ВИЗУАЛИЗАЦИЯ ДАННЫХ НА .NET F#

[1]***Бакунова Оксана Михайловна,***
[2]***Буркин Антон Владимирович,***
[2]***Протько Дмитрий Эдуардович,***
[2]***Петрович Арсений Сергеевич,***
[2]***Малофеевский Артем Дмитриевич***

Республика Беларусь, г. Минск, Институт информационных технологий Белорусского государственного университета информатики и радиоэлектроники;
[1]преподаватель кафедры Информационные системы и сети, магистр инженерии;
[2]student

ARTICLE INFO
Received 05 March 2018
Accepted 19 March 2018
Published 12 April 2018

KEYWORDS
Big data,
data,
visualization,
.net,
dotnet,
fsharp, f#

ABSTRACT
Data visualisation and sorting using dotnet libraries on F#. Language provides the type safety by using a strong code typing and a minimal amount of code due to functional paradigm. .Net Core framework and mono allow to run the code on any platform with framework support. .Net Libraries helps work on science projects without rich programming knowledges.

Введение. В наше время, человек часто сталкивается большими объемами данных. Они растут с каждой минутой, так как наша техника следит за нашими достижениями и записывает эти данные. Сервисы, которыми мы пользуемся, следят за нашими предпочтениями и тоже записывают эти данные. Изменения курсов валют, акций, популярность того или иного музыкального коллектива, альбома, песни, продажи какой-то техники, количество слов в речи политиков. Всё это несёт данные, на основе которых можно строить статистику, заниматься разработкой искусственного интеллекта.

Но как выбрать то, что действительно нам может быть полезно? Эти данные занимают много места, просто взять и прочитать их - займет сотни лет. Визуальная информация лучше воспринимается и позволяет быстро и эффективно донести до человека собственные мысли и идеи. Физиологически, восприятие визуальной информации является основной для потребителя. Есть многочисленные исследования, подтверждающие, что:

1. 90% информации человек воспринимает через зрение
2. 70% сенсорных рецепторов находятся в глазах
3. около половины нейронов головного мозга человека задействованы в обработке визуальной информации
4. на 19% меньше при работе с визуальными данными используется когнитивная функция мозга, отвечающая за обработку и анализ информации
5. на 17% выше производительность человека, работающего с визуальной информацией
6. на 4,5% лучше воспоминаются подробные детали визуальной информации

Пример: построив линейные графики продаж нескольких альбомов музыкальной группы на одной плоскости, мы сможем быстро выяснить, какие альбомы понравились потребителям больше. А на основе графика из значений (пользователь, версия браузера) мы сможем наглядно увидеть, какие версии актуальны для ваших потребителей и тратить деньги компании на поддержку именно их - что приведёт к росту популярности продукта. И таких исследований можно проводить много, повышая рентабельность вашего бизнеса.

Но возникает вопрос: чем и как сортировать данные, чем и как их отрисовывать. На данный момент это позволяет делать: JavaScript, Python, R, C#, F#. Все эти языки, кроме C# и

F# не являются строго типизированными, это даёт плюс в крайне низком пороге вхождения, но выливается в неудобство работы с данными.

F# это язык принадлежащий к множеству языков функциональной парадигмы, но имеющий и объектно ориентированные возможности. Это даёт всю функциональную мощь в сортировке данных и простоту ООП. Язык позволяет в пару лаконичных строчек способность фильтровать, дополнять и проецировать данные. Изначальные объектные модели для данных строятся на ходу, за счёт сложных алгоритмов Type Providers.

В среде .Net имеется научный пакет со всеми нужными библиотеками - fslab. (Он прежде всего ориентирован на F#, но никто не запрещает пробовать использовать их из других языков .Net семейства.)

Он включает в себя нужные для визуализации пакеты:

– F# Data - реализует все необходимое для доступа к данным в ваших приложениях и сценариях F #. Он содержит Type Provider'ы F # для работы со структурированными форматами файлов (CSV, HTML, JSON и XML) и для доступа к данным WorldBank. Он также включает помощники для разбора файлов CSV, HTML и JSON и для отправки HTTP-запросов. Эта библиотека фокусируется на предоставлении простого, в основном доступного только для чтения доступа к структурированным документам и другим источникам данных. Он не нацелен на то, чтобы быть всеобъемлющей коллекцией Type Provider'ов F # (которые могут использоваться для многих других целей). Он также создан для хорошей работы с другими библиотеками, такими как Deedle, R Type Provider, F # Charting и FunScript.

– R Provider - это механизм, обеспечивающий плавную интероперабельность между F # и R. Type Provider обнаруживает R-пакеты, доступные в вашей операционной системе, и делает их доступными как пространства имен .NET под родительским пространством имен RProvider. R Type Provider позволяет использовать все возможности R из интерактивной среды F #. Он позволяет осуществлять «на лету» анализировать и визуализировать данные с использованием пакетов R, с дополнительным преимуществом IntelliSense над R и проверкой типов во время компиляции, что проверяет существование функций R. Он позволяет использовать все библиотеки .NET, а также уникальные возможности F # для доступа и обработки данных из самых разных источников через поставщиков типов.

– XPlot - это пакет визуализации данных для языка программирования F #, основанного на популярных библиотеках графических библиотек JavaScript. Он использует мощные и бесплатные библиотеки визуализации Google и Plotly, основанные на технологии HTML5 / SVG. Вы можете обращаться к HTML для диаграмм программно и использовать библиотеку из F # Interactive, открывая автоматически график после выполнения кода.

– FSharp.Plotly - работает на базе популярной библиотеки графиков JavaScript Plotly. Библиотека предоставляет полную обёртку для параметров конфигурации библиотеки, но дает вам возможность использовать удобный стиль F # Charting. Таким образом, вы получаете приятную поддержку интерфейса F # с полной мощностью Plotly.

– F# Charting - визуализацию данных через .NET Chart Controls в ОС Windows и Gtk в других ОС. Содержит в себе функции:

- Кросс-платформенный 2D-график и поддержка псевдо-3D-диаграмм на .NET.
- Много кросс-платформенных диаграмм: Area, Bar, Bubble, Column, Line, Point и другие.
- Создание диаграммы непосредственно из данных F #, таких как списки и кортежи.
- Создание диаграммы «LiveChart» с обновлением F # или Rx.
- Может использоваться совместно с библиотекой FSharp.Data.
- Многие дополнительные типы диаграмм (только для Windows): BoxPlot, Candlestick, Donut, ErrorBar, FastLine, FastPoint, Funnel, Kagi и другие.

F# поддерживает интерактивный режим выполнения исходного кода. Для визуализации данных, это даёт неоспоримый плюс, по сравнению с только компилируемыми языками. Приемущество в том, что можно во время выполнения программы изменять и дополнять код, менять алгоритмы сортировки и фильтрации данных, не перезапуская все остальные вычисления и при этом получая визуализацию основанную на новых данных. Для этого в интегрированной среде разработки Visual Studio (code) следует просто выделить код и отправить его интерпретатору.

Для работы с F# требует установить: Microsoft Visual Studio/ Microsoft Visual Studio Code/ JetBrains Rider и среду исполнения .Net/.Net Core/mono. Опционально R. Стоит отметить, что Type

Provider'ы в .Net Core находятся в тестовом состоянии на момент написании статьи и не рекомендуется к повседневному использованию. Microsoft Visual Studio Code после установки является только редактором кода, для корректной и комфортной работы с F# следует установить пакеты: Ionide-fsharp, Ionide-Paket, Ionide-FAKE . После чего, можно приступать к работе.

При визуализации данных, важно структурировать данные в максимально понятном для пользователя виде. В основном программист получает данные в "грязном" виде и с этим нужно бороться. Зачастую, требуется быстро отобразить данные без нагромождения общей кодовой базы. Применяя Type Provider, он: генерирет внутри себя объекты для взаимодействия с данными, тем самым освобождает программиста от одной из самых скучных и долгих задач – воссоздания объектной модели со всеми свойствами, а так же написания кода для считывания самих по себе данных. Далее вступает в ход функциональная парадигма языка, позволяющая структурировать данные с минимальными кодо-затратами.

Отдельно стоит отметить Microsoft Azure Notebook. Это сервис, который позволяет писать интерактивную документацию со вставками кода на F# и последующей визуализацией данных. Исполняемой средой является браузер, что позволяет динамически создавать презентации для работы с публикой, на основе опросов в реальном времени.

Следует обратить внимание на поддержку вызова функций из языка R через R Type Provider. Этот язык давно считается одним из самых популярных в анализе данных и для него существует множество пакетов для работы с данными и последующей их визуализации, так же пакеты для машинного обучения. Это проставляет F# недостающий функционал по причине молодости языка.

Примеры.

Пример #1 демонстрирует исходный код визуализации данных через библиотеку Xplot.GoogleCharts. Она визуализирует данные в браузере через JS библиотеку Google charts

Листинг на языке программирования F#

```
#load "..\packages/FsLab/FsLab.fsx"
open System
open FSharp.Data
open XPlot.GoogleCharts
let rand = Random()
let users = [0..9] |> List.map (fun i -> ((sprintf "user %o" i)), rand.Next(0,100)) |> List.toArray
users|>Chart.Line|>Chart.WithOptions(Options(legend=Legend(position="bottom")))|>Chart.WithLabels["Users"]
```

Визуализация данных в браузере

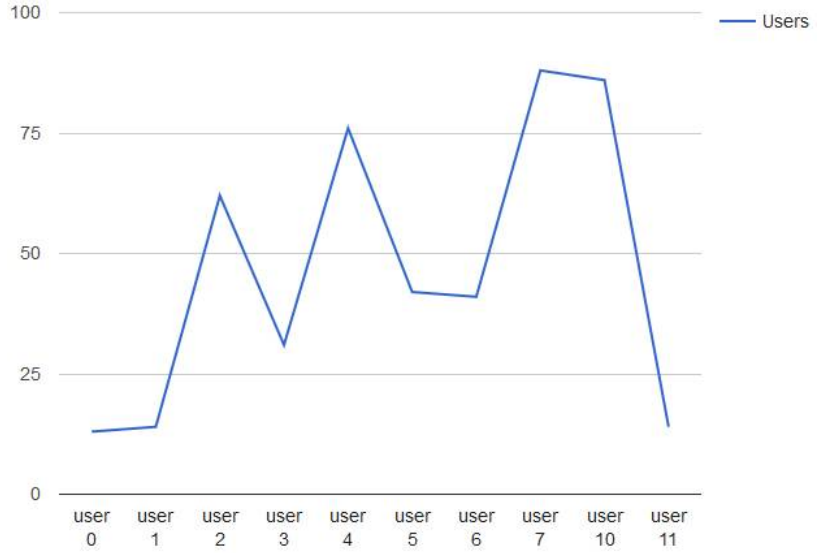

Рис. 1. Веб версия графика

Пример #2 демонстрирует визуализацию данных в приложении Windows через библиотеку F# Charting

Листинг на языке программирования F#

```
#load "..\packages/FsLab/FsLab.fsx"
open System
open FSharp.Data
open FSharp.Charting
let rand = Random()
let users = [0..9] |> List.map (fun i -> ((sprintf "user %o" i)), rand.Next(0,100)) |> List.toArray
users |> Chart.Line
```

Визуализация данных в окне ОС Windows

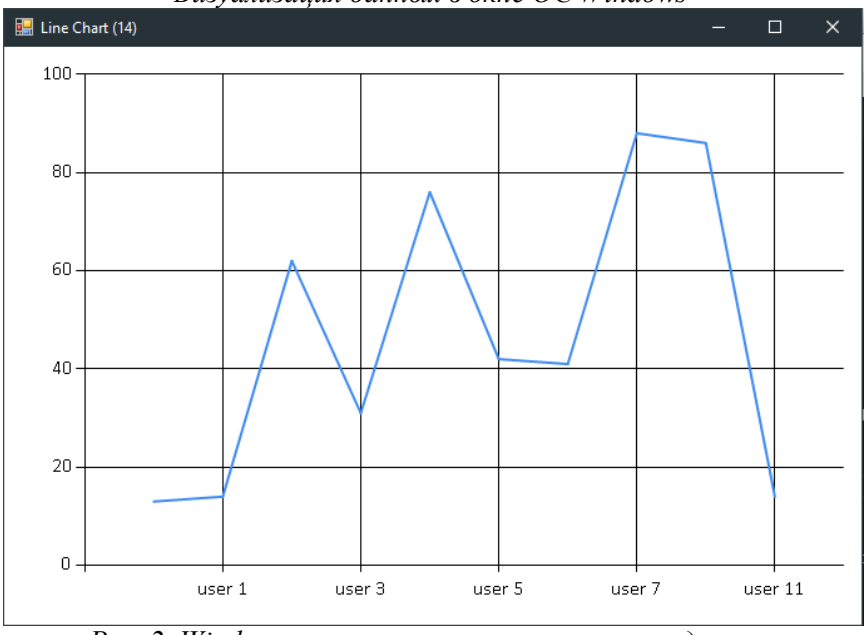

Рис. 2. Windows приложение визуализирующие данные

Выводы. F# является достаточно удобным и гибким языком, для манипуляции с большими данными и их визуализацией абсолютно на любой платформе на текущий момент времени. Платформа .net включает в себя большое количество библиотек которые помогут в любых начинаниях. Из минусов F# можно отметить высокий порог вхождения из за особенности функциональной парадигмы, она же и является его сильной стороной для подготовленного программиста.

ЛИТЕРАТУРА

1. Fsharp for Fun and Profit Authors: Scott Wlaschin
2. Expert F# 4.0 Authors: Syme, Don, Granicz, Adam, Cisternino, Antonio