

# アドバンス・トップエスイー 最先端ソフトウェアゼミ成果発表

## AI・機械学習 グループ

---

2018年 8月 30日

太田 裕一

北野 健太

早川 芳昭



# アジェンダ

---

## 1. 目標

## 2. 取組内容

- ① kaggleタイタニックチュートリアル(\*全員)
- ② 研究テーマ・問題意識に応じた取り組み(\*手分け)
  - A)時系列データの機械学習
  - B)機械学習とエンジニアリング

## 3. まとめ・所感

# 1. 目標

---

- 機械学習(ML)を自分で動かせるようになる
- 各自の研究テーマに関する知識やノウハウを習得する
- チームで進めることで相乗効果を発揮する

## (背景)

- 機械学習のプログラムを作って動かした経験がほぼ無かった
- 理論や知識よりも、実際に作って実感することを優先
- 研究テーマ(プロフェッショナルスタディ)でも機械学習の活用を検討

## 2. 取組内容

---

### ①(前半)kaggleタイタニックチュートリアル

- 機械学習プロジェクトの進め方の学習
- 参考情報を元に実行
- 追加で試行錯誤したこと
- 最終結果

### ②(後半)研究テーマ・問題意識に応じた取り組み

A) 時系列データの機械学習

B) 機械学習とエンジニアリング

## 2. ①機械学習プロジェクトの進め方

機械学習プロジェクトの進め方を書籍・ブログ等で学び、以下の流れで学習と実践を進めた。

1. 分析環境の構築
2. データの入手と予測対象の選定
3. データの観察と前処理
  1. 予測に利用する目的変数の選定
  2. データの欠損値の対処
  3. データの外れ値の対処
4. 機械学習のモデル作成
  1. モデルの評価
5. 予測モデルのハイパーパラメータチューニング
  1. 過学習と交差検証
6. 予測実施
7. 予測結果の評価

## 2. ①kaggle「Titanic: Machine Learning from Disaster」とは

- ・Kaggleとは、データサイエンティスト/機械学習エンジニアを繋げるプラットフォームで、最大の目玉がコンペ
- ・当該コンペ(チュートリアル)は、乗客データから生存・死亡を予測し、正解率を競うもの
- ・昨年の個別ゼミでも取り組んでおり、約79%
- ・**80%以上を目指す**ことを目標に設定

変数	内容
Survival	死亡:0、生存:1
PassengerId	乗客ID(昇順)
Pclass	チケット階級(1,2,3)
Name	名前
Sex	性別(male,female)
Age	年齢
Sibsp	同乗者(兄弟、配偶者)
Parch	同乗者(親、子供)
Ticket	チケット番号
Fare	旅客運賃
Cabin	キャビン番号
Embarked	乗船港

## 2. ①参考情報を元に行

- Webの参考情報を元に、以下実施

- 1) 分析環境の構築、データ入手

- 2) データの観測と前処理

- 3) モデル(決定木)による予測

- 約76%達成

### ▼Pandas の使い方を学習(以下、サンプル)

```
import pandas as pd
```

```
train = pd.read_csv("./train.csv")
```

```
test = pd.read_csv("./test.csv")
```

```
# 先頭5行を返す
```

```
print(train.head())
```

```
# 基本統計量(平均、標準偏差、MAX値など)を確認
```

```
print (train.describe())
```

### ▼欠損値の確認と補完

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	0	0	263	0	0	0	1	1014	2

平均値、中央値、ランダム値  
などで補完

予測に使わない

## 2. ①追加で試行錯誤したこと

・独自に、以下を試行錯誤

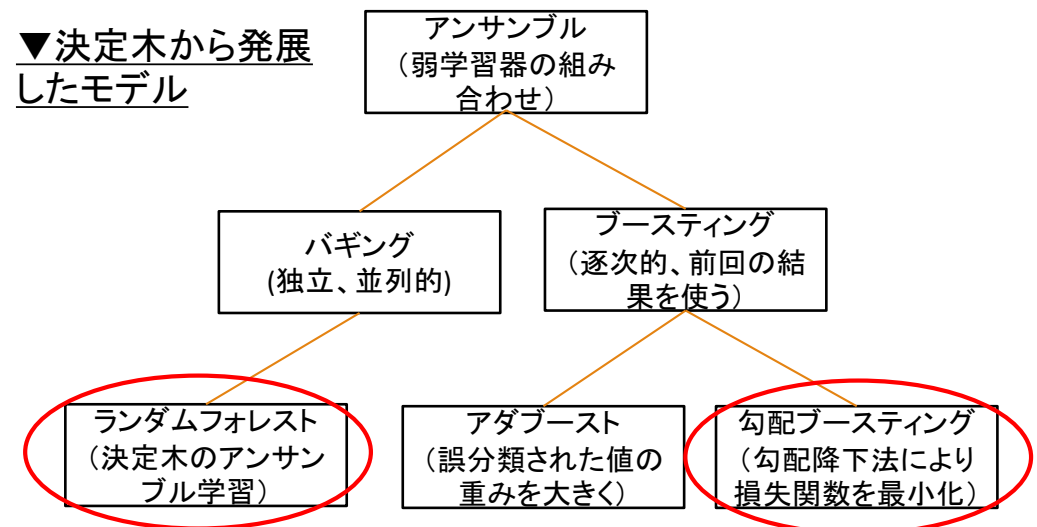
- 1) 予測に使う変数
- 2) 決定木以外のアルゴリズム
- 3) パラメータ調整

▼モデルのパラメータ調整

```
from sklearn.ensemble import GradientBoostingClassifier
features = train_two[["Pclass", "Age", "Sex", "Fare", "family_size", "Embarked"]].values
target = train["Survived"].values
forest = GradientBoostingClassifier(n_estimators=55, random_state=9)
forest = forest.fit(features, target)
```

決定木の数、乱数ジェネレータの初期値

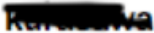

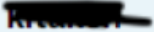



▼決定木から発展したモデル





## 2. ①最終結果

- ・約80.4%  
(1065位／9760チーム中)
- ・目標の80%を超えた

1064	new			0.80382
1065	new			0.80382
<b>Your Best Entry ↑</b> Your submission scored 0.80382, which is not an improvement of your best score. Keep trying!				
1066	new			0.80382

## 2. ① kaggleタイタニックチュートリアルまとめ

---

・機械学習のプログラムの組み方や学習、予測など一連の手続き、データ加工のライブラリの使い方などを学ぶことができた(ウォームアップ完了)

### (1) データの前処理が大事

・実際にチュートリアルを試してみて、欠損値の補完やデータ形式の変更などデータ加工の重要性を実感

### (2) 世の中の知見を使うと素人でもなんとかなる

・機械学習素人でも何かしら動くものが作れるほど、便利なライブラリ、試行錯誤の結果をまとめたブログ、サンプルコードなどが世の中に充実

### (3) そう甘くない、まだまだこれから

・結果として、目標の80%を超えることができたが、より根拠をもって精度を高めていけるように、データ分析やモデルのチューニングスキルの向上に努め、研究テーマへつなげていきたい

## 2. 取組内容

---

### ①(前半)kaggleタイタニックチュートリアル

- ・機械学習プロジェクトの進め方の学習
- ・参考情報を元に実行
- ・追加で試行錯誤したこと
- ・最終結果

### ②(後半)研究テーマ・問題意識に応じた取り組み

A)時系列データの機械学習

B)機械学習とエンジニアリング

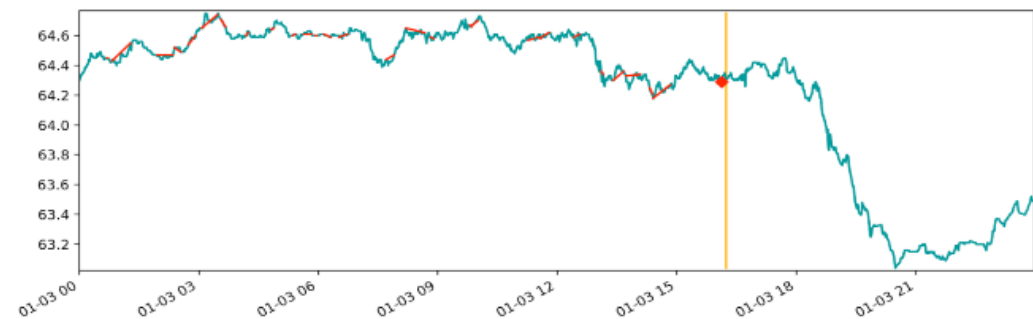
## 2. ②A)時系列

### 1)時系列データとは

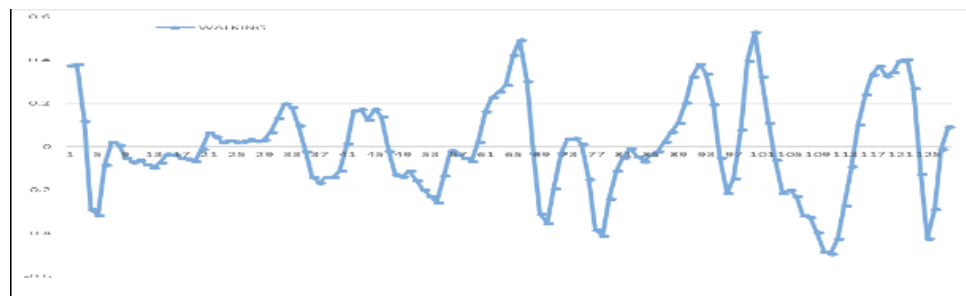
時間の推移とともに観測されるデータ。順序に意味をもち、相互に統計的依存関係が認められるようなデータの系列。(例:年度ごとの国民総所得, 月ごとの失業率, 毎日の株価, センサーデータなど)

### 2)分析の目的

- ・将来の平均値や変動幅などの予測
- ・変数間の動学的関係を明らかにする
- ・経済理論やファイナンス理論の検証



- ・人間行動認識
- ・行動結果予測
- ・動作検知



## 2. ②A)時系列

---

### 3)時系列データの種類

#### ① 原系列

時系列データそのもののこと

#### ② 対数系列

対数変換をかけたもの。対数変換を行う理由は、原系列では定常性の仮定が満たされないデータが、変換を行うことで解決する場合が多いため。

#### ③ 差分系列(階差系列)

1時点離れたデータとの差をとったデータのことを、差分系列または階差系列と呼ぶ。

#### ④ 対数差分系列

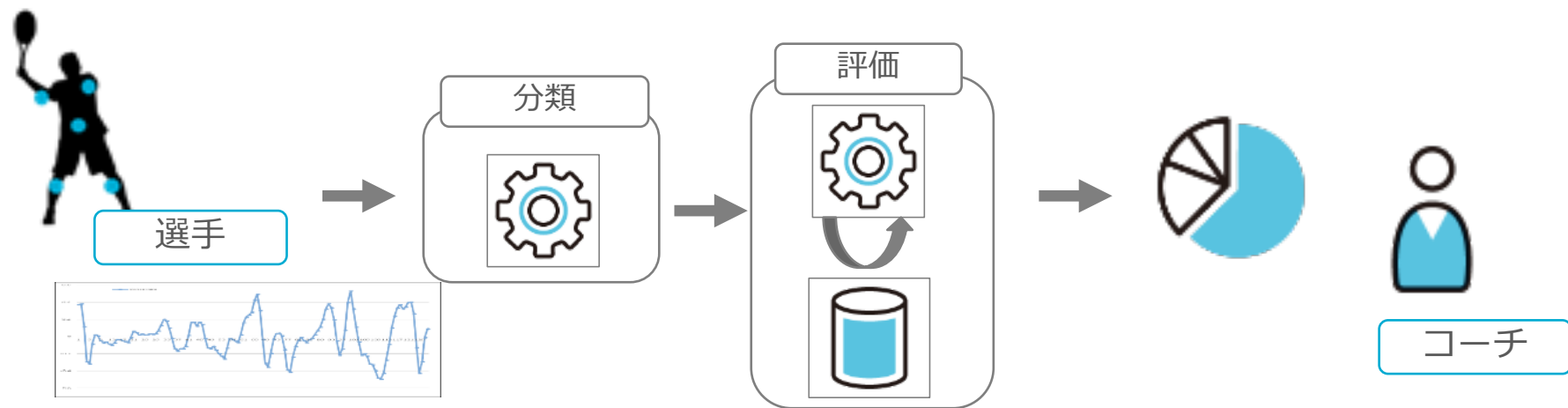
対数系列に対して差分系列をとったもの

( \* ほかには、季節変動による影響を取り除いた季節調整済み系列などもある )

## 2. ②A)時系列(センサーデータ分類問題)

### 背景

- ・スポーツのコーチングを支援するため、慣性センサーデータの活用を検討。
- ・時系列に採取したデータの分類問題を解くため、機械学習・深層学習を調査・試行。



## 2. ②A)時系列(センサーデータ分類問題)

---

### 取り組み

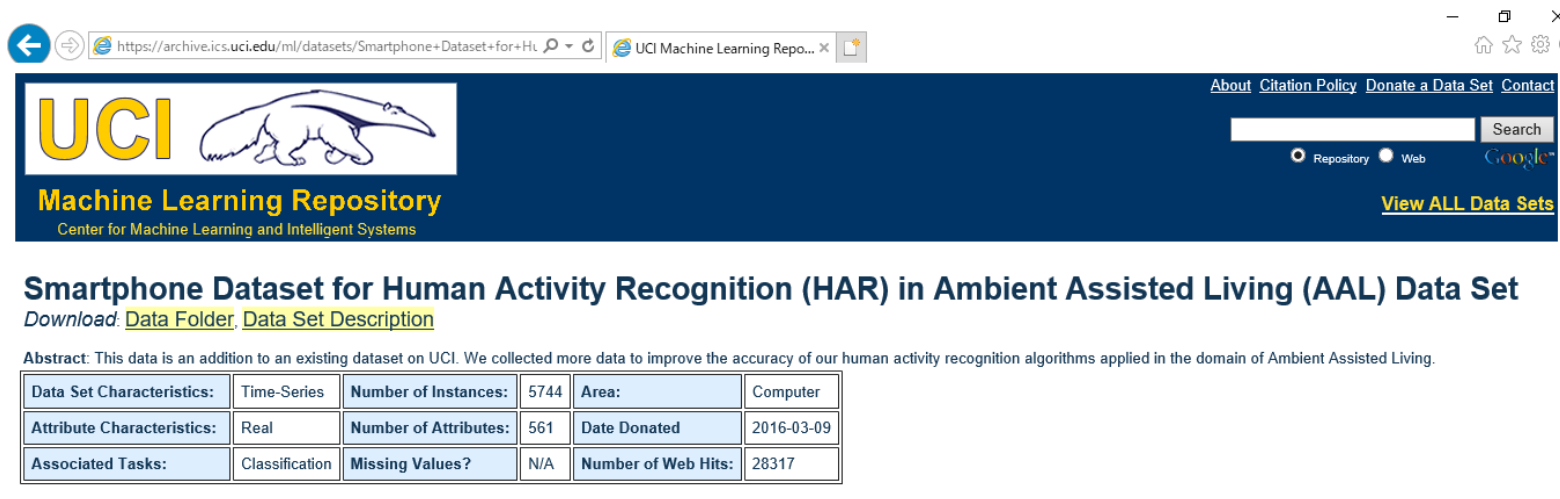
1. 既存の研究とデータセットを調査  
慣性センサーデータの分類問題を解く研究を調査。  
センサーを使って採取したデータセットを調査。
2. データ量と精度の関係を把握  
バドミントン関連のデータセットが見つからず、自前で調達を検討。  
データ件数の目標と、検証時の目標の目安となる基準を作成。
3. 深層学習の基本を調査  
モデルやハイパーパラメータの変更の影響確認。

## 2. ②A)時系列(センサーデータ分類問題)

### センサーデータの分類問題

#### ・調査結果

UCI Machine Learning Repository: Human Activity Recognition Using Smartphones Data Set (\*)



The screenshot shows the UCI Machine Learning Repository website. The header includes the UCI logo, the text "Machine Learning Repository", and the subtitle "Center for Machine Learning and Intelligent Systems". There are links for "About", "Citation Policy", "Donate a Data Set", and "Contact". A search bar is present with a "Search" button. Below the header, the title "Smartphone Dataset for Human Activity Recognition (HAR) in Ambient Assisted Living (AAL) Data Set" is displayed. Below the title, there are links for "Download: Data Folder" and "Data Set Description". An abstract paragraph follows, stating: "Abstract: This data is an addition to an existing dataset on UCI. We collected more data to improve the accuracy of our human activity recognition algorithms applied in the domain of Ambient Assisted Living." Below the abstract is a table with the following data:

Data Set Characteristics:	Time-Series	Number of Instances:	5744	Area:	Computer
Attribute Characteristics:	Real	Number of Attributes:	561	Date Donated	2016-03-09
Associated Tasks:	Classification	Missing Values?	N/A	Number of Web Hits:	28317

(\*) <https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones>



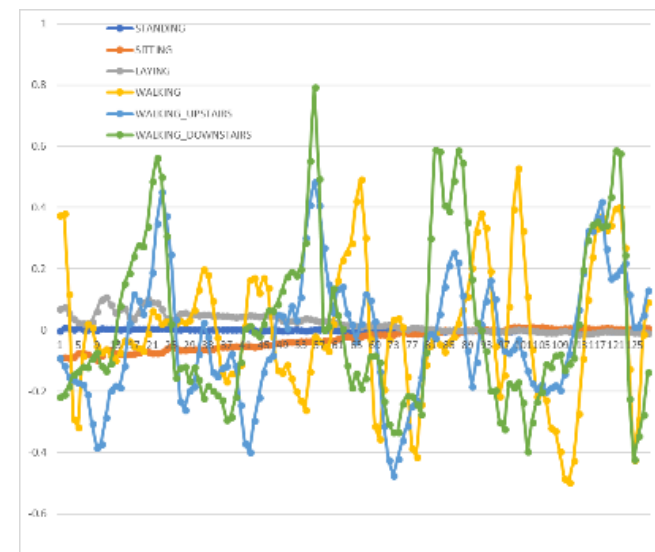
## 2. ②A)時系列(センサーデータ分類問題)

### センサーデータの分類問題

#### ・調査結果

UCI Machine Learning Repository: Human Activity Recognition Using Smartphones Data Set

	参考(※)
被験者	ボランティア 30人
年齢	19～48歳
行動種類	動的なデータ・・・歩く、昇る、降りる 静的なデータ・・・座る(座り続ける)、立つ(立ち続ける)、寝る
収集デバイス	スマートフォン(Samsung Galaxy S II)
装着箇所	腰
サンプリングレート	50Hz
ラベリング手法	映像からラベルを手動生成
データ量	10299(学習データ:テストデータ 7:3)(1データにつき2.56秒分のサンプリング)



## 2. ②A)時系列(センサーデータ分類問題)

### センサーデータの分類問題

- ・調査結果

「Deep Learning for Sensor-based Activity Recognition: A Survey」

CNNで、分類問題を解く。

- ・メリット

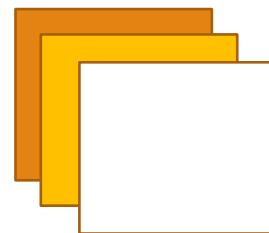
- ・局所依存性・・・隣り合う信号が相関する可能性が高い

- ・スケール不変性・・・ペース、周波数などが変わってもスケールは不変

- ・CNNの入力に合わせる

データ駆動型

モデル駆動型



画像の場合



センサー

- ・データ駆動型

センサーデータを

縦幅1の仮想的な画像として入力

センサーの次元(x軸など)を

チャンネルとして入力

(画像の場合はRGB等で3チャンネル)

(\*) Deep Learning for Sensor-based Activity Recognition: A Survey

## 2. ②A)時系列(センサーデータ分類問題)

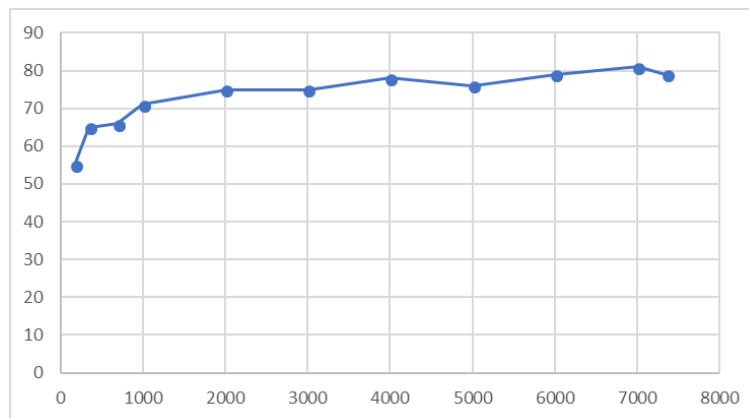
### センサーデータの分類問題

#### ・試行

TensorFlowを使って、CNNで分類問題を解く。

convolution + pooling + convolution + pooling + dense + dense + dense + output

#### ・学習データ量の変更



#### ○学習データ量の変化

・半人データ(176件)で55%

・約2000件で75%

→ 数百件でもある程度の精度が上昇する。

#### ○学習データ種類の変化

・動的なデータのみ100件 18%~20%

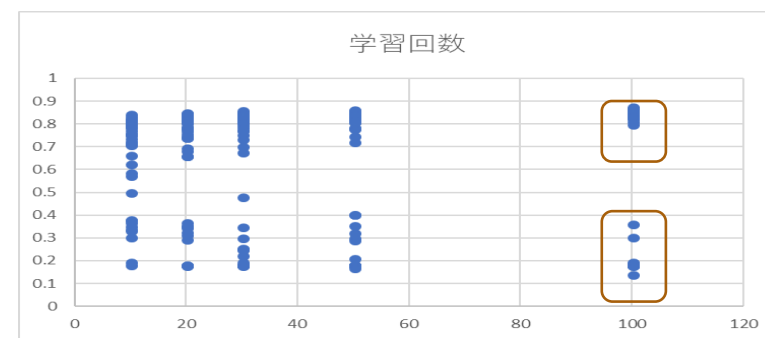
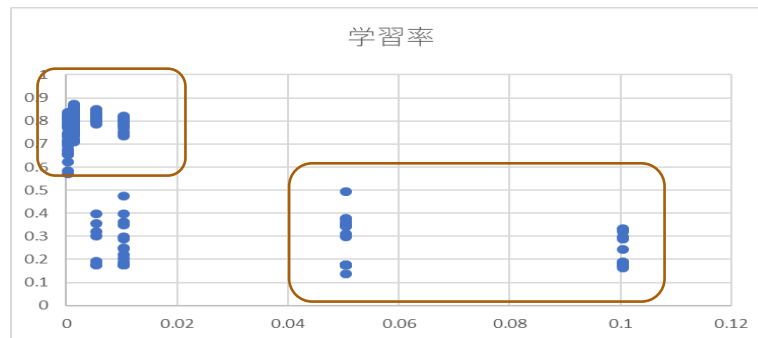
・静的なデータのみ78件 33%~43%

→ 動きによって精度が変わる。

## 2. ②A)時系列(センサーデータ分類問題)

### センサーデータの分類問題

#### ・ハイパーパラメータの変更



- ・学習率によって精度が大きく変わる。
  - ・学習率があていない状況で、学習回数を増やすと、かえって精度が下がる。
- パラメータ設定手順
- 1、学習回数を固定し、学習率の精度の変更。
  - 2、学習回数を変更。
  - 3、ドロップアウトを変更。

※グラフデータは「A tutorial for using deep learning for activity recognition」を参照  
<https://github.com/jindongwang/Deep-learning-activity-recognition>

## 2. ②A)時系列(センサーデータ分類問題)

---

### 取り組み

バドミントンのフォーム分類を行いたい。

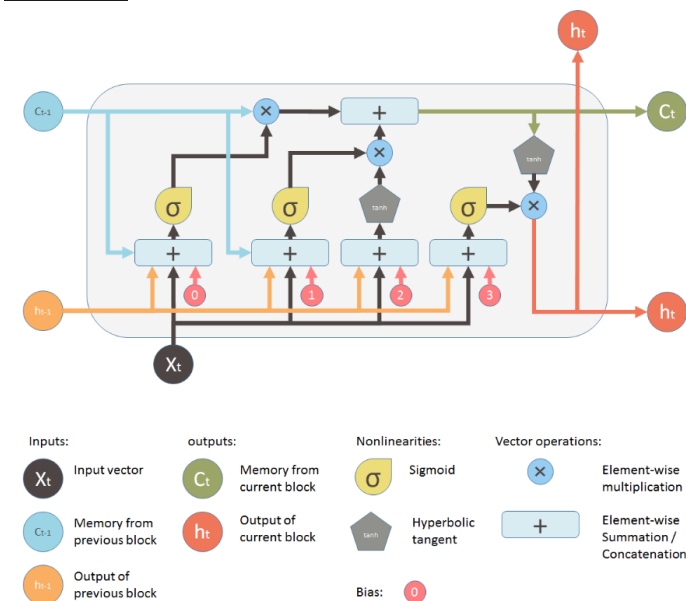
1. 既存の研究とデータセットを調査  
センサーデータを分類する深層学習を試行する環境ができた。  
入力(モデル駆動型)や、RNNなど他の手法も調査し、試行したい。
2. データ量と精度の関係を把握  
数百件のデータでも60%近い精度が出ることが分かった。  
動的データ、静的データの違いについて試行したい。
3. 深層学習の基本を調査  
大まかなパラメータ設定手順と精度の関係を把握した。  
モデルやフィルターの変更などが、どのような影響を与えるのか試行したい。

## 2. ②A)時系列(為替予測(狙い))

- ・ディープラーニングに関する知見を高めたい
- ・与える学習データ、学習パラメータを変えることで結果(精度や学習時間、リソースなど)がどのように変わるか

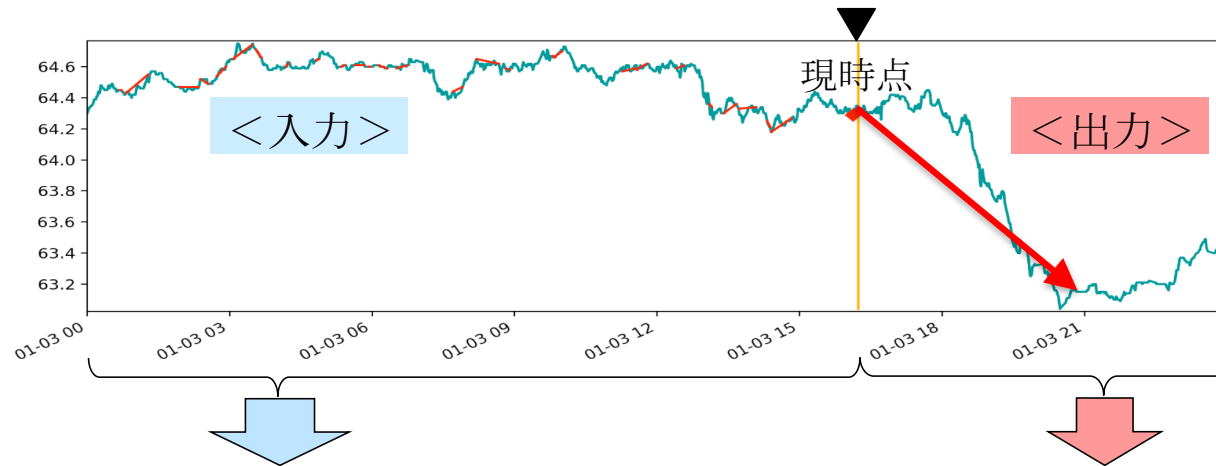
- ・過去の為替の値動き(時系列データ)を元に将来の為替の予測
- ・LSTM(Long short-term memory)というモデルを採用し、Kerasで実装
- ・LSTMは、RNNの拡張として登場、時系列データに向くモデル

### ▼LSTM



## 2. ②A)時系列(為替予測(問題設定))

- ・60秒間の入力を元に、5秒後の騰落予測(上がるか／下がるか)、および増減幅を絶対値で表現



過去の為替の値動きデータ(1秒足データ)  
60次元のベクトル(原系列を正規化して利用)

20160128 00:00:00,118.4815  
20160128 00:00:01,118.4815  
20160128 00:00:02,118.6315  
20160128 00:00:03,118.6304  
...

上がる → 正の値  
下がる → 負の値  
( -0.7 ~ 0 ~ 0.7 )

\*絶対値が大きいほど、大きく値が動くと予想

## 2. ②A)時系列(為替予測(実施内容))

- ・どのくらいの学習データが必要なのかわからない(1週間ずつ増やしていく)
- ・学習パラメータをチューニング(マシンリソースや過学習との兼ね合い)

項目	内容	備考
学習期間	2016/1/4~2016/1/10	86400秒/日 → 1440個のデータ/日 約1万データ/週
評価期間	2016/1/28~2/3 2016/11/7~11/11	各日約9万レコード
計算機	Macbook pro2015 2.7GHz、8G	
使用レート	USD/JPY	FX Gridと呼ばれるオープンデータを活用 ( <a href="https://www.truefx.com">https://www.truefx.com</a> )
パラメータ	・エポック数 ・バッチサイズ	エポック1から増加 バッチサイズ100, 1000

### ▼ミニバッチ法

(例) 1週間のデータ、エポック2、バッチサイズ1000で学習する場合

全1万データ

ランダムにバッチサイズごとに切り出し、全データ分処理する  
(10000/1000=10回)

1000 1000 ...

エポック数繰り返す  
(2回)



## 2. ②A)時系列(為替予測(結果1))

- ・学習データ1週間
- ・エポック2
- ・バッチサイズ1000

▼結果 49%

予測値、予測数、正解数

0.0,34123,17020

-0.1,5749,2829

-0.2,613,323

-0.3,55,15

-0.4,0,0

-0.5,0,0

-0.6,0,0

-0.7,0,0

0.1,7782,3987

0.2,60,26

0.3,0,0

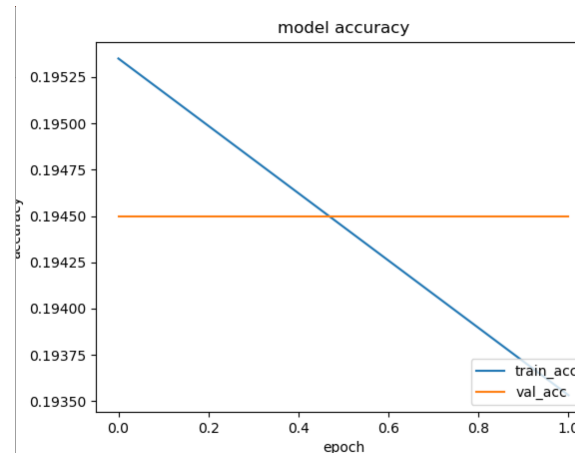
0.4,0,0

0.5,0,0

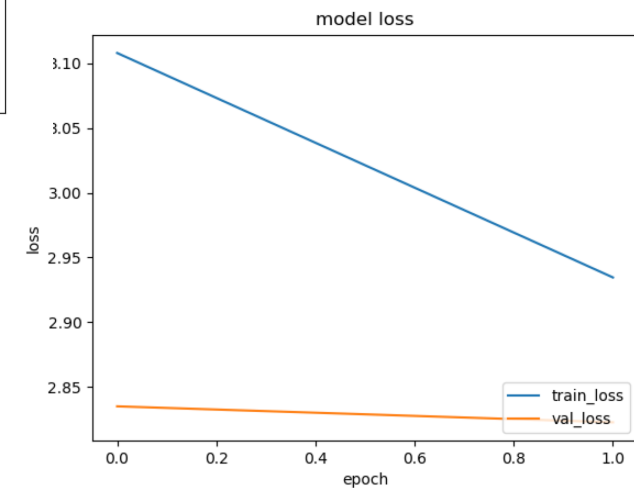
0.6,0,0

0.7,0,0

-0.3~0.3くらいの範囲の値に留まる



▼まだ収束していないように見える



## 2. ②A)時系列(為替予測(結果2))

- ・学習データ1週間
- ・エポック2
- ・バッチサイズ100

▼結果 47%

予測値、予測数、正解数

0.0,17078,8417

-0.1,421,188      0.1,44,24

-0.2,46,24      0.2,1,1

-0.3,4,1      0.3,0,0

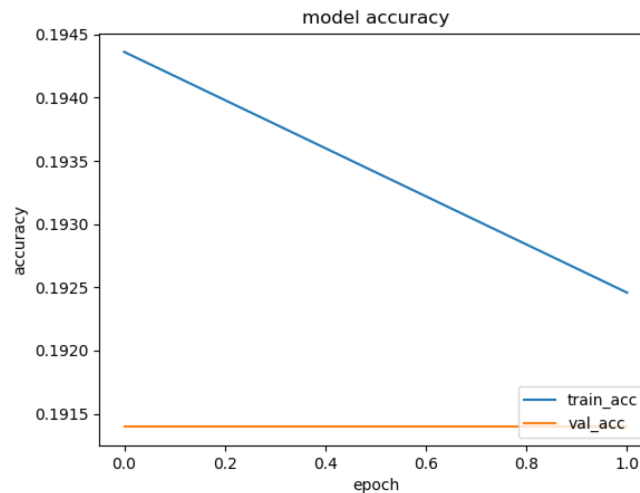
-0.4,0,0,0      0.4,0,0

-0.5,0,0,0      0.5,0,0

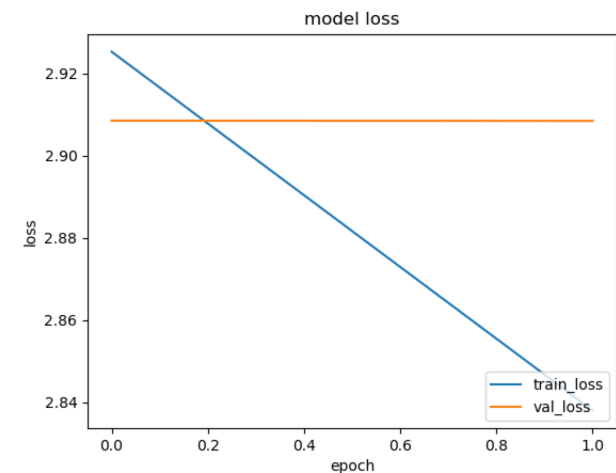
-0.6,0,0      0.6,0,0

-0.7,0,0      0.7,0,0

-0.3~0.3くらいの範囲の値に留まる



▼まだ収束していないように見える



## 2. ②A)時系列(為替予測(結果3))

- ・学習データ1週間
- ・エポック10
- ・バッチサイズ1000

▼結果 50%

予測値、予測数、正解数

0.0,43917,21970

-0.1,374,201

-0.2,38,21

-0.3,5,5

-0.4,0,0

-0.5,0,0

-0.6,0,0

-0.7,0,0

0.1,46,29

0.2,6,4

0.3,0,0

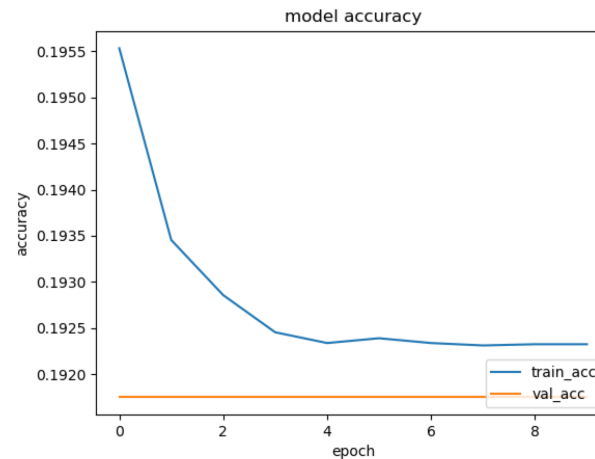
0.4,0,0

0.5,0,0

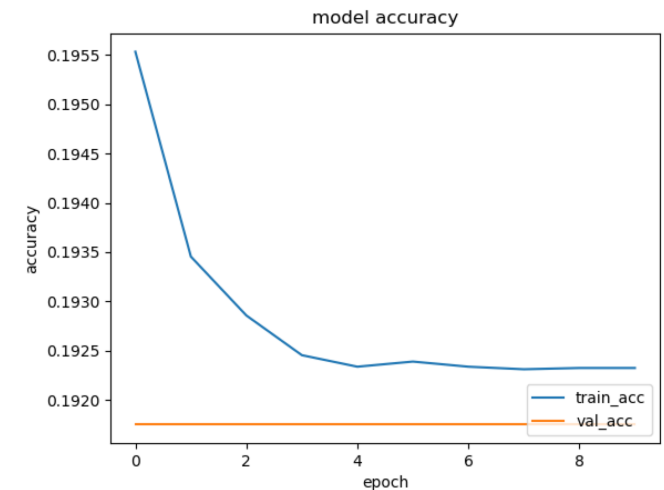
0.6,0,0

0.7,0,0

-0.3~0.3くらいの範囲の値に留まる



▼エポック4、あたりから収束



## 2. ②A)時系列(為替予測(結果4))

### ■上昇すると予測

- ・学習データ1ヶ月
- ・エポック10
- ・バッチサイズ100

予測値	1/28	1/29	2/1	2/2	2/3	11/9	11/10	11/11	11/14	11/15
1月  00	2%	45.8%	41.3%	42.7%	46.0%	47.9%	48.0%	45.7%	46.5%	45.7%
	2%	46.6%	41.0%	42.3%	44.9%	47.5%	47.5%	45.9%	44.4%	44.9%
	9%	48.7%	41.0%	41.1%	43.9%	45.6%	46.5%	46.0%	44.3%	44.1%
	8%	47.5%	42.2%	39.7%	39.1%	44.7%	47.0%	44.1%	45.6%	43.2%
	1%	48.3%	44.6%	41.5%	39.9%	45.0%	45.8%	43.9%	42.7%	40.8%
x > 0.6	36.5%	54.8%	43.3%	42.9%	40.7%	43.8%	49.6%	43.0%	41.5%	38.5%
x > 0.7	36.5%	47.5%	46.8%	45.2%	36.6%	47.5%	49.1%	38.9%	38.9%	41.9%

−0.7〜0.7の範囲  
で予測するように

下降に強い、とい  
う特徴を備えたモ  
デルができた

### ■下降すると予測 …… 下降ではうまく行っている??

予測値	1/28	1/29	2/1	2/2	2/3	11/9	11/10	11/11	11/14	11/15
$x < -0.1$	55.7%	53.4%	59.4%	57.4%	54.2%	51.5%	52.1%	53.5%	54.9%	54.5%
$x < -0.2$	56.3%	53.9%	61.2%	58.9%	55.1%	52.3%	52.5%	53.9%	55.4%	55.3%
$x < -0.3$	57.2%	54.2%	63.0%	59.6%	55.9%	54.0%	53.6%	53.8%	53.5%	54.9%
$x < -0.4$	59.2%	54.1%	62.0%	58.1%	52.9%	54.0%	56.6%	52.7%	51.1%	53.9%
$x < -0.5$	63.3%	54.7%	65.0%	62.7%	53.1%	54.2%	59.4%	54.1%	58.8%	57.2%
$x < -0.6$	68.8%	57.3%	73.8%	67.5%	60.0%	57.8%	55.2%	54.6%	58.1%	58.1%
$x < -0.7$	75.4%	58.3%	69.7%	71.4%	61.5%	47.4%	52.9%	45.7%	86.7%	53.6%

## 2. ②A)時系列(為替予測(まとめ))

- ・与える学習データ、パラメータを変えることで、学習に要する時間や学習の進行具合などがどのように変わるか、いくつかの知見を得ることができた
- ・今後は、モデル精度を上げるためのチューニング手法などを試行錯誤したい
  - 1)やはり学習には大量のデータ量が必要だということを実感
    - ・当初は、ほとんど予測をしていなかった( $-0.3 \sim 0.3$ くらいの範囲の値に留まる)
    - ・1ヶ月分(約4万件のデータ)であれば( $-0.7 \sim 0.7$ の幅で)予測するようになった
  - 2)学習を高速で回すための環境整備(マシンリソースの調達)の重要性を実感
    - ・データ量やエポック数を増やすにつれ、学習に時間がかかり過ぎるようになってきた
    - ・AzureにGPU(112 GB メモリ)のマシンを構築して学習  
(ローカルPCで、8時間以上かかっていたものが、1.5時間で完了)
  - 3)エポック数の増加により、学習が進むことを確認  
(同じデータで何度も学習するため、過学習する可能性が高まる)
  - 4)バッチサイズを大きくすると処理速度やリソースを必要とする傾向を確認

## 2. ②B)機械学習とエンジニアリング

---

機械学習という先端技術を取得・活用する上で、「エンジニアリング上課題となりうるポイントを把握する」という、目標を掲げ機械学習の活用に取り組んだ。今回その時に課題と感じた点と工夫し対応した、以下の4点について説明します。

1. クラウド上の実行環境を活用
2. ハイパーパラメータチューニングのコストの低減
3. コードと学習結果の管理
4. コードの品質担保

## 2. ②B)機械学習とエンジニアリング

---

### クラウド上の実行環境を活用

#### 実行環境の構築

機械学習の実行環境構築には扱うデータ量に応じて強力な計算能力を持つマシン環境の準備が必要となる。  
また、ミドルのインストールや共同作業時のコードの共有という点も配慮する必要がある。  
機械学習でシェアの高いpython, scikit-learnを利用したが、実行環境はクラウド上の実行環境を利用する事で、時間短縮を試みた。

本ゼミでの取り組み: Google社が提供しているGoogle Colaboratoryを利用  
従来: Linuxマシンを調達し導入しミドルの導入等環境を構築

#### Google Colaboratory利用の効果

実行環境の構築コストが0で即座に機械学習のコード試行に移れた。  
作成したコードはチーム内で共有し、相互にノウハウを共有できた。  
数百Mbyteにおよぶ学習・評価データの高速な分析が行えた。  
GPUが無料で使えDeepLearning系の学習が高速で行えた。

#### Google Colaboratoryとは

Google社が機械学習の教育や研究用に無償で提供している環境でブラウザ上でPythonの実行環境( Jupyter notebook )が利用できる。作成したコードについてはGoogle Driveを介して複数ユーザで共有できる。

参考: <https://colab.research.google.com/notebooks/welcome.ipynb>

## 2. ②B)機械学習とエンジニアリング

### ハイパーパラメータチューニングのコストの低減

機械学習ではモデル化(決定木・ランダムフォレスト)の後、学習時に様々なパラメータの設定が可能。

一方でパラメータは複数の種類があり人手での検証は費用対効果が見合わない。

scikit-learnではGridSearchというパッケージを持っており、評価関数と複数パラメータの組み合わせを試行し、最適なパラメータを評価用として利用できる。

### パラメータ例

決定木: 深さ(max\_depth)、ランダムフォレスト: 経過回数(n\_estimators)

### 利用の効果

以下の例で説明すると、18(6x3)パターンから最適なパラメータを自動決定できコード変更や結果の管理が不要であり、複数試行したこともあり、その分工数を圧縮できた。

### GridSearchの利用例

```
parameters = {  
    'n_estimators': [5, 10, 20, 30, 50, 55],  
    'max_depth': [5, 10, 15]  
}  
  
gsc = GridSearchCV(XGBClassifier(), parameters, cv = 10, scoring = 'roc_auc',  
    my_tree_two = gsc.fit(features_two, target)  
    print(gsc.best_params_)  
  
my_prediction_tree_two = my_tree_two.predict(test_features_2)
```

パラメータとパターンを列挙することで、総当たりで組み合わせを試行し、最もスコアの良いパラメータの抽出を行える。

→{'max\_depth': 5, n\_estimators: 20}の形で最良パラメータの表示もできる。

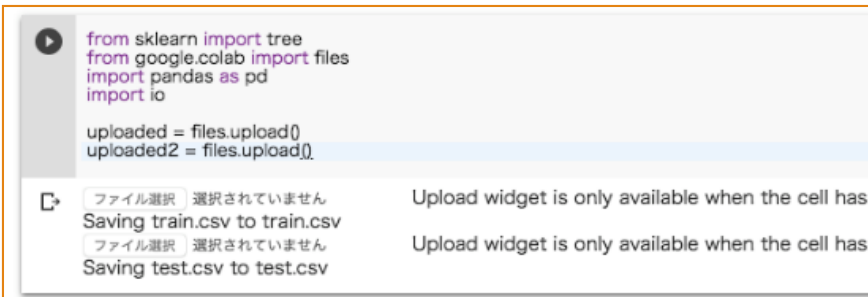


## 2. ②B)機械学習とエンジニアリング

### コードと学習結果の管理

システム開発のプログラミングとは異なり、前処理の変更や、モデル変更や試行錯誤の回数が極めて多い  
コードと予測結果の管理はjupyter notebook上でもコードと実行可能を残すことは可能であるが、複数回繰り返すと行が増えすぎ正しく対応が取れなくなる恐れがある  
→これからは構成管理サービス(GitHub等)と組み合わせるのが理想的と考える

### 現状の管理



```
from sklearn import tree
from google.colab import files
import pandas as pd
import io

uploaded = files.upload()
uploaded2 = files.upload()
```

ファイル選択 選択されていません Saving train.csv to train.csv  
ファイル選択 選択されていません Saving test.csv to test.csv

Upload widget is only available when the cell has

jupyter notebookでのコード例  
コードだけでなくテキストやグラフ描写も残す事ができる

## 2. ②B)機械学習とエンジニアリング

### コードの品質担保

従来のシステム開発と異なり、機械学習は処理によって論理的な誤り(例:前処理のミスやデータの選定ミス)があっても、ある程度動作し、それらしい結果を出してしまうという特徴がある。

一方、学習結果や予測結果によって前処理に工夫を行うことやモデルの見直し等、試行錯誤に近い作業が多々発生する。この結果、コードの修正が多々発生することで論理的な誤りを埋め込んでしまうか確率が上がる。この点については、従来のシステム開発を参考とし①関数化による処理の分割と②*unittest*フレームワークを活用することを検討した。

①については、前処理を関数化し、訓練データ、評価データともに同じ処理を適用できる設計にする等、可読性と保守性の良いコードを書くことができた。

②については、機械学習に向けた*unittest*フレームワークを見つけることができず、残念ながら適用を断念した。しかしながらQAサイトや技術サイト等でも必要性の言及が複数あったため、今後よいフレームワークが生み出されると期待できる。

```
def correct_data(titanic_data):  
    #Age欠損値の処理  
    titanic_data.Age = titanic_data.Age.fillna(titanic_data.Age.median())  
    #Sexの二値化  
    titanic_data.Sex = titanic_data.Sex.replace(['male', 'female'], [0, 1])  
    #Embarkedの欠損値の処理  
    titanic_data.Embarked = titanic_data.Embarked.fillna("S")  
    titanic_data.Embarked = titanic_data.Embarked.replace(['C', 'S', 'Q'], [0, 1, 2])
```

前処理の関数化コードの例

## 2. ②B)機械学習とエンジニアリング

---

### 取り組みの結果

1, 2に関しては課題として認識した点は技術の取得の過程で解決することができた。

特にクラウド上の環境を利用できたことでコードを書きさまざまなモデルを動かすことにより、機械学習を利用した、予測や分類といった分野の技術も取得できた。

一方、「コードと学習結果の管理」ではGitHubとの連携を、「コードの品質担保」ではユニットテストの活用に関してリソース的な制約で取り組めていない部分があり次回以降取り組んでいきたいと考える。

1. クラウド上の実行環境を活用
2. ハイパーパラメータチューニングのコストの低減
3. コードと学習結果の管理
4. コードの品質担保

### 3. まとめ・所感

---

#### (1) 機械学習(ML)を自分たちで動かせるようになった

- ・機械学習の動かした方、環境の構築やライブラリの実装方法などを学んだ
- ・データの傾向、特徴の確認や前処理のスキルを身につけられた
- ・一方で、大規模データの扱い方や数学的知識の乏しさを痛感

#### (2) 各自の研究テーマに関する知識やノウハウを習得できた

- ・時系列データおよび機械学習工学に関するノウハウを蓄積することができた
- ・各自の研究に直接的、間接的に役立てることができそうだ

#### (3) チームで進めることで相乗効果を発揮した

- ・週1回のゼミで議論したことで、理解が深まった
- ・時間を作るのが大変であったが、うまく役割分担できた