Data Science

Anomaly Detection

Asst. Prof. Dr. Santitham Prom-on

Department of Computer Engineering, Faculty of Engineering

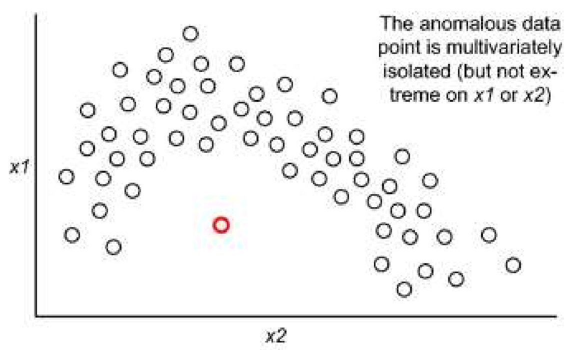
King Mongkut's University of Technology Thonburi

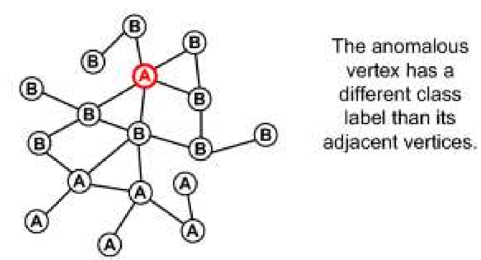
Topics

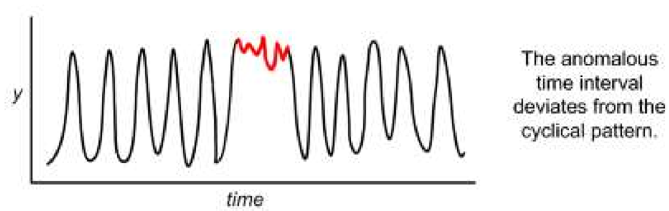
- Concepts of anomaly detection
- Types and methods of anomaly detection
- Python practices

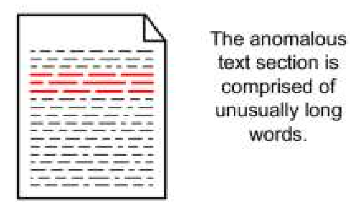
Anomaly detection

- Techniques of identifying rare events or observations
- They can raise suspicions by being statistically different from the rest of the observations









Types of anomalies

- **Point Anomaly:** A tuple in a dataset is said to be a Point Anomaly if it is far off from the rest of the data.
- **Contextual Anomaly:** An observation is a Contextual Anomaly if it is an anomaly because of the context of the observation.
- **Collective Anomaly:** A set of data instances help in finding an anomaly.

Methods of anomaly detections

- Rule-based approach: the fixed rule is used to define the detection threshold.

- Statistical analysis approach: the statistical analysis techniques are used to define the samples and detect those that are far from them.

- Machine learning approach: the ML models are created for the samples and detect those that are not likely to belong to them.

ML for anomaly detections

Supervised anomaly detection

- This method requires a labeled dataset containing both **normal** and **anomalous** samples.
- The problem is transformed into the classification task
- The most commonly used algorithms for this purpose are supervised Decision Tree, Support Vector Machine learning, K-Nearest Neighbors Classifier, etc.

Unsupervised anomaly detection

- This method does require any training data and instead assumes two things about the data
  - only a small percentage of data is anomalous and,
  - any anomaly is statistically different from the normal samples.
- Based on the above assumptions, the data is then
  - clustered using a similarity measure and,
  - the data points which are far off from the cluster are considered to be anomalies.

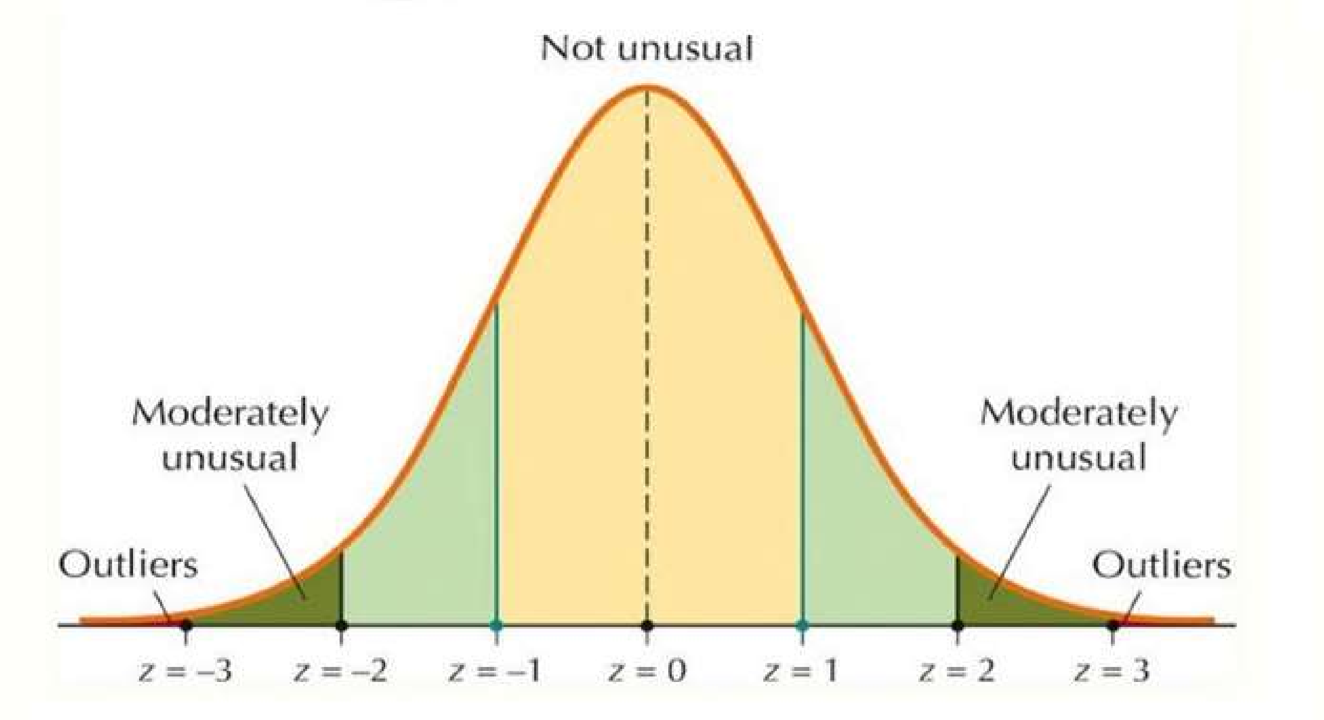
Popular statistical technique

- **Z-score (standard score)**: measures how many standard deviations a data point is from the mean. Typically, points with z-scores above 3 are considered outliers.

- **Median Absolute Deviation (MAD)**: similar to z-scores but use the median to find outliers. Since mean and standard deviation are easily skewed by outliers, this measurement are generally considered more robust.

- **Interquartile range (IQR)**: The IQR is the range between the first quartile (Q1) and the third quartile (Q3). An instance is considered an outlier if it falls outside the range [Q1 – 1.5*IQR, Q3 + 1.5*IQR].

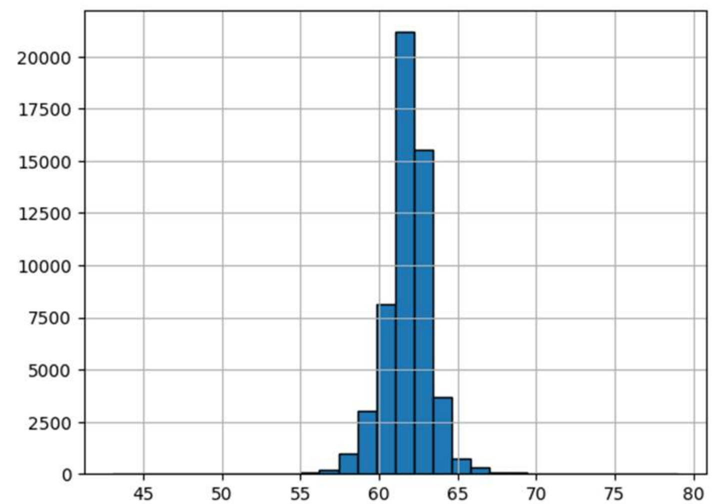
z-score


Not unusual
Moderately unusual
Moderately unusual
Outliers
Outliers
z = −3
z = −2
z = −1
z = 0
z = 1
z = 2
z = 3

z-score

```
[44] depth_z = StandardScaler().fit_transform(diamonds[['depth']])
     depth_z = pd.Series(depth_z.reshape(-1))
     outlier_z = (depth_z > 3)|(depth_z < -3)
     outlier_z
```

```
0        False
1        False
2         True
3        False
4        False
         ...
53935    False
53936    False
53937    False
53938    False
53939    False
Length: 53940, dtype: bool
```



Median Absolute Deviation (MAD) Python Outlier Detection (pyod)

```
# Initialize and fit a model
mad = MAD().fit(X)
```

```
mad
```

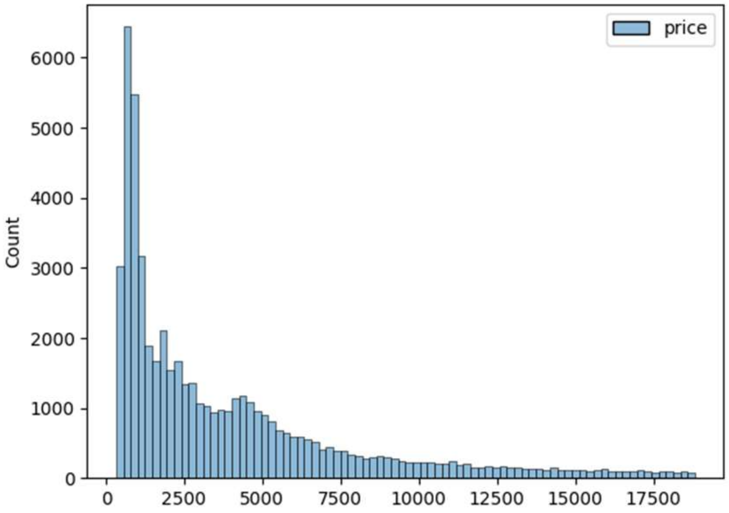
```
MAD(contamination=0.1, threshold=3.5)
```

```
# Resulting outlier label
labels = pd.Series(mad.labels_)
labels
```

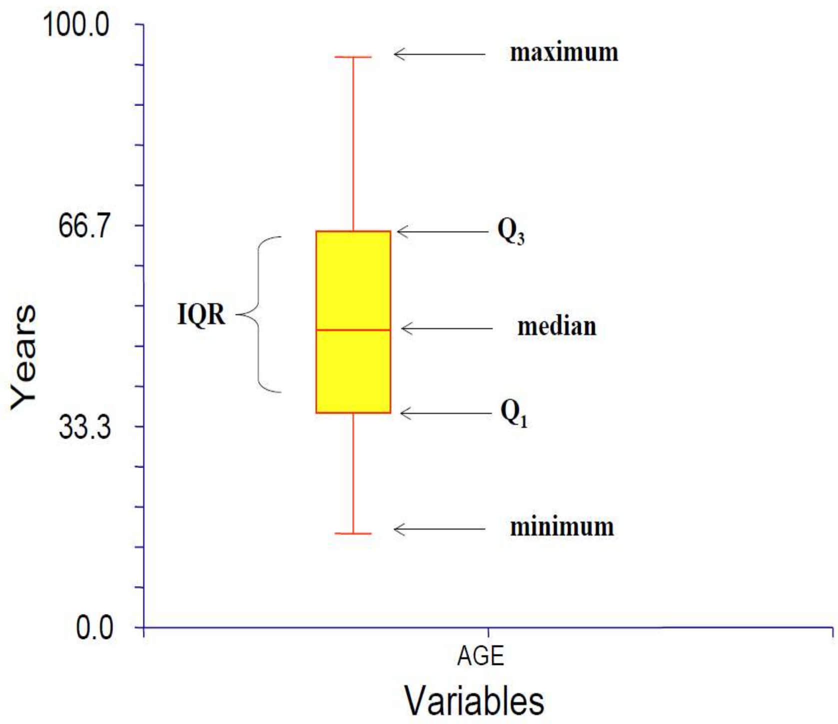
```
0        0
1        0
2        0
3        0
```

```
[56] labels.value_counts()
```

```
0    49708
1     4232
```

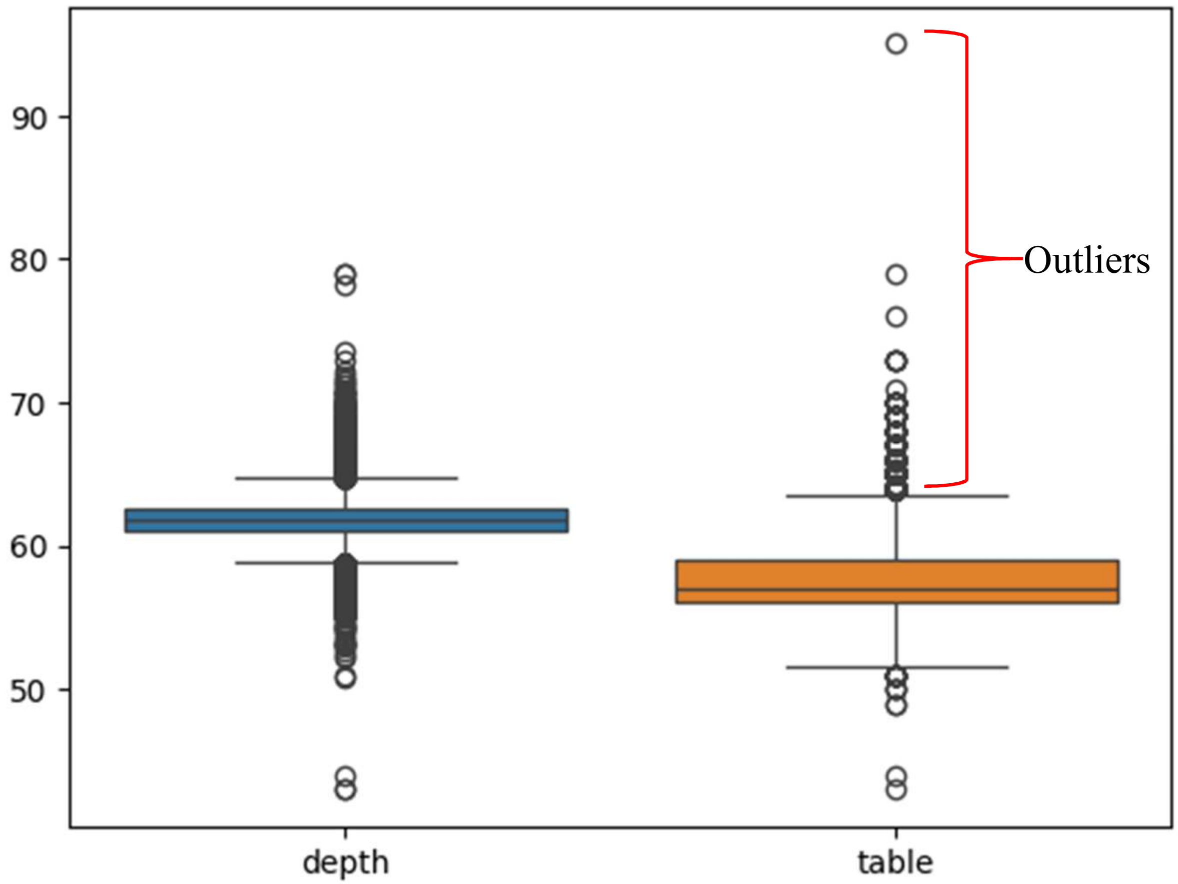


Boxplot



- The box in boxplot represents the middle 50% of the data
- The middle line indicates median
- Whiskers can be designated as either
  - Max/Min
  - Outlier boundaries
    - Upper = Q3 + 1.5*IQR
    - Lower = Q1 – 1.5*IQR

IQR (Boxplot)



ML-based approach

- **Outlier detection**
  - Training data contain outliers
  - Detect those outliers
- **Novelty detection**
  - Training data do NOT contain outlier
  - Detect whether NEW observation is outlier or not

Popular (unsupervised) ML techniques

Outlier detection

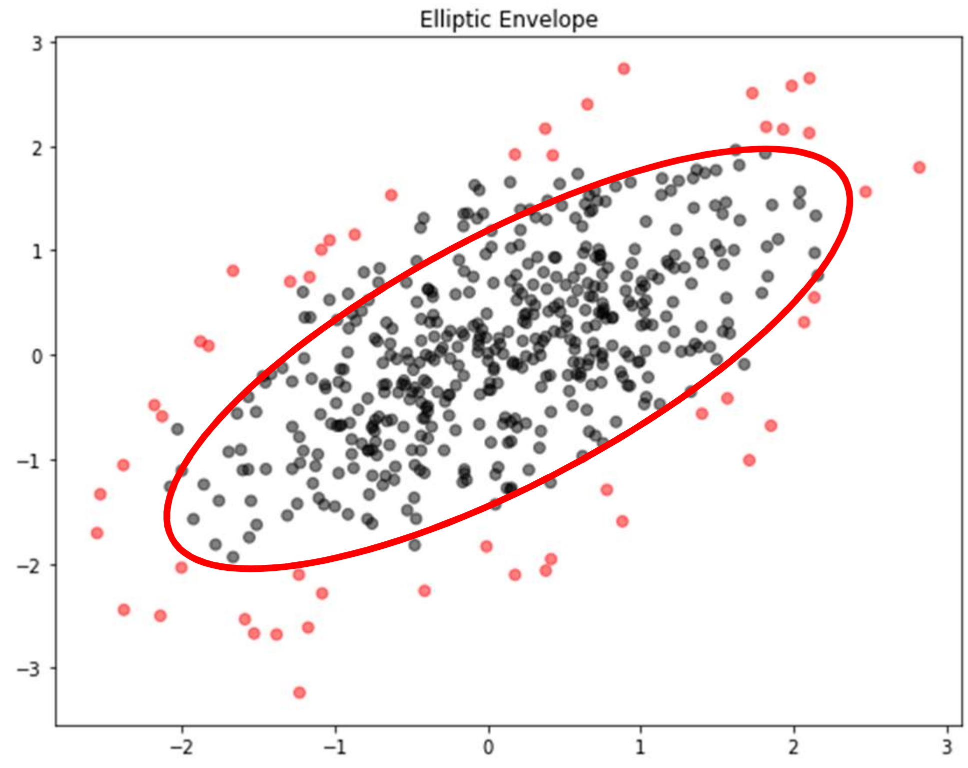
- **Elliptic envelope**: learn the envelope of the data
- **Isolation forest**: learn the tree/forest and find nodes with small supports

Novelty detection

- **Local outlier factor (LOF)**: learn outliers from the neighbors
- **One-class support vector machine (One-class SVM)**: use RBF to transform data and identify outliers that are far from them

Elliptical envelope

- Work only with Gaussian distributions
- Outliers are detected from covariance z-scores



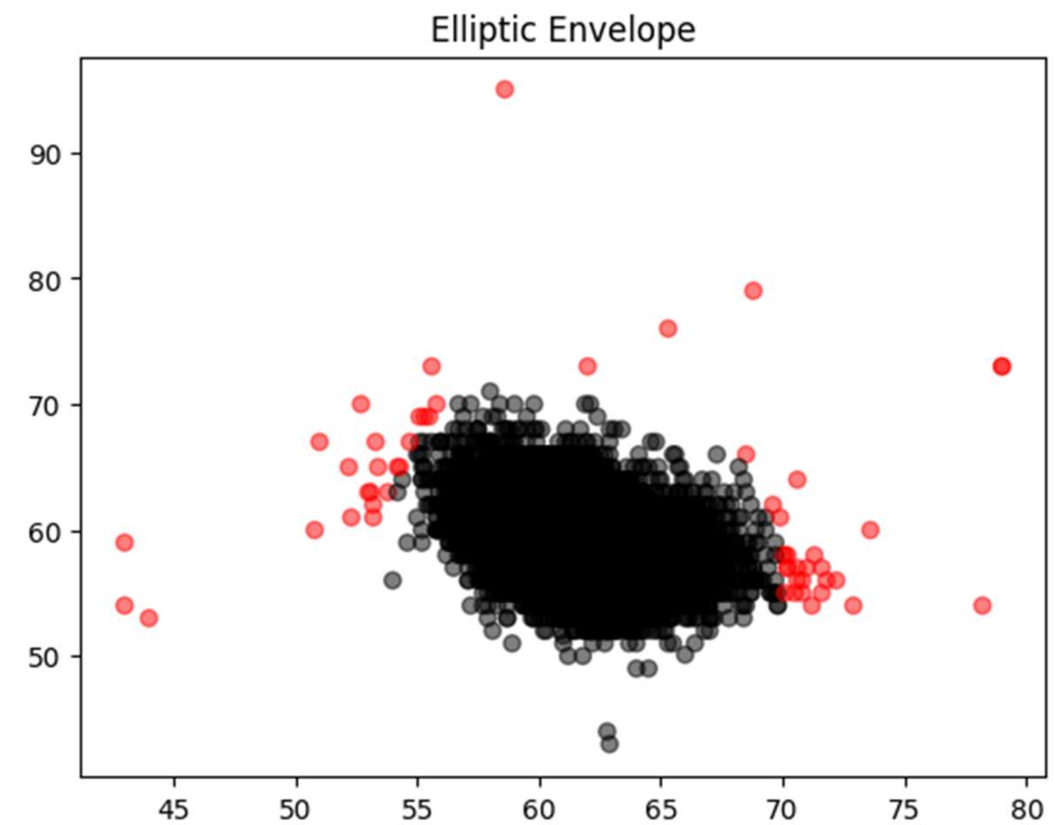
Elliptic envelope

```
[79] X = diamonds[['depth','table']]
     od_ee = EllipticEnvelope(contamination=0.001).fit(X)
     od_ee_res = od_ee.predict(X)
     od_ee_res

     array([1, 1, 1, ..., 1, 1, 1])
```

Contamination rate at 0.1%

Elliptic Envelope



Performance Evaluation

- Known labels: RUC-AUC, Accuracy, Precision, Recall

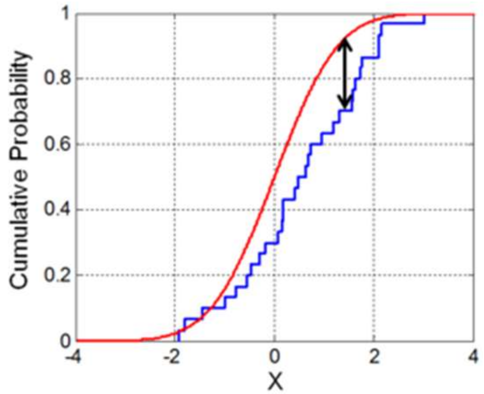
- Unknown labels:
  - Kolmogorov-Smirnov Test (ks-test): test whether 2 groups of data have the same distribution or not
  - Silhouette score: A higher coefficient provides stronger evidence the group designated as isotropic is cohesive
    - Before vs After removing outlier

KS-test

- Kolmogorov-Smirnov test (ks-test) performs a statistical test to accept or reject the null hypothesis that the two sets come from a common distribution.

Output

- p-value: a lower p-value provides greater statistical significance that the sets are distinct.
- statistic: a higher statistic provides stronger evidence that the distributions of the sets are different



Python: ks test (2 samples)

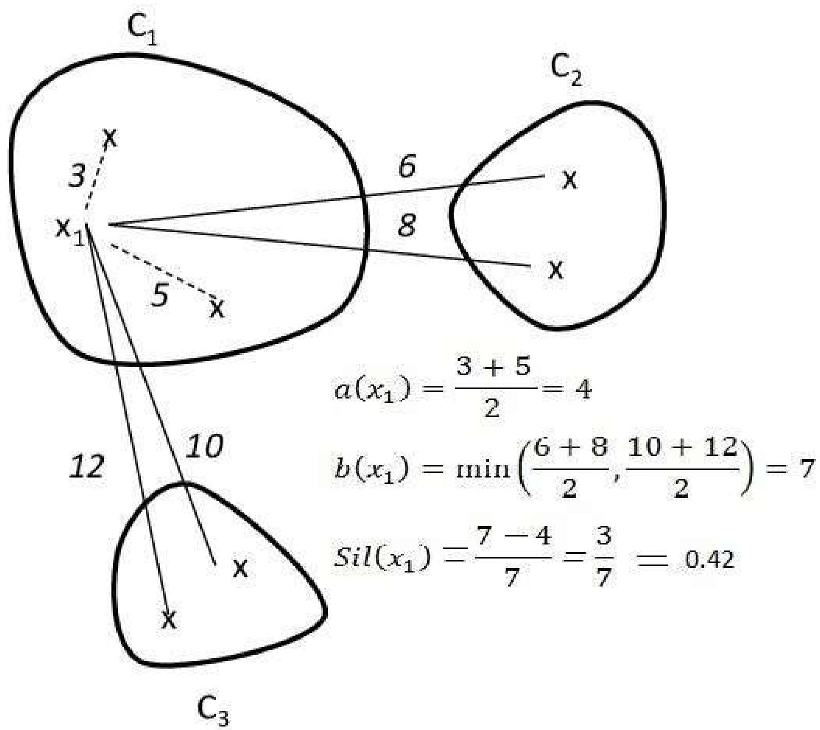
```
[106] from scipy.stats import ks_2samp
```

```
[108] ks_2samp(X.iloc[od_ee_res>0, 0], X.iloc[od_ee_res<0, 0])
```

```
KstestResult(statistic=0.5057508237555004, pvalue=2.5561587994143573e-13,
statistic_location=65.2, statistic_sign=1)
```

Silhoette Score

- Silhoette score measures the proportion of the intercluster distance and the data variability.
- It indicates how well the clustering separates data into groups.
- A higher coefficient provides stronger evidence the group designated as isotropic is cohesive.



Python: silhoette score

```
[114] from sklearn.metrics import silhouette_score
      import numpy as np
```

```
[111] silhouette_score(X,od_ee_res)
```

@ 0.1% contamination

```
0.7562952658758487
```

```
[119] od_ee_1 = EllipticEnvelope(contamination=0.01).fit(X)
      od_ee_res1 = od_ee_1.predict(X)
```

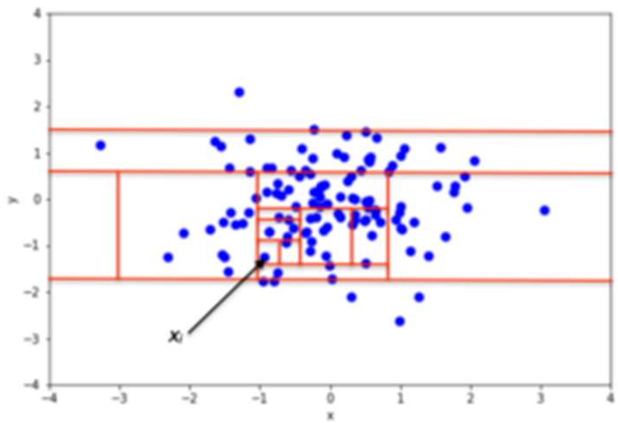
```
[120] silhouette_score(X,od_ee_res1)
```

@ 1% contamination

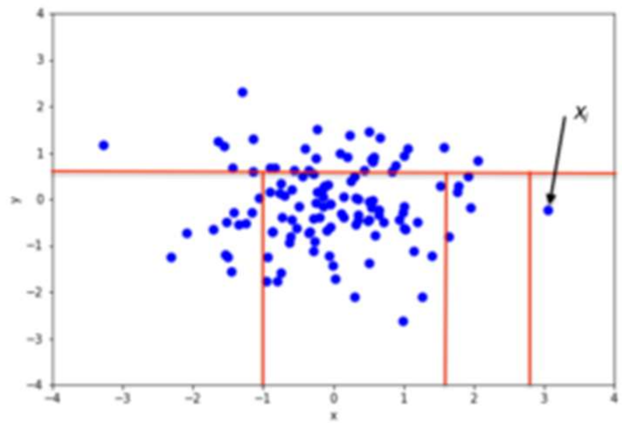
```
0.630073521574432
```

Isolation forest

- Anomalous data points are detected using binary tree.
- Anomalous points cannot can be isolated in early partition comparing to non-anomalous points (smaller path length)

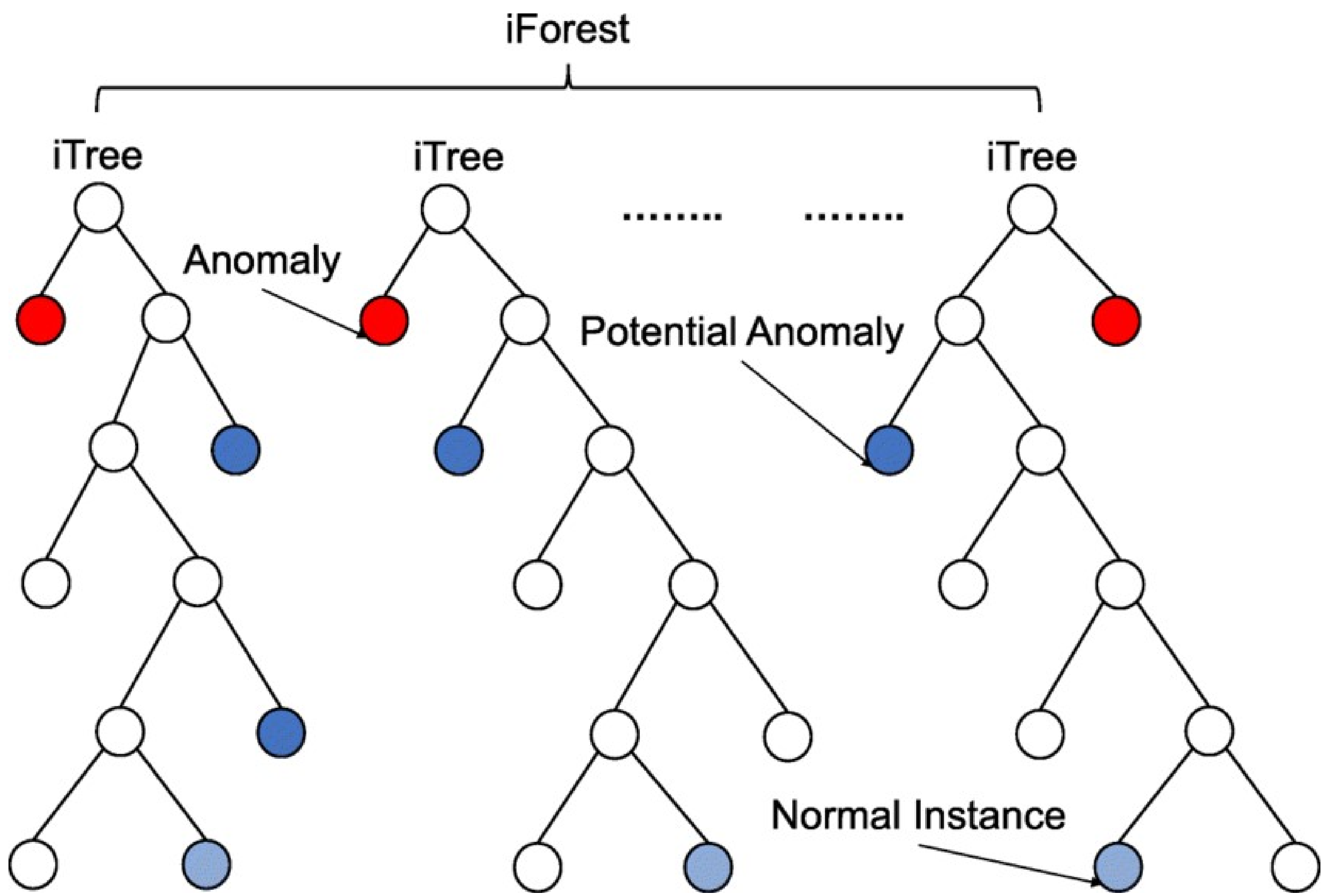


Non-anomalous



Anomalous

Liu, Fei Tony, Ting, Kai Ming and Zhou, Zhi-Hua. “Isolation forest.” Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on.


iForest
iTree
iTree
iTree
....
....
Anomaly
Potential Anomaly
Normal Instance

Python: isolation forest

```
od_if = IsolationForest(n_estimators=20, contamination=0.001).fit(X)
od_if_res = od_if.predict(X)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarr
  warnings.warn(
```

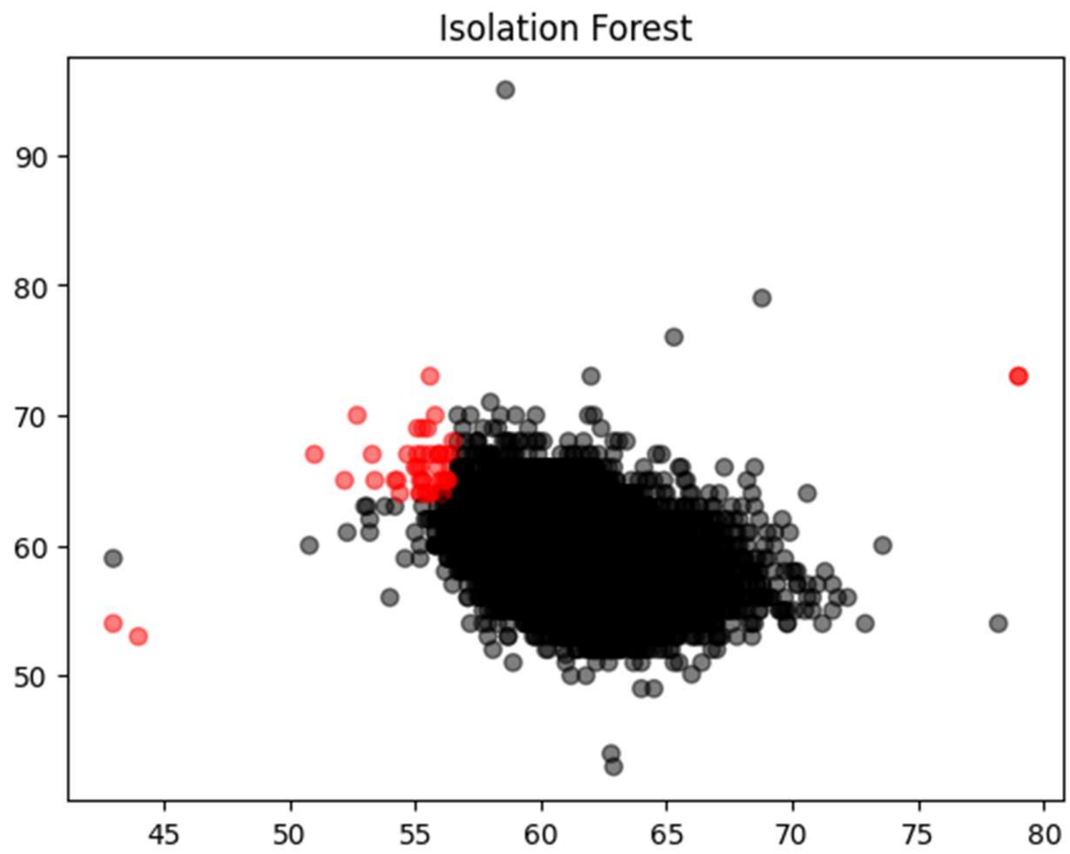
```
od_if_res
```

```
array([1, 1, 1, ..., 1, 1, 1])
```

```
silhouette_score(X,od_if_res)
```

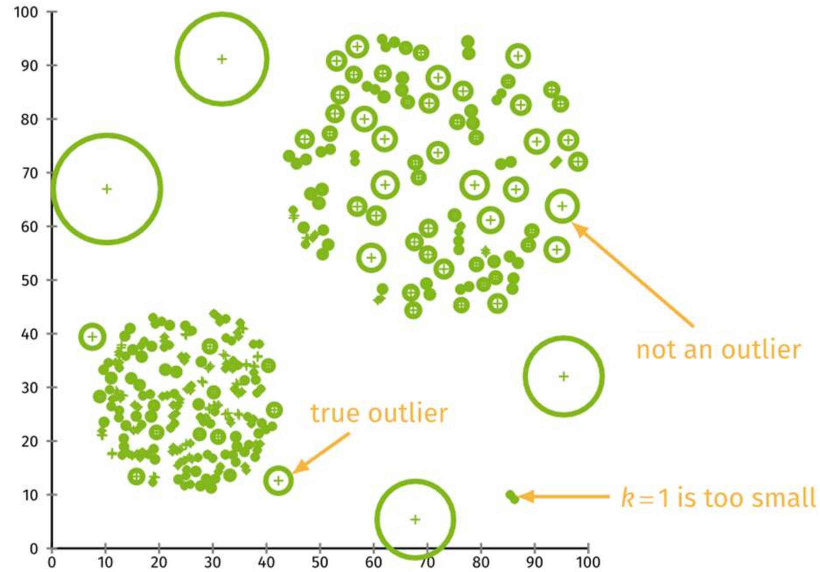
```
0.7174342124739187
```

Isolation Forest



Local Outlier Factor (LOF)

- Compute the point density
- Outliers are located in the low density area
- Observation:
  - different regions of a data set have different densities
  - distance- and density-based outlier detection may miss outliers in such data sets
- LOF
  - compare density with a local average
  - relatively low density ⇒ outliers



knn, k = 1

Breunig, Kriegel, Ng, and Sander (2000) LOF: identifying density-based local outliers. Proc. ACM SIGMOD

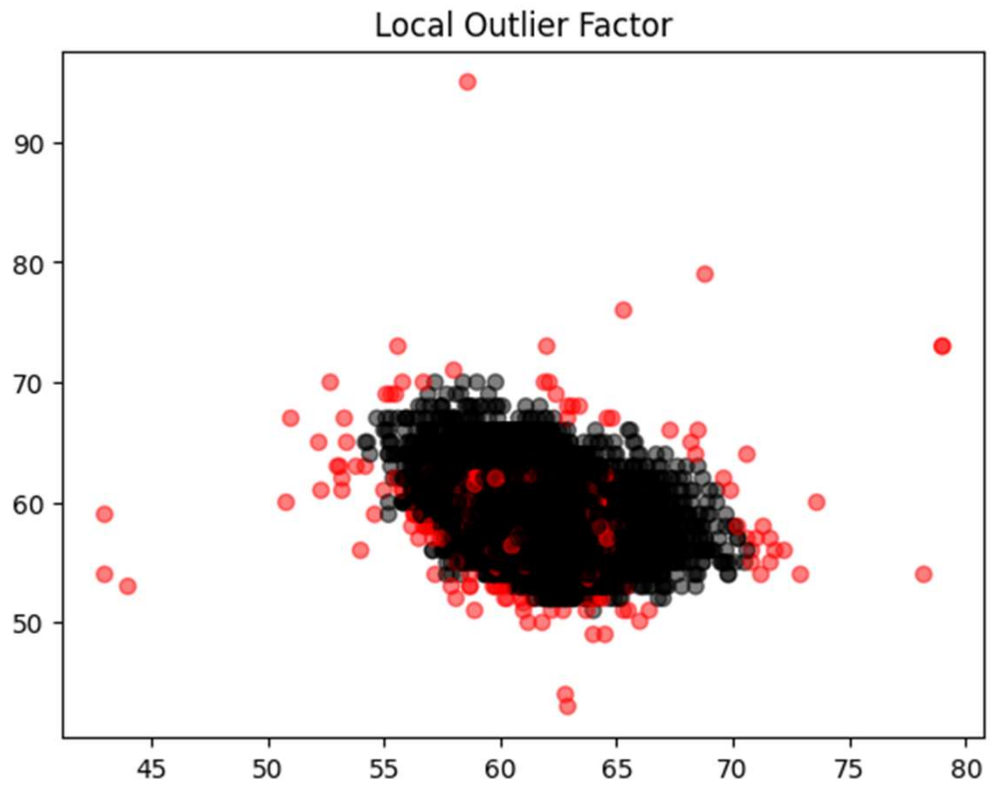
Python: LOF

```
[217] od_lof = LocalOutlierFactor(novelty=True)
      od_lof.fit(X)
```

```
▾ LocalOutlierFactor
LocalOutlierFactor(novelty=True)
```

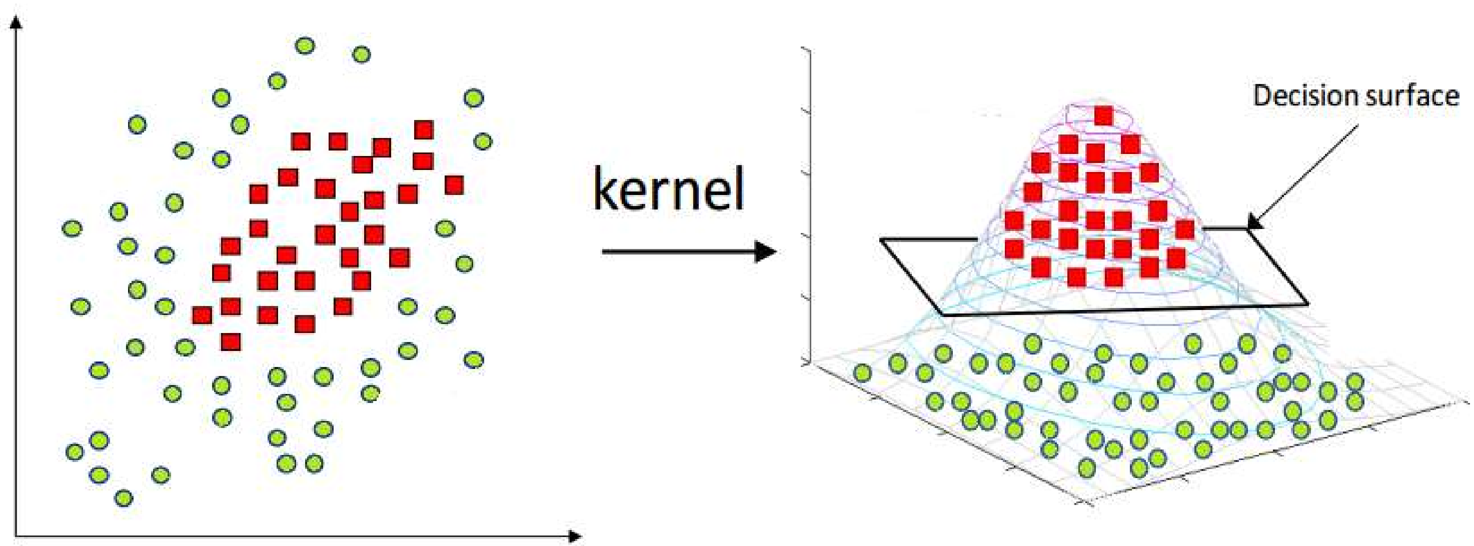
```
[221] od_lof_res = od_lof.predict(X)
      od_lof_res
```

```
/usr/local/lib/python3.10/dist-packages/sk
  warnings.warn(
array([1, 1, 1, ..., 1, 1, 1])
```



One-class SVM

- SVM uses kernel to transform data into another domain.
- It can be applied for detecting novelties



Python: One-class SVM

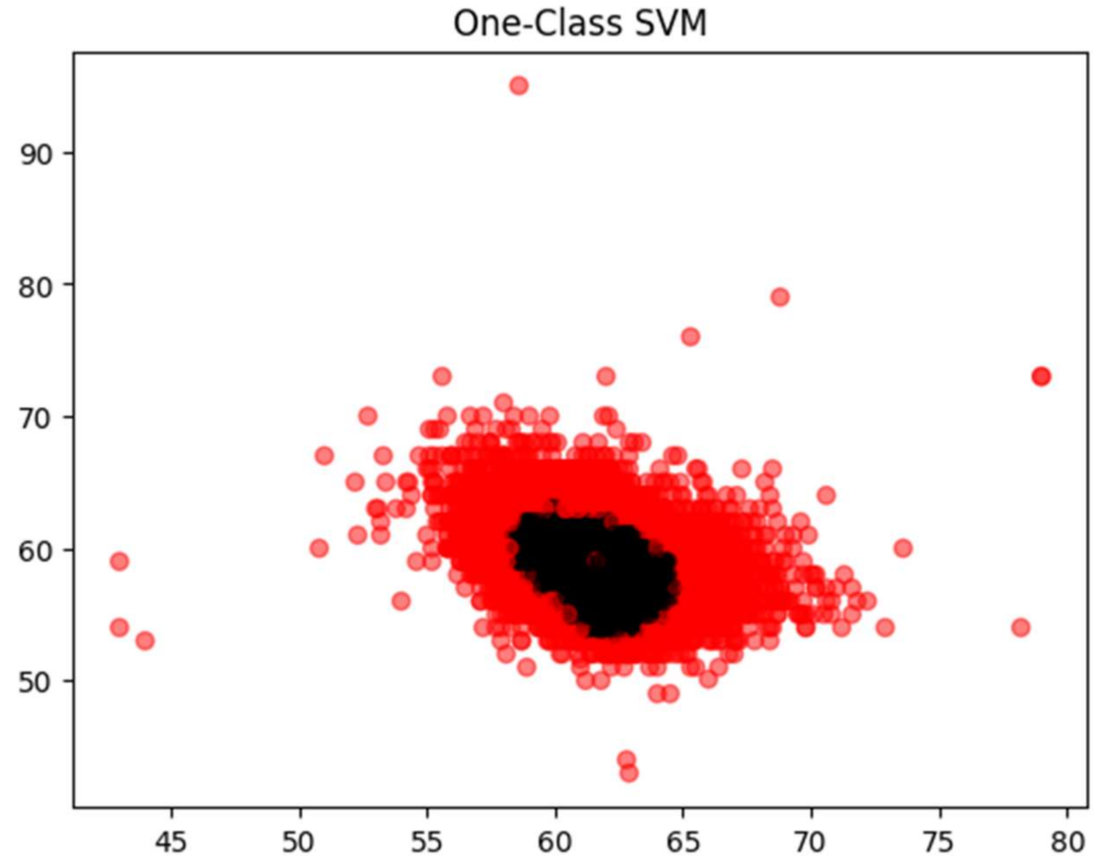
```
[244] od_svm = OneClassSVM(tol=0.1, nu=0.1, gamma=0.1)
      od_svm.fit(X)
```

```
▾ OneClassSVM
OneClassSVM(gamma=0.1, nu=0.1, tol=0.1)
```

```
[245] od_svm_res = od_svm.predict(X)
      od_svm_res
```

```
array([ 1,  1, -1, ...,  1,  1,  1])
```

Python: One-class SVM



Challenges

- No labels
- Unknown contamination level

Summary

- Concepts of anomaly detection
- Statistical analysis approach
- ML-based approach

QA

BIG DATA
EXPERIENCE
CENTER