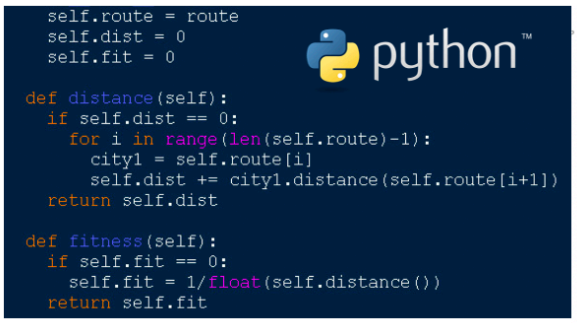
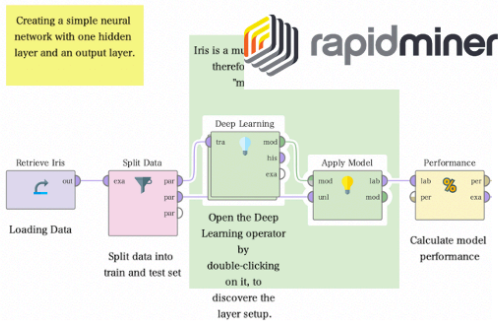
rapidminer
KSK ANALYTICS JAPAN PARTNER

機械学習モデル構築時の
Python と RapidMiner の比較





機械学習の目的は、人間が関与することなくコンピュータが自動的にデータからパターンを学習できるようにすることです。機械学習モデルは、Python や R などのプログラミング言語で作成された特定のライブラリを使って構築することができます。また、RapidMiner のような GUI（グラフィカル ユーザー インターフェース）を持つアプリケーションを使ってドラッグ＆ドロップで機械学習モデルを構築することも可能です。最近では、機械学習ライブラリを利用することで、プログラマーも機械学習モデルの構築が容易になりました。また、GUI を備えたデータサイエンスアプリケーションは、コードを書かずにデータ分析や機械学習モデルの構築を可能にしました。これらのアプリケーションは、ノンプログラマーや研究者にとって有用です。

本資料では、2 つの異なる方法を使って機械学習モデルを作成します。1 つは、プログラミング言語 Python であり、もう一つは RapidMiner アプリケーションです。

例題：糖尿病患者の予測モデル

データの準備

ハンズオンでは、ピマ・インディアンの糖尿病に関するデータセットを使用します。データは下記の URL よりダウンロードすることができます。

ダウンロード URL：https://ksk-anl.smktg.jp/public/file/document/download/1157

ピマ・インディアンは、アメリカ、アリゾナ州に住んでいる部族であり、氷河期の頃ベーリング海峡を渡りアジアから北米に移住したと言われています。人口の 50%以上が糖尿病で、有病率の高さで世界的に有名です。高い有病率の背景には、20 世紀の初めヨーロッパ系アメリカ人が彼らの生活環境を破壊したため、

多くのピマ・インディアンは保護地区で生活費の支給を受けるようになりました。その結果、欧米化した食事や運動不足により、肥満と糖尿病が急速に拡がったと言われています。

データは、21 歳以上のピマ・インディアン(女性)の糖尿病に関するデータであり、1 つの目的変数（糖尿病：糖尿病になっているかなっていないか）と 8 つの説明変数（妊娠回数、グルコース、血圧_mm HG、皮膚の厚さ、インスリン、BMI、血統（家族に糖尿病患者がいるか等を指数化）、年齢）で構成されています。

上記のデータを用いて、Deep Learning アルゴリズムを使った機械学習モデルを下記のステップで構築します。

1. データの読み込み
2. ターゲットラベルを指定
3. 交差検証・モデリング
4. モデルの評価結果の確認

Python コードを用いた予測モデル作成

Python コードを用いて機械学習モデルを構築するにあたり、本資料では、H2O ライブラリを使用します。H2O は、オープンソースで RapidMiner にも実装されているライブラリです。

1.データの読み込み

「diabetes_data.csv」を読み込みます。

```
##import library
import os
import h2o
from h2o.estimators import H2ODeepLearningEstimator
h2o.init()

##read csv
os.chdir('/Users/~~/Desktop')
df = h2o.import_file('diabetes_data.csv')
```

以下のようなテーブルが読み込めたことを確認できます。

```
>>> df.head
```

妊娠回数	グルコース	血圧_mm HG	皮膚の厚さ	インスリン	BMI	血統	年齢	糖尿病
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1

2.ターゲットラベルの指定

「diabetes_data.csv」のターゲットラベルは、“糖尿病”列です。ここで目的変数と説明変数、データ型（数値、文字型）を指定します。

```
#目的変数、説明変数の定義
response = "糖尿病"
predictors = df.names[0:8]

#型の定義
df["糖尿病"] = df["糖尿病"].asfactor()
```

3.交差検証・モデリング

5 分割の交差検証を実施します。今回の予測モデルが 2 値分類なので、distribution は、bernoulli を選択します。5 分割の交差検証を実施しますので、nfolds は 5 と入力します。その他のオプション（パラメーター）は下記で確認できます。

【Deep Learning パラメーター】

https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/deep-learning.html

```
#Build and train the model
dl = H2ODeepLearningEstimator(distribution="bernoulli",nfolds=5)

dl.train(x=predictors, y=response, training_frame=df)
```

4.モデルの評価結果の確認

3.で作成したモデルの評価結果（Performance）を確認します。

```
#モデルパフォーマンスの確認
perf = dl.model_performance()

#予測値の追加
pred = dl.predict(df)
```

結果として、下記のような混同行列が得られます。

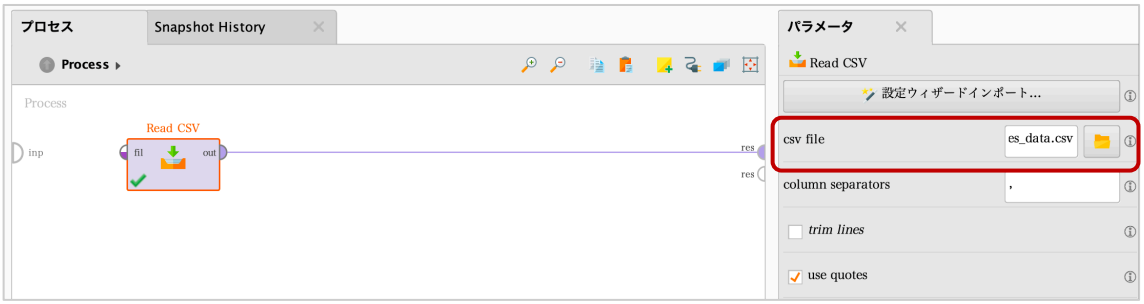
```
Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.20594693019844096:
       0    1    Error    Rate
-----  ---  ---  -------  -------------
0      359  141  0.282    (141.0/500.0)
1      59   435  0.1194   (59.0/494.0)
Total  418  576  0.2012   (200.0/994.0)
```

RapidMiner を用いた予測モデル作成

RapidMiner を起動し、早速、機械学習モデルの作成を進めてみましょう。

1.データの読み込み

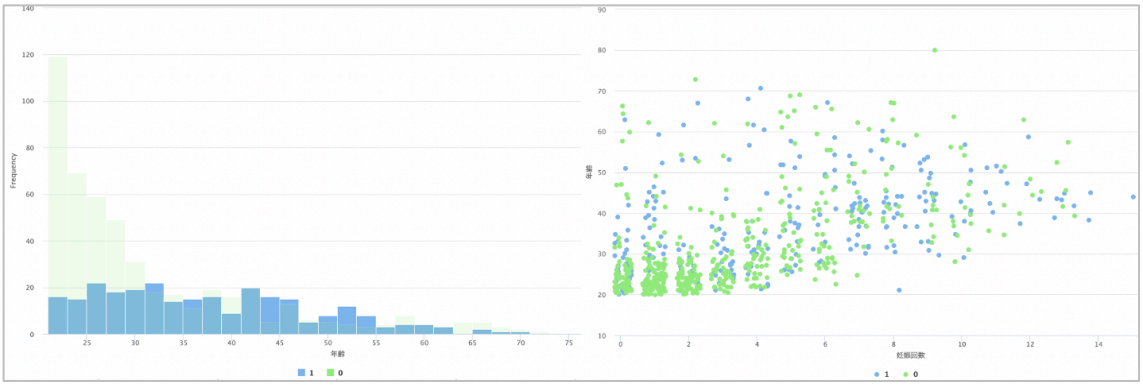
「diabetes_data.csv」ファイルを読み込みます。Read CSV オペレーターをパネル上に配置し、画面右のパラメータで csv ファイルの場所を指定します。



実行ボタン を押下すると以下のようなテーブルが読み込めたことを確認できます。また、左にある基本統計量や Visualization（可視化）を押すとデータの分布や傾向を捉えることができ、データセットの理解が進みます。下記のようなヒストグラムや散布図だけでなく、線グラフやボックスプロット、サンキーチャートなど 36 種類もの可視化を行うことができます。

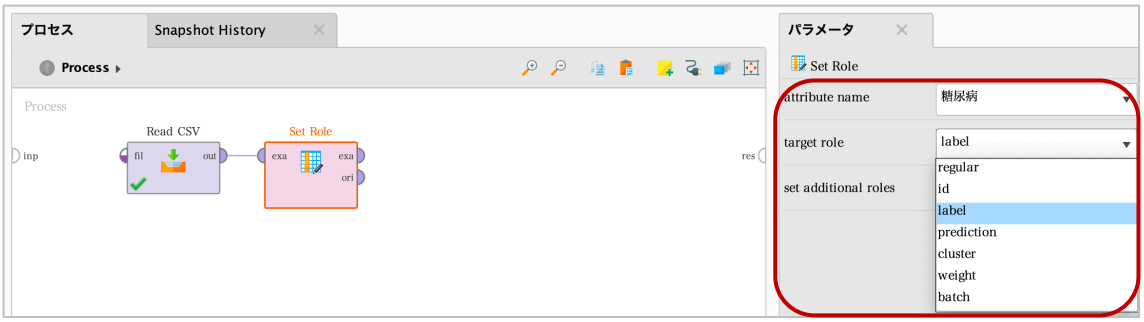
Row No.	妊娠回数	グルコース	血圧_mm HG	皮膚の厚さ	インスリン	BMI	血統	年齢	糖尿病
1	6	148	72	35	0	33.600	0.627	50	1
2	1	85	66	29	0	26.600	0.351	31	0
3	8	183	64	0	0	23.300	0.672	32	1
4	1	89	66	23	94	28.100	0.167	21	0
5	0	137	40	35	168	43.100	2.288	33	1
6	5	116	74	0	0	25.600	0.201	30	0
7	3	78	50	32	88	31	0.248	26	1

▶可視化例



2.ターゲットラベルの指定

「diabetes_data.csv」のターゲットラベルは、“糖尿病”列です。糖尿病であるかどうかは“1”、“”0“で表現されているため、読み込んだ時点で”binominal”型になっています。Set Role オペレーターを用いて、“糖尿病”列がターゲット列（目的変数）であることを指示します。

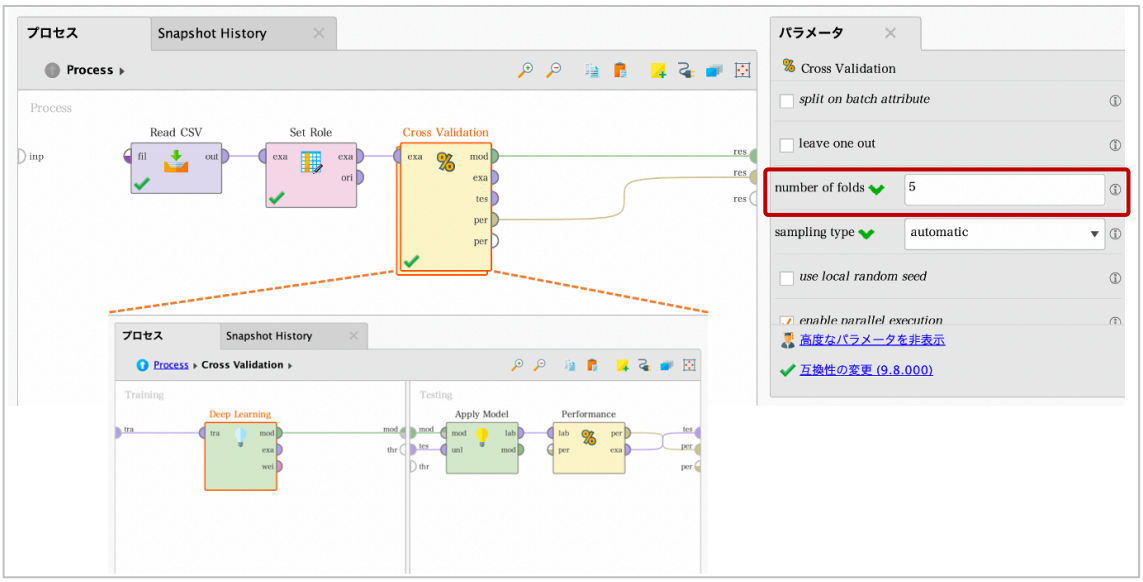


3.交差検証・モデリング

5 分割の交差検証を実施します。5 分割の交差検証を実施しますので、number of folds は 5 と入力します。Cross Validation オペレーターをダブルクリックし、Training パネルに Deep Learning を配置し、Testing パネルに Apply Model と Performance オペレーターを配置し、実行します。

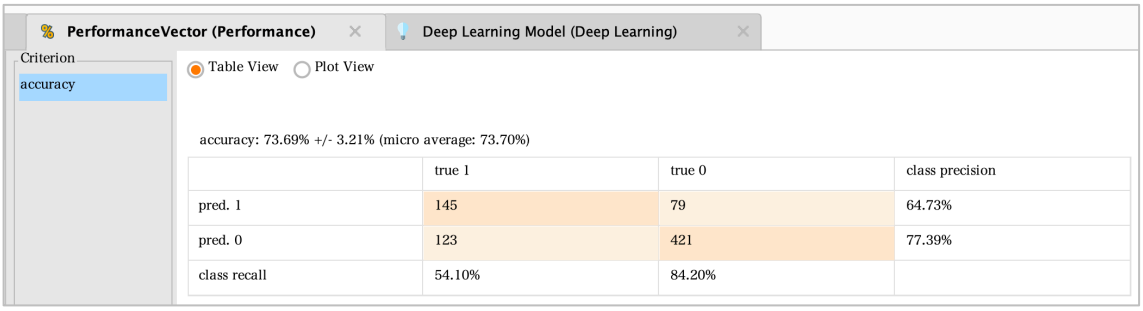
具体的には、左の Training パネルでは、5 分割されたデータの内、4 分割のデータを使って学習を行い、右の Testing パネルでは残りの 1 分割のデータを使って答え合わせを実施しています。それを被らないように合計 5 回実施し、全体のパフォーマンスとしています。

今回は、Deep Learning オペレータを使用していますが、機械学習アルゴリズム（オペレーター）を変更すれば、他のアルゴリズムとの精度比較を行うこともできます。



4.モデルの評価結果の確認

3.で作成したプロセスを実行すると下記のような混同行列が得られます。



	true 1	true 0	class precision
pred. 1	145	79	64.73%
pred. 0	123	421	77.39%
class recall	54.10%	84.20%	

まとめ

2 つの異なる方法を使って機械学習モデルの構築を行いました。まず、プログラミング言語である Python を用いた方法ですが、ソースコードを記述することに抵抗のないユーザーばかりの部署（チーム）であれば、問題なく進めることができると思います。ただ、そうでない場合、Python 環境の構築、ライブラリのインストールエラーで進まないことも想定されます。また、データの可視化を行いながら反復的に機械学習モデルを作成する際に、複数のライブラリを組み合わせながら可視化、分析を進めていくことも面倒です。モデルの結果についてもコマンドプロンプトにテキストが表示されるだけであり、モデルの解釈も進めづらいと感じる場合もあるでしょう。

一方、RapidMiner を用いた方法であれば、いくつかのオペレーター（今回は 6 つ）を組み合わせながらドラッグ&ドロップで配置するだけで簡単に機械学習モデルが作成できました。直感的に何をやっているかも理解しやすく、機械学習に取り組んだことのないチームメンバーも巻き込める可能性がありそうです。トップデータサイエンティストでない研究者や一般ユーザーであれば、まず RapidMiner のような GUI アプリケーションでモデル構築してみる方がいいかもしれません。