Anomaly Detection via Online Over-Sampling Principal Component Analysis

Yi-Ren Yeh[1], Yuh-Jye Lee[2] and Yu-Chiang Frank Wang[1]

[1]Research Center for Information Technology Innovation, Academia Sinica
[2]Department of Computer Science and Information Engineering, NTUST

December 30, 2010

Outline

Outline

Introduction

- Outlier detection is an important issue in data mining and has been studied in different research areas.
- Outlier detection methods are designed for finding the rare instances or deviated data.
- In this work, we use "Leave One Out" procedure to check each individual point the "with or without" effect on the variation of principal directions.
- An over-sampling principal component analysis (PCA) outlier detection method is proposed for emphasizing the influence of an abnormal instance as well.
- We also present a quick updating technique which satisfies the on-line scenarios.

One Possible Definition of Outliers

- An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism (by Hawkins).

One Possible Definition of Outliers

- An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism (by Hawkins).
- Michael Jordan is an outlier because of a well-known quotation by Charles Barkley: "I am the best basketball player in the earth, Jordan? He is an alien".

Another Possible Definition of Outliers

- An outlier is an observation that enormously affects model when we add or remove it from the entire dataset.

Another Possible Definition of Outliers

- An outlier is an observation that enormously affects model when we add or remove it from the entire dataset.
- Wilt Chamberlain is an outlier on account of his responsibility for several rule changes in basketball. In order to diminish his dominance, the basketball authorities set some rules including widening the lane, as well as changes to rules regarding inbounding the ball and shooting free throws.

Outline

Principal Component Analysis

- Let $\mathbf{A} = [\mathbf{x}_1^\top; \mathbf{x}_2^\top; \cdots; \mathbf{x}_n^\top] \in \mathbb{R}^{n \times p}$ be the data matrix.
- Typically, PCA is formulated as the following optimization problem

$$\max_{\mathbf{U} \in \mathbb{R}^{p \times k}, \|\mathbf{U}\| = \mathbf{I}} \sum_{i=1}^{n} \mathbf{U}^\top (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top \mathbf{U}. \tag{1}$$

- Alternatively, one can approach the PCA problem as minimizing the data reconstruction error, i.e.

$$\min_{\mathbf{U} \in \mathbb{R}^{p \times k}, \|\mathbf{U}\| = \mathbf{I}} J(\mathbf{U}) = \sum_{i=1}^{n} \|(\mathbf{x}_i - \boldsymbol{\mu}) - \mathbf{U}\mathbf{U}^\top (\mathbf{x}_i - \boldsymbol{\mu})\|^2. \tag{2}$$

- $\mathbf{U}$ is a matrix consisting of k dominant eigenvectors.

Principal Component Analysis (cont'd)

- Generally, the problem in either (1) or (2) can be solved by deriving an eigenvalue decomposition problem:

$$\mathbf{\Sigma_A U} = \mathbf{U}\Lambda \tag{3}$$

- $\mathbf{\Sigma_A}$ is the covariance matrix.
- The time complexity and memory requirement are $O(p^3)$ and $O(p^2)$ respectively.

The Effect of An Outlier on Principal Directions

- PCA is sensitive to outliers.
- We use the leave one out (LOO) procedure to explore the variation of principal direction.
- A particular instance with high variation of the principal directions will be an abnormal instance.

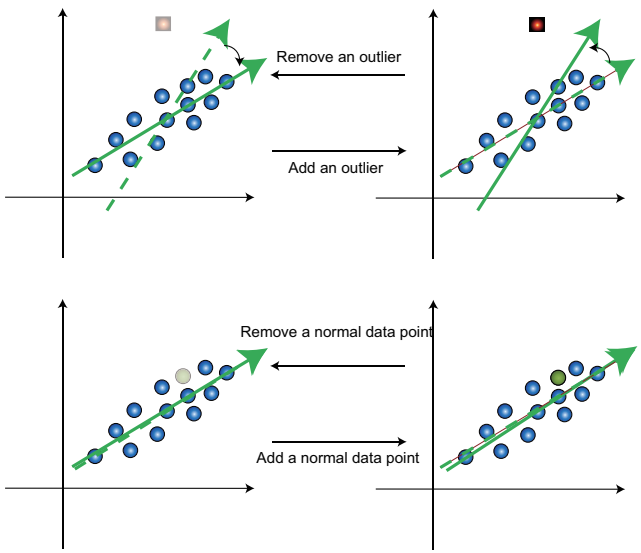


Figure: The effect of adding/removing an outlier or a normal data instance on the principal direction.

Decremental PCA with LOO Scheme for Anomaly Detection

- In our framework, we need to evaluate a decremental PCA problem n times in the LOO procedure:

$$\mathbf{\Sigma}_{\tilde{\mathbf{A}}}\tilde{\mathbf{u}}_t = \lambda\tilde{\mathbf{u}}_t, \tag{4}$$

  where $\tilde{\mathbf{A}} = \mathbf{A}/\{\mathbf{x}_t\}$ and $\mathbf{\Sigma}_{\tilde{\mathbf{A}}}$ is the covariance of $\tilde{\mathbf{A}}$.
- Use $s_t = 1 - |\frac{\langle\tilde{\mathbf{u}}_t,\mathbf{u}\rangle}{\|\tilde{\mathbf{u}}_t\|\|\mathbf{u}\|}|$ to measure the variation of the principal directions.
- Note that $\mathbf{u}$ is the the dominant principal direction from $\mathbf{A}$.
- A higher s_t score (closer to 1) means that the target instance is more likely to be an outlier.

Incremental PCA for Anomaly Detection

- In contrast with decremental PCA, we also consider the use of incremental PCA for outlier detection.
- This strategy is preferable in *online* anomaly detection applications.
- That is, we can use it to determine whether a newly received data instance is an outlier.
- The incremental PCA can be formulated as follows

$$\mathbf{\Sigma}_{\tilde{\mathbf{A}}}\tilde{\mathbf{u}}_t = \lambda\tilde{\mathbf{u}}_t, \tag{5}$$

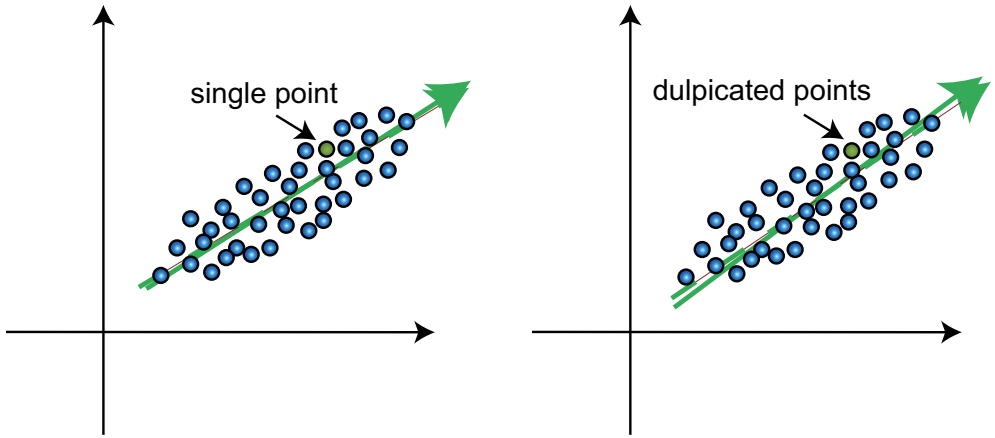
where $\tilde{\mathbf{A}} = \mathbf{A} \cup \{\mathbf{x}_t\}$.

- Similarly, we check the score s_t of each newly received instance and determine its outlierness accordingly.

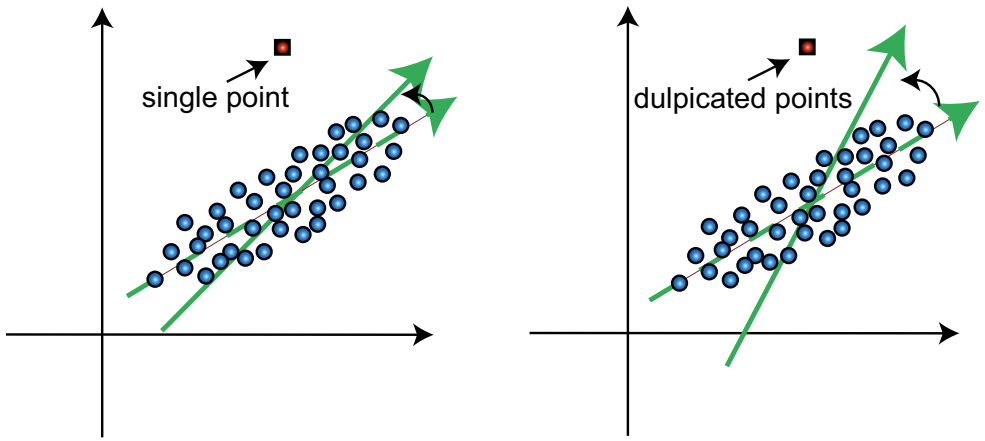
Outline

Over-Sampling Principal Components Analysis (osPCA)

- A single outlier instance will not significantly change the principal direction when the size of the data is large.
- We employ an over-sampling scheme to emphasize the influence of an outlier.
- The variation of principal directions and mean of the data will be enlarged if we duplicate an outlier.
- We integrate the over-sampling and LOO strategies together with the incremental PCA.



(a) Over-sampling a normal data point



(b) Over-sampling an outlier

Figure: The effect of an over-sampled normal data or outlier instance on the principal direction.

osPCA (cont'd)

- Our osPCA algorithm can be formulated as follows

$$\boldsymbol{\Sigma}_{\tilde{\mathbf{A}}}\tilde{\mathbf{u}}_t = \lambda\tilde{\mathbf{u}}_t, \tag{6}$$

where $\tilde{\mathbf{A}} = \mathbf{A} \cup \{\mathbf{x}_t, \ldots, \mathbf{x}_t\} \in \mathbb{R}^{(n+\tilde{n})\times p}$.

- Note that

$$\boldsymbol{\Sigma}_{\tilde{\mathbf{A}}} = \frac{1}{n+\tilde{n}}\sum_{\mathbf{x}_i\in\mathbf{A}}(\mathbf{x}_i - \tilde{\boldsymbol{\mu}})(\mathbf{x}_i - \tilde{\boldsymbol{\mu}})^\top + \frac{1}{n+\tilde{n}}\sum_{i=1}^{\tilde{n}}\mathbf{x}_t\mathbf{x}_t^\top - \tilde{\boldsymbol{\mu}}\tilde{\boldsymbol{\mu}}^\top, \tag{7}$$

i.e., we will duplicate the target instance $\tilde{n}$ times.

- The major concern is the computation cost of calculating or updating the principal directions in large-scale problems.

Remarks on μ and $\Sigma_{\mathbf{A}}$ for osPCA

- It is unnecessary to re-compute the covariance matrix in LOO.
- The covariance matrix can be easily updated while duplicating a target instance.
- Let $\mathbf{Q} = \frac{\mathbf{A}\mathbf{A}^\top}{n}$ be the original outer product matrix.
- We update $\tilde{\mu}$ and $\Sigma_{\tilde{\mathbf{A}}}$ by:

$$\tilde{\mu} = \frac{\boldsymbol{\mu} + r \cdot \mathbf{x}_t}{1+r} \text{ and } \Sigma_{\tilde{\mathbf{A}}} = \frac{1}{1+r}\mathbf{Q} + \frac{r}{1+r}\mathbf{x}_t\mathbf{x}_t^\top - \tilde{\boldsymbol{\mu}}\tilde{\boldsymbol{\mu}}^\top.$$

- Note that $0 < r < 1$ is the parameter controlling the size when over-sampling $\mathbf{x}_t$.

The Power Method for osPCA

- To alleviate this computation load, we apply the well-known *power method* to determine $\tilde{\mathbf{u}}$.
- This method starts with an initial normalized vector $\tilde{\mathbf{u}}^{(0)}$.
- $\tilde{\mathbf{u}}$ is determined by

$$\text{While } (\tilde{\mathbf{u}}^{(k)} \neq \tilde{\mathbf{u}}^{(k-1)})$$

$$\tilde{\mathbf{u}}^{(k+1)} = \frac{\mathbf{\Sigma}_{\tilde{\mathbf{A}}} \tilde{\mathbf{u}}^{(k)}}{\|\mathbf{\Sigma}_{\tilde{\mathbf{A}}} \tilde{\mathbf{u}}^{(k)}\|}$$

$$\text{End}$$

- We only use the first principal component in our experiments.

Some Remarks on Power Method

- Still need to solve an eigenvalue decomposition.
- We can use the previous principal direction as the initial point in power method to reduce computation time.
- For high dimensional data, it is not practical to keep the covariance matrix.
- An online PCA algorithm to update the eigenvector is preferable, which approximates the minimization of reconstruction error formulation.

Least Squares Approximation for PCA

- Standard PCA:

$$\min_{\mathbf{U}\in\mathbb{R}^{p\times k},\|\mathbf{U}\|=\mathbf{I}} J(\mathbf{U}) = \sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \mathbf{U}\mathbf{U}^\top \bar{\mathbf{x}}_i\|^2, \tag{8}$$

  where $\mathbf{U}$ is a set eigenvectors and $\bar{\mathbf{x}}_i$ is $(\mathbf{x}_i - \mu)$.

- The above formulation can be further approximated by a least squares form (i.e., has a closed form solution):

$$\min_{\mathbf{U}\in\mathbb{R}^{p\times k},\|\mathbf{U}\|=\mathbf{I}} J_{ls}(\mathbf{U}) = \sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \mathbf{U}\mathbf{y}_i\|^2, \tag{9}$$

  where $\mathbf{y}_i = \mathbf{U}'^\top \bar{\mathbf{x}}_i \in \mathbb{R}^k$ and $\mathbf{U}'$ is the approximation of $\mathbf{U}$.

- The trick for this least squares problem is the approximation of $\mathbf{U}^\top \bar{\mathbf{x}}_i$ by $\mathbf{y}_i = \mathbf{U}'^\top \bar{\mathbf{x}}_i$.

Online Updating for (Least Squares) osPCA

- In an online setting, we approximate the current $\mathbf{y}_i = \mathbf{U}_t^\top \bar{\mathbf{x}}_i$ by the previous solution $\mathbf{U}_{t-1}^\top \bar{\mathbf{x}}_i$ as follows

$$\min_{\mathbf{U}_t \in \mathbb{R}^{p \times k}, \|\mathbf{U}\| = \mathbf{I}} J_{ls}(\mathbf{U}_t) = \sum_{i=1}^{t} \|\bar{\mathbf{x}}_i - \mathbf{U}_t \mathbf{y}_i\|^2, \tag{10}$$

  where $\mathbf{y}_i = \mathbf{U}_{t-1}^\top \bar{\mathbf{x}}_i$.

- For a target instance, we have

$$\min_{\tilde{\mathbf{U}} \in \mathbb{R}^{p \times k}, \|\tilde{\mathbf{U}}\| = \mathbf{I}} J_{ls}(\tilde{\mathbf{U}}) \approx \sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \tilde{\mathbf{U}} \mathbf{y}_i\|^2 + \|\bar{\mathbf{x}}_t - \tilde{\mathbf{U}} \mathbf{y}_t\|^2, \tag{11}$$

  where $\mathbf{y}_t$ is approximated by $\mathbf{U}^\top \bar{\mathbf{x}}_t$.

Online Updating for osPCA (cont'd)

- When over-sampling the target instance $\tilde{n}$ times, we have

$$\min_{\tilde{\mathbf{U}}\in\mathbb{R}^{p\times k},\|\tilde{\mathbf{U}}\|=\mathbf{I}} J_{ls}(\tilde{\mathbf{U}}) \approx \sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \tilde{\mathbf{U}}\mathbf{y}_i\|^2 + \tilde{n}\|\bar{\mathbf{x}}_t - \tilde{\mathbf{U}}\mathbf{y}_t\|^2. \quad (12)$$

- Equivalently, we convert the above problem into the following form

$$\min_{\tilde{\mathbf{U}}\in\mathbb{R}^{p\times k},\|\tilde{\mathbf{U}}\|=\mathbf{I}} J_{ls}(\tilde{\mathbf{U}}) \approx \beta(\sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \tilde{\mathbf{U}}\mathbf{y}_i\|^2) + \|\bar{\mathbf{x}}_t - \tilde{\mathbf{U}}\mathbf{y}_t\|^2. \quad (13)$$

- β can be regarded as a weighting factor to suppress the information from existing data.
- The relation between β and the over-sampled number $\tilde{n}$ is $\beta = \frac{1}{\tilde{n}} = \frac{1}{nr}$.

Online Updating for osPCA (cont'd)

- We calculate the solution of $\tilde{\mathbf{u}}$ by taking the derivative of (13) with respect to $\tilde{\mathbf{u}}$, and thus we have

$$\tilde{\mathbf{u}} = \frac{\beta(\sum_{i=1}^{n} y_i \bar{\mathbf{x}}_i) + y_t \bar{\mathbf{x}}_t}{\beta(\sum_{i=1}^{n} y_i^2) + y_t^2}, \tag{14}$$

where $y_i = \mathbf{u}^\top \mathbf{x}_i$ and $y_t = \mathbf{u}^\top \mathbf{x}_t$ are the approximations of $\tilde{\mathbf{u}}^\top \mathbf{x}_i$ and $\tilde{\mathbf{u}}^\top \mathbf{x}_t$, respectively.

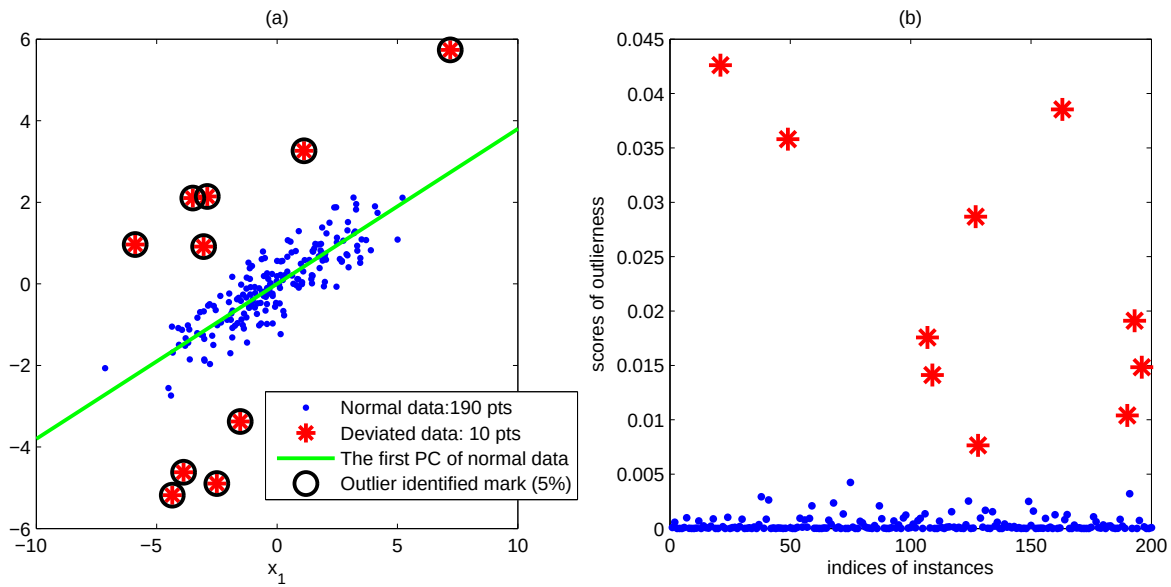
Table I. Comparisons of the power method and our proposed online osPCA for anomaly detection in terms of computational complexity and memory requirements. Note that m indicates the number of iterations.

	Power Method	Online Over-sampling PCA
Computation complexity	$O(nmp^2)$	$O(np)$
Memory requirement	$O(p^2)$	$O(p)$

Outline

2D Synthetic Data Set

- We generate a 2-D synthetic data, which consists of 190 normal instances and 10 deviated instances.
- We aim to identify the top 5% of the data as deviated data (the number of outliers we generated).
- The scores of outlierness of all 200 data points are shown the following plot.



UCI and KDD datasets

- In our experiments, we evaluate our methods on `pendigits` and KDD Cup 99 intrusion detection datasets.
- We compare our methods
  - dPCA (only removing one instance in LOO)
  - osPCA with power method
  - OsPCA with online updating

  with
  - LOF (local outlier factor, ACM SIGMOD 2000)
  - Fast ABOD (angle-based outlier detection, ACM SIGKDD 2008)
- In our experiments, we use AUC to evaluate the suspicious outlier ranking in outlier detection phase

Compared with Other Methods (pendigits dataset)

- Fixed the digit "0" as the normal data (780 instances) and set up 9 different combination via other digits (20 data points for each)

Scenario	dPCA (power method)	osPCA (power method)	osPCA (online updating)	Fast ABOD (SIGKDD 2008)	LOF (SIGMOD 2000)
0 vs. 1	0.9145 (0.0385)	0.9965 (0.0004)	0.9869 (0.0104)	0.9519 (0.0287)	0.9943 (0.0007)
0 vs. 2	0.9573 (0.0317)	0.9959 (0.0003)	0.9879 (0.0225)	0.9214 (0.0279)	0.9966 (0.0002)
0 vs. 3	0.4570 (0.0554)	0.9987 (0.0003)	0.9199 (0.0453)	0.9342 (0.0157)	0.9970 (0.0002)
0 vs. 4	0.7392 (0.0686)	0.9897 (0.0016)	0.8442 (0.0582)	0.9737 (0.0069)	0.9859 (0.0017)
0 vs. 5	0.8126 (0.0485)	0.9961 (0.0005)	0.9623 (0.0260)	0.9721 (0.0086)	0.9980 (0.0003)
0 vs. 6	0.9773 (0.0077)	0.9793 (0.0015)	0.9851 (0.0176)	0.9447 (0.0196)	0.9741 (0.0028)
0 vs. 7	0.8387 (0.0439)	0.9968 (0.0003)	0.9800 (0.0305)	0.9642 (0.0087)	0.9968 (0.0004)
0 vs. 8	0.8519 (0.0476)	0.9816 (0.0172)	0.9245 (0.0395)	0.9913 (0.0019)	0.9939 (0.0016)
0 vs. 9	0.6914 (0.0635)	0.9968 (0.0008)	0.9776 (0.0290)	0.9901 (0.0025)	0.9945 (0.0006)

Table: The AUC scores of decremental PCA (dPCA), over-sampling PCA (osPCA) with power method, our osPCA with online updating algorithm, fast ABOD, and LOF on the pendigits data set.

Methods	dPCA	osPCA (with power method)	osPCA (with online updating)	Fast ABOD	LOF
Time (sec.)	0.0589	0.0892	0.0121	13.804	0.0789

Table: Average CPU time (in seconds) of decremental PCA (dPCA), over-sampling PCA (osPCA) with power method, our osPCA with online updating algorithm, fast ABOD, and LOF on the `pendigits` data set.

Results on KDD 99 Data (Outlier Detection)

- We extract instances under the tcp protocal in 10% KDD cup data and test our method and LOF on them
- The size of normal data is 76813 and we also extract four different attacks as the outliers respectively.

Types & sizes of outliers	osPCA (online updating)		LOF	
	AUC	Time (sec.)	AUC	Time (sec.)
dos (50)	0.9145	1.784	0.9287	24.84
probe (50)	0.9824	1.784	0.9631	24.72
r2l (50)	0.8009	1.787	0.8253	22.12
u2r (49)	0.8902	1.765	0.8868	20.36

*It takes about 24 seconds to complete the procedure by using power mehtod

Results on KDD 99 Data (On-line Anomaly Detection)

- We extract 2000 normal instances points as the training set and apply the data cleaning phase to filter 100 points (5%) in the normal data to avoid the deviated data
- For testing, we select another 2000 normal instances and different size of attacks as our testing set.

Attack type	Testing data size		TP Rate	FP Rate	Error Rate
	normal	attack			
Dos	2000	100	0.940	0.073	0.073
Probe	2000	100	0.980	0.022	0.023
R2L	2000	100	0.900	0.071	0.072
U2R	2000	49	0.816	0.038	0.038

*TP rate is the percentage of attacks detected; FP rate is the percentage of normal connections falsely classified as attacks.

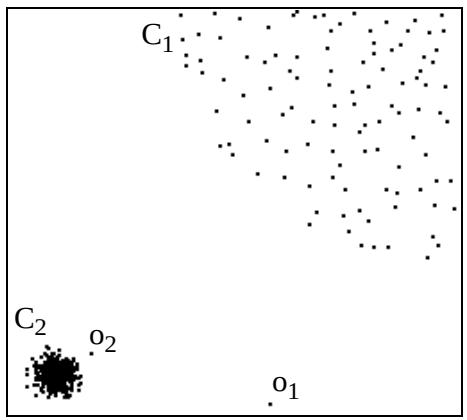
Outline

Conclusion and Future Work

- Variation of principal directions caused by outliers can determine data anomaly.
- The proposed osPCA can be used to enlarge the outlierness of an outlier in large-scale problems.
- Our online osPCA algorithm efficiently updates the principal directions without solving eigenvalue decomposition problems.
- Our method does not need to keep the entire covariance or data matrices during the evaluation process.
- Future research directions:
  - multi-clustering structure
  - data in a extremely high dimensional space

Local Outlier Factor

- One of the most popular outlier detection methods.
- A local density-based method to evaluate the outlierness for each instance.
- Considers the local data structure for estimating the density.
- The density of each individual instance's k-nearest neighbors is used to define the degree of outlierness.



Angle-based Outlier Detection

- Main concept of ABOD is using the variation of the angles between the each target instance and the rest instances
- An outlier or deviated instance will generate a smaller variance among its associated angles