ЗАДАЧА ОБНАРУЖЕНИЯ АНОМАЛИЙ В СРЕДЕ ML.NET

Бадасян Т.С. Email: Badasyan1167@scientifictext.ru

Бадасян Тигран Смбатович – магистрант,
Факультет прикладной математики и физики,
Национальный политехнический университет Армении, г. Ереван, Республика Армения

Аннотация: *поддержание безопасности цифровых систем с огромным объемом данных является одной из основных задач ИТ-специалистов в настоящее время. Обнаружение аномалий в системах является одним из решений для преодоления этой проблемы. Обнаружение аномалий означает обнаружение данных, которые не являются нормальными или отклоняются от нормального поведения в системе. Обнаружение аномалий имеет различные приложения в биоинформатике, обработке изображений, кибербезопасности, безопасности для баз данных и т.д.*
В данной статье рассмотрена задача обнаружения двух видов аномалий (всплески или пики (spikes) и точки изменения (change points)) в среде ML.NET на данных временных рядов с информацией о ценах товара.
Ключевые слова: *аномалия, временные ряды, среда ML.NET, всплески, точки изменения.*

THE ANOMALY DETECTION PROBLEM WITH ML.NET FRAMEWORK

Badasyan T.S.

Badasyan Tigran Smbatovich - Graduate Student,
FACULTY OF APPLIED MATHEMATICS AND PHYSICS,
NATIONAL POLYTECHNIC UNIVERSITY OF ARMENIA, YEREVAN, REPUBLIC OF ARMENIA

Abstract: *maintaining the security of digital systems with a huge amount of data is one of the main tasks of IT professionals at present. Detecting anomalies in systems is one of the solutions to overcome this problem. Anomaly detection means detecting data that is not normal or deviates from system normal behavior. Anomaly detection has various applications in bioinformatics, image processing, cybersecurity, database security, etc.*
This article discusses the problem of detecting two types of anomalies (spikes and change points) in the ML.NET framework based on time series data with information on product prices.
Keywords: *anomaly, time series, ML.NET framework, spikes, change points.*

УДК 004.43:517.91

Введение: Аномалия является важной концепцией анализа данных. Аномалия определяется как отклонение от нормальных данных. Это означает, что объект данных не похож на другие наблюдения в наборе данных. Очень важно обнаруживать эти объекты во время анализа данных, чтобы отличать их от других данных. Методы обнаружения аномалий широко используются в следующих целях:

- обнаружение мошенничества с кредитными картами (и мобильными телефонами),
- обнаружение подозрительных веб-сайтов,
- сопоставление ДНК всего генома,
- фильтрация ЭКГ-сигналов,
- обнаружение подозрительных транзакций и т.д.

Исследование методов обнаружения аномалий очень важно в таких областях, как медицинское и общественное здоровье, принятие решений, бизнес-аналитика, обнаружение промышленных повреждений и др.

Проблема обнаружения аномалий стала признанная быстро развивающаяся область анализа данных. В последнее время было предложено большое количество методов. Многие обзоры и исследования посвящены этой проблеме, см., например, [1] - [4]. Они касаются разных аспектов обнаружения аномалий. [1] рассматривает традиционные алгоритмы обнаружения аномалий, [2] рассматривает существующие исследования по проблеме обнаружения аномалии в дискретных последовательностях, [3] ориентирована на изучение ансамблей, в то время как [4] использует только самые последние и популярные методы обнаружения аномалий для данных с высокой размерностью и смешанными типами, для которых классические методы обнаружения не могут работать очень хорошо.

Общепринятого формального определения аномалий не существует. Точное определение зависит от конкретной проблемы и ее представления данных. На абстрактном уровне аномалия определяется как шаблон, который не соответствует ожидаемому нормальному поведению. Поэтому простой подход обнаружения аномалий состоит в том, чтобы определить область, представляющую нормальное поведение, и объявить любое наблюдение в данных, которые не принадлежат этому нормальному региону, как аномалия. Но несколько факторов делают этот подход очень сложным. Конкретная

формулировка проблемы определяется несколькими различными факторами, такими как характер входных данных, наличие меток, ограничения и требования, налагаемые областью приложения. Это оправдывает необходимость широкого спектра методов обнаружения аномалий. Исходя из области исследований, большинство методов обнаружения аномалий можно разделить на классификацию, метод ближайшего соседа, кластеризацию, статистические и информационно-теоретические методы, машинное обучение. Каждый из большого числа методов обнаружения аномалий имеет свои уникальные достоинства и недостатки. Важно знать, какой метод обнаружения аномалий лучше всего подходит для данной проблемы. Учитывая сложность проблемного пространства, не представляется возможным дать такое понимание каждой проблемы обнаружения аномалий.

В частности обнаружение аномалий временных рядов - это процесс обнаружения всплеска данных временных рядов; указывает на заданный входной временной ряд, где поведение не соответствует ожидаемому или «странному».

Существует два типа аномалий временных рядов, которые можно обнаружить. Всплески или пики (spikes) указывают на временное аномальное поведение в системе. Точки изменения (change point) указывают на начало постоянных изменений во времени в системе.

В данной статье рассмотрена задача обнаружения всплесков и точек изменения для временных рядов данных о продажах продукта с помощью среды ML.NET.

Среда ML.NET: ML.NET - это бесплатная библиотека машинного обучения для языков программирования C # и F # . Она также поддерживает модели Python при использовании вместе с NimbusML. Предварительный выпуск ML.NET [5] включал в себя преобразования для проектирования функций, таких как создание n-граммы, и обучающихся для обработки бинарной классификации, многоклассовой классификации и задач регрессии.

ML.NET предоставляет разработчикам .NET возможности анализа и прогнозирования машинного обучения на основе моделей. Каркас построен на основе .NET Core и .NET Standard, унаследовав возможность кроссплатформенности в Linux, Windows и macOS.

ML.NET изначально разрабатывался в Microsoft Research и превратился в значительную структуру за последнее десятилетие [6]; он используется во многих группах продуктов в Microsoft, таких как Windows, Bing, Azure и т. д.

Разработчики могут обучать модели машинного обучения или повторно использовать существующую модель сторонними разработчиками и запускать ее в любой среде в автономном режиме. Это означает, что разработчикам не нужно иметь опыт работы с Data Science, чтобы использовать эту среду.

ML.NET – это, прежде всего, среда, что означает, что его можно расширить, добавив популярные библиотеки ML, такие как TensorFlow, Accord.NET и CNTK.

Сегодня ML.NET - это кроссплатформенная среда машинного обучения с открытым исходным кодом для разработчиков .NET. ML.NET также включает в себя Model Builder (простой в использовании инструмент пользовательского интерфейса в Visual Studio) и CLI (интерфейс командной строки), которые упрощают создание пользовательских моделей машинного обучения (ML) с использованием автоматического машинного обучения (AutoML).

Используя ML.NET, разработчики могут использовать свои существующие инструменты и наборы навыков для разработки и внедрения пользовательских ML в свои приложения, создавая пользовательские модели машинного обучения для распространенных сценариев, таких как прогнозирование цен, прогнозирование продаж, сегментация клиентов, классификация изображений и многое другое!

В [7] описываются различные задачи машинного обучения, которые можно выбрать в ML.NET, а также некоторые распространенные варианты использования: бинарная классификация, мультиклассовая классификация, регрессия, кластеризация, обнаружение аномалий, ранжирование, рекомендация, прогнозирование.

Задача обнаружения аномалий создает модель с использованием анализа основных компонентов (Principle Component Analysis PCA). Обнаружение аномалий на основе PCA помогает построить модель в сценариях, в которых легко получить данные обучения из одного класса, например, действительные транзакции, но сложно получить достаточные выборки целевых аномалий.

Поскольку по определению аномалии являются редкими событиями, может быть трудно собрать репрезентативную выборку данных для использования при моделировании. Алгоритмы, включенные в эту категорию, были специально разработаны для решения основных задач построения и обучения моделей с использованием несбалансированных наборов данных.

Обнаружение всплесков и точек изменения в ML.NET на данных временных рядов о продажах продукта: В ML.NET алгоритмы IID Spike Detection или IID Change point Detection подходят для независимых и идентично распределенных наборов данных. Преобразования обнаружения аномалий временных рядов работают непосредственно с входными данными. Метод IEstimator.Fit() не нуждается в обучающих данных для создания преобразования.

Процесс построения и обучения модели одинаков для обнаружения пиков и точек изменения. Основное отличие заключается в конкретном алгоритме обнаружения.

Целью обнаружения пиков является выявление внезапных, но временных всплесков, которые значительно отличаются от большинства значений данных временного ряда. Важно своевременно обнаруживать эти подозрительные редкие предметы, события или наблюдения, чтобы свести их к минимуму.

Точки изменения - это постоянные изменения в распределении значений потока событий временного ряда, например изменения уровня и тренды. Эти постоянные изменения длятся намного дольше, чем всплески, и могут указывать на катастрофические события.

Рассмотрим диаграмму временного ряда в некоторой базе данных, на примере которой будем проводить обнаружение аномалий.

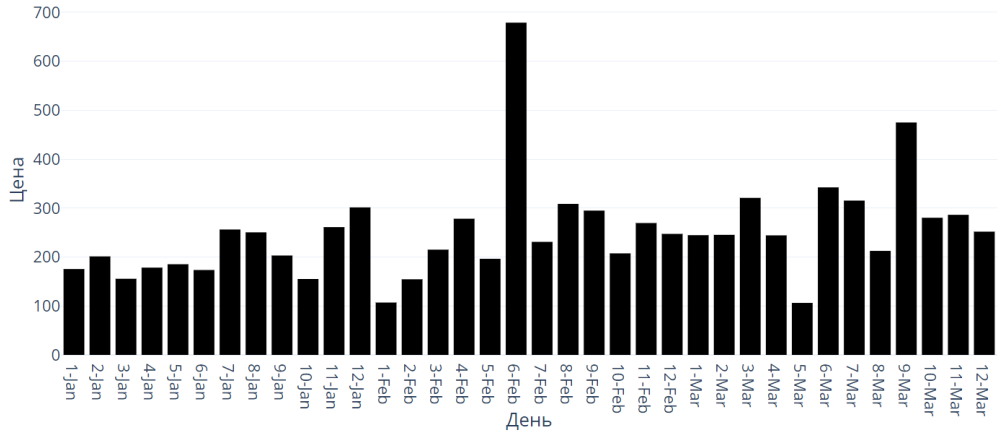

Рис. 1. Временные ряды в некоторой базе данных

Добавим CreateEmptyDataView() метод, который создает пустой объект представления данных с правильной схемой для использования в качестве входных данных для метода IEstimator.Fit().

```
private static IDataView CreateEmptyDataView(MLContext mlContext)
{
   IEnumerable<ProductSalesData> enumerableData = new List<ProductSalesData>();
   return mlContext.Data.LoadFromEnumerable(enumerableData);
}
```

Далее надо построить DetectSpike() метод, который создает преобразование из оценки, обнаруживает пики на основе исторических данных о продажах, отображает результаты.

```
private static void DetectSpike(
   MLContext mlContext,
   IDataView productSales,
   int docSize)
{
   // STEP 1: Set the training algorithm
   var iidSpikeEstimator = mlContext.Transforms.DetectIidSpike(
      outputColumnName: nameof(ProductSalesPrediction.Prediction),
      inputColumnName: nameof(ProductSalesData.Price),
      confidence: 95,
      pvalueHistoryLength: docSize / 4);

   // STEP 2: Create the transform
   ITransformer iidSpikeTransform = iidSpikeEstimator
      .Fit(CreateEmptyDataView(mlContext));

   // STEP 3: Apply data transformation to create predictions.
   IDataView transformedData = iidSpikeTransform.Transform(productSales);
   var predictions = mlContext.Data
      .CreateEnumerable<ProductSalesPrediction>(
      transformedData,
      reuseRowObject: false);
```

```
    Console.WriteLine("Alert\tScore\tP-Value");
    foreach (var p in predictions)
    {
        var res=$"{p.Prediction[0]}\t{p.Prediction[1]:f2}\t{p.Prediction[2]:F2}";

        if (p.Prediction[0] == 1)
        {
            res += " <-- Spike detected";
        }
        Console.WriteLine(res);
    }
}
```

Здесь

- outputColumnName - это имя столбца, полученного в результате изменения inputColumnName, который представляет собой вектор, содержащий 3 элемента: alert (ненулевое значение означает скачок), raw score, и p-value.
- inputColumnName - имя столбца, который нужно изменить.
- confidence - это достоверность обнаружения всплеска в диапазоне 0-100.
- pvalueHistoryLength - размер скользящего окна для вычисления значения p.

На втором этапе нужно построить модификацию, используя iidSpikeEstimator, а на третьем этапе применить это преобразование для составления прогнозов.

Далее вызываем метод DetectSpike() из метода Main(), как показано ниже:

```
static readonly string _dataPath = Path.Combine(
    Environment.CurrentDirectory,
    "Data", "productSalesData.csv");
const int _docSize = 36;
static void Main(string[] args)
{
    MLContext mlContext = new MLContext();
    IDataView dataView = mlContext.Data
        .LoadFromTextFile<ProductSalesData>(
        path: _dataPath,
        hasHeader: true, separatorChar: ',');

    DetectSpike(mlContext, dataView, _docSize);
}
```

Запустив программу получаются результаты обнаружения пиков.

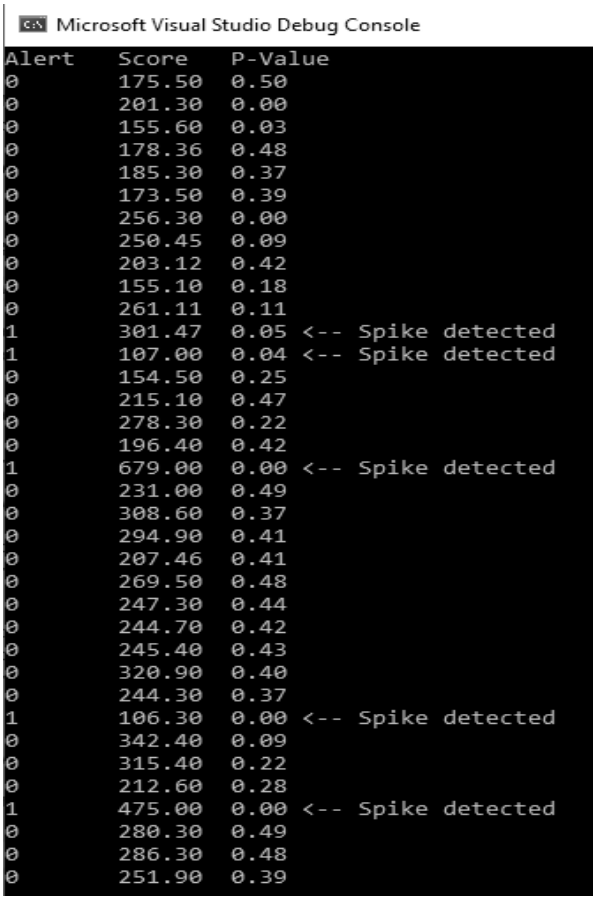

Рис. 2. Результаты обнаружения пиков на консоли

На диаграмме аномалии видны более отчетливо.

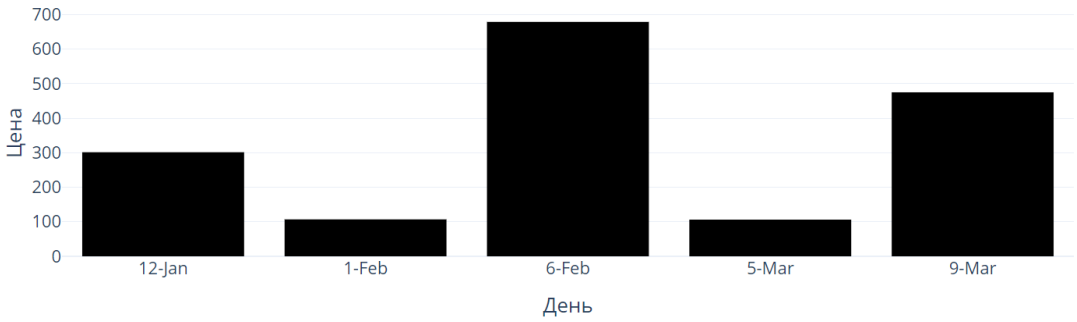

Рис. 3. Результаты обнаружения пиков на диаграмме

Для обнаружения точки изменения через ML.NET надо создать метод DetectChangePoint():

```
private static void DetectChangePoint(
    MLContext mlContext,
    IDataView productSales,
    int docSize)
{
    // STEP 1: Set the training algorithm
    var iidChangePointEstimator = mlContext.Transforms.DetectIidChangePoint(
        outputColumnName: nameof(ProductSalesPrediction.Prediction),
        inputColumnName: nameof(ProductSalesData.Price),
        confidence: 95,
        changeHistoryLength: docSize / 4);
```

```
        // STEP 2: Create the transform
        ITransformer iidChangePointTransform = iidChangePointEstimator
            .Fit(CreateEmptyDataView(mlContext));

        // STEP 3: Apply data transformation to create predictions.
        IDataView transformedData = iidChangePointTransform.Transform(productSales);
        var predictions = mlContext.Data
            .CreateEnumerable<ProductSalesPrediction>(
            transformedData,
            reuseRowObject: false);

        Console.WriteLine("Alert\tScore\tP-Value\tMartingale value");
        foreach (var p in predictions)
        {
            var res=$"{p.Prediction[0]}\t{p.Prediction[1]:f2}\t{p.Prediction[2]:F2}\t{p.Prediction[3]:F2}";

            if (p.Prediction[0] == 1)
            {
                res += " <-- alert is on, predicted changepoint";
            }
            Console.WriteLine(res);
        }
    }
```

Здесь

- Alert указывает на предупреждение об изменении точки для данных.
- Score - это значение ProductSales для данной точки в наборе данных.
- P-значение чем ближе к 0, тем более вероятно, что точка данных является аномалией.
- Martingale value используется для определения того, насколько «странной» является точка данных, на основе последовательности P-значений.

Запустив программу, получаются результаты обнаружения точек изменения.

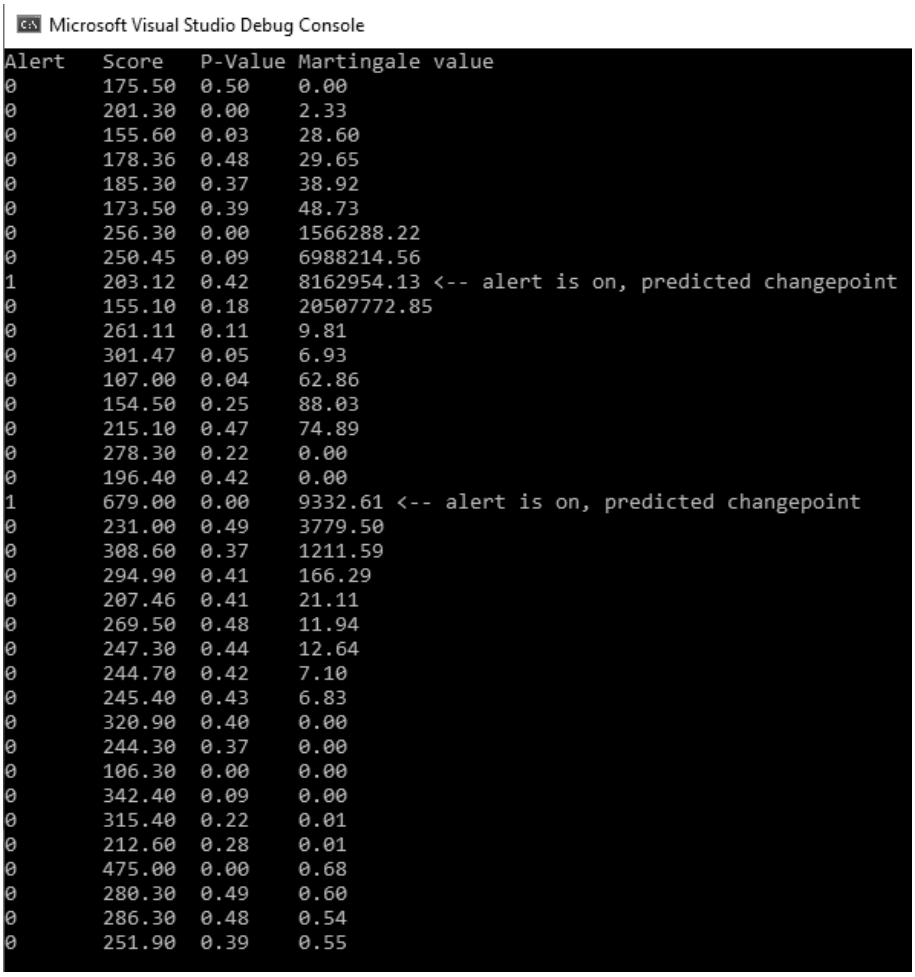

Рис. 4. Результаты обнаружения точек изменения на консоли

На диаграмме получается следующая картина.

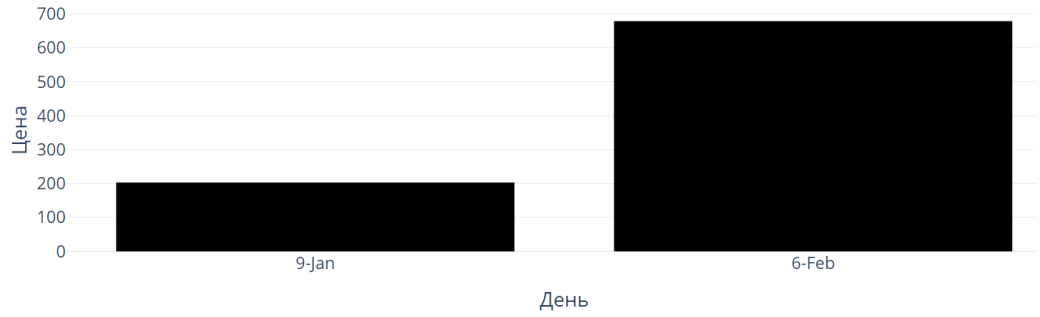

Рис. 5. Результаты обнаружения точек изменения на диаграмме

Заключение: Рассмотрена важная задача обнаружения аномалий для временных рядов. Изучена возможность обнаружения всплесков и точек изменения в среде ML.NET. Построены модели машинного обучения для обнаружения всплесков и аномалий точек изменения в данных о продажах.

Список литературы / References

1. *Chandola V., Banerjee A., Kumar V.* "Anomaly detection: a survey". ACM Computing Surveys. Vol. 41. № 3, Pp. 1–58, 2009.
2. *Chandola V., Banerjee A., Kumar V.* "Anomaly detection for discrete sequences: a survey", IEEE Trans. Knowl. Data Eng. Vol. 24. № 5. Pp. 823-839, 2012.
3. *Aggarwal C.C., Sathe S.* Outlier ensembles. An introduction, Springer, 2017.
4. *Xu X., Liu H., Yao M.* "Recent progress of anomaly detection". Hindawi. Complexity, 2019.

5. Cesar De la Torre, Introducing ML.NET: Cross-platform, Proven and Open Source Machine Learning Framework. May, 2018. [Электронный ресурс]. Режим доступа: https://devblogs.microsoft.com/dotnet/introducing-ml-net-cross-platform-proven-and-open-source-machine-learning-framework/ (дата обращения: 31.03.2020).
6. Cesar De la Torre, Announcing ML.NET 1.4 general availability (Machine Learning for .NET). November, 2019. [Электронный ресурс]. Режим доступа: https://devblogs.microsoft.com/dotnet/announcing-ml-net-1-4-global-availability-machine-learning-for-net/ (дата обращения: 31.03.2020).
7. Machine learning tasks in ML.NET. December, 2019. [Электронный ресурс]. Режим доступа:https://docs.microsoft.com/en-us/dotnet/machine-learning/resources/tasks#anomaly-detection/ (дата обращения: 31.03.2020).