RapidMiner チュートリアルとその解説

東京電機大学工学部3年   
情報通信工学科  
17EC084 平田智剛

# 第1章 目的

本稿はデータマイニングソフトRapidMinerのtutorialの英文を和訳し、同時に専門用語に関して補足説明をしていくことで、データマイニングソフトが何をどうするためのものであるか、解説する目的で執筆されたものである。

# 第2章 RapidMinerとは

## 2-1. RapidMinerの用途

RapidMinerは、データマイニングを行うソフトウェアである。複雑なテキスト入力をほとんど必要とせず、ブロックを組み立てることで、直観的、視覚的に操作できることが特徴といえるだろう。

データマイニングとは、統計学、パターン認識、人工知能、機械学習などデータ解析の技法を大量のデータセットに対して適用し、異常値、パターン、相関を発見するプロセス\*のことである。[1]

以下、RapidMinerの公式ホームページ[2]からの引用である。

RapidMiner（ラピッドマイナー）はプログラムの知識がなくても、データサイエンティストが行うような高度な分析業務をドラッグ&ドロップの簡単な操作で行うことができる、世界中で使われている機械学習プラットフォームです。

データ加工（前処理）からデータ可視化、モデル作成、評価、運用までを一つのプラットフォームで行うことができるのが特徴です。

## 2-2. RapidMinerの導入方法

以下の手順で、無料でRapidMiner Free版を導入することができた。  
1. ブラウザでhttps://rapidminer.com/get-started/ へアクセスした。

2. 右上にあるGET STARTED をクリックした。

3. 「Start your free 30-day trial」と書かれていることを確認の上、フォームに大学のメールアドレスを入力[3]し、job FunctionではAcademic / Researchを選択し、IndustryではOtherを選択した。その後DOWNLOADをクリックした。

# 第3章 Tutorialsの解説

## 3-1.Tutorialsとは

RapidMinerをダウンロード、インストールしたのち起動すると、図3.1.1のように、Startタブ、Recentタブ、Learnタブを持つ画面が表示される。この画面が表示されず、図3.1.2の画面が表示された場合、HelpタブからTutorialsを選ぶことで、図3.1.1の画面を得る。

図3.1.1のLearnタブをクリックすると、Get started(0/8)、Prepare data(0/6)などのメニューが表示される。これらに番号Aを振る(Get startedを1、Prepare dataを2・・・とする)。各メニューをクリックすると、細分化されたメニューがすぐ右隣りに表示される。(ここに振られている番号をB)とする。細分化されたメニューをクリックすると、図3.1.2の画面が変化し、図3.1.3のようなTutorialsパネルが左側に表示される。図3.1.3のような、ウィンドウのようなものを、RapidMinerではパネルという[4]。

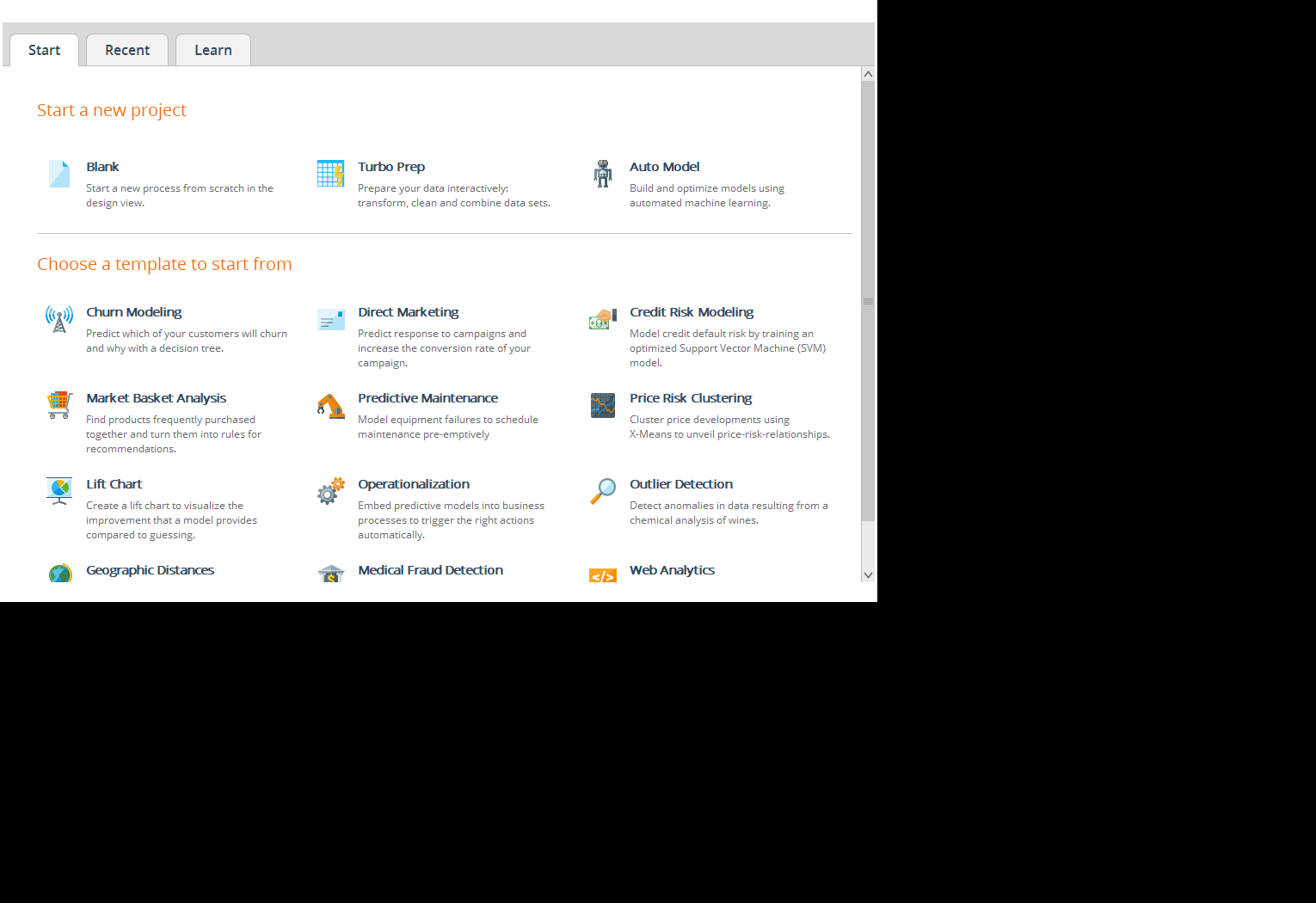
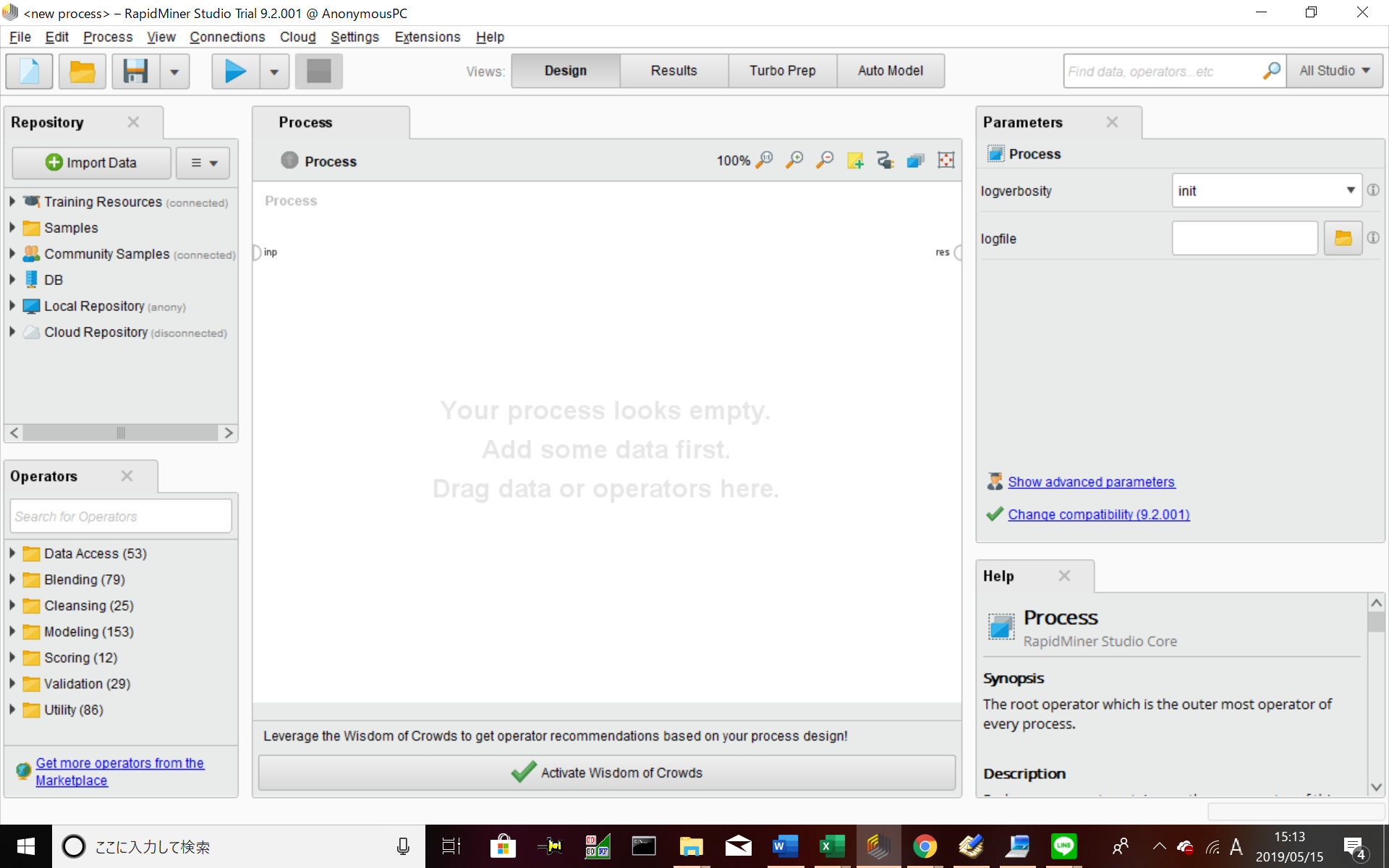


図3.1.1 Welcome to RapidMiner Studioウィンドウ

  
図3.1.2 RapidMinerのメインウィンドウ

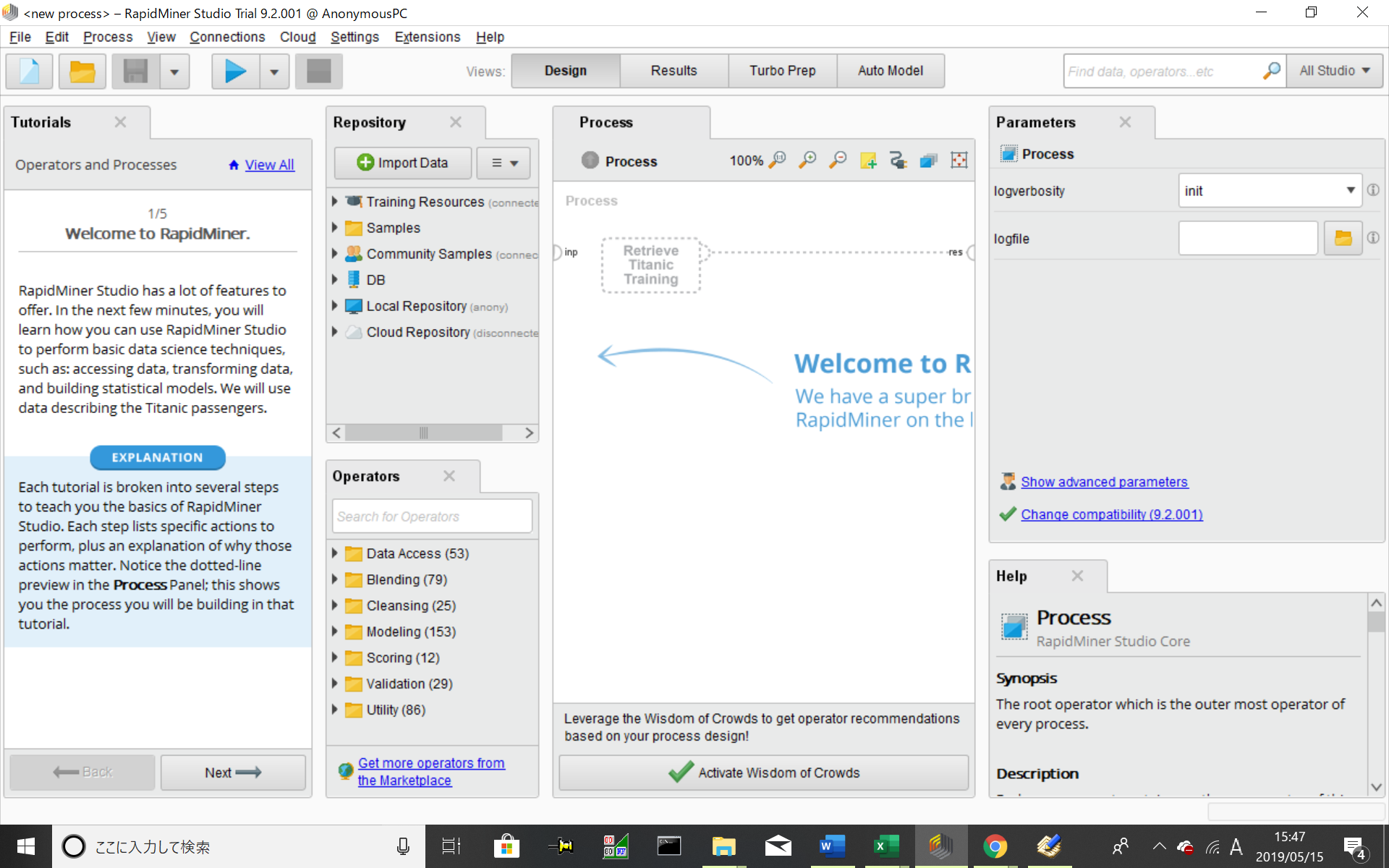


図3.1.3 Tutorials パネルの例

Tutorialsパネルでは、選択された図3.1.1のLearnタブで選んだものに応じて、RapidMinerを実際に動かしながら、このソフトに何ができるのかを紹介する文章が表示される。本稿ではこの内容をTutorials A-Bと呼ぶ。(AとBは前述した番号である)

## 3-2.Tutorialsの和訳と解説

Tutorialsは英語で書かれていて、しかも専門用語の解説がないため、  
データサイエンス\*(以下、「\*」は「用語等解説」にて解説されている単語であることを示す)の入門者には少々不便である。そこで、本稿ではTutorialsの和訳を行うと同時に、専門用語をその都度解説していく。

### Tutorials1-1 Operators and Processes

#### 原文と和訳

Welcome to RapidMiner.

ようこそRapidMinerへ

RapidMiner Studio has a lot of features to offer.

RapidMiner Studio(デスクトップ版RapidMinerの正式名称)はたくさんの機能を提供します。

In the next few minutes, you will learn how you can use RapidMiner Studio to perform basic data science techniques, such as: (～～～)

これから、RapidMiner Studioを使って(～～～)などといった基本的なデータサイエンスを体験する方法を学んでいきましょう。

accessing data, transforming data, and building statistical models.

データのアクセス、変換、統計モデルの構築

explanation

説明

Each tutorial is broken into several steps to teach you the basics of RapidMiner Studio.

各チュートリアルはRapidMinerの基本を(わかりやすく)説明するため、数ステップに分かれています。

Each step lists specific actions to perform, plus an explanation of why those actions matter.

各ステップでは、実行していただきたいことが、目的と合わせてリストされています。

Notice the dotted-line preview in the Process Panel; this shows you the process you will be building in that tutorial.

Process\* パネルの中の点線で描かれているものに注目してください。この点線は、このチュートリアルで構成してほしいブロックを示しています。

Retrieve data.

データの取り込み

Let’s start by retrieving some data about the Titanic passengers.

(今回は例として、)タイタニック号の乗客のデータを取得してみましょう。

Activity1

やること1(番号は本稿で勝手に振りました)

1.Find the Repository panel on the left of the screen.

1.画面左側にあるRepository\*パネルをみてください。

2.Open the folders Samples, and then data.

2.Samplesフォルダを開き、さらにその下のdataフォルダも開きます

3.Drag the Titanic Training dataset and drop it in the Process panel in the middle of the screen.

3.(dataフォルダの中の)Titanic Training dataset\*を画面中央のプロセス\*パネル(の、点線のところ)までドラッグします。

explanation

説明

Good job! You have just added your first operator to RapidMiner Studio, namely the Retrieve Operator.

　　Activity1の操作により、RapidMiner Studioに「演算子」を追加することができました(※1)。今回の演算子は「データ取得演算子」  
(Retrieve\* 演算子\*)です。

Operators perform actions: in this case, it retrieves data from the Repository.  
演算子は、処理を実行してくれます。今回の場合は、Repositoryからデータを取得します。

Now let’s see what you can do with it!

以降では、演算子がどう働くのか、実際に確認していきます。

Building your first process.

プロセスの構築

As a RapidMiner analyst, you create processes by adding and connecting operators. Operators are connected together via their Ports:

RapidMinerアナリストは、演算子を追加してそれぞれを接続することでプロセスを組み立てます。演算子はポートを介して互いに接続されます。

Activity2

やること2

1.Connect the output port ("out") of Retrieve Titanic Training with the result port ("res") on the right side of the Process panel.

1.Titanic Trainingを取得したデータ取得演算子

(以下、これを「データ取得演算子(Titanic Training)」または  
「Retrieve(Titanic Training)」と表現します)

の “out” ポート(出力ポート)を、Processパネルの右端にある結果ポート( “res” ポート)に接続してください。

2.Make the connection either by dragging a line between the ports, or by first clicking on one port and then on the other port.

2.両ポートの間をドラッグして接続するか、片方のポートをクリックしてそれに続けてもう片方のポートをクリックします。

explanation

説明

Excellent! You have built your first process in RapidMiner Studio.

以上で、RapidMiner Studioにおけるプロセスの構築ができました。

We can now look at the output of the Retrieve operator.

これで、データ取得演算子が出力する内容を見ることができるようになりま  
した。

Whenever you want to see the output of an operator, make sure it is connected to the 'res' port.

演算子の出力を確認したいときは、必ずその演算子を “res” ポートに接続するようにしてください。

Running processes.

プロセスの実行

After we have connected the elements of our process, we can hit the 'Run' button to view the output.

プロセスを正しく接続したら、(データ取得演算子の)出力を確認するために “Run” ボタンを押してください。

Activity3

やること3

1.Click Run (top left) to execute the process.

1.画面左上にある実行ボタン(▶)をクリックし、プロセスを実行します。

explanation

説明

Nice! You have run your first process in RapidMiner Studio.

これで、RapidMiner Studioで「プロセスを実行する」ことができました。

When you click "Run", the operators in your process will perform their actions.

▶をクリックすると、プロセス内の演算子が動作します。

RapidMiner executes the process and shows the data connected to the result port.

そして、“res”ポートに接続されているデータを表示します。

In the center of the Results view, you can see our raw data about the Titanic passengers such as their family size or their age.

Resultsビュー(の中央付近)に、タイタニック号の乗客に関する生データ(家族の人数や年齢など)を確認できます。

On the Statistics tab we get a statistics summary, which provides useful information; for example, 349 passengers survived the Titanic accident.

Statistics(統計)タブで、有用な情報を提供する統計の概要が表示されます。例えば、349人の乗客がタイタニック号の事故で生還したことがわかります。

Congratulations - Halfway there!

半人前になりました。

Well done - you just mastered your first tutorial! To quickly review:

これでTutorial1-1をマスターしました。要点をまとめると次の通りです。

･Operators perform individual actions.

演算子にはそれぞれ固有の働きがあります。

･You can connect operators together to build a Process.

演算子を接続することでプロセスを構成することができます。

･To see the output of an operator, connect it to the 'res' port.

演算子の出力をみたいときは、その演算子を “res” ポートに接続します。

･Running the process performs all operators and automatically shows the results

プロセスを実行すると、すべての演算子を動作させ、その結果をみることができます。

explanation

説明

To become a true master move on to the next tutorial below.

真のマスターになるには、以降のチュートリアルにも取り組んでください。

#### Activity

Activity1を実行した結果、図3.2.1.1.2.1のようになった。

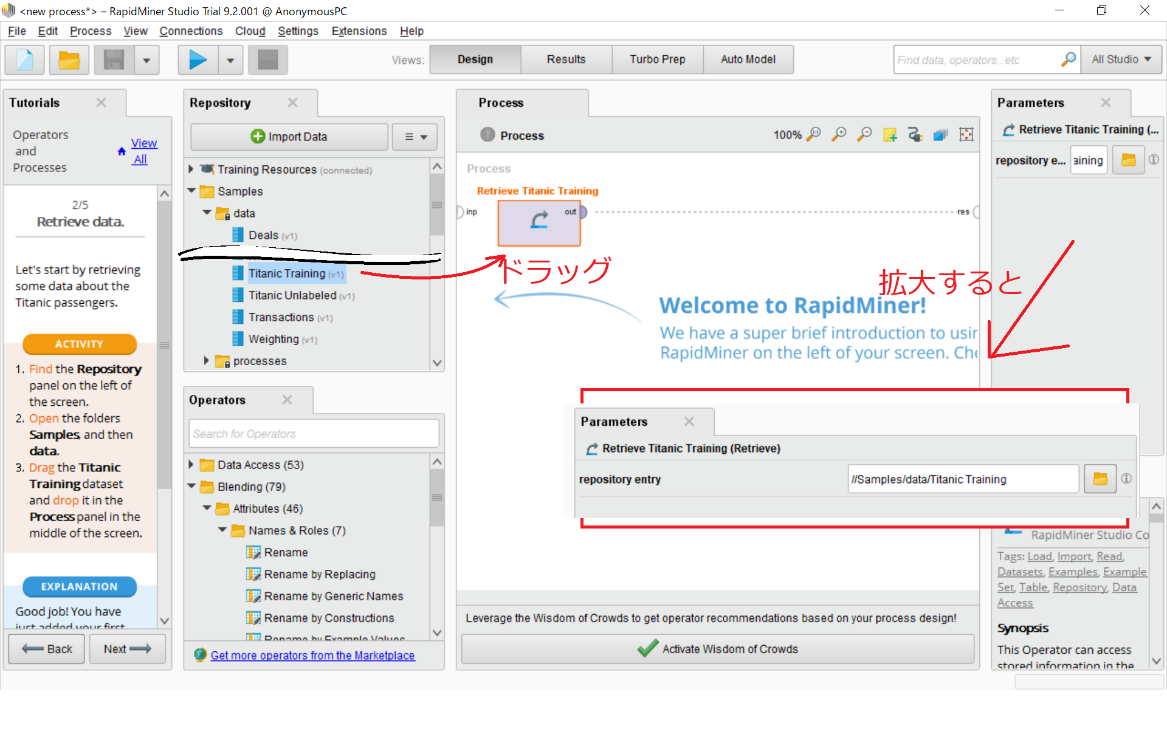
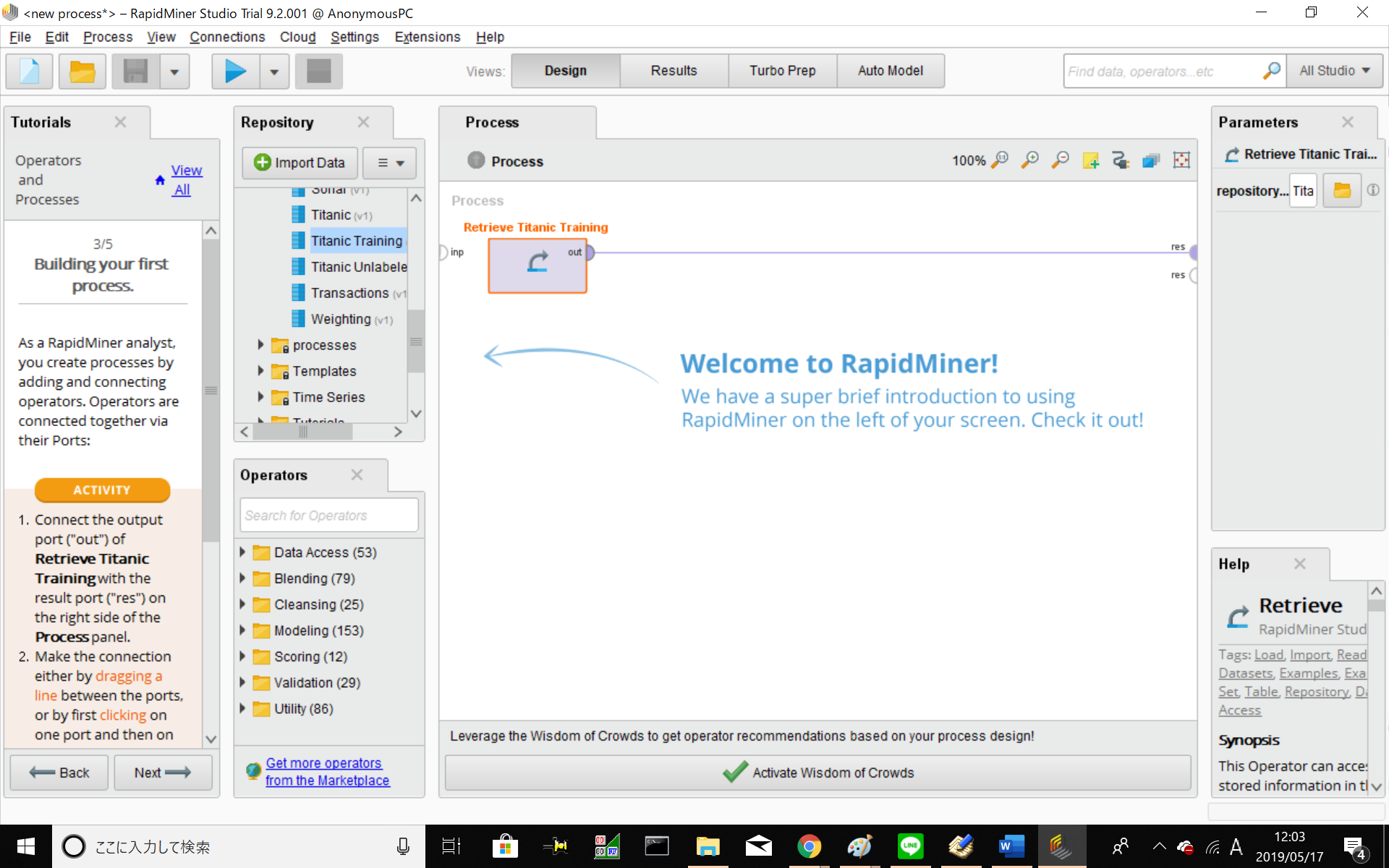


図3.2.1.1.2.1 Tutorial1-1 Activity1 の実行画面

RepositoryパネルからTitanic Training というデータをProcessパネルまでドラッグすると、図3.2.1.1.2.1の通り、データ取得演算子が生成され、その引数はTitanic Training となる。演算子の引数は、その演算子を選択の上、Parametersパネルから確認できる。

　Activity2を実行した結果、プロセスパネルは図3.2.1.1.2.2のようになった。

  
図3.2.1.1.2.2 Tutorial1-1 Activity2 を実行したときのProcessパネル

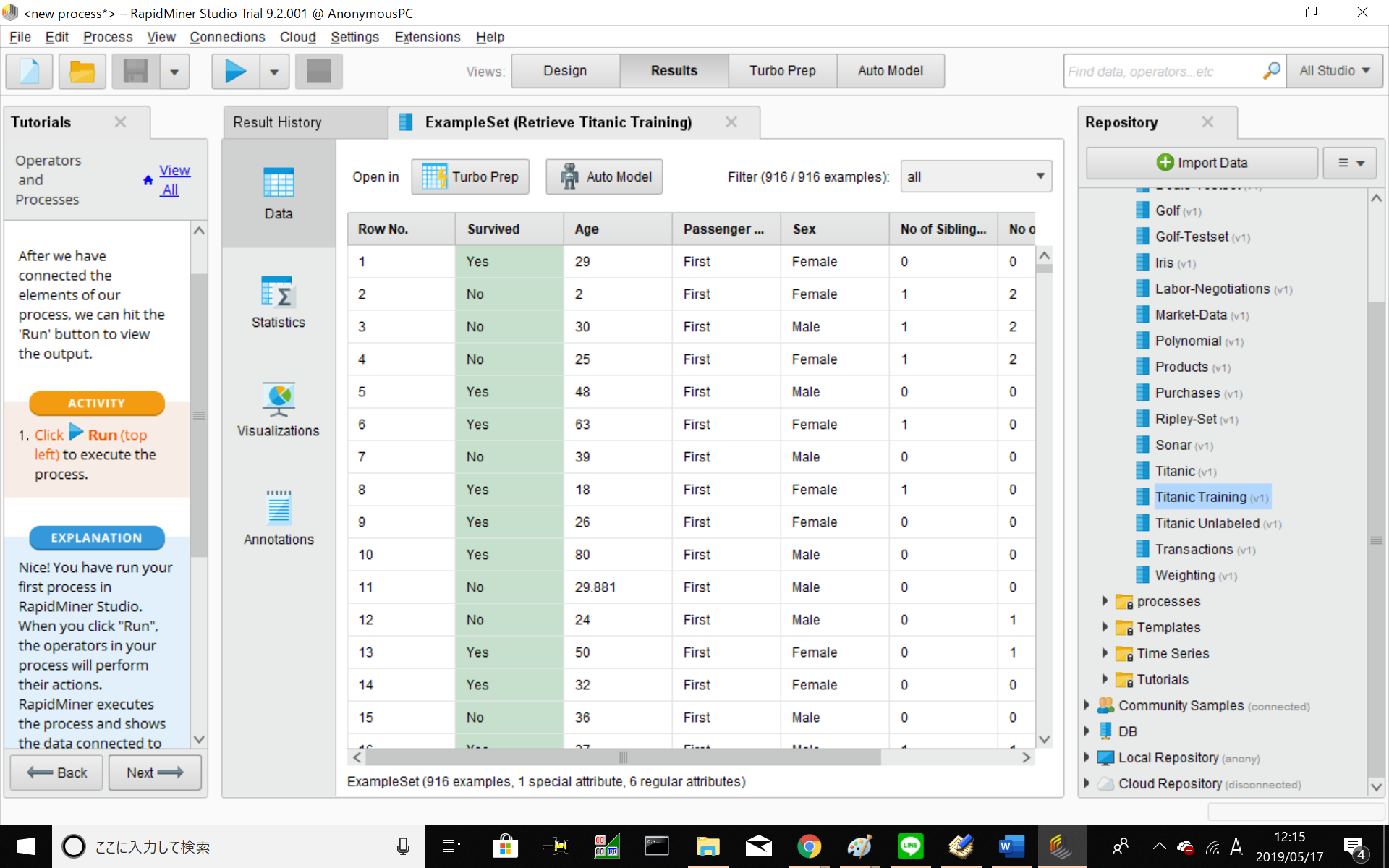


図3.2.1.1.2.3 Tutorial1-1 Activity3 を実行したときの実行画面

図3.2.1.1.2.3ではDataタブが選択されていて、そこでは生のデータ(分析などを施さず、アンケートなどで集計した直後のデータ)が表示されている。  
Row No.は行の番号  
Survivedは生存した(Yes)か否(No)か  
Ageは年齢  
Passenger Classはグレード(JRの電車でいう普通車やグリーン車のようなもの)、  
Sexは性別(Female=女、male=男)  
である。

Statisticsタブに切り替えると、パネルは図3.2.1.1.2.4のようになる。

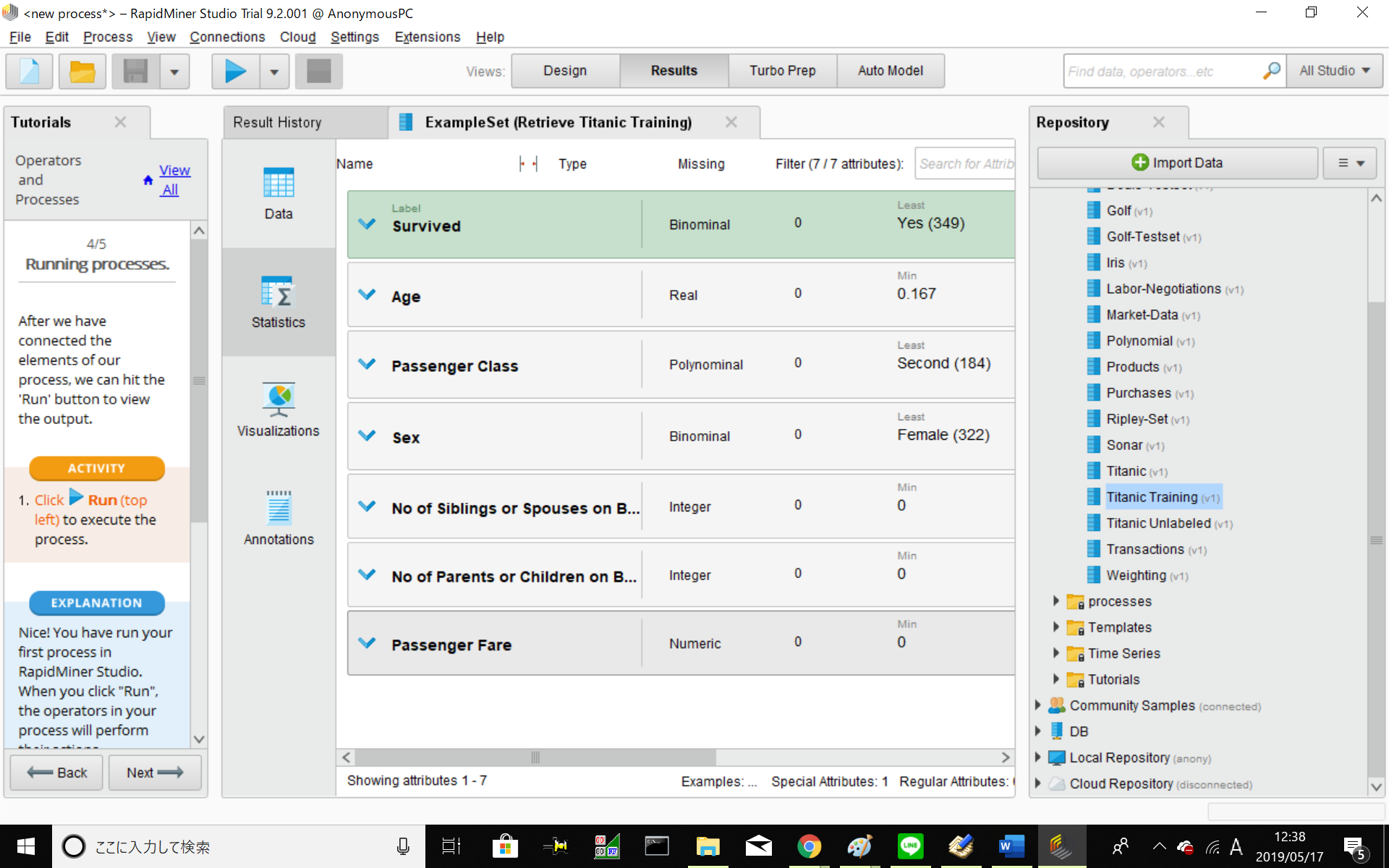
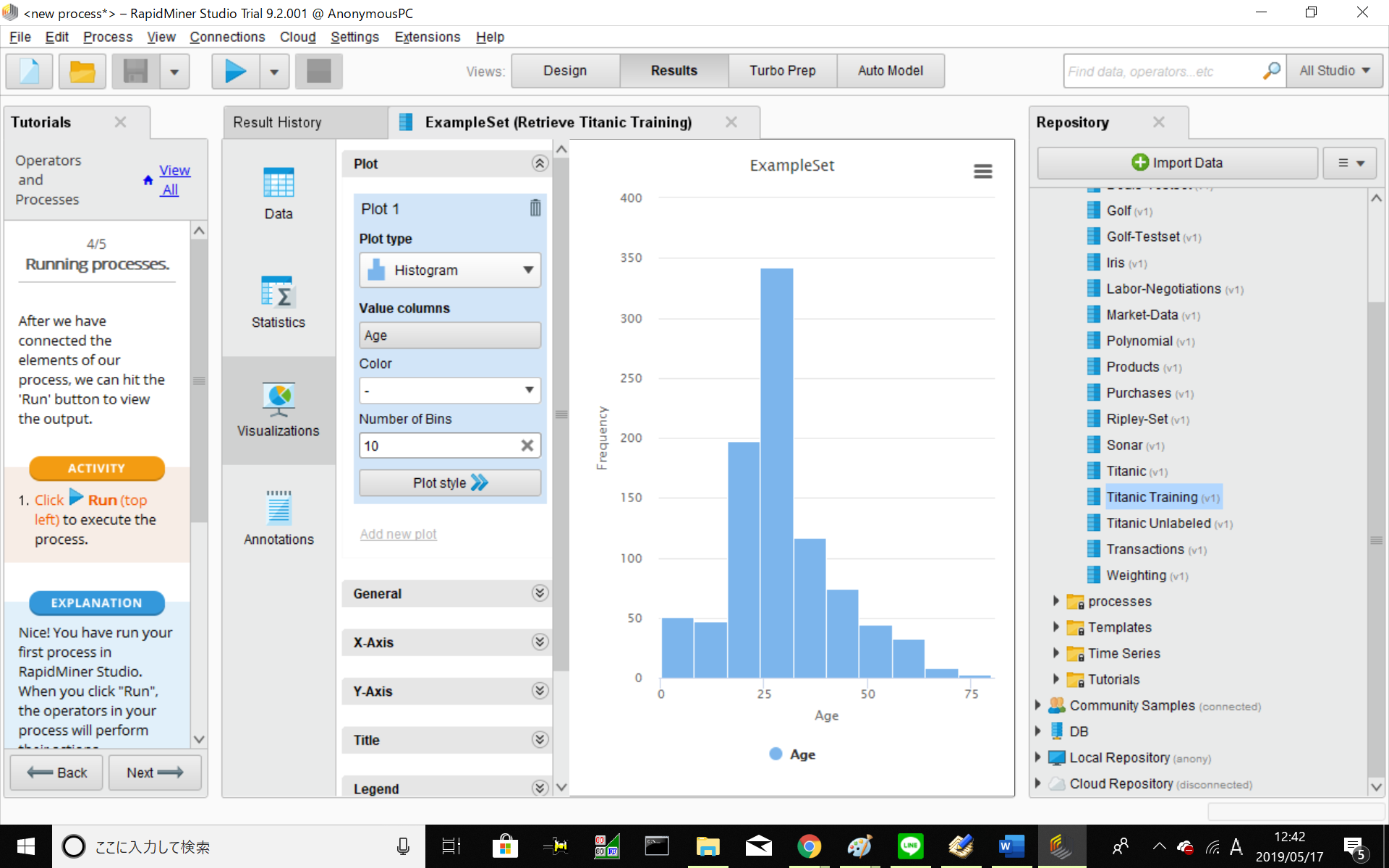
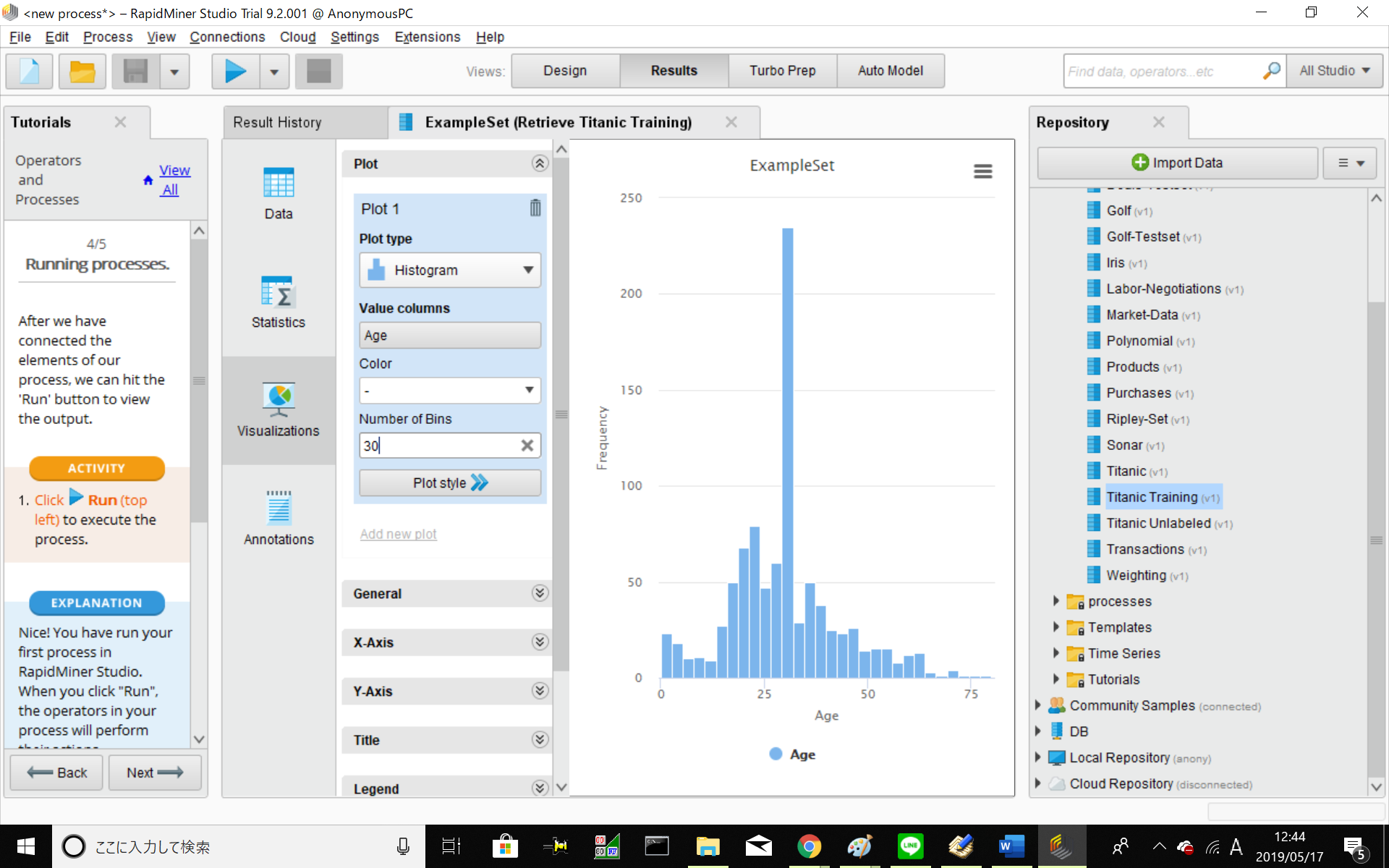


図3.2.1.1.2.4 Statisticsタブの内容

　図3.2.1.1.2.4の画面でAgeをクリックし、open visualizationをクリックすると、図3.2.1.1.2.5のような棒グラフを得る。  
 

(a) (b)

図3.2.1.1.2.5 年齢(Age)の出現数(Frequency)を示す棒グラフ

(a)では、binの値を10に、(b)では30に設定したものである。図と見比べることで、binは棒グラフの棒の本数を示すものと分かる。binを大きくすると、(b)のようにより細かくデータをみることができる。

#### 用語等解説

##### データサイエンス[5][6]

様々なデータの共通点を見つけ、そこから何らかの結論を導くために用いられる学問のこと。デジタル大辞泉[6]による説明を引用すると、次の通り。

データの分析についての学問分野。統計学、数学、計算機科学などと関連し、主に大量のデータから、何らかの意味のある情報、法則、関連性などを導き出すこと、またはその処理の手法に関する研究を行う。これらの研究者および技術者はデータサイエンティストとよばれる。

RapidMinerでは、自動モデリングの機能があり、またデータ可視化の方法も豊富である。さらに本来複雑な処理を、演算子を組み立ててプロセスを構築するだけで実行可能である。つまり数学、統計学、計算機科学の難しさを、ソフトウェアによる自動化が解消しているということもできるだろう。

##### retrieve(リトリブ、レトリブ、データ取得)[7][8]

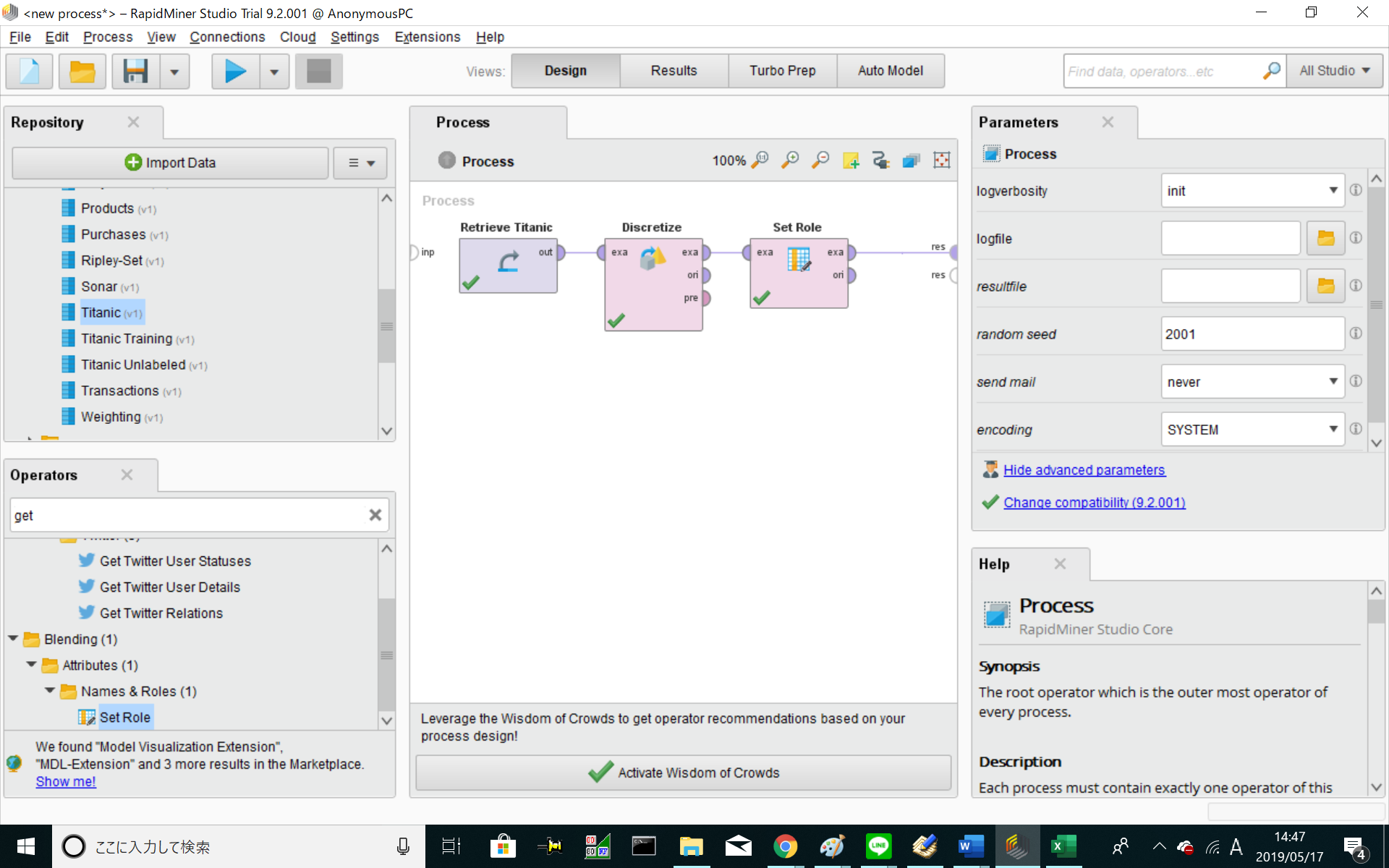
　retrieveは、RapidMinerにおいて、「データ取得」を意味する単語である。

retrieveは「取り戻す」、「回収する」、「救い出す」などを意味する他動詞であるが、目的語がdataなどの場合、「検索する」や「取得する」という意味になる。「マイクロソフト用語集」ではretrieveの意味の一つを「保留解除」と説明している。今、自分の手元にない状態を仮の状態とみなし、その状態を解除することと考えれば「取り戻す」と「検索する」や「取得する」の言葉の共通点が見いだせる。

また、コンピュータに関わる用語は日外アソシエーツが提供する  
「コンピュータ用語辞典」[8]でも解説されている場合がある。

##### process(プロセス)

process(プロセス)は、実行したい処理を記述した意味の塊であり、視覚的にはブロックの塊である。プログラミングにおけるコードのようなものである。それぞれのブロックはポートを介して接続されていて、最終的な出力が “res” ポートに接続されて初めて成立する。プロセスはProcessパネル内で構築される。図3.2.1.1.3.1はプロセスの例である。

  
図3.2.1.1.3.1 Processパネル内に構築されたプロセスの例

##### repository(リポジトリ、レポジトリ)[8]

repositoryとは設計情報を保管するデータベースやディレクトリのことである。

##### training dataset[8]

training datasetは訓練データを意味する単語である。trainingが訓練の意味で、datasetはある規則に則って配列されたデータの集合のことである。

訓練データというのは主に機械学習の教師あり学習で使われる、学習用のデータを意味する。(ここでも同じ意味かは不明)

##### operator(オペレータ、演算子)

RapidMinerにおける演算子は、プロセスを構成する要素であり、入力されたデータなどに適切な処理を施して出力を行うものである。

演算子というものの、入力を受け取り処理を行って出力を行う本質は、プログラミングでいうところの関数(メソッド)に似ているものである。

そこで、本稿では演算子やその引数を  
演算子名(引数名=値, 引数名=値, ...)  
のように表現することがある。

##### row

表において、列のこと

##### bin[9]

RapidMinerにおけるbinは棒グラフでいう、棒の本数である。厳密には、数値属性を(等範囲で)離散化するとき、いくつに離散化するかを示す数字のことである。

#### 演算子の説明[9]

##### データ取得演算子(Retrieve演算子)

###### 概要

　データ取得演算子はレポジトリに保存された情報にアクセスし、それをプロセスへ読み込むという働きをする。

　アクセスされる情報(RapidMiner Object)はExampleSet、Collection、Modelである。

この演算子では、メタデータを一緒に読み込むことができるのが特徴である。メタデータは、RapidMiner Objectに関する追加情報で、ExampleSetの場合、属性の名前、型、範囲、欠落地の数などである。メタデータがあることで、他の演算子のパラメータをプルダウンメニューから選べるようになり、簡単に設定できるようになる。

※1  
Repositoryパネルから演算子(＝operators)を追加したことになるのならOperatorsパネルに並んでいるもの(図3.2.1.1.2.1にみられるRename、Rename by Replacingなど)は演算子ではないのかという疑問が生じるであろう。実はOperatorsパネルにはRetrieveというものも並んでおり、RepositoryパネルからProcessパネルへ、データ(Titanic Training datasetやTitanic datasetなど)がドラッグされた場合、実際にはデータ取得演算子(Retrieve演算子)が追加され、データ取得演算子の引数(Parameters)として、データが指定される。(図3.2.1.1.2.1のParametersパネルでも、Titanic Training datasetが引数repository entryの値に指定されていることが確認できる。)

###### 出力

引数「repository\_entry」で指定されたパスを持つRapidMiner Object

###### 引数

repository\_entry

読み込みたいRapidMiner Objectのパス

(パスの文法は、参考資料[9]に詳しい)

### Tutorials1-2 Modeling

#### 原文と和訳

Let's do some data science.

データ科学

In the last tutorial, we imported a dataset and ran a process.

Tutorials1-1では、データセットを読み込み、プロセスの実行をしてみました。

We found out that 349 passengers survived the Titanic accident.

その結果、タイタニック号の事故で349名の乗客が生還したことがわかりました。(図3.2.1.1.2.4でsurvivedの行にYes(349)とある)

Let's build a Process to see if those survivors have something in common or not.

これらの生存者に何か共通点があるのか否かを探るべく、プロセスを構築しましょう。

Retrieve the Titanic Training data.

Titanic Training データの取得

Load the Titanic Training data.

Titanic Trainingデータを読み込みます。

Activity1

やること1

1.Drag the Titanic Training data from the Samples repository into the Process panel.

SamplesレポジトリからProcessパネルへTitanic Trainingデータをドラッグします。

explanation

説明  
You have defined the first operation of your process.

これにより最初のオペレータをプロセスに入れることができました。

Now we will learn how easy it is to build a machine learning model on this data.

ここからは、データの機械学習モデルがいとも簡単に構築できてしまうことを見ていきましょう。

Build a decision tree.

決定木の構築

Decision Trees are a popular statistical modeling technique that finds hidden patterns in the data. Let’s build one!

#### Activity

#### 用語等解説

##### decision tree[8]

決定木。意思決定に利用される2分木のリスト構造(で書かれたグラフ)で、葉以外の各ノードに条件判断がつく。

#### 演算子の説明[9]

##### ＿＿演算子(＿＿演算子)

###### 概要

###### 出力

###### 引数

(引数名)

# 第4章 英文和訳のコツ

英文和訳のフローを、プログラム言語のように述べると次のようになる。

|  |
| --- |
| main()  {  sentence=getSentence();//文を読む  /\*  例:sentence= “Let’s build a decision tree, which is a popular statistical modeling technique that finds hidden patterns in the data.”  \*/  sentenceIdiom=convertIdiom(sentence);//イディオムを見つけ、そこだけ和訳  /\*  “a decision treeをbuildしましょう, which is a popular statistical modeling technique  that finds hidden patterns in the data.”  \*/  while(sentenseOld!=sentense)  {  sentenseOld=sentense;  sentense=parse(sentence);//文法を解析して  print(buildJapanese(sentence) + “\n---\n”);//日本語の語順にして、メモ書き。  }  /\*  ☆  \*/  print(chokuyaku=translate(buildJapanese(sentense)) + “\n---\n”);//各単語を和訳  print(iyaku=naturalize(chokuyaku));//自然にする  /\*  ☆2  \*/  }  //以下、関数定義  String function parse(sentence)  {  //引数sentenceは、文字列の場合と、配列の場合の2通りが考えられる。  if(isString(sentence))//文字列だったら  arr=parseString(sentence);  else//配列だったら  {  arr=sentence;  foreach arr as elem  elem=parseString(elem);  }  return arr;//配列を返却  }  String function parseString(sentence)  {  //こちらでは、sentenceは必ず文字列  arr=array(V=0,S=0,O=0,O2=0,C=0); //まずVを見つける  V=getV(sentence);  tmp=rtrim(sentence,V);//Vより右を無視  S=getV(tmp);  tmp=ltrim(sentence,V); //Vより左を無視  O=getO(tmp,V);  C=getC(tmp,V); //Vの隣がOかCかは、Vによって決まる  if(O!=0)  O2=getO2(tmp);  while((M=getM(tmp))!=-1)  array\_push(arr,getM(tmp));  }  String function getV(sentence)  {  ...(略)//動詞となる単語のデータベースと照合し、探す  }  ...(略) |

☆の部分について解説する。

parse関数は、入力された文字列あるいは配列を、一つ次元が上な配列に変換することで、

一段階ずつ構文解析を行っていく。その為、返り値は配列となるが、その詳細は次の通り。

array(V,S,O,O2,C,M1,M2,M3,...):

ループ1回目

whileループ1回目におけるparse関数の返り値は次の通り。

array(“build”, 0, “a decision tree”, 0, 0, “which is a popular statistical modeling technique that finds hidden patterns in the data.”, -1)  
最後の「-1」は、これ以上Mが続かないことを意味する

この配列(array)を人間用に書くと

V= “build”

S=無

O=“a decision tree”

O2=無

C=無

M1= “which is a popular statistical modeling technique that finds hidden patterns in the data.”

となる。

ループ2回目

whileループ2回目におけるparse関数の返り値は次の通り。

array(“build”, 0, “a decision tree”, 0, 0, array(“is”, “which”, 0, 0, “a technique”, “popular statistical modeling”, “that finds hidden patterns”, “in the data.”, -1), -1);

この配列を人間用に書くと

V= “build”

S=無

O=“a decision tree”

O2=無

C=無

M1=

V’ = “is”

S’ = “which”

O’ =無

O2’ =無  
C’ = “a technique”

M1’= “popular statistical modeling”

M2’= “that finds hidden patterns”

M3’= “in the data”

ループ3回目

whileループ3回目におけるparse関数の返り値は次の通り。

array(“build”, 0, “a decision tree”, 0, 0, array(“is”, “which”, 0, 0, “a technique”, “popular statistical modeling”, array(“finds”, “that” , “hidden patterns” , -1), “in the data.”, -1), -1);

V= “build”

S=無

O=“a decision tree”

O2=無

C=無

M1=

V’ = “is”

S’ = “which”

O’ =無

O2’ =無  
C’ = “a technique”

M1’= “popular statistical modeling”

M2’=

V’’ = “finds”

S’’ = “that”

O’’ = “hidden patterns”  
O2’’ =無

C’’ =無

M3’= “in the data”

したがって、whileループを抜けた直後、printによる「メモ書き」は次のようになる。

a decision treeをbuildしましょう, which is a popular statistical modeling technique that finds hidden patterns in the data.

---

a decision treeをbuildしましょう, which は popular statistical modeling なa technique ですthat finds hidden patterns、in the data.

---

a decision treeをbuildしましょう, which は popular statistical modeling なa technique をisする、that はhidden patterns をfinds する、in the data.

---

但し、「a decision treeをbuildしましょう」はsentenceIdiomから持ってくる。

☆2

printによるメモ書きは、結局全体で次のようなものになる。

a decision treeをbuildしましょう, which is a popular statistical modeling technique that finds hidden patterns in the data.

---

a decision treeをbuildしましょう, which は popular statistical modeling なa technique ですthat finds hidden patterns、in the data.

---

a decision treeをbuildしましょう, which は popular statistical modeling なa technique をisする、that はhidden patterns をfinds する、in the data.

---

決定木を構築しましょう、それは人気のある統計モデリングな方法である、それは隠れた法則を発見する、データの中で

---

決定木を構築しましょう。決定木はデータに潜む隠れた法則を見つけるためによく使われる統計モデリング手法です。

再勉強のフローをtry-catchで追記すると、次の通り。

# 参考、注釈、翻訳元原文

[1]  
<https://www.sas.com/ja_jp/insights/analytics/data-mining.html>

[2]  
https://www.rapidminer.jp/  
[3]Enter your work email とあるためフリーメールなどのアドレスではダウンロードできない可能性があると判断した。

[4] <https://docs.rapidminer.com/latest/cloud/getting-started/running-process.html>  
パネルの呼称:Tutorials1-1

[5] <https://data.wingarc.com/what-is-datascience-6119>

[6] <https://dictionary.goo.ne.jp/jn/277810/meaning/m0u/>

[7] <https://ejje.weblio.jp/content/retrieve>

[8] <https://ejje.weblio.jp/cat/computer/cmpyg>

[9]<https://docs.rapidminer.com/latest/studio/operators/rapidminer-studio-operator-reference.pdf> (翻訳元)