भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Indian Institute of Technology Hyderabad

Fraud Analytics

CS6890

Assignment - 3

Identifying outliers in the data by using Variational Autoencoders

Team Members

CS24MTECH14013	Mohit Manoj Patil
CS24MTECH14010	Veeresh Shukla
CS24MTECH12018	Deeba Afridi

Contents

List of Figures

1 Problem Definition

The primary objective is to identify anomalous data points, or outliers, within a given dataset (`data.csv`). Outliers are data points that deviate significantly from the general pattern or distribution of the majority of the data. The specific problem addressed by the provided script (`Fraud_Assignment_3.py`) is to implement an unsupervised outlier detection methodology that leverages deep learning for dimensionality reduction and traditional clustering for pattern identification.

The core tasks are:

1. **Dimensionality Reduction:** Compress the high-dimensional input data into a lower-dimensional latent space using a Variational Autoencoder (VAE). The VAE should capture the most salient features and underlying structure of the data.
2. **Clustering in Latent Space:** Apply the K-Means clustering algorithm to the low-dimensional latent representations generated by the VAE. This groups data points based on their proximity in the compressed space.
3. **Outlier Identification:** Identify data points that are likely outliers based on their position relative to the identified clusters in the latent space. The specific method implemented defines outliers as points that are furthest from their assigned cluster's centroid, determined by a distance percentile threshold. This implicitly targets points on the boundaries of clusters or potentially points belonging to very small, sparse clusters which tend to have larger distances to their centroids.

2 Expected Outcome

The successful execution of the implemented pipeline aims to achieve the following:

- **Effective Data Compression:** The VAE should learn a meaningful, lower-dimensional representation of the input data that preserves significant structural information relevant for identifying anomalies.
- **Meaningful Clustering:** K-Means, applied to the latent space, should partition the data into distinct groups reflecting underlying patterns.
- **Robust Outlier Identification:** The distance-based threshold method should effectively flag data points that are genuinely atypical compared to the core members of their respective clusters in the latent space.
- **Model Selection:** The process should systematically evaluate different VAE configurations (latent dimensions) and K-Means cluster counts to find an optimal combination based on a quantitative metric (Silhouette Score).
- **Clear Visualization:** For optimal models resulting in a 2D latent space, a scatter plot should visualize the clusters, their centroids, and the identified outliers, providing intuitive insight into the results.
- **Actionable Output:** A list (CSV file) of the identified outlier records, including their original features and relevant metadata (assigned cluster, distance to centroid), should be generated for further analysis or investigation.

3 Dataset Specification

3.1 Input Data File

The primary data source is specified as `data.csv`. The script (`Fraud_Assignment_3.py`) includes functionality to generate synthetic dummy data (1000 samples, 10 features, structured with clusters and outliers) if this file is not found. The analysis described in this report appears to be based on execution with an existing `data.csv`, represented by the provided data snippet.

3.2 Features Used

Based on the provided data snippet which shows the header row and initial data points, the features utilized for the analysis are the numeric columns present in the input file. These columns are explicitly identified as:

- `cov1`
- `cov2`
- `cov3`
- `cov4`
- `cov5`
- `cov6`
- `cov7`
- `sal_pur_rat`
- `igst_itc_tot_itc_rat`
- `lib_igst_itc_rat`

The data snippet confirms these columns contain numerical (floating-point) values, including positive, negative, and zero values. The script's default behavior, likely used here, is to automatically select all available numeric columns if specific ones are not provided during the detector's initialization. Therefore, these 10 columns form the input feature space for the VAE model.

3.3 Data Volume

The exact number of records (rows) processed depends on the full contents of the actual `data.csv` file used during the execution that produced the results.

3.4 Preprocessing Steps

The following preprocessing steps, as defined within the `VAEKMeansOutlierDetector` class in the script, are applied to the 10 selected numeric features:

- **Feature Selection:** The 10 numeric columns listed above are automatically selected and used for the analysis.
- **Missing Value Handling:** Although the small provided snippet does not explicitly show missing values (e.g., NaN), the script is designed to handle them. Any potential NaNs within these 10 columns would be imputed using the mean value calculated from the non-missing values in that respective column.
- **Scaling:** The selected numeric features undergo standardization using `sklearn.preprocessing.StandardScaler`. This process transforms the data for each feature to have a mean of approximately zero and a standard deviation of approximately one. This scaling is crucial for the optimal performance of both the VAE neural network (ensuring features with larger ranges don't dominate the learning process) and the K-Means algorithm (which relies on Euclidean distance).

4 Algorithm

The outlier detection pipeline employs a combination of algorithms:

1. **Variational Autoencoder (VAE):**
   - *Type:* An unsupervised deep learning generative model.
   - *Architecture:* Consists of an Encoder, a Sampling Layer, and a Decoder.
     - *Encoder:* Maps input x to latent distribution parameters (z_{mean}, $z_{\log \text{variance}}$).

- *Sampling Layer:* Samples latent vector z using the reparameterization trick: $z = z_{\text{mean}} + \exp(0.5 \times z_{\log \text{variance}}) \times \epsilon$, where ϵ is random noise.
- *Decoder:* Reconstructs input x from latent vector z.

- *Loss Function:* Minimizes a combination of Reconstruction Loss and **Kullback-Leibler (KL) Divergence Loss** (regularizer).
- *Purpose:* To learn a compressed, meaningful representation of the data in the latent space z.

2. **K-Means Clustering:**
   - *Type:* An iterative partitioning clustering algorithm.
   - *Goal:* Partition n data points into k distinct clusters based on proximity to cluster centroids.
   - *Process:* Iteratively assigns points to the nearest centroid and updates centroid positions as the mean of assigned points.
   - *Purpose:* To group similar data points together in the VAE's latent space.
3. **Elbow Method:**
   - *Type:* A heuristic for determining a suitable number of clusters k.
   - *Process:* Runs K-Means for a range of k values, calculates a clustering metric (like average distance to centroid), and plots it against k. The "elbow" point suggests a good k.
   - *Purpose:* To provide an informed starting point for k.
4. **Silhouette Score:**
   - *Type:* A metric for evaluating cluster quality.
   - *Calculation:* Measures how similar a point is to its own cluster compared to other clusters. Calculated as $(b - a)/\max(a, b)$, where a is mean intra-cluster distance and b is mean nearest-cluster distance.
   - *Interpretation:* Ranges from -1 to 1. Values near +1 indicate dense, well-separated clusters.
   - *Purpose:* To quantitatively compare clustering results and select the best VAE/K-Means configuration.
5. **Distance-Based Outlier Score:**
   - *Process:* Calculates the Euclidean distance between each point (in latent space) and its assigned cluster centroid. Determines a threshold based on a high percentile (e.g., 95th) of these distances.
   - *Identification:* Points whose distance exceeds the threshold are flagged as outliers.
   - *Purpose:* To identify points unusually far from the center of their group in the latent representation.

5 Approach

The `VAEKMeansOutlierDetector` class implements the following step-by-step pipeline:

1. **Initialization:** Configure the detector with file path, feature columns (optional), VAE/K-Means parameters (`latent_dim_options`, `max_clusters_to_try`, `vae_epochs`, `vae_batch_size`), `outlier_percentile`, and `output_dir`.
2. **Load and Prepare Data (`_load_and_prepare_data`):** Read CSV, select numeric features, impute NaNs (mean), and standardize data (`StandardScaler`).
3. **Model Training and Selection (`_train_evaluate_and_select_best_model`):**
   - Loop through each `latent_dim` in `latent_dim_options`.
   - Train a VAE for the current `latent_dim`.
   - Encode data into the latent space.
   - Use the Elbow Method (`_find_optimal_clusters_elbow`) to suggest `suggested_k`.
   - Loop through `k_clusters` around `suggested_k`.

- Perform K-Means for each `k_clusters`.
- Calculate the Silhouette Score.
- Keep track of the configuration (`latent_dim`, `k_clusters`) yielding the highest Silhouette Score, storing the corresponding VAE model, K-Means model, and latent representation.

4. **Outlier Identification (`_identify_outliers`):**
   - Use the best K-Means model and latent representation.
   - Calculate distance from each point to its assigned centroid.
   - Determine the distance threshold using `outlier_percentile`.
   - Identify indices of points exceeding the threshold.
   - Create `outlier_details_df` with original data, assigned cluster, and distance.
5. **Visualization (`visualize_latent_space`):**
   - If the best latent dimension is 2, generate a scatter plot of the latent space.
   - Color points by cluster, mark centroids ('X'), and highlight outliers (red circles).
   - Save the plot (e.g., `clusters_and_outliers_2d.png`).
6. **Save Outliers (`save_outliers`):** Save the `outlier_details_df` to a CSV file (e.g., `identified_outliers.csv`).
7. **Run Pipeline (`run_detection_pipeline`):** Execute steps 2-6 sequentially, printing progress and results.

6 Results and Analysis

Based on the execution producing the provided plot and outlier CSV file:

6.1 Model Selection

The script selected a VAE with a **latent dimension of 2** and determined that **3 clusters** ($k = 3$) provided the best clustering quality in that latent space, as measured by the Silhouette Score. This choice enables the 2D visualization.

6.2 Latent Space Visualization (`clusters_and_outliers_2d.png`)

- The plot clearly shows the result of dimensionality reduction (original 10 features to 2 latent dimensions) and clustering.
- Three clusters (blue - Cluster 0, orange - Cluster 1, green - Cluster 2) are visible and reasonably well-separated, although with some overlap.
- The black 'X' markers accurately represent the calculated centroids of these clusters in the latent space.
- The points circled in red are the identified outliers. They are visually confirmed to be points primarily located on the periphery of their respective clusters or somewhat detached from the main mass. This aligns with the definition of outliers based on exceeding the 95th percentile distance to the cluster centroid in this latent space. Outliers are found across all three clusters.

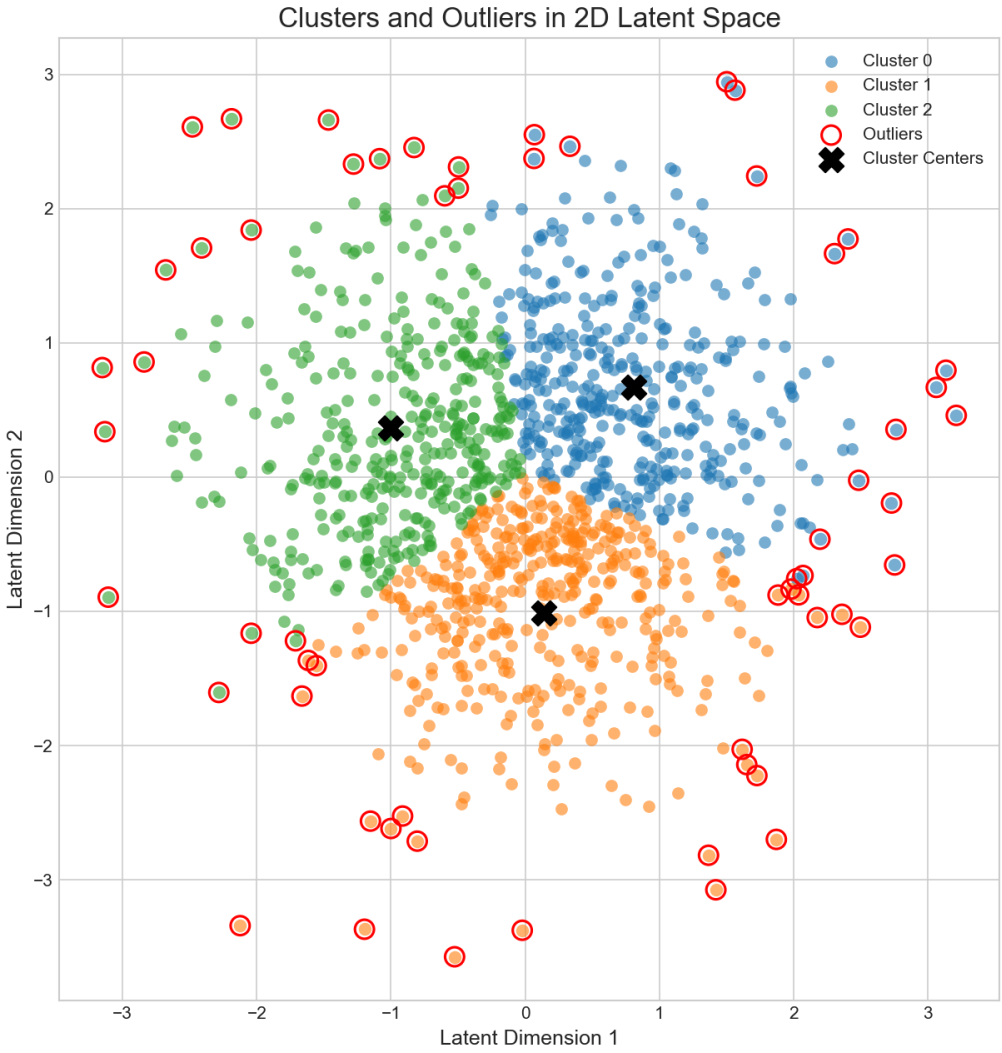

Figure 1: Clusters and Outliers

6.3 Outlier Data (`identified_outliers.csv`)

- This file provides the concrete list of records flagged as outliers, containing their original feature values.
- It includes two crucial additional columns derived from the analysis:
  - `latent_cluster`: The cluster (0, 1, or 2) assigned to the outlier by K-Means in the 2D latent space.
  - `distance_to_centroid`: The calculated Euclidean distance in the 2D latent space that caused the point to exceed the 95th percentile threshold.
- The number of outliers corresponds to approximately 5% of the dataset, consistent with the 95th percentile threshold used.