Machine Learning for Cities
CUSP-GX 5006.001, Spring 2019

Lecture 12: Anomalous Pattern Detection

Anomalous pattern detection

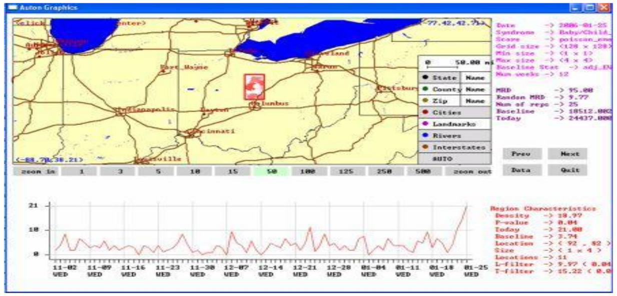
Main goal of pattern detection: to **identify** and **characterize** relevant subsets of a massive dataset, i.e. groups of records that differ from the rest of the data in an interesting way.

Question 1: Are any relevant patterns present in the data, or is the entire dataset “normal”?

Question 2: If there are any patterns, identify the pattern type and the affected subset of data records for each.

Example: outbreak detection
Are there any emerging outbreaks of disease? If so, what type of outbreak, and what areas are affected?

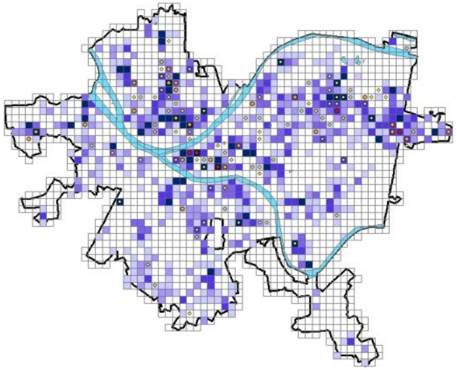
Example: crime prevention
Can we detect, or ideally predict, emerging hot-spots of violent crime, and prevent crime by targeted police patrols?



```
Date        -> 2006-01-25
Syndrome    -> Baby/Child_
Score       -> poisson_ene
Grid size   -> (128 x 128)
Min size    -> (1 x 1)
Max size    -> (4 x 4)
Baseline Stat   -> adj_EU
Num weeks   -> 12

MRD           -> 95.00
Random MRD    -> 9.77
Num of reps   -> 25
Baseline      -> 18512.002
Today         -> 24437.000
```

```
Region Characteristics
 Density     -> 10.97
 P-value     -> 0.04
 Today       -> 21.00
 Baseline    -> 3.74
 Location    -> < 92 , 82 >
 Size        -> < 1 x 4 >
 Locations   -> 11
 L-filter    -> 9.97 < 0.04
 T-filter    -> 15.22 < 0.0
```



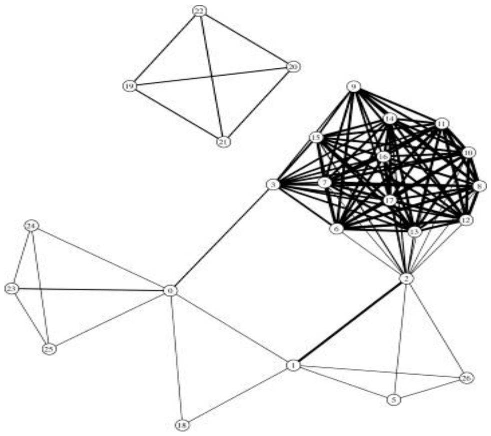
Anomalous pattern detection

Main goal of pattern detection: to **identify** and **characterize** relevant subsets of a massive dataset, i.e. groups of records that differ from the rest of the data in an interesting way.

What makes a group of records "relevant"?

1. Matching some known pattern or structure.

Group detection: given a social network, find highly connected sets of individuals.



Many efficient algorithms have been developed to find dense subgraphs or other structures in network data.

Anomalous pattern detection

Main goal of pattern detection: to **identify** and **characterize** relevant subsets of a massive dataset, i.e. groups of records that differ from the rest of the data in an interesting way.

What makes a group of records "relevant"?

1. Matching some known pattern or structure.

2. Multiple related records that are individually anomalous.

Fraud detection: look for individuals with a history of suspicious transactions.

Network intrusion detection: look for suspicious combinations of activities (e.g. port scanning).

In these domains, multiple "slightly anomalous" behaviors may together provide evidence of a major deviation from normal.

Anomalous pattern detection

Main goal of pattern detection: to **identify** and **characterize** relevant subsets of a massive dataset, i.e. groups of records that differ from the rest of the data in an interesting way.

What makes a group of records "relevant"?

1. Matching some known pattern or structure.

2. Multiple related records that are individually anomalous.

3. Higher (or lower) than expected number of records with some combination of attributes.

4. Change in data distribution as compared to the rest of the dataset.

Cluster detection: find spatial areas or periods of time with more records than expected.

Event detection: is the recent data differently distributed than the past?

Key concept: A group of records may be highly anomalous or interesting even if none of the individual records is itself anomalous.

Pattern detection = classification?

"Does the dataset contain a pattern?" This question could be treated as a problem of classifying datasets.

"Which records are affected?" This question could be treated as classification or anomaly detection.

Dataset D → {normal, contains a pattern}
Or D → {normal, pattern 1, pattern 2, …}

Record x_i → {normal, is part of a pattern}
Or x_i → {normal, pattern 1, pattern 2, …}

For the first approach, what features of the dataset should we use?

Most relevant patterns only affect a small proportion of the dataset, and would not be visible looking only at summary statistics.

On the other hand, data sparsity prevents using each record-attribute combination as a different attribute of the dataset.

For the second approach, how can we combine data from multiple records?

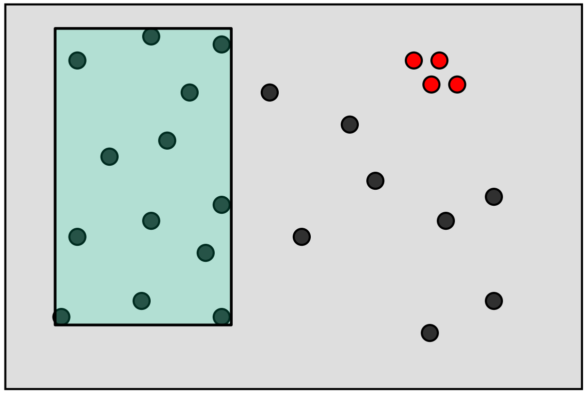
This is essential since none of the records may be individually sufficient to detect the anomalous pattern.

Question: Why search over subsets? Answer: Simpler approaches can fail.

Top-down detection approaches

Are there any globally interesting patterns? If so, recursively search the most interesting sub-partition.

Two examples: bump hunting; "cluster then detect".

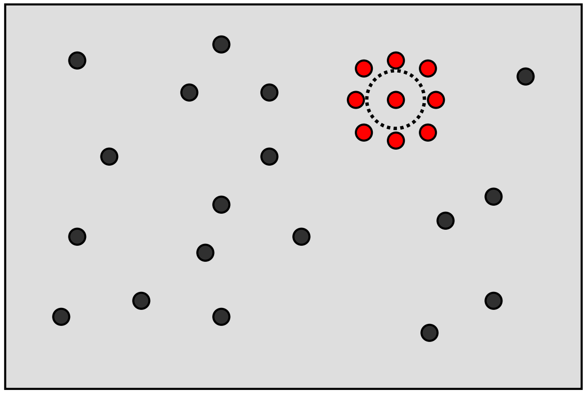


Top-down fails for **small-scale patterns** that are not evident from the global aggregates.

Bottom-up detection approaches

Find individually (or locally) anomalous data points, and optionally, aggregate into clusters.

Two examples: anomaly/outlier detection; density-based clustering.



Bottom-up fails for **subtle patterns** that are only evident when a group of data records are considered collectively.

Subset scanning

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

What’s Strange About Recent Events?

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

In the WSARE method (“What’s Strange About Recent Events”), we consider the subsets of the data defined by a one- or two-component rule R, and find rules where the current data is significantly different than the past.

For each rule, we create a 2x2 contingency table comparing current and past data:

	Current	Past
# records satisfying R	48	45
# records satisfying ~R	86	220

Compute p-value using a statistical test (X^2 or Fisher’s Exact). Lower p-value = higher score.

What’s Strange About Recent Events?

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

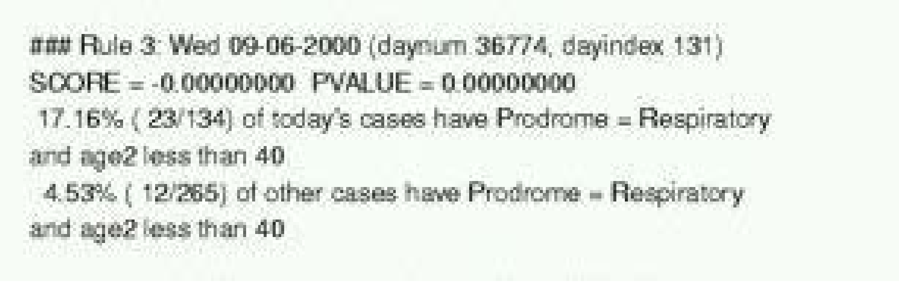
Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

In the WSARE method (“What’s Strange About Recent Events”), we consider the subsets of the data defined by a one- or two-component rule R, and find rules where the current data is significantly different than the past.

For example, using WSARE for hospital Emergency Department surveillance resulted in finding the following significant rule, corresponding to an outbreak of respiratory illness on 9/6/2000.



```
### Rule 3: Wed 09-06-2000 (daynum 36774, dayindex 131)
SCORE = -0.00000000  PVALUE = 0.00000000
 17.16% ( 23/134) of today's cases have Prodrome = Respiratory
and age2 less than 40
  4.53% ( 12/265) of other cases have Prodrome = Respiratory
and age2 less than 40
```

Anomaly pattern detection

We can scan over <u>subsets</u> of the dataset in order to find those groups of records that correspond to a pattern.

<u>Step 1</u>: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

<u>Step 2</u>: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

If we do not have access to past data, but we have access to the output of an anomaly detector, we can modify WSARE to detect rules R that correspond to a higher than expected number of anomalous records.

Now we create tables comparing the numbers of anomalous and normal records:

	Anomalous	Normal
# records satisfying R	17	5
# records satisfying ~R	93	400

→

Compute p-value using a statistical test (χ^2 or Fisher's Exact). Lower p-value = higher score.

Model-based pattern detection

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

In the **model-based** anomalous pattern detection approach, we model the effects of each pattern type P on the affected subset of the data S.

We then compute the **likelihood ratio statistic** Pr(Data | H_1(S, P)) / Pr(Data | H_0) for each (S, P).

In **event detection**, we model the null hypothesis H_0 by estimating expected counts for each data stream assuming no events.

Each pattern P is assumed to increase the counts for some data streams in the affected set of spatial locations S.

Anomalous group detection

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset $S = \{x_i\}$ and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

There are many options for computing the score of a subset S.

In the **model-based** anomalous pattern detection approach, we model the effects of each pattern type P on the affected subset of the data S.

We then compute the **likelihood ratio statistic** $\Pr(\text{Data} \mid H_1(S, P)) / \Pr(\text{Data} \mid H_0)$ for each (S, P).

In our **AGD** ("Anomalous Group Detection") approach, we model the null hypothesis by learning a Bayes Net from training data.

Under the alternative hypothesis $H_1(S)$, we assume that records in S are drawn from a different Bayes Net.

Which patterns to report?

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

Option 1: Report the k highest scoring subsets, ordered by score.

The disadvantage of this approach is that the user is not informed whether any of the discovered patterns are likely to be relevant.

However, this may be acceptable in monitoring systems or scientific discovery applications where the user is willing to evaluate a fixed number of potential patterns.

Which patterns to report?

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset S = {x_i} and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

Option 2: Perform hypothesis tests, and report all **significant** patterns (S, P).

In the hypothesis testing framework, we must adjust for the fact that we're performing so many tests. Otherwise we will report too many false positives!

In model-based approaches, one way to do this is **randomization**: we generate a large number of simulated datasets assuming the null model, and compare the scores of the potential patterns in the real dataset to the highest scoring patterns in the simulated data.

An alternative is to adjust the p-value threshold for each test based on the number of tests performed (e.g. Bonferroni threshold = .05 / # tests)

Which patterns to report?

We can scan over subsets of the dataset in order to find those groups of records that correspond to a pattern.

Step 1: Compute **score** F(S, P) for each subset $S = \{x_i\}$ and for each pattern type P, where higher score means more likely to be a pattern.

Step 2: Consider the highest scoring potential patterns (S, P) and decide whether each actually represents a pattern.

Option 3: Compute the **posterior probability** of each hypothesis $H_1(S, P)$.

In a Bayesian framework, we must spread the prior probability of a pattern over all possible hypotheses $H_1(S, P)$.

We then compute the likelihood of the data given each hypothesis $H_1(S, P)$, as well as the null hypothesis of no patterns, H_0.

We can then compute the posterior probability of each hypothesis by Bayes' Theorem:

$$\Pr(H \mid D) = \Pr(D \mid H) \Pr(H) / \Pr(D)$$

Best for known pattern types!

Option 2 better for anomalies!

Which subsets to scan?

Since there are exponentially many subsets of the data, it is often computationally infeasible to search all of them.

The most common approach is to use domain knowledge to restrict our search space: for example, in spatial cluster detection, we assume that a pattern will affect a spatially localized group of records, and often further restrict the cluster size and shape.

e.g. “search over circular regions centered at a data point” → only N^2 regions instead of 2^N.

Another common approach is to perform a greedy search. For example, we grow subsets starting from each record, repeatedly adding the additional record that gives the highest scoring subset.

Tradeoff: much more efficient than naïve search, but not guaranteed to find highest scoring region.

In some cases, we can find the highest-scoring subsets without actually computing the scores of all possible subsets!

Example: fast spatial scan

In **spatial cluster detection**, we search over sets of adjacent locations and find spatial regions with significantly higher than expected counts.

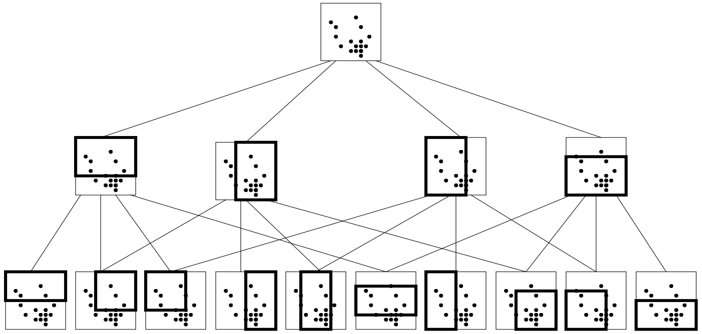
We restrict our search to rectangular regions for computational efficiency. This gives us high detection power for both compact and elongated clusters.

For massive datasets (e.g. disease surveillance for nationwide health data), we have to search over billions of possible regions, which could take weeks.

We can find the highest scoring clusters without an exhaustive search using **branch and bound**: we keep track of the highest region score that we have found so far, and prune sets of regions with provably lower scores.

A new multi-resolution data structure, the overlap-kd tree, enables us to make this search efficient.

We can now monitor nationwide health data in 20 minutes (vs. 1 week).



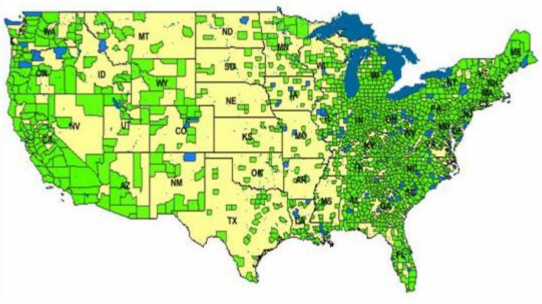
Linear-time subset scanning

Given a score function F(S) which satisfies the **linear-time subset scanning** property, we can optimize F(S) over the exponentially many subsets of data records, while evaluating only O(N) regions instead of O(2^N).

Just sort the locations from highest to lowest **priority** according to some function, then search over groups consisting of the top-k highest priority locations (k = 1..N). The highest scoring subset will be one of these!

Many useful score functions satisfy the LTSS property. In the spatial cluster detection setting, we can efficiently optimize a **spatial scan statistic** over subsets of locations to find the most interesting spatial region.

This works both for **univariate data**, monitoring a single data stream across time and space, and **multivariate data**, monitoring multiple data streams.



We can also incorporate relevant constraints such as spatial proximity, temporal consistency, or graph connectivity into the detection process.

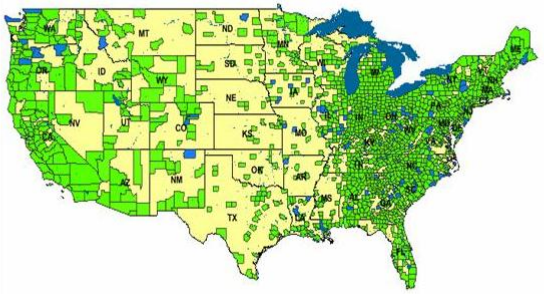
Linear-time subset scanning

Given a score function F(S) which satisfies the **linear-time subset scanning** property, we can optimize F(S) over the exponentially many subsets of data records, while evaluating only O(N) regions instead of O(2^N).

Just sort the locations from highest to lowest **priority** according to some function, then search over groups consisting of the top-k highest priority locations (k = 1..N). The highest scoring subset will be one of these!

Many useful score functions satisfy the LTSS property. In the spatial cluster detection setting, we can efficiently optimize a **spatial scan statistic** over subsets of locations to find the most interesting spatial region.

This works both for **univariate data**, monitoring a single data stream across time and space, and **multivariate data**, monitoring multiple data streams.



LTSS allows us to solve problems in milliseconds that would previously have required hundreds of millions of years!

DBSCAN in scikit-learn

DBSCAN is a bottom-up **density-based clustering** method which can be used either for clustering or for anomalous (dense) cluster detection.

Main steps: DBSCAN identifies all individual points with "dense" local neighborhoods (at least *MinPts* points within radius *eps)*.

Then, it merges nearby dense points (and all other points within radius *eps* of them) into clusters. Remaining points in the dataset are not placed into any cluster.

Results of DBSCAN are critically dependent on choice of parameters *MinPts* and *eps*:

Smaller *MinPts*, larger *eps*: almost all points assigned to clusters. Use like a standard clustering method, but robust to outliers, don't need to specify # clusters.

Larger *MinPts*, smaller *eps*: most points are unassigned. Clusters represent small spatial areas that are unusually dense compared to the rest of the data.

Our approach to using DBSCAN for anomalous cluster detection:

Scan through a wide range of parameter values. Choose values where # of points assigned to clusters is >0 but <10% of the data.

References

- A coherent text on anomalous pattern detection has yet to be written, but many methods have been proposed and are becoming common.
  - WSARE: W.-K. Wong et al. Rule-based anomaly pattern detection for detecting disease outbreaks. *Proc. AAAI Conf.,* 2002.
  - APD: K. Das, J. Schneider, and D.B. Neill. Anomaly pattern detection in categorical datasets. *Proc. KDD Conf.,* 2008.
  - DBSCAN: M. Ester et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proc. KDD Conf.,* 1996. http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
  - Spatial scan: M. Kulldorff. A spatial scan statistic. *Comm. Stat. Theor. Meth*., 1997.
  - Fast subset scan: D.B. Neill. Fast subset scan for spatial pattern detection, *J. Royal Stat. Soc. B*, 2012.
- More detailed tutorial slides on event detection:
  - D.B. Neill and W.-K. Wong, “A Tutorial on Event Detection,” presented at *KDD 2009* conference.