



機械学習プラットフォーム・ サービスの選び方

自己紹介

- 田上 健太
- @haizine_regonn
- 2018年島根県にIターン
- フリーランスとして活動
- 機械学習コンテンツの
ポッドキャスト配信
- VR技術やVTuber活動



機械学習(データ分析)PaaS



Amazon SageMaker



ABEJA
PLATFORM



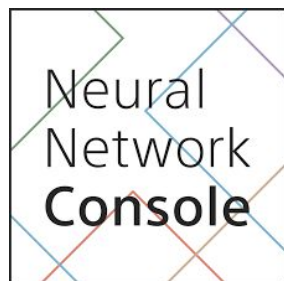
data
iku



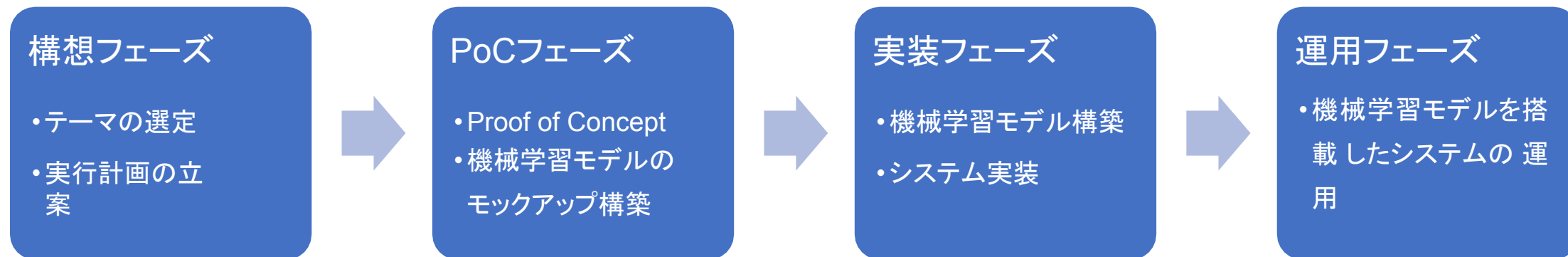
Watson Machine Learning



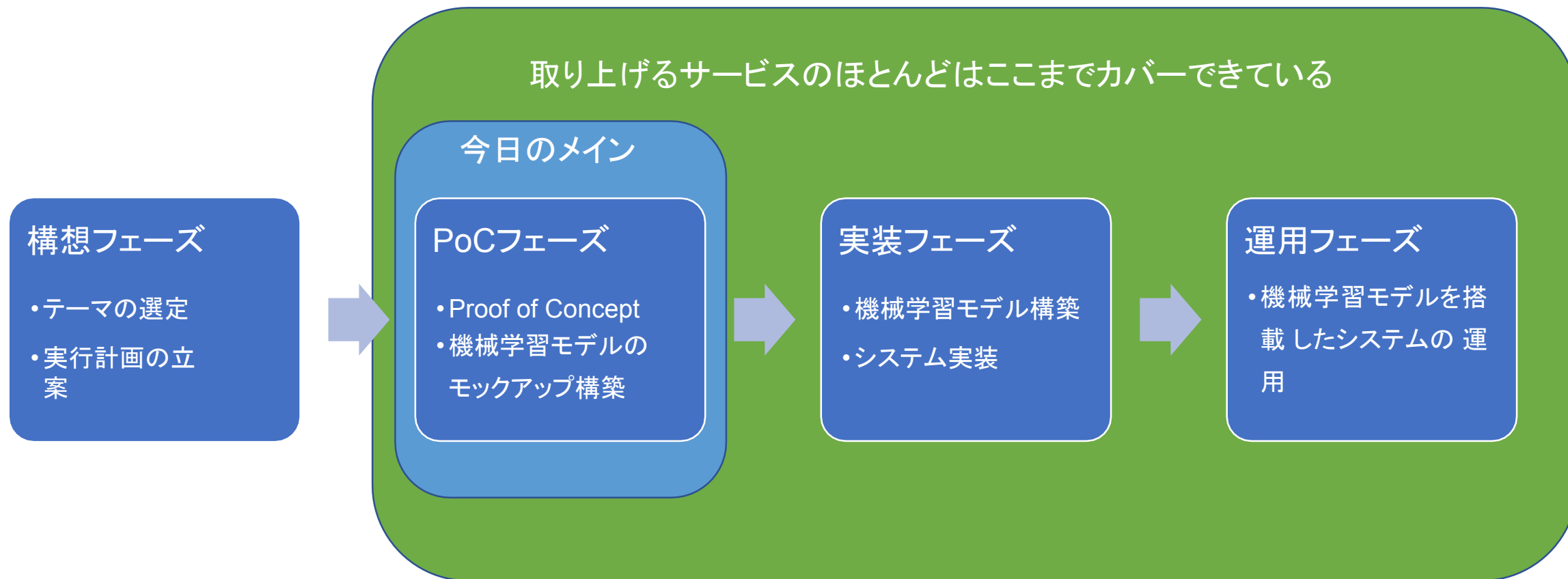
DataRobot



機械学習プロジェクトの全体像



機械学習プロジェクトの全体像



本日あまり話さないこと

- TensorFlow、PyTorch等のフレームワーク紹介
- それぞれのプラットフォームの詳しい使い方等
 - 一部サービスでのデモンストレーションあり

今回の調査
対象
(2019年3
月
調査時)

Amazon/AWS

Google/GCP

IBM/Watson

Microsoft/Azure

Sony

Uber

調査以降の動向

- Microsoft
 - ドラッグ&ドロップの 機械学習ツールを発売

日本発の分析サービスも色々出てきた

MatrixFlow

nehan

最近の機械 学 習 プラッ トフォーム の傾向

GUIでの機械学習

- フローを組み立てればよい
- 非エンジニア向けというわけではなく、ある程度知識のあるエンジニアが素早くプロトタイプを作れる

データを渡すだけで自動分析

- ある程度の精度を出すだけだったら十分
- AIの民主化

API利用等の本番運用もプラットフォームでできる

- データさえあれば、動くものを作ることができる
- パラメータ調整等もプラットフォーム上で完結する

サービス公開に向けて

すでにモデル構築済み

GCP/CLOUD MACHINE
LEARNING ENGINE

AWS/Amazon SageMaker

Jupyter Notebookで構築

Microsoft/Azure Machine
Learning Service

(AWS/AmazonSageMaker)

コードを書かないで構築

ブロック組み立て型

- データ処理フロー構築
- ニューラルネットワーク層を構築

データ登録型

- 画像系
- テーブルデータ系

サービス公開に向けて

すでにモデル構築済み

GCP/CLOUD MACHINE
LEARNING ENGINE

AWS/Amazon SageMaker

Jupyter Notebookで構築

Microsoft/Azure Machine
Learning Service
(AWS/Amazon SageMaker)

コードを書かないで構築

ブロック組み立て型

- データ処理フロー構築
- ニューラルネットワーク層を構築

データ登録型

- 画像系
- テーブルデータ系

すでにモデルが構築済み

- GCP/CLOUD MACHINE LEARNING ENGINE

- TensorFlow/scikit-learn/XGBoost等のライブラリで構築済みであれば、TPUやGPUを利用した高速なモデルトレーニング、評価、デプロイができるようになっている
- ローカル環境でモデルを構築して、GCP上でジョブとして実行する
- Docker, TensorFlow関連が強い

```
[10:45:42] tensorflow.python.trainer.trainer:11:17: tree pruning end, 1 n
outs, 44 extra nodes, 0 pruned nodes, max_depth=6
[10:45:43] tensorflow.python.trainer.trainer:11:17: tree pruning end, 1 n
outs, 32 extra nodes, 0 pruned nodes, max_depth=6
[10:45:43] tensorflow.python.trainer.trainer:11:17: tree pruning end, 1 n
outs, 42 extra nodes, 0 pruned nodes, max_depth=6
[10:45:43] tensorflow.python.trainer.trainer:11:17: tree pruning end, 1 n
outs, 34 extra nodes, 0 pruned nodes, max_depth=6
[10:45:43] tensorflow.python.trainer.trainer:11:17: tree pruning end, 1 n
outs, 44 extra nodes, 0 pruned nodes, max_depth=6

モデルをファイルとして出力する
In [9]: model_filename = 'model.h5'
       bst.save_model(model_filename)

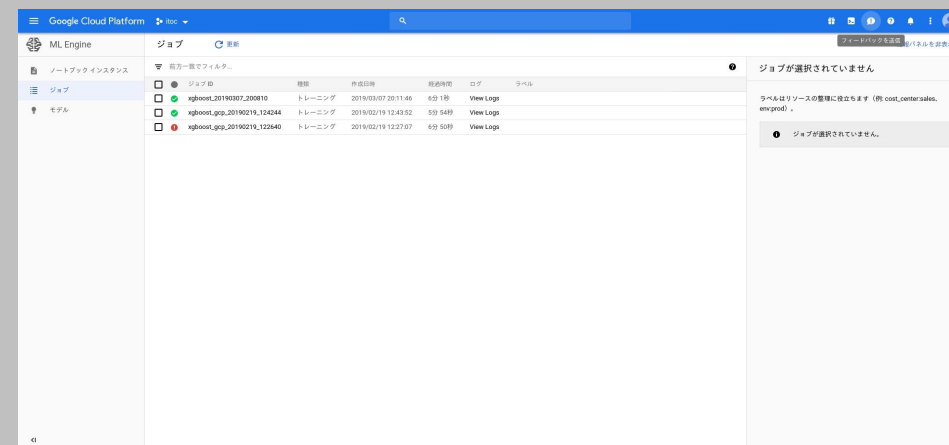
モデルファイルをGoogle Cloud Storageに
In [10]: gcs_model_path = os.path.join('gs://', BUCKET_NAME,
       datacite.datetime.now().strftime('%Y%m%d_%H%M%S'), model_filename)
       subprocess.check_call(['gsutil', 'cp', model_filename, gcs_model_path])
Out[10]: 0

モデル作成
上記のコードをtrain.py というファイルに保存し、次のようなディレクトリ構造
を構築する。
  xgboost_trainer/
  __init__.py (空ファイル)
  train.py

次の準備作業を並行して次のコマンドをコンソール上で実行することでローカルでモデルの
学習も実行できる。
  TRAINING_PACKAGE_PATH="/xgboost_trainer/"
  MODEL_TRAINER_MODULE="xgboost_trainer.train"

> gcloud ml-engine local train --package-path $TRAINING_PACKAGE_PATH --
  module-name $MODEL_TRAINER_MODULE

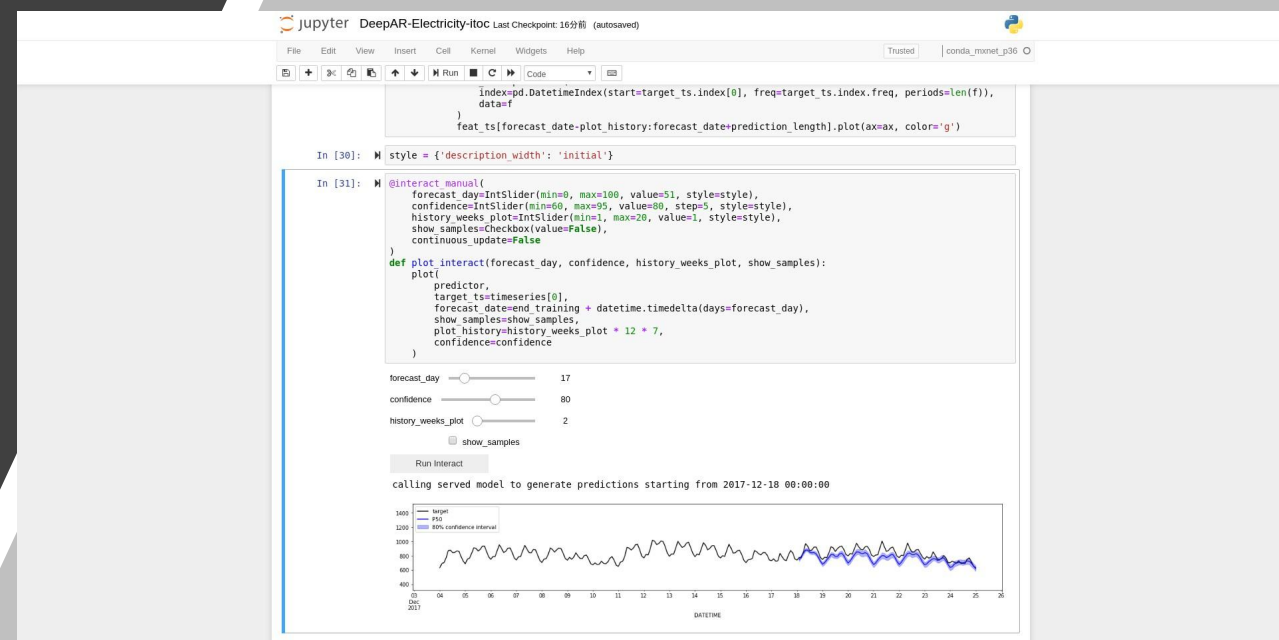
今後本文の準備作業を簡便化する。
PROJECT_ID=[BUCKET-00]
BUCKET_ID=[BUCKET-00]
JOB_NAME="xgboost_ml_data_$(date +%Y%m%d_%H%M%S)"
JOB_PATH=$(BUCKET_ID)/xgboost_ml_$(
  TRAINING_PACKAGE_PATH="/xgboost_trainer/"
  MODEL_TRAINER_MODULE="xgboost_trainer.train"
  RESOURCES=central1
  BOOSTING_VERSION=1.2
  PYTHON_VERSION=3.5
```



ジョブ	ジョブ ID	名前	ステータス	作成日時	終了日時	経過時間	ログ	ラベル
☐	gboost_20190307_200810	トレーニング	完了	2019/03/07 20:11:46	6分 1秒		View Logs	
☐	gboost_gcp_20190219_124244	トレーニング	完了	2019/02/19 12:43:52	5分 54秒		View Logs	
☐	gboost_gcp_20190219_112840	トレーニング	完了	2019/02/19 12:27:57	6分 50秒		View Logs	

すでにモデルが構築済み

- AWS/Amazon SageMaker
 - Tensorflow/Chainer/PyTorch/MXN et といったフレームワークの SageMaker 用Wrapperが用意されており、それに合わせてコードを変更すれば、SageMaker上でトレーニングを行い、デプロイができる
 - S3上にワークスペースを作成して、AWSのインスタンス上でJupyter Notebookを動かす
 - ONNX(オニキス)という共通のディープラーニングモデルフォーマットに強い



サービス公開に向けて

すでにモデル構築済み

GCP/CLOUD MACHINE
LEARNING ENGINE

AWS/Amazon SageMaker

Jupyter Notebookで構築

Microsoft/Azure Machine
Learning Service

(AWS/AmazonSageMaker)

コードを書かないで構築

ブロック組み立て型

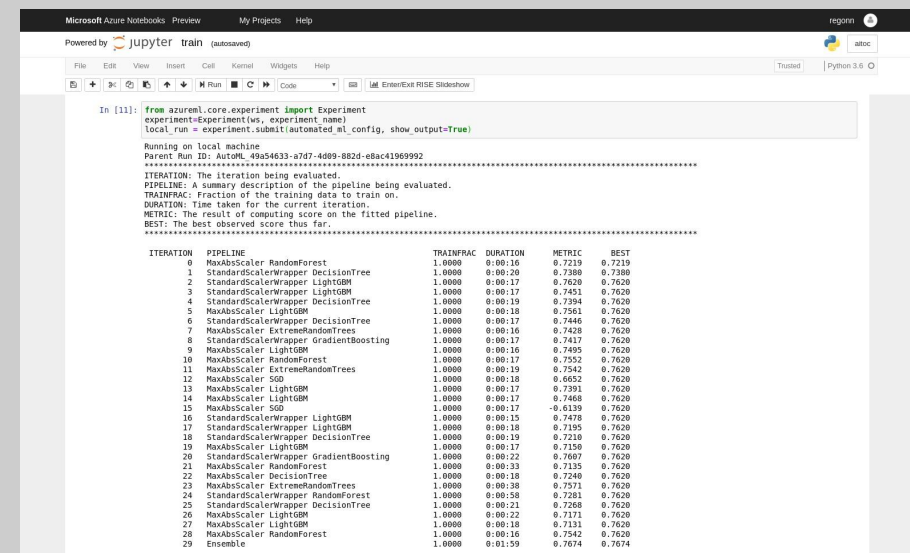
- データ処理フロー構築
- ニューラルネットワーク層を構築

データ登録型

- 画像系
- テーブルデータ系

Jupyter Notebook上でモデル構築をしたい

- Azure Machine Learning Service
 - 独自のライブラリでデータ処理を行うためなれが 必要 だけど、学習部分は色々な分類機を用いて学習を行ってくれる
- Azure上でJupyter Notebookを動かす
- Xboxサービスを展開しているためIoTのノウハウが強い



```
In [11]: from azureml.core.experiment import Experiment
experiment=Experiment(ws, experiment_name)
local_run = experiment.submit(automated_ml_config, show_output=True)
```

Running on local machine
Parent Run ID: AutoML_69a54633-a7d7-4d89-882d-e8ac41969992

ITERATION: The iteration being evaluated.
PIPELINE: A summary description of the pipeline being evaluated.
TRAINFRAC: Fraction of the training data to train on.
DURATION: Time taken for the current iteration.
METRIC: The result of computing score on the fitted pipeline.
BEST: The best observed score thus far.

ITERATION	PIPELINE	TRAINFRAC	DURATION	METRIC	BEST
0	MaxAbsScaler RandomForest	1.0000	0:00:16	0.7219	0.7219
1	StandardScalerWrapper DecisionTree	1.0000	0:00:20	0.7380	0.7380
2	StandardScalerWrapper LightGBM	1.0000	0:00:17	0.7620	0.7620
3	StandardScalerWrapper LightGBM	1.0000	0:00:17	0.7451	0.7620
4	StandardScalerWrapper DecisionTree	1.0000	0:00:19	0.7394	0.7620
5	MaxAbsScaler LightGBM	1.0000	0:00:18	0.7561	0.7620
6	StandardScalerWrapper DecisionTree	1.0000	0:00:17	0.7446	0.7620
7	MaxAbsScaler ExtremeRandomTrees	1.0000	0:00:16	0.7428	0.7620
8	StandardScalerWrapper GradientBoosting	1.0000	0:00:17	0.7417	0.7620
9	MaxAbsScaler LightGBM	1.0000	0:00:16	0.7405	0.7620
10	MaxAbsScaler RandomForest	1.0000	0:00:17	0.7552	0.7620
11	MaxAbsScaler ExtremeRandomTrees	1.0000	0:00:19	0.7542	0.7620
12	MaxAbsScaler SGD	1.0000	0:00:18	0.7155	0.7620
13	MaxAbsScaler LightGBM	1.0000	0:00:17	0.7391	0.7620
14	MaxAbsScaler LightGBM	1.0000	0:00:17	0.7466	0.7620
15	MaxAbsScaler SGD	1.0000	0:00:17	-0.6139	0.7620
16	StandardScalerWrapper LightGBM	1.0000	0:00:15	0.7478	0.7620
17	StandardScalerWrapper LightGBM	1.0000	0:00:18	0.7155	0.7620
18	StandardScalerWrapper DecisionTree	1.0000	0:00:19	0.7210	0.7620
19	MaxAbsScaler LightGBM	1.0000	0:00:17	0.7210	0.7620
20	StandardScalerWrapper GradientBoosting	1.0000	0:00:22	0.7607	0.7620
21	MaxAbsScaler RandomForest	1.0000	0:00:13	0.7155	0.7620
22	MaxAbsScaler DecisionTree	1.0000	0:00:18	0.7240	0.7620
23	MaxAbsScaler ExtremeRandomTrees	1.0000	0:00:18	0.7571	0.7620
24	StandardScalerWrapper RandomForest	1.0000	0:00:58	0.7261	0.7620
25	StandardScalerWrapper DecisionTree	1.0000	0:00:21	0.7268	0.7620
26	MaxAbsScaler LightGBM	1.0000	0:00:22	0.7171	0.7620
27	MaxAbsScaler LightGBM	1.0000	0:00:18	0.7131	0.7620
28	MaxAbsScaler RandomForest	1.0000	0:00:16	0.7542	0.7620
29	Ensemble	1.0000	0:01:59	0.7674	0.7674

サービス公開に向けて

すでにモデル構築済み

GCP/CLOUD MACHINE
LEARNING ENGINE

AWS/Amazon SageMaker

Jupyter Notebookで構築

Microsoft/Azure Machine
Learning Service
(AWS/AmazonSageMaker)

コードを書かないで構築

ブロック組み立て型

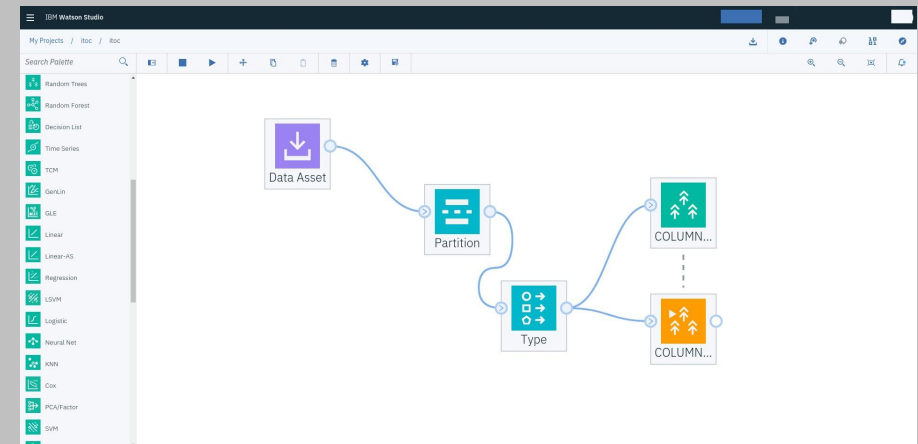
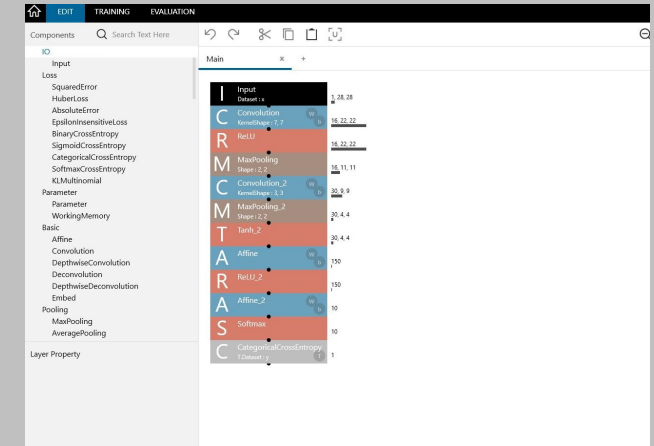
- データ処理フロー構築
- ニューラルネットワーク層を構築

データ登録型

- 画像系
- テーブルデータ系

ブロックで組み立てる

- 先程も書いたが最近特にサービスが多い
(MatrixFlow, nehan, Dataiku)
 - IBM/Watson Studio
 - (後でデモンストレーション予定)
 - Microsoft/Azure Machine Learning Studio
 - サービスまでデプロイはできないが、画像処理ディープラーニングの相性が良い



サービス公開に向けて

すでにモデル構築済み

GCP/CLOUD MACHINE
LEARNING ENGINE

AWS/Amazon SageMaker

Jupyter Notebookで構築

Microsoft/Azure Machine
Learning Service
(AWS/AmazonSageMaker)

コードを書かないで構築

ブロック組み立て型

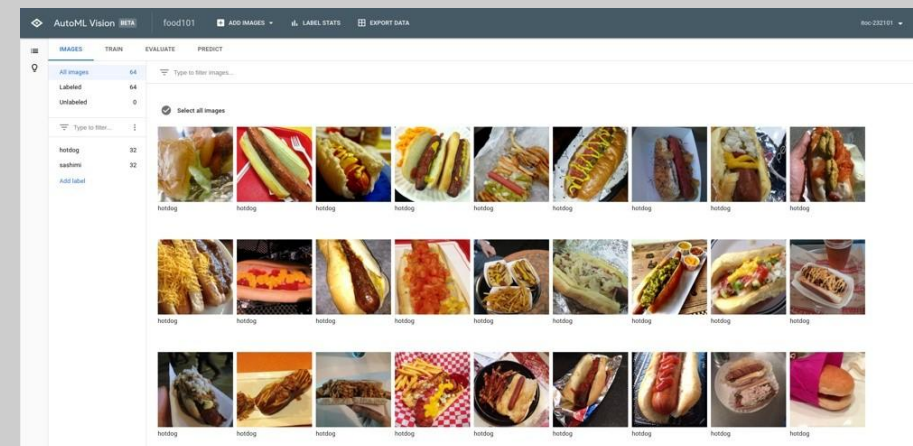
- データ処理フロー構築
- ニューラルネットワーク層を構築

データ登録型

- 画像系
- テーブルデータ系

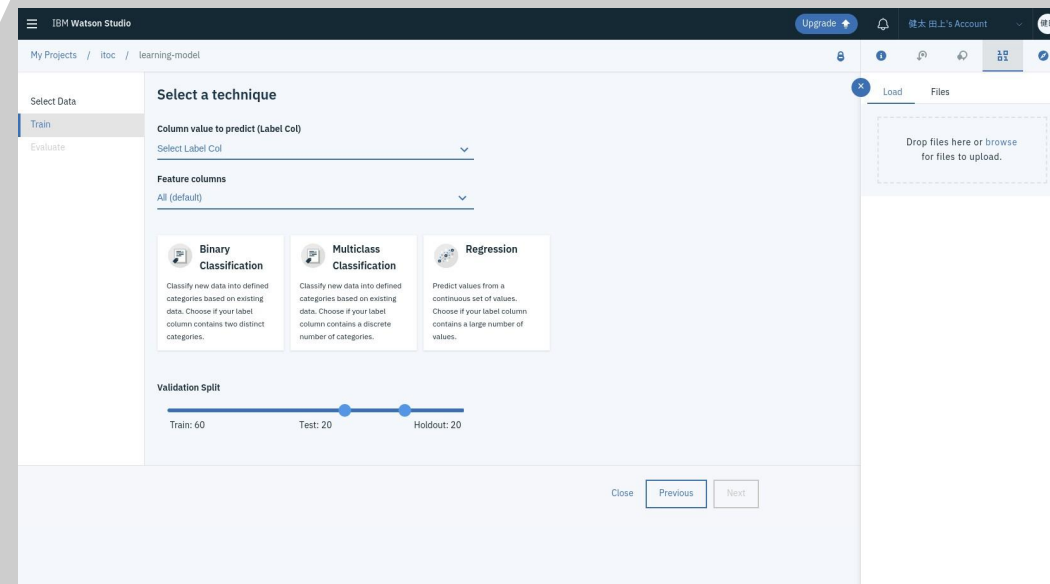
データ登録型(画像系)

- GCP/Cloud AutoML
 - 画像を登録してラベルを付けるだけで学習を行うことが可能



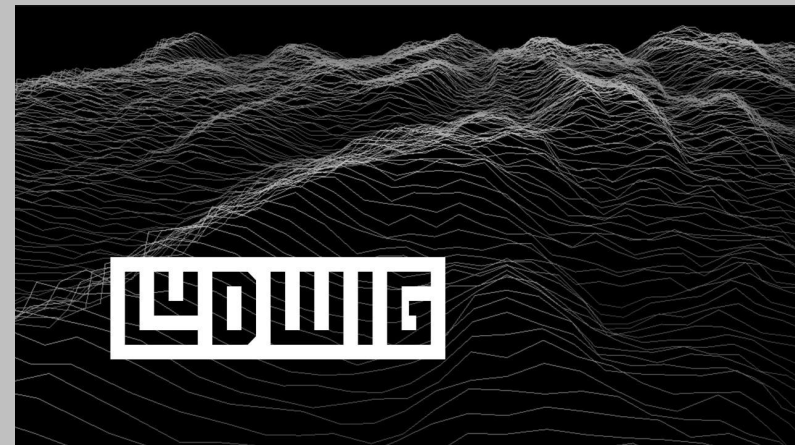
データ登録型(テーブルデータ)

- IBM/Watson Machine Learning
 - CSV等でデータを登録して、項目を設定するだけで学習が行えて、デプロイまで可能



データ登録型(テーブルデータ)

- [Uber/Ludwig](#)
 - カラム情報を記載したyamlファイル とデータCSVを準備すれば、ローカルのコンソールで学習ができてしまう
 - 画像データ等もある
- [Facebook/Prophet](#)
 - 時系列に特化しており、DataFrame を渡すだけで、未来の予測をしてくれる

The image shows the Prophet logo, which consists of the word "PROPHET" in a bold, white, sans-serif font. The text is centered within a solid blue rectangular box. The letter 'O' is stylized with a small white dot above it, resembling a prophetic or celestial symbol.

デモンストレーション

- 実際に新しくなったMicrosoftのAzure Machine Learning serviceの新機能 Visual interface を触ってみる
- UberのLudwigでもやってみる
- [Kaggle](#)にコードを書かずに挑む!!