

一から始める機械学習 (Kaggleで学ぶ機械学習)

阿部 泰之 [twitter:@taki_tflare](https://twitter.com/taki_tflare)

このスライドの対象読者、目的

対象読者

- 機械学習の概要がわかっている方
- もしくは一から始める機械学習(機械学習概要)を読んだ方が対象です

目的

- Kaggleについて理解する
- 機械学習について実際の流れを理解する
- Kaggleのチュートリアルを用いて実践を行う
- scikit-learnを用いて実践を行う

アジェンダ

1. Kaggleとは
2. なぜKaggle?
3. Titanic: Machine Learning from Disaster
4. データの中でどれを使う?
5. データ前処理
6. 使用データ
7. 使用する学習手法
8. 学習実行と交差検証
9. 最適化
10. 機械学習の流れ
11. 企業のコンテストを一部紹介

1. Kaggleとは

- 誤解を恐れずにまとめれば

Kaggleは企業や研究家が、データサイエンス、機械学習関連のお題をだしてそれを解くサイトです。

賞金がつくものもあり（さらにそれを解くためのコードを公開しかつ説明する。その説明についてコメント等でコミュニケーションを行う機能もあり）

2. なぜKaggle？

- 本などの説明では、説明のためのデータセットが使われていることが多く、実感が湧きにくい。
- 本などの説明では端折られる箇所も、実施する必要があるため機械学習の実際の流れが理解できる。
- 順位が出るためやる気が出る。（全世界のデータ分析家と勝負できますし、協力もできます）
- 賞金がでます（150万ドルの賞金が出るものもあり）

3. Titanic: Machine Learning From Disaster

- 乗客がタイタニックの沈没を生き延びたかどうかを予測
- 訓練用データ (891行 × 12列のcsv) データに一部欠損あり
- テストデータ (418行 × 11列のcsv) データに一部欠損あり
- 訓練用データで学習し、テストデータに対して、生き延びたかどうかを予測する。

4. データの中でどれを使う？(1/4)

- PassengerId : データにシーケンシャルでついている番号
- Survived : 生存 (0 = No, 1 = Yes) 訓練用データにのみ存在
- Pclass : チケットのクラス (1 = 1st, 2 = 2nd, 3 = 3rd)
- Name : 名前
- Sex : 性別
- Age : 年齢

4. データの中でどれを使う？(2/4)

- SibSp : タイタニック号に乗っていた兄弟と配偶者の数
- Parch : タイタニック号に乗っていた両親と子どもの数
- Ticket : チケット番号
- Fare : 旅客運賃
- Cabin : 船室番号
- Embarked : 乗船場 (C = Cherbourg, Q = Queenstown, S = Southampton)

4. データの中でどれを使う？ (3/4)

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Archem)	female	14.0	1	0	237736	30.0708	NaN	C

- `import numpy as np`
- `import pandas as pd`
- `train = pd.read_csv("train.csv", dtype={"Age": np.float64},)`
- `test = pd.read_csv("test.csv", dtype={"Age": np.float64},)`
- `train.head(10)`

実行コード

4. データの中でどれを使う？(4/4)

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000

```
In [4]: #-1.0から-0.7 強い負の相関  
        #-0.7から-0.4 負の相関  
        #-0.4から-0.2 弱い負の相関  
        #-0.2から+0.2 ほとんど相関がない  
        #+0.2から+0.4 弱い正の相関  
        #+0.4から+0.7 正の相関  
        #+0.7から+1.0 強い正の相関
```

実行コード

```
train_corr = train.corr()  
train_corr
```

5. データ前処理(1/2)

PassengerId以外は使えそうです。

現在解析に使えていないデータがあるので使えるデータ（数値）に変換します。

また欠損データがあるので補正します。

実行コード

```
def correct_data(titanic_data):  
  
    titanic_data.Age = titanic_data.Age.fillna(titanic_data.Age.median())  
  
    titanic_data.Sex = titanic_data.Sex.replace(['male', 'female'], [0, 1])  
  
    titanic_data.Embarked = titanic_data.Embarked.fillna("S")  
    titanic_data.Embarked = titanic_data.Embarked.replace(['C', 'S', 'Q'], [0, 1, 2])  
  
    titanic_data.Fare = titanic_data.Fare.fillna(titanic_data.Fare.median())  
  
    return titanic_data  
  
train_data = correct_data(train)  
test_data = correct_data(test)
```


5. データ前処理(2/2)

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId	1.000000	-0.005007	-0.035144	-0.042939	0.034212	-0.057527	-0.001652	0.012658	-0.017443
Survived	-0.005007	1.000000	-0.338481	0.543351	-0.064910	-0.035322	0.081629	0.257307	-0.125953
Pclass	-0.035144	-0.338481	1.000000	-0.131900	-0.339898	0.083081	0.018443	-0.549500	0.305762
Sex	-0.042939	0.543351	-0.131900	1.000000	-0.081163	0.114631	0.245489	0.182333	-0.022521
Age	0.034212	-0.064910	-0.339898	-0.081163	1.000000	-0.233296	-0.172482	0.096688	-0.040166
SibSp	-0.057527	-0.035322	0.083081	0.114631	-0.233296	1.000000	0.414838	0.159651	0.030874
Parch	-0.001652	0.081629	0.018443	0.245489	-0.172482	0.414838	1.000000	0.216225	-0.035957
Fare	0.012658	0.257307	-0.549500	0.182333	0.096688	0.159651	0.216225	1.000000	-0.268865
Embarked	-0.017443	-0.125953	0.305762	-0.022521	-0.040166	0.030874	-0.035957	-0.268865	1.000000

```
In [ ]: #-1.0から-0.7 強い負の相関  
        #-0.7から-0.4 負の相関  
        #-0.4から-0.2 弱い負の相関  
        #-0.2から+0.2 ほとんど相関がない  
        #+0.2から+0.4 弱い正の相関  
        #+0.4から+0.7 正の相関  
        #+0.7から+1.0 強い正の相関
```

実行コード

```
train_data_corr = train_data.corr()  
train_data_corr
```

6. 使用データ

今回は以下項目を使用します。

- チケットのクラス
- 性別
- 年齢
- タイタニック号に乗っていた兄弟と配偶者の数
- タイタニック号に乗っていた両親と子どもの数
- 旅客運賃
- 乗船場

7. 使用する学習手法(1/2)

- ロジスティック回帰
- サポートベクターマシン
- k-最近傍法
- 決定木
- ランダムフォレスト
- ニューラルネットワーク

7. 今回使用する学習手法(2/2)

参考文献

- Pythonではじめる機械学習 scikit-learnで学ぶ特徴量エンジニアリングと機械学習の基礎
- <https://www.oreilly.co.jp/books/9784873117980/>



8. 学習実行と交差検証(1/4)

データと学習手法を指定します。

実行コード

```
predictors = ["Pclass", "Sex", "Age", "SibSp", "Parch", "Fare", "Embarked"]
```

```
models = []
```

```
models.append(("LogisticRegression", LogisticRegression()))
```

```
models.append(("SVC", SVC()))
```

```
models.append(("LinearSVC", LinearSVC()))
```

```
models.append(("KNeighbors", KNeighborsClassifier()))
```

```
models.append(("DecisionTree", DecisionTreeClassifier()))
```

```
models.append(("RandomForest", RandomForestClassifier()))
```

```
models.append(("MLPClassifier", MLPClassifier(solver='lbfgs',  
random_state=0)))
```

```
# import https://www.kaggle.com/tflare/testing-multiple-models-with-scikit-learn 参照
```

8. 学習実行と交差検証(2/4)

交差検証を実施します。

交差検証ではデータセットを訓練データとテストデータに分割し（ここでは3分割）、それぞれで評価することで精度を安定させる方法です



実行コード

```
results = []
names = []
for name,model in models:
    result = cross_val_score(model, train_data[predictors],
train_data["Survived"], cv=3)
    names.append(name)
    results.append(result)
```


8. 学習実行と交差検証(3/4)

3分割されている結果を平均し、評価を出します。

ランダムフォレストが良い結果を出しました。

LogisticRegression 0.785634118967

SVC 0.687991021324

LinearSVC 0.58810325477

KNeighbors 0.701459034792

DecisionTree 0.766554433221

RandomForest 0.796857463524

MLPClassifier 0.785634118967

```
for i in range(len(names)):
    print(names[i],results[i].mean())
```

実行コード

8. 学習実行と交差検証(4/4)

ランダムフォレストで学習したものを元に、
テストデータで予想をし、その結果をcsvにして送ります。

実行コード

```
alg = RandomForestClassifier()
alg.fit(train_data[predictors], train_data["Survived"])

predictions = alg.predict(test_data[predictors])

submission = pd.DataFrame({
    "PassengerId": test_data["PassengerId"],
    "Survived": predictions
})

submission.to_csv('submission.csv', index=False)
```

9. 最適化(1/7)

正解率0.74163

**7922人中7043位でした。ちょっと悔しい
ので最適化します。**

ランダムフォレストを最適化します。

9. 最適化(3/7)

最適化の説明をします。

grid searchを使用すればハイパーパラメータを自動的に最適化してくれます。

ただし実行時間がかかることに注意して下さい。

そして上記等で最適化した結果を適応してみます。


実行コード

```
parameters = {  
    'n_estimators'      : [5, 10, 20, 30, 50, 100, 300],  
    'max_depth'         : [3, 5, 10, 15, 20, 25, 30, 40, 50, 100]  
    'random_state'      : [0],  
}  
gsc = GridSearchCV(RandomForestClassifier(), parameters,cv=3)  
gsc.fit(train_data[predictors], train_data["Survived"])
```

9. 最適化(2/7)

正解率0.77990

7922人中4129位まで行きました。

4129	▼ 39	tflare		0.77990	3	1d
------	------	--------	---	---------	---	----

Your Best Entry ↑

Your submission scored 0.77990, which is an improvement of your previous score of 0.74163. Great job!



Tweet this!

9. 最適化(4/7)

先程のコードをKaggleに公開するとコメントがつきました。

欠損値を訓練用データからではなくテストデータから求めるようにしたほうが良いとのことでした。やってみました。

コードは次ページ

9. 最適化(5/7)

実行コード

```
def correct_data(train_data, test_data):

    # Make missing values for training data from test data as well
    train_data.Age = train_data.Age.fillna(test_data.Age.median())
    train_data.Fare = train_data.Fare.fillna(test_data.Fare.median())

    test_data.Age = test_data.Age.fillna(test_data.Age.median())
    test_data.Fare = test_data.Fare.fillna(test_data.Fare.median())

    train_data = correct_data_common(train_data)
    test_data = correct_data_common(test_data)

    return train_data, test_data

def correct_data_common(titanic_data):
    titanic_data.Sex = titanic_data.Sex.replace(['male', 'female'], [0, 1])
    titanic_data.Embarked = titanic_data.Embarked.fillna("S")
    titanic_data.Embarked = titanic_data.Embarked.replace(['C', 'S', 'Q'], [0, 1, 2])


    return titanic_data

train_data, test_data = correct_data(train, test)
```

9. 最適化(6/7)

正解率0.79426

7922人中2189位まで行きました。

2189 ▲1899 tflare  0.79425 4 8m

Your Best Entry ↑

Your submission scored 0.79426, which is an improvement of your previous score of 0.77990. Great job!



Tweet this!

9. 最適化(7/7)

- さらなる最適化例えば、名前の解析をする。(Mr. Mrs. Miss 等があるためここから推測できないか等)

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

- 別の学習手法を使用する (例えばXGBoost、LightGBM) を使用して見る

10. 機械学習の流れ(1/4)

1. 実施内容の決定



2. データ入手



3. データ前処理



4. 手法選択



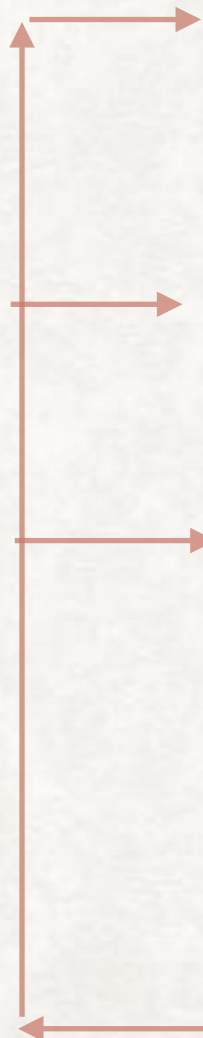
5. ハイパーパラメータ選択



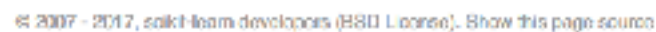
6. モデルの学習



7. モデルの評価



手法選択はアルゴリズム チートシート



10. 機械学習の流れ(3/4)

ハイパーパラメータ選択

grid search

10. 機械学習の流れ(4/4)

モデルの評価

交差検証

11. 企業のコンテストを一部紹介(1/2)

Prudential Life Insurance Assessment

- Can you make buying life insurance easier?
- 生命保険申請者の属性からリスクレベルを算出する
- 賞金3万ドル
- 既に終了済み（コードが参照できる）
- <https://www.kaggle.com/c/prudential-life-insurance-assessment>

11. 企業のコンテストを一部紹介(2/2)

Zillow Prize: Zillow's Home Value Prediction (Zestimate)

- Can you improve the algorithm that changed the world of real estate?
- 自宅のすべての機能を考慮して、Zestimateと実際の販売価格との間の誤差を予測する
- 賞金120万ドル
- 4ヶ月後に終了
- <https://www.kaggle.com/c/zillow-prize-1>

最後に

ご清聴ありがとうございました。