Mining contextual outliers

Introduction

- **Contextual Outlier**: An object is considered a contextual outlier if it significantly deviates with respect to a specific context.
- Context is defined by contextual attributes (e.g., location, time).
- Behavioral Attributes are used to determine outlier-ness in the specified context.
- Example: A temperature of 28°C is an outlier in winter in Toronto but not in summer.
  - Contextual Attributes: Time, location.
  - Behavioral Attributes: Temperature value.
- Contextual outlier detection focuses on the relationship between context and behavior, unlike general outlier detection.

1. Transforming contextual outlier detection to conventional outlier detection

Transforming Contextual Outlier Detection to Conventional Outlier Detection

- This category of methods is for situations where the contexts can be clearly identified.
- For a given data object, we can evaluate whether the object is an outlier in two steps.
  - Step 1: Identify the context of the object using the contextual attributes
  - Step 2: Calculate the outlier-ness score for the object in the context using a conventional outlier detection method.

Example: Contextual Outlier Detection

Scenario: Electronics store managing customer attributes:

- All Attributes: age_group, postal_code, number_of_transactions_per_year, annual_total_transaction_amount.
- Contextual Attributes: age_group and postal_code.
- Behavioral Attributes: number_of_transactions_per_year and annual_total_transaction_amount.

Steps:

1. For a given customer (c), use **contextual attributes** (age group and postal code) to find their context group (e.g., "25–45 years old in postal code 12345").
2. Compare customer c's behavior (e.g., transactions per year and transaction amount) with others in the **same group**.

Addressing Granularity and Challenges in Contextual Outliers

- Granularity of Contexts can vary:
  - Broad: Age group and town.
  - Detailed: Age, postal code, number of transactions per year.
- **Challenge**: Sparse Contexts
  - What if a context has very few or no other customers?
  - Few or no peers in the same context make evaluation unreliable.
  - For instance, a group might have no other customers with the same age and postal code, making comparisons unreliable.

Solution: Generalize the Context

- Broaden the context by grouping customers with similar normal behaviors.
  - Example: Customers of the same age group who live in nearby postal codes may share common spending patterns.
- This allows for better comparisons even in sparse contexts.
- This approach is called **Mixture Models**

Using Mixture Models

- Mixture Models are used to generalize and map:
  - U: Clusters based on contextual attributes (e.g., age group and postal code).
  - V: Clusters based on behavioral attributes (e.g., spending patterns).
- The outlier score is computed using probabilities:
  - How likely is customer c to belong to each contextual cluster?
  - How likely is c's behavior within each behavioral cluster?

Steps in using Mixture Models:

1. Creating Clusters Based on Two Types of Attributes
2. Linking the Two Types of Clusters
3. Combining information about contextual clusters and behavioral clusters

Step 1: Creating Clusters Based on Two Types of Attributes

- **Mixture Model U:** Clustering Contextual Attributes
  - Imagine grouping customers based on **contextual attributes** like "age group" and "postal code."
  - For example, you might create clusters such as:
    - **Cluster U1:** Customers aged 25-45 in postal codes starting with 123.
    - **Cluster U2:** Customers aged 45-65 in postal codes starting with 456.
- **Mixture Model V:** Clustering Behavioral Attributes
  - Separately, you group customers based on **behavioral attributes** like "number of transactions" and "total transaction amount."
  - For example, you might create clusters such as:
    - **Cluster V1:** Customers with high transaction amounts but few transactions.
    - **Cluster V2:** Customers with many small transactions.

Step 2: Linking the Two Types of Clusters

- Mapping $p(V_i|U_j)$
  - This captures how likely it is that a customer in **Cluster Uj (context)** belongs to **Cluster Vi (behavior)**
- Examples:
  - Customers in U1 (age 25-45, postal code 123) tend to exhibit behaviors similar to V2 (many small transactions) with a probability of 80%.
  - Customers in U2 (age 45-65, postal code 456) tend to exhibit behaviors similar to V1 (high-value transactions, few in number) with a probability of 90%.

Step 3: Combining information about contextual clusters and behavioral clusters

The outlier score $S(o)$ for a customer o is:

$$S(o) = \sum_{U_j} p\left(o \in U_j\right) \sum_{V_i} p(o \in V_i) p\left(V_i \middle| U_j\right)$$

- $p\left(o \in U_j\right)$ Probability that o belongs to a specific contextual cluster.
- $p(o \in V_i)$ Probability that o belongs to a specific behavioral cluster.
- $p\left(V_i \middle| U_j\right)$ Probability that a behavioral cluster V_i aligns with a contextual cluster U_j.

- We use these **probabilities** to calculate how unusual a customer's behavior is, given their context.
- By combining information about **contextual clusters** and **behavioral clusters** through the mapping, the method can better detect unusual or unexpected behaviors in a meaningful way.
- For instance: If a 25-45-year-old customer in postal code 123 is behaving like V1 (high-value transactions), but most customers in their context (U1) behave like V2, that customer might be flagged as an outlier.

2 Modeling normal behavior with respect to contexts

Challenges in Defining Contexts

- Inconvenience of Partitioning Data:
  - Some applications make it difficult to clearly separate data into distinct contexts.
- Example—Online Store Browsing Behavior:
  - Customers have sequences of searched and browsed products.
  - Determining the context (how many previous products to consider) is unclear and varies per product.

Modeling Behavior with Contexts

- **Step 1**: Use training data to build models that predict expected behavior attributes based on contextual attributes.
- By linking context and behavior through modeling, we can avoid explicit context definition manually
- **Step 2**: Apply the predictive model on new data objects' contextual attributes.
- **Step 3**: If the actual behavior significantly deviates from the model's prediction, the object is flagged as a contextual outlier.

Example—Online Store Browsing Behavior

- Data: A sequence of products browsed by a customer in a session:
  - Example: [{Product A, Category: Kitchen}, {Product B, Category: Kitchen}, {Product C, Category: Kitchen Appliances}]
  - A final purchase:
    - Example: {Product X, Category: Electronics}.
- Context = {Browsing History, Time Spent}, Behavior = {Purchase Category}.
- Goal: Detect if the final purchase is a contextual outlier compared to the customer's browsing behavior.

Step 1: Train a Predictive Model

- Use historical data from many customers to train a model that predicts the behavioral attribute (final purchase category) based on contextual attributes (browsing sequence).
- The model learns patterns like:
  - If a user browses Kitchen and Kitchen Appliances, they are likely to purchase from categories such as Kitchen Appliances or Home Improvement.
  - Users browsing Electronics categories are likely to purchase Electronics products.
- Example of possible models:
  - Neural Networks or Regression Models: Use browsing features to predict the likelihood of each product category being purchased.

Step 2: Apply the Model

- When a customer makes a purchase, apply the trained model to the contextual attributes (their browsing behavior).
- The model predicts a probability distribution over possible product categories:
- Example Prediction:
  - Kitchen Appliances: 70%
  - Home Improvement: 25%
  - Electronics: 1%
  - Other: 4%.

Step 3: Detect Outliers

- Compare the actual purchased category (Electronics) to the predicted probabilities.
- If the actual behavior (purchase of Electronics) significantly deviates from the predicted probabilities (e.g., very low likelihood of Electronics), flag it as a contextual outlier.

Benefits of This Approach

- It avoids manually defining the "context" (e.g., specifying rules like "use the last 3 browsed items").
- The model dynamically learns how past browsing influences likely purchases, adapting to different users and contexts.
- Outlier detection becomes contextual:
  - Browsing Kitchen Appliances → Purchase of Electronics = Outlier.
  - Browsing Electronics → Purchase of Electronics = Normal.