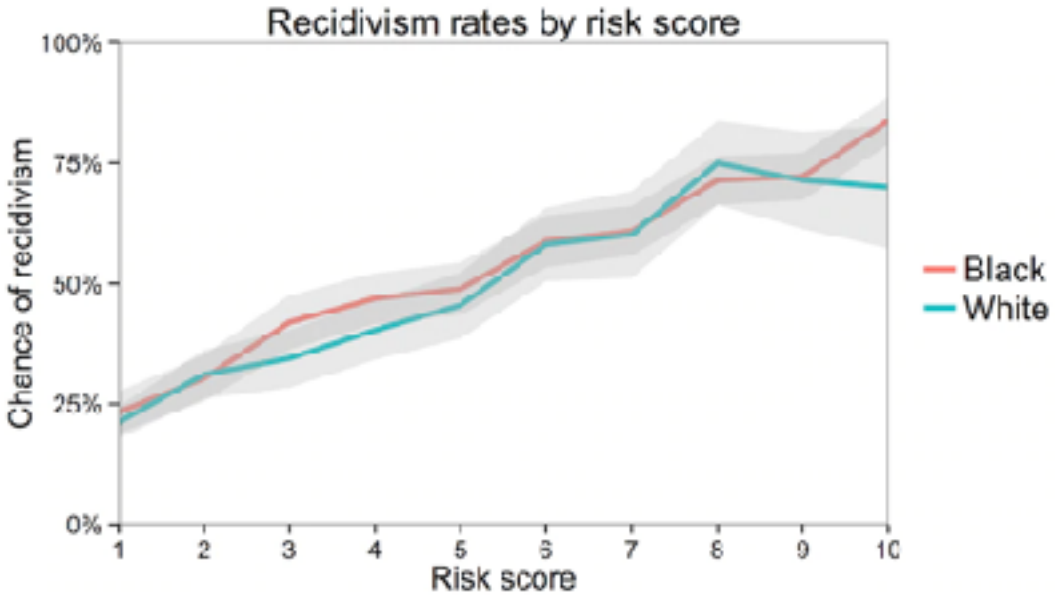
Calibration of scores w.r.t. race

Source: Washington Post

Can such predictions be fair with respect to race?

NORTHPOINTE

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale



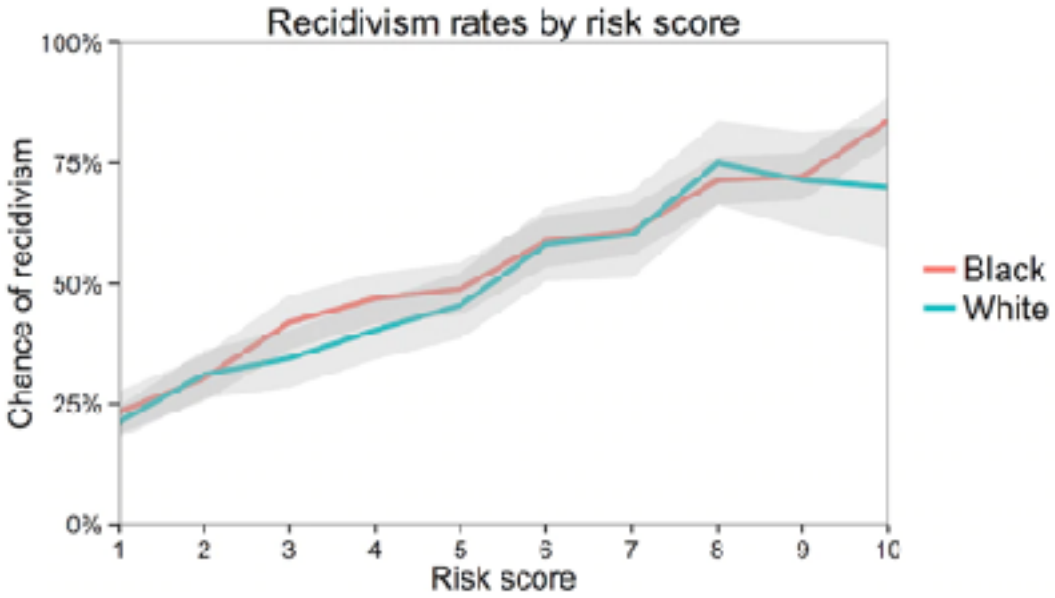
Calibration of scores w.r.t. race

Source: Washington Post

Can such predictions be fair with respect to race?

NORTHPOINTE

Pretrial release risk scale: 1-10
General recidivism scale
Violent recidivism scale



$$\mathbb{P}[\text{reoffend}|\text{prediction, white}] \approx \mathbb{P}[\text{reoffend}|\text{prediction, black}]$$

Calibration of scores w.r.t. race

Source: Washington Post

Can any predictions be fair with respect to race?

$$\mathbb{P}[\text{does not reoffend}|\text{predicted low risk, white}] < \mathbb{P}[\text{does not reoffend}|\text{predicted low risk, black}]$$

while

$$\mathbb{P}[\text{reoffend}|\text{prediction, white}] \approx \mathbb{P}[\text{reoffend}|\text{prediction, black}]$$

How is this possible?

Is this avoidable?

Can any predictions be fair with respect to race?

False positives approximately equal and calibration are mutually exclusive, unless we have perfect predictions or the rate of what we are predicting is equal in every racial group.

… and this is true for more settings than just criminal justice.

Lending
Advertising
…

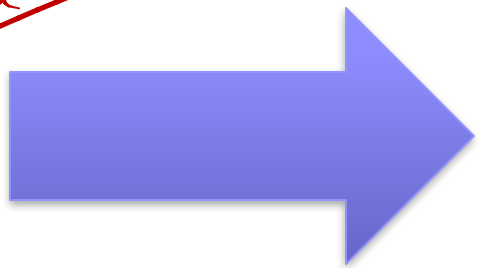
We are making a choice about how we allocate predictions.

Classification! Logistic "Regression" (linear classification), oh my…

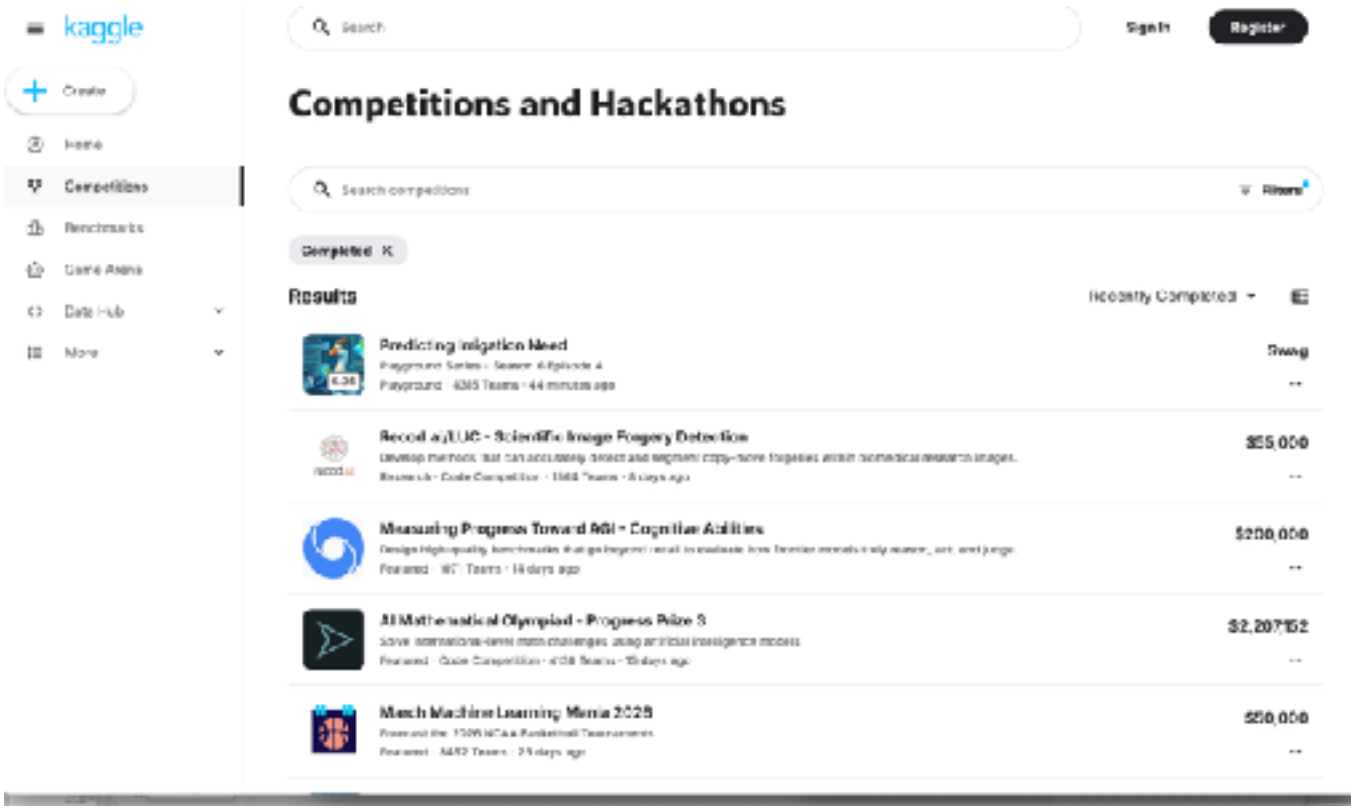
Thus far, regression:

predict a continuous value given some inputs

Weather prediction revisted



Recent Kaggle competitions



Classification

- Learn f: X -> Y
  - X - features
  - Y - target classes

- Loss Function
- Expected loss of f:

- Suppose you knew P(YIX) exactly, how should you classify?
  - Bayes-Optimal classifier:

Classification

- Learn f: X -> Y
  - X - features
  - Y - target classes

- Loss Function

$$\ell(f(x), y) = \mathbf{1}\{f(x) \neq y\}$$

- Expected loss of f:

$$\mathbb{E}_{XY}[\mathbf{1}\{f(X) \neq Y\}] = \mathbb{E}_X[\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X = x]]$$

$$\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X = x] = \sum_i P(Y = i|X = x)\mathbf{1}\{f(x) \neq i\} = \sum_{i \neq f(x)} P(Y = i|X = x)$$

$$= 1 - P(Y = f(x)|X = x)$$

- Suppose you knew P(YIX) exactly, how should you classify?
  - Bayes-Optimal classifier:

$$f(x) = \arg\max_y \mathbb{P}(Y = y|X = x)$$

Binary Classification

- Learn f: X -> Y
  - X - features
  - Y - target classes

$$Y \in \{0, 1\}$$

- Loss Function

$$\ell(f(x), y) = \mathbf{1}\{f(x) \neq y\}$$

- Expected loss of f:

$$\mathbb{E}_{XY}[\mathbf{1}\{f(X) \neq Y\}] = \mathbb{E}_X[\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X = x]]$$

$$\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X = x] = \sum_i P(Y = i|X = x)\mathbf{1}\{f(x) \neq i\} = \sum_{i \neq f(x)} P(Y = i|X = x)$$

$$= 1 - P(Y = f(x)|X = x)$$

- Suppose you knew P(YIX) exactly, how should you classify?
  - Bayes-Optimal classifier:

$$f(x) = \arg\max_y \mathbb{P}(Y = y|X = x)$$

Bayes Optimal Binary Classifier

$$Y \in \{0, 1\}$$

- Suppose you knew P(YIX) exactly, how should you classify?
  - Bayes-Optimal classifier:

$$f(x) = \arg\max_y \mathbb{P}(Y = y | X = x)$$

- Suppose we don't know P(YIX), but have n iid examples

$$\{(x_i, y_i)\}_{i=1}^n$$

- What is a natural estimator for P(Y I X)?

Bayes Optimal Binary Classifier

- **Suppose we don't know P(YIX), but have n iid examples**

$$\{(x_i, y_i)\}_{i=1}^n \qquad Y \in \{0, 1\}$$

- **What is a natural estimator for P(Y I X)?**

$$\text{Fix some } \tilde{x} \in X$$

$$\text{Suppose } x_i = \tilde{x} \text{ for } m \leq n \text{ samples}$$

$$\text{What is a natural estimator for } \theta_* := \mathbb{P}(Y = 1 | X = \tilde{x})?$$

$$\text{If } k \text{ of the } m \text{ labels are equal to } Y = 1 \text{ then}$$

Bayes Optimal Binary Classifier

- **Suppose we don't know P(YIX), but have n iid examples**

$$\{(x_i, y_i)\}_{i=1}^n \qquad Y \in \{0, 1\}$$

- **What is a natural estimator for argmax_y P(Y = y I X)?**

$$\text{If } X = \{0,1\}^d, \text{ or is generally discrete}$$

$$\hat{f}(x) = \arg\max_{y \in \{0,1\}} \frac{\sum_{i=1}^n \mathbf{1}[\mathbf{x_i}=\mathbf{x},\mathbf{y_i}=\mathbf{y}]}{\sum_{i=1}^n \mathbf{1}[\mathbf{x_i}=\mathbf{x}]}$$

Issues?

Bayes Optimal Binary Classifier

- **What is a natural estimator for argmax_y P(Y = y I X)?**

If $X = \{0,1\}^d$, or is generally discrete $\quad Y \in \{0,1\}$

$$\hat{f}(x) = \arg\max_{y \in \{0,1\}} \frac{\sum_{i=1}^{n} \mathbf{1}[\mathbf{x_i}=\mathbf{x},\mathbf{y_i}=\mathbf{y}]}{\sum_{i=1}^{n} \mathbf{1}[\mathbf{x_i}=\mathbf{x}]}$$

Issues?

2^d possible inputs, for small d requires huge n

To make predictions for unseen inputs (xs),

need a **general** model for $\mathbb{P}(Y = 1|X = x)$

Process

Decide on a **model**

Find the function which fits the data best

- **Choose a loss function**
- **Pick the function which minimizes loss on data**

Use function to make prediction on new examples

Decide on a model, Binary Classification

To make predictions for unseen inputs (xs),

need a **general** model for $\mathbb{P}(Y = 1|X = x)$

- **What about standard linear regression model?**

- **Need to map real values to [0,1]**
  - **We call such maps "link functions"**

Logistic Regression

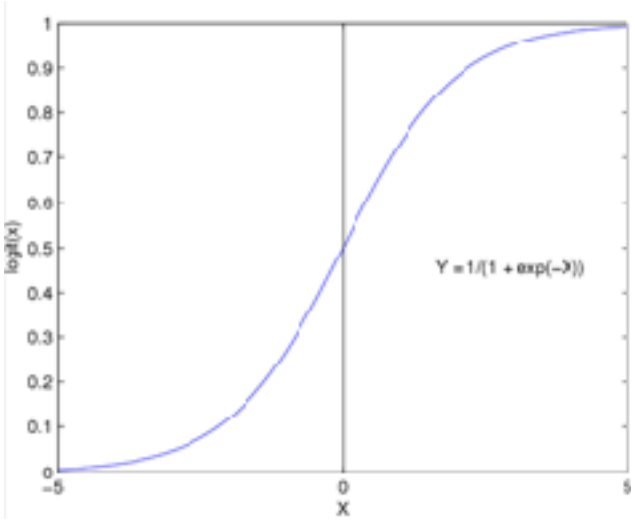
Actually classification, not regression :)

$$\text{Learn } \mathbb{P}(Y = 1|X = x) \text{ using } \sigma(w^T x), \text{ for link function } \sigma =$$

Logistic function(or Sigmoid): $\dfrac{1}{1 + exp(-z)}$

$$\mathbb{P}[Y = 1|X = x, w] = \sigma(w^T x) = \frac{1}{1 + \exp(-w^T x)}$$

$$\mathbb{P}[Y = 0|X = x, w] = 1 - \sigma(w^T x) = \frac{\exp(-w^T x)}{1 + \exp(-w^T x)} = \frac{1}{1 + \exp(w^T x)}$$

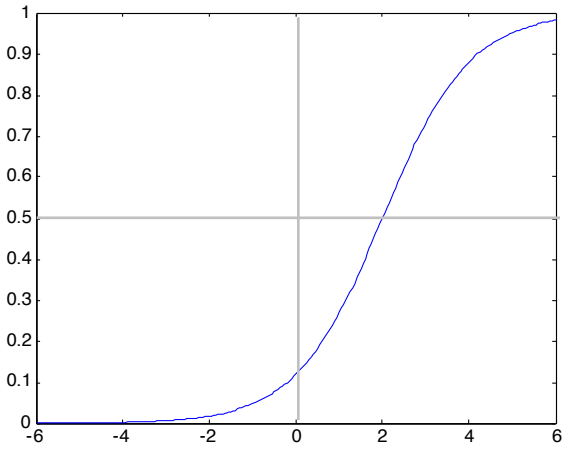


Features can be discrete or continuous!

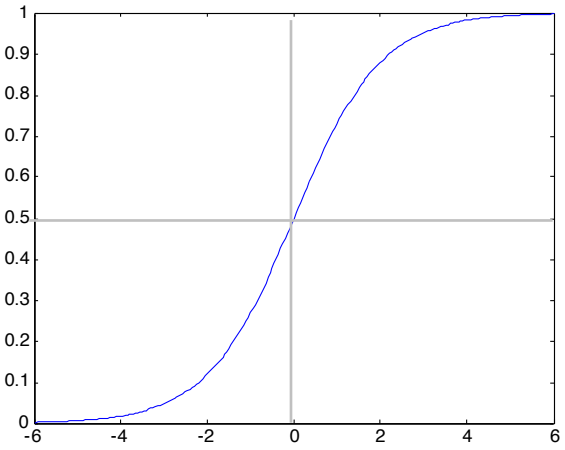
Understanding the sigmoid

$$\sigma(w_0 + \sum_k w_k x_k) = \frac{1}{1 + e^{w_0 + \sum_k w_k x_k}}$$

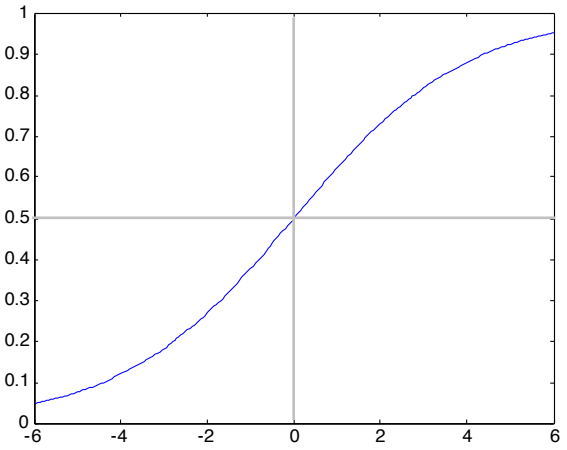
w_0=-2, w_1=-1



w_0=0, w_1=-1



w_0=0, w_1=-0.5



Sigmoid for binary classes

$$\mathbb{P}(Y=0|w,X) = \frac{1}{1+\exp(w_0+\sum_k w_k X_k)}$$

$$\mathbb{P}(Y=1|w,X) = 1-\mathbb{P}(Y=0|w,X) = \frac{\exp(w_0+\sum_k w_k X_k)}{1+\exp(w_0+\sum_k w_k X_k)}$$

$$\frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} =$$

Sigmoid for binary classes

$$\mathbb{P}(Y=0|w,X) = \frac{1}{1+\exp(w_0+\sum_k w_k X_k)}$$

$$\mathbb{P}(Y=1|w,X) = 1-\mathbb{P}(Y=0|w,X) = \frac{\exp(w_0+\sum_k w_k X_k)}{1+\exp(w_0+\sum_k w_k X_k)}$$

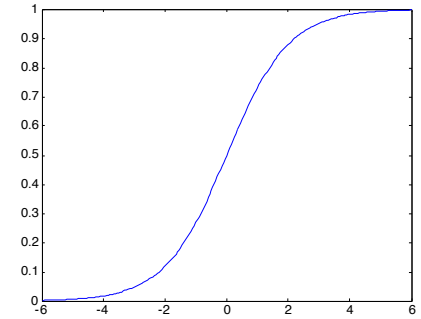
$$\frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} = \exp(w_0+\sum_k w_k X_k)$$

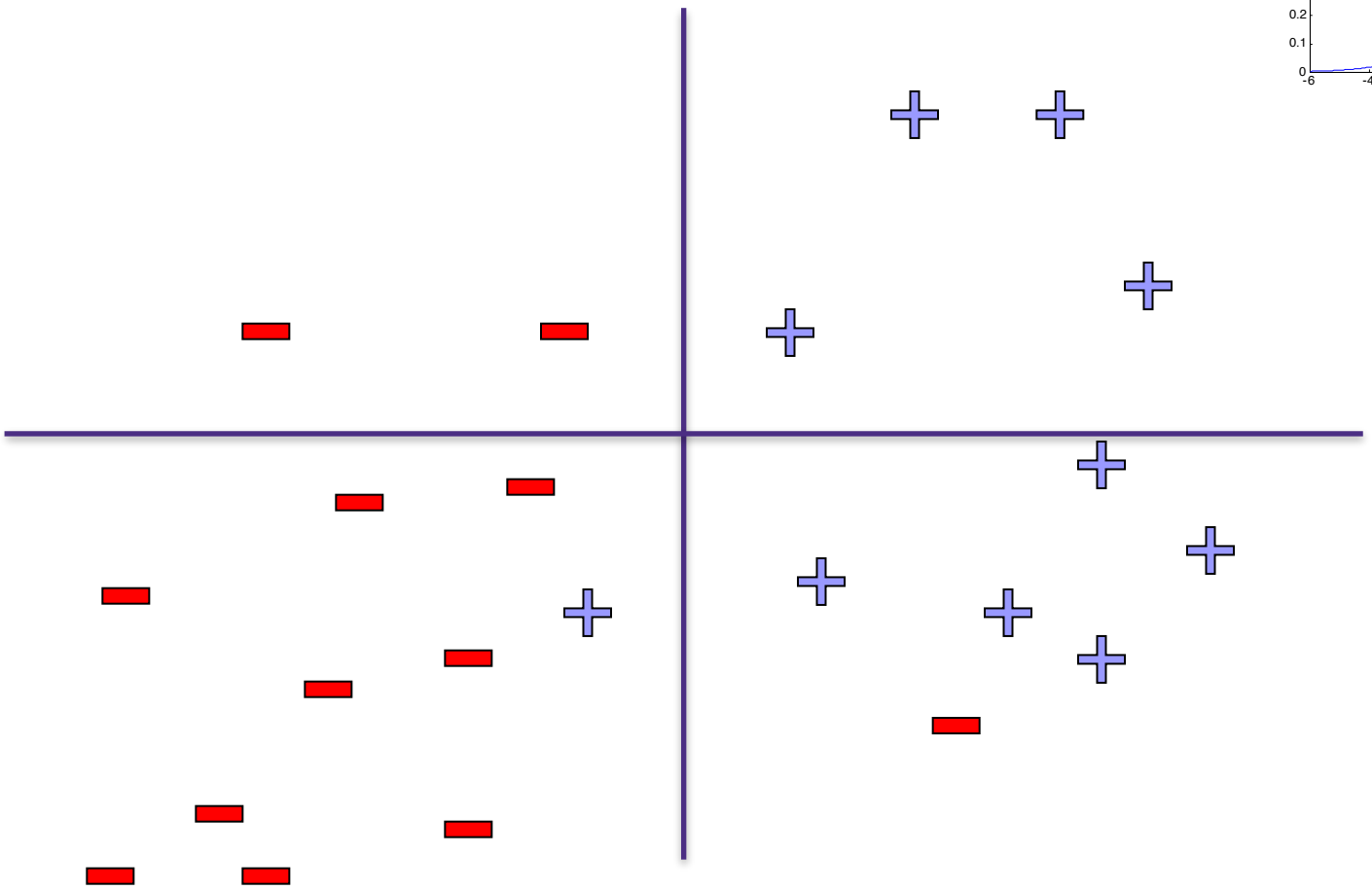
Linear Decision Rule!

$$\log\frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} = w_0+\sum_k w_k X_k$$

Logistic Regression – a Linear classifier

$$\frac{1}{1 + exp(-z)}$$





$$\ln \frac{P(Y=0|X)}{P(Y=1|X)} = w_0 + \sum_i w_i X_i$$

Process

Decide on a **model**

Find the function which fits the data best

- **Choose a loss function**
- **Pick the function which minimizes loss on data**

Use function to make prediction on new examples

Loss function: Conditional Likelihood

- **Have a bunch of iid data:** $\{(x_i, y_i)\}_{i=1}^{n} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$P(Y = -1|x, w) = \frac{1}{1 + \exp(w^T x)}$$

$$P(Y = 1|x, w) = \frac{\exp(w^T x)}{1 + \exp(w^T x)}$$

- **This is equivalent to:**

$$P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

- **So we can compute the maximum likelihood estimator:**

$$\widehat{w}_{MLE} = \arg\max_{w} \prod_{i=1}^{n} P(y_i|x_i, w)$$

Loss function: Conditional Likelihood

- **Have a bunch of iid data:** $\{(x_i, y_i)\}_{i=1}^{n} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^{n} P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^{n} \log(1 + \exp(-y_i\, x_i^T w))$$

Loss function: Conditional Likelihood

- **Have a bunch of iid data:** $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

Logistic Loss: $\ell_i(w) = \log(1 + \exp(-y_i\, x_i^T w))$

Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$ (MLE for Gaussian noise)

Process

Decide on a **model**

Find the function which fits the data best

- **Choose a loss function**
- **Pick the function which minimizes loss on data**

Use function to make prediction on new examples

Loss function: Conditional Likelihood

- **Have a bunch of iid data:** $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w)) = J(w)$$

What does $J(w)$ look like? Is it convex?

Loss function: Conditional Likelihood

$$\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w)) = J(w)$$

Good news: $J(\mathbf{w})$ is convex function of $\mathbf{w}$, no local optima problems

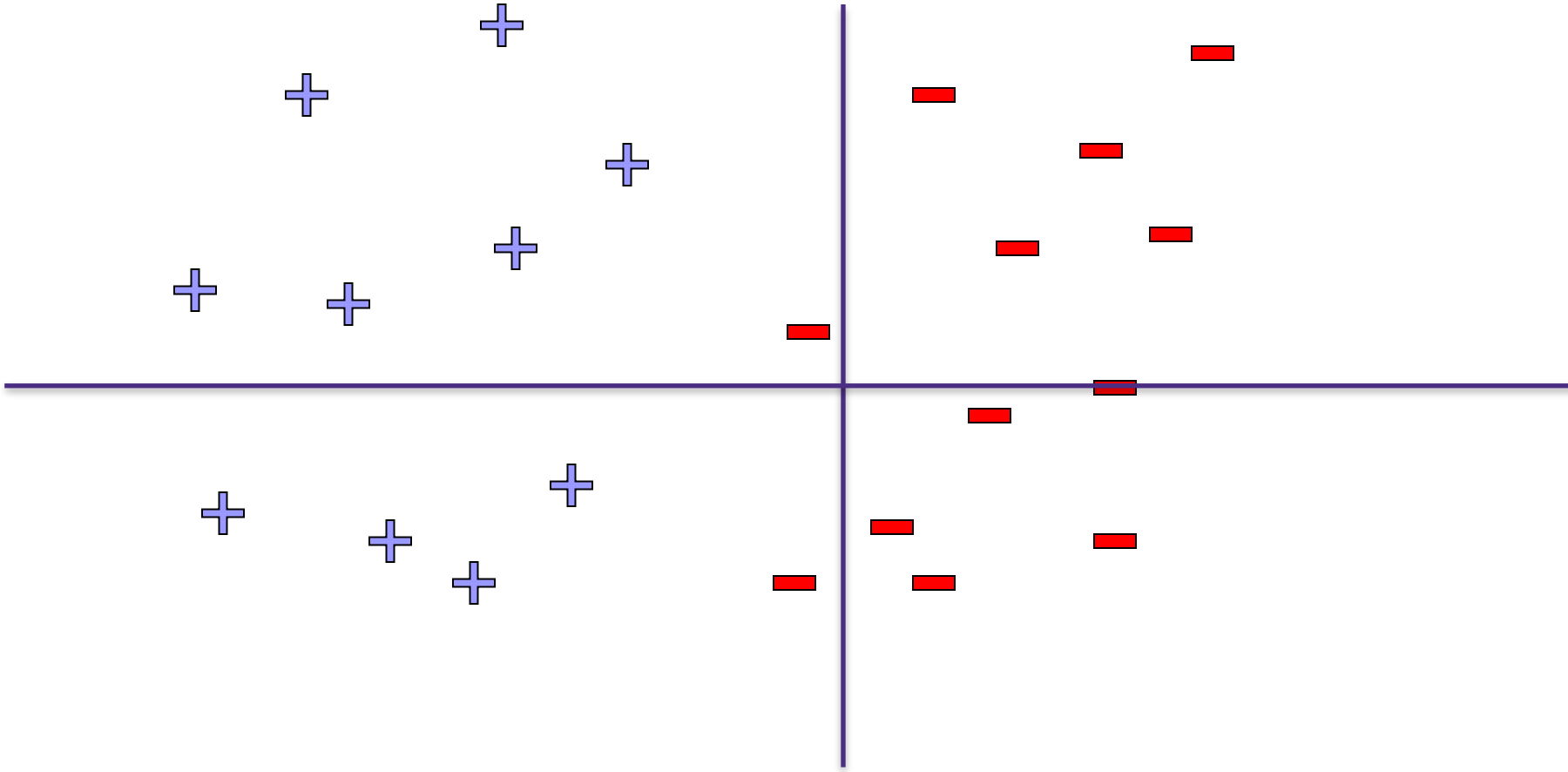
Bad news: no closed-form solution to maximize $J(\mathbf{w})$

Good news: convex functions easy to optimize

Linear Separability

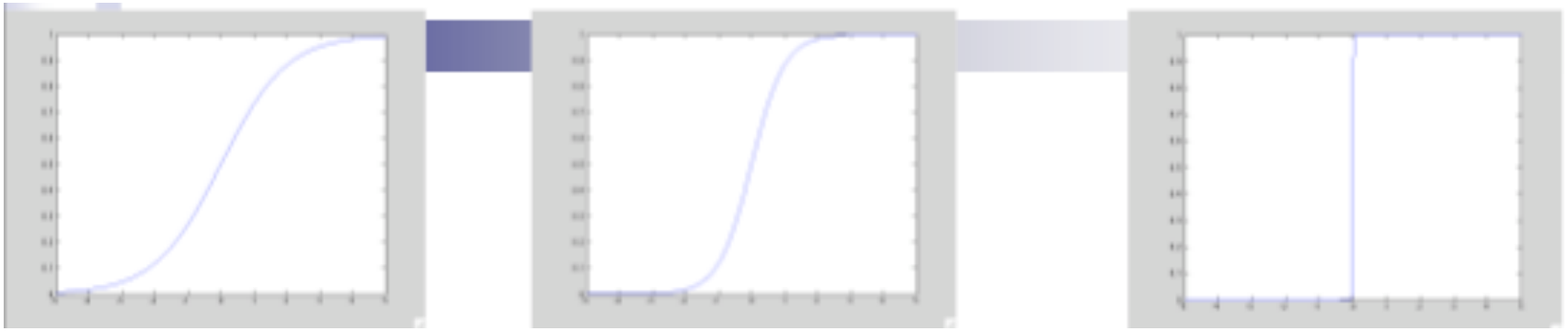
$$\arg\min_{w} \sum_{i=1}^{n} \log(1 + \exp(-y_i \, x_i^T w))$$

When is this loss small?



Large parameters → Overfitting

When data is linearly separable, weights ⇒ ∞



$$\frac{1}{1+e^{-x}} \qquad \frac{1}{1+e^{-2x}} \qquad \frac{1}{1+e^{-100x}}$$

Overfitting

Penalize high weights to prevent overfitting?

Regularized Conditional Log Likelihood

Add a penalty to avoid high weights/overfitting?:

$$\arg\min_{w,b} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i\left(x_i^T w + b\right))\right) + \lambda||w||_2^2$$

Be sure to not regularize the offset b!

Multi-class classification

Multi-class classification

- So far: binary $y \in \{-1, +1\}$

Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$

Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Multi-class classification

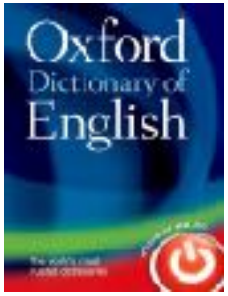
- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

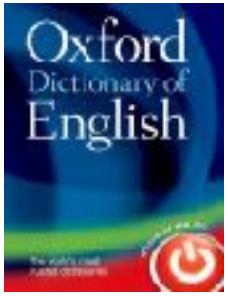
Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



- A k-class classifier predicts y given x

How to handle categorical labels

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

How to handle categorical labels

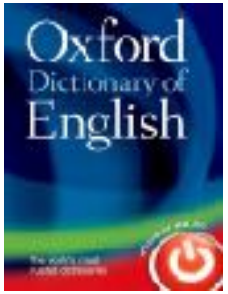
- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

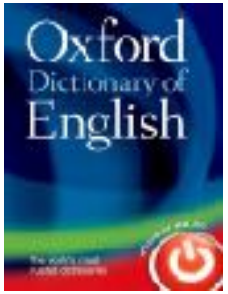
How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



How to handle categorical labels

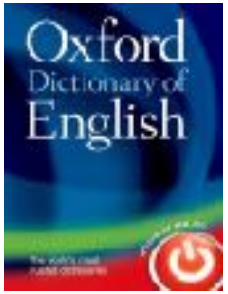
- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



- A k-class classifier predicts y given x

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

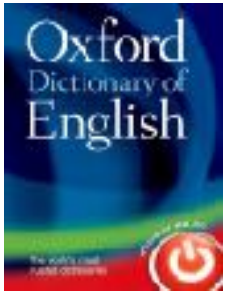


- A k-class classifier predicts y given x

Ideas for how to use logistic regression for multi class prediction:

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



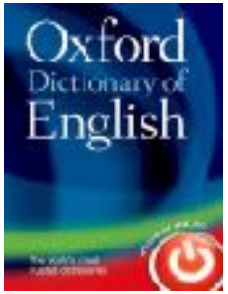
- A k-class classifier predicts y given x

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i

How to handle categorical labels

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels



- A k-class classifier predicts y given x

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i
- Try to predict $\mathbb{P}[c_i \mid x]$ as a linear function $w_i \cdot x$ for each i

One-HOT encoding labels + softmax

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

  Ideas for how to use logistic regression for multi class prediction:

  - Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

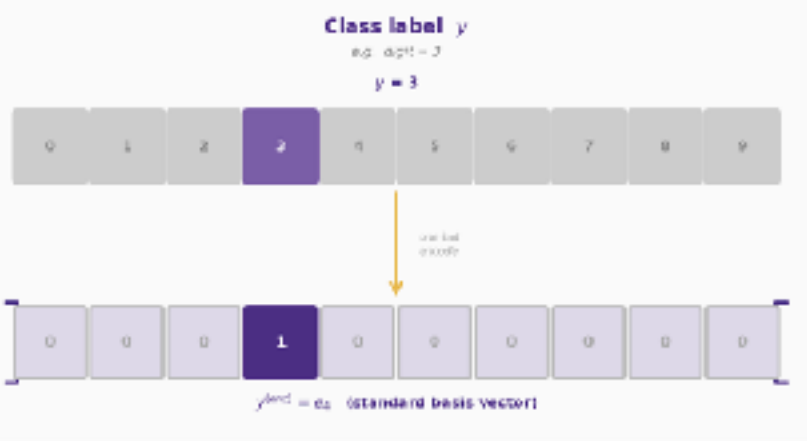
- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i
- Try to predict $\mathbb{P}[c_i \mid x]$ as a linear function $w_i \cdot x$ for each i

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i

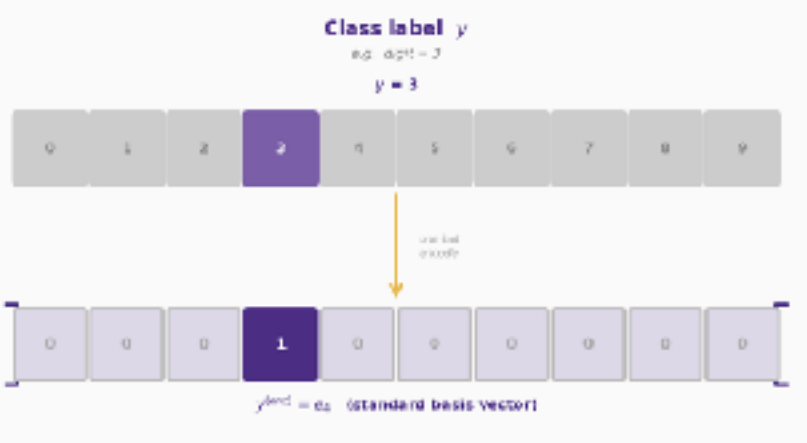


One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



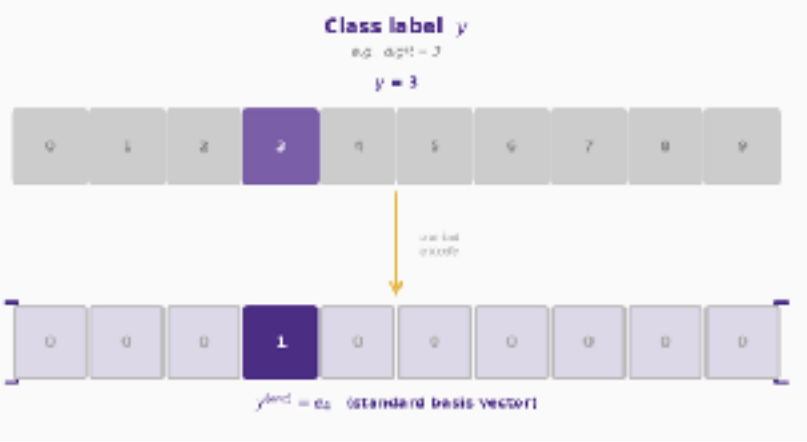
$\mathbb{P}[c_1 \,|\, x]$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



$\mathbb{P}[c_1 \,|\, x]$

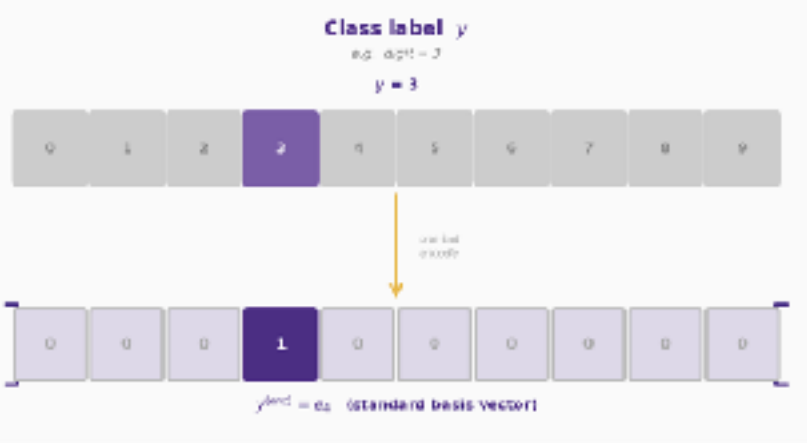
$\mathbb{P}[c_2 \,|\, x]$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



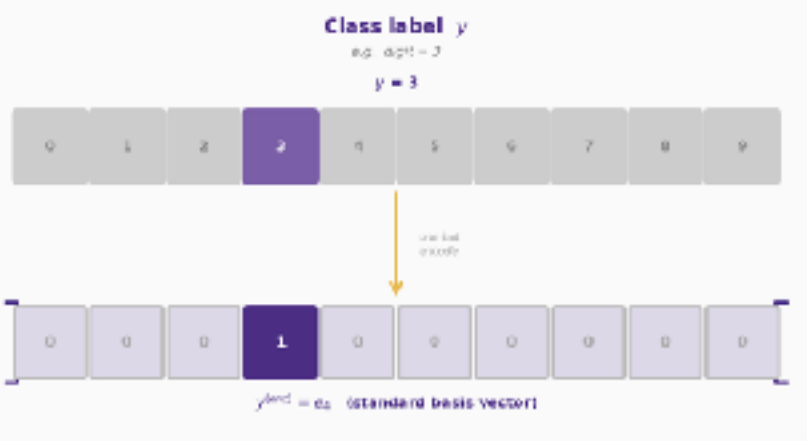
$$\mathbb{P}[c_1 \,|\, x]$$
$$\mathbb{P}[c_2 \,|\, x]$$
$$\vdots$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



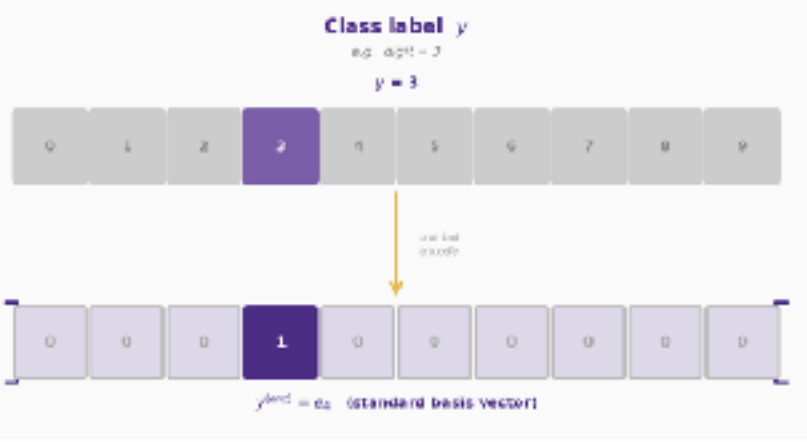
$$\mathbb{P}[c_1 \,|\, x]$$
$$\mathbb{P}[c_2 \,|\, x]$$
$$\vdots$$
$$\mathbb{P}[c_k \,|\, x]$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



$w_1 \cdot x$

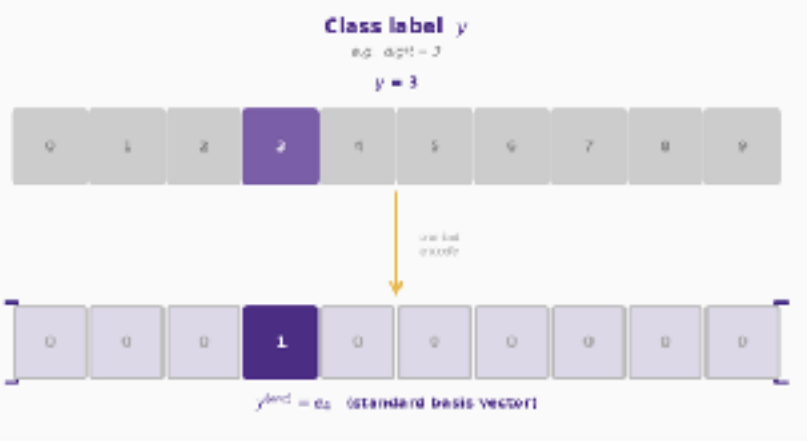
$\mathbb{P}[c_1 \,|\, x]$
$\mathbb{P}[c_2 \,|\, x]$
$\vdots$
$\mathbb{P}[c_k \,|\, x]$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



$w_1 \cdot x$ $\quad \mathbb{P}[c_1 \,|\, x]$

$w_2 \cdot x$ $\quad \mathbb{P}[c_2 \,|\, x]$

$\vdots$

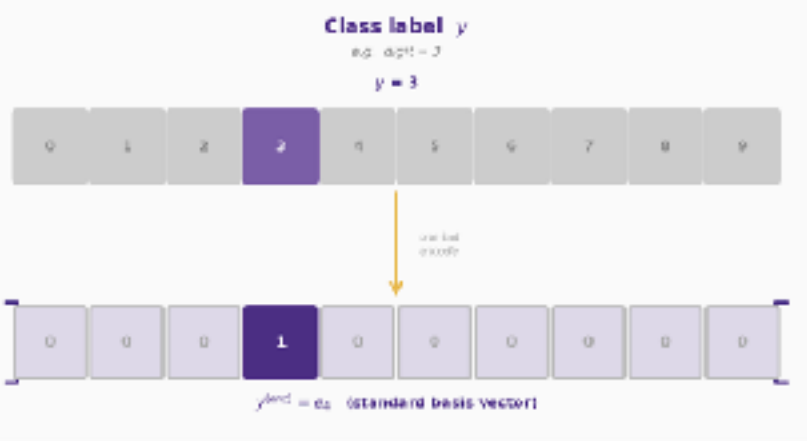
$\mathbb{P}[c_k \,|\, x]$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i
- Try to predict $\mathbb{P}[c_i \mid x]$ as a linear function $w_i \cdot x$ for each i



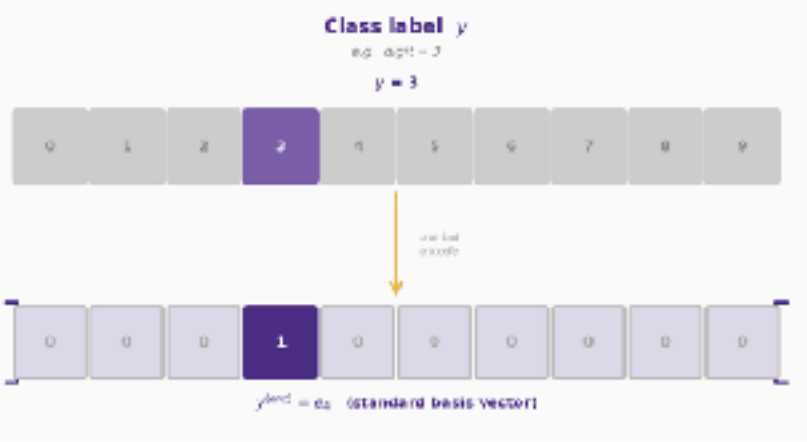
$$\begin{matrix} w_1 \cdot x \\ w_2 \cdot x \\ \vdots \end{matrix} \qquad \begin{matrix} \mathbb{P}[c_1 \mid x] \\ \mathbb{P}[c_2 \mid x] \\ \vdots \\ \mathbb{P}[c_k \mid x] \end{matrix}$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i
- Try to predict $\mathbb{P}[c_i \mid x]$ as a linear function $w_i \cdot x$ for each i



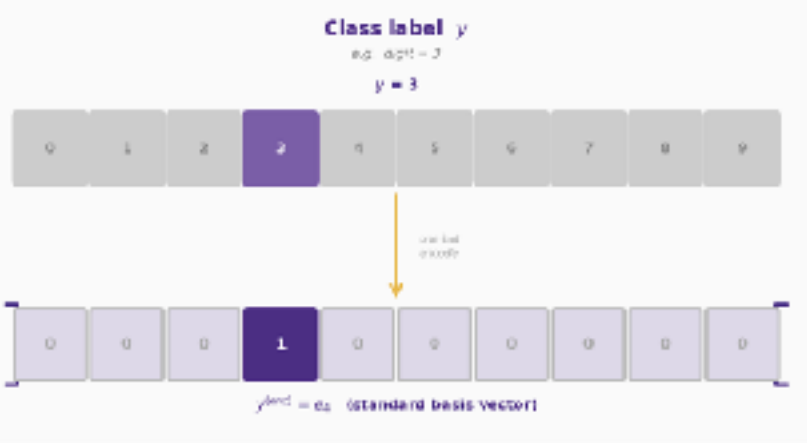
$$\begin{matrix} w_1 \cdot x \\ w_2 \cdot x \\ \vdots \\ w_k \cdot x \end{matrix} \qquad \begin{matrix} \mathbb{P}[c_1 \mid x] \\ \mathbb{P}[c_2 \mid x] \\ \vdots \\ \mathbb{P}[c_k \mid x] \end{matrix}$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



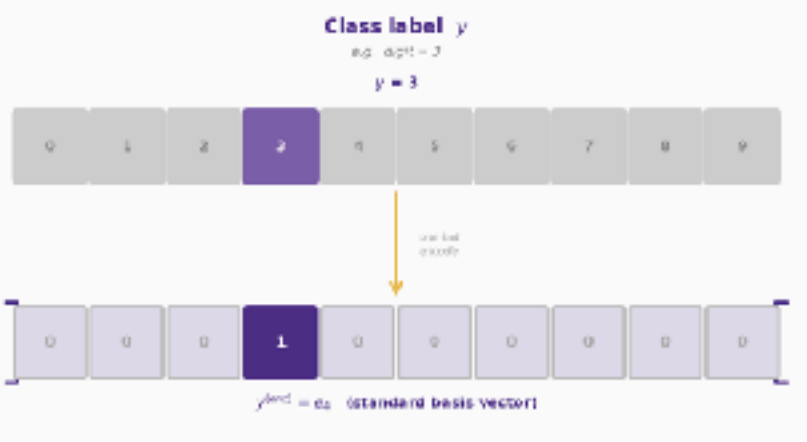
$$w_1 \cdot x \qquad \mathbb{P}[c_1 \,|\, x] \qquad e^{w_1 \cdot x}/N$$
$$w_2 \cdot x \qquad \mathbb{P}[c_2 \,|\, x]$$
$$\vdots \qquad\qquad \vdots$$
$$w_k \cdot x \qquad \mathbb{P}[c_k \,|\, x]$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \mid x]$ versus $\mathbb{P}[\text{not } c_i \mid x]$ for each i
- Try to predict $\mathbb{P}[c_i \mid x]$ as a linear function $w_i \cdot x$ for each i



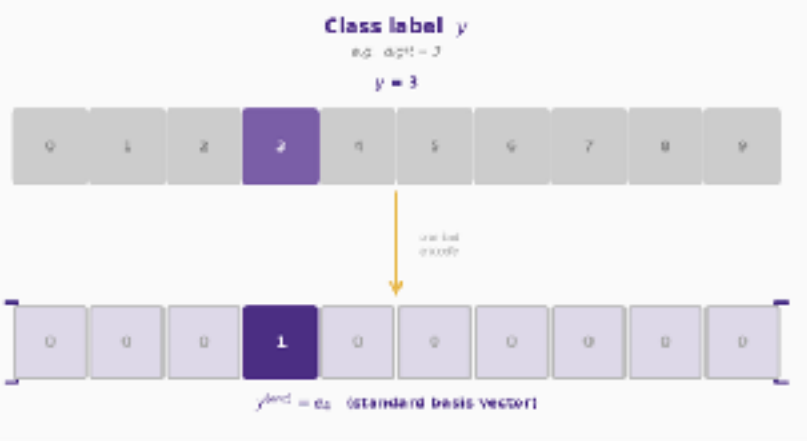
$$w_1 \cdot x \qquad \mathbb{P}[c_1 \mid x] \qquad e^{w_1 \cdot x}/N$$
$$w_2 \cdot x \qquad \mathbb{P}[c_2 \mid x] \qquad e^{w_2 \cdot x}/N$$
$$\vdots \qquad\qquad \vdots$$
$$w_k \cdot x \qquad \mathbb{P}[c_k \mid x]$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



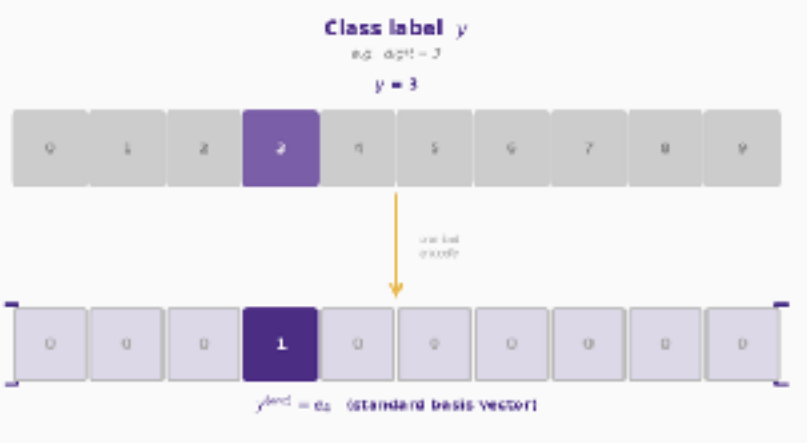
$$w_1 \cdot x \qquad \mathbb{P}[c_1 \,|\, x] \qquad e^{w_1 \cdot x}/N$$
$$w_2 \cdot x \qquad \mathbb{P}[c_2 \,|\, x] \qquad e^{w_2 \cdot x}/N$$
$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$
$$w_k \cdot x \qquad \mathbb{P}[c_k \,|\, x]$$

One-HOT encoding labels + softmax

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \ldots, c_k\}$
- c_j's are called classes or labels

Ideas for how to use logistic regression for multi class prediction:

- Try to predict $\mathbb{P}[c_i \,|\, x]$ versus $\mathbb{P}[\text{not } c_i \,|\, x]$ for each i
- Try to predict $\mathbb{P}[c_i \,|\, x]$ as a linear function $w_i \cdot x$ for each i



$$w_1 \cdot x \qquad \mathbb{P}[c_1 \,|\, x] \qquad e^{w_1 \cdot x}/N$$
$$w_2 \cdot x \qquad \mathbb{P}[c_2 \,|\, x] \qquad e^{w_2 \cdot x}/N$$
$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$
$$w_k \cdot x \qquad \mathbb{P}[c_k \,|\, x] \qquad e^{w_k \cdot x}/N$$

Softmax classification

Softmax classification

2 classes

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

k classes

$$P(y=c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

k classes

Softmax

$$P(y=c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

k classes

Softmax

$$P(y=c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

MLE

$$\text{argmax}_w \sum_{i=1}^{n} \log\left(\frac{1}{1+\mathrm{e}^{-y_i w^\top x_i}}\right)$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

k classes

Softmax

$$P(y=c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

MLE

$$\text{argmax}_w \sum_{i=1}^{n} \log\left(\frac{1}{1+\mathrm{e}^{-y_i w^\top x_i}}\right)$$

Logistic loss

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P(y=1 \mid x) = \frac{1}{1+e^{-w^\top x}} = \frac{e^{w^\top x}}{1+e^{w^\top x}}$$

$$P(y=-1 \mid x) = \frac{1}{1+e^{w^\top x}}$$

MLE

$$\operatorname{argmax}_w \sum_{i=1}^{n} \log\left(\frac{1}{1+\mathrm{e}^{-y_i w^\top x_i}}\right)$$

Logistic loss

k classes

Softmax

$$P(y=c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

$$\operatorname{argmax}_w \sum_{i=1}^{n} \sum_{j=1}^{k} 1\{y_i = c_j\} \log\left(\frac{\mathrm{e}^{w_j^\top x_i}}{\sum_{j'=1}^{k} e^{w_{j'}^\top x_i}}\right)$$

Softmax classification

2 classes

Sigmoid

Conditional probabilities

$$P\left(y=1\middle|x\right)=\frac{1}{1+e^{-w^{\top}x}}=\frac{e^{w^{\top}x}}{1+e^{w^{\top}x}}$$

$$P\left(y=-1\middle|x\right)=\frac{1}{1+e^{w^{\top}x}}$$

MLE

$$\operatorname{argmax}_{w}\sum_{i=1}^{n}\log\left(\frac{1}{1+\mathrm{e}^{-y_i w^{\top}x_i}}\right)$$

Logistic loss

k classes

Softmax

$$P\left(y=c_j\middle|x\right)=\frac{e^{w_j^{\top}x}}{\sum_{j'}e^{w_{j'}^{\top}x}}$$

$$\operatorname{argmax}_{w}\sum_{i=1}^{n}\sum_{j=1}^{k}1\left\{y_i=c_j\right\}\log\left(\frac{\mathrm{e}^{w_j^{\top}x_i}}{\sum_{j'=1}^{k}e^{w_{j'}^{\top}x_i}}\right)$$

Binary cross entropy

Softmax classification

without loss of generality, set $k = 2,\ w_1 = 0$

2 classes

k classes

Sigmoid

Conditional probabilities

Softmax

$$P(y = 1 \mid x) = \frac{1}{1 + e^{-w^\top x}} = \frac{e^{w^\top x}}{1 + e^{w^\top x}}$$

$$P(y = -1 \mid x) = \frac{1}{1 + e^{w^\top x}}$$

$$P(y = c_j \mid x) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

MLE

$$\text{argmax}_w \sum_{i=1}^{n} \log\left(\frac{1}{1 + \mathrm{e}^{-y_i w^\top x_i}}\right)$$

Logistic loss

$$\text{argmax}_w \sum_{i=1}^{n} \sum_{j=1}^{k} 1\{y_i = c_j\} \log\left(\frac{\mathrm{e}^{w_j^\top x_i}}{\sum_{j'=1}^{k} e^{w_{j'}^\top x_i}}\right)$$

Binary cross entropy

Regression and classification

Regression and classification

- ML paradigm: define prediction $f_w(x)$ and loss $\ell\left(f_w(x),\ y\right)$

Regression and classification

- ML paradigm: define prediction $f_w(x)$ and loss $\ell\left(f_w(x),\ y\right)$
- Then optimize:

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \ell\left(f_w(x_i),\ y_i\right)$$

Regression and classification

- ML paradigm: define prediction $f_w(x)$ and loss $\ell\left(f_w(x),\ y\right)$
- Then optimize:

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \ell\left(f_w(x_i),\ y_i\right)$$

- Squared error loss: $\ell\left(f_w(x),\ y\right) = \left(y - f_w(x)\right)^2$

Regression and classification

- ML paradigm: define prediction $f_w(x)$ and loss $\ell\left(f_w(x),\ y\right)$
- Then optimize:

$$\hat{w} = \text{argmin}_w \sum_{i=1}^{n} \ell\left(f_w(x_i),\ y_i\right)$$

- Squared error loss: $\ell\left(f_w(x),\ y\right) = \left(y - f_w(x)\right)^2$
- Logistic loss: $\ell\left(f_w(x),\ y\right) = \log(1 + \exp\left(-y f_w(x)\right))$

Regression and classification

- Can we treat classification as a regression problem?

- Can we treat regression as a classification problem?

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

- Can we treat regression as a classification problem?

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

Is this a good idea?

- Can we treat regression as a classification problem?

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

Is this a good idea?

No, no smooth relationship between label values

- Can we treat regression as a classification problem?

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

Is this a good idea?

No, no smooth relationship between label values

- Can we treat regression as a classification problem?

Yes: classify which "bin" the true number falls into: c_1 = [0-10], c_2 = [11-20], …

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

Is this a good idea?

No, no smooth relationship between label values

- Can we treat regression as a classification problem?

Yes: classify which "bin" the true number falls into: c_1 = [0-10], c_2 = [11-20], …

Is this a good idea?

Regression and classification

- Can we treat classification as a regression problem?

Yes: could regress to a continuous output y and round to a valid label

Is this a good idea?

No, no smooth relationship between label values

- Can we treat regression as a classification problem?

Yes: classify which "bin" the true number falls into: c_1 = [0-10], c_2 = [11-20], …

Is this a good idea?

You lose some precision, but it actually can be. Sometimes easier to optimize. Can win a kaggle competition this way. Look into "distributional RL"

The ML pipeline

1. Task
2. Data
3. Model family
4. Training loss
4. Optimize
5. Pick hyperparameters
6. Evaluate

Non-parametric methods Nearest Neighbours

Natasha Jaques

UNIVERSITY of WASHINGTON

Parametric vs non-parametric

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # Learning a bunch of parameters / weights. Like linear regression, neural networks

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # Learning a bunch of parameters / weights. Like linear regression, neural networks

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # Learning a bunch of parameters / weights. Like linear regression, neural networks

- A model is non-parametric if # parameters increases with # samples
  - Does not mean absence of parameters!

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # Learning a bunch of parameters / weights. Like linear regression, neural networks

- A model is non-parametric if # parameters increases with # samples
  - Does not mean absence of parameters!

  # Today's class

This lecture: k nearest neighbors

This lecture: k nearest neighbors

Simple

This lecture: k nearest neighbors

Simple

- Assume we have a classification task

This lecture: k nearest neighbors

Simple

- Assume we have a classification task
- To classify a new point x:
  - Find its k nearest neighbors in the training data
  - Set y to be the majority vote of the labels of these nearest neighbors

This lecture: k nearest neighbors

Simple

- Assume we have a classification task
- To classify a new point x:
  - Find its k nearest neighbors in the training data
  - Set y to be the majority vote of the labels of these nearest neighbors

What do we mean by nearest neighbors?

This lecture: k nearest neighbors

Simple

- Assume we have a classification task
- To classify a new point x:
  - Find its k nearest neighbors in the training data
  - Set y to be the majority vote of the labels of these nearest neighbors

What do we mean by nearest neighbors?
Smallest distance in feature space (with d features)

This lecture: k nearest neighbors

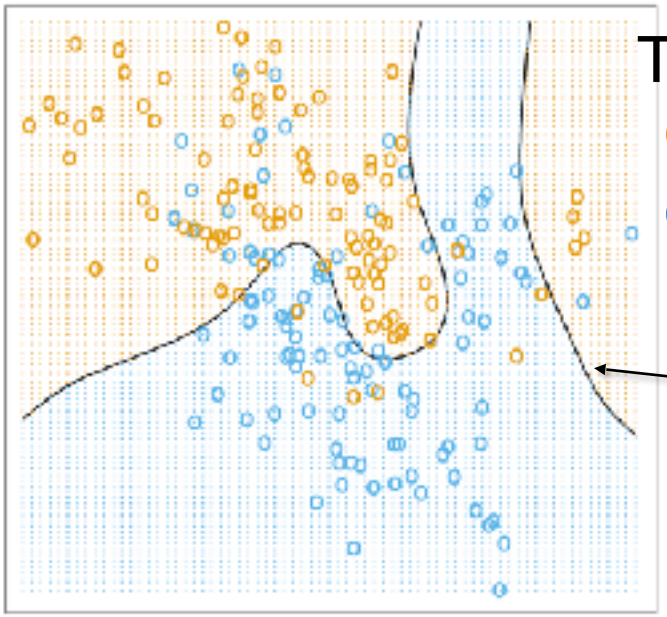
Simple

- Assume we have a classification task
- To classify a new point x:
  - Find its k nearest neighbors in the training data
  - Set y to be the majority vote of the labels of these nearest neighbors

What do we mean by nearest neighbors?
Smallest distance in feature space (with d features)

- Design choices / hyperparameters:
  - Number of nearest neighbors k
  - Distance metric
  - Aggregation method

Example: Bayes classifier



Training data:

- True label: +1
- True label: -1

Optimal Bayes classifier:

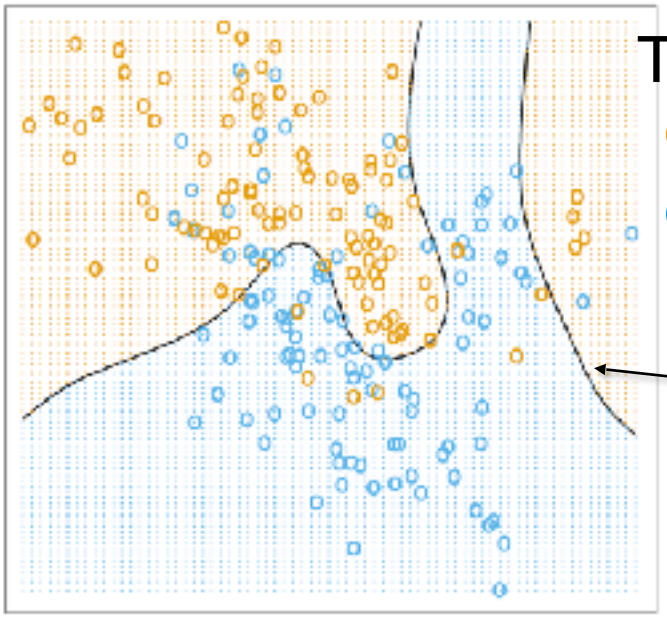
$$\mathbb{P}(Y = 1 | X = x) = \frac{1}{2}$$

Predicted label: +1

Predicted label: -1

Figures from Hastie et al.

Example: Bayes classifier



Training data:

- True label: +1
- True label: -1

Optimal Bayes classifier:

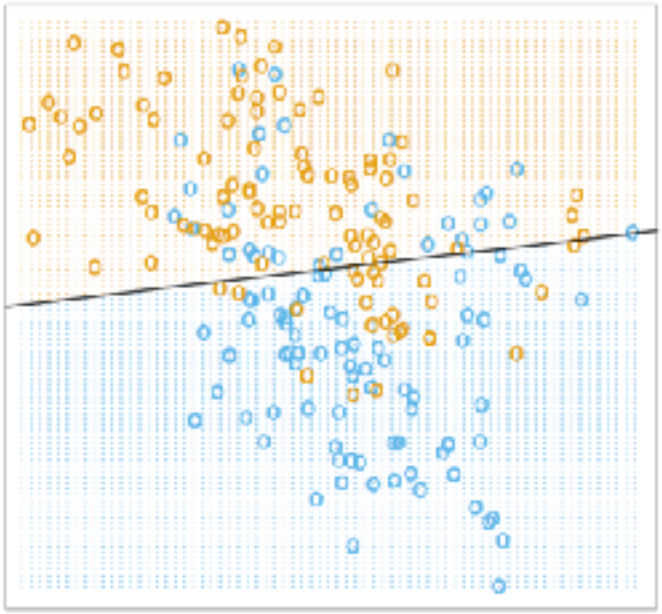
$$\mathbb{P}(Y = 1 | X = x) = \frac{1}{2}$$ # Ground truth

Predicted label: +1

Predicted label: -1

Figures from Hastie et al.

Linear decision boundary



Training data:

- True label: +1
- True label: -1

Learned linear decision bound

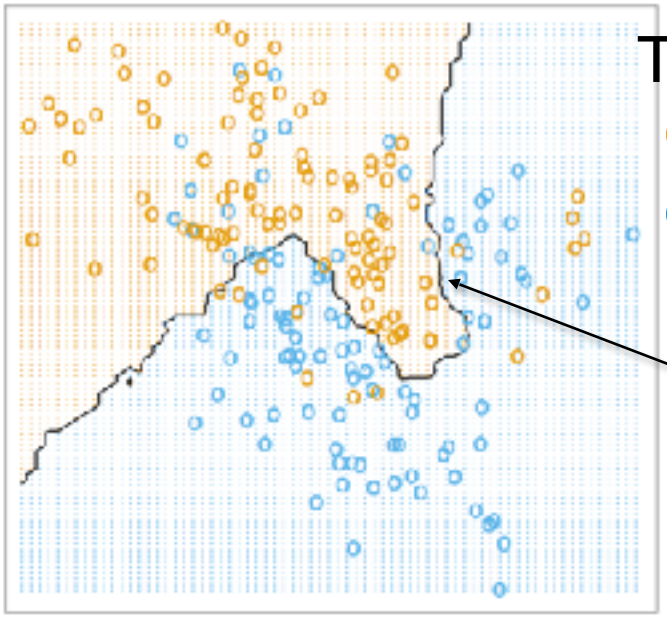
$$x^T w + b = 0$$

Predicted label: +1

Predicted label: -1

Figures from Hastie et al.

$k = 15$ nearest neighbors boundary



Training data:

- True label: +1
- True label: -1

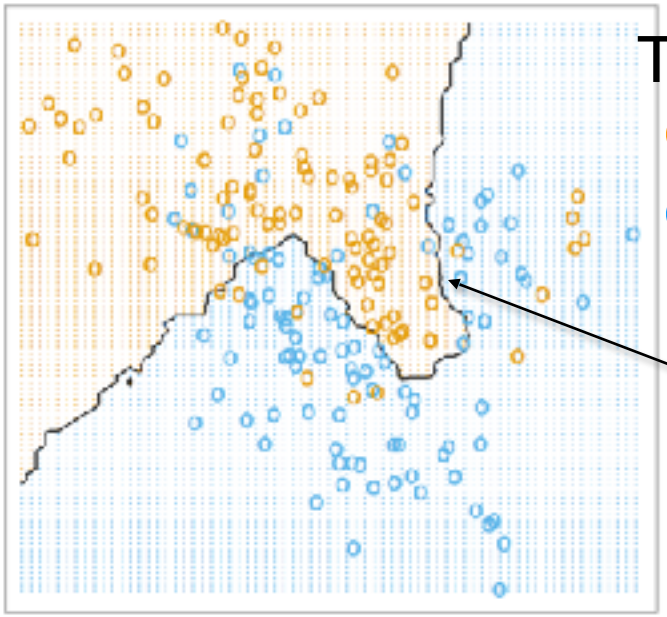
15 nearest neighbors decision boundary (majority vote)

Predicted label: +1

Predicted label: -1

Figures from Hastie et al.

$k = 15$ nearest neighbors boundary



Training data:

- True label: +1
- True label: -1

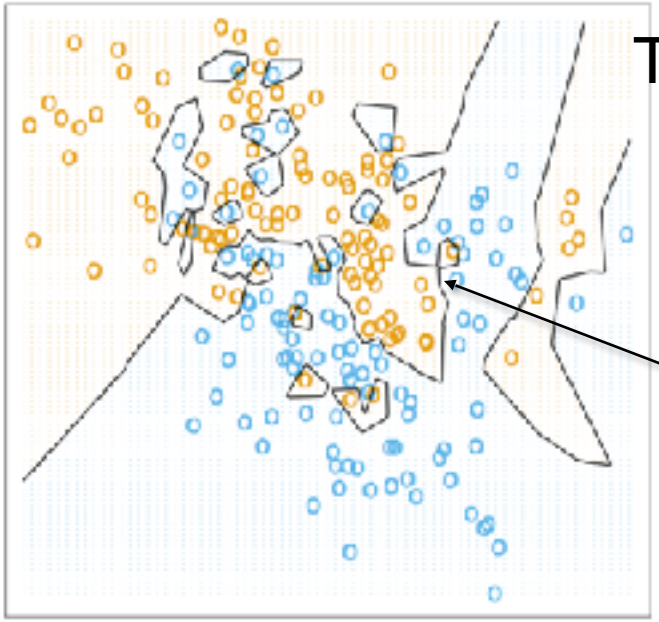
15 nearest neighbors decision boundary (majority vote)

- Predicted label: +1
- Predicted label: -1

Boundary depends on the training data points

Figures from Hastie et al.

$k = 1$ nearest neighbor boundary



Training data:

- True label: +1
- True label: -1

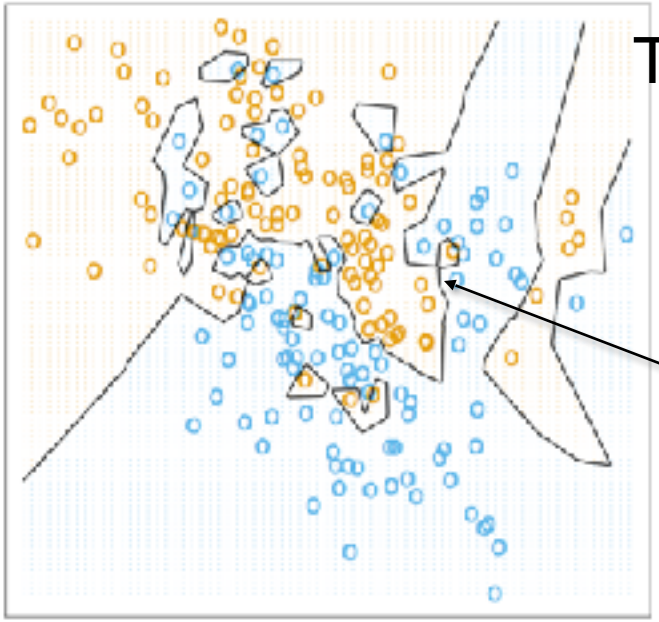
1 nearest neighbor decision boundary (majority vote)

Predicted label: -1

Predicted label: +1

Figures from Hastie et al.

$k = 1$ nearest neighbor boundary



Overfitting

Training data:

- True label: +1
- True label: -1

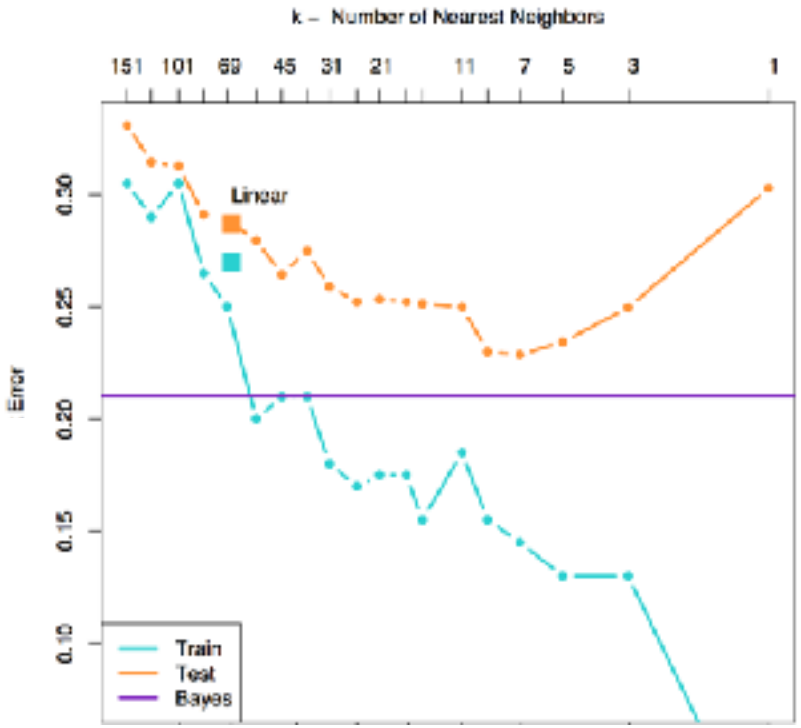
1 nearest neighbor decision boundary (majority vote)

Predicted label: -1

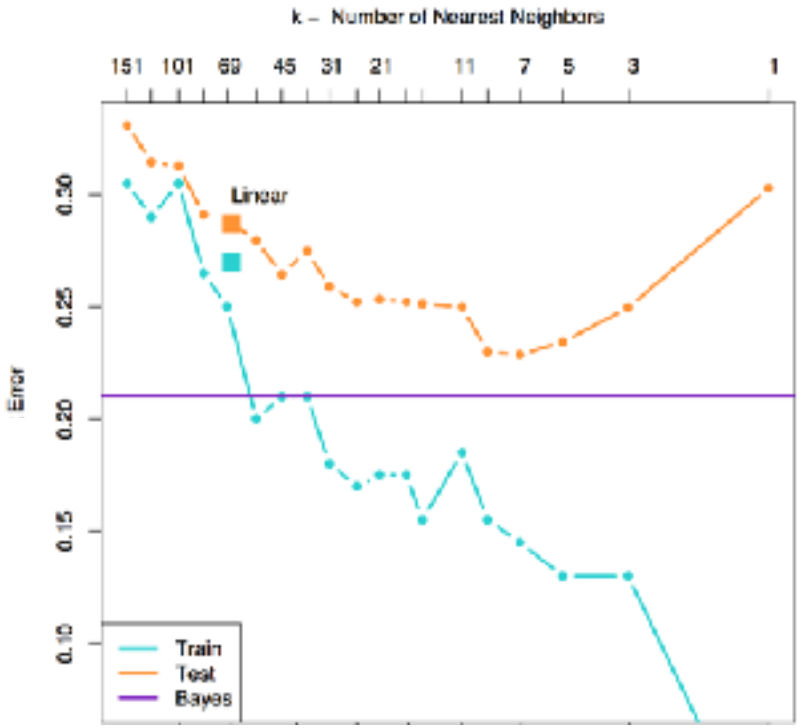
Predicted label: +1

Figures from Hastie et al.

k nearest neighbors error

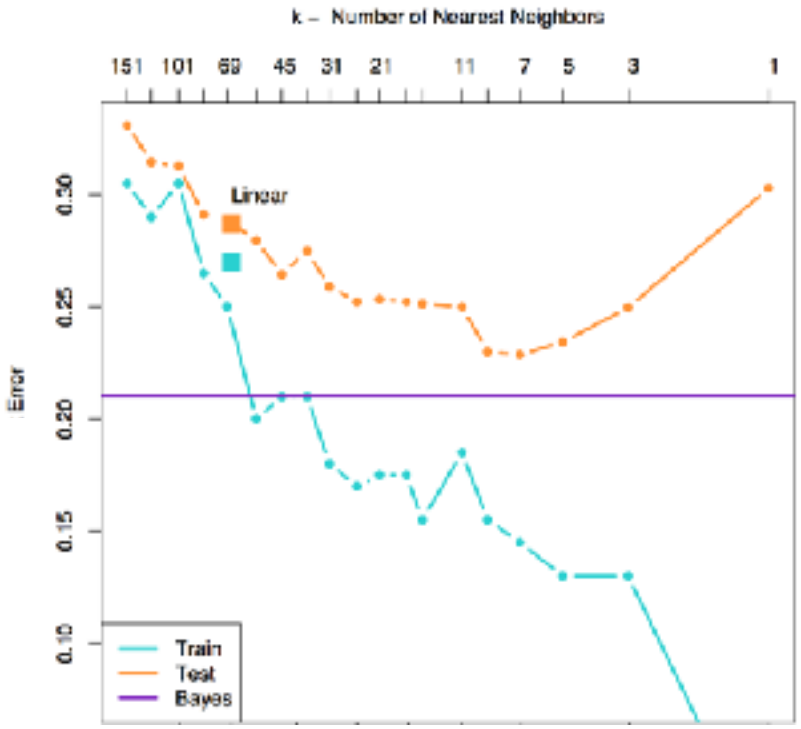


k nearest neighbors error



Overfitting

k nearest neighbors error



Underfitting

Overfitting

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

- A model is non-parametric if # parameters increases with # samples

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # After training, you can discard your data

- A model is non-parametric if # parameters increases with # samples

Parametric vs non-parametric

- A model is parametric if # parameters does not depend on # samples

  # After training, you can discard your data

- A model is non-parametric if # parameters increases with # samples

  # Keep training data around for inference

Notable distance metrics & level sets

Notable distance metrics & level sets

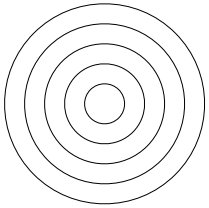
ℓ_2 norm (Euclidean)

Notable distance metrics & level sets

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$

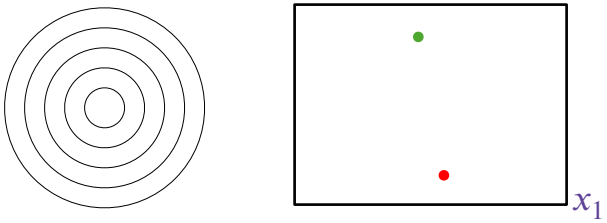
Notable distance metrics & level sets

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



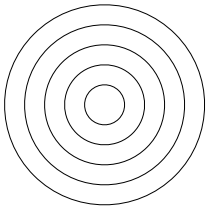
Notable distance metrics & level sets

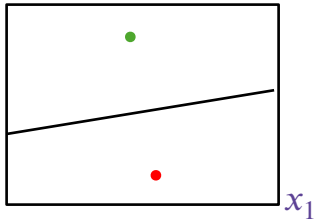
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



Notable distance metrics & level sets

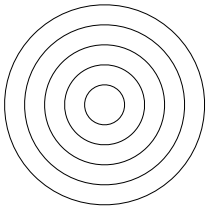
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$

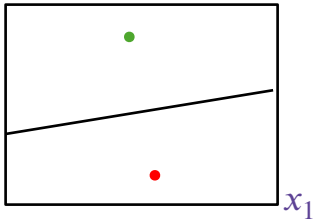




Notable distance metrics & level sets

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



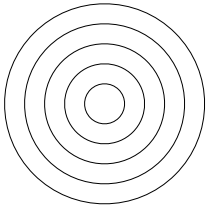


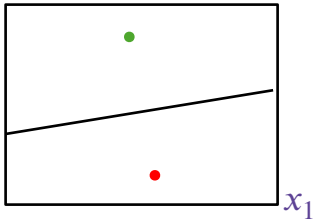
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$

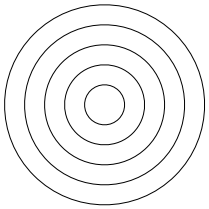
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

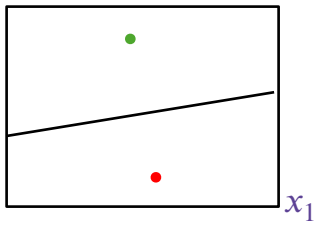




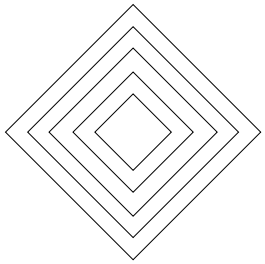
Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



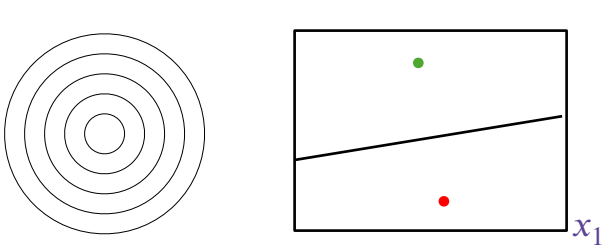


ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

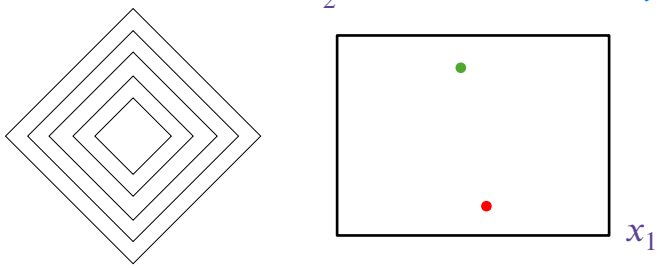


Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$

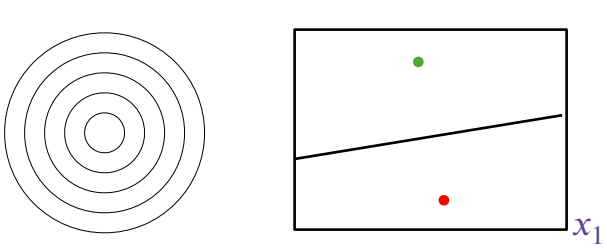


ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

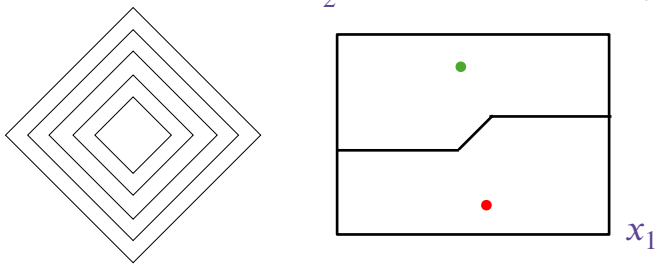


Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$

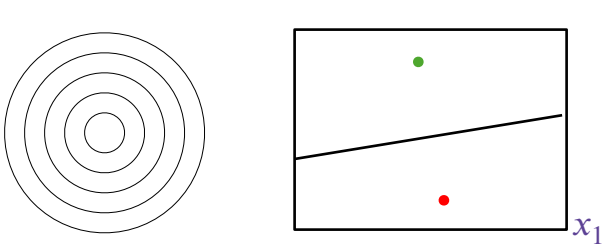


ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

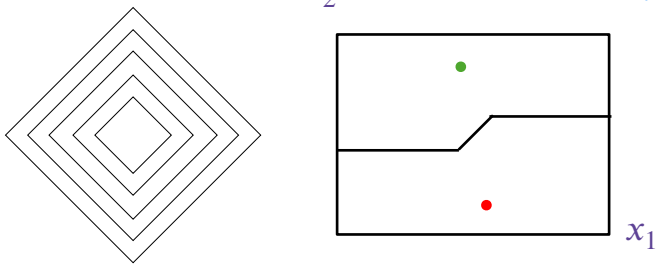


Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



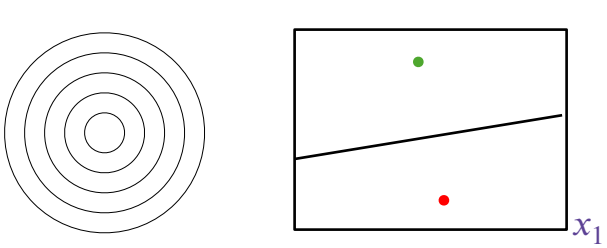
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



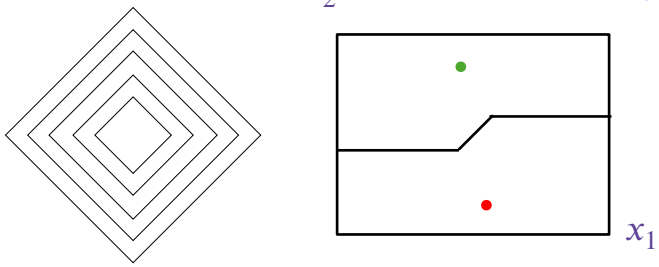
Mahalanobis norm

Notable distance metrics & level s

ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



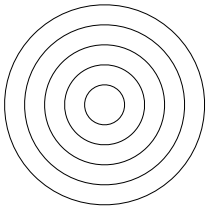
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$

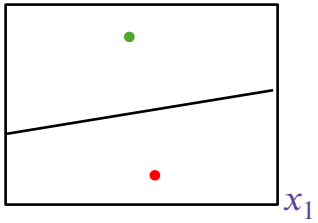


Mahalanobis norm $(u - v)^T M(u - v)$

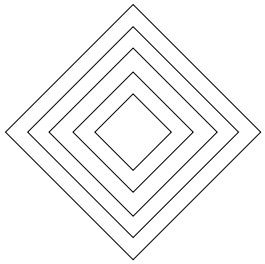
Notable distance metrics & level s

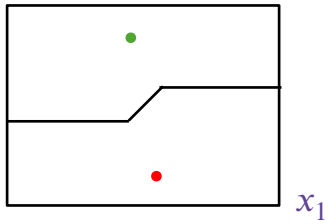
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



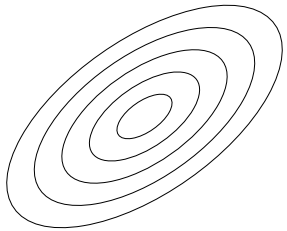


ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



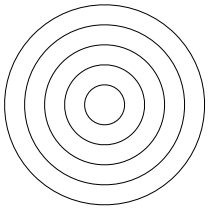


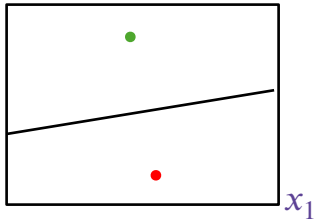
Mahalanobis norm $(u - v)^T M(u - v)$



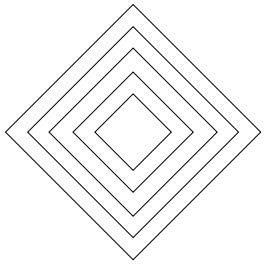
Notable distance metrics & level s

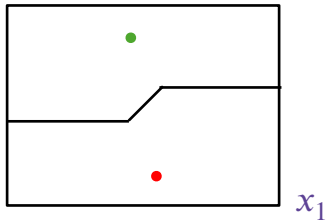
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



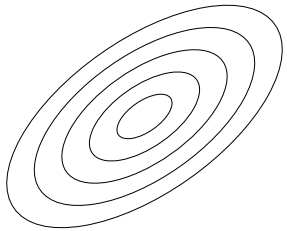


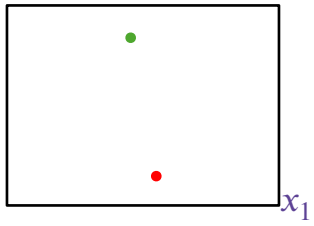
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$





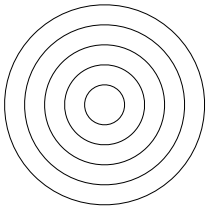
Mahalanobis norm $(u - v)^T M(u - v)$

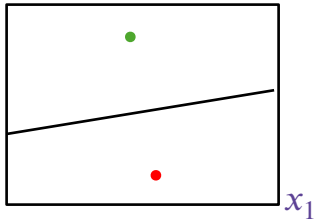




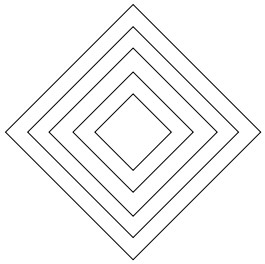
Notable distance metrics & level s

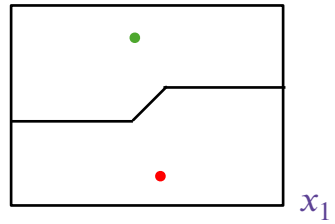
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



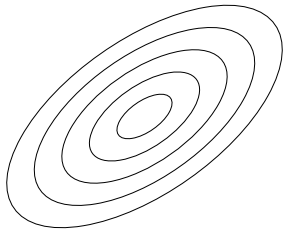


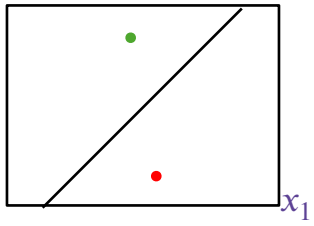
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$





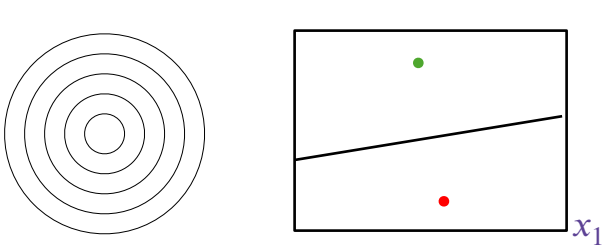
Mahalanobis norm $(u - v)^T M(u - v)$



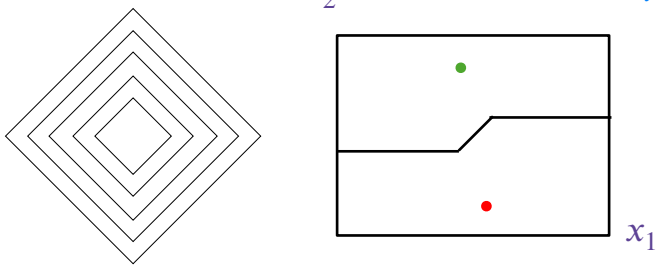


Notable distance metrics & level s

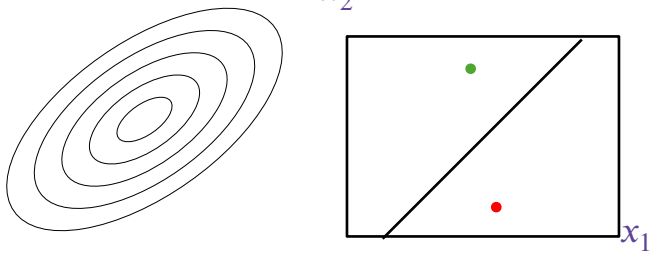
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



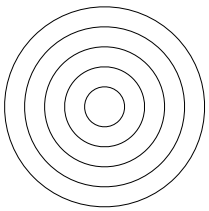
Mahalanobis norm $(u - v)^T M(u - v)$

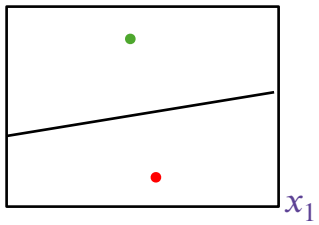


Weight different dimensions differently

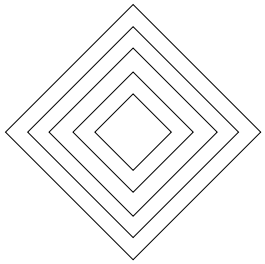
Notable distance metrics & level s

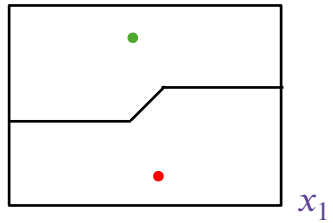
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



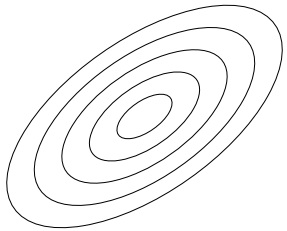


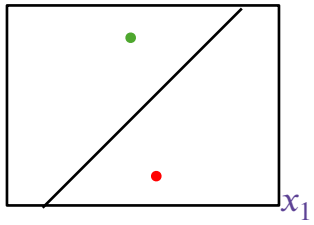
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$





Mahalanobis norm $(u - v)^T M(u - v)$



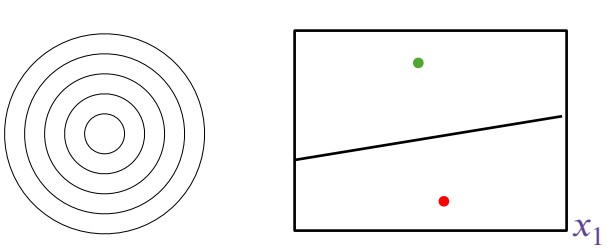


ℓ_∞ norm (max)

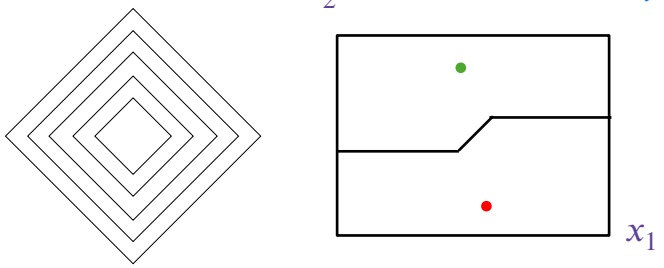
Weight different dimensions differently

Notable distance metrics & level s

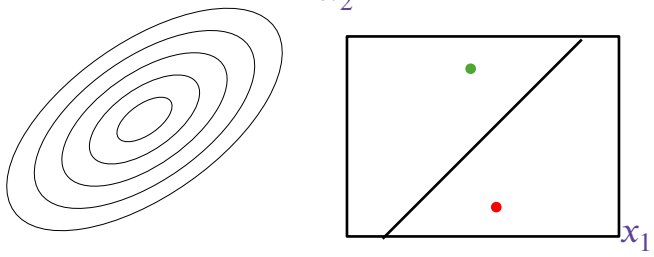
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



Mahalanobis norm $(u - v)^T M(u - v)$

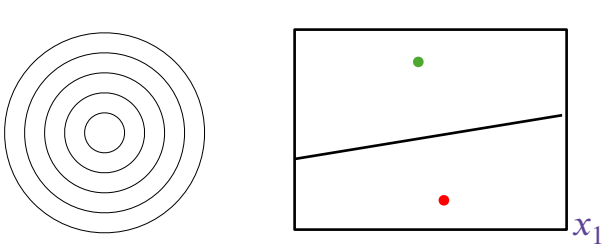


ℓ_∞ norm (max) $\|u - v\|_\infty = \max_i |u_i - v_i|$

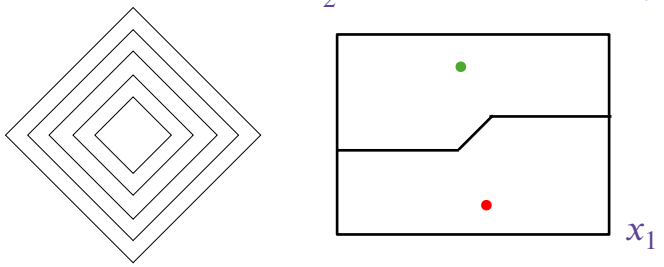
Weight different dimensions differently

Notable distance metrics & level s

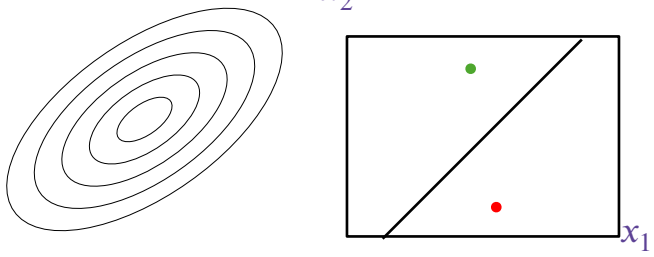
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



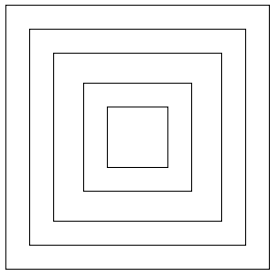
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



Mahalanobis norm $(u - v)^T M(u - v)$



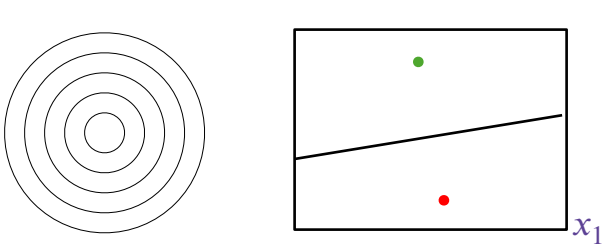
ℓ_∞ norm (max) $\|u - v\|_\infty = \max_i |u_i - v_i|$



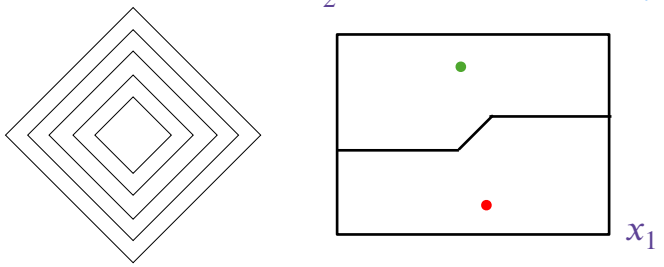
Weight different dimensions differently

Notable distance metrics & level s

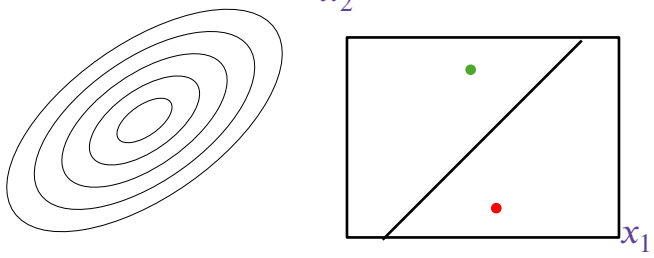
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



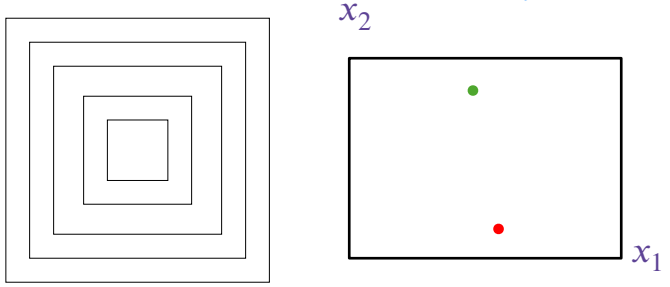
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



Mahalanobis norm $(u - v)^T M(u - v)$



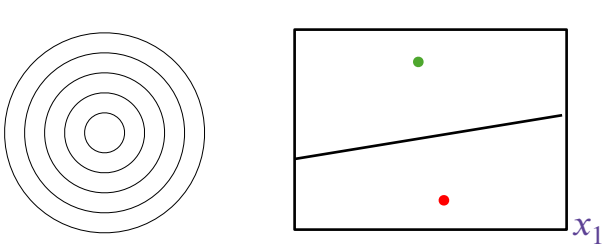
ℓ_∞ norm (max) $\|u - v\|_\infty = \max_i |u_i - v_i|$



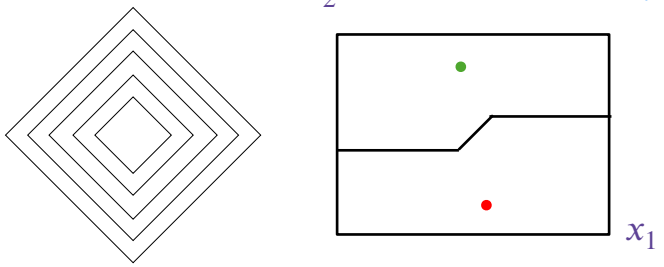
Weight different dimensions differently

Notable distance metrics & level s

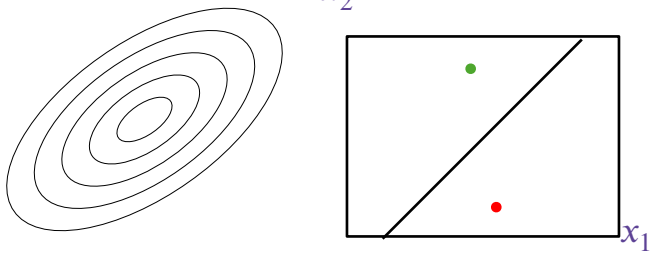
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



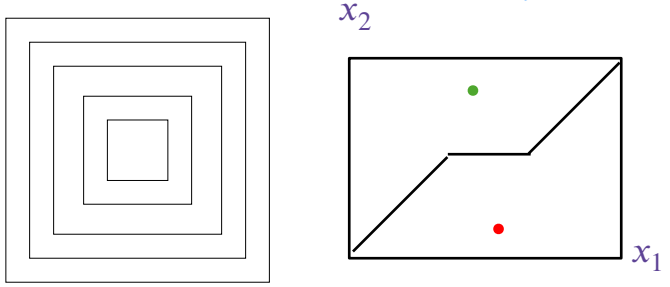
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



Mahalanobis norm $(u - v)^T M(u - v)$



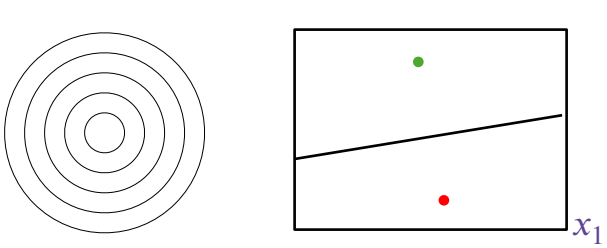
ℓ_∞ norm (max) $\|u - v\|_\infty = \max_i |u_i - v_i|$



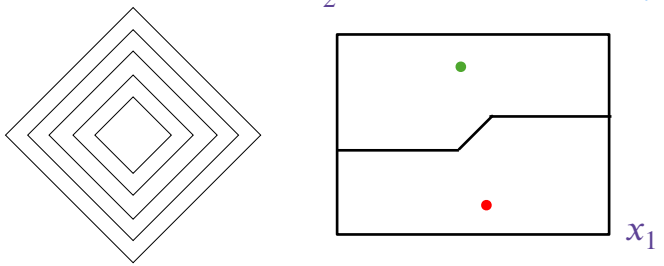
Weight different dimensions differently

Notable distance metrics & level s

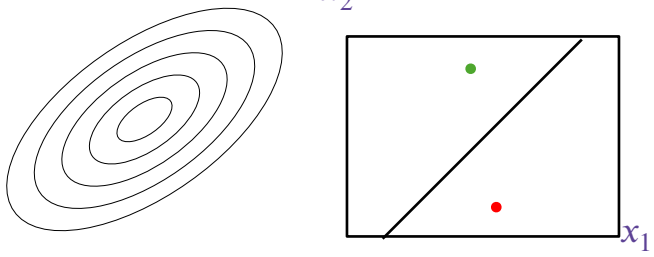
ℓ_2 norm (Euclidean) $d(u, v) = \|u - v\|_2^2$



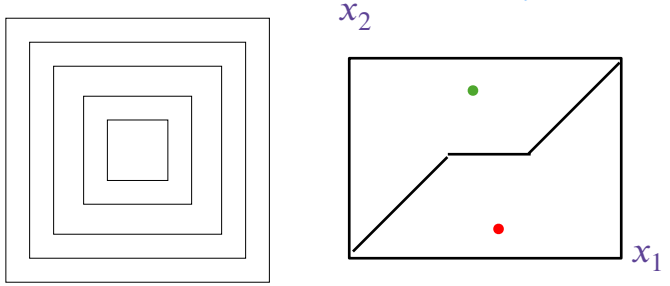
ℓ_1 norm (Manhattan, taxicab) $\|u - v\|_1 = \sum_{i=1}^{d} |u_i - v_i|$



Mahalanobis norm $(u - v)^T M(u - v)$



ℓ_∞ norm (max) $\|u - v\|_\infty = \max_i |u_i - v_i|$

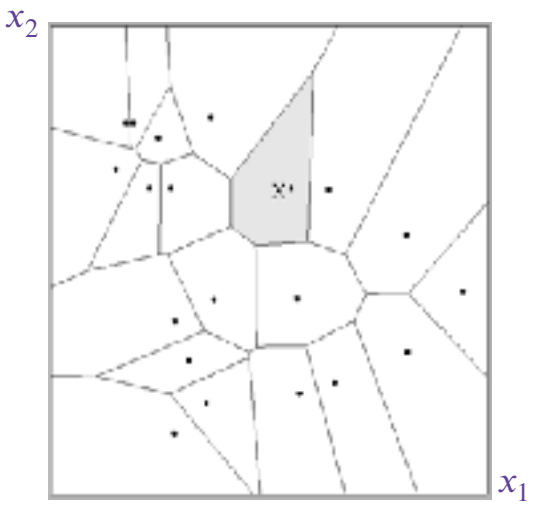


Weight different dimensions differently

Max distance between vectors

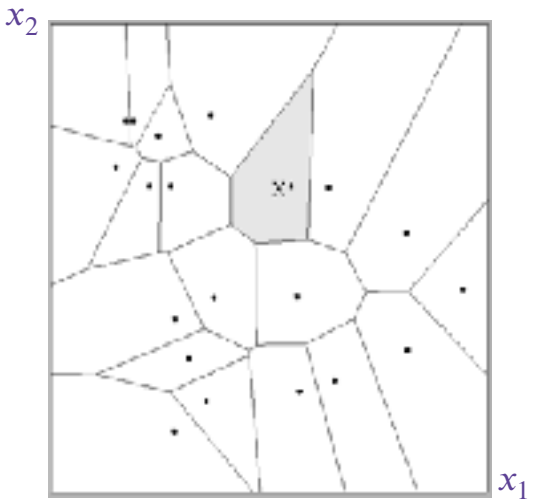
Example: distance metrics with $k = 1$ NN

$$d(x, x') = (x_1 - x_1')^2 + (x_2 - x_2')^2$$

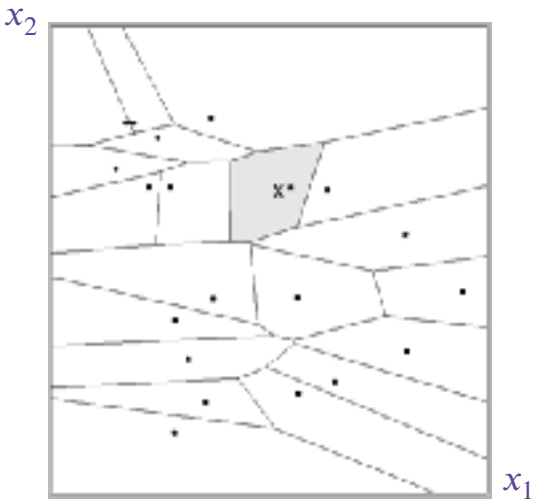


Example: distance metrics with $k = 1$ NN

$$d(x, x') = (x_1 - x_1')^2 + (x_2 - x_2')^2$$



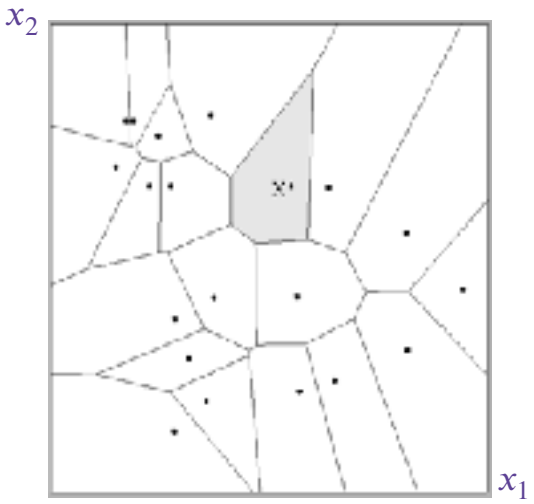
$$d(x, x') = (x_1 - x_1')^2 + 9(x_2 - x_2')^2$$



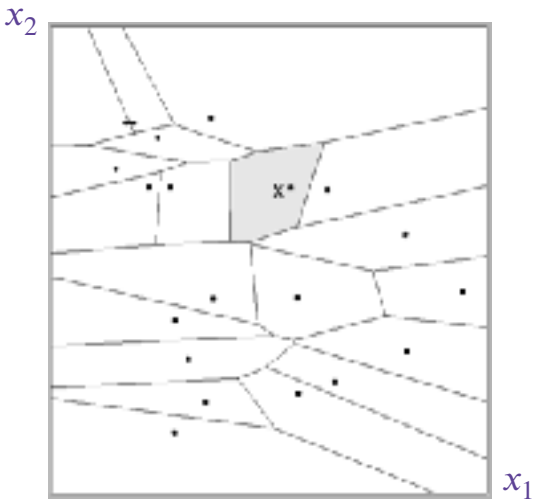
Example: distance metrics with $k = 1$ NN

L2

$$d(x, x') = (x_1 - x_1')^2 + (x_2 - x_2')^2$$



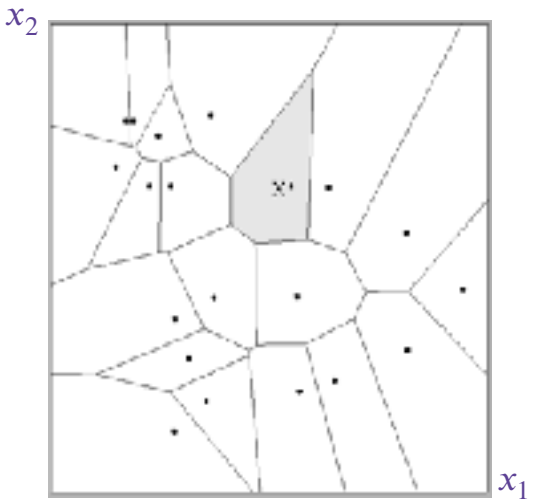
$$d(x, x') = (x_1 - x_1')^2 + 9(x_2 - x_2')^2$$



Example: distance metrics with $k = 1$ NN

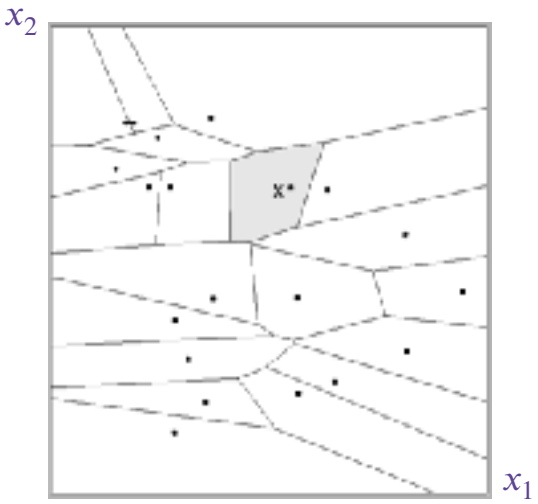
L2

$$d(x, x') = (x_1 - x'_1)^2 + (x_2 - x'_2)^2$$



Mahalanobis

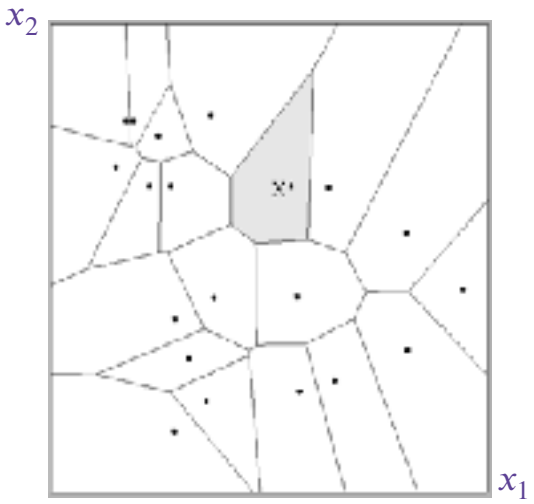
$$d(x, x') = (x_1 - x'_1)^2 + 9(x_2 - x'_2)^2$$



Example: distance metrics with $k = 1$ NN

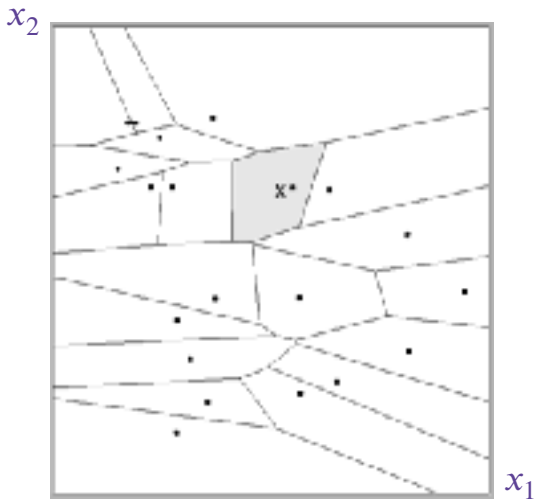
L2

$$d(x, x') = (x_1 - x_1')^2 + (x_2 - x_2')^2$$



Mahalanobis

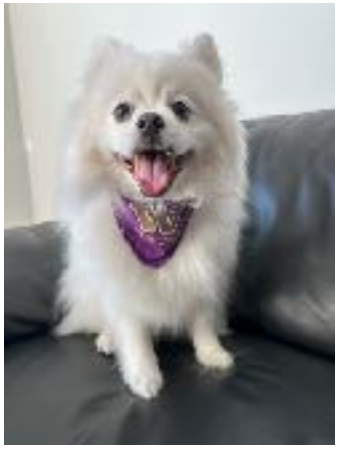
$$d(x, x') = (x_1 - x_1')^2 + 9(x_2 - x_2')^2$$



Know the geometry of your feature space

Learned distance metrics

Training data



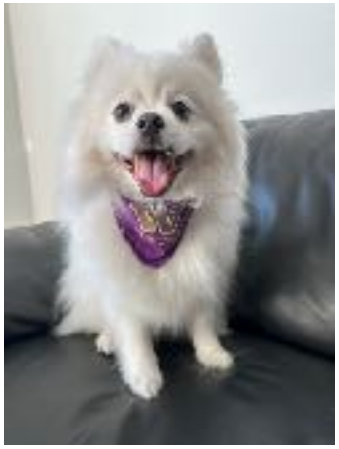
Dog

Cat

Test data

Learned distance metrics

Training data



Dog

Cat

Test data

Use a neural net to learn the distance function

kNN demo

- http://vision.stanford.edu/teaching/cs231n-demos/knn/

1-NN classification: Theoretical guarantees

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

1-NN classification: Theoretical guarantees

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P$$

Assume: we have enough data and **our true function is smooth**

1-NN classification: Theoretical guarantees

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d$$

Assume: we have enough data and **our true function is smooth**

1-NN classification: Theoretical guarantees

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$$

Assume: we have enough data and **our true function is smooth**

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{\text{NN}} \neq y$

Case 1: $y_{\text{NN}} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{\text{NN}}) P(y = 0 \,|\, x)$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{\text{NN}} \neq y$

Case 1: $y_{\text{NN}} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{\text{NN}}) P(y = 0 \,|\, x)$

Case 2: $y_{\text{NN}} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{\text{NN}}) P(y = 1 \,|\, x)$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{\text{NN}} \neq y$

Case 1: $y_{\text{NN}} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{\text{NN}}) P(y = 0 \,|\, x)$

Case 2: $y_{\text{NN}} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{\text{NN}}) P(y = 1 \,|\, x)$

As $n \to \infty$,

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{\text{NN}} \neq y$

Case 1: $y_{\text{NN}} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{\text{NN}}) P(y = 0 \,|\, x)$

Case 2: $y_{\text{NN}} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{\text{NN}}) P(y = 1 \,|\, x)$

As $n \to \infty$,

$P(y \,|\, x_{\text{NN}}) \to P(y \,|\, x)$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{\text{NN}} \neq y$

Case 1: $y_{\text{NN}} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{\text{NN}}) P(y = 0 \,|\, x)$

Case 2: $y_{\text{NN}} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{\text{NN}}) P(y = 1 \,|\, x)$

As $n \to \infty$,

$P(y \,|\, x_{\text{NN}}) \to P(y \,|\, x)$ # Why?

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

Case 1: $y_{NN} = 1, y = 0$ w.p. $P(y = 1 \mid x_{NN})P(y = 0 \mid x)$

Case 2: $y_{NN} = 0, y = 1$ w.p. $P(y = 0 \mid x_{NN})P(y = 1 \mid x)$

As $n \to \infty$,

$P(y \mid x_{NN}) \to P(y \mid x)$ # Why?

By assumptions, distance between x and $x_{NN} \to 0$ as $n \to \infty$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

Case 1: $y_{NN} = 1, y = 0$ w.p. $P(y = 1 \mid x_{NN})P(y = 0 \mid x)$

Case 2: $y_{NN} = 0, y = 1$ w.p. $P(y = 0 \mid x_{NN})P(y = 1 \mid x)$

As $n \to \infty$,

$P(y \mid x_{NN}) \to P(y \mid x)$ # Why?

Error: $2P(y = 1 \mid x)P(y = 0 \mid x)$

By assumptions, distance between x and $x_{NN} \to 0$ as $n \to \infty$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

Case 1: $y_{NN} = 1, y = 0$ w.p. $P(y = 1 \mid x_{NN})P(y = 0 \mid x)$

Case 2: $y_{NN} = 0, y = 1$ w.p. $P(y = 0 \mid x_{NN})P(y = 1 \mid x)$

As $n \to \infty$,

$P(y \mid x_{NN}) \to P(y \mid x)$ # Why?

Error: $2P(y = 1 \mid x)P(y = 0 \mid x)$

By assumptions, distance between x and $x_{NN} \to 0$ as $n \to \infty$

Define Bayes error:

$p^* = \min\{P(y = 1 \mid x), P(y = 0 \mid x)\}$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

Case 1: $y_{NN} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{NN})P(y = 0 \,|\, x)$

Case 2: $y_{NN} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{NN})P(y = 1 \,|\, x)$

As $n \to \infty$,

$P(y \,|\, x_{NN}) \to P(y \,|\, x)$ # Why?

By assumptions, distance between x and $x_{NN} \to 0$ as $n \to \infty$

Error: $2P(y = 1 \,|\, x)P(y = 0 \,|\, x)$

$= 2p^*(1 - p^*)$

Define Bayes error:

$p^* = \min\{P(y = 1 \,|\, x), P(y = 0 \,|\, x)\}$

1-NN classification: Theoretical guarantees

Assume: we have enough data and **our true function is smooth**

$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n} \sim P \qquad x^{(i)} \in \mathbb{R}^d \qquad y \in \{0, 1\}$

Given test point x, let x_{NN} be the nearest neighbour in D

Error if $y_{NN} \neq y$

Case 1: $y_{NN} = 1, y = 0$ w.p. $P(y = 1 \,|\, x_{NN})P(y = 0 \,|\, x)$

Case 2: $y_{NN} = 0, y = 1$ w.p. $P(y = 0 \,|\, x_{NN})P(y = 1 \,|\, x)$

As $n \to \infty$,

$P(y \,|\, x_{NN}) \to P(y \,|\, x)$ # Why?

By assumptions, distance between x and $x_{NN} \to 0$ as $n \to \infty$

Error: $2P(y = 1 \,|\, x)P(y = 0 \,|\, x)$

$= 2p^*(1 - p^*) \quad \leq 2p^*$

Define Bayes error:

$p^* = \min\{P(y = 1 \,|\, x), P(y = 0 \,|\, x)\}$

1-NN classification: Theoretical guarantees

As $n \to \infty$, Error $= 2p^*(1 - p^*) \leq 2p^*$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

1-NN classification: Theoretical guarantees

As $n \to \infty$, Error $= 2p^*(1 - p^*) \leq 2p^*$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

Bayes error is the best you can do given measurement error, so with infinite data nearest neighbors is a really good thing to do!

1-NN classification: Theoretical guarantees

As $n \to \infty$, Error $= 2p^*(1 - p^*) \leq 2p^*$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

Bayes error is the best you can do given measurement error, so with infinite data nearest neighbors is a really good thing to do

Can fit arbitrarily complicated functions

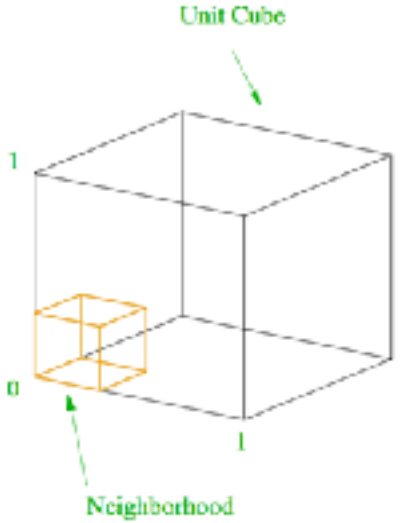
1-NN classification: Theoretical guarantees

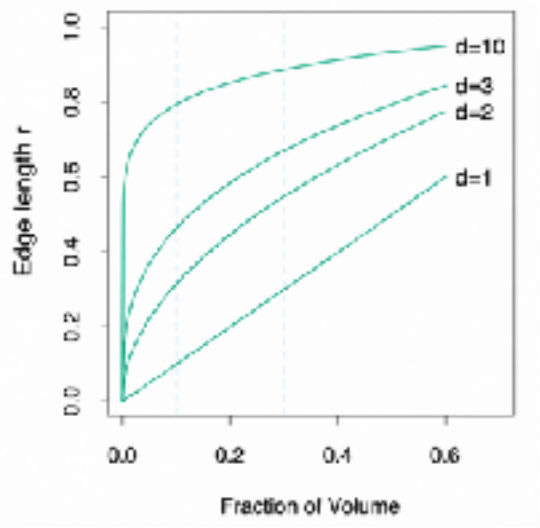
As $n \to \infty$, Error $= 2p^*(1-p^*) \leq 2p^*$

Theorem[Cover, Hart, 1967] If P_X is supported everywhere in $\mathbb{R}^d$ and $P(Y = 1|X = x)$ is smooth everywhere, then as $n \to \infty$ the 1-NN classification rule has error at most twice the Bayes error rate.

Bayes error is the best you can do given measurement error, so with infinite data nearest neighbors ng to
Can fit arbitrarily complicated functio

Curse of dimensionality, example 1





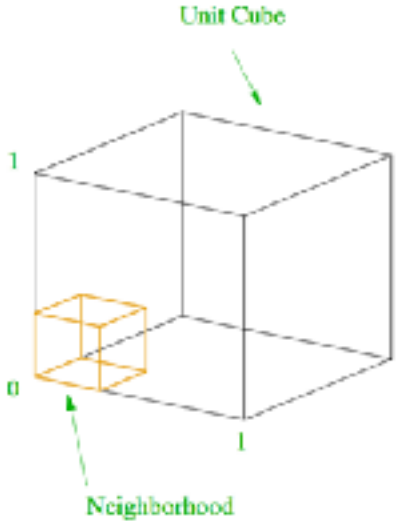
side length r

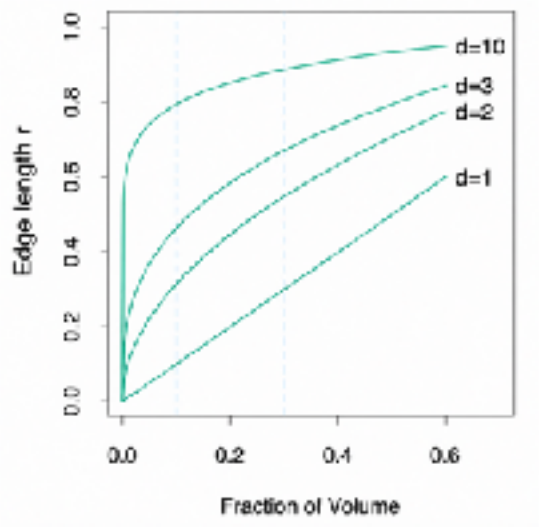
X is uniformly distributed over [0,1]^d. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of s length r?

Curse of dimensionality, example 1

If d = 1



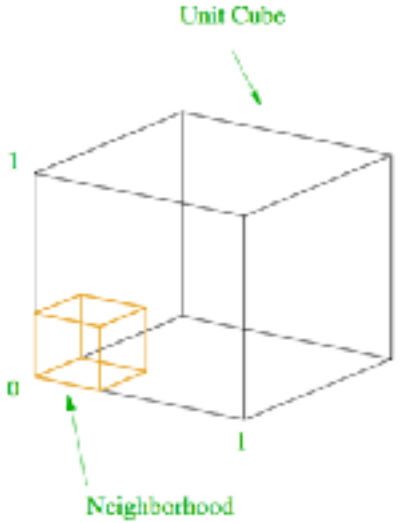


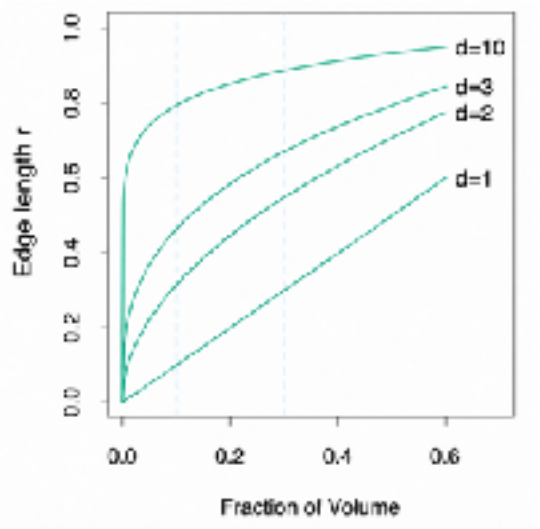
side length r

X is uniformly distributed over [0,1]. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of s length r?

Curse of dimensionality, example 1





side length r

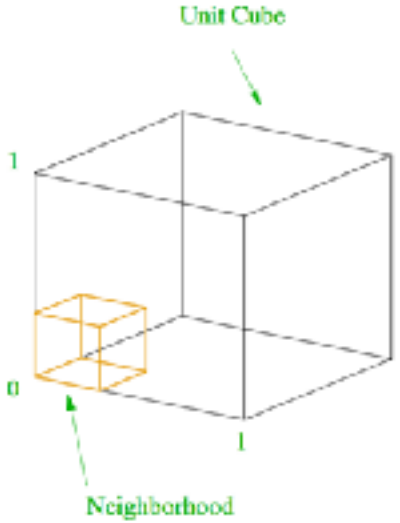
If d = 1

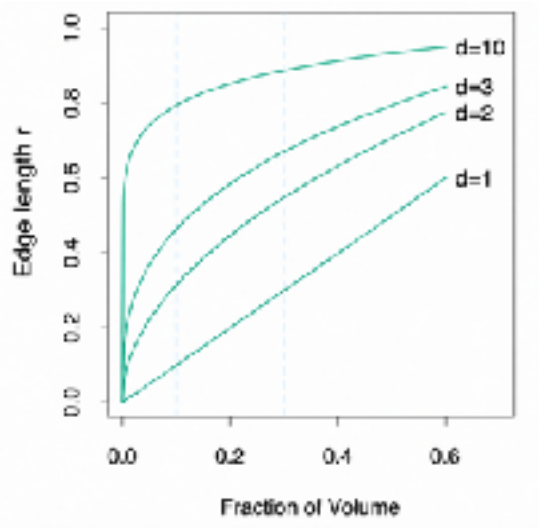
X is uniformly distributed over [0,1]^d. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of s length r?

Curse of dimensionality, example 1

If d = 1 $P = 0.3$



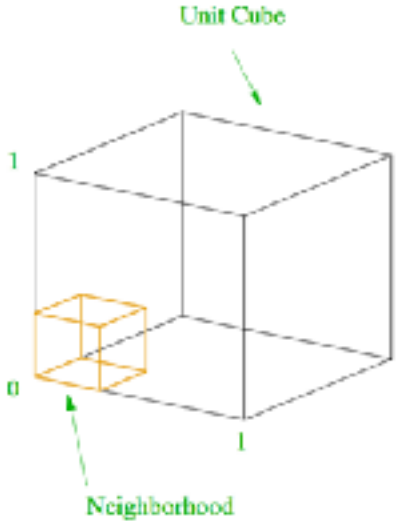


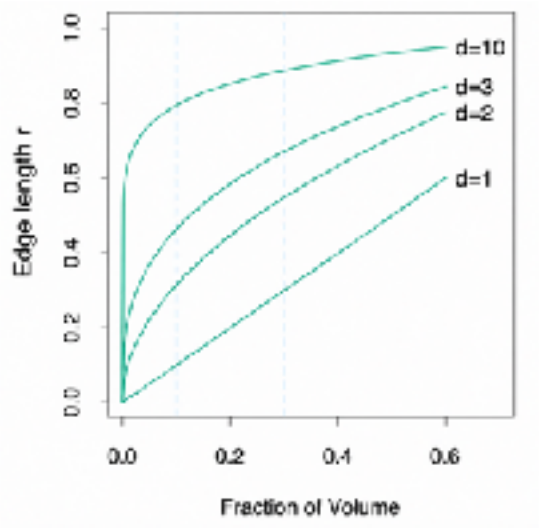
side length r

X is uniformly distributed over [0,1]^d. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of s length r?

Curse of dimensionality, example 1





side length r

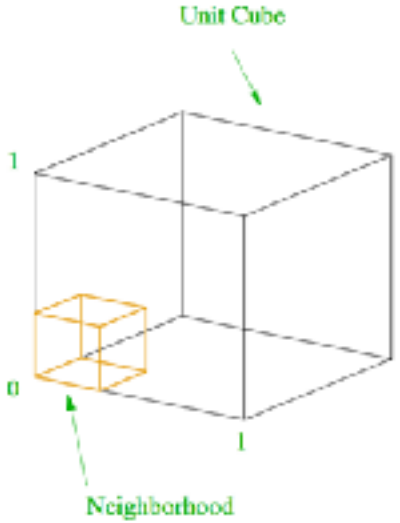
If d = 1 $P = 0.3$

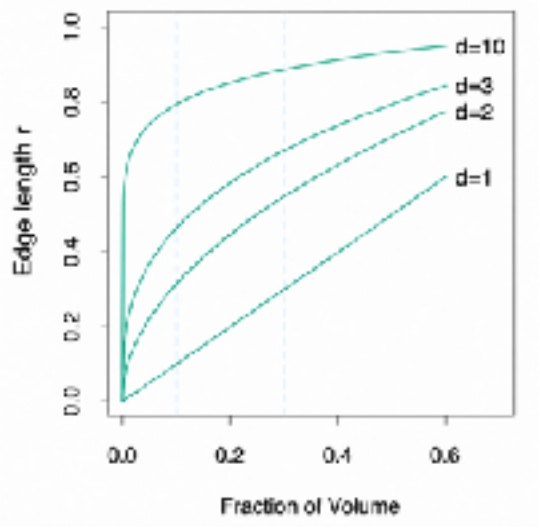
If d = 2

X is uniformly distributed over [0,1]^d. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of length r?

Curse of dimensionality, example 1





side length r

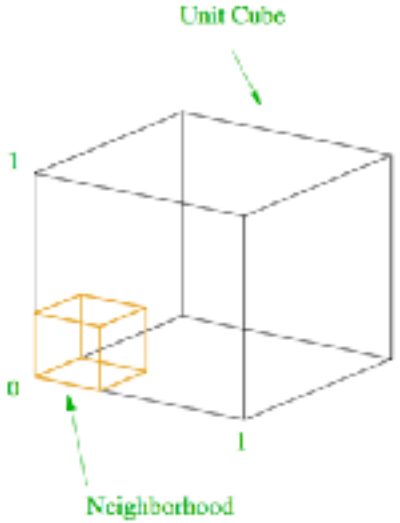
If d = 1 $P = 0.3$

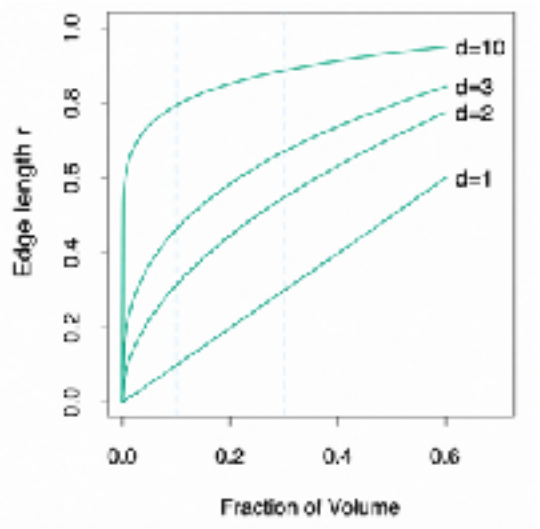
If d = 2

X is uniformly distributed over [0,1]. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of length r?

Curse of dimensionality, example 1





side length r

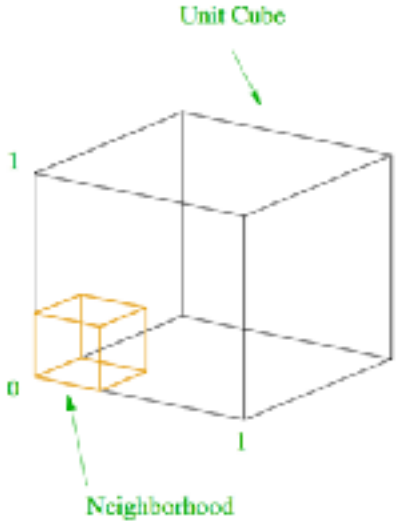
If d = 1 $P = 0.3$

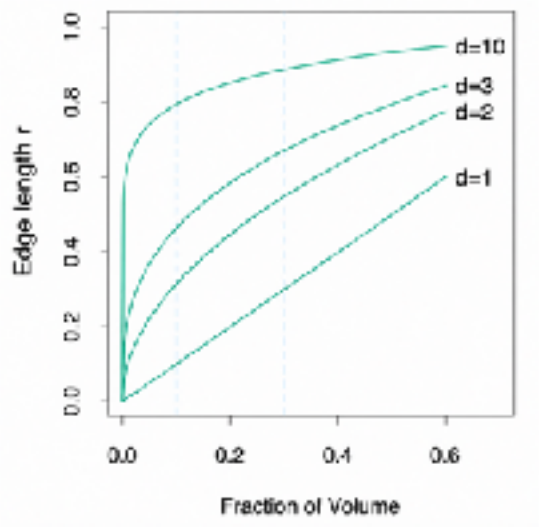
If d = 2 $P = 0.3^2 = 0.09$

X is uniformly distributed over [0,1]. What is $P(X \in [0, r]^d)$?

How many samples do we need so that a nearest neighbor is within a cube of length r?

Curse of dimensionality, example 1





If d = 1 $P = 0.3$

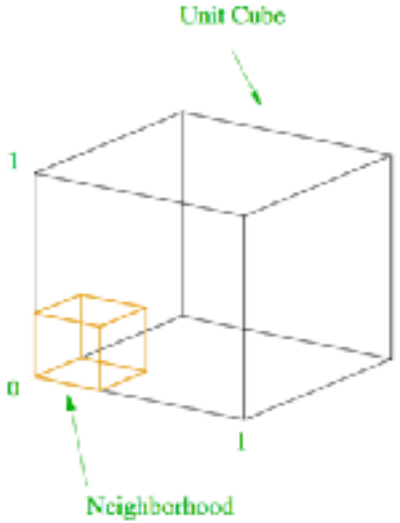
If d = 2 $P = 0.3^2 = 0.09$

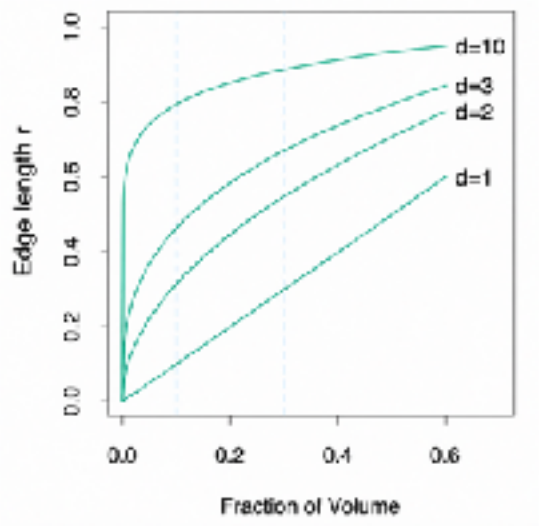
side length r

X is uniformly distributed over [0,1]^d. What is $P(X \in [0, r]^d) = \frac{1}{r^d}$ **?**

How many samples do we need so that a nearest neighbor is within a cube of length r**?**

Curse of dimensionality, example 1





If d = 1 $P = 0.3$

If d = 2 $P = 0.3^2 = 0.09$

side length r

X is uniformly distributed over [0,1]^d. What is ?

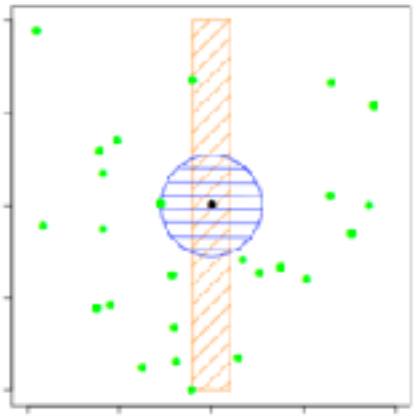
$$P(X \in [0, r]^d) = \frac{1}{r^d}$$

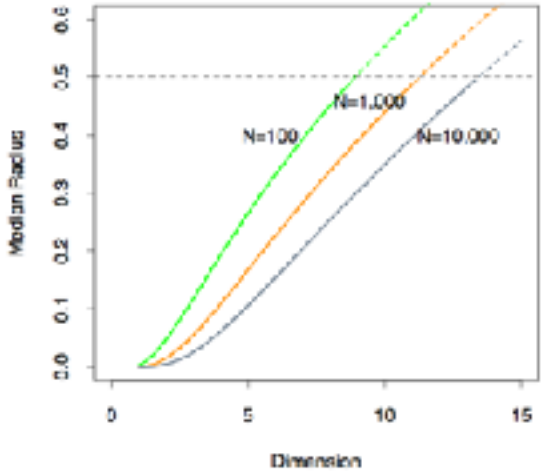
How many samples do we need so that a nearest neighbor is within a cube of length r?

Increases exponentially with dimension p!

Curse of dimensionality, example 2

$\{X_i\}_{i=1}^n$ are uniformly distributed over $[-.5, .5]^p$.



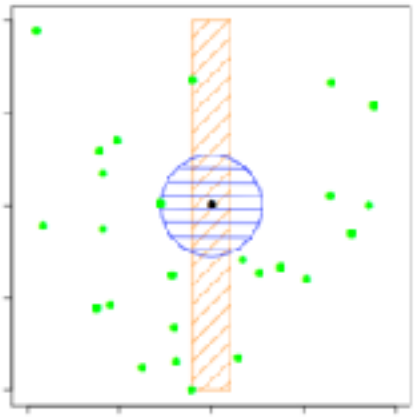


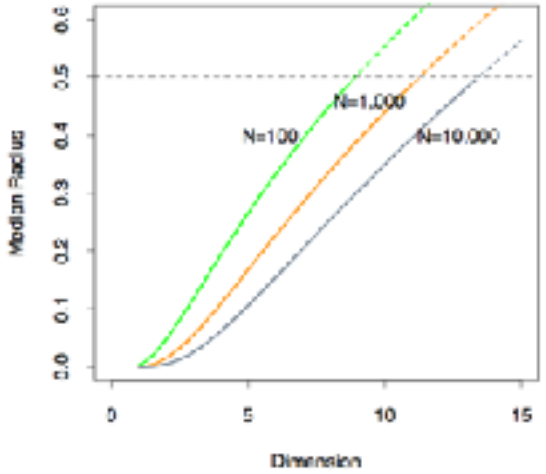
What is the median distance from a point at origin to its 1NN?

How many samples do we need so that a median Euclidean distance is within r?

Curse of dimensionality, example 2

$\{X_i\}_{i=1}^n$ are uniformly distributed over $[-.5, .5]^p$.





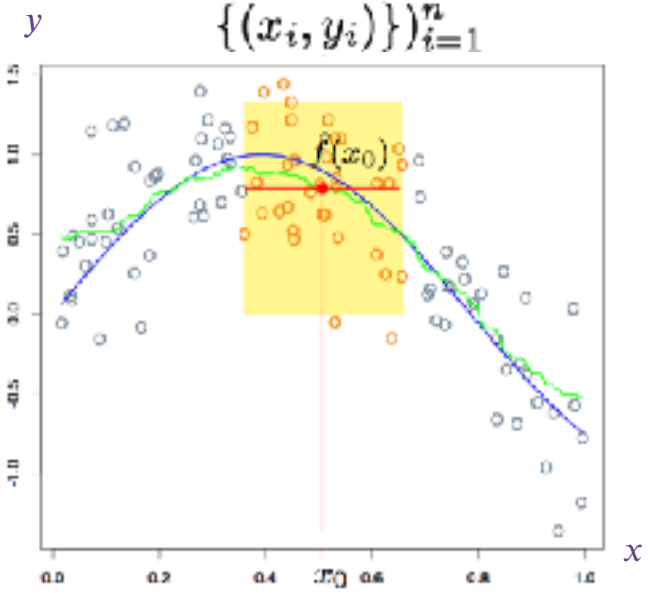
High dimensional space is hard to cover

What is the median distance from a point at origin to its 1NN?

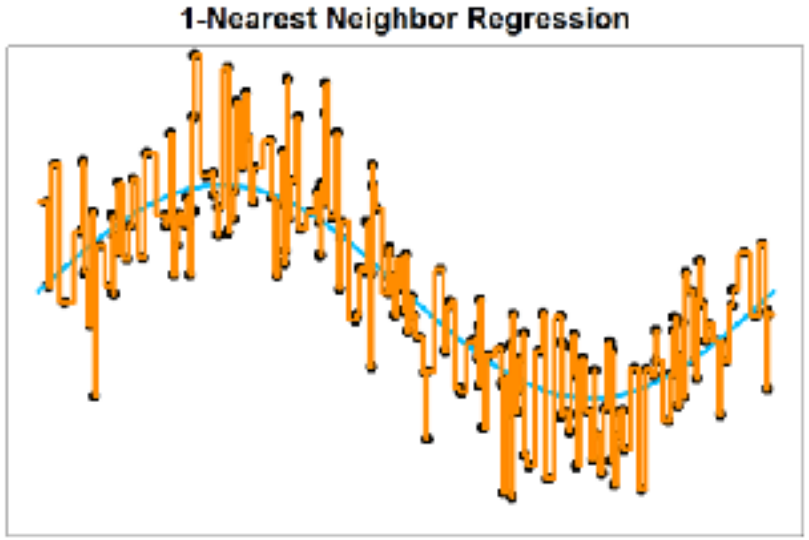
How many samples do we need so that a median Euclidean distance is within r?

Slides are not updated beyond this point,
download the latest ones from the course website

Nearest neighbor regression

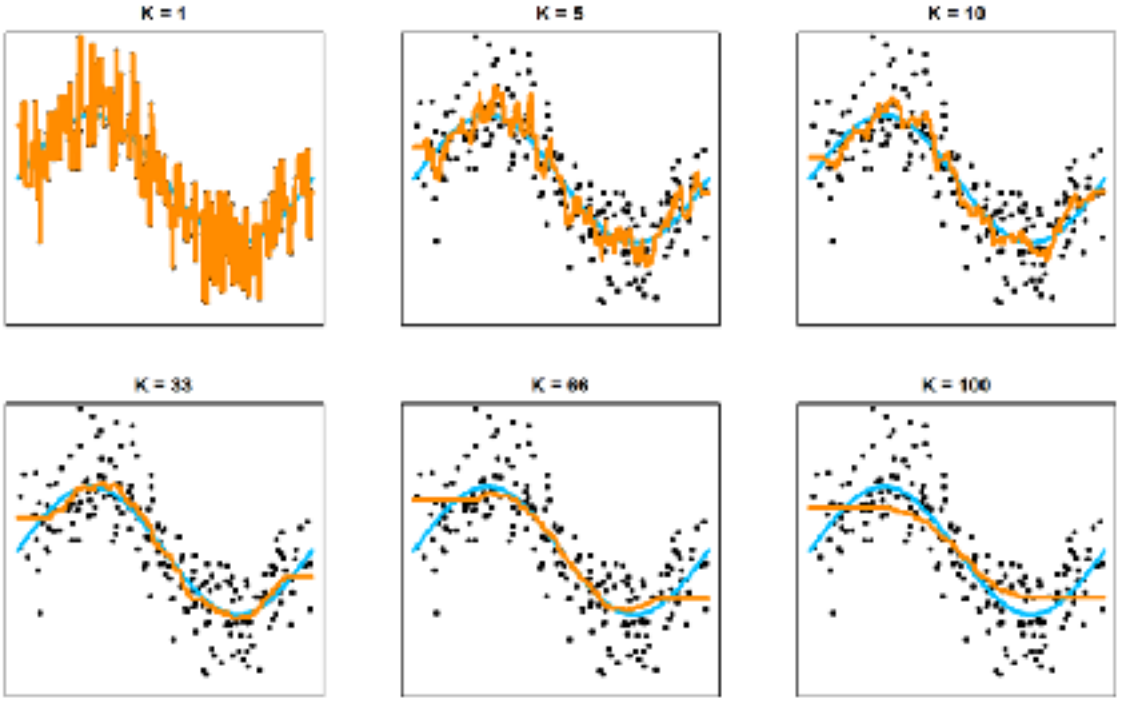


Overfitting



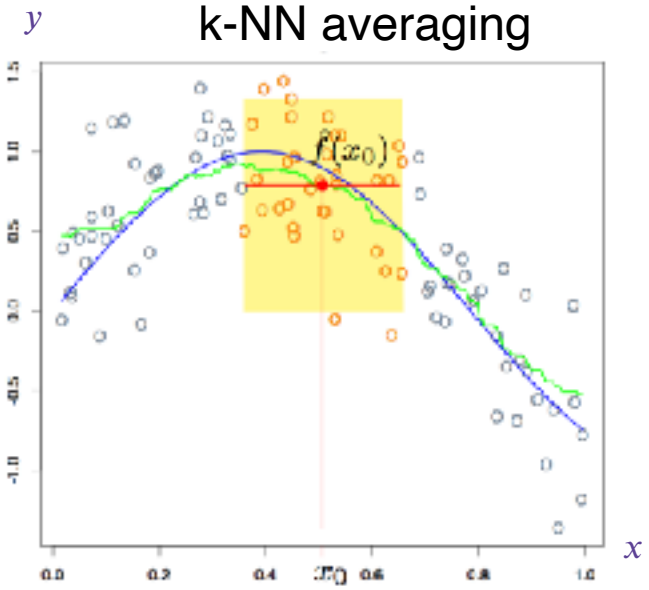
Figures from Ruoqing Zhu, Stat 542

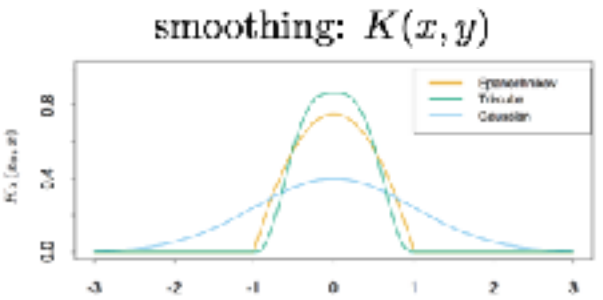
Bias vs variance



Figures from Ruoqing Zhu, Stat 542

Smoothed nearest neighbor regression

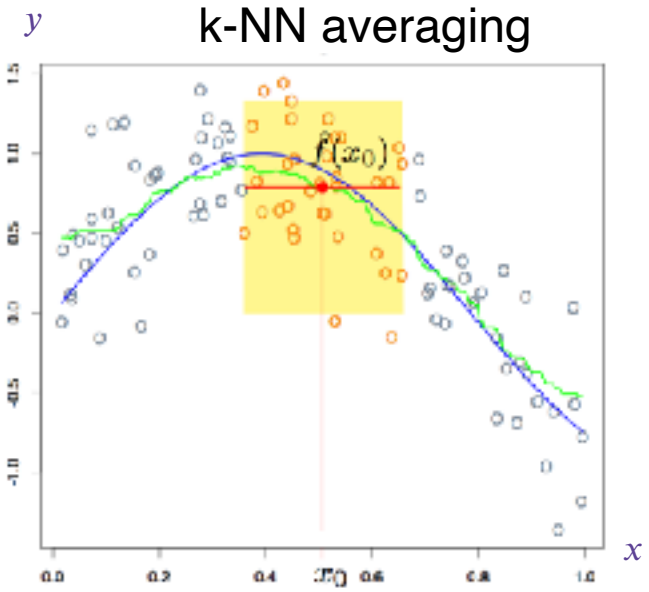


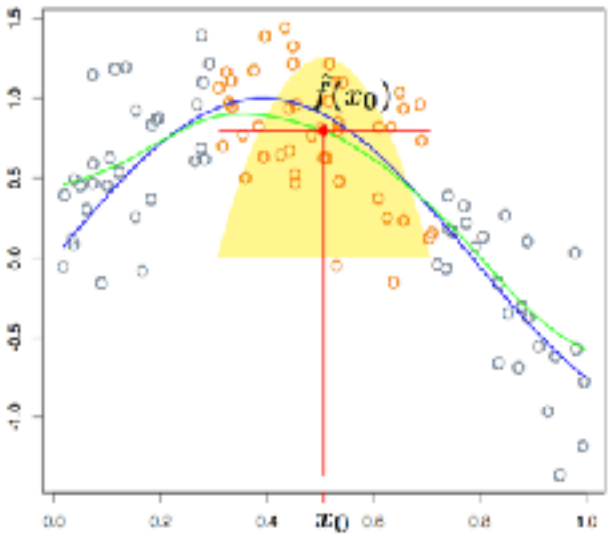


$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(x, x^{(i)}\right) y^{(i)}}{\sum_{i=1}^{n} K\left(x, x^{(i)}\right)}$$

Smoothed nearest neighbor regression

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(x, x^{(i)}\right) y^{(i)}}{\sum_{i=1}^{n} K\left(x, x^{(i)}\right)}$$

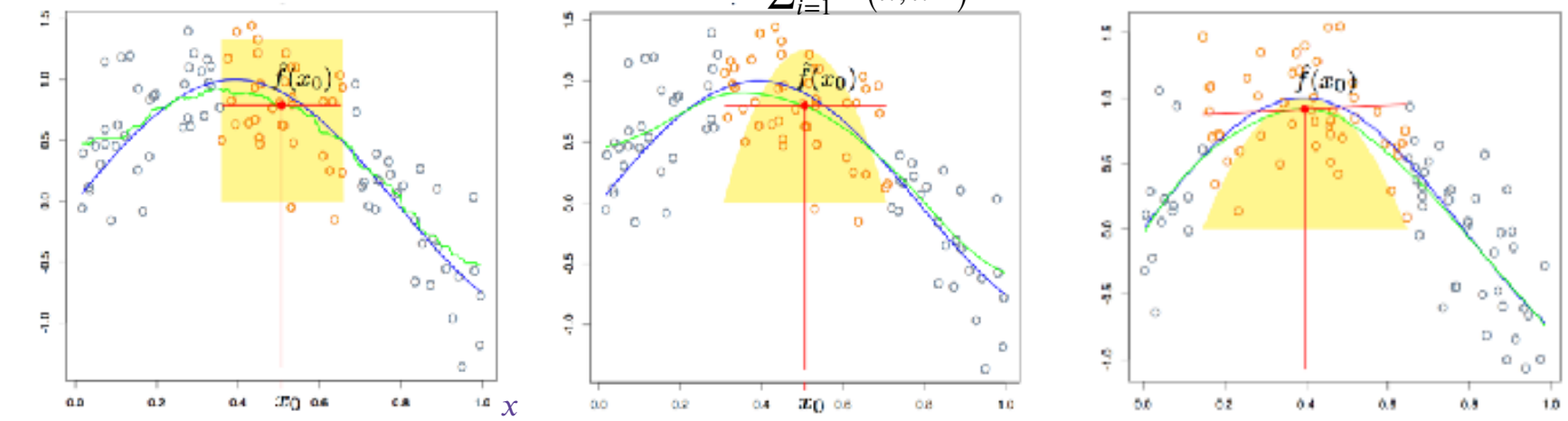




Locally linear regression

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(x, x^{(i)}\right) y^{(i)}}{\sum_{i=1}^{n} K\left(x, x^{(i)}\right)}$$

k-NN averaging



Have we seen non-parametric methods before?

Have we seen non-parametric methods before?

- Kernel methods are non-parametric:

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

Have we seen non-parametric methods before?

- Kernel methods are non-parametric:

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

  # parameters goes up with # data

Have we seen non-parametric methods before?

- Kernel methods are non-parametric:

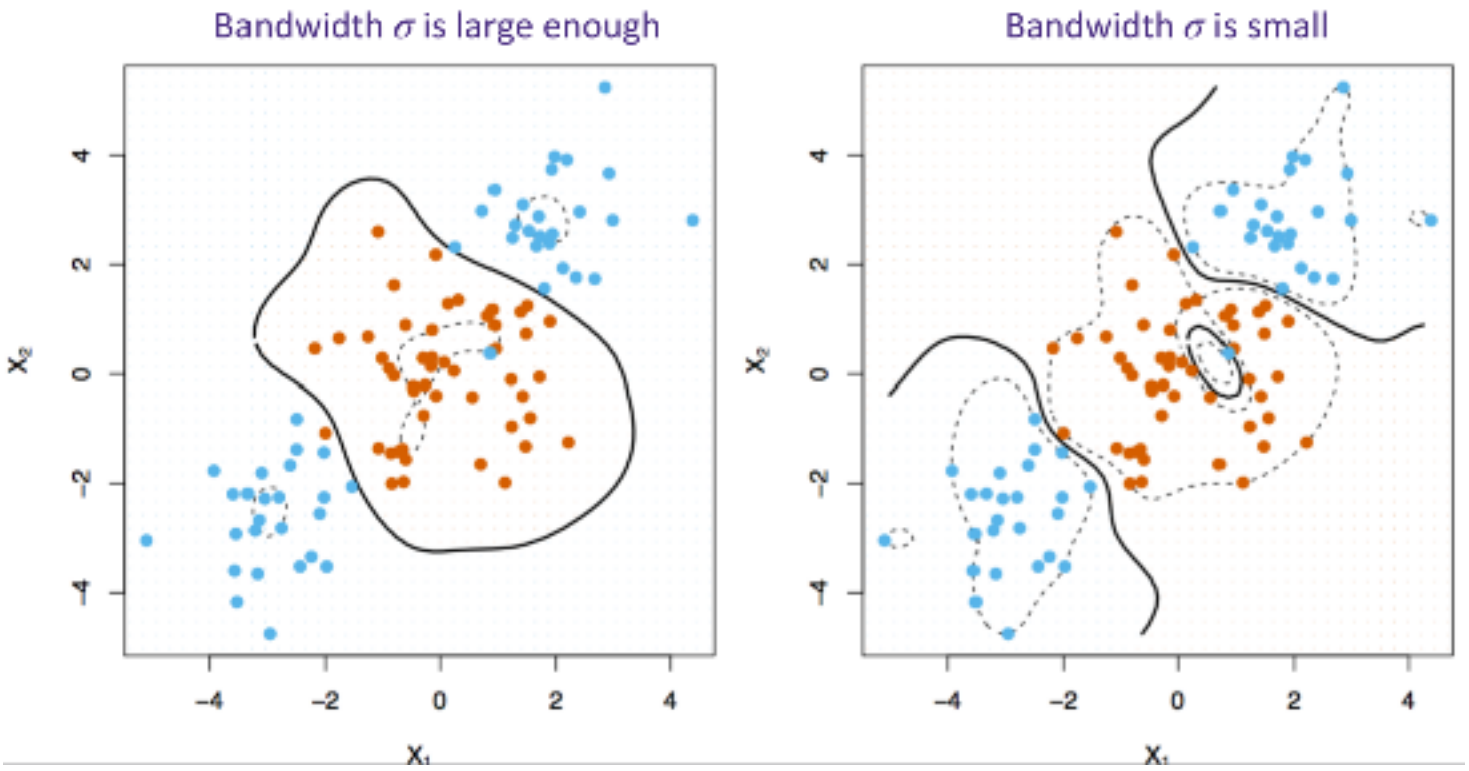
$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

  # parameters goes up with # data

- Compare with (smoothed) nearest neighbors:

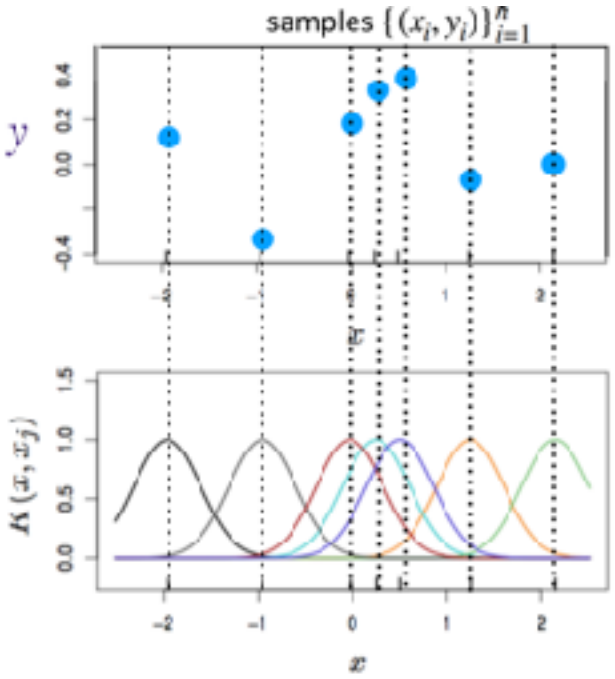
$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(x, x^{(i)}\right) y^{(i)}}{\sum_{i=1}^{n} K\left(x, x^{(i)}\right)}$$

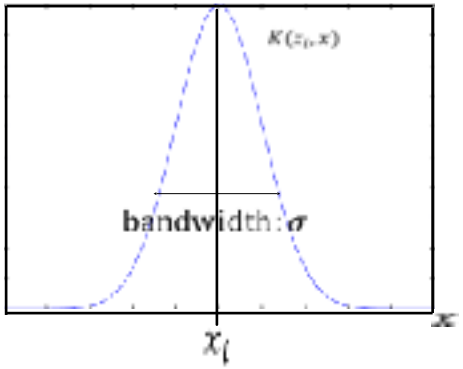
The Radial Basis Function (RBF) kernel $\exp\left(-\frac{\left\|x^{(i)}-x\right\|_2^2}{2\sigma^2}\right)$



The Radial Basis Function (RBF) kernel $\exp\left(-\frac{\left\|x^{(i)}-x\right\|_2^2}{2\sigma^2}\right)$

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x^{(i)}, x)$$





Kernel methods

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

Kernel methods

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

- Can be seen as a soft, learned version of "nearest" neighbors

Kernel methods

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

- Can be seen as a soft, learned version of "nearest" neighbors
- $K\left(x^{(i)}, x\right) = \phi\left(x^{(i)}\right)^{\top} \phi(x)$ defines "similarity" between $x^{(i)}$ and x

Kernel methods

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x^{(i)}, x\right)$$

- Can be seen as a soft, learned version of "nearest" neighbors
- $K\left(x^{(i)}, x\right) = \phi\left(x^{(i)}\right)^{\top} \phi(x)$ defines "similarity" between $x^{(i)}$ and x
- How many parameters?

Takeaways

Takeaways

- k-NN is very simple to explain and implement

Takeaways

- k-NN is very simple to explain and implement
- No training! But inference can still be computationally demanding.

Takeaways

- k-NN is very simple to explain and implement
- No training! But inference can still be computationally demanding.
- You can use other forms of distance (not just Euclidean)

Takeaways

- k-NN is very simple to explain and implement
- No training! But inference can still be computationally demanding.
- You can use other forms of distance (not just Euclidean)
- Smoothing and local linear regression can improve performance (at the cost of higher variance)

Takeaways

- k-NN is very simple to explain and implement
- No training! But inference can still be computationally demanding.
- You can use other forms of distance (not just Euclidean)
- Smoothing and local linear regression can improve performance (at the cost of higher variance)
- With a lot of data, “local methods” have strong, simple theoretical guarantees

Takeaways

- k-NN is very simple to explain and implement
- No training! But inference can still be computationally demanding.
- You can use other forms of distance (not just Euclidean)
- Smoothing and local linear regression can improve performance (at the cost of higher variance)
- With a lot of data, “local methods” have strong, simple theoretical guarantees
- Without a lot of data, neighborhoods aren’t “local” and methods suffer (curse of dimensionality)