SEMINARARBEIT

im Studiengang MSE

Lehrveranstaltung BLD

Data Engineering & Data Science

Ausgeführt von:

se15m002: Dominik Figl, BSc

Ort, Datum

Wien, 04.06.2016

FH University of Applied Sciences TECHNIKUM WIEN

Inhaltsverzeichnis

Data Engineering

1 Assignment: Big Data in Ihrem Umfeld

1.1 Assignment

Schemalose Daten (unstrukturiert): Die von mir entwickelten Dienste schreiben Zugriffsstatistiken in eine MongoDB.

Schematische Daten (strukturiert): Ich arbeite oft mit Sensorwerten. Meine Aufgabe ist es die Messwerte in Form einer XML-Datei zur Verfügung zu stellen. Diese XML-Datei ist durch ein XML-Schema definiert.

1.2 Assignment

gestreamt-verarbeitete Daten: Sensordaten (kommen in einem kontinuierlichen Strom) werden über Corba empfangen und in der Datenbank gespeichert.

Batch-verarbeitete Daten: Backup/Replikations Jobs der Datenbank

2 Assignment: Big Data in Ihrem Umfeld

2.1 Erklären Sie ihre Entscheidung

Die Entscheidung fiel auf **Apache Flink**, weil mich einerseits die auf den Folien erklärt Streamverarbeitung interessierte und weil zum Einrichten von Apache Flink auch das pdf „Chapter 02 – SettingUp Flink“ zur Verfügung gestellt wurde.

2.2 Screenshot der installierten Umgebung

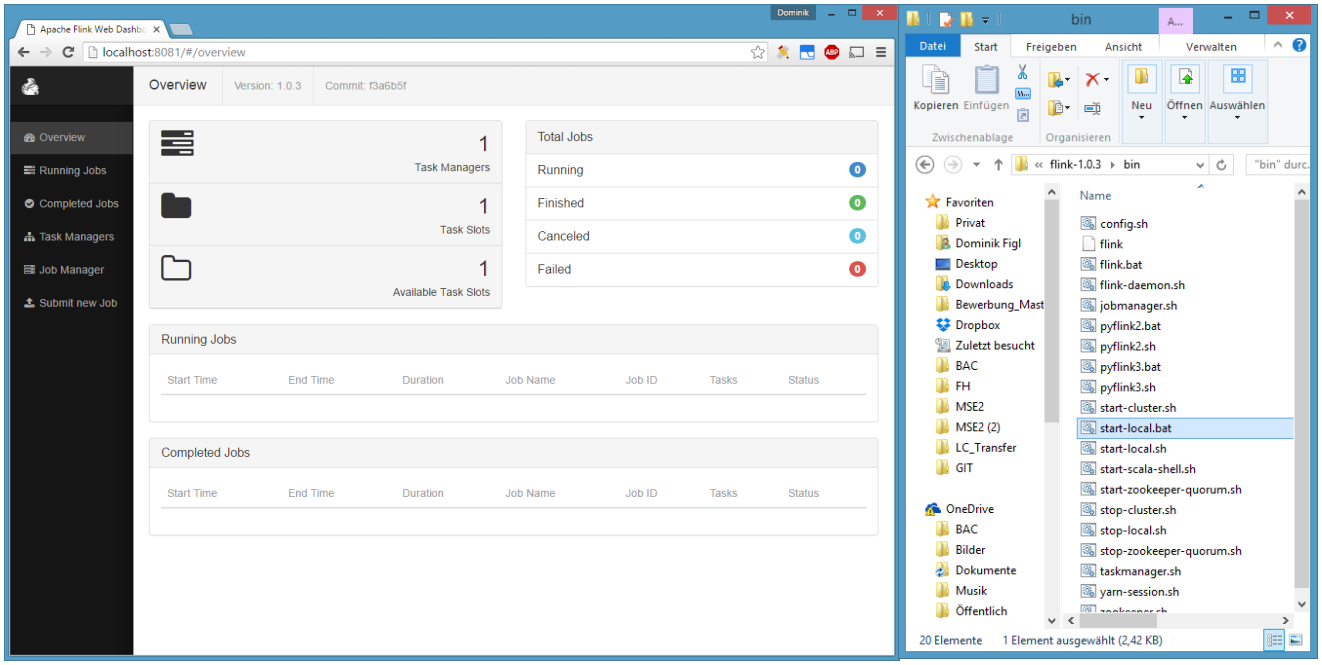

Abbildung 1: Screenshot - Apache Flink & start-local.bat

2.3 Beschreibung der Toolchain

Ich würde Apache Flink in Verbindung mit IntelliJ (Maven Projekt) verwenden. Für die Entwicklung ist eine lokale Apache Flink Installation ausreichend. In Produktion würde ich Apache Flink mit Hadoop kombinieren.

3 Assignment: Big Data in Ihrem Umfeld

Als „Hello World“ für Apache Flink wurde ein Programm entwickelt, mithilfe dessen der Word Count in einem String ermittelt werden kann. Das Ergebnis ist also wie oft welches Wort vorkommt. Das Projekt mit dem Namen „Apache Flink Java Hello World example“ befindet sich im git Repository. Es handelt sich um ein IntelliJ (Maven) Projekt.

Program:
Quelle: http://10minbasics.com/apache-flink-hello-world-java-example/

FlinkHelloWorld.java:

```
import org.apache.flink.api.common.functions.FlatMapFunction;
import org.apache.flink.api.java.DataSet;
import org.apache.flink.api.java.ExecutionEnvironment;
import org.apache.flink.api.java.aggregation.Aggregations;
import org.apache.flink.api.java.tuple.Tuple2;
import org.apache.flink.util.Collector;

public class FlinkHelloWorld {
    public static void main(String[] args) throws Exception {
        final ExecutionEnvironment env =
ExecutionEnvironment.getExecutionEnvironment();

        DataSet<String> sampleData = env.fromElements(
                "Hello! This is a text sample",
                "to represent Apache Flink streaming",
                "Hello Hello Hello"
        );

        DataSet<Tuple2<String, Integer>> wordCountResults =
                sampleData.flatMap(new LineSplitter())
                        .groupBy(0)
                        .aggregate(Aggregations.SUM, 1);

        wordCountResults.print();
    }
}

class LineSplitter implements FlatMapFunction<String, Tuple2<String,
Integer>> {
    @Override
    public void flatMap(String value, Collector<Tuple2<String, Integer>>
out) {
        String[] tokens = value.toLowerCase().split("\\W+");
        for (String token : tokens) {
            if (token.length() > 0) {
                out.collect(new Tuple2<String, Integer>(token, 1));
            }
        }
    }
}
```

pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>flink01</groupId>
    <artifactId>flink01</artifactId>
    <version>1.0</version>
    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <configuration>
                    <source>1.6</source>
                    <target>1.6</target>
                </configuration>
            </plugin>
        </plugins>
    </build>

    <dependencies>
        <dependency>
            <groupId>org.apache.flink</groupId>
            <artifactId>flink-java</artifactId>
            <version>1.0.3</version>
        </dependency>
        <dependency>
            <groupId>org.apache.flink</groupId>
            <artifactId>flink-streaming-java_2.10</artifactId>
            <version>1.0.3</version>
        </dependency>
        <dependency>
            <groupId>org.apache.flink</groupId>
            <artifactId>flink-clients_2.10</artifactId>
            <version>1.0.3</version>
        </dependency>
    </dependencies>

</project>
```

Ausgabe in der Java Console:

```
(sample,1)
(text,1)
(apache,1)
(flink,1)
(to,1)
(a,1)
(is,1)
(represent,1)
(streaming,1)
(this,1)
(hello,4)
```

Abbildung 2: Java Console Output Word Count

Um das Programm auszuführen einfach die Kommandozeile im Unterordner „Apache Flink Java Hello World example\target“ öffnen und folgenden Befehle ausführen:

java -jar BLD-flink-helloWorld.jar

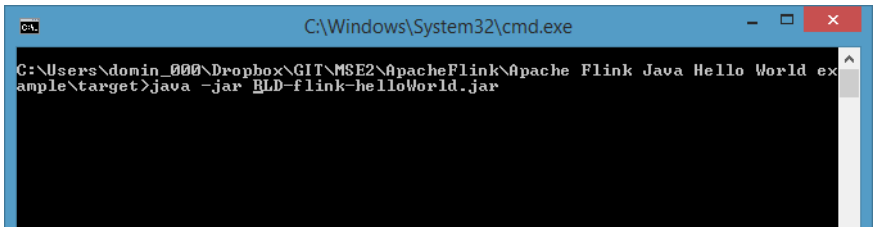

Abbildung 3: Flink HelloWorld jar in cmd starten

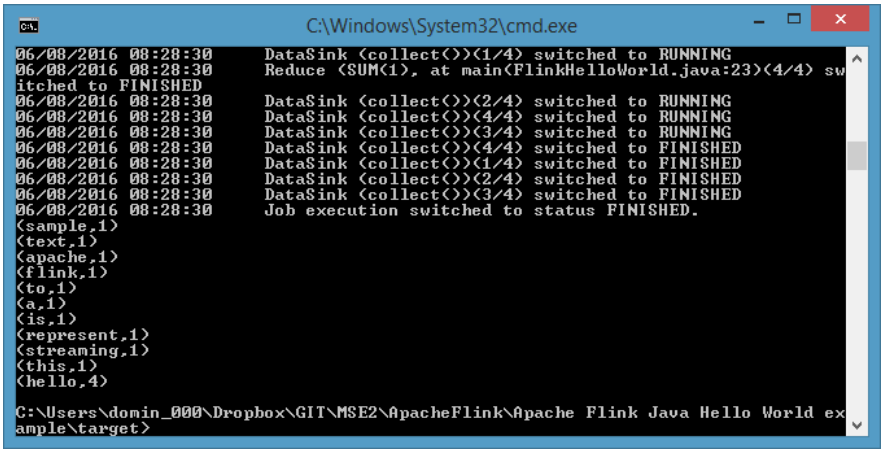

Abbildung 4: Erfolgreiches Ergebnis von Flink HelloWorld

Data Science

1 Assignment: Technologien

1.1 Weiter Technologien:

- Jupyter
- Apache Zeppelin
- Tableau
- RapidMiner

1.2 Auftrag Data Sciene Technologie

Auf den ersten Blick erscheint für mich „R“ am besten. Ich arbeite sehr viel mit Microsoft Produktion und deshalb ist die hochgradige Integration von „R“ in diverse Microsoft Produkte sehr vorteilhaft. Weiters habe ich mich bereits in der Vergangenheit kurz mit „R“ beschäftigt und einige Blog Posts zu „R“ gelesen.

2 Assignment: Technologien

Als Technologie wurde „R“ gewählt.

2.1 Begründung für „R“

Wie bereits im Unterabschnitt „1.2 Auftrag Data Sciene Technologie“ ausgeführt überzeugt mich „R“ einerseits durch die gute Integration in diverse Microsoft Produkte als auch dadurch, dass ich mich bereits etwas mit „R“ beschäftigt habe. Dennoch sollte die grundlegende Entscheidung für oder gegen eine Technologie auf den Anforderungen des Projekts beruhen.

2.2 Screenshot der installierten Umgebung

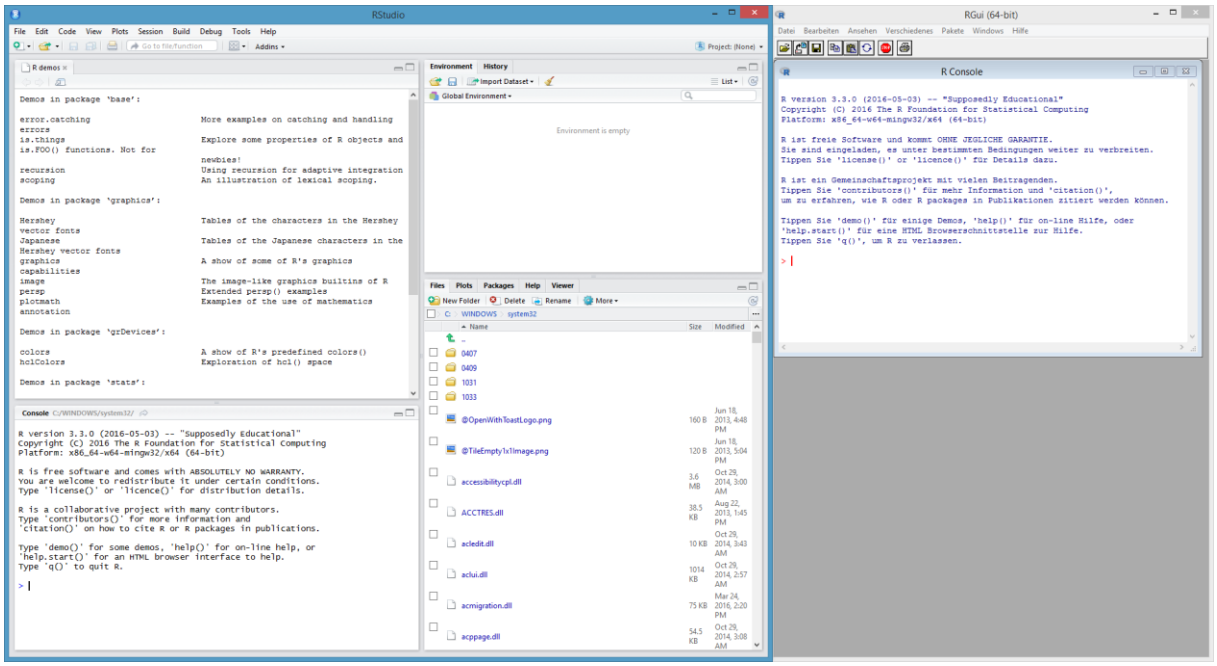

Abbildung 5: Screenshot RStudio & RGui

2.3 Beschreibung der Toolchain

Im Unterabschnitt „2.2 Screenshot der installierten Umgebung“ wurde „RStudio Desktop“ für Windows installiert. In Produktion würde ich „RStudio Server“ (nicht verfügbar für Windows) installieren und mit Hive verbinden, um so auf die Hadoop Daten zuzugreifen. Natürlich unter der Annahme, dass die Hadoop Daten die Quelle für die Datenanalyse darstellen.

3 Assignment: Big Science

Classification:
Hier geht es darum, Daten in Kategorie einzuteilen. Z.B. wenn ich mir ein Auto kaufen möchte, dann ist die Auswahl schlichtweg riesig. Man kann die Autos, aber anhand ihrer Eigenschaften (Antriebsart, PS, Anzahl Türen, Farbe, etc.) unterteilen. Aufgrund dieser Einteilungen kann man nun die Daten je nach gewünschter Eigenschaft sortieren und Überdeckungen zwischen verschiedenen Autos erkennen.

Regression:
Regression ist die am häufigsten eingesetzte Methode für Vorhersagen. Sie versucht einen bestimmten Wert vorherzusagen. Um beim Thema Autokauf bzw. -verkauf zu bleiben könnte man sagen, dass man mithilfe von Regression versuchen kann den Verkaufspreis eines Gebrauchtwagens vorherzusagen. Um den Verkaufspreis zu ermitteln zieht man vergleiche zu anderen Gebrauchtwagen und deren erzielte Verkaufspreise.

Clustering:
Clustering ähnelt der Classification, nur sind hier die Kategorien nicht vordefiniert. Auch hier eignet sich das Beispiel Autokauf: Angenommen ich begebe mich zu einem Autohändler und schildere die Eigenschaften, die ich mir bei einem Fahrzeug wünsche, dann kann der Verkäufer aufgrund der Fahrzeugdaten (die er natürlich aufgrund langer Berufserfahrung über alle Fahrzeuge im Kopf hat) eine Empfehlung für ein möglichst passendes Fahrzeug abgeben.

Dimensional reduction:
Ist die Datenmenge so groß, dass es bei der Verwendung von Analyse Tools zu schlechte Performance kommt oder die Daten aufgrund der vielen Dimensionen zu unübersichtlich sind, dann kann eine „dimensional reduction“ helfen. Typischerweise wird so etwas bei „machine learning“ - Problemen angewendet, um bessere Aussagen für classification- oder regression- Aufgaben tätigen zu können.
Um auch hier wieder beim Beispiel der Fahrzeuge zu bleiben: Hat der Datensatz z.B. eine Vielzahl an Herstellern, Modellen und Eigenschaften der Fahrzeuge inkl. der jeweiligen Herstellungsjahre und des Produktionsstandortes dann stellt dies eine Datenmenge mit sehr vielen Dimensionen dar. Sind für eine Data Science Aufgabe nun z.B. nur die PS-Leistung und Verbrauch von Autos, die in Europa im Jahr 2005 gefertigt wurden, relevant, so kann eine „dimensional reduction“ des Datensatzes durchgeführt werden, um den Datensatz auf diese wenigen Dimensionen zu verkleinern.

Abbildungsverzeichnis