

CMPT 733

Anomaly Detection

SLIDES BY:

JIANNAN WANG

<https://www.cs.sfu.ca/~jnwang/>

Outline

A Brief Introduction of Anomaly Detection Application: Network Intrusion Detection

What is Anomaly Detection?

Definition from dictionary

a·nom·a·ly

/ə'naməlē/

noun

1. something that deviates from what is standard, normal, or expected.

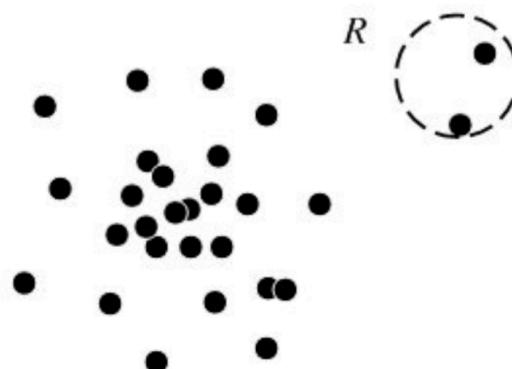


Also known as Outlier Detection

Anomaly Categories (I)

Global Anomaly

- A data point is considered anomalous with respect to **the rest of data**
- Example: There is a person whose age is 110

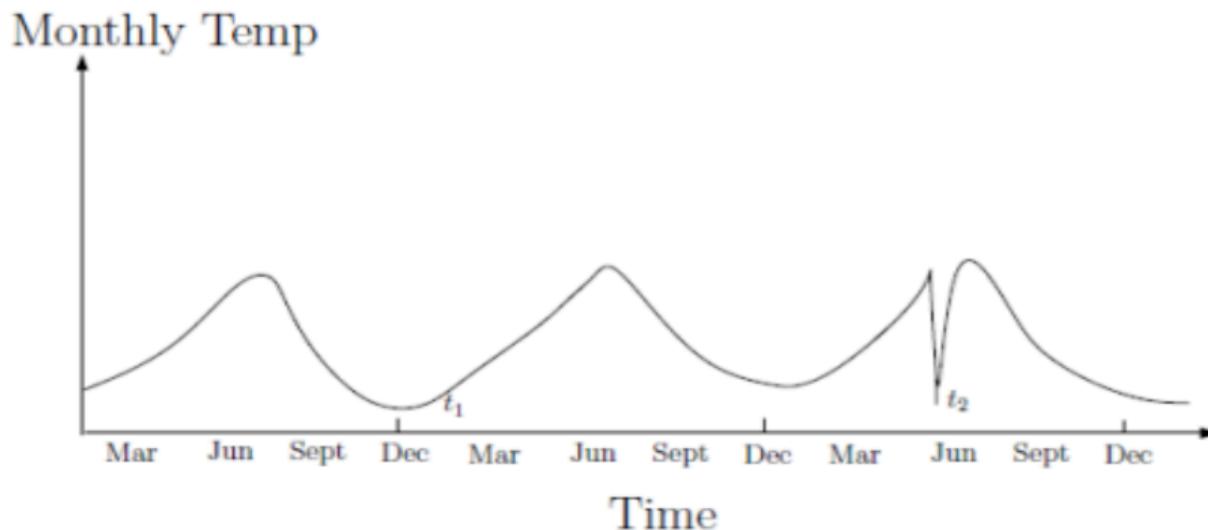


The objects in region R are outliers.

Anomaly Categories (II)

Context Anomaly

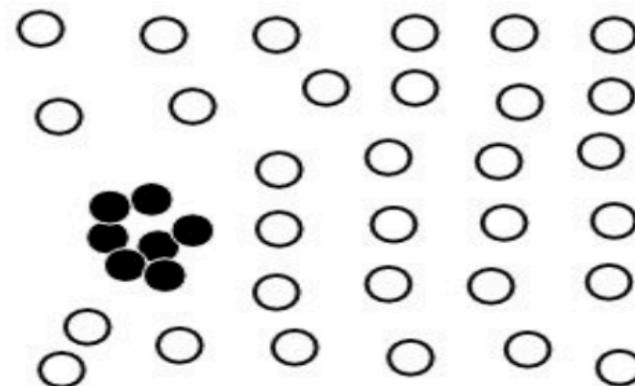
- A data point is considered anomalous with respect to **a specific context**
- Example: There is a student in our class whose age is 10.



Anomaly Categories (III)

Collective Anomaly

- A subset of data points as a whole deviates significantly from the entire dataset
- Example: An order may have some delay to be processed. But, what if 1000 orders are processed with delay?



The black objects form a collective outlier.

Real-world Applications

Fraud Detection

Medical Care

Public Safety and Security

Network Intrusion

Challenges

Modeling normal objects and anomalies effectively:

- Hard to enumerate all possible normal behaviors
- The border between normal objects and anomalies can be gray area

Application specific anomaly detection:

- Hard to develop general purpose anomaly detection tools

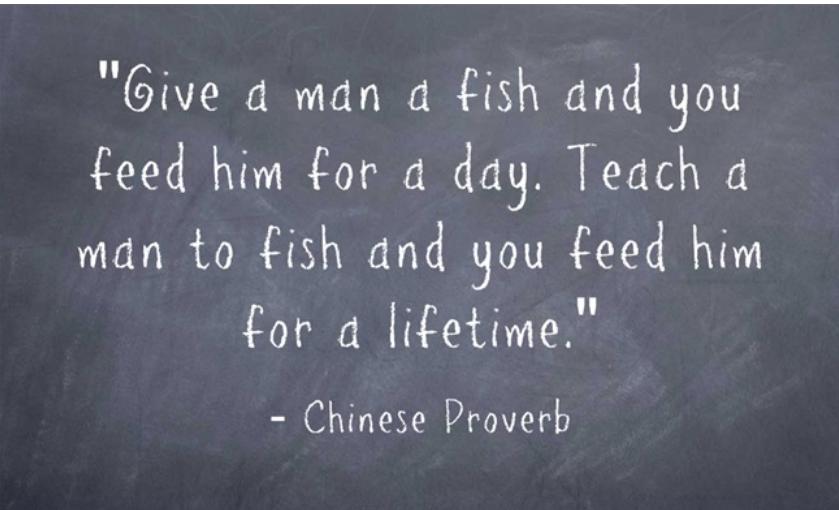
Understandability:

- Not only detect the anomalies, but also understand why they are anomalies

Outline

Recap of Anomaly Detection

Application: Network Intrusion Detection



"Give a man a fish and you feed him for a day. Teach a man to fish and you feed him for a lifetime."

- Chinese Proverb

Teach you a network-intrusion solution
vs.
Teach you how to come up with this solution

Network Intrusion



*“Our web servers got attacked yesterday.
I don’t want it happen again.
Please build a system to address it!”*

TO DO Lists:

1. Find related datasets (e.g., `/var/log/apache2/access.log`)
2. Figure out how to detect attacks (anomalies) ← **Key Problem**
3. Trigger an alert when an attack is detected (e.g., send an email)

How to come up with a solution?

1. Doing a survey on related work

Anomaly Detection Methods

Survey Paper

[Anomaly detection: A survey](#)

[V Chandola, A Banerjee, V Kumar](#) - ACM computing surveys (CSUR), 2009 - dl.acm.org

Abstract Anomaly detection is an important problem that has been researched within diverse research areas and application domains. Many anomaly detection techniques have been specifically developed for certain application domains, while others are more generic. This

 99 Cited by 4705 Related articles All 36 versions

[Intrusion detection: A survey](#)

[F Sabahi, A Movaghar](#) - Systems and Networks ..., 2008 - ieeexplore.ieee.org

... presents a taxonomy of intrusion **detection** systems that is then used to **survey** and classify ... This method works by using the definition "**anomalies** are not normal ... There are many **anomaly detection** that proposed algorithms with differences in the information used for analysis and ...

 99 Cited by 117 Related articles All 4 versions

1. Supervised Learning (e.g., Sentiment Analysis)
2. Unsupervised Learning (e.g., Find the top-10 hot topics in twitter)

Why is unsupervised learning more common?

No need to label data

- Labeling is a tedious and expensive process

Able to identify “unknown unknowns”

- Not only detect a known attack pattern (e.g., red apple)
- but also detect an unknown attack pattern (e.g., watermelon)



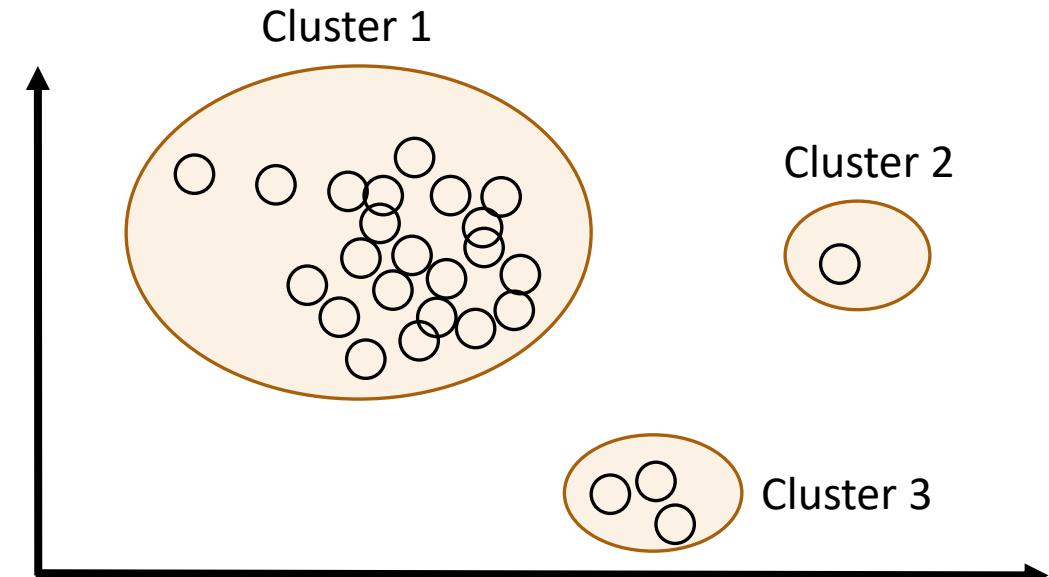
How to come up with a solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach

Clustering-based

Basic Idea

- Cluster data points into groups.
- Decide which points are anomalies:
 - Points in small clusters
 - Using distance to the closest cluster



How to come up with a solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm

K-Means

Iterative Algorithm

- This is the initial motivation for creating Spark

Algorithm Overview

1. Picking up K random points as cluster centers
2. Assigning each point to the closest center
3. Updating cluster centers accordingly
4. Repeat Steps 2 and 3 until some termination conditions are met

How to optimize?

1. RDD.cache()
2. K-Means++ Initialization

How to come up with a solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm
4. Selecting and transforming features

Feature Extraction

Raw Data

```
1 in24.inetnebr.com - - [01/Aug/1995:00:00:01 -0400] "GET /shuttle/missions/sts-68/news/sts-68-mcc-05  
2 uplherc.upl.com - - [01/Aug/1995:00:00:07 -0400] "GET / HTTP/1.0" 304 0  
3 uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/ksclogo-medium.gif HTTP/1.0" 304 0  
4 uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/MOSAIC-logosmall.gif HTTP/1.0" 304 0
```

Turning Raw Data into Connection Data

- A connection is a sequence of HTTP requests starting and ending at some well defined times

Turning Connection Data into Feature Vectors

- Requiring a fair bit of domain knowledge
- Asking yourself how to distinguish attacks from normal connections (e.g., number of failed login attempts, duration of the connection)

Feature Scaling

Feature Vector

- e.g., [http, BC, 0, 105, 146, 0, ..., 0.00, 0.00]
 - Categorical feature
 - Numerical feature

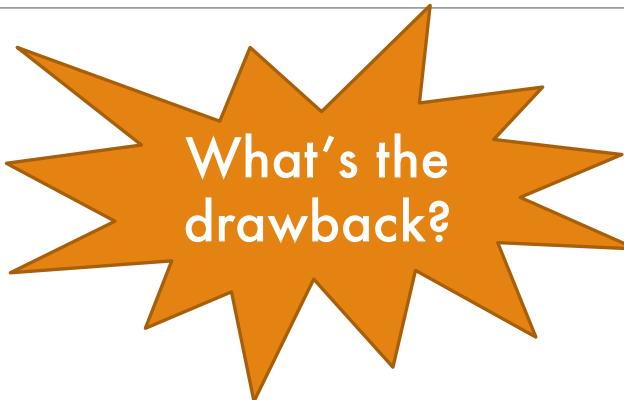
Will this feature vector work for KMeans?

- What's the distance between "http" and "ftp"?
- The distance between two feature vectors will be dominated by the features with a broad range of values (e.g., the 4th feature)

Categorical Features → Numerical Features

Naïve solution

- http → 0
- ftp → 1
- ssh → 2



$\text{Distance}(\text{"http"}, \text{"ssh"}) > \text{Distance}(\text{"http"}, \text{"ftp"})$

One-hot encoding

- http → [1,0,0]
- ftp → [0,1,0]
- ssh → [0,0,1]

Scaling Numerical Features

1. Rescaling

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

2. Standardization

$$x' = \frac{x - \bar{x}}{\sigma}$$

3. Scaling to unit length

$$x' = \frac{x}{\|x\|}$$



It depends on requirements:

- Hard bounds: (1), (3)
- PCA: (2)
- Tree-based models: no need

Feature Selection

What? and Why?

Data are often in the form of a table

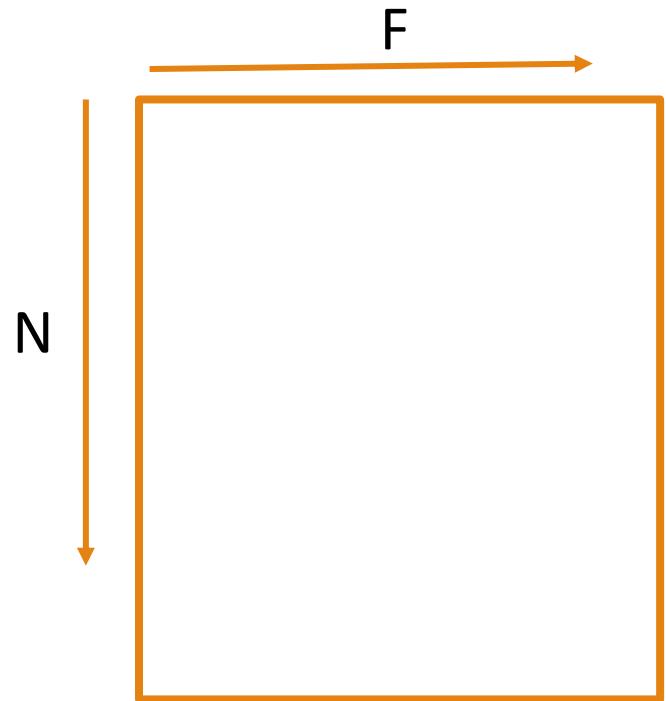
- N: # of training examples (e.g., tweets, images)
- F: # of features (e.g., bag of words, color histogram)

Feature Selection

- Selecting a subset of features for use in model construction.

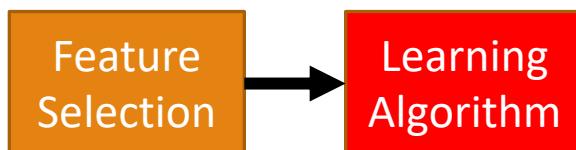
What's bad about "Big F"?

- Slow (training time)
- Inaccurate (due to overfitting and the curse of dimensionality)
- Hard to interpret models



How?

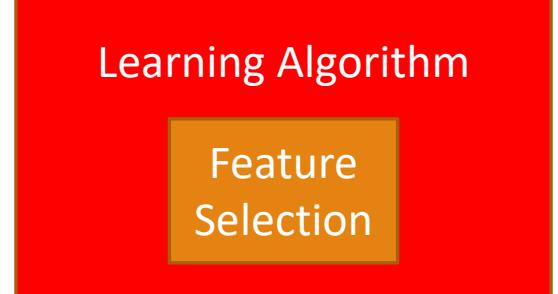
Filter Method



Wrapper Method



Embedded Method



Filter Method

Basic Idea

- Assign a score to each feature
- Filter out useless features based on the scores

Many popular scores [see Yang and Pederson '97]

- Classification: Chi-squared, information gain, document frequency
- Regression: correlation, mutual information

Wrapper Method

Basic Idea

- Evaluate subsets of features
- Select the best subset

How to evaluate a subset of features?

- Test Error (estimated by cross validation)

How to find the best subset?

- Greedy Algorithms (e.g., forward selection, backward elimination)

Embedded Method

Basic Idea

- Modify a learning algorithm such that it can automatically penalize useless features

Lasso Regression

$$\operatorname{argmin}_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$


Penalize useless features

Comparisons

Filter Method

Efficient

Robust to overfitting

Fails to capture relationships between features

Wrapper Method

Capture relationships between features

Inefficient

May suffer from overfitting

Embedded Method

Combine the advantages of the above methods

Specific to a learning algorithm

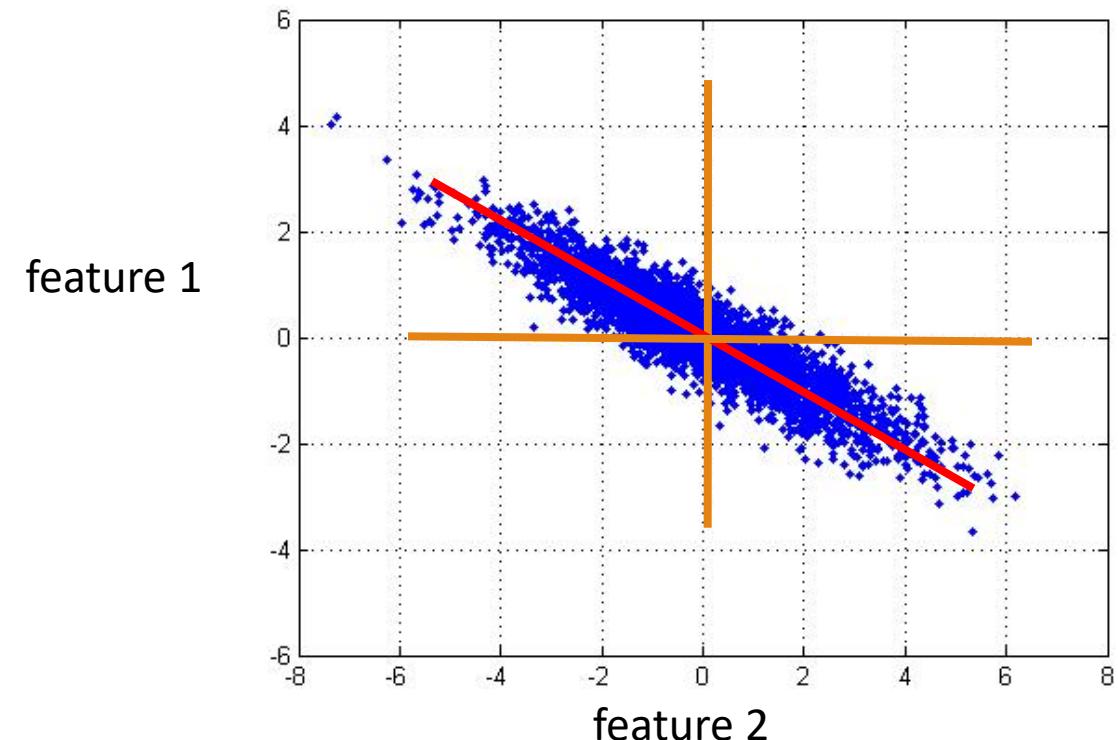
Dimensionality Reduction

Feature Selection

- New features have to be a subset of old features

Feature Transformation (e.g., PCA)

- New features may NOT be a subset of old features



How to come up with a solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm
4. Selecting and transforming features
5. Parameter tuning and evaluation

Parameter Tuning and Evaluation

Evaluation

- Ground-truth Label?
- Evaluation Metric?

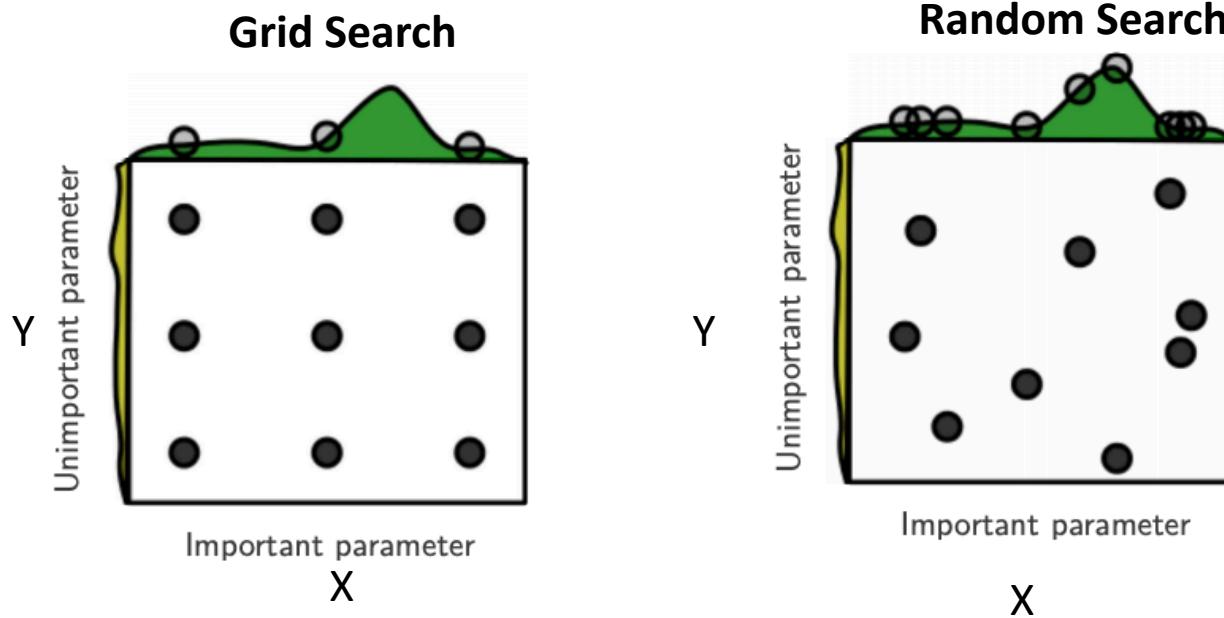
Parameter Tuning

- Grid Search
- Random Search
- Bayesian Optimization

Grid Search & Random Search

x: # of working hours (1, 2, ..., 12)
y: # of sleeping hours (1, 2, ..., 12)

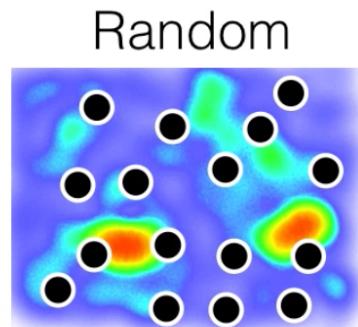
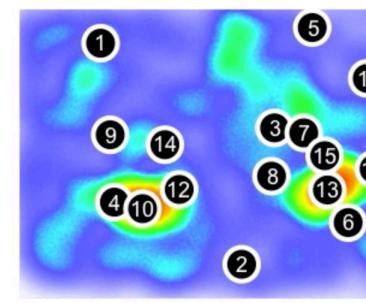
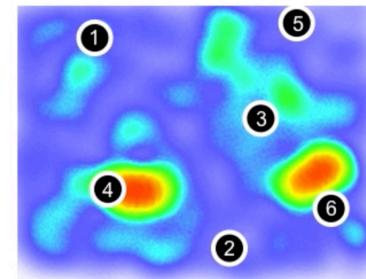
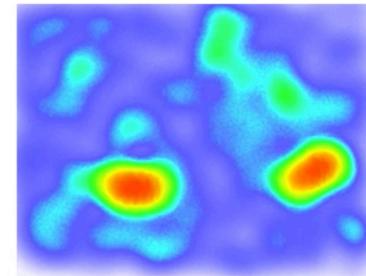
$$\text{Income}(x, y) = \text{Work}(x) + \text{Sleep}(y)$$



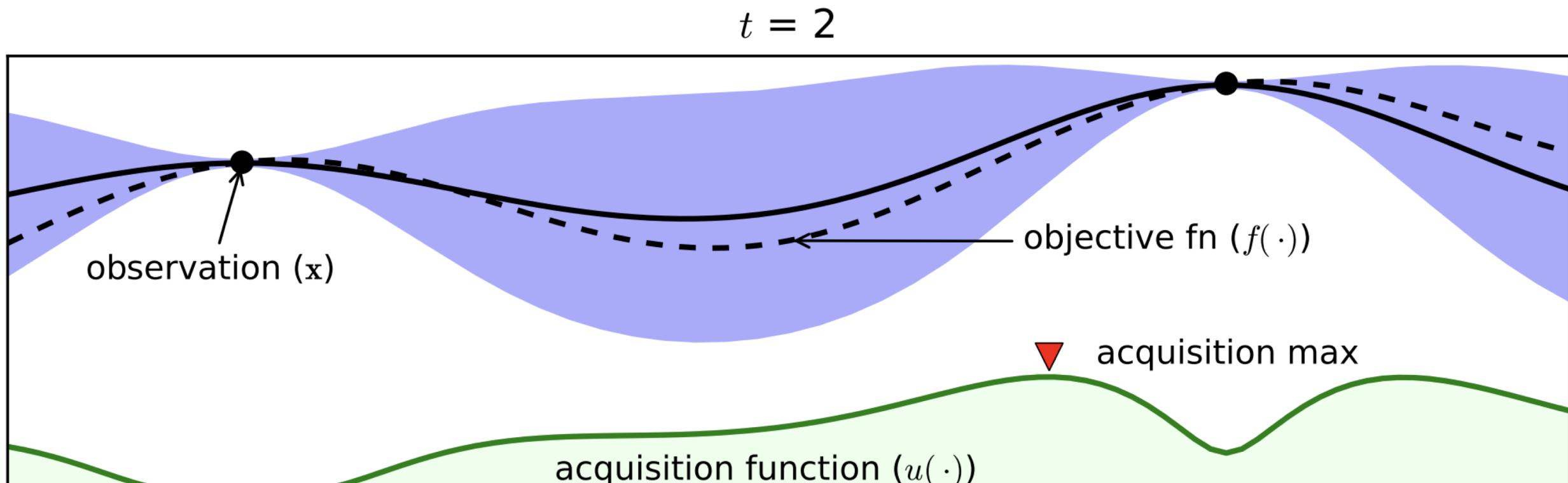
Bayesian Optimization

❖ Intuition

- Want to find the peak point of objective function (eg. accuracy as a function of parameters)
- Fit a statistical model to the observed points and pick next best point where we believe the maximum will be
- Next point is determined by an acquisition function that trades off exploration(objective) and exploitation(uncertainty)



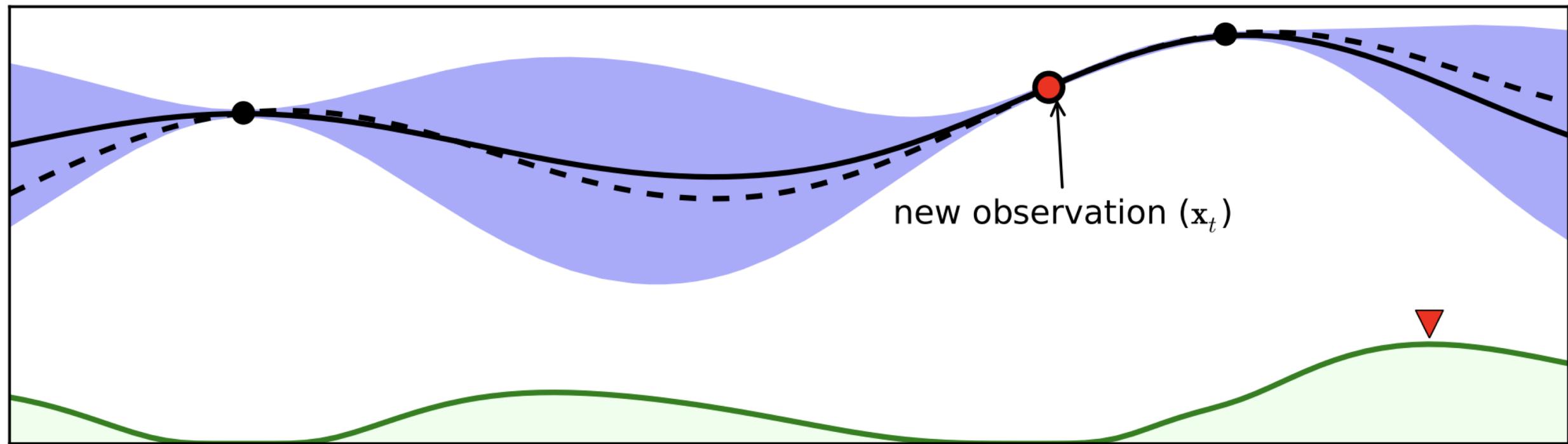
Example



Find next point that **max** the acquisition function

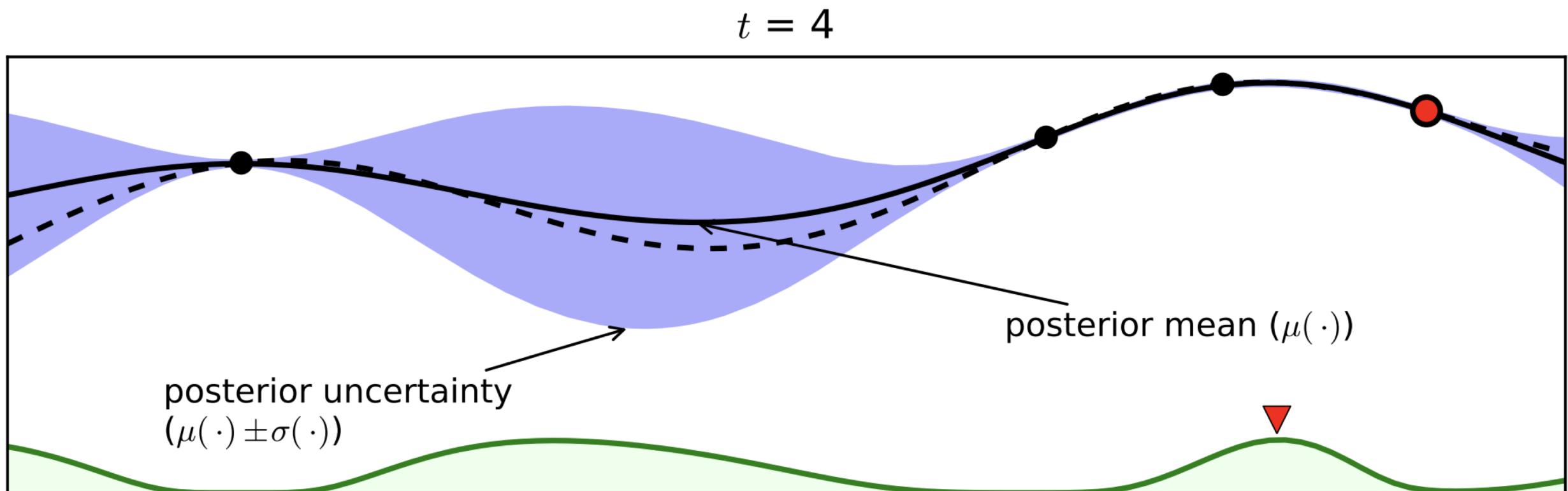
Example

$t = 3$



Evaluate at the new observation point x_t and update posterior
Update acquisition function from new posterior and find the next best point

Example



How to come up the solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm
4. Selecting and transforming features
5. Tuning parameters and evaluation
6. If not satisfied, go back to previous steps

How to come up the solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm
4. Selecting and transforming features
5. Tuning Parameters and evaluation
6. If not satisfied, go back to previous steps
7. Deploying your model in production

Model Serving

Very Important Topic

- Model has to reflect to the latest data updates.
- Kmeans → Streaming Kmeans
- Predictions have to be made in real-time.
 - [Deploy a Model in Amazon SageMaker](#)
 - [TensorFlow Serving](#)
 - [Deploy models with Azure Machine Learning](#)
 - [Serving of ML models in Kubeflow](#)

Summary

How to come up the solution?

1. Doing a survey on related work
2. Choosing an unsupervised learning approach
3. Picking up a clustering algorithm
4. Selecting and transforming features
5. Tuning parameters and evaluation
6. If not satisfied, go back to previous steps
7. Deploying your model in production

