■概要

IBM Power Systems 汎用モデル（S914：4core）の None GPU モデルで「IBM i の横に Driverless AI」と題して、H2O Driverless AI(以下 Driverless AI)の導入および稼働検証を実施しました。

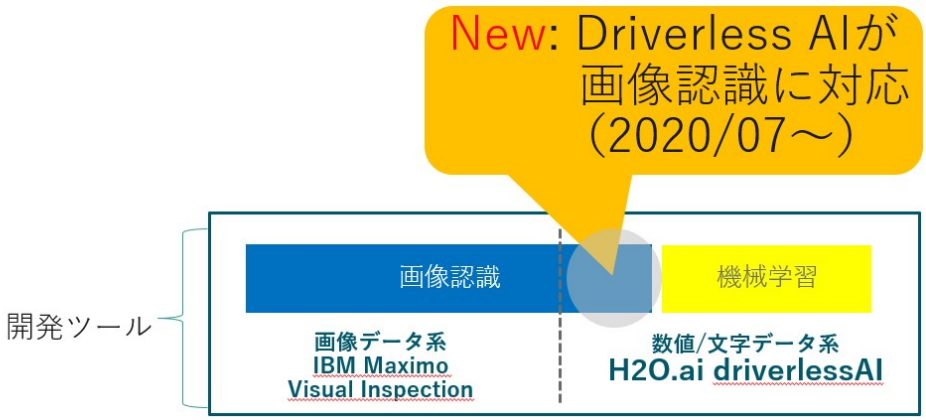
■背景・課題

IBM Power Systems 汎用モデル（None GPU）でも以前から Driverless AI を使用して、数値データや文字データから洞察（需要予測、不正検知など）をすることができましたが、それに加えて画像認識による画像分類ができるようになりました。(2020 年 7 月に発表)
最初に IBM Power Systems 汎用モデルの S914 に IBM i ホストークライアント区画があり、追加で Linux クライアント区画(RHEL7.6)環境を作成して Driverless AI を導入するのか？
続いて、Driverless AI 導入後、どのように画像認識（画像分類）を行うのか？
上記について実際に検証確認いたしました。

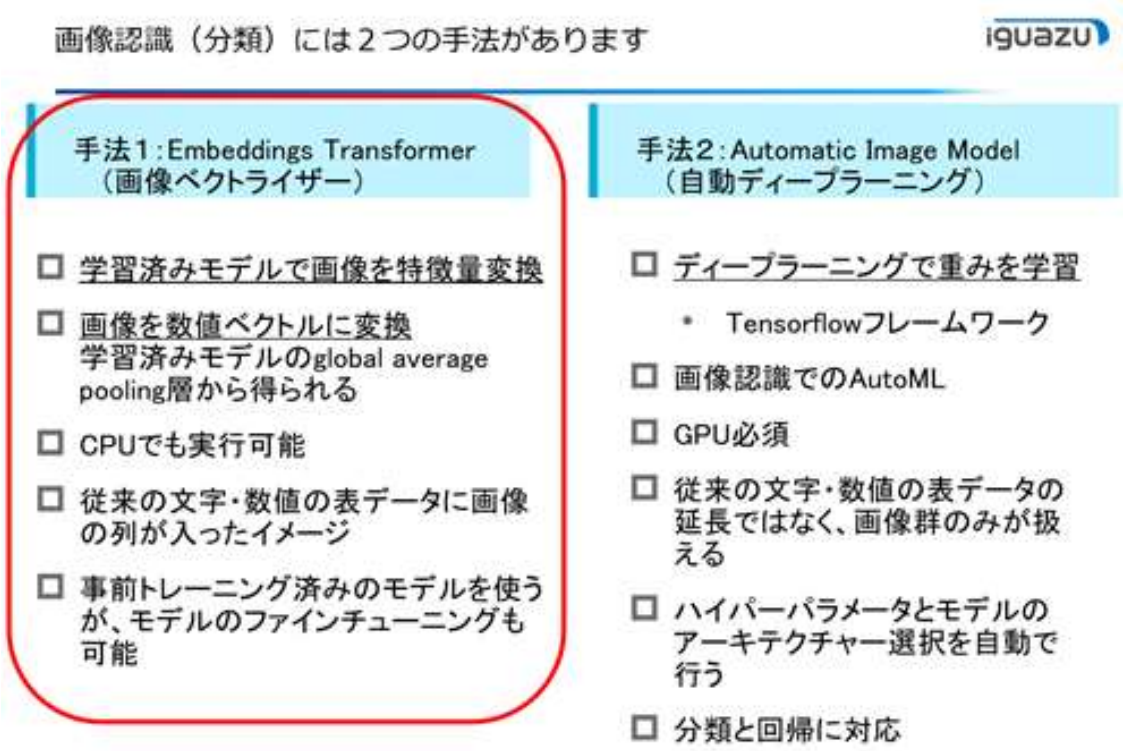
■製品概要

・Driverless AI

数値データ・文字データから需要予測や不正検知のモデルを全自動で高精度に導出し、現場でポータブルに予測や検知を実行できます。



※Driverless AI の画像認識（画像分類）について
IBM Power Systems 汎用モデルで利用する場合、「画像ベクトライザー」の手法です。



メリットとしては、GPU 非搭載でも利用可能なところです。

画像認識（分類）の２つの手法比較

iguazu

	Embeddings Transformer（画像ベクトライザー）	Automatic Image Model（自動ディープラーニング）
概要	事前学習済みモデルの重みで画像を数値ベクトル群に変換	画像認識モデルをディープラーニングで作成
サーバー環境	GPU不要（CPU環境でも可能）	GPU必要
ジョブの所要時間	15～30分程度	30分～数10時間
認識精度	中	高
推論パイプライン	Python, MOJO ※	Python

※数値/文字における機械学習モデルと同様の方法にて、画像の推論もMOJOパイプラインを使って行える。
- 現在はEmbeddings Transformerのみ。
- Automatic Image ModelはMOJOサポートなし。Pythonパイプラインは対応済

■構成イメージ

- Driverless AI 環境
  使用した IBM Power Systems の仕様は以下の通り
  モデル：S914(9009-41A：4core)、CPU：1core、MEM：12GB、
  OS：RHEL7.6（IBM i ホスト－Linux クライアント）環境
  Driverless AI：V1.9.0.6
  画像認証で使ったサンプルデータ：
  ・bird.zip（モデルデータとして、鳩、カラス、カモメなど 16 種 323 枚の鳥画像）
  ・TEST_Bird.zip（予測用データとして、鳩、カモメの 2 種 4 枚の鳥画像）

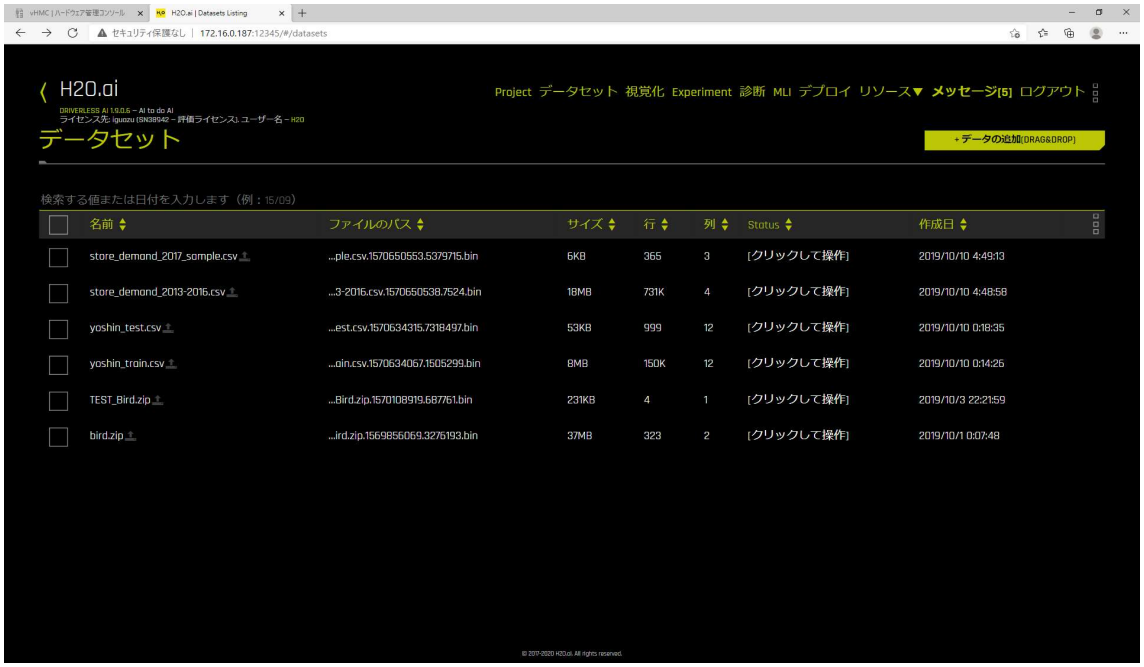
■検証内容

- Driverless AI の導入検証（Linux 環境の認証完了してることが前提です）
  S914 の IBM i 区画の横に Linux 区画を作成して認証処理まで完了します。
  その Linux 環境に Driverless AI の導入します。以下の URL を参照して実施
  https://dai-doc-jp.au-syd.mybluemix.net/install/ibm-docker.html#install-on-ibm-with-cpusl
  ① yum install コマンドより、Docker を導入
  ② systemctl start コマンドより、Docker を開始
  ③ systemctl enable コマンドより、再起動しても自動で Docker が起動済にできる設定
  ④ docker load コマンドより、Driverless AI を導入
  ⑤ docker run コマンドより、Driverless AI の起動
  ⑥ http://IP アドレスまたはホスト名:12345/にアクセスしてログイン画面が表示確認
- Driverless AI の画像認証（画像分類）検証
  ① Driverless AI にログイン
  ② 「データセット」の「データの追加」-「ファイルをアップロードする」より、bird.zip のモデルデータと、TEST_Bird.zip の予測用データを取り込む
  ③ 取込み後の確認を bird.zip をクリックして「詳細」-「生データ」にて画像を確認
  ④ 確認後、再度 bird.zip をクリックして「EXPERIMENT」をを選択し、「ターゲット列を選択」で“label”選択して「EXPERIMENT の開始」を押下
  ⑤ 完了後、上部メニューから「Experiment」を選択し、作成された bird.zip をクリックしてモデルデータ分析結果が表示する。
  ⑥ 「他のデータセットで予測」をクリックし、「TEST_Bird.zip」を選択して「完了」ボタンを押下することで画像分類予測を実施する。
  ⑦ 分析が完了して「開く」を押下し、「予測結果のダウンロード」をクリックすると CSV ファイルに出力され、ベクトル数値の割合で分類されます。

■検証結果

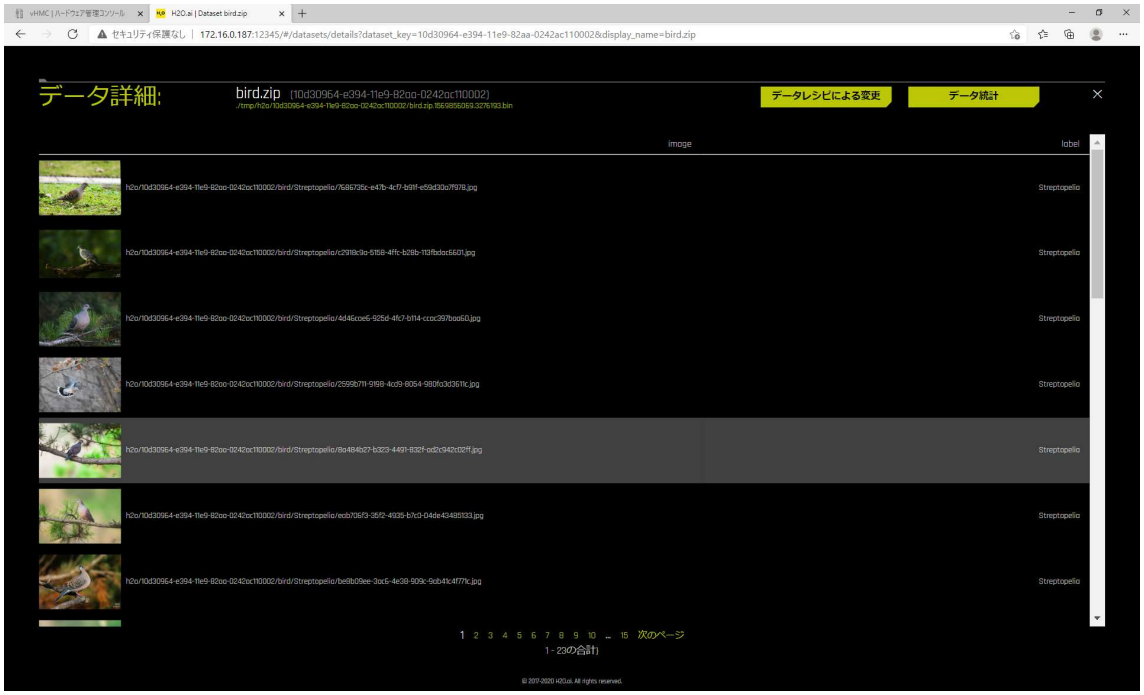
・DriverlessAI の導入に関しては、「検証内容」の記載の通りで問題無く導入ができました。

・データセット画面

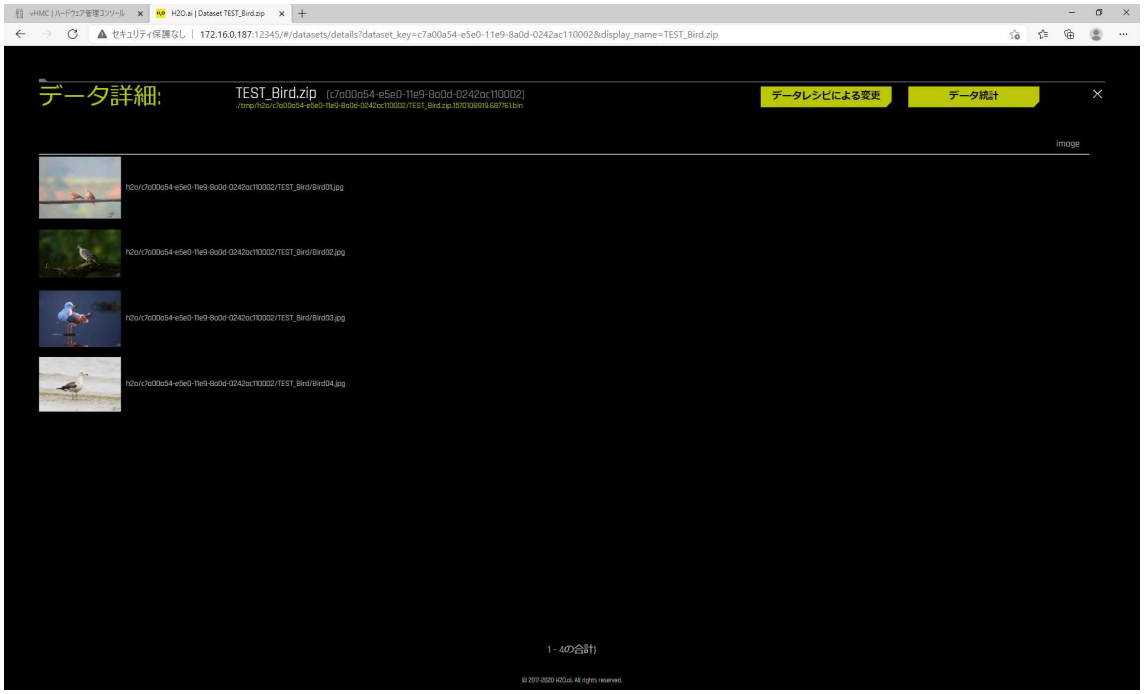


・画像データ確認画面

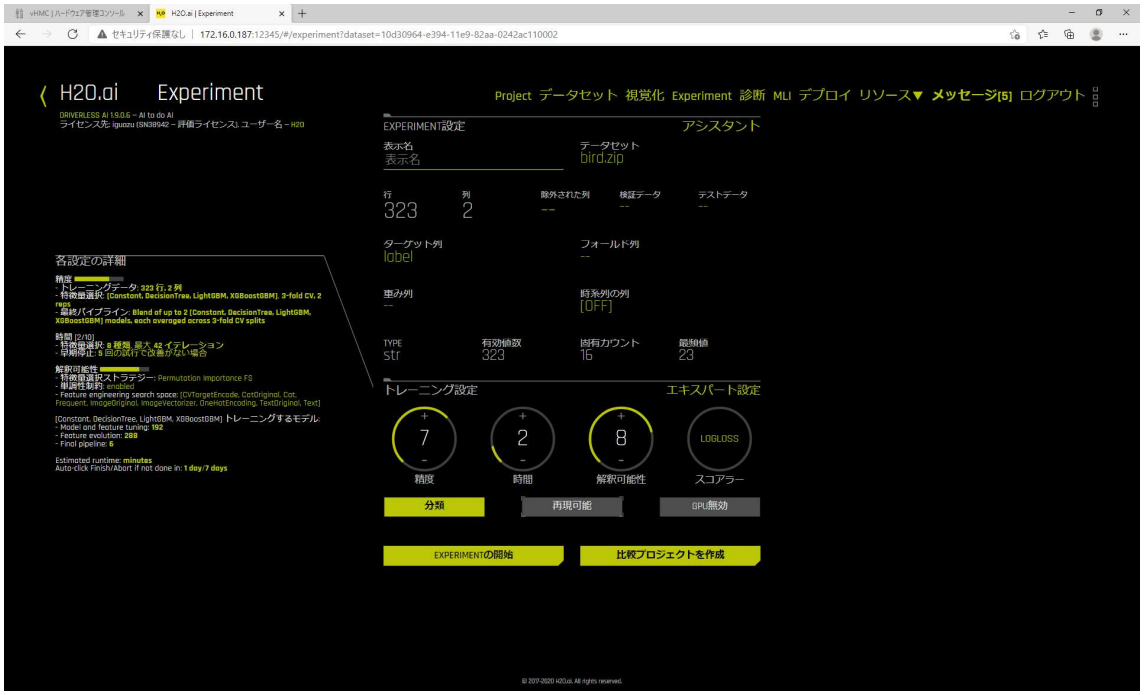
bird.zip（モデルデータサンプル）



TEST_Bird.zip（予測用データサンプル）
上 2 つは鳩、下 2 つはカモメです。

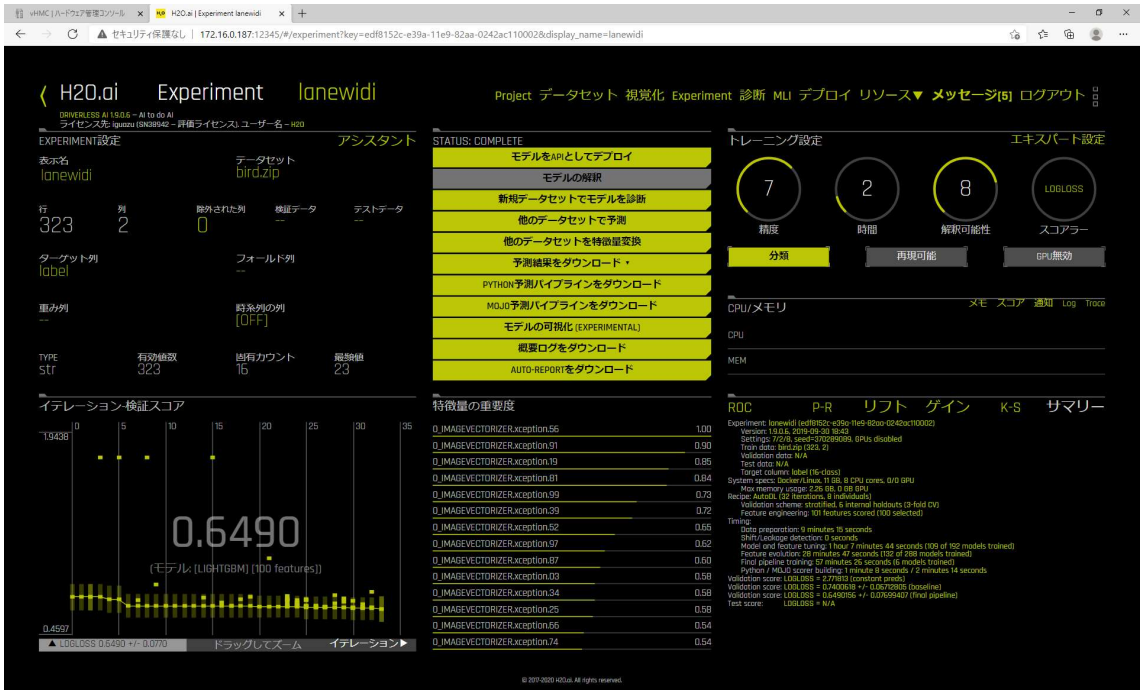


・EXPERIMENT 指定画面（データセット名：bird.zip、ターゲット列名：label）



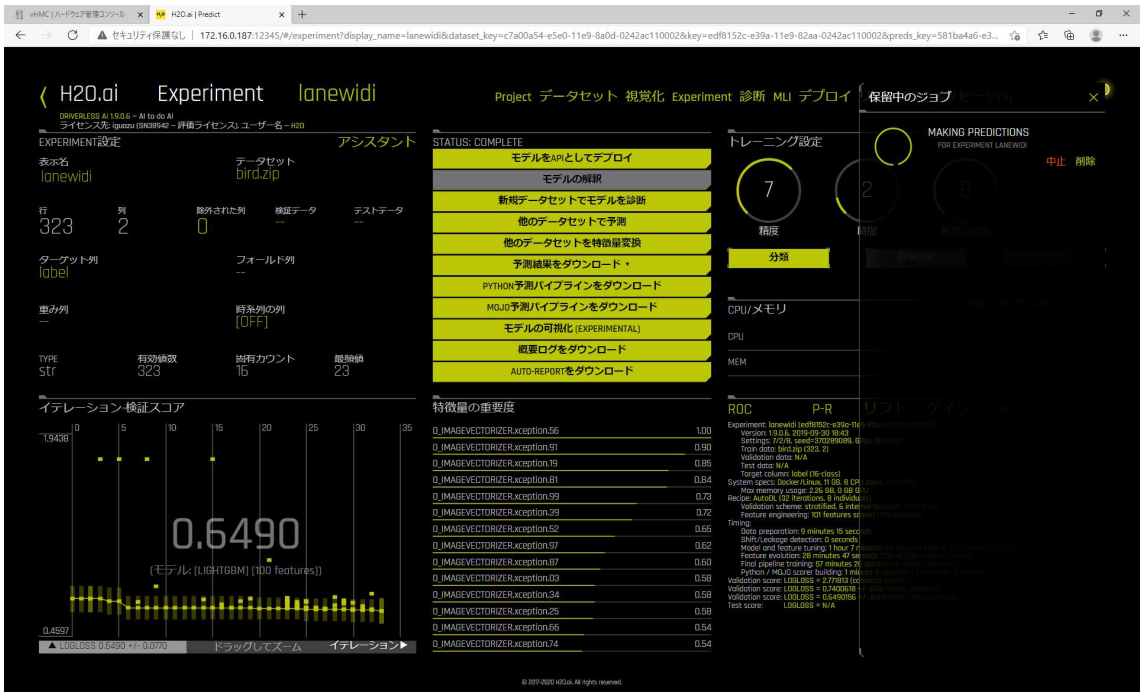
・EXPERIMENT 処理がモデルデータ完了後の画面

※因みに完了までに 2:46:45（約 3 時間）で処理が完了しました。



・「他のデータセットで予測」処理より TEST_Bird.zip で分析中（画面右横）

※約 1 分で結果が出ました



・処理後、CSV ファイルの内容（ベクトル数値から確率（9 割以上）で鳥の種類を分類）

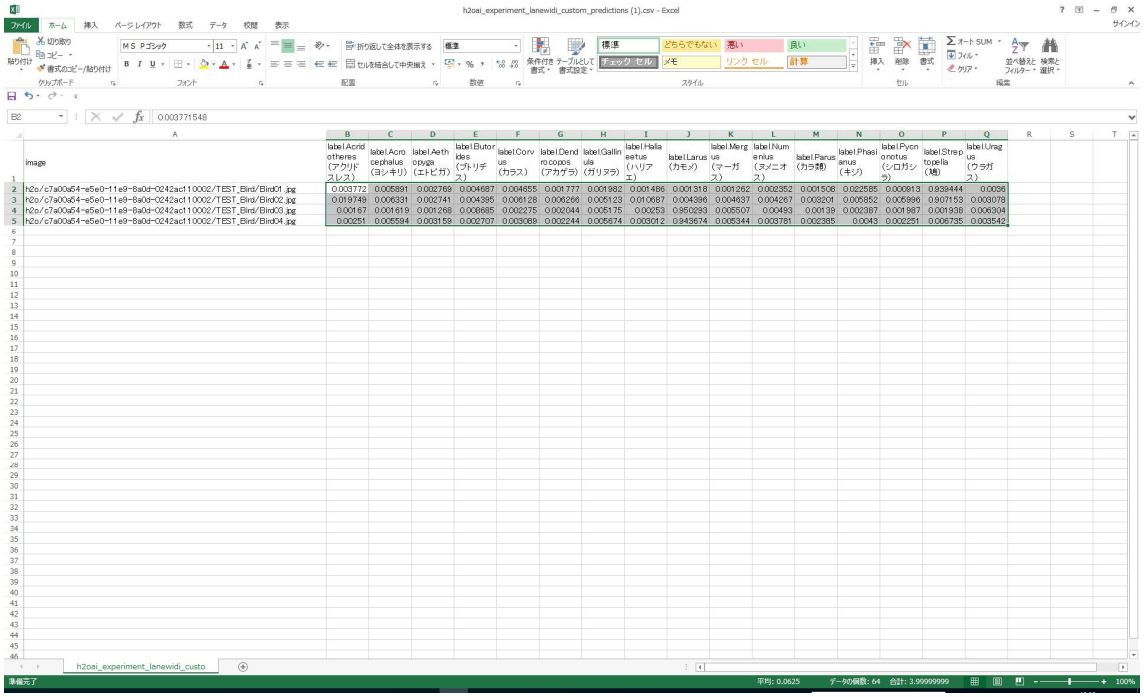


	image	label:Acridotheres (アクリドテレス)	label:Acrocephalus (ヨシキリ)	label:Aethopyga (エトピガ)	label:Butorides (ブトリデス)	label:Corvus (カラス)	label:Dendrocopos (アカゲラ)	label:Gallinula (ガリヌラ)	label:Haliaeetus (ハリアエ)	label:Larus (カモメ)	label:Merganser (マーガンス)	label:Numenius (ヌメニオス)	label:Parus (カラ類)	label:Phasianus (キジ)	label:Pycnonotus (シロガシラ)	label:Streptopelia (鳩)	label:Uraginus (ウラガス)
2	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird01.jpg	0.003772	0.005891	0.002769	0.004687	0.004655	0.001777	0.001982	0.001486	0.001318	0.001262	0.002352	0.001506	0.022585	0.000813	0.939444	0.0036
3	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird02.jpg	0.019749	0.006331	0.002741	0.004395	0.006128	0.006266	0.005123	0.010687	0.004396	0.004637	0.004267	0.003201	0.005852	0.005996	0.907153	0.003078
4	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird03.jpg	0.00167	0.001619	0.001268	0.006685	0.002275	0.002044	0.005175	0.00253	0.950293	0.005507	0.00493	0.00139	0.002387	0.001987	0.001938	0.006304
5	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird04.jpg	0.00251	0.005594	0.003159	0.002707	0.003069	0.002244	0.005674	0.003012	0.943674	0.005344	0.003781	0.002385	0.0043	0.002251	0.006735	0.003542

※割合設定を 9 割にして、上 2 つが「鳩」、下 2 つが「カモメ」という結果で分類。

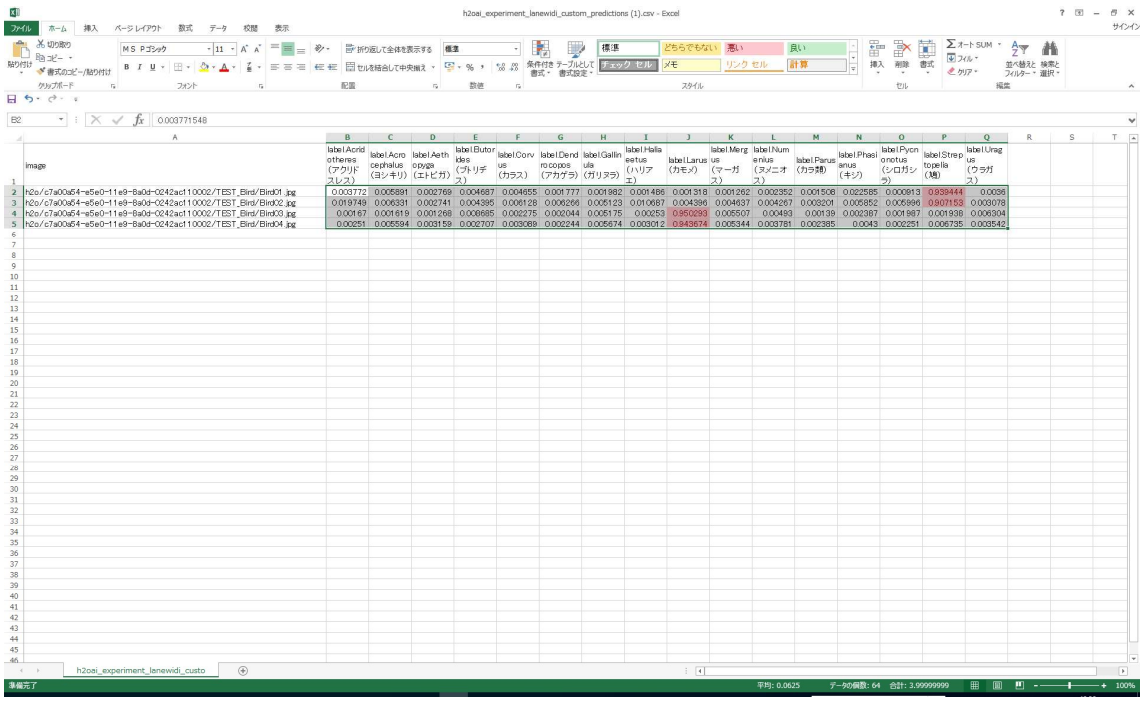


	image	label:Acridotheres (アクリドテレス)	label:Acrocephalus (ヨシキリ)	label:Aethopyga (エトピガ)	label:Butorides (ブトリデス)	label:Corvus (カラス)	label:Dendrocopos (アカゲラ)	label:Gallinula (ガリヌラ)	label:Haliaeetus (ハリアエ)	label:Larus (カモメ)	label:Merganser (マーガンス)	label:Numenius (ヌメニオス)	label:Parus (カラ類)	label:Phasianus (キジ)	label:Pycnonotus (シロガシラ)	label:Streptopelia (鳩)	label:Uraginus (ウラガス)
2	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird01.jpg	0.003772	0.005891	0.002769	0.004687	0.004655	0.001777	0.001982	0.001486	0.001318	0.001262	0.002352	0.001506	0.022585	0.000813	0.939444	0.0036
3	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird02.jpg	0.019749	0.006331	0.002741	0.004395	0.006128	0.006266	0.005123	0.010687	0.004396	0.004637	0.004267	0.003201	0.005852	0.005996	0.907153	0.003078
4	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird03.jpg	0.00167	0.001619	0.001268	0.006685	0.002275	0.002044	0.005175	0.00253	0.950293	0.005507	0.00493	0.00139	0.002387	0.001987	0.001938	0.006304
5	h2o/c7a00a54-e5e0-11e9-8a0d-0242ac110002/TEST_Bird/Bird04.jpg	0.00251	0.005594	0.003159	0.002707	0.003069	0.002244	0.005674	0.003012	0.943674	0.005344	0.003781	0.002385	0.0043	0.002251	0.006735	0.003542

実際はNoneGPU環境では処理上、以下のような画面上に数値表現する機能はありません。
※以下は画像と数値をマージ加工したものです

■所感

IBM Power Systems 汎用モデル（S914:None GPU）で検証しましたがモデルデータが出来た後の予測分析もストレスなく分析結果が得られました。
S914 の 4core モデルは、最初から 4core 分がアクティベーションされています。
IBM i が 1core で利用している場合、3core 分余ったリソースで Linux 区画を構築して Driverless AI を利用することで需要予測や不正検知や画像分類といった付加価値が付けられるのではないでしょうか？
是非、IBM i の横から AI を始めてみませんか？