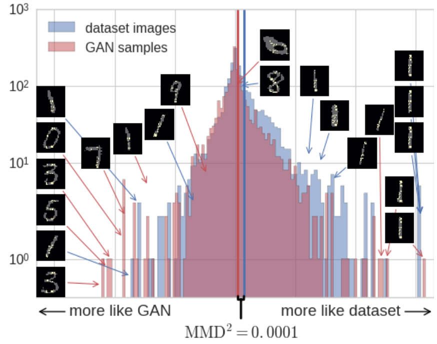
Are these datasets the same?
Two-sample testing for data scientists

Danica J. Sutherland (she)

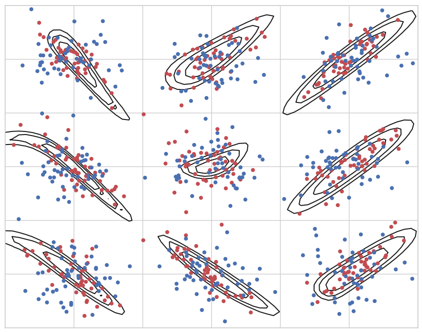
University of British Columbia (UBC) / Alberta Machine Intelligence Institute (Amii)

[ICLR-17]



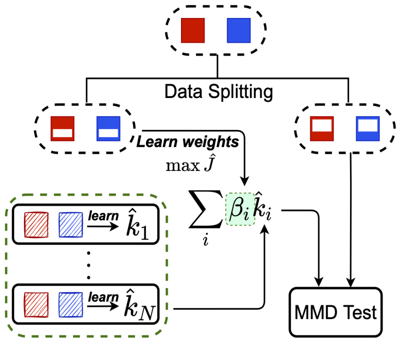
Hsiao-Yu (Fish) Tung
Heiko Strathmann
Soumyajit De
Aaditya Ramdas
Alex Smola
Arthur Gretton

[ICML-20]



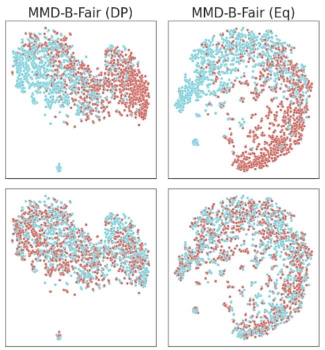
Feng Liu
Wenkai Xu
Jie Lu
Guangquan Zhang
Arthur Gretton

[NeurIPS-21]



Feng Liu
Wenkai Xu
Jie Lu

[AISTATS-23]



Namrata Deka

Pacific Conference on Artificial Intelligence – April 2, 2023

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X
  - Pretend there's a $\mathbb{P}$ that X is an i.i.d. sample from

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X
  - Pretend there's a $\mathbb{P}$ that X is an i.i.d. sample from
  - If it works on X, maybe it sorta works on $\mathbb{P}$

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X
  - Pretend there's a $\mathbb{P}$ that X is an i.i.d. sample from
  - If it works on X, maybe it sorta works on $\mathbb{P}$
  - Deploy on something that's maybe a distribution $\mathbb{Q}$
    - which might be sort of like $\mathbb{P}$

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X
  - Pretend there's a $\mathbb{P}$ that X is an i.i.d. sample from
  - If it works on X, maybe it sorta works on $\mathbb{P}$
  - Deploy on something that's maybe a distribution $\mathbb{Q}$
    - which might be sort of like $\mathbb{P}$
    - but probably changes over time...

Data drift

- Textbook machine learning:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - If it works on X, probably works on new samples from $\mathbb{P}$
- Really:
  - Train on X
  - Pretend there's a $\mathbb{P}$ that X is an i.i.d. sample from
  - If it works on X, maybe it sorta works on $\mathbb{P}$
  - Deploy on something that's maybe a distribution $\mathbb{Q}$
    - which might be sort of like $\mathbb{P}$
    - ~~but probably changes over time…~~

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- How is $\mathbb{P}$ different from $\mathbb{Q}$?

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- ~~Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?~~

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- ~~Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?~~
- Is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?

Two-sample testing

- Given samples from two unknown distributions

$$\color{blue}{X \sim \mathbb{P}} \qquad \color{orange}{Y \sim \mathbb{Q}}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]
- Are these neurons' behavior affected by this odor?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [`MMDiff2`]
- Are these neurons' behavior affected by this odor?
- Do these `dob` and `birthday` columns mean the same thing?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [`MMDiff2`]
- Are these neurons' behavior affected by this odor?
- Do these `dob` and `birthday` columns mean the same thing?
- Does my generative model $\mathbb{Q}_\theta$ match $\mathbb{P}_{\text{data}}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Canadians have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [`MMDiff2`]
- Are these neurons' behavior affected by this odor?
- Do these `dob` and `birthday` columns mean the same thing?
- Does my generative model $\mathbb{Q}_\theta$ match $\mathbb{P}_{\text{data}}$?
- Independence testing: is $P(X, Y) = P(X)P(Y)$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?
- Hypothesis testing approach:

$$H_0 : \mathbb{P} = \mathbb{Q} \qquad H_1 : \mathbb{P} \neq \mathbb{Q}$$

Two-sample testing

- Given samples from two unknown distributions

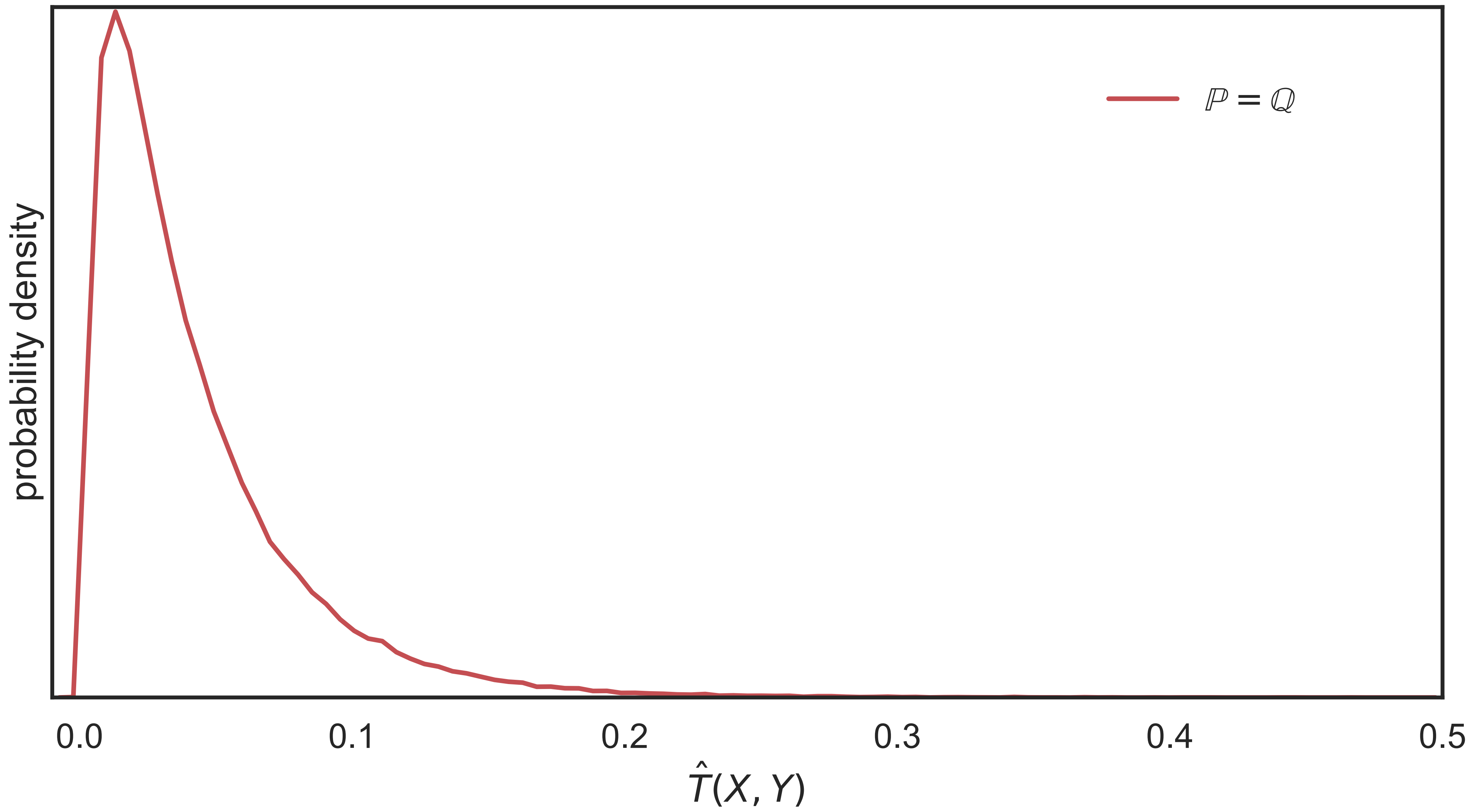
$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?
- Hypothesis testing approach:

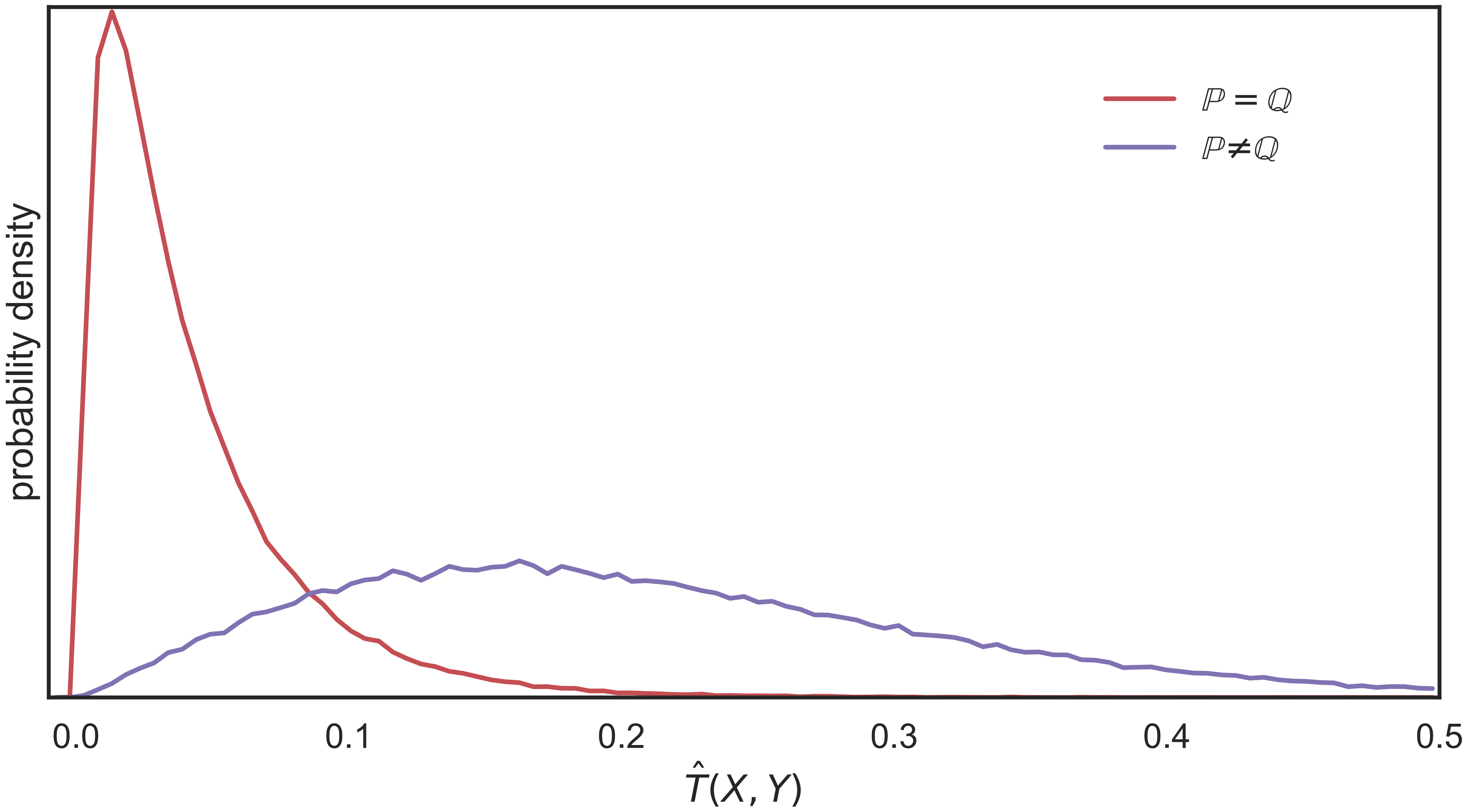
$$H_0 : \mathbb{P} = \mathbb{Q} \qquad H_1 : \mathbb{P} \neq \mathbb{Q}$$

- Reject null hypothesis H_0 if test statistic $\hat{T}(X, Y) > c_\alpha$

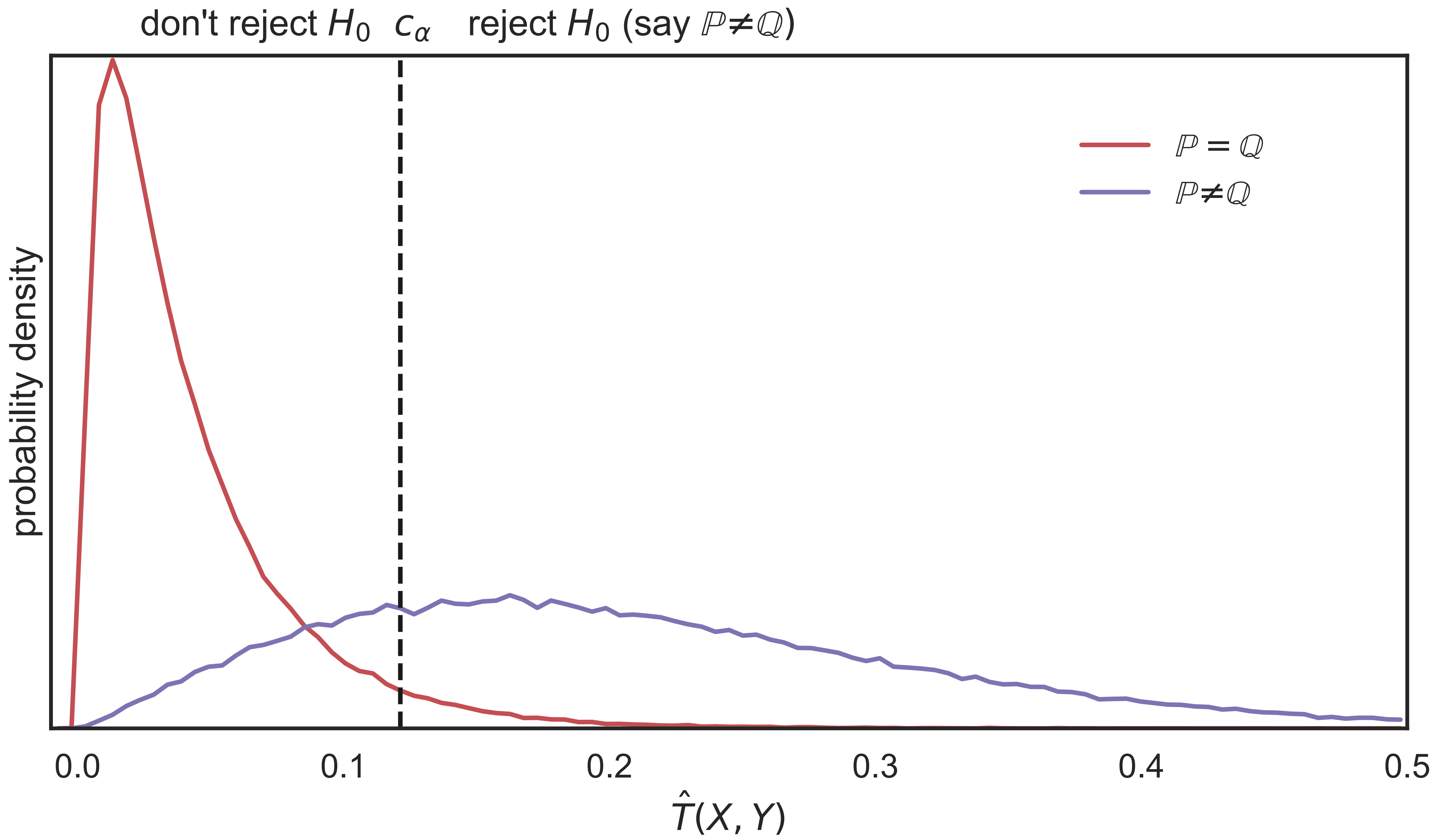
What's a hypothesis test again?



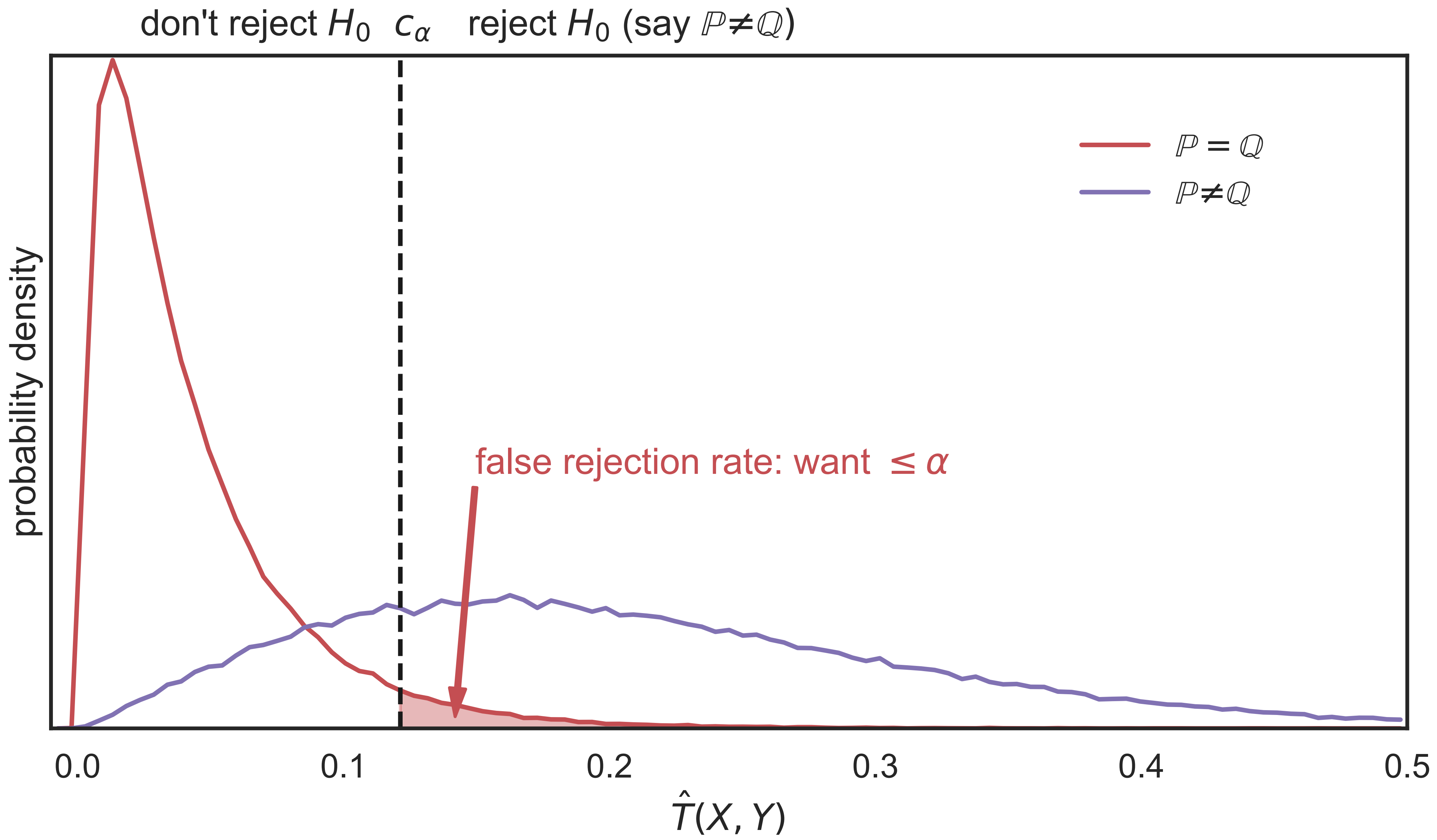
What's a hypothesis test again?



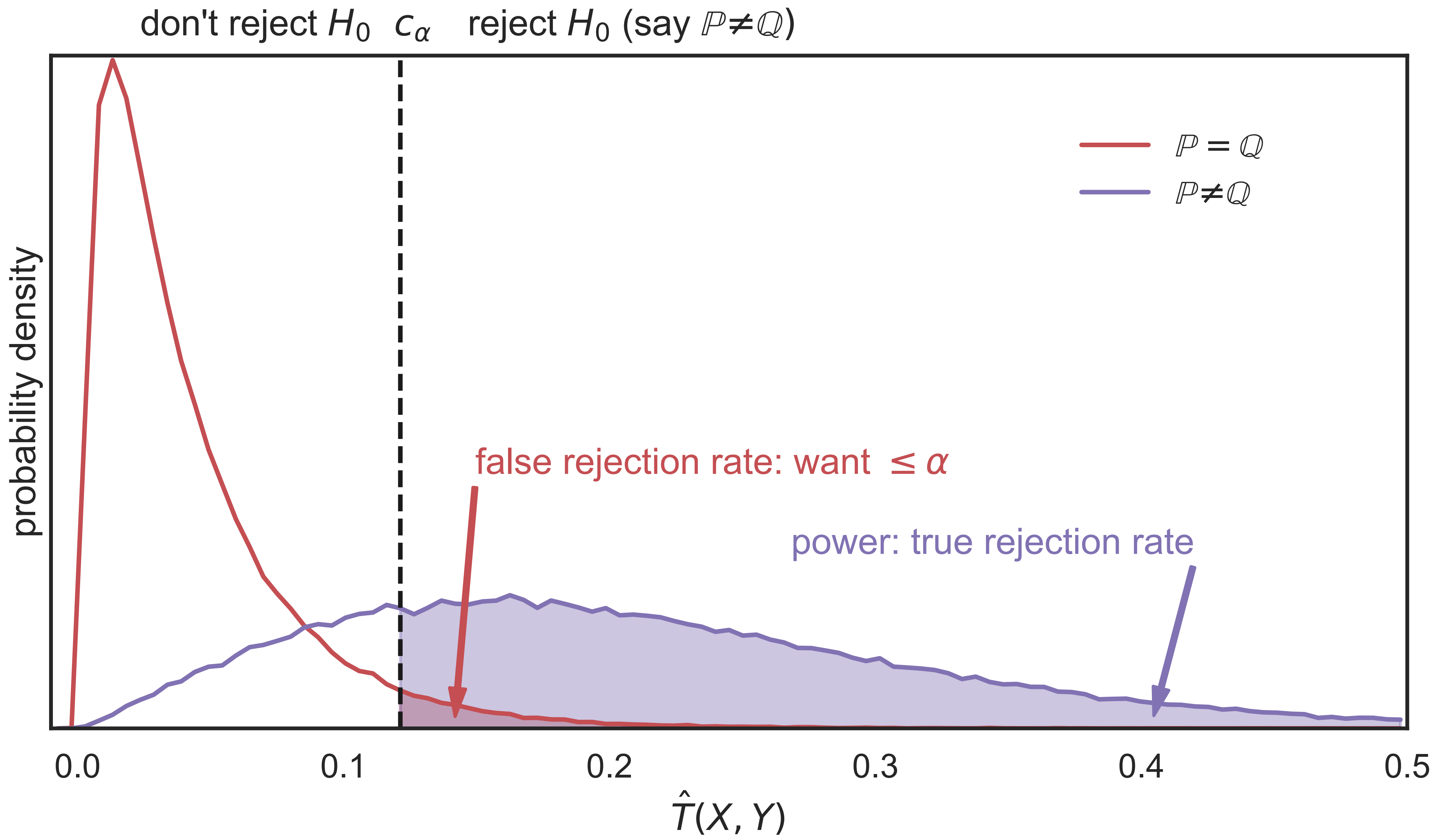
What's a hypothesis test again?



What's a hypothesis test again?



What's a hypothesis test again?



Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \le \alpha$

$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \qquad\qquad Y_1 \quad Y_2 \quad Y_3 \quad Y_4 \quad Y_5$

c_α: $1 - \alpha$th quantile of $\left\{ \qquad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

X_1 X_2 X_3 X_4 X_5 Y_1 Y_2 Y_3 Y_4 Y_5

c_α: $1-\alpha$th quantile of $\left\{ \quad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

X_1 X_2 X_3 X_4 X_5 Y_1 Y_2 Y_3 Y_4 Y_5

c_α: $1-\alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1), \qquad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

c_α: $1-\alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1),\ \hat{T}(\tilde{X}_2, \tilde{Y}_2), \qquad \right\}$

Permutation testing to find c_α

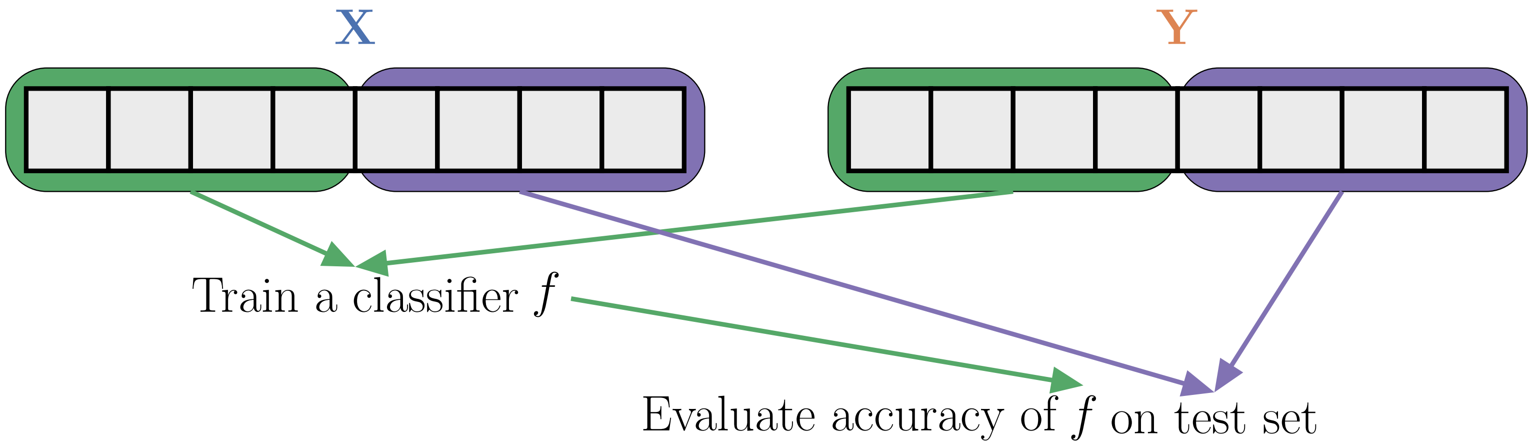
Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \qquad\qquad Y_1 \quad Y_2 \quad Y_3 \quad Y_4 \quad Y_5$

c_α: $1-\alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1),\ \hat{T}(\tilde{X}_2, \tilde{Y}_2),\ \cdots\right\}$

Classifier two-sample tests

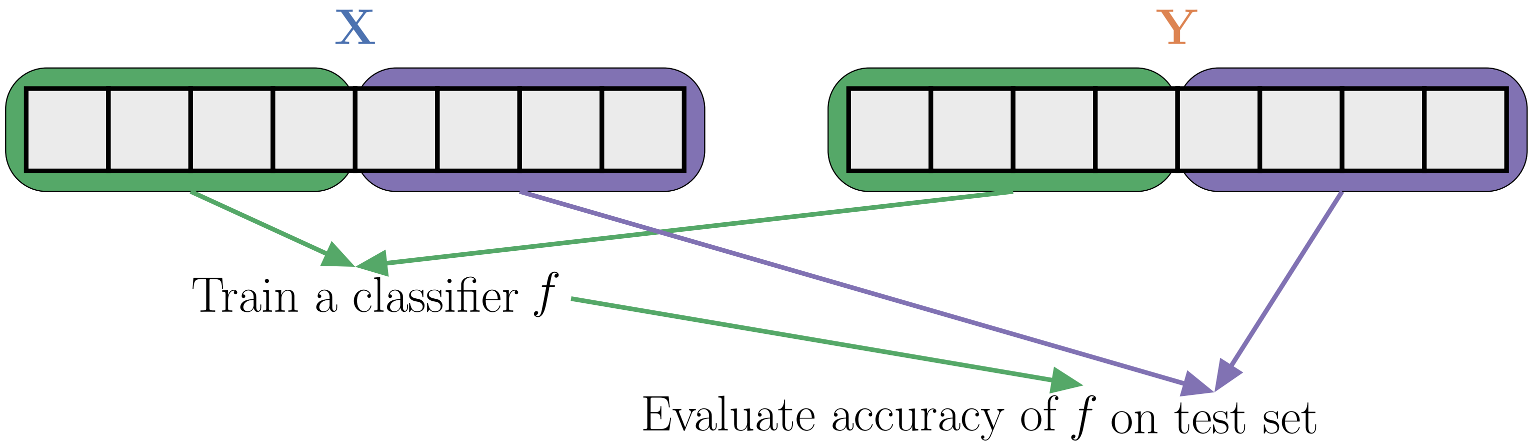
- We need a $\hat{T}(X, Y)$ that's large if $\mathbb{P} \neq \mathbb{Q}$, small if $\mathbb{P} = \mathbb{Q}$



- Can choose $\hat{T}(X, Y)$ as the accuracy of f on the test set

Classifier two-sample tests

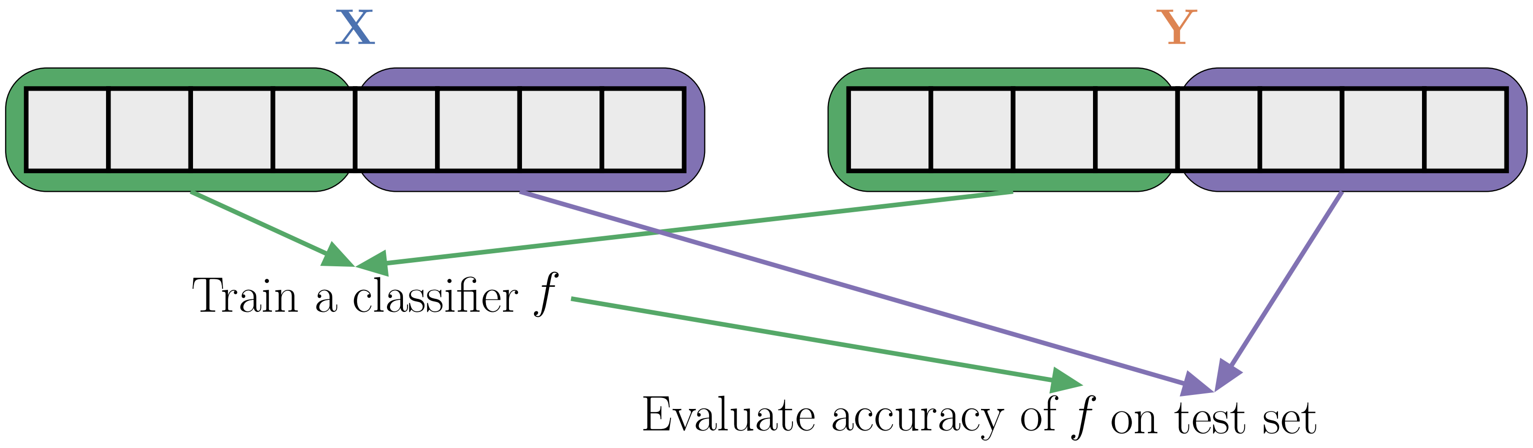
- We need a $\hat{T}(X, Y)$ that's large if $\mathbb{P} \neq \mathbb{Q}$, small if $\mathbb{P} = \mathbb{Q}$



- Can choose $\hat{T}(X, Y)$ as the accuracy of f on the test set
  - If $\mathbb{P} = \mathbb{Q}$, classification is impossible, and so $\hat{T} \sim \text{Binomial}(n, \frac{1}{2})$

Classifier two-sample tests

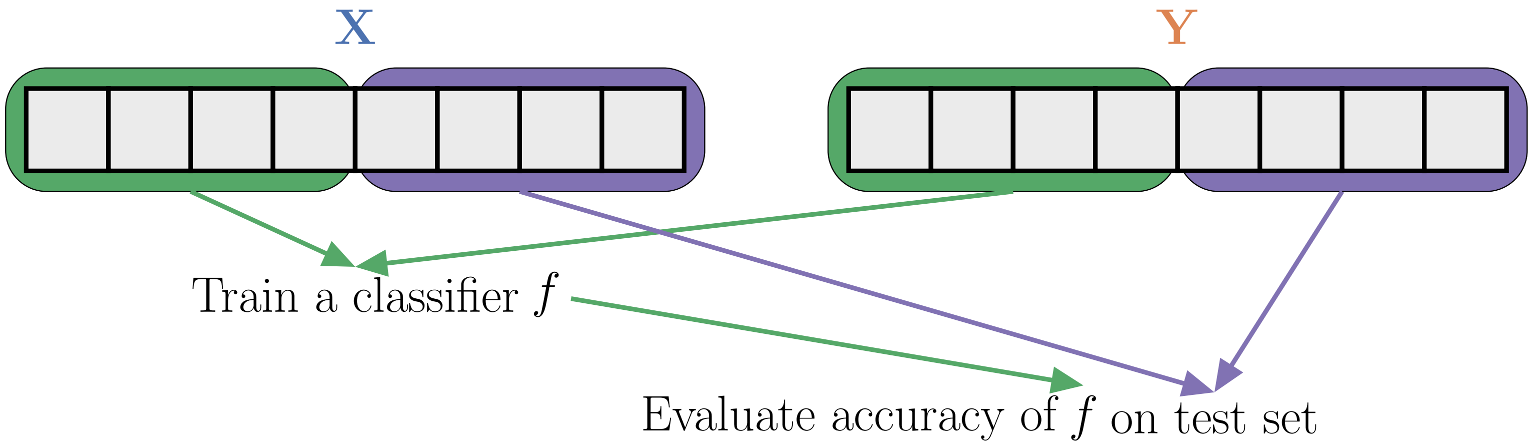
- We need a $\hat{T}(X, Y)$ that's large if $\mathbb{P} \neq \mathbb{Q}$, small if $\mathbb{P} = \mathbb{Q}$



- Can choose $\hat{T}(X, Y)$ as the accuracy of f on the test set
  - If $\mathbb{P} = \mathbb{Q}$, classification is impossible, and so $\hat{T} \sim \mathrm{Binomial}(n, \frac{1}{2})$
- Usually performs better: $\hat{T}(X, Y) = \underset{x \in X_{test}}{\mathrm{mean}}[f(x)] - \underset{y \in Y_{test}}{\mathrm{mean}}[f(y)]$

Classifier two-sample tests

- We need a $\hat{T}(X, Y)$ that's large if $\mathbb{P} \neq \mathbb{Q}$, small if $\mathbb{P} = \mathbb{Q}$



- Can choose $\hat{T}(X, Y)$ as the accuracy of f on the test set
  - If $\mathbb{P} = \mathbb{Q}$, classification is impossible, and so $\hat{T} \sim \mathrm{Binomial}(n, \frac{1}{2})$
- Usually performs better: $\hat{T}(X, Y) = \underset{x \in X_{test}}{\mathrm{mean}}[f(x)] - \underset{y \in Y_{test}}{\mathrm{mean}}[f(y)]$
  - If $\mathbb{P} = \mathbb{Q}$, $\hat{T} \to$ normal distribution (but permuting on test set is better)

A more general framework

- C2ST-L: $\hat{T}(X, Y) = \operatorname*{mean}_{x \in X_{test}}[f(x)] - \operatorname*{mean}_{y \in Y_{test}}[f(y)]$
  - $f(x) \in \mathbb{R}$ is a classifier's "logit":
    log probability x is from $\mathbb{P}$ rather than $\mathbb{Q}$, plus const

A more general framework

- C2ST-L: $\hat{T}(X, Y) = \operatorname*{mean}_{x \in X_{test}}[f(x)] - \operatorname*{mean}_{y \in Y_{test}}[f(y)]$
  - $f(x) \in \mathbb{R}$ is a classifier's “logit”:
    log probability x is from $\mathbb{P}$ rather than $\mathbb{Q}$, plus const
- Basically the same: $\hat{T}(X, Y) = \left| \operatorname*{mean}_{x \in X_{test}}[f(x)] - \operatorname*{mean}_{y \in Y_{test}}[f(y)] \right|$

A more general framework

- C2ST-L: $\hat{T}(X, Y) = \operatorname*{mean}_{x \in X_{test}}[f(x)] - \operatorname*{mean}_{y \in Y_{test}}[f(y)]$
  - $f(x) \in \mathbb{R}$ is a classifier's “logit”: log probability x is from $\mathbb{P}$ rather than $\mathbb{Q}$, plus const
- Basically the same: $\hat{T}(X, Y) = \left| \operatorname*{mean}_{x \in X_{test}}[f(x)] - \operatorname*{mean}_{y \in Y_{test}}[f(y)] \right|$
- What if we use more general *features* of the data?

$$\hat{T}(X, Y) = \left\| \operatorname*{mean}_{x \in X_{test}}[\varphi(x)] - \operatorname*{mean}_{y \in Y_{test}}[\varphi(y)] \right\|$$

Difference between mean embeddings

$$\hat{T}(X, Y)^2 = \left\| \underset{x \in X_{test}}{\text{mean}}[\varphi(x)] - \underset{y \in Y_{test}}{\text{mean}}[\varphi(y)] \right\|^2$$

Difference between mean embeddings

$$\hat{T}(X, Y)^2 = \left\| \underset{x \in X_{test}}{\text{mean}}[\varphi(x)] - \underset{y \in Y_{test}}{\text{mean}}[\varphi(y)] \right\|^2$$

$$= \underset{x \neq x'}{\text{mean}}[\varphi(x) \cdot \varphi(x')] - 2 \underset{x,y}{\text{mean}}[\varphi(x) \cdot \varphi(y)] + \underset{y \neq y'}{\text{mean}}[\varphi(y) \cdot \varphi(y')]$$

Difference between mean embeddings

$$\hat{T}(X, Y)^2 = \left\| \operatorname*{mean}_{x \in X_{test}} [\varphi(x)] - \operatorname*{mean}_{y \in Y_{test}} [\varphi(y)] \right\|^2$$

$$= \operatorname*{mean}_{x \neq x'} [\varphi(x) \cdot \varphi(x')] - 2 \operatorname*{mean}_{x,y} [\varphi(x) \cdot \varphi(y)] + \operatorname*{mean}_{y \neq y'} [\varphi(y) \cdot \varphi(y')]$$

Only use data through $\varphi(x) \cdot \varphi(y) = k(x, y)$: can **kernelize**!

Difference between mean embeddings

$$\hat{T}(X,Y)^2 = \left\| \underset{x \in X_{test}}{\text{mean}}[\varphi(x)] - \underset{y \in Y_{test}}{\text{mean}}[\varphi(y)] \right\|^2$$

$$= \underset{x \neq x'}{\text{mean}}[\varphi(x) \cdot \varphi(x')] - 2 \underset{x,y}{\text{mean}}[\varphi(x) \cdot \varphi(y)] + \underset{y \neq y'}{\text{mean}}[\varphi(y) \cdot \varphi(y')]$$

Only use data through $\varphi(x) \cdot \varphi(y) = k(x,y)$: can **kernelize**!

K_{XX}

			
	1.0	0.2	0.6
	0.2	1.0	0.5
	0.6	0.5	1.0

Difference between mean embeddings

$$\hat{T}(X,Y)^2 = \left\| \operatorname*{mean}_{x \in X_{test}}[\varphi(x)] - \operatorname*{mean}_{y \in Y_{test}}[\varphi(y)] \right\|^2$$

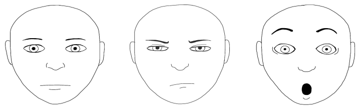
$$= \operatorname*{mean}_{x \neq x'}[\varphi(x) \cdot \varphi(x')] - 2 \operatorname*{mean}_{x,y}[\varphi(x) \cdot \varphi(y)] + \operatorname*{mean}_{y \neq y'}[\varphi(y) \cdot \varphi(y')]$$

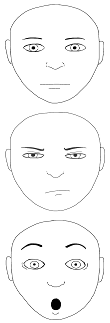
Only use data through $\varphi(x) \cdot \varphi(y) = k(x, y)$: can **kernelize**!

K_{XX}

	1.0	0.2	0.6
	0.2	1.0	0.5
	0.6	0.5	1.0

K_{YY}





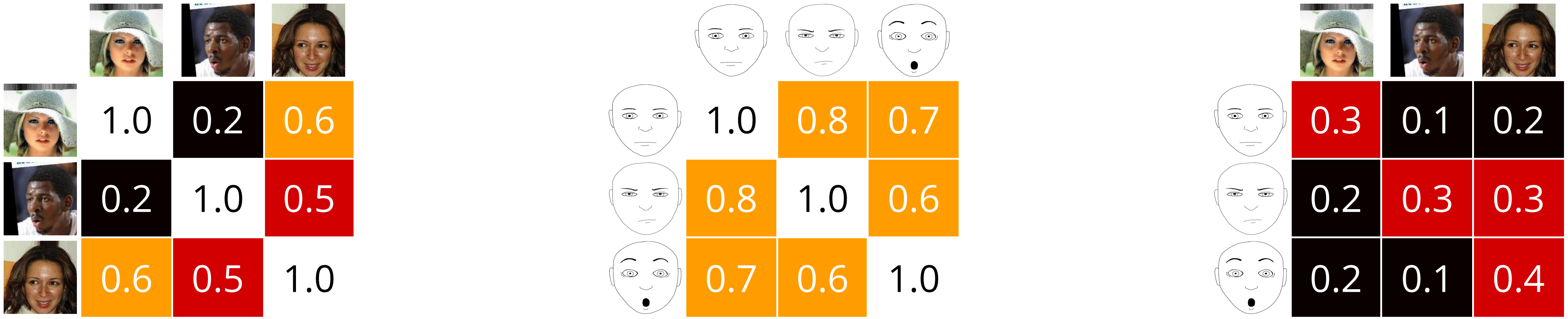
	1.0	0.8	0.7
	0.8	1.0	0.6
	0.7	0.6	1.0

Difference between mean embeddings

$$\hat{T}(X, Y)^2 = \left\| \underset{x \in X_{test}}{\text{mean}}[\varphi(x)] - \underset{y \in Y_{test}}{\text{mean}}[\varphi(y)] \right\|^2$$

$$= \underset{x \neq x'}{\text{mean}}[\varphi(x) \cdot \varphi(x')] - 2 \underset{x,y}{\text{mean}}[\varphi(x) \cdot \varphi(y)] + \underset{y \neq y'}{\text{mean}}[\varphi(y) \cdot \varphi(y')]$$

Only use data through $\varphi(x) \cdot \varphi(y) = k(x, y)$: can **kernelize**!



K_{XX}

	1.0	0.2	0.6
	0.2	1.0	0.5
	0.6	0.5	1.0

K_{YY}

	1.0	0.8	0.7
	0.8	1.0	0.6
	0.7	0.6	1.0

K_{XY}

	0.3	0.1	0.2
	0.2	0.3	0.3
	0.2	0.1	0.4

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathbf{sign}(f(x))$, $f(x) = w^\mathsf{T} (x, 1)$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathrm{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$

What's a kernel again?

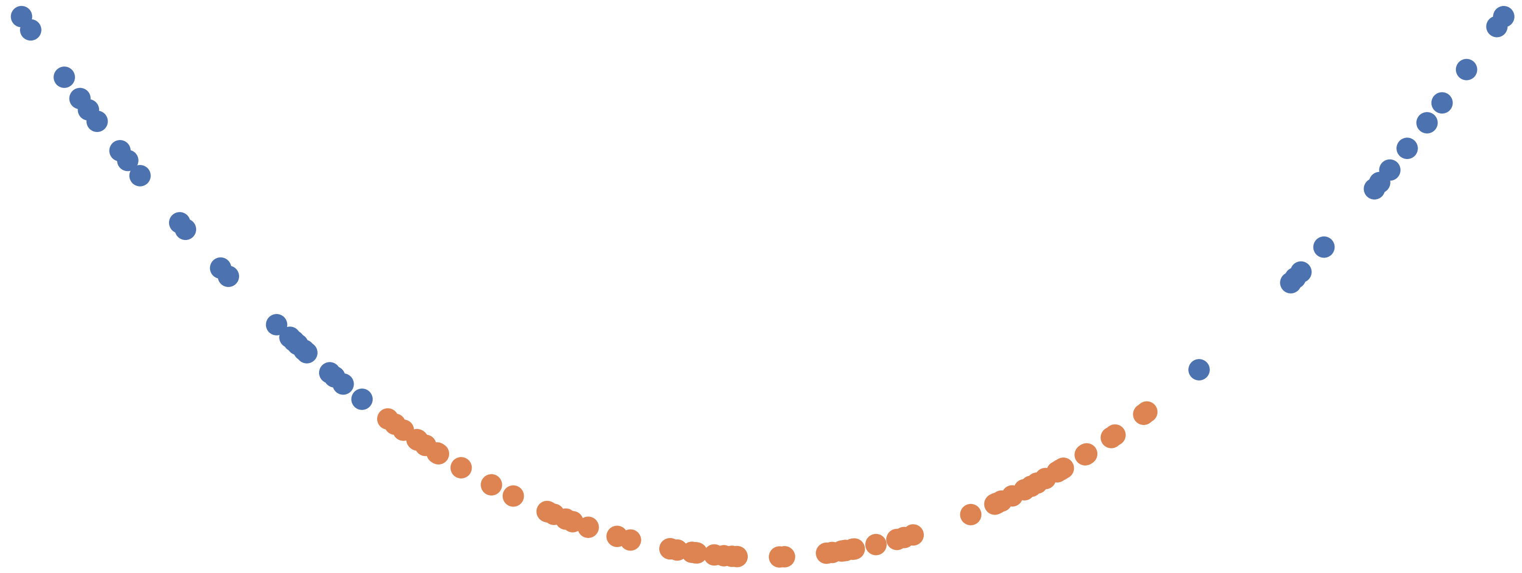
- Linear classifiers: $\hat{y}(x) = \mathrm{sign}(f(x))$, $f(x) = w^\mathsf{T} (x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T} \left(x, x^2, 1\right) = w^\mathsf{T} \varphi(x)$$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T}\,(x, 1)$
- Use a "richer" x:

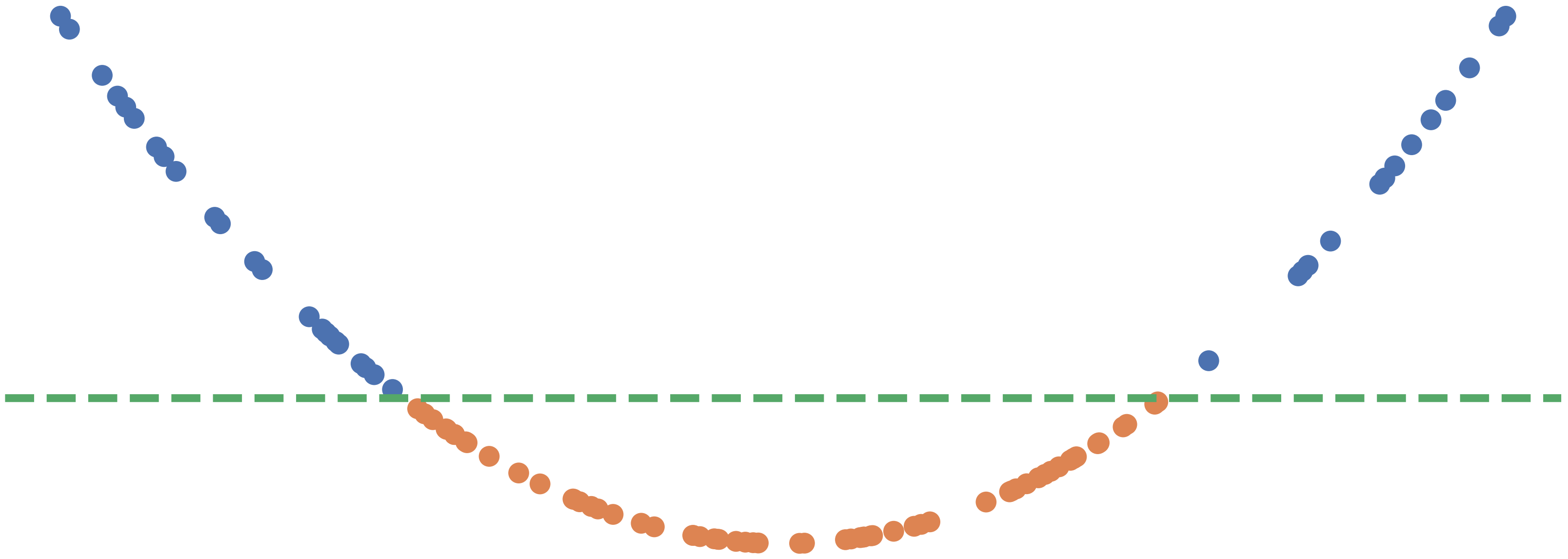
$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T}\varphi(x)$$



What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T}\varphi(x)$$



What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathbf{sign}(f(x))$, $f(x) = w^\mathsf{T} \, (x, 1)$
- Use a “richer” x:

$$f(x) = w^\mathsf{T} \left(x, x^2, 1\right) = w^\mathsf{T} \varphi(x)$$

- Can avoid explicit $\varphi(x)$; instead $k(x, y) = \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathbf{sign}(f(x))$, $f(x) = w^\mathsf{T}\,(x, 1)$
- Use a “richer” x:

$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T} \varphi(x)$$

- Can avoid explicit $\varphi(x)$; instead $k(x, y) = \langle \varphi(x), \varphi(y) \rangle_\mathcal{H}$
- “Kernelized” algorithms access data only through $k(x, y)$

$$f(x) = \langle w, \varphi(x) \rangle_\mathcal{H} = \sum_{i=1}^{n} \alpha_i k(X_i, x)$$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathbf{sign}(f(x))$, $f(x) = w^\mathsf{T} \ (x, 1)$
- Use a “richer” x:

$$f(x) = w^\mathsf{T} \ \left(x, x^2, 1\right) = w^\mathsf{T} \varphi(x)$$

- Can avoid explicit $\varphi(x)$; instead $k(x, y) = \langle \varphi(x), \varphi(y) \rangle_\mathcal{H}$
- “Kernelized” algorithms access data only through $k(x, y)$

$$f(x) = \langle w, \varphi(x) \rangle_\mathcal{H} = \sum_{i=1}^{n} \alpha_i k(X_i, x)$$

- Induces a notion of “smoothness” on functions, $\|f\|_\mathcal{H} = \sqrt{\alpha^\mathsf{T} K \alpha}$

Reproducing Kernel Hilbert Space (RKHS)

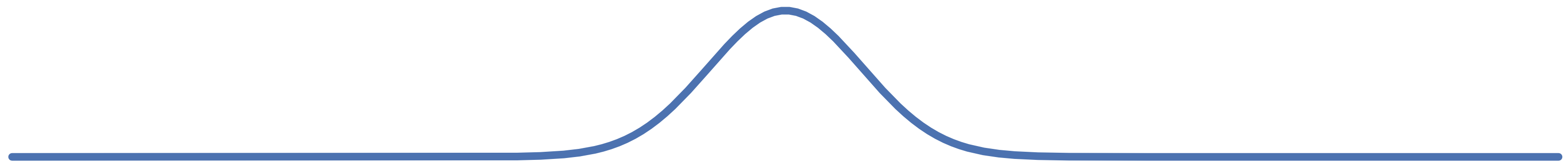
- Example: Gaussian RBF

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF

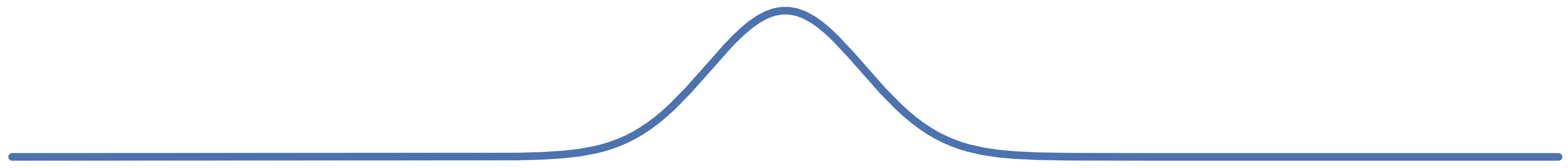
$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$



Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF / exponentiated quadratic / squared exponential / …

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

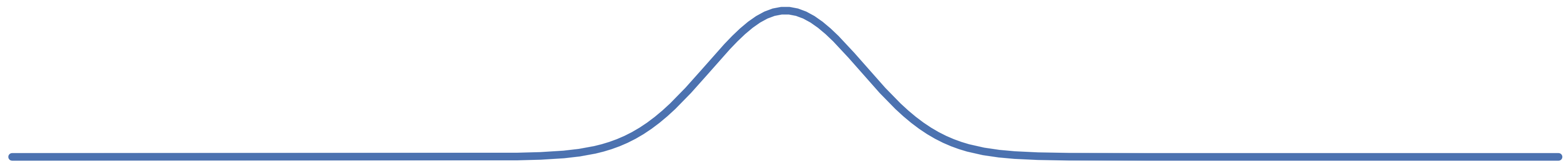


Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF / exponentiated quadratic / squared exponential / …

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Some functions with small $\|f\|_{\mathcal{H}}$:



Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF / exponentiated quadratic / squared exponential / …

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Some functions with small $\|f\|_{\mathcal{H}}$:

Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF / exponentiated quadratic / squared exponential / …

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Some functions with small $\|f\|_{\mathcal{H}}$:

Reproducing Kernel Hilbert Space (RKHS)

- Example: Gaussian RBF / exponentiated quadratic / squared exponential / ...

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Some functions with small $\|f\|_{\mathcal{H}}$:

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}} [f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}} [f(Y)]$$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

Maximum Mean Discrepancy (MMD)

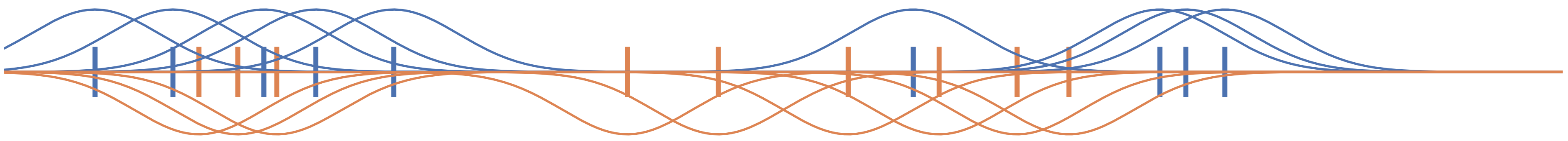
$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

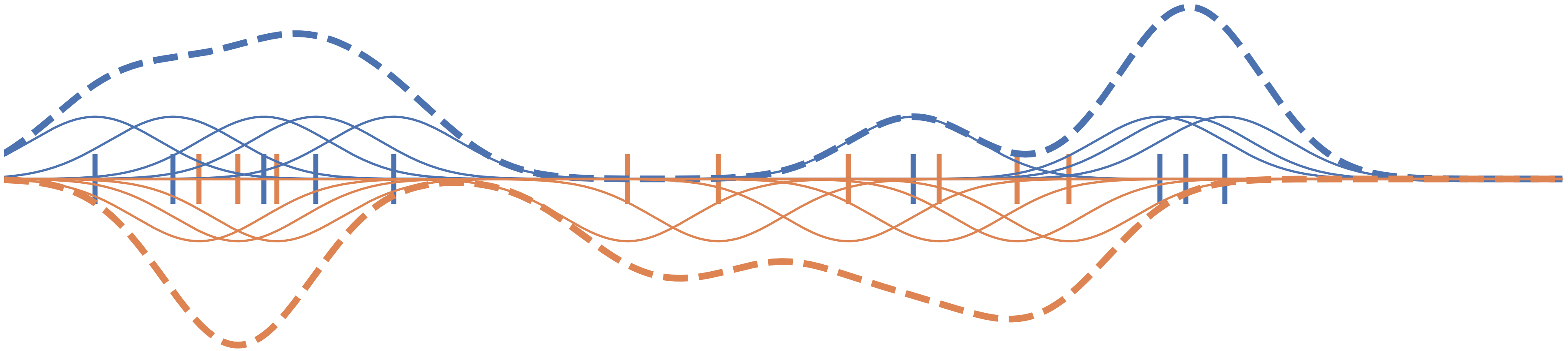
The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

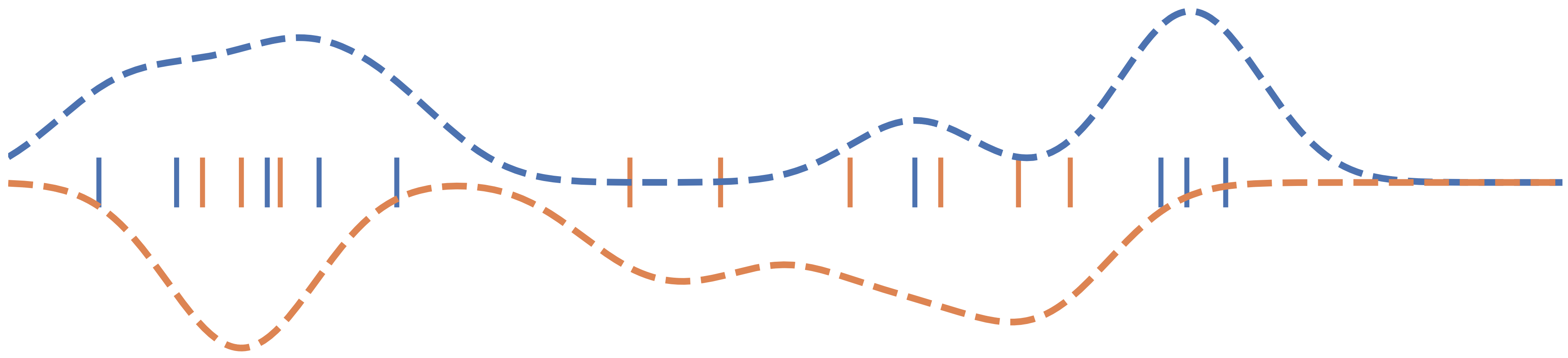
The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

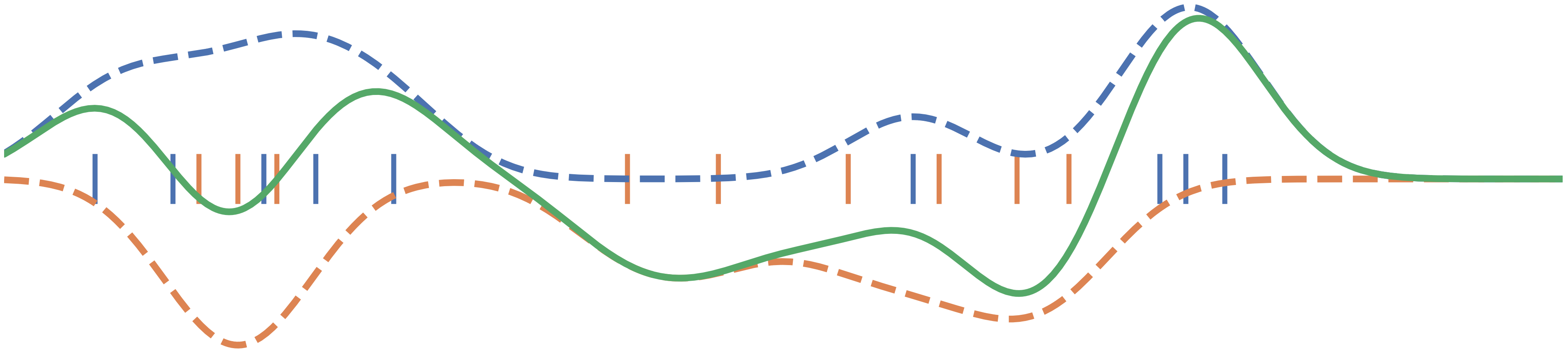
The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}} [f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}} [f(Y)]$$

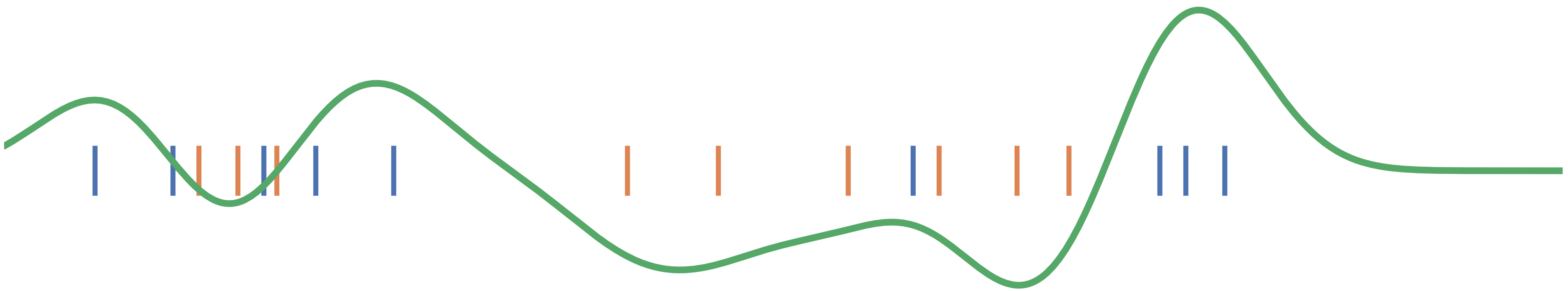
The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

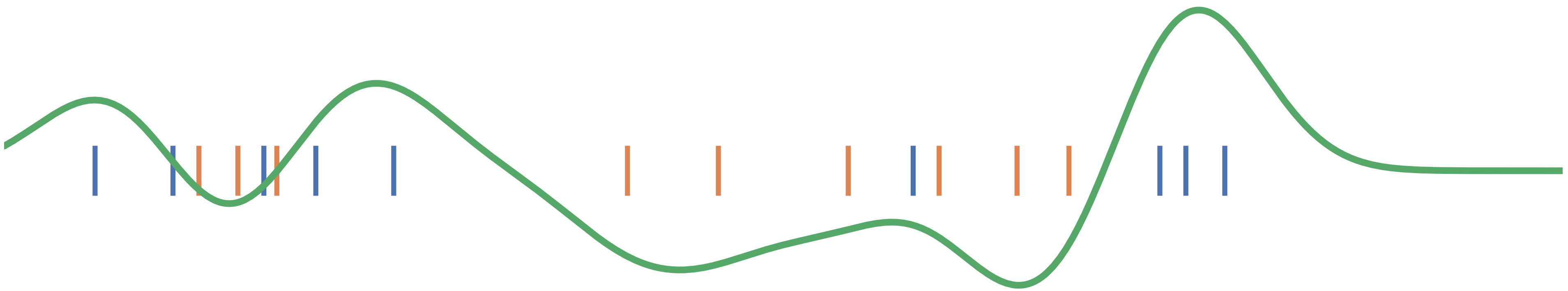


Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{\substack{X, X' \sim \mathbb{P} \\ Y, Y' \sim \mathbb{Q}}}[k(X, X') + k(Y, Y') - 2k(X, Y)]$$

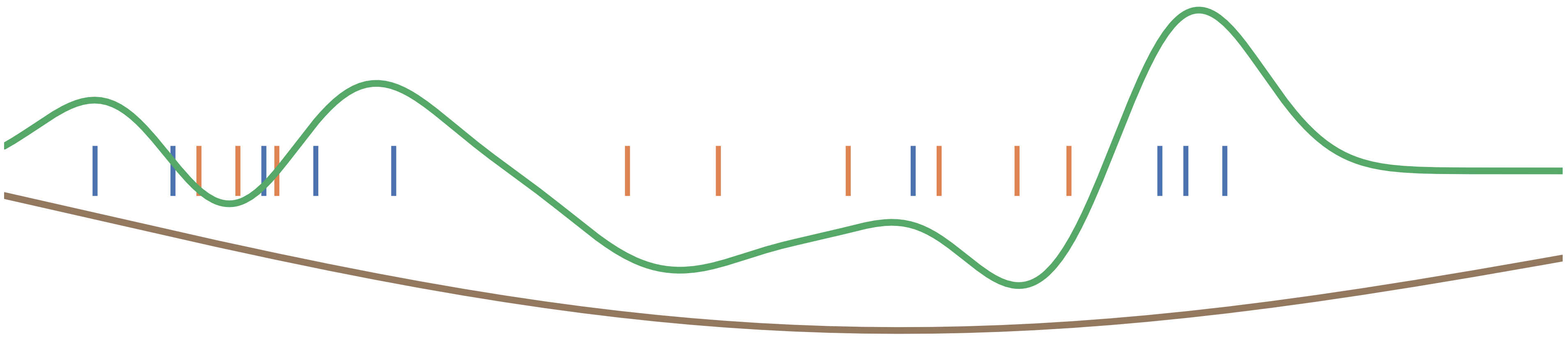


Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}} [f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}} [f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X\sim\mathbb{P}}[k(X,t)] - \mathbb{E}_{Y\sim\mathbb{Q}}[k(Y,t)]$

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{\substack{X, X' \sim \mathbb{P} \\ Y, Y' \sim \mathbb{Q}}} [k(X, X') + k(Y, Y') - 2k(X, Y)]$$

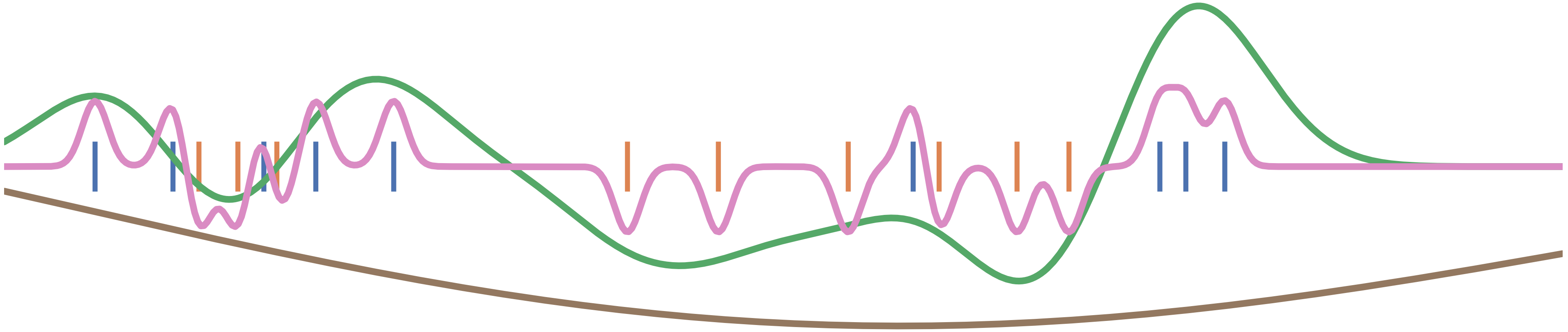


Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \max_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The max is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[k(X, t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[k(Y, t)]$

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{\substack{X, X' \sim \mathbb{P} \\ Y, Y' \sim \mathbb{Q}}}[k(X, X') + k(Y, Y') - 2k(X, Y)]$$



MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - H_0: $n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - H_0: $n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - $H_0: n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - $H_1: \sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test…eventually

MMD-based tests

- If k is *characteristic*, $\operatorname{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\operatorname{MMD}}(X, Y)$
  - H_0: $n\widehat{\operatorname{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\operatorname{MMD}}^2 - \operatorname{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test…eventually
- Need enormous n if the kernel is bad for this problem!

Deep learning and deep kernels

- C2ST-L is basically MMD with $k(x, y) = f(x)f(y)$
  - f is a (learned) deep net – a learned kernel

Deep learning and deep kernels

- C2ST-L is basically MMD with $k(x, y) = f(x) f(y)$
  - f is a (learned) deep net – a learned kernel
- We can generalize some more to **deep kernels**:

$$k_{\psi}(x, y) = \kappa\left(\phi_{\psi}(x), \phi_{\psi}(y)\right)$$

Deep learning and deep kernels

- C2ST-L is basically MMD with $k(x, y) = f(x) f(y)$
  - f is a (learned) deep net – a learned kernel
- We can generalize some more to **deep kernels**:

$$k_{\psi}(x, y) = \kappa\left(\phi_{\psi}(x), \phi_{\psi}(y)\right)$$

  - ϕ is a deep net, maps data points to $\mathbb{R}^D$

Deep learning and deep kernels

- C2ST-L is basically MMD with $k(x, y) = f(x) f(y)$
  - f is a (learned) deep net – a learned kernel
- We can generalize some more to **deep kernels**:

$$k_\psi(x, y) = \kappa\left(\phi_\psi(x), \phi_\psi(y)\right)$$

  - ϕ is a deep net, maps data points to $\mathbb{R}^D$
  - κ is a simple kernel on $\mathbb{R}^D$

Deep learning and deep kernels

- C2ST-L is basically MMD with $k(x, y) = f(x)f(y)$
  - f is a (learned) deep net – a learned kernel
- We can generalize some more to **deep kernels**:

$$k_{\psi}(x, y) = \kappa\left(\phi_{\psi}(x), \phi_{\psi}(y)\right)$$

  - ϕ is a deep net, maps data points to $\mathbb{R}^D$
  - κ is a simple kernel on $\mathbb{R}^D$
  - $\kappa(u, v) = u \cdot v$ gives MMD as $\|\mathbb{E}\,\phi(x) - \mathbb{E}\,\phi(y)\|$

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

MMD, σ_{H_1}, c_α are constants: first term usually dominates

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

  MMD, σ_{H_1}, c_α are constants: first term usually dominates

- Pick k to maximize an estimate of $\mathrm{MMD}^2 / \sigma_{H_1}$

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

MMD, σ_{H_1}, c_α are constants: first term usually dominates

- Pick k to maximize an estimate of $\mathrm{MMD}^2 / \sigma_{H_1}$
- Use $\widehat{\mathrm{MMD}}$ from before, get $\hat{\sigma}_{H_1}$ from U-statistic theory

Optimizing power of MMD tests

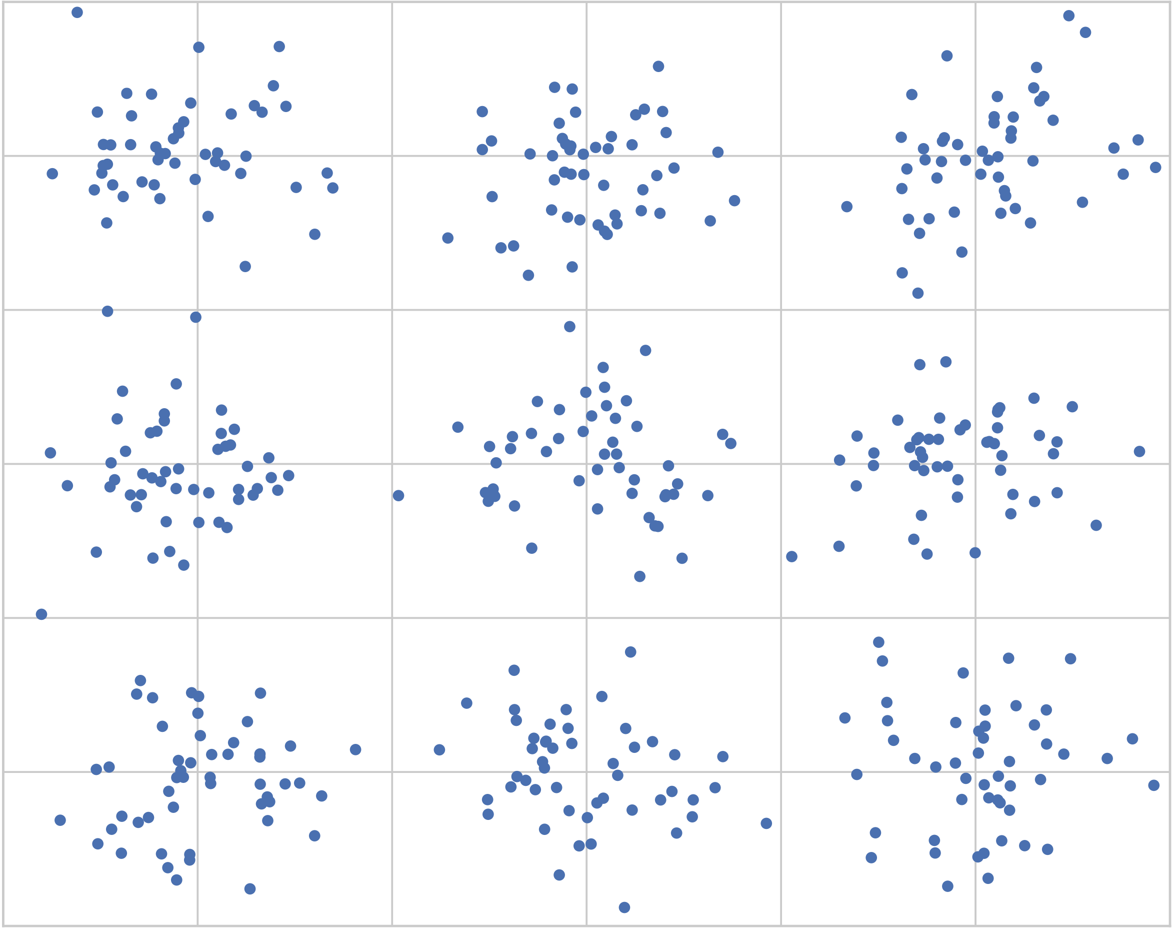
- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

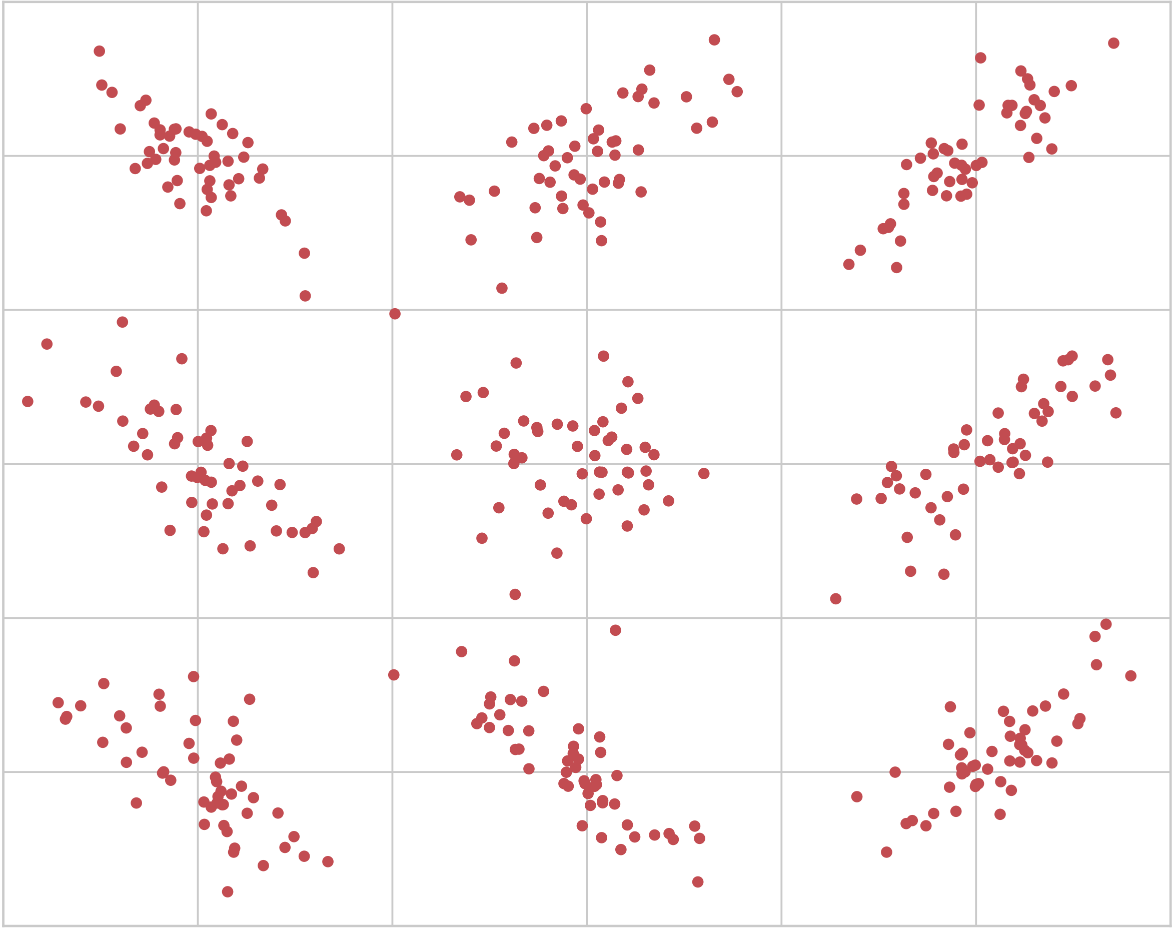
$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

  MMD, σ_{H_1}, c_α are constants: first term usually dominates

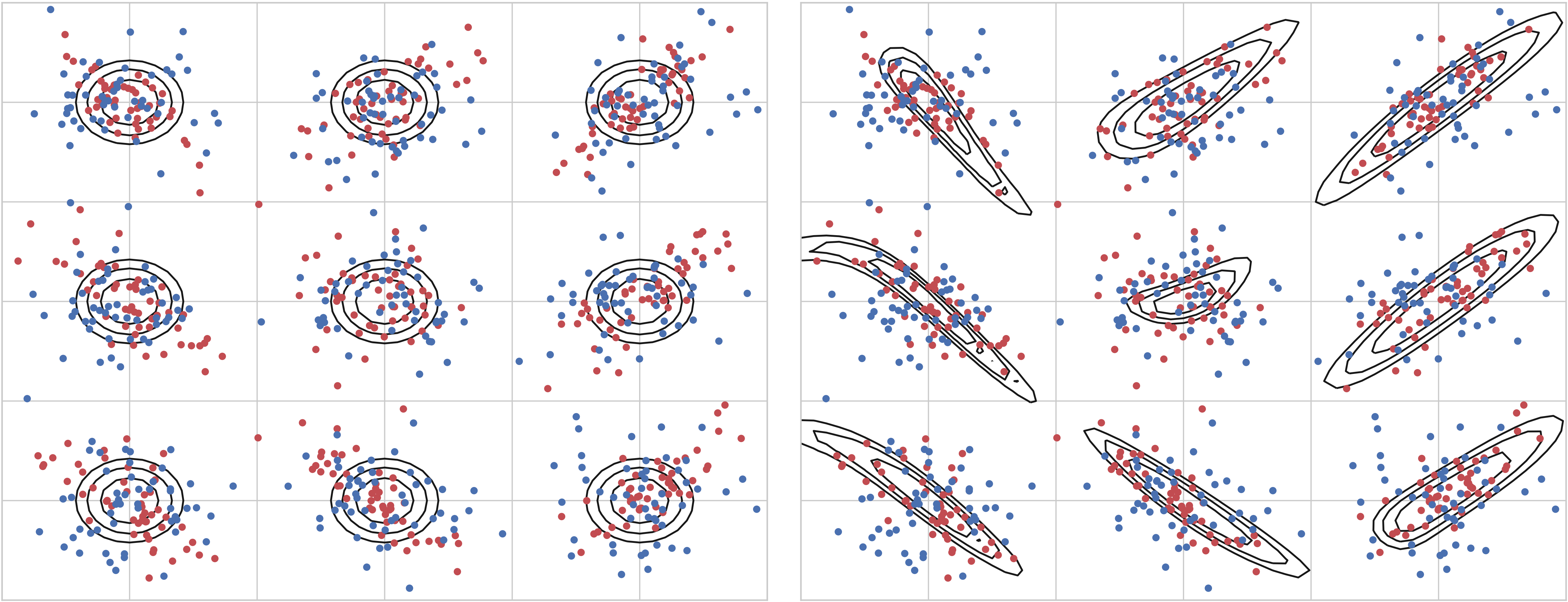
- Pick k to maximize an estimate of $\mathrm{MMD}^2/\sigma_{H_1}$
- Use $\widehat{\mathrm{MMD}}$ from before, get $\hat{\sigma}_{H_1}$ from U-statistic theory
- Can show uniform $\mathcal{O}_P(n^{-\frac{1}{3}})$ convergence of estimator

Blobs dataset

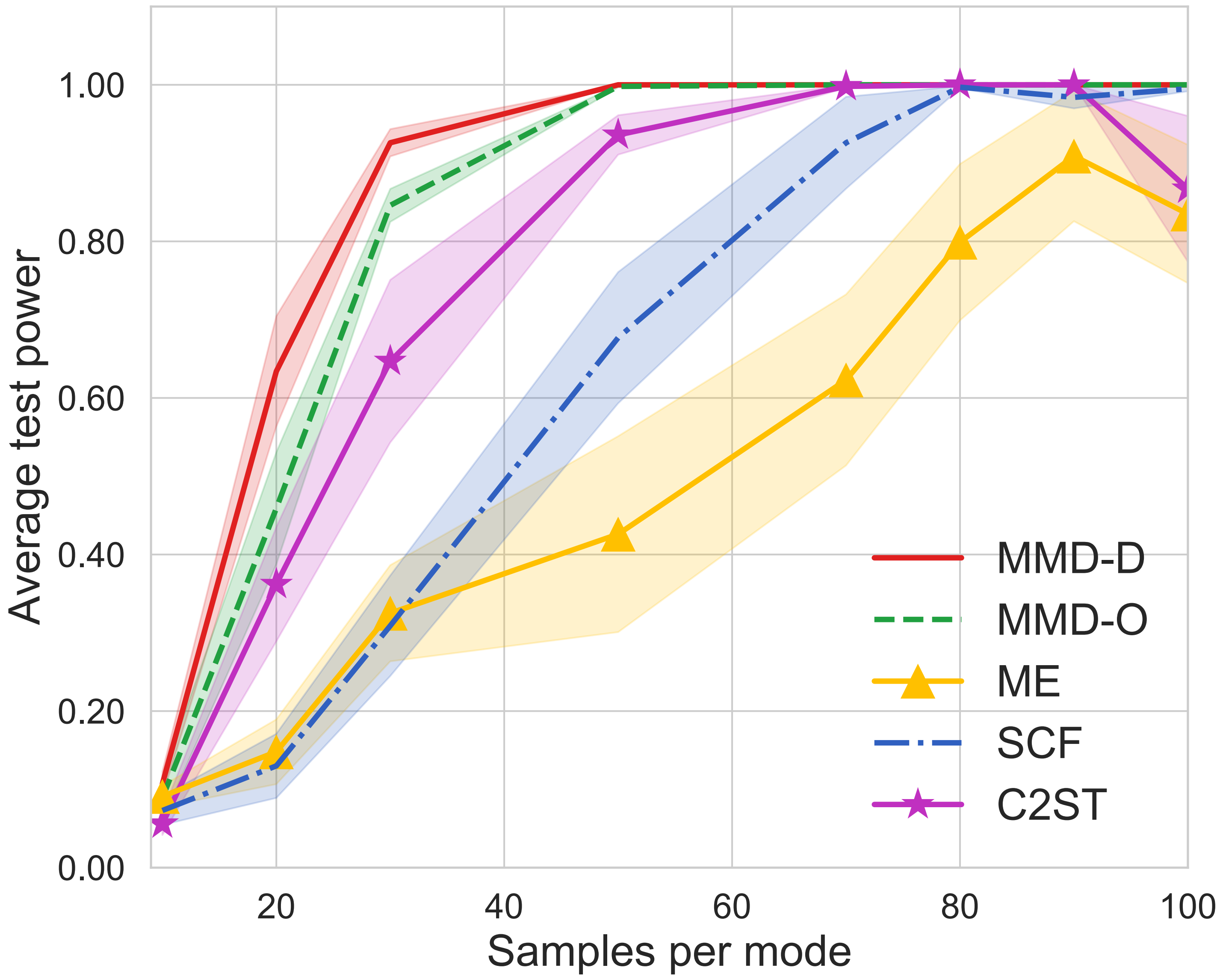




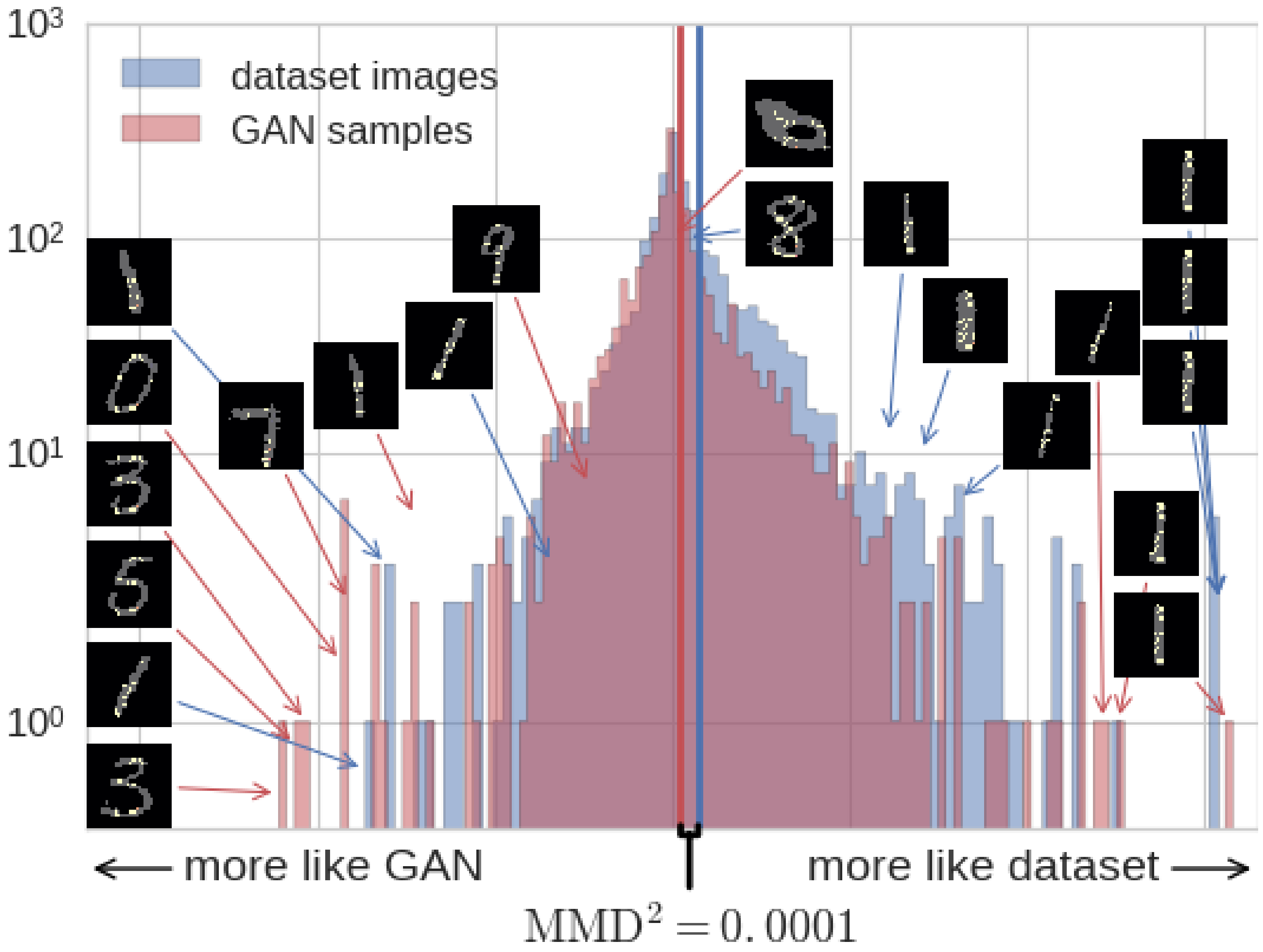
Blobs kernels



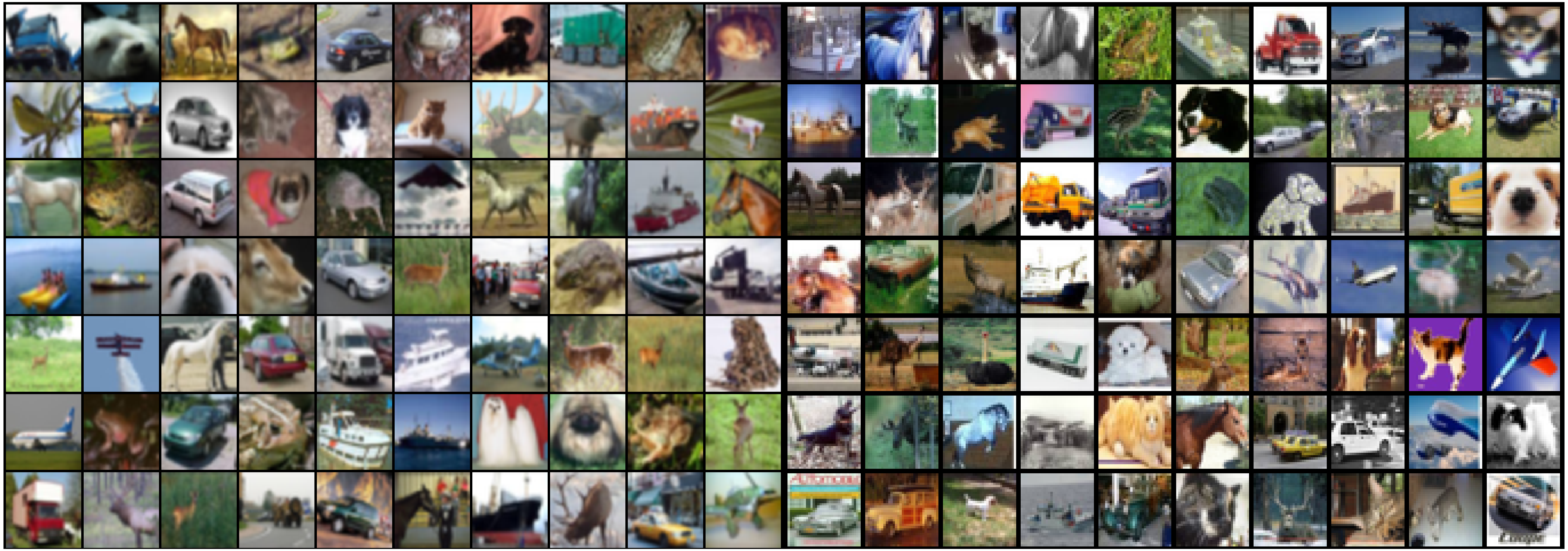
Blobs results



Investigating a GAN on MNIST



CIFAR-10 vs CIFAR-10.1



Train on 1 000, test on 1 031, repeat 10 times. Rejection rates:

ME	SCF	C2ST	MMD-O	MMD-D
0.588	0.171	0.452	0.316	**0.744**

Ablation vs classifier-based tests

	Cross-entropy			Max power		
Dataset	**Sign**	**Lin**	**Ours**	**Sign**	**Lin**	**Ours**
Blobs	0.84	0.94	0.90	–	0.95	0.99
High-d Gauss. mix.	0.47	0.59	0.29	–	0.64	0.66
Higgs	0.26	0.40	0.35	–	0.30	0.40
MNIST vs GAN	0.65	0.71	0.80	–	0.94	1.00

But...

- What if you don't have much data for your testing problem?

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find

But...

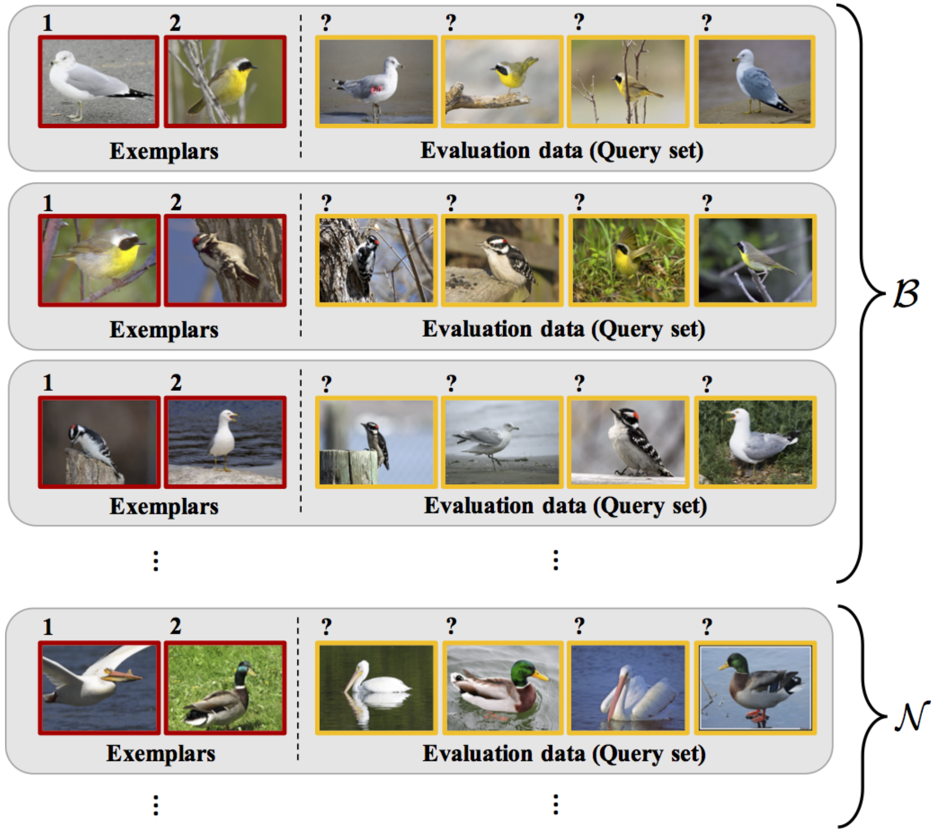
- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find
  - Don't know that ahead of time; can't try more than one

Meta-testing

- One idea: what if we have *related* problems?

Meta-testing

- One idea: what if we have *related* problems?
- Similar setup to meta-learning:



(from Wei+ 2018)

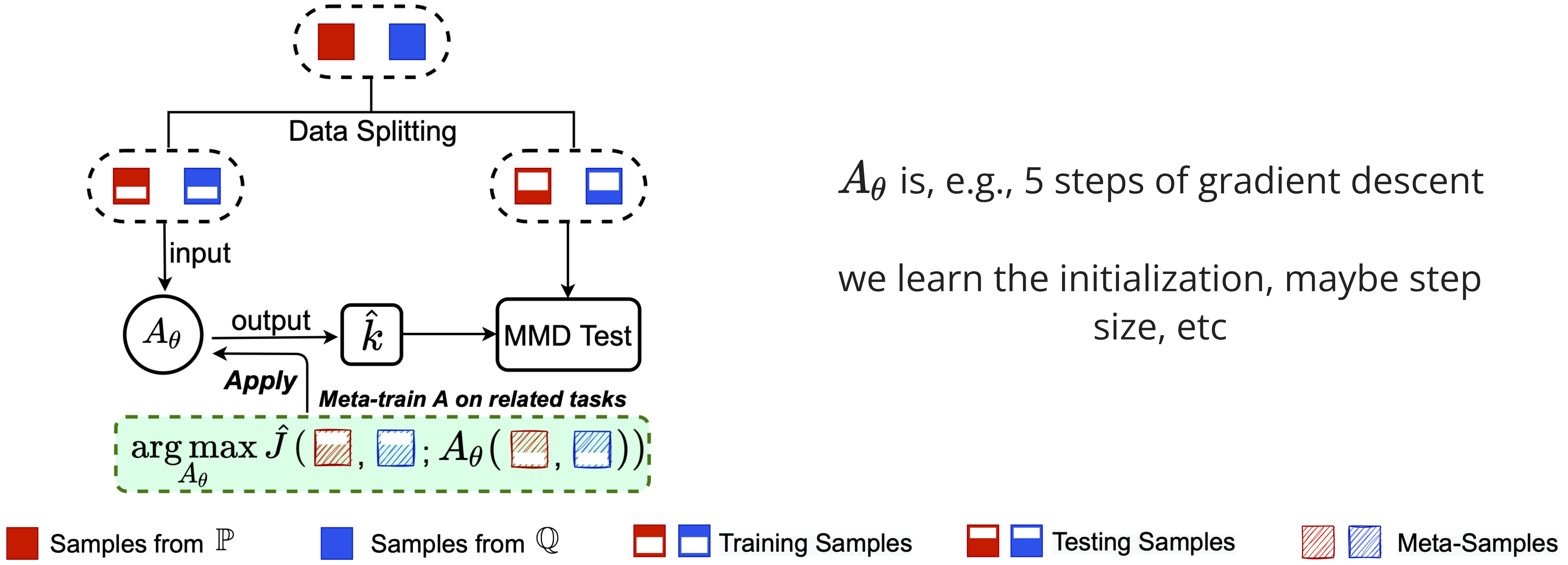
Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031
- Where do we get related data from?

Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031
- Where do we get related data from?
- One option: set up tasks to distinguish classes of CIFAR-10
  - airplane vs automobile, airplane vs bird, ...

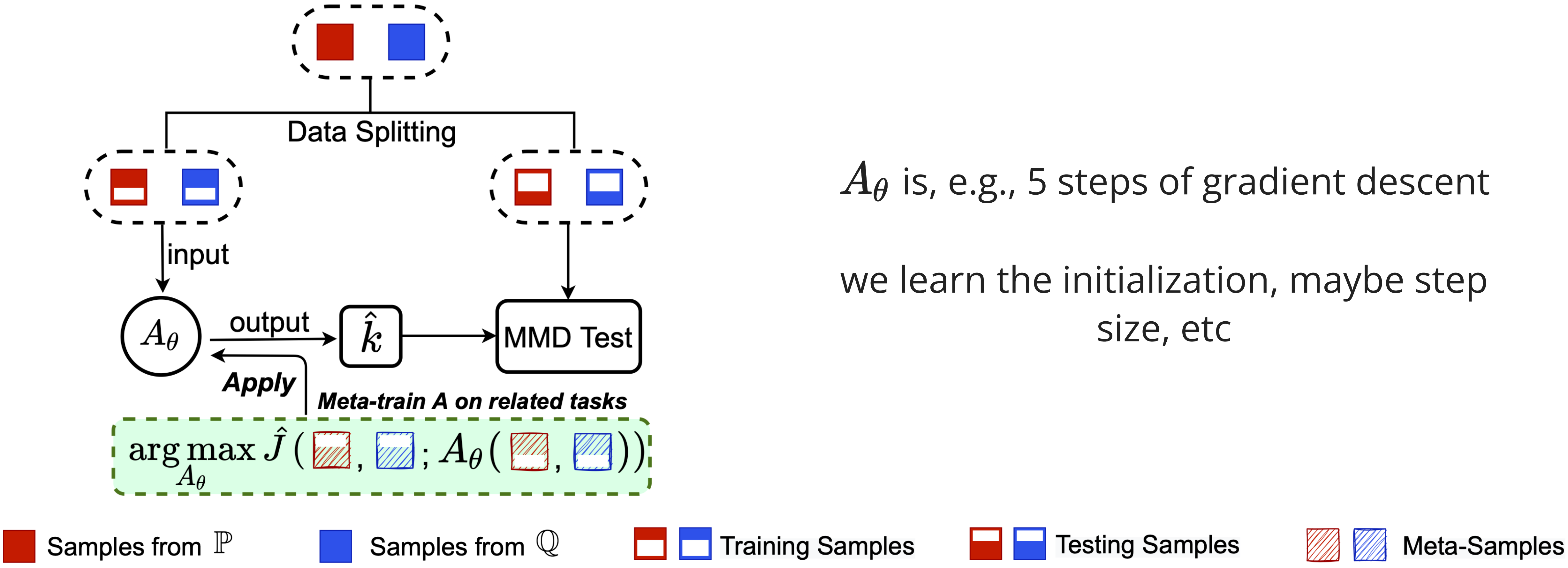
One approach (MAML-like)



A_θ is, e.g., 5 steps of gradient descent

we learn the initialization, maybe step size, etc

One approach (MAML-like)

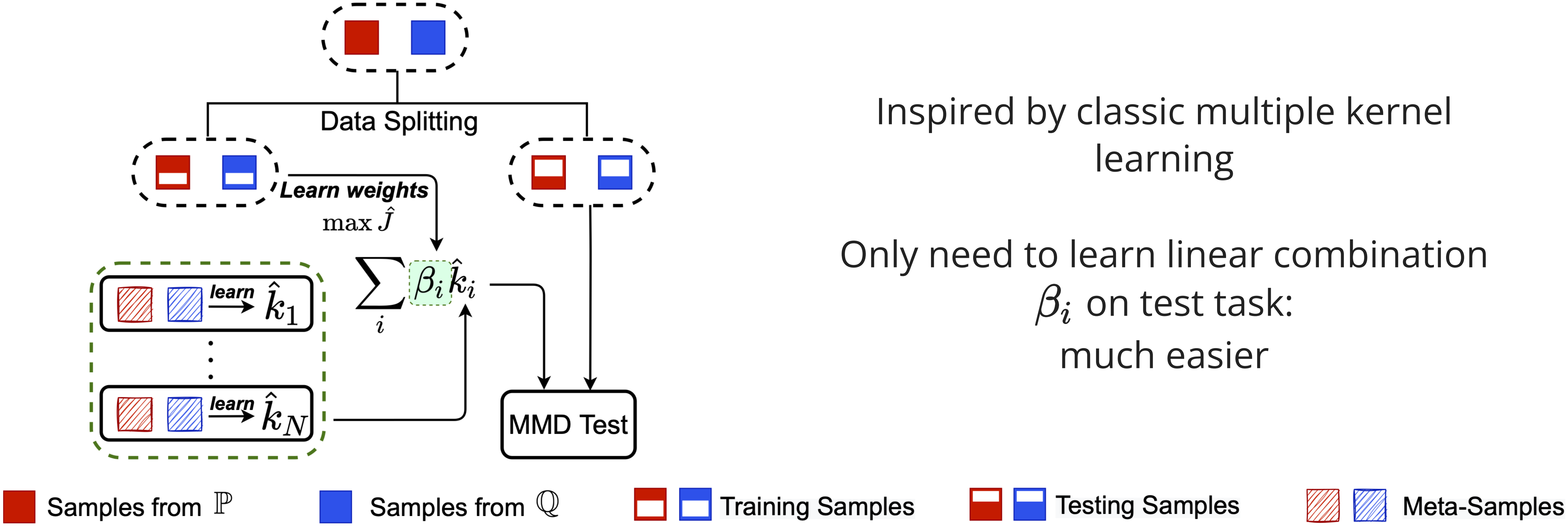


A_θ is, e.g., 5 steps of gradient descent

we learn the initialization, maybe step size, etc

This works, but not as well as we'd hoped...
Initialization might work okay on everything, not really adapt

Another approach: Meta-MKL



Inspired by classic multiple kernel learning

Only need to learn linear combination β_i on test task: much easier

Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😐
- But multiplier is *much* better:
  based on number of meta-training tasks, not on network size

Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😐
- But multiplier is *much* better:
  based on number of meta-training tasks, not on network size
- Coarse analysis: assumes one meta-tasks is “related” enough
  - We compete with picking the single best related kernel
  - Haven't analyzed meaningfully combining related kernels (yet!)

Results on CIFAR-10.1

Methods	$m_{tr} = 100$			$m_{tr} = 200$		
	$m_{te} = 200$	$m_{te} = 500$	$m_{te} = 900$	$m_{te} = 200$	$m_{te} = 500$	$m_{te} = 900$
ME	$0.084_{\pm 0.009}$	$0.096_{\pm 0.016}$	$0.160_{\pm 0.035}$	$0.104_{\pm 0.013}$	$0.202_{\pm 0.020}$	$0.326_{\pm 0.039}$
SCF	$0.047_{\pm 0.013}$	$0.037_{\pm 0.011}$	$0.047_{\pm 0.015}$	$0.026_{\pm 0.009}$	$0.018_{\pm 0.006}$	$0.026_{\pm 0.012}$
C2ST-S	$0.059_{\pm 0.009}$	$0.062_{\pm 0.007}$	$0.059_{\pm 0.007}$	$0.052_{\pm 0.011}$	$0.054_{\pm 0.011}$	$0.057_{\pm 0.008}$
C2ST-L	$0.064_{\pm 0.009}$	$0.064_{\pm 0.006}$	$0.063_{\pm 0.007}$	$0.075_{\pm 0.014}$	$0.066_{\pm 0.011}$	$0.067_{\pm 0.008}$
MMD-O	$0.091_{\pm 0.011}$	$0.141_{\pm 0.009}$	$0.279_{\pm 0.018}$	$0.084_{\pm 0.007}$	$0.160_{\pm 0.011}$	$0.319_{\pm 0.020}$
MMD-D	$0.104_{\pm 0.007}$	$0.222_{\pm 0.020}$	$0.418_{\pm 0.046}$	$0.117_{\pm 0.013}$	$0.226_{\pm 0.021}$	$0.444_{\pm 0.037}$
AGT-KL	$0.170_{\pm 0.032}$	$0.457_{\pm 0.052}$	$0.765_{\pm 0.045}$	$0.152_{\pm 0.023}$	$0.463_{\pm 0.060}$	$0.778_{\pm 0.050}$
Meta-KL	$0.245_{\pm 0.010}$	$0.671_{\pm 0.026}$	$0.959_{\pm 0.013}$	$0.226_{\pm 0.015}$	$0.668_{\pm 0.032}$	$0.972_{\pm 0.006}$
Meta-MKL	$\mathbf{0.277}_{\pm 0.016}$	$\mathbf{0.728}_{\pm 0.020}$	$\mathbf{0.973}_{\pm 0.008}$	$\mathbf{0.255}_{\pm 0.020}$	$\mathbf{0.724}_{\pm 0.026}$	$\mathbf{0.993}_{\pm 0.003}$

But...

- Sometimes we already know there are differences we don't care about

But...

- Sometimes we already know there are differences we don't care about
  - In the MNIST GAN criticism, first just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

But...

- Sometimes we already know there are differences we don't care about
  - In the MNIST GAN criticism, first just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$,
  but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?

But...

- Sometimes we already know there are differences we don't care about
  - In the MNIST GAN criticism, first just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has
- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$, but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?
- Also useful for **fair representation learning**

But...

- Sometimes we already know there are differences we don't care about
  - In the MNIST GAN criticism, first just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has
- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$, but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?
- Also useful for **fair representation learning**
  - e.g. can distinguish “creditworthy” vs not, but **can't** distinguish by race

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\text{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\text{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\mathrm{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$
  - Can estimate c_α inside the optimization

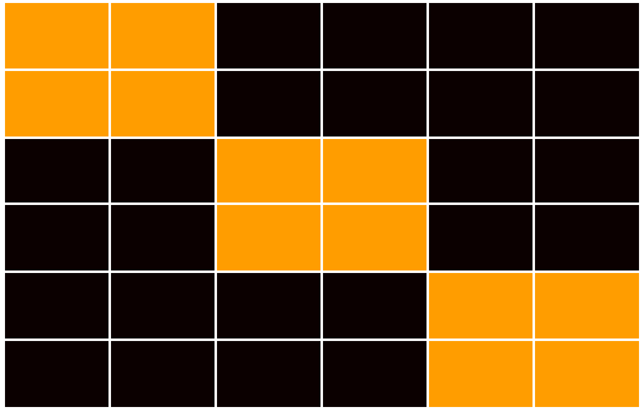
High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\mathrm{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$
  - Can estimate c_α inside the optimization
  - Better, but tends to “stall out” in minimizing ρ_k^s

Block estimator [Zaremba+ NeurIPS-13]

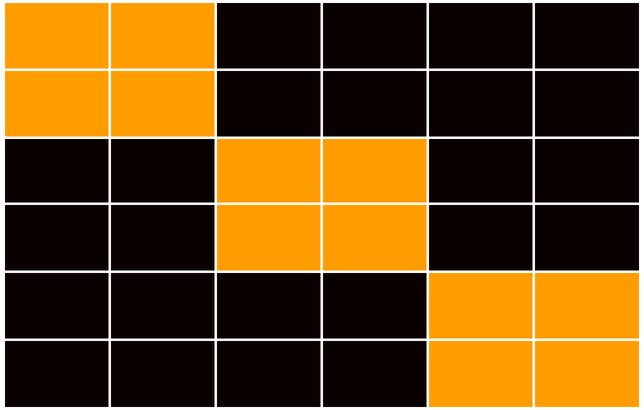
- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate

Block estimator [Zaremba+ NeurIPS-13]

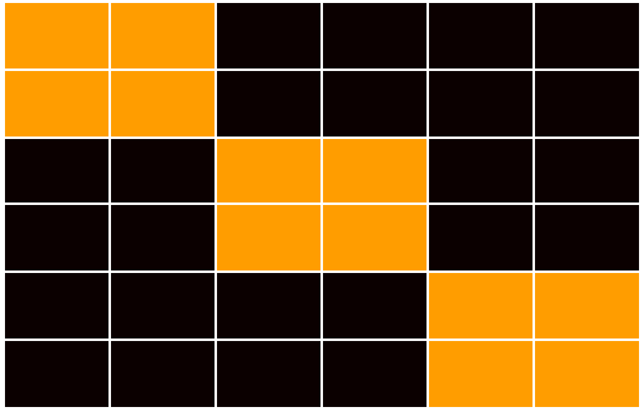
- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics

Block estimator [Zaremba+ NeurIPS-13]

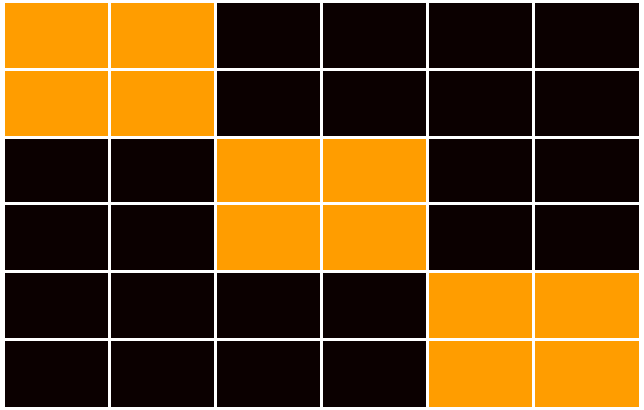
- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks

Block estimator [Zaremba+ NeurIPS-13]

- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks
- Power is $\rho = \Phi\left(\sqrt{bB}\frac{\mathrm{MMD}^2}{\sigma_{H_1}^2} - \Phi^{-1}(1-\alpha)\right)$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size

MMD-B-Fair

- Choose k as $\min_k \rho^s_k - \rho^t_k$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \left(\max_\kappa \rho^s_{\kappa\circ\phi} - \max_\kappa \rho^t_{\kappa\circ\phi} \right)$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \left(\max_\kappa \rho^s_{\kappa \circ \phi} - \max_\kappa \rho^t_{\kappa \circ \phi} \right)$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \left(\max_\kappa \rho^s_{\kappa\circ\phi} - \max_\kappa \rho^t_{\kappa\circ\phi} \right)$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - κ could be deep itself, with adversarial optimization

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \left(\max_\kappa \rho_{\kappa\circ\phi}^s - \max_\kappa \rho_{\kappa\circ\phi}^t \right)$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - κ could be deep itself, with adversarial optimization
  - For now, just Gaussians with different lengthscales

Adult Data Set

Download: Data Folder, Data Set Description

Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

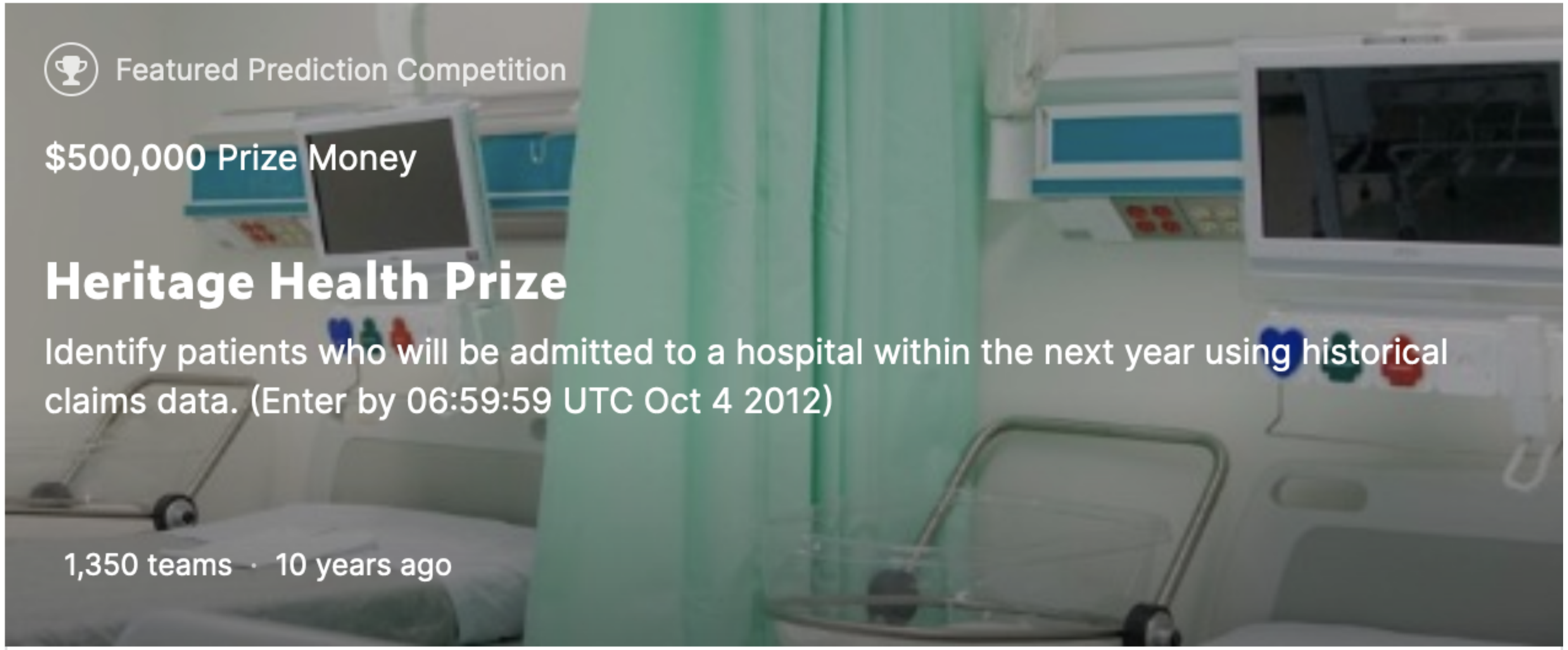
Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	2390574

Adult Data Set

Download: Data Folder, Data Set Description

Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	2390574



Learned representations

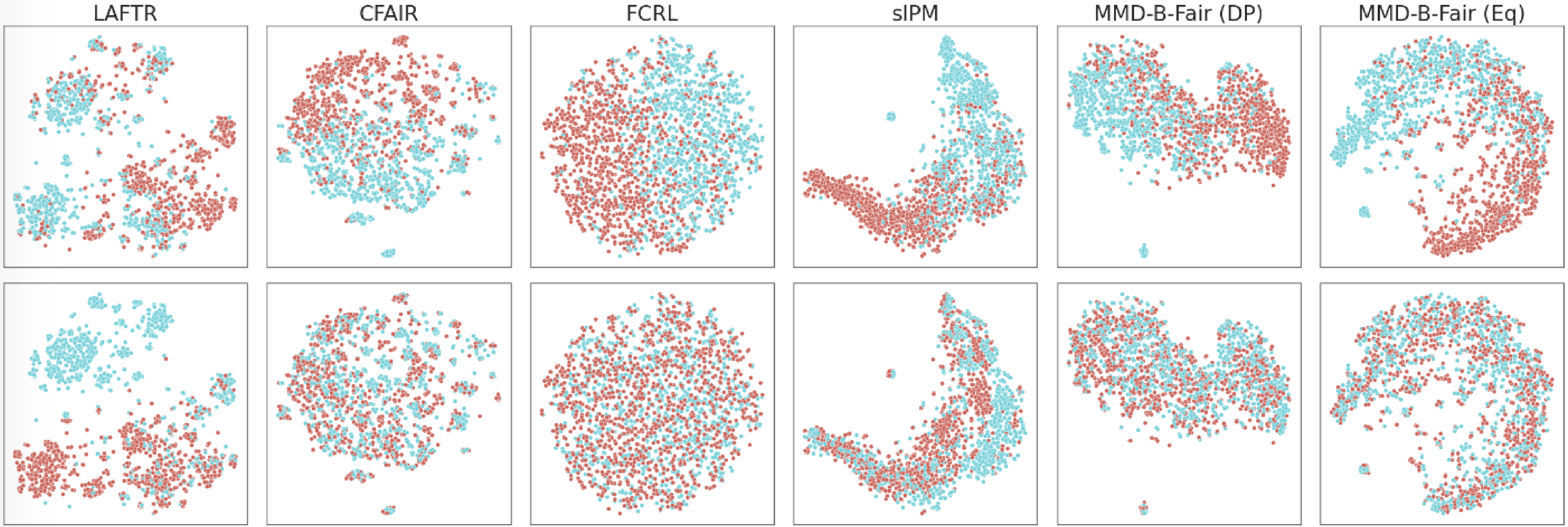


Figure 4: t-SNE visualizations of Adult representations, colored by target attribute (top) and sensitive attribute (bottom).

Quality of transfer learning

Transfer Label		LAFTR	CFAIR	FCRL	sIPM	MMD-B-Fair (DP)	MMD-B-Fair (Eq)
MSC2a3	acc	57.2	62.5	58.0	**72.8**	71.3	70.3
	DP	52.3	65.1	**99.2**	69.3	72.2	84.5
	Eq	57.4	70.1	**98.0**	69.9	71.8	86.6
METAB3	acc	**72.9**	72.2	53.9	72.4	70.7	69.4
	DP	52.3	65.1	**97.7**	54.5	65.6	82.1
	Eq	61.3	77.1	**97.6**	63.4	74.6	92.1
ARTHSPHIN	acc	66.4	65.9	59.3	**70.6**	67.5	67.8
	DP	52.3	65.1	**98.0**	74.6	83.0	87.7
	Eq	54.9	70.1	**98.1**	76.7	84.9	90.0
NEUMENT	acc	64.4	61.9	60.1	**68.0**	67.1	67.3
	DP	52.3	65.1	**99.1**	72.9	86.8	94.5
	Eq	54.9	69.7	**97.5**	73.2	86.7	95.4
MISCHRT	acc	71.0	67.3	69.3	**73.5**	73.0	72.5
	DP	52.3	65.1	**98.6**	85.0	87.2	96.4
	Eq	59.4	79.0	**98.2**	88.5	88.6	97.5

Table 1: Using Heritage Health representations to predict various downstream tasks. **Red** marks the best result per row, blue second-best, and green third-best.

- Check if your data is different than it used to be!
- Pretty good method: train a classifier, check how accurate
- More powerful: use an optimized kernel method

A good takeaway

Combining a deep architecture with a kernel machine that takes the higher-level learned representation as input can be quite powerful.

— Y. Bengio & Y. LeCun (2007), "Scaling Learning Algorithms towards AI"