Probabilistic Machine Learning Project: Anomaly Detection in Stock Data using Variational Autoencoders

Course of AA 2023-2024 - Data Science and Artificial Intelligence

Andrea Buscema
`SM3800034`

Tanja Derin
`SM3800013`

Contents

1 Introduction

In the finance sector, quickly identifying irregularities in stock data is crucial for maintaining market stability and preventing significant financial losses. Variational Autoencoders (VAEs) have shown promise in identifying these anomalies due to their ability to model data distributions in complex environments.

VAEs operate by encoding data into a compressed, latent representation and then decoding this representation back to the original input dimension. The reconstruction error—how well the model can reconstruct the original input from the latent encoding—serves as a critical metric. In the context of anomaly detection, data points with high reconstruction errors are typically considered anomalies as they deviate significantly from the model's learned representation. [2]

This project aims to explore the application of VAEs to detect unusual patterns in stock data, which could offer substantial improvements over conventional techniques.

1.1 Autoencoders in Anomaly Detection

An autoencoder is a type of neural network. The basic model consists of two main parts: the encoder and the decoder.

- Encoder: Compresses the input into a lower-dimensional representation. This encoding captures the most significant features of the input data necessary for reconstruction.
- Decoder: Attempts to reconstruct the input data from the compressed encoding.

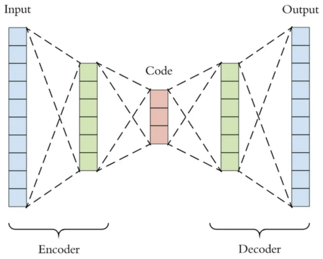


Figure 1: A flow chart of the autoencoder.

Autoencoders can be effectively applied in anomaly detection due to their ability to reconstruct input data. Anomalies manifest as data points that exhibit higher reconstruction errors compared to normal data.

However, classic autoencoders often fail to produce very useful, well-structured latent spaces due to the discontinuity of these spaces [1]. To address these limitations and achieve a more continuous latent space, we explore the use of Variational Autoencoders.

1.1.1 Variational Autoencoders

Variational Autoencoders (VAEs), introduced by Kingma and Welling in their 2014 paper [3], are an extention of classical autoencoders, since they introduce a probabilistic twist, making them capable not only of efficient dimensionality reduction but also of generating new data instances that resemble the training data.

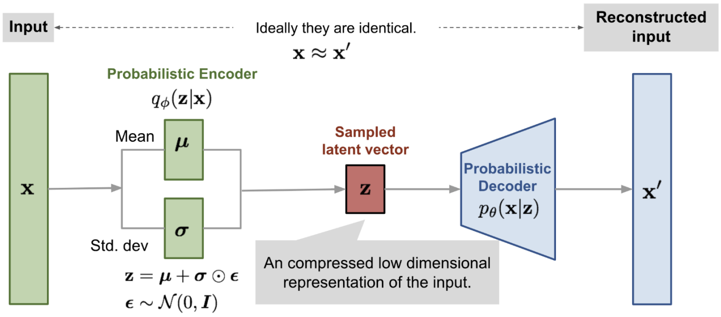

Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
$\mathbf{x}$
$\boldsymbol{\sigma}$
Std. dev
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

Figure 2: VAE with the encoder colored green, latent space colored red, and the decoder colored blue.[9]

VAEs operate on the principle of encoding input data X into a latent variable z. Instead of directly encoding an input into a fixed vector, VAEs map inputs into a distribution characterized by parameters (mean and variance). This approach originates from the intent to represent the probability distribution $P(X)$ of the data through its latent representations. The main components of the VAE architecture are

- $P(z)$: The prior distribution over the latent variables, typically assumed to be a standard normal distribution.
- $P(X|z)$: The likelihood of the data given the latent variables, modeled by the decoder.
- $P(z|X)$: The posterior distribution of the latent variables given the data, which is approximated by $Q(z|X)$ using variational inference.

Variational inference turns the problem into an optimization problem by introducing a tractable approximation $Q(z|X)$ to the intractable true posterior $P(z|X)$, where we minimize the Kullback-Leibler (KL) divergence between the approximate posterior $Q(z|X)$ and the true posterior [4]. Starting from the KL Divergence:

$$D_{KL}[Q(z|X)\|P(z|X)] = \mathbb{E}[\log Q(z|X) - \log P(z|X)]$$

We can derive the ELBO as shown in class:

$$ELBO = \mathbb{E}[\log P(X|z)] - D_{KL}[Q(z|X)\|P(z)]$$

Thus, the VAE loss function, or the negative of the ELBO, is given by:

$$L(\theta, \phi; X) = -\mathbb{E}_{z\sim Q_\phi}[\log P_\theta(X|z)] + D_{KL}[Q_\phi(z|X)\|P_\theta(z)]$$

Where, θ and ϕ denote the parameters of the decoder and encoder networks, respectively. We can modify the loss function to explicitly control the trade-off between the reconstruction fidelity and the regularization of the latent space by including some weights. The modified ELBO loss function becomes:

$$L(\theta, \phi; X) = \gamma \cdot \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - \beta \cdot D_{KL}(q_\phi(z|x)\|p_\theta(z))$$

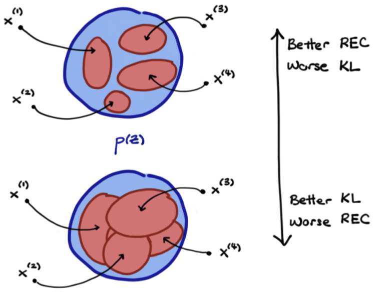


Figure 3: This diagram illustrates the interplay between reconstruction loss and KL-divergence in a Variational Autoencoder (VAE) [7]

- Increasing γ prioritizes better reconstruction
- Increasing β emphasizes the regularization effect, promoting a latent space that closely matches the assumed prior, typically a standard Gaussian.

1.1.2 The Reparameterization Trick

The reparameterization trick, discussed in class, is a key technique in training Variational Autoencoders. This method allows for the computation of gradients with respect to the parameters of the approximate posterior $Q(z|X)$, which is crucial for optimizing the ELBO.

The key part of the this trick is expressing z as a deterministic function of ϵ, a noise variable from a simpler, parameter-independent distribution $p(\epsilon)$, typically a standard Gaussian $N(0, 1)$. This transformation is defined as:

$$z = g_\phi(x, \epsilon), \quad \text{where} \quad \epsilon \sim p(\epsilon)$$

By shifting randomness away from the model parameters, it ensures that the gradients are more informative and less noisy, leading to more robust learning outcomes.

1.2 Anomaly Detection Strategies in VAE

Supervised Anomaly Detection: Requires labeled data for both normal and anomalous instances.The VAE is trained explicitly to differentiate between the two classes by leveraging labeled examples.
Semi-Supervised Anomaly Detection: Assumes access to labeled data for normal instances but not for anomalies. The VAE learns to reconstruct normal data and any significant deviation during testing suggests an anomaly.
Unsupervised Anomaly Detection: No labels are required. Focuses on learning the distribution of the training data assumed to be mostly normal.In the unsupervised context, which is our focus, two primary techniques are employed to identify anomalies [6]:

- Reconstruction Quality: The VAE learns to reconstruct inputs as closely as possible. After training, the reconstruction error can be used as an anomaly score. Anomalies are likely to have higher reconstruction errors because the model is trained predominantly on normal data and thus struggles to reconstruct outliers accurately.
- Anomalous Representation in Latent Space: This section examines how data points are represented in the latent space.
  - Clustering in Latent Space (μ_z space): Involves grouping the latent representations based on the mean vector μ_z. Techniques such as hierarchical clustering and k-means are applied to identify clusters that predominantly consist of normal or anomalous samples.
  - Wasserstein Distance-Based Detection: This method utilizes the Wasserstein distance to measure the variability among samples in the latent space. Given that anomalous samples often exhibit higher variability, this metric can effectively highlight anomalies.

Algorithm 1 VAE Anomaly Detection

```
Input: Test dataset $X_{test}$
Output: Anomaly detection results
Initialize lists: $recon$, $errors$, $means$, $log\_vars$
for each batch $x$ do
    $(reconstructed, z\_mean, z\_log\_var) \leftarrow VAE(x)$
    $error \leftarrow \text{mean}((x - reconstructed)^2)$
    Append $reconstructed$, $error$, $z\_mean$, $z\_log\_var$ to lists
end for
$threshold \leftarrow \text{percentile}(errors, 95)$
$anomalies \leftarrow errors > threshold$
```

2 Problem Statement

In the project we tried two applications of Variational Autoencoders to detect anomalies in stock data. First, approach utilizes an LSTM-VAE model analyzing 30 years of AAPL stock data to capture deeper, more complex temporal patterns in anomalies. The second a Conditional VAE is used on the past five years of Dow 30 stocks data, conditioned on stock symbols. Comparing the findings from these two models helps validate the effectiveness of VAEs in recognizing anomalies in stock behavior over different periods.

3 Proposed Solution

3.1 LSTM-based VAE

3.1.1 Theoretical explanation

We employed a hybrid model combining Variational Autoencoders and Long Short-Term Memory (LSTM) networks to analyze anomalies in stock data. VAEs are providing a robust framework for grabbing local features which capture the majority of training data characteristics. However, their limitation in capturing longer-term dependencies is addressed by integrating an LSTM layer. LSTMs excel in recognizing long-term trends and dependencies in sequential data. See similar work in [5].

The LSTM layers implemented into the CVAE architecture is to effectively encode and decode sequential data that exhibits temporal dependencies. The procedure is described in the following steps

1. Encoder with LSTM: The LSTM layer processes the input sequences for learning the temporal structures within the data. It output is then utilized to determine the parameters of the latent space. Specifically, the last hidden state of the LSTM is fed into dense layers that produce mean of the latent distribution and log variance.

2. Sampling Mechanism: The latent vector z is generated through a sampling process which uses the reparameterization trick.

3. Decoder with LSTM: A decoder LSTM then takes this sampled input to reconstruct the sequence. This layer mirrors the encoder's LSTM but operates in reverse, transforming latent representations back into the original data format.

This architecture allows the model not only to generate data that adheres to the temporal patterns in the training set but also to potentially flag anomalies when the reconstructed data significantly deviates from the expected sequence patterns, based on the learned distribution in the latent space.

3.1.2 Data and Procedure

The dataset consists of daily stock prices for Apple Inc. (AAPL), spanning from January 2, 1990, to December 29, 2023. The 'Date' column has been converted into datetime format, and all numerical features have been normalized between 0 and 1. This model uses a sequence-based approach, generating 8,505 overlapping windows, each 60 timesteps long with 5 features per timestep, to capture and learn from the temporal patterns in the data.

The VAE-LSTM model first encodes input sequences to a lower-dimensional latent space, capturing temporal dependencies and main features. It then reconstructs the input from this latent representation. Anomalies are identified by calculating the mean squared error between the original sequences and their reconstructions. Different thresholds (90th, 95th, 97th, and 99th percentiles) are set to differentiate between normal variations and true anomalies.

3.1.3 Results

The visualizations display stock prices, annotated with detected anomalies at different percentile thresholds (95th and 99th). Anomalies identified at the 95th percentile are more frequent and may correspond to less significant market events, while anomalies at the 99th percentile highlight more significant or rare market events.

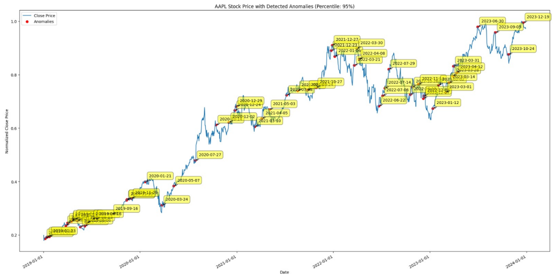


Figure 4: Anomalies detected: 95 percentile

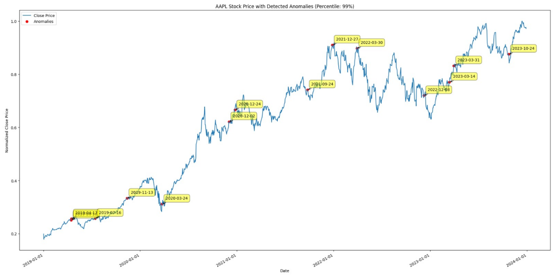


Figure 5: Anomalies detected: 99 percentile

It was used t-SNE (t-Distributed Stochastic Neighbor Embedding) to visualise the latent space representation of data, that is a technique for dimensionality reduction and visualization of high-dimensional data.

From the plots of the latent space we are able to understand how well the model has learned to represent the data. It can be also helpful in identifying clusters in the data. Each point represents a stock price data point that has been encoded into a latent space by the VAE-LSTM model.

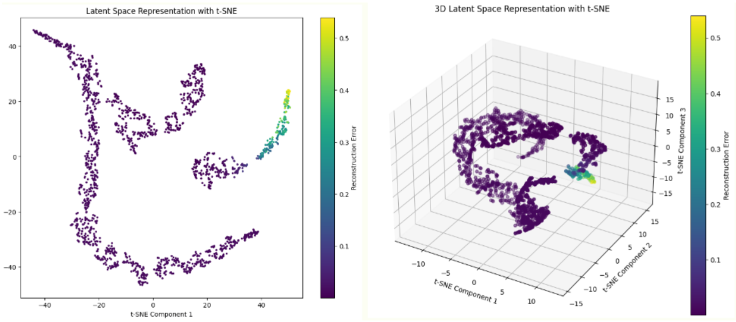


Figure 6: Latent Space

- The color gradient represents the reconstruction error, with higher values indicating potential anomalies. The presence of distinct clusters in the latent space suggests that the VAE has learned to group similar sequences together.
- The purple points represent normal data points where the reconstruction error is low, indicating that the model has accurately reconstructed these points from the latent space.
- The yellow and green points represent anomalies where the reconstruction error is higher, suggesting unusual or abnormal patterns in the stock prices

From key findings of the anomaly detection analysis on Apple stock data, highlighting potentially significant events that impacted the company's performance (using 99th percentile threshold to identify anomalies), it's possible to extract some dates that likely founds corresponding major events or news that affected Apple's stock price or trading patterns.

The headlines suggest that the anomalies detected could be related to various factors, including company-specific decisions, broader market trends, analyst ratings, regulatory challenges, macroeconomic events, Covid-19 world pandemic.

The combination of the anomaly detection results and the news headlines demonstrates the model's ability to identify dates of significant market movements or company events.

3.2 CVAEs for Stock Data

3.2.1 Theoretical explanation

Conditional Variational Autoencoders (CVAEs) extend the traditional one by conditioning both the encoder and the decoder on additional variables c, representing attributes or labels of the data. This modification allows the model to learn representations conditioned on auxiliary information, enhancing the flexibility and applicability of the generative process.

- Encoder: Models $Q(z|X, c)$, taking into account both the input data and the conditioning variable.
- Decoder: Decoder becomes $P(X|z, c)$, aiming to reconstruct the input based on the latent variables and the conditioning context.
- ELBO: The objective function in CVAEs is adjusted to account for the conditioning variable, and is expressed as

$$\log P(X|c) - D_{KL}[Q(z|X,c)\|P(z|X,c)] = \mathbb{E}[\log P(X|z,c)] - D_{KL}[Q(z|X,c)\|P(z|c)]$$

This equation states that the ELBO now conditions all the distributions with c, making the latent variable and output generation dependent on the conditioning variable. This adjustment allows the model to generate different outputs for the same input based on different values of C. [8] CVAEs can generate specific types of outputs based on the condition imposed.

3.2.2 Data and Procedure

The dataset is stock market data from the $S\&P500$ index covering a period of five years from January 2, 2019, to December 29, 2023. Each entry in the dataset for a specific stock symbol includes the following features:

- Date: The date of trading.
- Symbol: The stock ticker symbol.
- Adj Close: The closing price of the stock adjusted for dividends and splits.
- Close: The closing price of the stock for the trading day.
- High: The highest price of the stock during the trading day.
- Low: The lowest price of the stock during the trading day.
- Open: The opening price of the stock.
- Volume: The number of shares traded during the trading day.

To ensure data and integrity, missing values were addressed through forward and backward filling methods. The original data was enriched through the extraction of additional features:

- Temporal Features: Days of the week and month were derived from the date to capture potential temporal cycles affecting stock prices.
- Technical Indicators: Several metrics were computed to provide deeper insights like: Moving Averages for 3, 5, and 10 days, Volatility, Volume Changes, Daily Price Changes and Relative Position, indicating where the closing price sat within the daily high-low range

All numerical features were normalized to a range of 0 to 1 facilitating more stable and faster convergence during training.

To condition the CVAE model on the specific stock symbols represented in the dataset, an important step was transforming the 'Symbol' column into a one-hot encoded format. By doing this the model can condition its latent space generation and reconstruction phases on specific stock symbols, allowing it to learn and generate output that is specific to each stock symbol.

The data was divided into training, validation, and testing sets. The split was such that 80% was used for training to maximize the learning potential, while 20% was used for testing to assess the model's performance on unseen data. From the training set, 10% was a validation set used for tuning model parameters and implementing early stopping techniques during training.

Algorithm 2 Training CVAE

```
Input: Normalized stock dataset $X_{train}$, labels $y_{train}$, validation data $X_{valid}$, labels $y_{valid}$
Output: Trained CVAE model
Initialize model parameters $\phi, \theta$
repeat
    for each batch $(x, y)$ in $X_{train}$, $y_{train}$ do
        Compute z_mean, z_log_var from encoder
        Sample z using reparameterization trick:
        Decode z to reconstruct $\hat{x}$
        Compute reconstruction loss and KL divergence
        Backpropagate to update $\phi, \theta$
    end for
    Validate model on $X_{valid}$, $y_{valid}$
until convergence
Return trained model
```

3.2.3 Results

With the CVAE, we can ask the model to recreate data for a particular label. In the example of stock market data, we can ask it to recreate data for a particular stock symbol-AAPL that we used for anomaly detection analysis. This involved processing the last 100 data points of AAPL stock data and generating features relevant to anomaly detection. The CVAE was then employed to identify reconstruction errors and pinpoint anomalies. A threshold was set

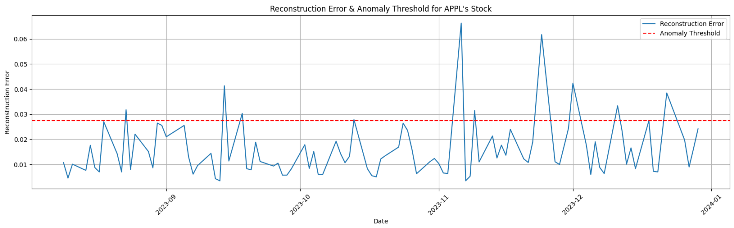


Figure 7: Reconstruction Error and Anomaly Threshold

at the 95th percentile of these errors to define what constitutes an anomaly. The results, visualized through a plot of reconstruction errors over time with a designated anomaly threshold, help to flag significant deviations from typical stock behavior, indicating potential anomalies.

To monitor anomalies across different stock features such as closing prices, trading volumes, and the differences between opening and closing prices a series of plots were generated (see in notebook). Such visualizations are useful for pinpointing specific days with unusual price movements or trading volumes that may require further investigation or could signify critical market events.

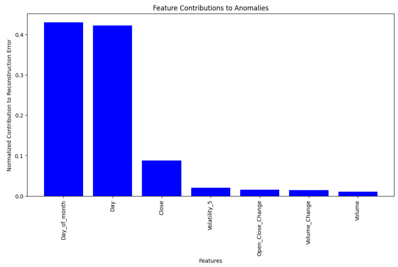


Figure 8: Feature Contribution

The bar graph show how different features contribute to the overall reconstruction error. In this analysis, features like 'Day-of-month' and 'Day' show significantly higher contributions, indicating that temporal aspects of the data are pivotal in defining an anomaly. Meanwhile, other features like 'Close', 'Volatility-5', and various price changes contribute to varying lesser extents.

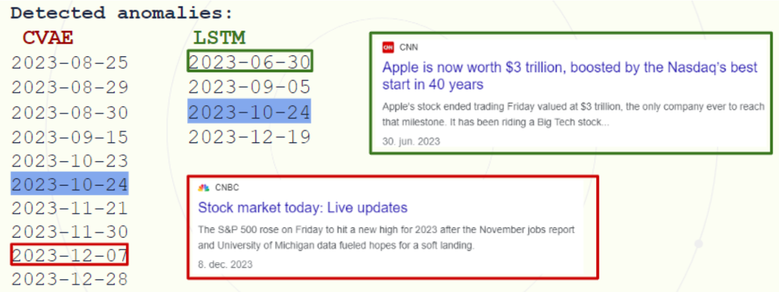
This pattern suggests that anomalies are strongly tied to specific times rather than only to movements in price or volume. As seen in the first model, given the importance of temporal features, employing a model that explicitly captures sequence dependencies like an LSTM enhances the model's ability to discern anomalies.

4 Conclusion

The CVAE, while generalizing across multiple stocks, detected anomalies on days including significant company events and broader market movements. In contrast, the LSTM model, specifically trained on Apple stock, identified anomalies that aligned closely with significant individual stock events.

This experiment underscores the influence of model design on anomaly detection in financial time series. The conditioning on stock symbols in the CVAE allows for a broader perspective, potentially making it better suited for applications needing a generalized model across various stocks. Meanwhile, the LSTM's focus on long-term data from a single stock provide deeper insights into specific stock behaviors but might miss broader market trends affecting other stocks.

In addition the inclusion of LSTM in this framework is critical for capturing the sequence of events leading up to an anomaly, offering a deeper understanding of temporal dynamics in stock price movements.



```
Detected anomalies:
 CVAE            LSTM
2023-08-25      2023-06-30
2023-08-29      2023-09-05
2023-08-30      2023-10-24
2023-09-15      2023-12-19
2023-10-23
2023-10-24
2023-11-21
2023-11-30
2023-12-07
2023-12-28
```

CNN

Apple is now worth $3 trillion, boosted by the Nasdaq's best start in 40 years

Apple's stock ended trading Friday valued at $3 trillion, the only company ever to reach that milestone. It has been riding a Big Tech stock...

30. jun. 2023

CNBC

Stock market today: Live updates

The S&P 500 rose on Friday to hit a new high for 2023 after the November jobs report and University of Michigan data fueled hopes for a soft landing.

8. dec. 2023

Figure 9: Date comparison

References

[1] François Chollet. *Deep Learning with Python*. Manning Publications, 2017.

[2] Cloud Software Group. The ultimate guide to anomaly detection. `https://www.spotfire.com`, 2024. Accessed: 2024-06-15.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2014.

[4] Agustinus Kristiadis. Variational autoencoder. `https://agustinus.kristia.de/techblog/2016/12/10/variational-autoencoder/`, December 2016. Accessed: 2024-06-14.

[5] Shuyu Lin, Ronald Clark, Robert Birke, Sandro Schönborn, Niki Trigoni, and Stephen Roberts. Anomaly detection for time series using vae-lstm hybrid model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4322–4326. Ieee, 2020.

[6] João Pereira and Margarida Silveira. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1275–1282, 2018.

[7] R. Shu. Density estimation: Variational autoencoders. `http://ruishu.io/2018/03/14/vae/`, 2018.

[8] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015.

[9] Lilian Weng. From autoencoder to beta-vae. `https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html`, May 2019. Accessed: 2024-06-14.