МИНОБРНАУКИ РОССИИ
ФГБОУ ВО «СГУ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

ПОИСК АНОМАЛИЙ ВО ВРЕМЕННЫХ РЯДАХ НЕЙРОСЕТЯМИ

РЕФЕРАТ

студента 5 курса 531 группы
направления 100501 — Компьютерная безопасность
факультета КНиИТ
Улитина Ивана Владимировича

Проверил
доцент _____ И. И. Слеповичев

Саратов 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Временным рядом называют последовательность значений, упорядоченных по времени, в которое то или иное значение было получено/зафиксировано. В качестве примеров временного ряда можно выделить:

1. курсы валют/акций (стоимость валюты/акции в конкретный момент времени);
2. значение прогноза погоды, параметры работы двигателя (изменяющееся с течением времени значение некоторой физической величины);
3. электрокардиограмма (биологический параметр человека);
4. передаваемый объём сетевого трафика;

Временной ряд обладает типовыми характеристиками (иногда называемые **компонентами**), которые точно описывают характер временного ряда и в виде совокупности которых временной ряд можно представить:

1. тренд (долгосрочное увеличение или уменьшение значений ряда);
2. сезонность, сезонная вариация (краткосрочное регулярно повторяющееся колебание значений временного ряда вокруг тренда);
3. цикл, циклические колебания (характерные изменения ряда, связанные с повторяющимися глобальными причинами — цикл деловой активности или экономический цикл, состоящий из экономического подъема, спада, депрессии и оживления);
4. остаточная вариация, которая может быть двух видов:
   - аномальная вариация — неестественно большое отклонение временного ряда, которое оказывает воздействие на единичное наблюдение;
   - случайная вариация — малое отклонение, которое невозможно предвидеть.

Вследствие развития нейросетей и искусственного интеллекта, появилась возможность решать задачи, связанные с временными рядами, решение которых могло бы быть полезно для той или иной области жизнедеятельности человека. Особенно полезно это может быть в тех областях, где обычные статистические модели (обычная или интегрированная модель авторегрессии - скользящего среднего, векторная авторегрессия и т.д.) неприменимы или неэффективны.

Среди задач области временных рядов можно выделить несколько наиболее популярных:

1. Прогнозирование следующего значения временного ряда

2. Классификация, кластеризация, поиск паттернов временных рядов
3. Обнаружение выбросов или аномалий во временных рядах
4. Генерация, моделирование или выделение признаков временных рядов
5. Модификация значений временного ряда

Временные ряды могут содержать аномалии. **Аномалией** называется отклонение в стандартном поведении какого-то процесса, который описывается этим временным рядом. По сути её можно определить через аномальную вариацию, являющейся возможной компонентой этого временного ряда. В зависимости от предметной области описываемого процесса в выборке (датасете) могут быть аномалии разного вида. Можно выделить одни из самых распространенных видов аномалий:

1. точечные аномалии (наблюдается отклонение в поведении в отдельных точках);
2. групповые аномалии (группа точек, которая ведет себя аномально, но каждая точка которой отдельно аномальной не является);
3. контекстные аномалии, суть которых в связи аномалии с внешними данными, которые не присущи значениям ряда (например, отрицательная температура на улице летом).

Настоящий реферат посвящен рассмотрению и изучению вопроса поиска аномалий во временных рядах с помощью нейросетей.

Необходимость поиска аномалий может быть обусловлена тем, что обнаружение девиации значения процесса может способствовать улучшению решения более общей задачи, связанной с временным рядом этого процесса. Например, это может быть полезно при решении с помощью статистических моделей/моделей машинного обучения/нейросетей задачи классификации отрезка временного ряда или задачи прогнозирования. Знание того факта, что конкретный отрезок временного ряда содержит аномалию, наличие которой может плохо сказываться на обучении нейросети, дает возможность устранить аномалию для получения более высоких итоговых значений метрик, показывающих качество работы нейросети.

Можно выделить несколько методов обнаружения аномалий:

1. STL декомпозиция, или же разложение ряда на основные компоненты;
2. Использование изолированного леса для определения аномалий;
3. Forecasting-based подход, или подход, основанный на определении анома-

лий с помощью предсказания значений;

4. Clustering-based подход, или подход, основанный на решении задачи кластеризации;
5. Использование автокодировщиков (глубокое обучение).

1 STL декомпозиция

STL декомпозиция, также известная как сезонно-трендовая декомпозиция, основанная на LOESS (locally estimated scatterplot smoothing, или же локально оцененное сглаживание диаграммы рассеяния) — метод, использующий для обнаружения аномалий факт того, что временной ряд можно разложить на его компоненты.

Метод эффективно работает для сезонных временных рядов (обладающих ярко выраженным свойством сезонности), которые являются наиболее распространенным типом временных рядов.

При разложении на компоненты с последующих их визуализацией, можно отследить наличие аномалии. Рассмотрим пример с декомпозицией временного ряда, определяющих данные о продажах сома за 1996–2000 гг. Декомпозицию можно осуществить с помощью программного кода на Python с использованием библиотеки **statmodels**.

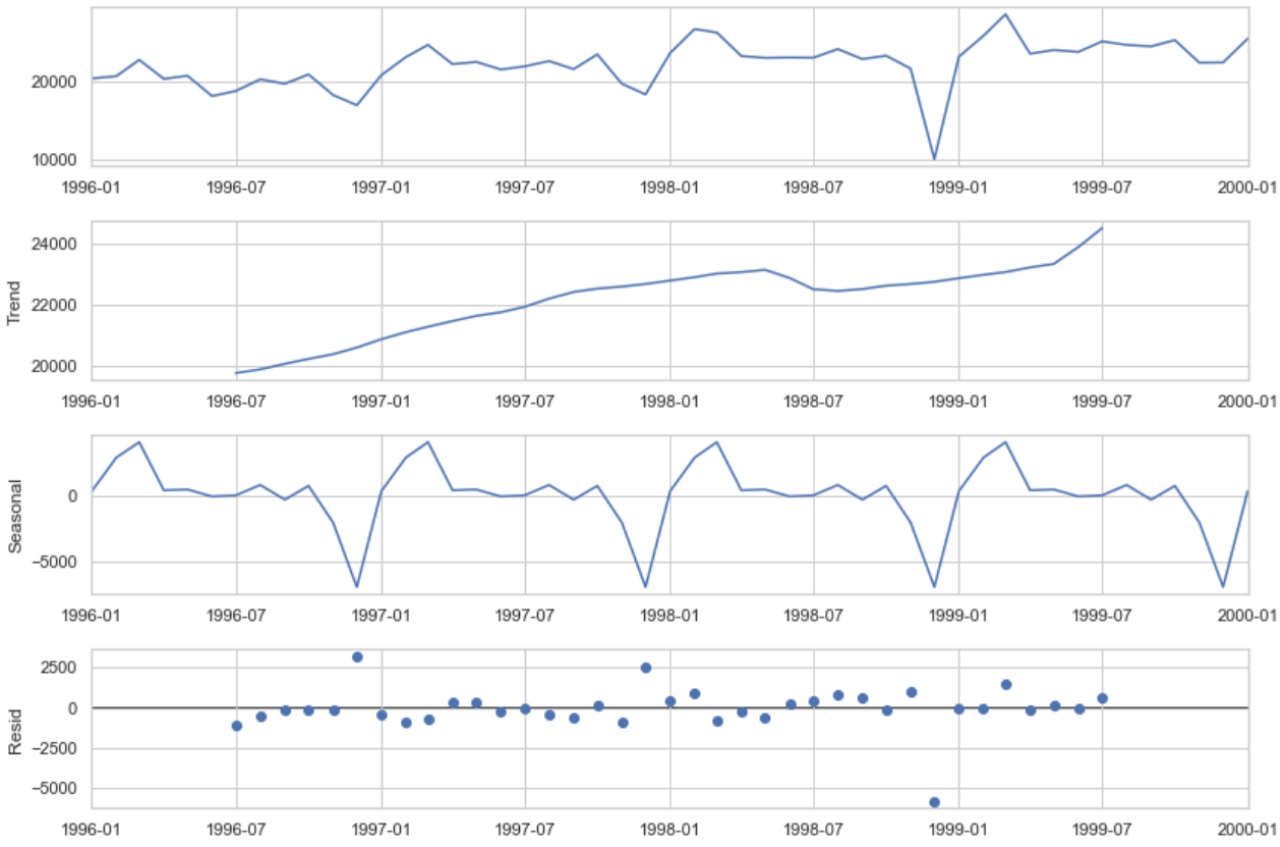

Рисунок 1 – Декомпозиция временного ряда (исходный ряд, тренд, сезонность, остаточная вариация)

На рисунке 1 на последней компоненте ряда можно наблюдать точку, сильно отличающуюся своим значением относительно других точек ряда. В

данном случае совершенно необязательно использовать нейронные сети для определения аномалии — достаточно проанализировать отклонение последней компоненты ряда (остаточной девиации) и ввести для него некоторый порог, и таким образом получить простой алгоритм обнаружения аномалий.

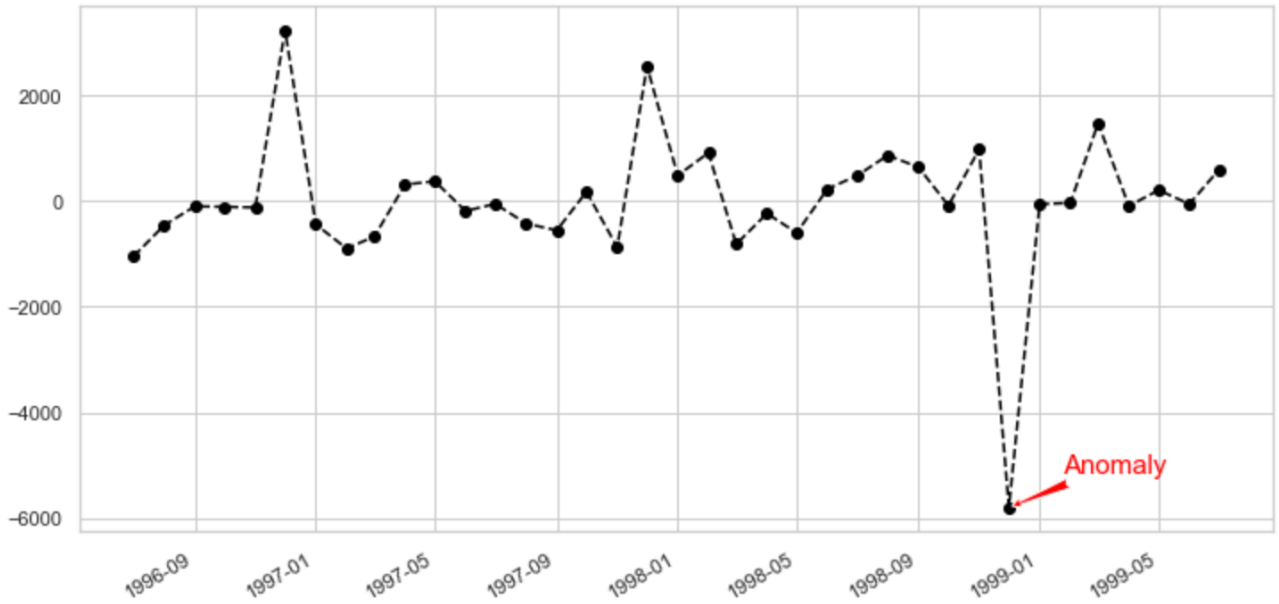

Рисунок 2 – Определение аномалии с помощью декомпозиции

Данный метод в контексте поиска аномалий с помощью нейросетей может быть полезен для предварительного преобразования временного ряда перед тем, чтобы применять различные нейронные сети. Например, можно использовать изображения отрезков временного ряда и использовать сверточные нейронные сети (или же их дообученные или модифицированные версии), которые будут работать с изображениями временного ряда.

2 Изолированный лес

Данный метод использует популярные алгоритмы машинного обучения — деревья решений и случайные леса, чтобы решать задачу выявления выбросов/аномалий во временных рядах.

Можно выделить два подхода:

1. Обучение с учителем: с помощью размеченных данных (где для конкретных блоков временных рядов сопоставляется значение, которое определяет наличие аномалии в этом блоке). Однако для этого нужно осуществить разметку данных, которая чаще всего является трудоёмким процессом и занимает много времени.
2. Обучение без учителя: можно использовать алгоритм изолированного леса (Isolation Forest) для предсказания наличия аномалии.

Основная идея второго способа, отличающаяся от других популярных методов обнаружения выбросов, заключается в том, что Isolation Forest явно идентифицирует аномалии вместо профилирования нормальных точек данных. То есть Isolation Forest обнаруживает аномалии исключительно на основе того факта, что аномалии — это точки данных, которых мало и они отличаются от большинства. Изоляция аномалий осуществляется без использования каких-либо мер расстояния или энтропии (например, Евклидово расстояние, DTW или взаимная информация). Изолированный лес, как и любой метод ансамбля деревьев, основан на деревьях решений.

Можно выделить три основных шага применения изолированного леса:

1. Для модели устанавливается значение того, какая доля выбросов присутствует в данных.
2. После обучения модели совершить предсказание, тем самым выполнить обнаружение выбросов в данных.
3. Визуализировать временной ряд с обнаруженными на нём аномалиями (например, для оценки качества определения выбросов).

Листинг для выполнения всех трёх шагов выглядит примерно следующим образом (при этом также используются библиотеки **matplotlib, pandas, sklearn**):

```
1               plt.rc('figure',figsize=(12,6))
2               plt.rc('font',size=15)
3               # визуализация исходного графика
```

```
4               catfish_sales.plot()
5
6               # определение доли выбросов
7               outliers_fraction = float(.01)
8               scaler = StandardScaler()
9
10              # преобразование выборки для применения изолированного леса
11              np_scaled = scaler.fit_transform(catfish_sales.values.reshape(-1,
                ↪  1))
12              data = pd.DataFrame(np_scaled)
13
14              # обучение изолированного леса и предсказание с помощью него
15              model =  IsolationForest(contamination=outliers_fraction)
16              model.fit(data)
17              catfish_sales['anomaly'] = model.predict(data)
18
19              # визуализация графика с обнаруженными аномалиями
20              fig, ax = plt.subplots(figsize=(10,6))
21              a = catfish_sales.loc[catfish_sales['anomaly'] == -1, ['Total']]
                ↪  #anomaly
22              ax.plot(catfish_sales.index, catfish_sales['Total'],
                ↪  color='black', label = 'Normal')
23              ax.scatter(a.index,a['Total'], color='red', label = 'Anomaly')
24              plt.legend()
25              plt.show()
```

С помощью программы выше можно получить результат в виде двух изображений на рисунке 3.

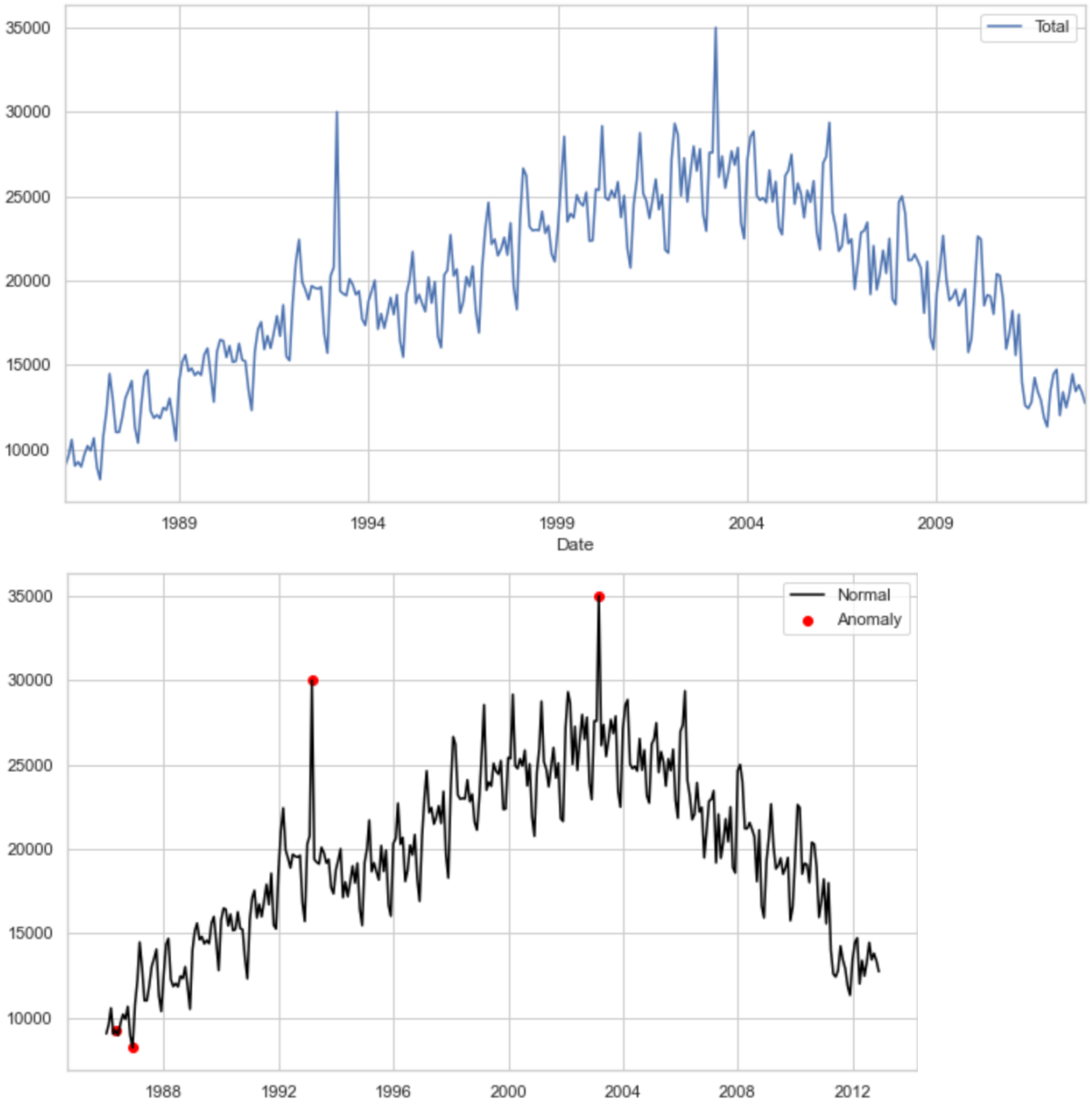

Рисунок 3 – Обнаружение аномалий с помощью изолированного леса

Хоть деревья решений это не нейронные сети, а методы машинного обучения, нельзя не сказать об их возможной эффективности решения данной задачи без применения методов глубокого обучения. Кроме того, не исключена возможность применения данных алгоритмов в качестве элементов ансамбля, в котором следующим элементом после изолированного леса будет применяться нейронная сеть.

3 Forecasting-based подход

Обнаружение аномалий с помощью прогнозирования основано на подходе, при котором на основе нескольких точек из прошлого генерируется значение следующей точки с добавлением некоторой случайной величины, которая обычно представляет собой белый шум. Таким образом можно осуществить предсказание нескольких значений сигнала подряд, что будет влиять на горизонт дальнейшего прогноза — спрогнозированный далее сигнал становится более плавным, гладким. Следовательно, те значения временного ряда, которые будут значительно отличаться от спрогнозированного, являются выбросами.

Сложность использования этого метода заключается в том, чтобы выбрать то, по каким признакам и параметрам будет осуществляться проверка на различия двух значений временного ряда — будь то значение их разности, или же что-то более универсальное и общее, например:

1. общее расстояние между блоками временного ряда (например, Евклидово), которое определяется следующей формулой:

$$Euc(x, y) = \left(\sum |(x_i - y_i)|^2 \right)^{\frac{1}{2}},$$

   где x_i — i-е значение первого временного ряда x, y_i — i-е значение второго временного ряда y;

2. значение взаимной информации:

$$MI(x, y) = \sum_{i=1}^{|x|} \sum_{j=1}^{|y|} \frac{|x_i \cap y_j|}{N} \log \frac{N|x_i \cap y_j|}{|x_i||y_j|},$$

   где x_i — i-е значение первого временного ряда x, y_i — i-е значение второго временного ряда y, $N = |x| + |y|$;

3. значение динамической трансформации временной шкалы (DTW, или же dynamic time warping):

$$DTW(x, y) = \min_{\pi} \sqrt{\sum_{(i,j) \in \pi} d(x_i, y_j)^2},$$

   где x_i — i-е значение первого временного ряда x, y_i — i-е значение второго временного ряда y, $\pi = [\pi_0, \ldots, \pi_K]$ — это путь, который удовлетворяет следующим свойствам:

— это список пар индексов $\pi_k = (i_k, j_k)$ с $0 \leq i_k < n$ и $0 \leq j_k < m$
— $\pi_0 = (0, 0)$ и $\pi_K = (n-1, m-1)$
— $\forall k > 0, \pi_k = (i_k, j_k)$ относится к $\pi_{k-1} = (i_{k-1}, j_{k-1})$ следующим образом: $i_{k-1} \leq i_k \leq i_{k-1} + 1$ и $j_{k-1} \leq j_k \leq j_{k-1} + 1$.

Здесь путь можно рассматривать как временное выравнивание временных рядов, при котором евклидово расстояние между выровненными временными рядами минимально.

Еще одним препятствием является то, что после дифференциации сигнала он должен оставаться стационарным. Это означает, что сигнал не должен зависеть от времени, что является существенным ограничением.

Для применения данного метода необходимо использовать некоторую модель для прогнозирования значения временного ряда, будь то модель нейронной сети или же некоторая статистическая модель:

1. ARMA (модель авторегрессии - скользящего среднего). Модель ARMA обобщает две более простые модели временных рядов — модель авторегрессии (AR) и модель скользящего среднего (MA);
2. ARIMA (модель Бокса — Дженкинса) — является расширением моделей ARMA для нестационарных временных рядов;
3. SARIMA (модель сезонного авторегрессивного интегрированного скользящего среднего) — расширение модели ARIMA, основанной на концепции сезонных трендов. SARIMA вместе со своими предшественниками представляет группу наиболее распространенных математических моделей, использующихся для анализа и прогнозирования стационарных временных рядов в статистике;
4. RNN (рекуррентные нейронные сети) — вид нейронной сети, который хорошо подходит для задач обработки последовательностей данных (которыми в том числе могут являться временные ряды);
5. LSTM (long short-term memory) — рекуррентная нейронная сеть, направленная на решение проблемы исчезающего градиента, характерной для рекуррентной нейронной сети.

Рассмотрим последние две более подробно.

3.1 RNN

Рекуррентная нейронная сеть или РНС (англ. Recurrent Neural Network, RNN) — это тип искусственной нейронной сети, которая обрабатывает после-

довательности данных и временные ряды. Подобно нейронным сетям с прямой связью (англ. Feedforward Neural Network, FNN) и CNN, рекуррентные нейронные сети используют обучающие данные для изменения своих весов. Основным отличием от других видов сетей является ”память”, суть которой в том, что в процессе обработки входной информации особым текущим слоем в RNN, используются входные параметры к некоторым предыдущим слоям сети (таким образом влияя на результат работы текущего слоя сети). В то время как традиционные глубокие нейронные сети предполагают, что входные и выходные данные слоев независимы друг от друга, выходы рекуррентных нейронных сетей зависят от предшествующих элементов внутри последовательности этих слоев. Хотя будущие преобразования также могут быть полезны для определения результата данной последовательности, однонаправленные рекуррентные нейронные сети не могут учитывать эти преобразования в своих прогнозах.

Однако при использовании первых архитектур RNN возникала проблема потери способности связывать информацию в силу уменьшения влияния аргументов слоев сети на текущий обрабатываемый слой по мере увеличения ”расстояния” между слоем, для которого были изначально предназначены аргументы, и текущим слоем. Уменьшение влияния выражается через проблему исчезающего градиента (англ. Vanishing gradient problem), которая возникает в процессе обучения ANN с помощью методов, основанных на градиентном спуске (англ. Gradient Descent) и методе обратного распространения ошибки (англ. Backpropagation). В этих способах обучения, на протяжении всей итерации обучения или эпохи, каждый из весов нейросети обновляется пропорционально частной производной функции ошибки от текущего веса. Время от времени значение градиента может становиться бесконечно малым, что препятствует обновлению значения веса. На практике, в силу отсутствия возможности сохранения качества передачи параметров между слоями, была представлена реализация модификации рекуррентной нейросети, которая способна к обучению долговременным зависимостям. Название такого подкласса RNN — сеть с долгой краткосрочной памятью (англ. Long Short-Term Memory, LSTM).

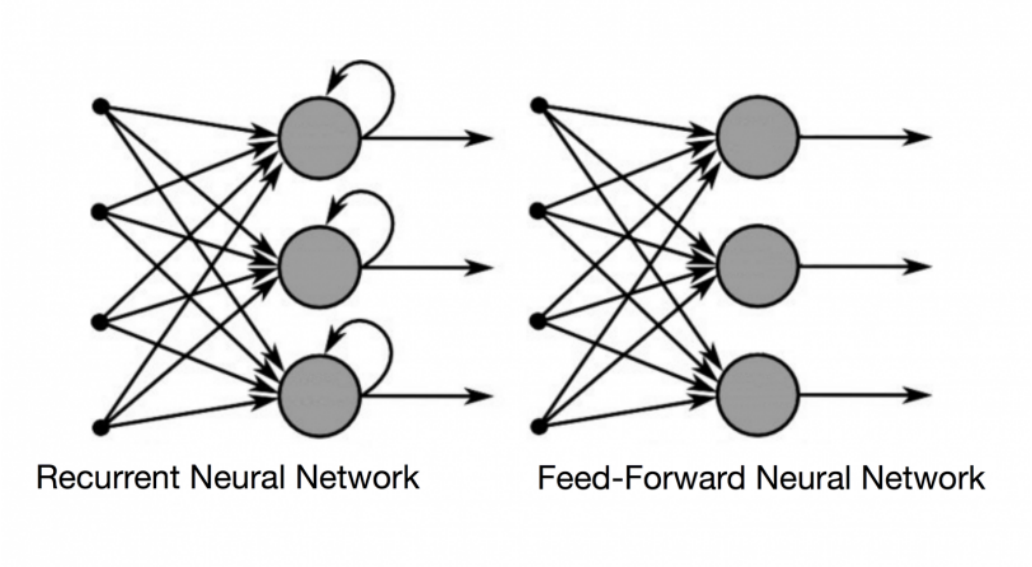

Рисунок 4 – Абстрактное сравнение архитектур FNN и RNN

3.2 LSTM

Решение с помощью LSTM использует ”карусель с постоянными ошибками” (англ. Constant Error Carousel, CEC), которые обеспечивают постоянный поток ошибок (необходимый для хранения значений ошибки для дальнейшего обучения модели) в специальных ячейках. Доступ к ячейкам (и их применение) осуществляется мультипликативными блоками ворот (англ. gate units), которые учатся своевременно предоставлять этот доступ. CEC являются центральной функцией LSTM, где осуществляется хранение краткосрочной памяти в течении длительных периодов времени. В ходе выполнения обработки соединений между другими блоками сети LSTM может также возникнуть конфликт обновления веса. Входные соединения некоторого нейрона u могут конфликтовать в процессе обновления веса по причине того, что один и тот же вес может как использоваться для хранения некоторого входного значения, так и не использоваться. Для взвешенных выходов соединений, идущих от нейрона u, одинаковые веса могут вернуть содержимое u и сделать поток вывода ошибки в другие нейроны сети некорректным. Эту проблему решает расширение CEC входными и выходными блоками ворот, которые соединены с входным слоем сети и с другими ячейками памяти, что ведет к формированию особого блока LSTM, называемого блоком памяти (англ. Memory Block).

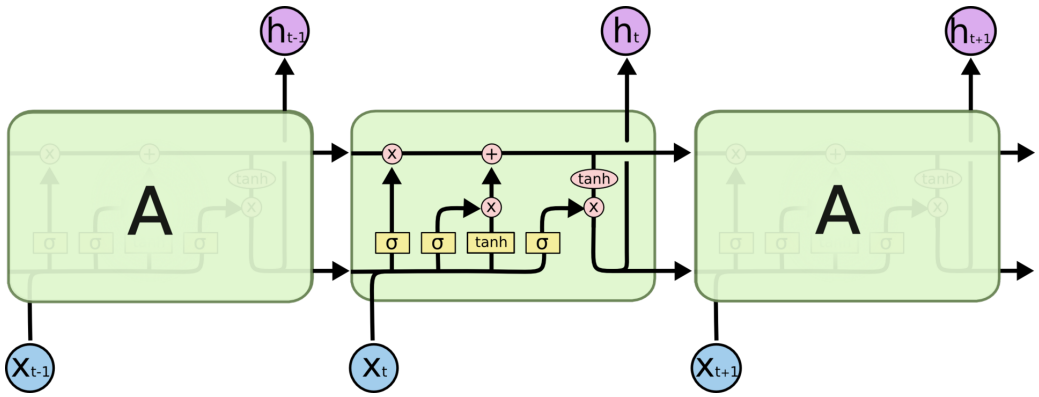

Рисунок 5 – Схема содержимого модуля LSTM-сети

Таким образом, для каждого элемента входной последовательности каждый t-ый слой LSTM модели осуществляет следующие вычисления:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi});$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf});$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg});$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho});$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t;$$

$$h_t = o_t \odot \tanh(c_t),$$

где h_t, c_t, x_t — скрытое состояние слоя модели, состояние клетки и входной параметр в момент времени t соответственно; h_{t-1} определяет скрытое состояние слоя в момент времени $t-1$ или начальное скрытое состояние в момент времени o. Элементы i_t, f_t, g_t, o_t являются входными, забывающими, клеточными и выходными воротами соответственно. Символ σ определяет функцию сигмоиды, $\odot$ — поэлементное произведение, также называемое произведением Адамара.

4 Clustering-based подход

Если вернуться к теме применения методов определения аномалий, предполагающих обучение без учителя, то следует упомянуть подход, связанный с кластеризацией.

Подход подразумевает рассмотрение элементов выборки как точек в пространстве, которые можно определить в группы ближайших друг к другу точек — кластеров. Элементы выборки, выходящие за пределы определенных кластеров, потенциально могут быть помечены как аномалии.

Существует несколько методов кластеризации, среди которых можно выделить:

1. Метод k-средних (англ. k-means) — итеративный алгоритм, основанный на минимизации суммарного квадратичного отклонения точек кластеров от центров этих кластеров;
2. Распространение похожести (англ. affinity propagation) — распространяет информацию о похожести между парами объектов для выбора типичных представителей каждого кластера;
3. Сдвиг среднего значения (англ. mean shift) — выбирает центроиды кластеров в областях с наибольшей плотностью.

Далее будет рассмотрен пример применения k-средних на выборке многомерного временного ряда. Однако, чтобы применить этот метод, необходимо указать количество кластеров, на которые следует делить элементы выборки. Для этого можно использовать метод локтя (англ. Elbow method). Метод локтя представляет собой график зависимости количества кластеров от оценки дисперсии. Чтобы реализовать это, можно использовать реализацию k-средних в **sklearn**:

```
1  data = df[['price_usd', 'srch_booking_window',
   ↪  'srch_saturday_night_bool']]
2  n_cluster = range(1, 20)
3  kmeans = [KMeans(n_clusters=i).fit(data) for i in n_cluster]
4  scores = [kmeans[i].score(data) for i in range(len(kmeans))]
5  fig, ax = plt.subplots(figsize=(10,6))
6  ax.plot(n_cluster, scores)
7  plt.xlabel('Number of Clusters')
8  plt.ylabel('Score')
9  plt.title('Elbow Curve')
10 plt.show()
```

Таким образом можно получить следующее изображение:

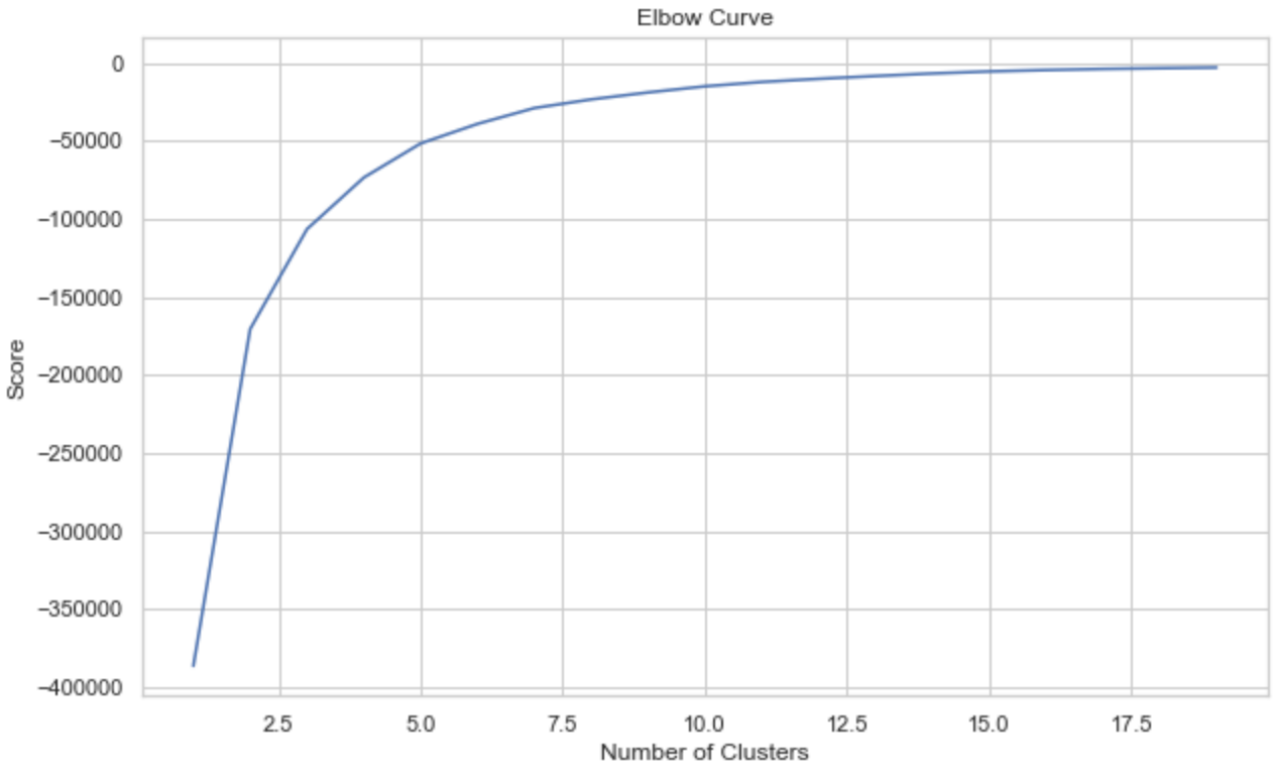

Рисунок 6 – График для метода локтя

Из приведенного выше рисунка видно, что график выравнивается после 10 кластеров, а это означает, что добавление большего количества кластеров не способствует более эффективному разделению элементов выборки.

Таким образом, определение аномалий можно осуществить сначала относительно кластеров, а затем применить полученные результаты относительно исходного временного ряда.

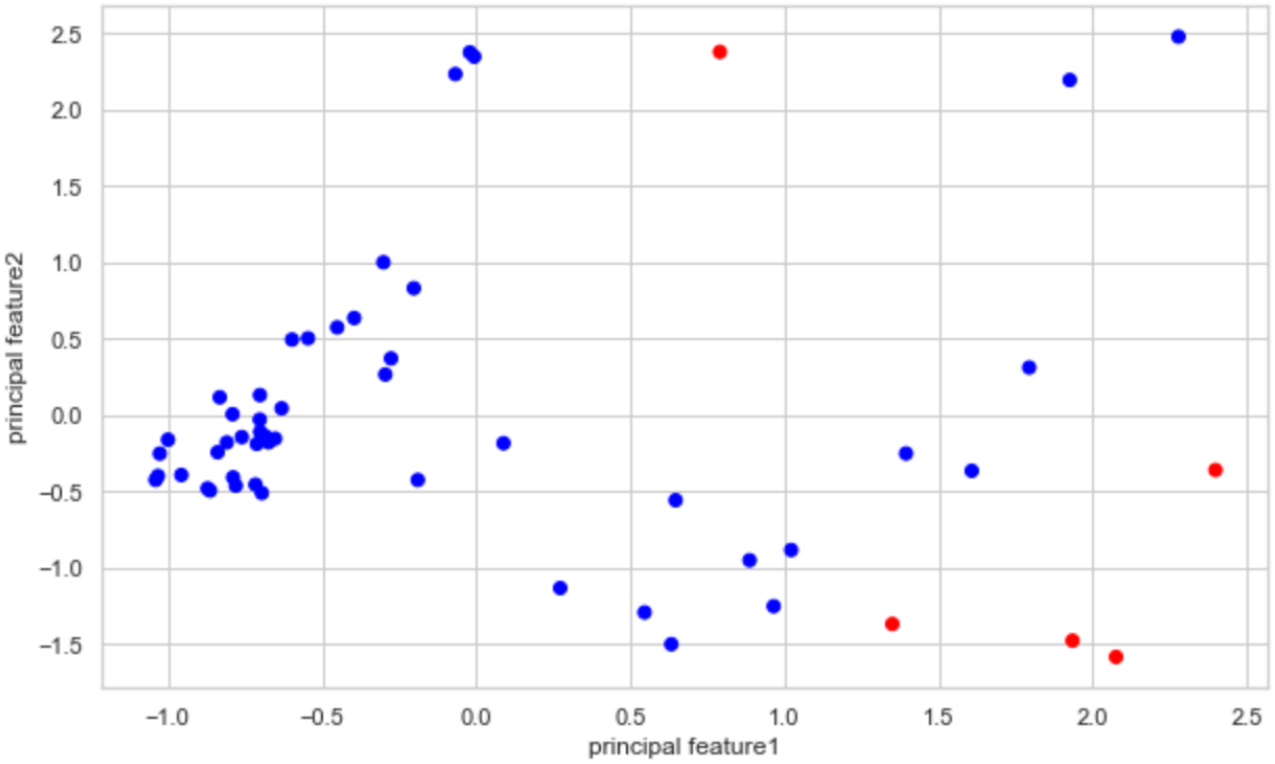

Рисунок 7 – Определение аномалий относительно кластера

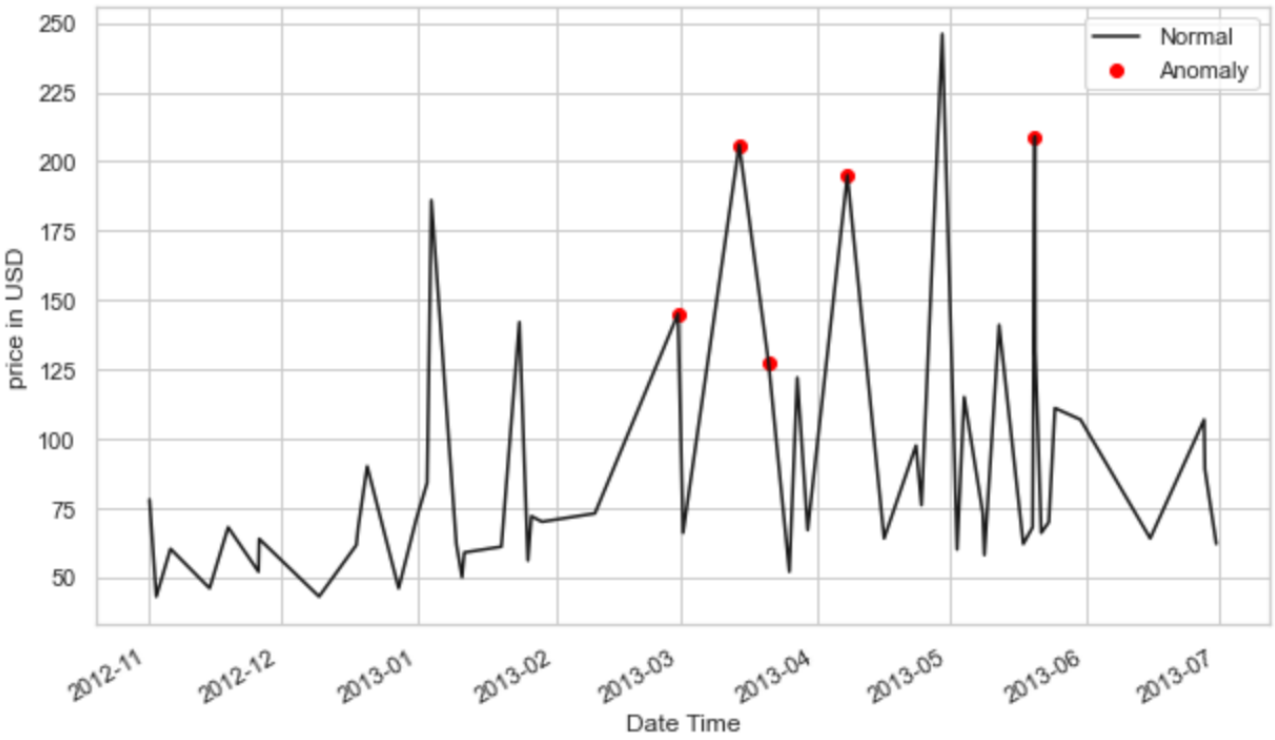

Рисунок 8 – Определение тех же аномалий на исходном временном ряду

5 Автокодировщики

Автокодировщик (англ. autoencoder) — специальная архитектура искусственных нейронных сетей, позволяющая применять обучение без учителя при использовании метода с обратного распространения ошибки. Простейшая архитектура автокодировщика — сеть прямого распространения, без обратных связей, наиболее схожая с перцептроном и содержащая входной слой, промежуточный слой и выходной слой. В отличие от перцептрона, выходной слой автокодировщика должен содержать столько же нейронов, сколько и входной слой.

Автокодировщик включает в себя кодировщик (который преобразует входной сигнал в код) и декодировщик (который восстанавливает сигнал по его коду). Скрытый слой автокодировщика называется bottleneck-слоем, ключевая особенность которого — наименьшая размерность из всех скрытых слоев. Размерность слоев автокодировщика при переборе от входного слоя к bottleneck-слою последовательно уменьшается, а при переборе от bottleneck-слоя к выходному слою последовательно увеличивается. Набор слоев автокодировщика от входного до bottleneck-слоя называется кодировщиком, а от bottleneck к выходному — декодировщиком.

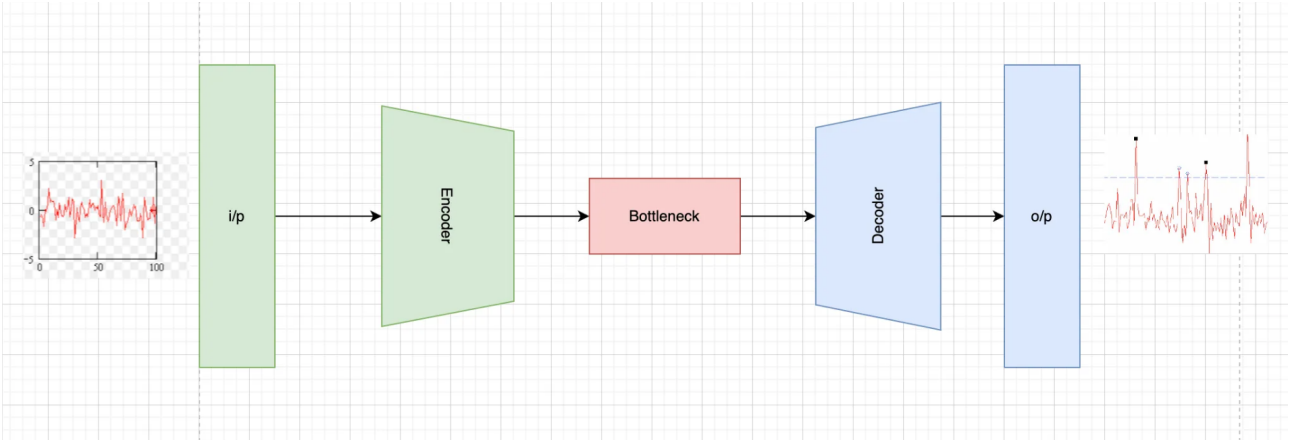

Рисунок 9 – Абстрактная схема автокодировщика

В данном случае применение скрытого представления данных автокодировщика аналогично уменьшению размерности.

При уменьшении размерности таким образом выявляются как основные закономерности в данных, так и выбросы/аномалии. Многие методы, основанные на расстоянии (например, kNN — метод k ближайших соседей), страдают от проклятия размерности при вычислении расстояния между всеми точками дан-

ных в полном пространстве признаков. Поэтому большую размерность может быть полезно уменьшать.

Основное преимущество по сравнению с методом главных компонент (PCA) в том, что автокодировщик (в отличии от PCA) может осуществлять нелинейные преобразования с помощью нелинейной функции активации и нескольких слоев. Более эффективно обучать несколько слоев с помощью автокодировщика, чем обучать одно огромное преобразование с помощью PCA. Таким образом, автокодировщик особенно эффективен при сложных и нелинейных данных, которыми и могут являться временные ряды с аномалиями.

ЗАКЛЮЧЕНИЕ

В данном реферате были рассмотрены различные методы поиска аномалий во временных рядах как с помощью нейросетей, так и с помощью алгоритмов статистики и машинного обучения. На основе всей вышеизложенной информации, можно выделить сильные и слабые стороны каждого из подходов:

1. STL декомпозиция

   Плюсы:

   — проста, может обрабатывать множество различных ситуаций, и все обнаруженные аномалии можно интерпретировать интуитивно;

   — может быть полезна для предварительного преобразования временного ряда перед тем, чтобы применять различные другие подходы (в том числе включающие применение нейронных сетей).

   Минус: ограниченные возможности настройки — пороговое значение определения аномалии и, возможно, значения доверительного интервала.

2. Изолированный лес

   Плюс: можно добавлять сколь угодно случайных переменных или признаков для создания более сложных моделей.

   Минус: растущее число признаков может быстро начать влиять на вычислительную производительность.

3. Forecasting-based подход

   Плюсы:

   — прекрасно обрабатывает различные виды сезонности (например, ежемесячные или ежегодные значения), и с помощью различных метрик временных рядов можно просто оценить промежуточные результаты работы этого подхода;

   — лучше справляется с пограничными случаями по сравнению с алгоритмом изолированного леса.

   Минус: поскольку этот метод основан на прогнозировании, он будет неэффективным в случаях с ограниченным количеством данных. Качество прогнозирования при меньшей выборке будет ниже, следовательно будет ниже точность обнаружения аномалий.

4. Clustering-based подход

   Плюс: аналогичен другим алгоритмам, которым характерно обучение без учителя: можно добавлять сколь угодно случайных переменных или при-

знаков для создания более сложных моделей.

Минус: растущее число признаков может быстро начать влиять на вычислительную производительность. Кроме того, будет расти число гиперпараметров, которые нужно качественно подобрать, поэтому всегда существует вероятность большой разницы в качестве работы моделей.

5. Автокодировщики

Плюсы:

- могут легко обрабатывать многомерные данные;
- благодаря нелинейным преобразованиям могут находить сложные закономерности в многомерных наборах данных.

Минусы:

- теряют эффективность при небольшом объеме данных;
- вычислительная сложность резко возрастет, если глубина сети или объём данных сильно увеличатся.