Table of Contents

PBI - Fase 2

Exportação conjuntos de dados

Foi utilizado o software kettle referido na primeira fase, para gerar seis coleções de dados distintas com o objetivo de identificar regras de associação presentes nos dados. A imagem 1 demonstra a estrutura no kettle que realiza a query à base de dados Steelwheels e exporta para um ficheiro CSV o resultado.

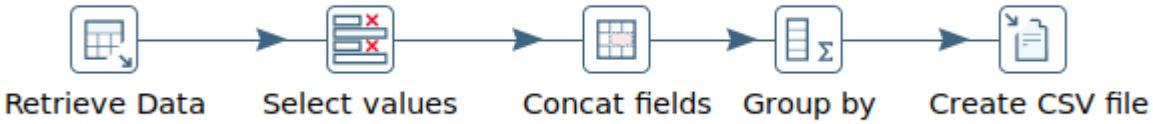


Illustration 1: Estrutura Kettle para criação dos conjuntos de dados

Inicialmente foram elaboradas seis perguntas de negócio relevantes, uma para cada conjuntos de dados. Estas poderão ser alteradas futuramente, consoante os resultados obtidos nos próximos passos.

Algumas questões foram adaptadas ou removidas de modo a satisfazer as condições presentes na identificação das regras de associação. Segue-se uma listagem das algumas questões que não cumpriram estes requisitos.

- Vendedor de produtos que tem sucesso num dado país, consegue transmitir esse sucesso para os mesmos países desse território no ano 2005;
- Clientes com crédito limite elevado realizam compras com valores mais elevados na América do Norte;
- Relação entre quantidade em stock e soma do número de artigos em França para 2003;
- Cliente que compra grandes quantidades de "Classic Cars" compra menos quantidades "Trains".
- Vendedor de produtos que realiza mais de 60 vendas num dado território, consegue transmitir esse sucesso para outros territórios;

Segue-se as perguntas definidas tais como os campos selecionados para fazer parte do ficheiro csv resultante:

1. Cliente que compra um produto também compra outro produto em 2004;

PRODUCTNAME	ORDERNUMBER	YEAR_ID = 2004

2. Cliente que compra uma linha de produtos também compra outra linha de produtos nos EUA;

PRODUCTLINE	ORDERNUMBER	COUNTRY = EUA

3. Clientes com crédito limite elevado realizam compras com valores mais elevados na América do Norte;
   (a) Foram criados grupos de valores referentes ao BUYPRICE e ao CREDITLIMIT, nomeadamente <30, 30-100 e >100 para o BUYPRICE e <100k, 100k-200k e >200k para o CREDITLIMIT.

BUYPRICE	CREDITLIMIT	TERRITORY = NA

4. Clientes de um dado território compram brinquedos numa escala maior que noutro território.
   (a) Foram categorizados em 3 classes os valores referentes ao PRODUCTSCALE nomeadamente "large" para as escalas "1:10 e 1:12", "medium" para as escalas "1:18, 1:24 e 1:32" e "small" para as restantes escalas inferiores.

PRODUCTSCALE	TERRITORY

5. Clientes com limite de crédito superior a 100.000 que compram uma linha de produtos também compram outra linha de produtos;

PRODUCTLINE	ORDERNUMBER	CREDITLIMIT > 100k

6. Clientes que realizam mais de 5 compras numa dada linha de produtos realizam menos de 5 compras noutra linha de produtos.
   (a) Foram criados grupos de valores referentes ao NMR_ORDERS, nomeadamente <=5 e >5.

PRODUCTLINE	ORDERNUMBER	NMR_ORDERS

O grupo tentou criar uma amostra de perguntas de negócio diversificadas que demonstrem a flexibilidade e capacidades das ferramentas utilizadas.

Transformação do CSV para ARFF

Foi desenvolvido um script em Java que realiza uma pesquisa na diretoria que contém os ficheiros CSV, armazena o nome dos ficheiros num array e seguidamente converte-os para formato ARFF, com auxílio da Função ArffSaver().

Segue-se os blocos de código referentes ao main onde está a ser implementada uma console application, listagem de ficheiros csv, modificação dos ficheiros csv para a conversão e conversão para ARRF.

Classe CSVtoARFF

```
public static void ConvertFiles() {
        String folderPath = "XXXX";

        ArrayList<String> csvFilesList = GetCSVFilesList(folderPath);
        for (int i = 0; i < csvFilesList.size(); i++){
            String fileName = csvFilesList.get(i);
            fileName = fileName.substring(0,fileName.length()-4);

            String filePath = folderPath + fileName;

            ConvertCSVtoARFF(filePath, fileName);
        }
    }



private static ArrayList<String> GetCSVFilesList(String path) {
        File folder = new File(path);
        File[] listOfFiles = folder.listFiles();

        ArrayList<String> result = new ArrayList<String>();

        for(int i = 0; i < listOfFiles.length; i++){
            String fileName = listOfFiles[i].getName();
            String fileExtension = fileName.substring(fileName.length()-3);

            if( fileExtension.equals("csv")){
                result.add(fileName);
            }
        }

        return result;
    }


 private static void ConvertCSVtoARFF(String filePath, String fileName){

        try{
            ModifyCSVFile(filePath);

            //Load CSV
            CSVLoader loader = new CSVLoader();
            loader.setSource(new File(filePath+"_mod.csv"));
            Instances data = loader.getDataSet();

            // Save ARFF
            ArffSaver saver = new ArffSaver();

            // Save as ARFF
            saver.setFile(new File("C:\\Users\\tadeu\\Desktop\\SAD2021\\SADG05\\PBI\\PBI2\\Current\\implementacao\\
test_conv\\"+fileName+".arff"));
            saver.setInstances(data);
            saver.writeBatch();
            System.out.println("SUCCESS: File Created -> "+fileName);
        }
        catch(IOException E){
            System.out.println("ERROR: The File "+ fileName +" cannot be converted");
            System.out.println(E);
        }
    }
```

```
  private static void ModifyCSVFile(String filePath) throws IOException {
     File csvFile = new File(filePath+".csv");

      try {

         BufferedWriter bw = new BufferedWriter(new FileWriter(filePath+"_mod.csv"));

         Scanner csvFileScanner = new Scanner(csvFile);
         csvFileScanner.useDelimiter("\n");

         while(csvFileScanner.hasNext()){
            String data = csvFileScanner.next();
            data = data.trim();
            data = data.replaceAll(" ","_");
            data = data.replaceAll("'","_");
            bw.write(data);
            bw.newLine();
         }
         bw.close();
         csvFileScanner.close();
      } catch (FileNotFoundException ex) {
         //Se der erro
         System.out.println(ex);
      }
  }
```

Classe WekaApriori

```
  public static void RunAlgorithm() throws FileNotFoundException, IOException, Exception  {

      String filePath = "C:\\Users\\tadeu\\Desktop\\SAD2021\\SADG05\\PBI\\PBI2\\Current\\implementacao\\
TASKDATA1.arff";

         // load data
      Instances data = new Instances(new BufferedReader(new FileReader(filePath)));

      System.out.println(data);
         // build model
         Apriori model = new Apriori();
         model.buildAssociations(data);
         System.out.println(model);

         FPGrowth fpgModel = new FPGrowth();
         fpgModel.buildAssociations(data);
         System.out.println(fpgModel);
   }
```

Classe Main

```
public static void main(String[] args) throws Exception{
        Scanner scan = new Scanner(System.in);
        Boolean running = true;
       while(running){
          DisplayOption();
          int i = scan.nextInt();
          switch (i){
             case 1:
                CSVtoARFF.ConvertFiles();
                break;
             case 2:
                WekaApriori.RunAlgorithm();
                break;
             case 3:
                running = false;
                break;
             default:
                System.out.println("Keep runing");
          }
       }
   }

   private static void DisplayOption() {
      System.out.println("1- Convert Files");
      System.out.println("2- Apriori Algorithm");
      System.out.println("3- Exit");
   }
```

Obtenção de regras de associação

1. Cliente que compra um produto também compra outro produto em 2004;

- [1930_Buick_Marquette_Phaeton=y] ==> [American_Airlines:_B767-300=y] Confidence: 1.0
- [American_Airlines:_B767-300=y] ==> [1930_Buick_Marquette_Phaeton=y] Confidence: 1.0
- [The_Schooner_Bluenose=y] ==> [1912_Ford_Model_T_Delivery_Wagon=y] Confidence: 1.0
- [1912_Ford_Model_T_Delivery_Wagon=y] ==> [The_Schooner_Bluenose=y] Confidence: 1.0
- [1936_Harley_Davidson_El_Knucklehead=y] ==> [1996_Moto_Guzzi_1100i=y] Confidence: 1.0
- [1904_Buick_Runabout=y] ==> [18th_century_schooner=y] Confidence: 1.0
- [18th_century_schooner=y] ==> [1904_Buick_Runabout=y] Confidence: 1.0
- [1932_Model_A_Ford_J-Coupe=y] ==> [1939_Chevrolet_Deluxe_Coupe=y] Confidence: 1.0
- [1939_Chevrolet_Deluxe_Coupe=y] ==> [1932_Model_A_Ford_J-Coupe=y] Confidence: 1.0
- [F/A_18_Hornet_1/72=y, 1930_Buick_Marquette_Phaeton=y] ==> [American_Airlines:_B767-300=y] Confidence: 1.0

2. Cliente que compra uma linha de produtos também compra outra linha de produtos nos EUA;

- [Trucks_and_Buses=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Vintage_Cars=y, Trucks_and_Buses=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Motorcycles=y] ==> [Classic_Cars=y] Confidence: 0.95
- [Ships=y] ==> [Vintage_Cars=y] Confidence: 0.9473684210526315
- [Trucks_and_Buses=y] ==> [Vintage_Cars=y] Confidence: 0.9444444444444444
- [Classic_Cars=y, Trucks_and_Buses=y] ==> [Vintage_Cars=y] Confidence: 0.9444444444444444
- [Trucks_and_Buses=y] ==> [Vintage_Cars=y, Classic_Cars=y] Confidence: 0.9444444444444444
- [Vintage_Cars=y, Motorcycles=y] ==> [Classic_Cars=y] Confidence: 0.9411764705882353
- [Classic_Cars=y, Ships=y] ==> [Vintage_Cars=y] Confidence: 0.9411764705882353
- [Vintage_Cars=y] ==> [Classic_Cars=y] Confidence: 0.9375

3. Clientes com crédito limite elevado realizam compras com valores mais elevados na América do Norte;

- [<30____^_<100k____=y] ==> [30-100_^_<100k____=y] Confidence: 1.0
- [>100___^_<100k____=y] ==> [30-100_^_<100k____=y] Confidence: 1.0
- [<30____^_100k-200k=y] ==> [30-100_^_100k-200k=y] Confidence: 1.0
- [30-100_^_100k-200k=y] ==> [<30____^_100k-200k=y] Confidence: 1.0
- [<30____^_<100k____=y, >100___^_<100k____=y] ==> [30-100_^_<100k____=y] Confidence: 1.0
- [>100___^_100k-200k=y] ==> [30-100_^_100k-200k=y] Confidence: 1.0
- [>100___^_100k-200k=y] ==> [<30____^_100k-200k=y] Confidence: 1.0
- [<30____^_100k-200k=y, >100___^_100k-200k=y] ==> [30-100_^_100k-200k=y] Confidence: 1.0
- [30-100_^_100k-200k=y, >100___^_100k-200k=y] ==> [<30____^_100k-200k=y] Confidence: 1.0
- [>100___^_100k-200k=y] ==> [30-100_^_100k-200k=y, <30____^_100k-200k=y] Confidence: 1.0

4. Clientes de um dado território compram brinquedos numa escala maior que noutro território;

- [large_^EMEA=y] ==> [medium^EMEA=y] Confidence: 1.0
- [small_^EMEA=y] ==> [medium^EMEA=y] Confidence: 1.0
- [large_^NA=y] ==> [medium^NA=y] Confidence: 1.0
- [small_^NA=y] ==> [medium^NA=y] Confidence: 1.0
- [large_^EMEA=y, small_^EMEA=y] ==> [medium^EMEA=y] Confidence: 1.0
- [large_^NA=y, small_^NA=y] ==> [medium^NA=y] Confidence: 1.0
- [small_^NA=y] ==> [large_^NA=y] Confidence: 0.9393939393939394
- [medium^NA=y, small_^NA=y] ==> [large_^NA=y] Confidence: 0.9393939393939394
- [small_^NA=y] ==> [medium^NA=y, large_^NA=y] Confidence: 0.9393939393939394
- [medium^NA=y] ==> [large_^NA=y] Confidence: 0.9210526315789473

Com esta pergunta negócio e tendo em conta as regras de associação que foram geradas com o algoritmo apriori, é possível verificar que no mercado situado na Europa, Africa e Médio Oriente que os clientes compram mais brinquedos de escala média enquanto que na América do Norte os clientes optam por comprar brinquedos de grande escala. Desta forma, a Steel Wheels pode investir mais em brinquedos de diferentes escalas conforme cada território, aumentando o stock desses produtos e gerar mais vendas.

5. Clientes com limite de crédito superior a 100.000 que compram uma linha de produtos também compram outra linha de produtos;

- [Vintage_Cars=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Trucks_and_Buses=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Vintage_Cars=y, Trucks_and_Buses=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Motorcycles=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Ships=y] ==> [Vintage_Cars=y] Confidence: 1.0
- [Ships=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Vintage_Cars=y, Motorcycles=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Classic_Cars=y, Ships=y] ==> [Vintage_Cars=y] Confidence: 1.0
- [Vintage_Cars=y, Ships=y] ==> [Classic_Cars=y] Confidence: 1.0
- [Ships=y] ==> [Vintage_Cars=y, Classic_Cars=y] Confidence: 1.0

Através desta hipótese de negócio é possível verificar através das regras de associação que o clientes com crédito superior a 100.000 que compram uma certa linha de produtos têm tendência a comprar também carros clássicos. Este padrão encontrado é bastante relevante no ponto de vista do negócio, pois desta maneira a companhia pode decidir investir mais nesta linha de produtos e desta forma não estagnar stock de produtos de outras linhas.

6. Clientes que realizam mais de 5 compras numa dada linha de produtos realizam menos de 5 compras noutra linha de produtos.

- [Vintage_Cars_^_>5=y, Trucks_and_Buses_^_>5=y] ==> [Classic_Cars_^_>5=y] Confidence: 0.9166666666666666

Através desta pergunta de negócio conseguiu-se obter a regra de associação que dita que um cliente que faça mais de 5 compras de Vintage Cars e mais de 5 compras de camiões e autocarros também fazem mais de 5 compras em carros clássicos com uma confiança de 92%. Desta forma, é possível verificar um cliente que compre estas linhas de produtos regularmente venham também a comprar carros clássicos e assim a companhia poderá por exemplo fazer campanhas de "bundles" ou descontos agregando um carro clássico de forma a aumentar as vendas, pois a procura existe.

Utilizaçao da API do Kettle

Foi utilizada a API do Kettle de modo a executar as transformações desenvolvidas anteriormente atrevés de código Java.

Primeiramente foi criado um projeto maven com um ficheiro pom.xml para importar as bibliotecas necessários com origem no repositório "http://nexus.pentaho.org/content/groups/omni". Segue-se a listagem da sbibliotecas importadas.

- kettle-core
- commons-vfs
- kettle-engine
- hsqldb
- kettle-ui-swt
- mysql-connector-java

Os blocos de código que se seguem representam o ficheiro pom.xl e RunTransformation.java, respetivamente.

```
  <dependencies>
    <dependency>
      <groupId>pentaho-kettle</groupId>
      <artifactId>kettle-core</artifactId>
      <version>7.1.0.32-246</version>
    </dependency>
    <dependency>
      <groupId>commons-vfs</groupId>
      <artifactId>commons-vfs</artifactId>
      <version>1.0</version>
    </dependency>
    <dependency>
      <groupId>pentaho-kettle</groupId>
      <artifactId>kettle-engine</artifactId>
      <version>7.1.0.32-246</version>
    </dependency>
    <dependency>
      <groupId>org.hsqldb</groupId>
      <artifactId>hsqldb</artifactId>
      <version>2.3.2</version>
     </dependency>
    <dependency>
      <groupId>pentaho-kettle</groupId>
      <artifactId>kettle-ui-swt</artifactId>
      <version>7.1.0.32-246</version>
    </dependency>
    <dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
      <version>8.0.23</version>
    </dependency>
  </dependencies>
```

```
import org.pentaho.di.core.KettleEnvironment;
import org.pentaho.di.core.exception.KettleException;
import org.pentaho.di.core.logging.LogLevel;
import org.pentaho.di.trans.Trans;
import org.pentaho.di.trans.TransMeta;

public class RunTransformation {
   public static void main(String[] args) throws KettleException {

      KettleEnvironment.init();
      TransMeta transMeta = new TransMeta("../TASKDATA1.ktr");
      Trans trans = new Trans(transMeta);
      trans.setLogLevel(LogLevel.ERROR);
      trans.execute(null);
      trans.waitUntilFinished();

      if(trans.getErrors() > 0){
         System.out.println("Some errors ocurred");
      }
   }
}
```

O projeto foi denvolvido para a versão 7 do data integration, pois as dependências não ocorrem como esperado nas versões mais recentes.

De forma a resolver este problema, foi criado uma nova versão desta funcionalidade num projeto não maven. Desta forma importamos as dependências através da importação direta de ficheiros jar, nomeadamente todas os ficheiros da pasta lib no data-integration e os ficheiros jar da pasta plugins/pdi-core-plugins.

De seguida criamos uma pasta Files e adicionamos o ficheiro kettle-password-encoder-plugins.xml que se encontra na pasta classes. Na IDE Netbeans apenas necessitamos de adicionar esta pasta na source.

Relativamente à integração com a componente weka, esta foi feita através de uma nova classe acessível pelo menu de consola desenvolvido. Nesta realizamos um loop nos ficheiros ktr das transformações e executamo-os um a um.

```
import java.io.File;
import java.util.ArrayList;
import org.pentaho.di.core.KettleEnvironment;
import org.pentaho.di.core.exception.KettleException;
import org.pentaho.di.core.logging.LogLevel;
import org.pentaho.di.trans.Trans;
import org.pentaho.di.trans.TransMeta;

public class Transformation {
   static void RunTransformations() throws KettleException {
      File folder = new File("../");
      File[] listOfFiles = folder.listFiles();

      KettleEnvironment.init();

      for (File listOfFile : listOfFiles) {
         String fileName = listOfFile.getName();
         String fileExtension = fileName.substring(fileName.length()-3);
         if( fileExtension.equals("ktr")){
                  TransMeta transMeta = new TransMeta("../" + fileName);
            Trans trans = new Trans(transMeta);
            trans.setLogLevel(LogLevel.ERROR);
            trans.execute(null);
            trans.waitUntilFinished();

            if(trans.getErrors() > 0){
               System.out.println("Some errors ocurred");
            }
         }
      }
   }
}
```

Java (Servlets/JSP)

Foram implementadas as funcionalidades desenvolvidas anteriormente com recurso a Servlets/JSP de modo a realizar as transformações da API do kettle e apressentar as regras obtidas através do algortimo Apriori na Dashboard CDF da plataforma pentaho.

Foi editado o ficheiro web.xml do tomcat para apontar para o ficheiro JSP de acordo com o how to disponibilizado.

```
<servlet>
        <servlet-name>PBI2</servlet-name>
        <jsp-file>/jsp/PBI2.jsp</jsp-file>
</servlet>

<servlet-mapping>
        <servlet-name>PBI2</servlet-name>
        <url-pattern>/PBI2</url-pattern>
</servlet-mapping>
```

Adicionalmente foi adicionado na pasta lib todas as dependências .jar do nosso projeto java, incluindo o ficheiro .jar gerado na IDE NetBeans.

Foi necessário definir os ficheiros incluídos na package de exportação como se segue na imagem, desta forma obtemos uma pasta lib com todas as dependências para correr o programa e acesso a todas as classes desenvolvidas.

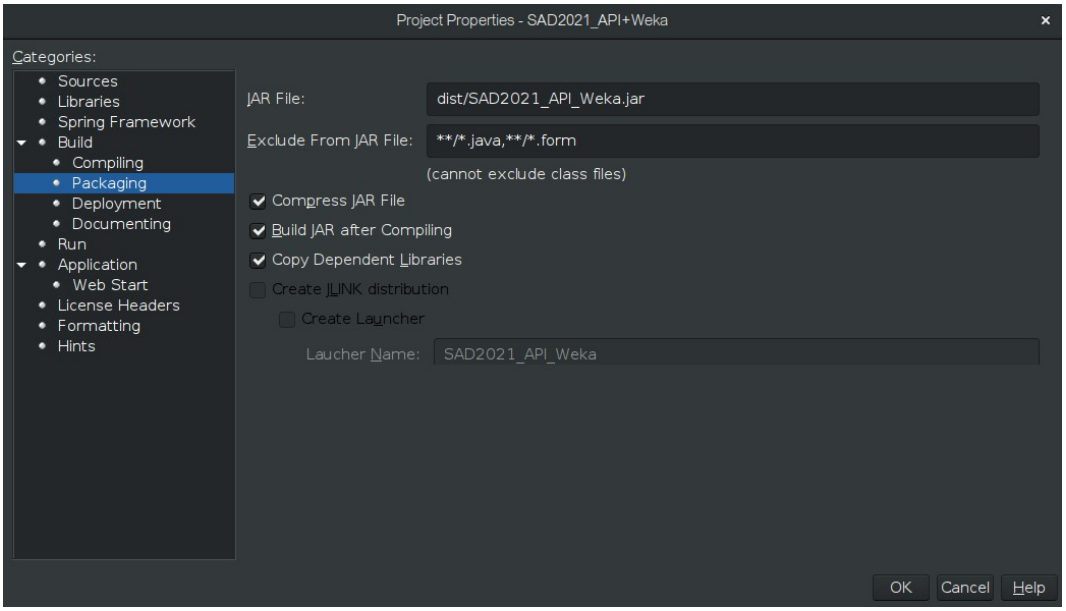


Illustration 2: Criação ficheiro jar

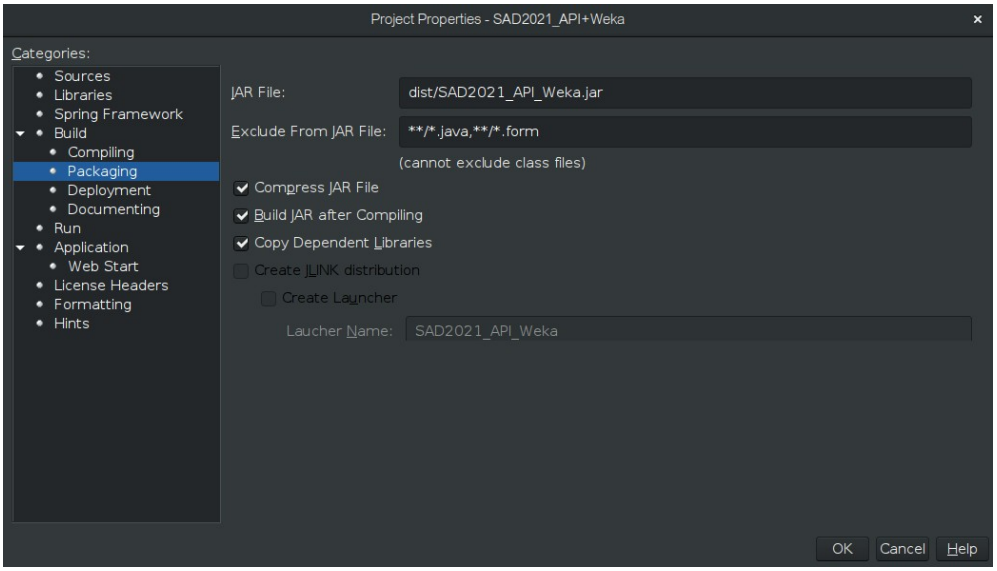
Seguidamente basta carregar no projeto e selecionar a opção “Clean and Build”, desta forma esta irá gerar uma pasta dist na raiz do projeto que contém o ficheiro .jar pretendido.

Relativamente ao ficheiro JSP este possui a seguinte estrutura.

```
<%@ page language="java" %>
   <%@ page import="java.util.List , WekaSolution.WekaApriori, weka.associations.AssociationRule" %>
<html>

<head>
   <meta http-equiv="content-type" content="text/html; charset=utf-8" />
   <title>PBI2</title>
</head>

<body>
   <%
   String path = "XXXX";
   WekaApriori apriori = new WekaApriori();
   List<AssociationRule> rulesX = apriori.RunAlgorithmToFile(path + "TASKDATAX.arff");
   %>

   <%
   out.println("<h3>TASKDATA X - XXXX</h3>");
      out.println("<ul>");
      for (AssociationRule item : rulesX) {
         out.println("<li>" + item + "</li>");
      }
      out.println("</ul>");
   %>

</body>
</html>
```

Foram importadas as funções necessárias no cabeçalho, nomeadamente listagem de regras, função do nosso projeto para obter as regras e a classe destas. Inicializamos a classe desenvolvida, neste caso WekaApriori e evocamos o método RunAlgorithmToFile para o ficheiro em questão. Seguidamente iteramos sobre a lista obtida e imprimimos no formato de lista.

De seguida foi criada uma página através do CDE - Dashboard Editor da community edition do pentaho, onde foi criado um iframe para a página JSP desenvolvida anteriormente, como visível na imagem que se segue.



Howtos

De forma a sistematizar os problemas encontrados e facilitar a execução de trabalho futuro usando as ferramentas do Pentaho, esta secção contém uma listagem de itens que, no momento da realização do trabalho, não constavam nos howtos online disponibilizados, de modo a que outros grupos possam usufruir destas novas soluções.

JAVA_HOME em sistemas UNIX.

1. Instalar JAVA 8 - algumas funcionalidades não funcionam em outras versões após consulta de documentação
    1. ***sudo apt install openjdk-8-jdk***
2. Descobrir caminho para a pasta JAVA
    1. ***dirname $(dirname $(readlink -f $(which javac)))***
3. Adicionar o caminho anterior ao ambiente de desenvolvimento
    1. Editar o ficheiro */etc/profile*
        1. Adicionar export ***JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64***
    2. Reiniciar sessão do utilizador
4. Verificar sucesso da operação
    1. ***echo $JAVA_HOME*** - deverá retornar o caminho detetado anteriormente
    2. ***java -version*** - deverá retornar a versão atualmente instalada do java

Considerações adicionais

1. Verificar a versão do java necessária para o software Pentaho aquando da instalação, pois a versão 8 poderá não ser a utilizada atualmente.
2. Como opção poderá definir o caminho da variável num ficheiro *.bashrc* ou *.zprofile*, dependo do interpretador de comandos UNIX que esteja a utilizar.
3. Poderá ser necessário exportar a variável PATH
    1. ***export PATH=${PATH}:${JAVA_HOME}/bin***

Software Data Integration em sistemas UNIX.

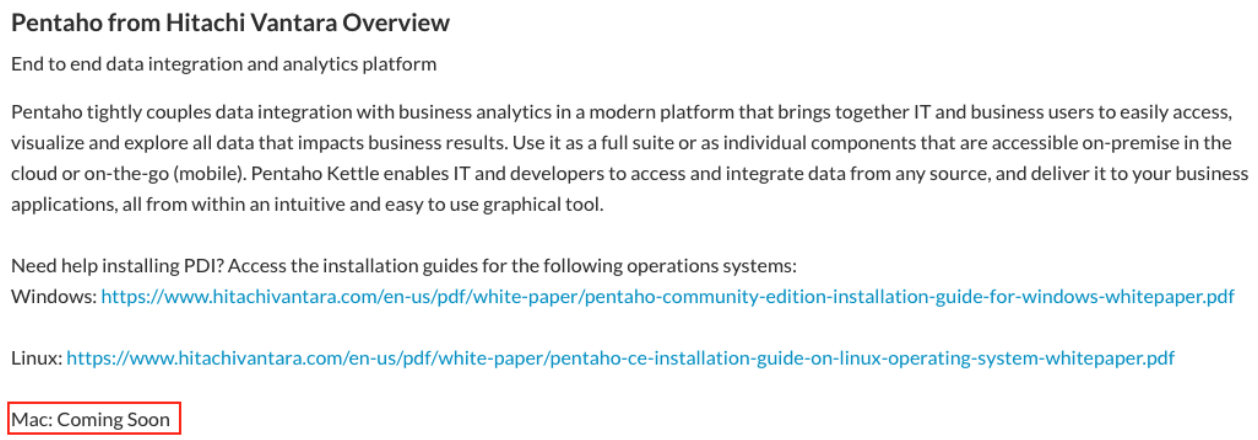
1. Requisitos
    1. Java
    2. Variáveis de ambiente configuradas
2. Download do ficheiro zip
    1. ***https://sourceforge.net/projects/pentaho/***
3. Extrair ficheiros para uma pasta à sua escolha
4. Executar o ficheiro spoon.sh que se encontra dentro da pasta data-integration
    1. *./spoon.sh*

Considerações adicionais

1. Como opção poderá seguir o tutorial fornecido na documentação da plataforma Pentaho. O sistema pedirá que efetue o registo, mas na realidade pode fornecer dados aleatórios pois não é realizada uma verificação do email.
    1. ***https://www.hitachivantara.com/en-us/pdf/white-paper/pentaho-ce-installation-guide-on-linux-operating-system-whitepaper.pdf***

Considerações MacOS:

Ao tentar instalar o pentaho community edition, deparamo-nos com a informação que este ainda não estava disponível para MacOS, para tal teve-se de recorrer à versão enterprise.
Nota: A versão enterprise quando se desinstala e se volta a instalar efetua uma nova renovação de licença.



Pentaho from Hitachi Vantara Overview

End to end data integration and analytics platform

Pentaho tightly couples data integration with business analytics in a modern platform that brings together IT and business users to easily access, visualize and explore all data that impacts business results. Use it as a full suite or as individual components that are accessible on-premise in the cloud or on-the-go (mobile). Pentaho Kettle enables IT and developers to access and integrate data from any source, and deliver it to your business applications, all from within an intuitive and easy to use graphical tool.

Need help installing PDI? Access the installation guides for the following operations systems:
Windows: https://www.hitachivantara.com/en-us/pdf/white-paper/pentaho-community-edition-installation-guide-for-windows-whitepaper.pdf

Linux: https://www.hitachivantara.com/en-us/pdf/white-paper/pentaho-ce-installation-guide-on-linux-operating-system-whitepaper.pdf

Mac: Coming Soon

Instalação plugin Saiku:

Nos sistemas MacOS tendo a versão enterprise instalada existe uma alteração na diretoria.
Sendo assim deve-se introduzir a pasta que está dentro do ficheiro zip (disponível aqui), na seguinte diretoria: **/Applications/Pentaho/server/pentaho-server/pentaho-solutions/system**, ficando assim **/Applications/Pentaho/server/pentaho-server/pentaho-solutions/system/saiku**. Os restantes passos podem ser seguidos no Howto I.
Para iniciar o servidor local é executar o ficheiro **start.command**, disponível na diretoria **/Applications/Pentaho**, para desligar e salvar as alterações do servidor local é só executar o ficheiro **stop.command** disponível na mesma diretoria.

Conexão à base de dados no data-integration

1. *Ao configurar uma nova ligação à base de dados que contém a amostra Steelwheels* a password da conexão poderá ser não só "pentaho_user", mas também "**password**".

Registo de Trabalho

Global	Individual	Individual	Individual	
198,5	126,5	126,5	126,5	
# Participantes	**Leonardo Abreu**	**José Freitas**	**João Franco**	**Descrição**
1	X	O	O	Modificação das transformações no Kettle para a exportação ficheiro csv
1	X	O	O	Modificação das transformações no Kettle para a exportação ficheiro csv
1	X	O	O	Modificação das transformações no Kettle para a exportação ficheiro csv
1	X	O	O	Modificação das transformações no Kettle para a exportação ficheiro csv
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Modificação do código java, para modificar o ficheiro CSV
1	X	O	O	Alterções no código responsável por gerar o ficheiro ARFF
1	X	O	O	Alterções no código responsável por gerar o ficheiro ARFF
1	X	O	O	Alterções no código responsável por gerar o ficheiro ARFF
1	X	O	O	Escrita Relatório
1	X	O	O	Escrita Relatório
1	X	O	O	Limpeza do diretorio eliminando os temporários
1	X	O	O	Limpeza do diretorio eliminando os temporários
1	O	O	X	Modificação das transformações no Kettle para a exportação ficheiro csv
1	O	O	X	Modificação das transformações no Kettle para a exportação ficheiro csv

1	O	O	X	Modificação das transformações no Kettle para a exportação ficheiro csv
1	O	O	X	Modificação das transformações no Kettle para a exportação ficheiro csv
1	O	O	X	Modificação das transformações no Kettle para a exportação ficheiro csv
1	O	O	X	Análise da regras de associação criadas
1	O	O	X	Análise da regras de associação criadas
1	O	O	X	Análise da regras de associação criadas
1	O	O	X	Modificação do código java, para modificar o ficheiro CSV
1	O	O	X	Modificação do código java, para modificar o ficheiro CSV
1	O	O	X	Debug de erros do script em java
1	O	O	X	Debug de erros do script em java
1	O	O	X	Debug de erros do script em java
1	O	O	X	Escrita do relatório
1	O	O	X	Escrita do relatório
1	O	O	X	Escrita do relatório
1	O	O	X	Escrita do relatório
1	O	O	X	Escrita do relatório
1	O	X	O	Usar Java (Servlets/JSP) no portal pentaho
1	O	X	O	Usar Java (Servlets/JSP) no portal pentaho
1	O	X	O	Aprendizagem relativa a JSP
1	O	X	O	Aprendizagem relativa a JSP
1	O	X	O	Aprendizagem relativa a JSP
1	O	X	O	Inicializacao de java API para transformacoes através de JSP
1	O	X	O	Inicializacao de java API para transformacoes através de JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	Obtenção de regras no JSP
1	O	X	O	CDF - Dashboard Framework e o CDE - Dashboard Editor
1	O	X	O	CDF - Dashboard Framework e o CDE - Dashboard Editor
1	O	X	O	CDF - Dashboard Framework e o CDE - Dashboard Editor
1	O	X	O	CDF - Dashboard Framework e o CDE - Dashboard Editor