やまぐち
高校生データサイエンティスト育成講座
～day３～

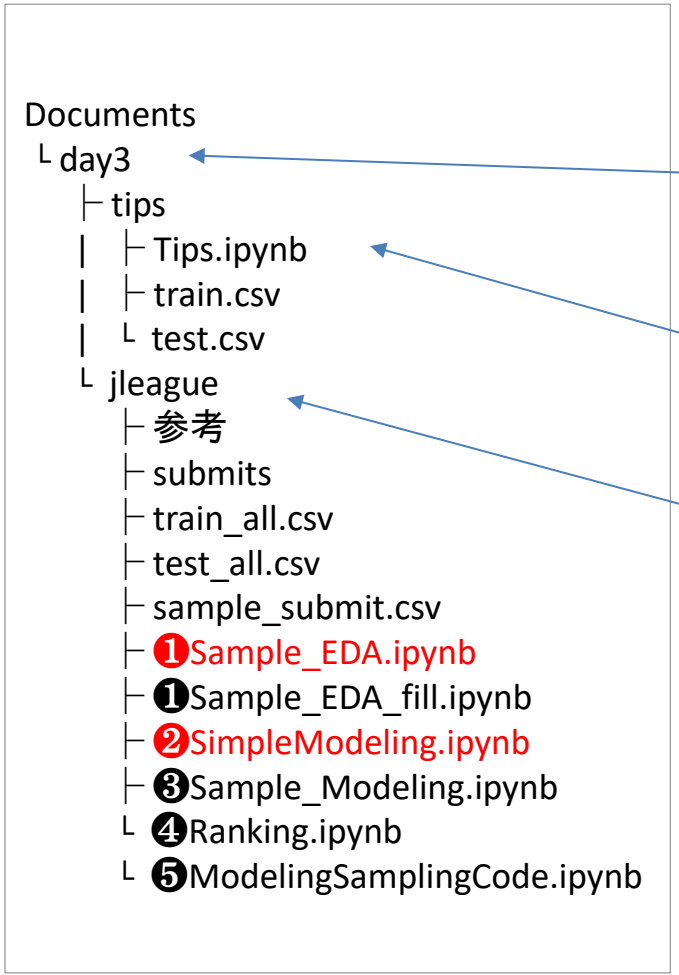
本日のテーマ

COMPETITION

本日のスケジュール

時間	内容
10：10　～　10：35	今回の課題説明
10：35　～　12：00	プログラムハンズオン
12：00　～　13：00	昼休憩
13：00　～　15：30	グループワーク
15：30　～　15：50	グループ発表
15：50　～　16：00	閉講・アンケート

配布データについて



day3.zipを解凍下さい

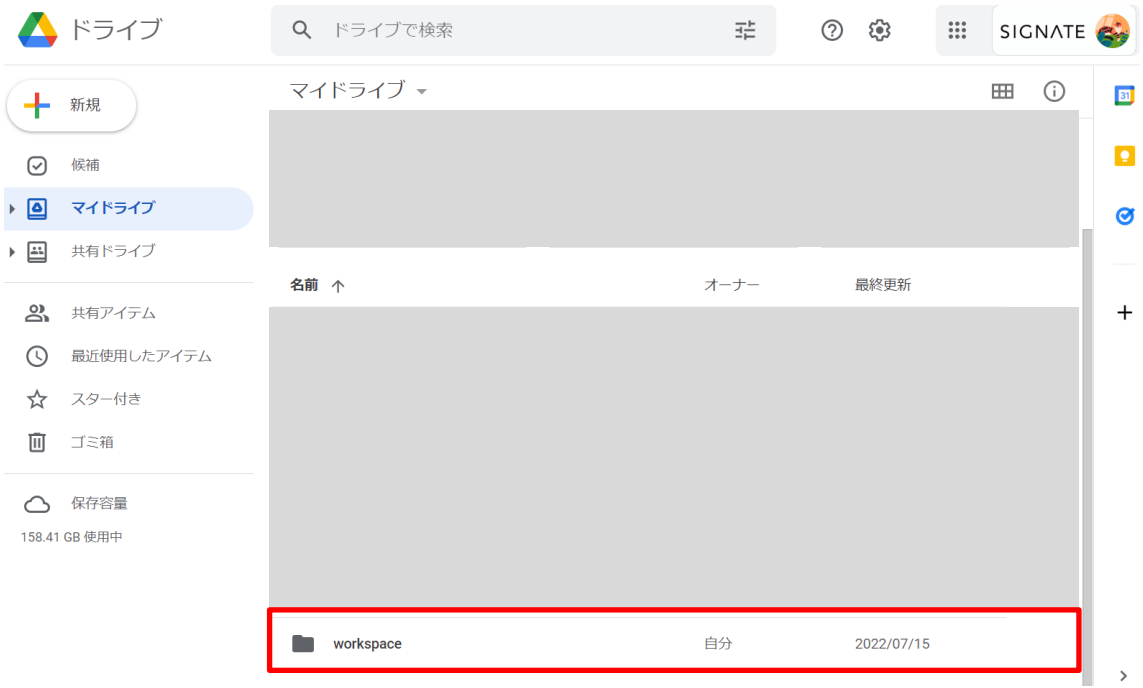
tipsフォルダには３つのファイルがあります
(今回のhandsonでは説明しません)

jleagueフォルダにあるデータがメインデータです

加えて下記の資料が同封されています。
- ハンズオン講座資料（本資料）
- 仮説共有フォーマット
- 中間発表フォーマット
- 最終プレゼン例

Google Colaboratoryの設定①

①Googleドライブを開き、マイドライブ直下に「workspace」というフォルダを作成ください

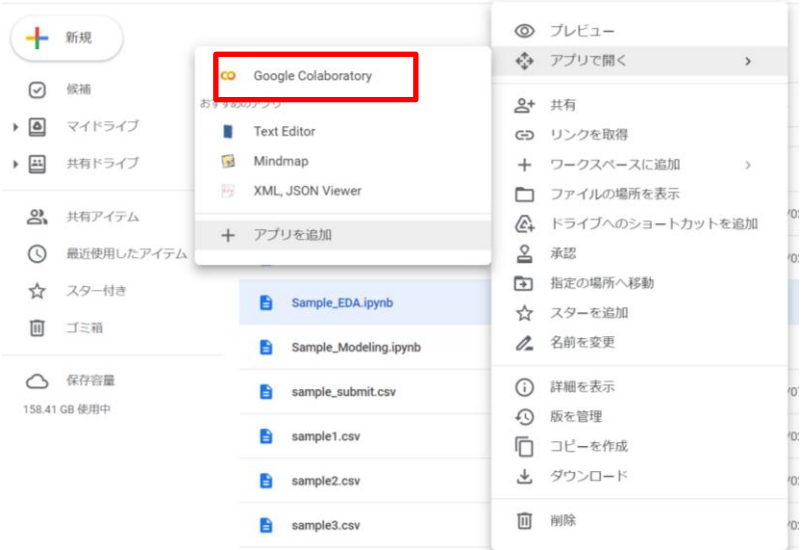


Google Colaboratoryの設定②

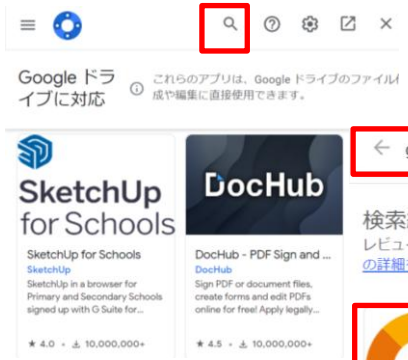
②workspace内に解凍した「day3」フォルダをアップロードしてください

Google Colaboratoryの設定③

③「jleage」フォルダ内の「❶Sample_EDA.ipynb」を右クリックし、”Google Colaboratory”を選択し、notebookを起動します。もし”Google Colaboratory”がない場合には、アプリを追加より、Google Colaboratoryを追加してください。



①検索アイコンをクリック



②以下で検索

③クリックしてインストール

はじめに

分析コンペのメリット

- なにより楽しい
- 分析技術はコンペティションで加速している
- 自身の力量がわかる
- 最近はリクルーティング／人事考課の為のツールとしても活用

【経済産業省主催】
観光客数を予測する
モデルの精度を競う

参加者：128人
分析回数：2789件

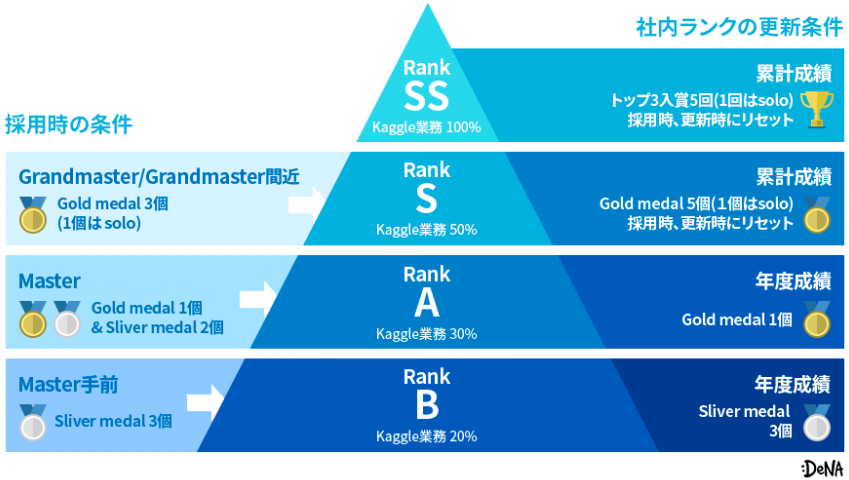
総合部門受賞により
就職活動でとても有利
になりました！

分析の実力が認められ、
会社で分析プロジェクト
責任者に選出されました！

総合部門受賞
Aさん（学生）

個別部門受賞
Bさん（会社員）

▼人事考課として使われるように



データ分析コンペ事例

クレーン旋回操作最適化

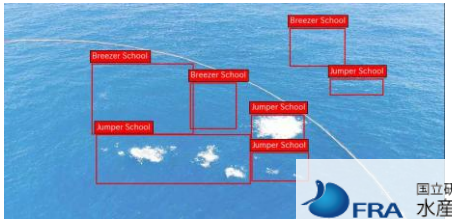
熟練が求められる建設用クレーン車の旋回操作の自動化を目的にシミュレータを利用した最適化アルゴリズム作成にチャレンジ。

TADANO

総賞金：400万円
シミュレータを利用した新ジャンルのコンペ。

魚群探知

時間と労力が大きい魚群探索を目的にドローンによる空撮画像の魚群物体検出の精度を競うコンペを実施。

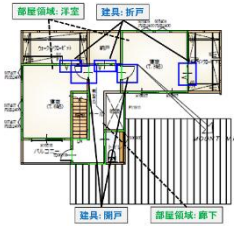


FRA 国立研究開発法人 水産研究・教育機構

間取り図解析

人手による工数が大きい床材やドア部材の量の算出を目的とした間取り図に対する物体検出アルゴリズムの作成。

Panasonic



総賞金：200万円
1400人を超える参加者

ひろしまQuest: プロ野球データを用いた配球予測

広島県のAI人材開発プラットフォーム「ひろしまQuest」企画によるプロ野球の配球予測コンペティション。

HIROSHIMA SANDBOX ひろしまサンドボックス

広島県

列車の運行予測/新幹線台車部の着雪量予測

ユーザビリティ改善を目的とした列車の運行予測や新幹線の安全運行の為の着雪量予測など社会インフラの改善。

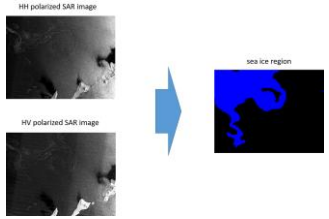
JR西日本　JR東日本

Tellus Satellite Challenge

経産省による衛星データ産業利用の促進を目的としたデータ分析コンテスト。海氷領域の検知や海岸線検出など。

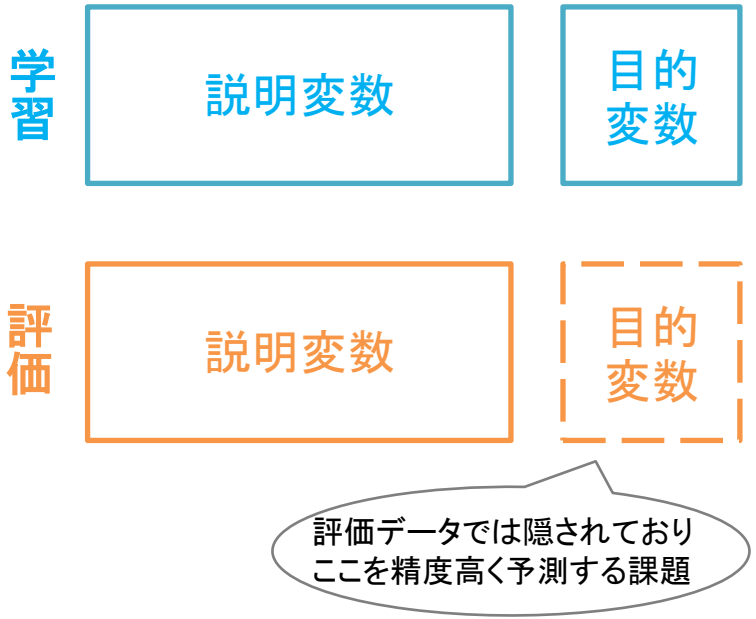
Tellus

経済産業省 Ministry of Economy, Trade and Industry



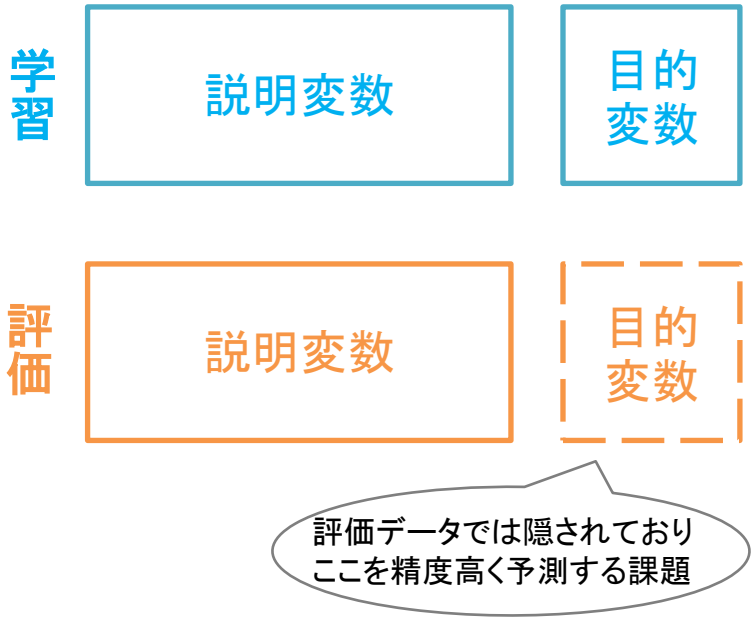
分析コンペティションでは

既にtrainとtestに分割されて配布されます



分析コンペティションでは

既にtrainとtestに分割されて配布されます



【分析のステップ】

①学習データを使い、モデルを作成する

②学習データを使い、良いモデルが作れているかどうかを確認する

③作成したモデルを使い、評価データを予測し、投稿する

Jリーグ観客動員数予測に挑戦！

Jリーグの観客動員数予測

- **お題**
  - 2014年後半のJ1及びJ2全試合の観客動員数を予測いただきます

- **データ概要**
  - 学習：2012～2014年シーズン前半の試合及びそれに関連するデータ
  - 予測：2014年シーズン後半の試合及びそれに関連するデータ

- **問題のポイント**
  - 今回は回帰問題です（数値を当てる問題）
  - データが一部分割されている為、データ結合が必要です
  - 文字列を分割して変数を作る等、データ加工も必要です
  - 目的変数の性質把握や予測に何が寄与するかの仮説設計が重要です

Jリーグの観客動員数予測

- **実習の狙い**
  - E-learning教材で学んだ基本が定着しているかどうかチャレンジ
  - グループで取り組み、モデリングに関する知識や工夫を深めていただく

- **評価関数**
  - RMSE（Root Mean Squared Error）
    - 大きく予測値が外れるとペナルティが大きくなる
    - 大はずれしない予測モデルを作ることがポイント

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(y_i - p_i)^2}$$

N:予測対象数
y_i:i番目の実績値
p_i:i番目の予測値

本コンペへの取り組み方

① **【前処理】**
- データを結合し、分析の準備をする

② **【可視化・アイデア出し】**
- 各説明変数と目的変数の関係を可視化してみる
- どの変数が予測に有効かをグループで話し合い、検討する
- その他、提供データ以外にも使えそうなデータがないかを検討する

③ **【モデリング】**
- グループ内で様々な手法・特徴量を使ってモデルを試してみる
  - 一人最低１つのモデルを作りましょう
- 作ったモデルは学習データ内での予測精度を検証し、良さそうかどうかを確認してみる
- 単体モデルだけでなく、アンサンブル手法も試してみる

④ **【投稿】**
- SIGNATEに予測結果を投稿する
  - １日10回まで

⑤ ②～④を繰り返し、グループ毎に予測精度を競っていただく

基本データ

- **モデル学習用データ（train.csv）**
  - J1,J2の2012～2014年のシーズン前半（7/31）
  - y（目的変数）となる各対戦カードの観客動員数も含む
- **モデル検証用データ（test.csv）**
  - J1,J2の2014年シーズン後半（8/2～11/23）
  - yは含まれていない
  - 今回の問題ではこのデータ中の対戦カードのみを予測
- **スタジアムデータ（stadium.csv）**
  - 各スタジアムの所在地と収容人数（※結構重要です）
- **試合詳細データ（condition.csv）**
  - 各対戦カードのスコアやスタメン、天候等
- **応募用サンプルファイル（sample_submit.csv）**
  - 対戦カードid，そのidの観客動員数の予測値

一部データに
抜けがあります

e.g)
第１試合　マリノスvsアントラーズ
第２試合　マリノスvsエスパルス
第４試合　マリノスvsガンバ

追加データ

- **モデル学習用試合追加データ（train_add.csv）**
  - 2012年86件、2013年93件、2014年53件の追加対戦データ（全232件）
  - モデルの学習に使っていただいて構いません

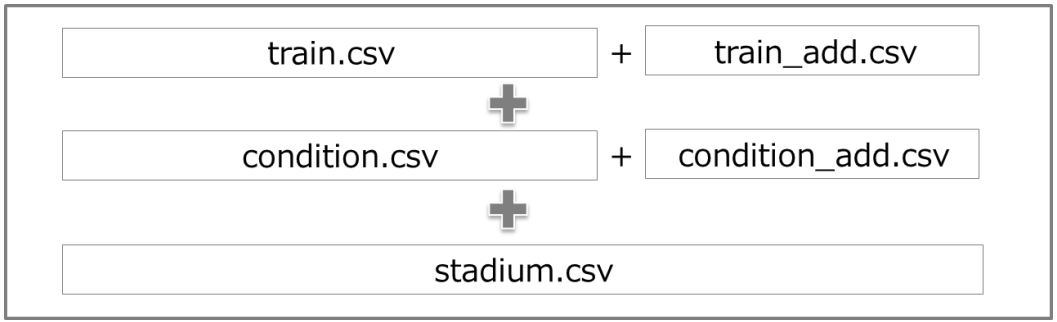
- **2014年度後半試合追加データ（2014_add.csv）**
  - 2014年後半の38件の対戦データ
  - あくまで補足データとなります
  - 得点が含まれているので、モデル検証用データと組み合わせることで、その対戦時にチームが現在何位なのかを割り出すことができます
    - 順位等を説明変数として加えたい時に利用してください

- **試合詳細追加データ（condition_add.csv）**
  - condition.csvの中で不足している試合の詳細データ（全270件）

2014年度後半試合追加データはあくまで補足データです。
応募時にはこのデータの対戦idの予測値は追加しないでください。

結局どうすればよいのか？

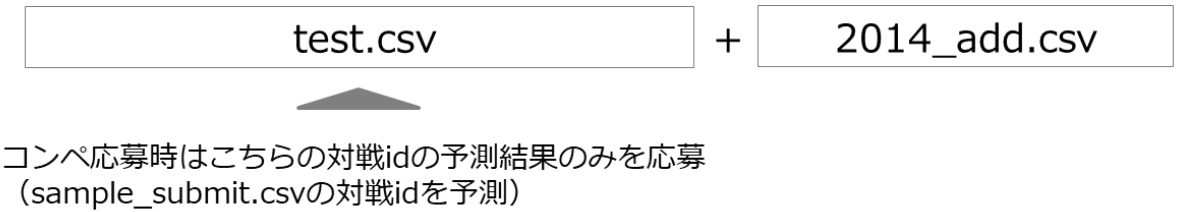
- **学習期間用の全データを作るには…**



- **2014年度後半試合追加データは何に使うの？**

  -> 順位を表す説明変数を作りたい時に利用します

  ※追加の試合データはコンペでの予測対象ではないので注意してください



精度向上のアイデア①

- **特徴量を作ってみる**
  - データを加工することで予測に寄与しそうな新しい変数を作成

【チーム・スタジアム】
・所在地は関係があるか？
・対戦カードで変わるのか？

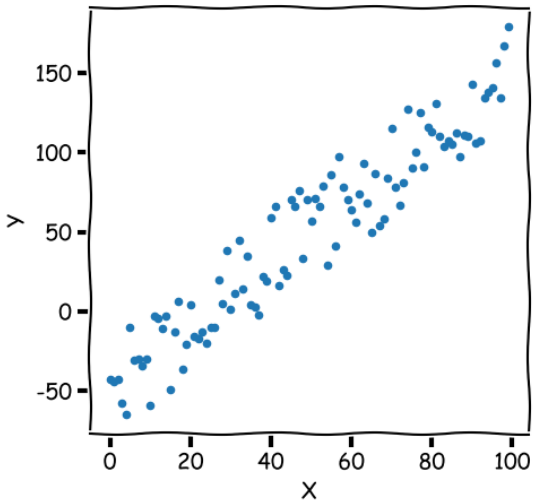
【TV】
・動員しやすいチャンネルはあるか？
・放送チャンネルの数はどうか？

【選手・審判】
・出場選手・審判は関係があるか？
・有名な選手は影響を及ぼすか？

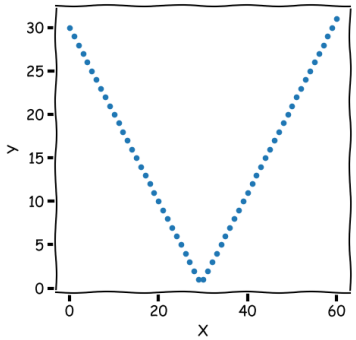
【時間・天気】
・動員しやすい日程・時間はあるのか？
・天気は動員数に関係があるのか？

精度向上のアイデア②

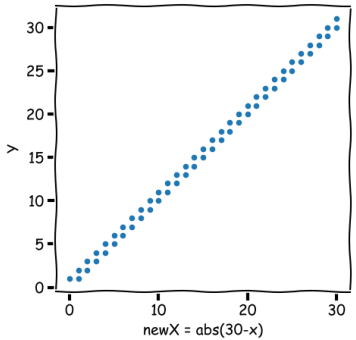
- 線形性を見つける、もしくは線形な関係に変換する



xとyの相関が高そう
→予測に寄与する変数である可能性



xとyが線形な関係に
なるように加工

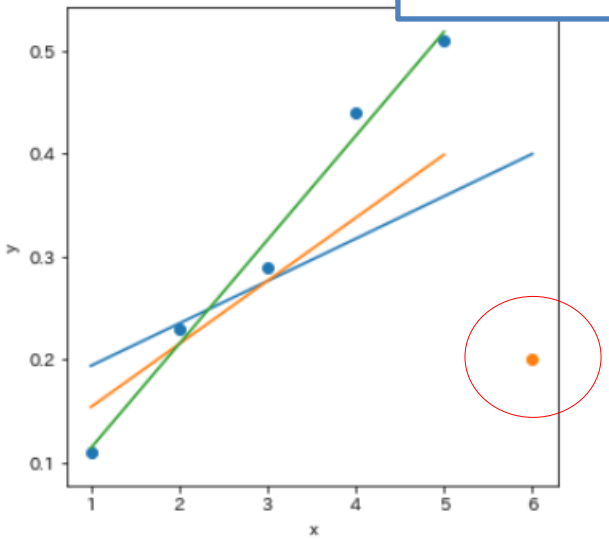


線形回帰モデルを利用する場合

精度向上のアイデア③

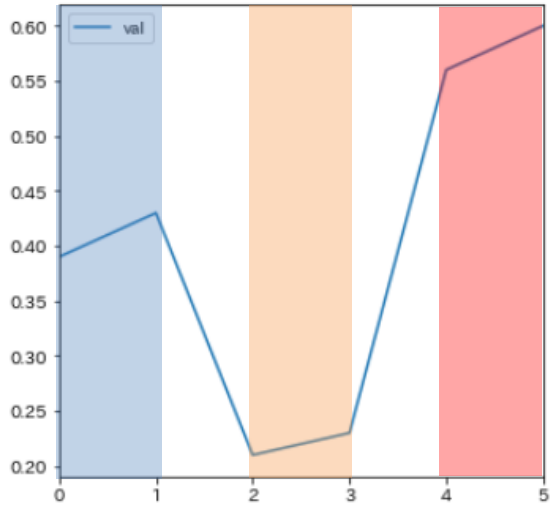
外れ値を外す、丸める

青：未加工
緑：if x>5 then x=5
緑：除去



外れ値？

量的データを質的データに変換



線形性が見えずらい変数についてはビニングによってカテゴリ変数化して扱うのも有効

こちらも参考になるかと思います。
scikit-learn cheat-sheet（http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html）

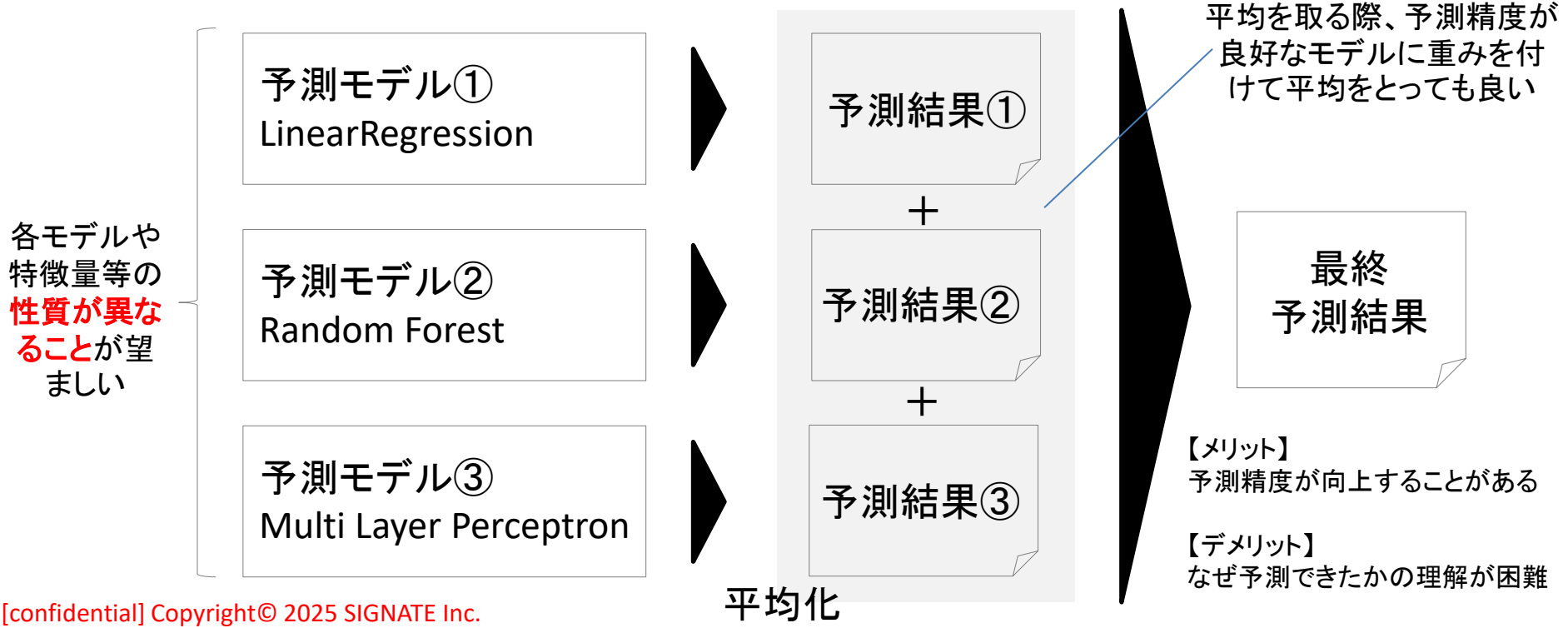
精度向上のアイデア④

- **色々なアルゴリズムを使ってみる**
  - 多くの場合、どのアルゴリズムが適切か自明でない為、実験が必要

1. Lasso回帰（http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html）
2. Ridge回帰（http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html）
3. ElasticNet（http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html）
4. サポートベクター回帰（http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html）
5. Multi Layer Perceptron（https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html）
6. RandomForest（http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html）
7. GradientBoosting（http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html）
8. Xgboost（https://github.com/dmlc/xgboost　注：ライブラリのインストールが必要）
9. Catboost（https://github.com/catboost/catboost　注：ライブラリのインストールが必要）

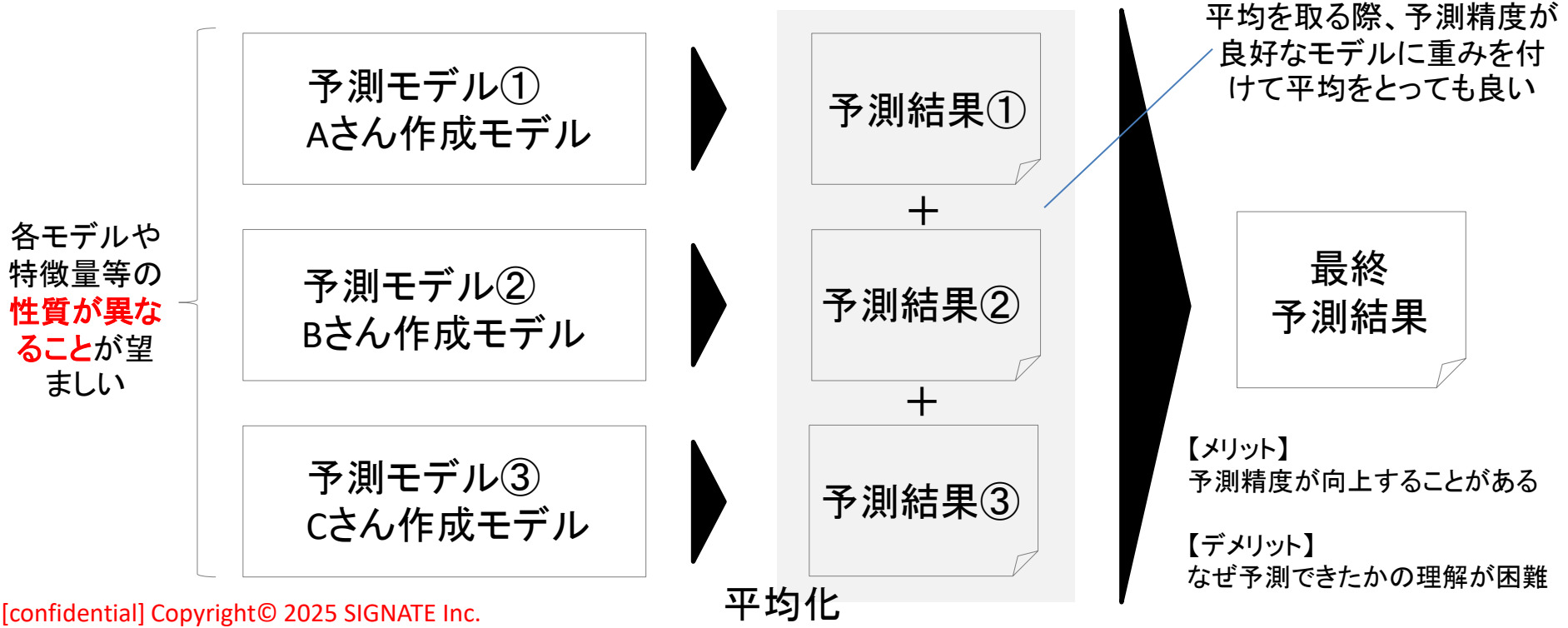
精度向上のアイデア⑤

- **複数の予測モデルの結果を組み合わせる（アンサンブル）**
  - ここでは単純に予測結果の平均をとることを考える



精度向上のアイデア⑤'

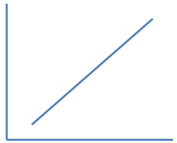
- **グループ内でメンバーそれぞれが作成したモデルを組み合わせる**
  - コンペでは精度向上の為によく実施される



精度向上のアイデア⑥

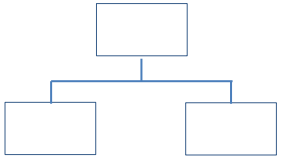
- **モデルの特性を考えてみる**

線形回帰モデル



- ✓ 単調性（xが増えればyも増）により、学習データ内の目的変数の値域を超えて予測できる
- ✓ 交互作用（AかつB等の複合的な条件）がモデル内に内包されていない為、特徴量を作る必要がある
- ✓ 相関が互いに強い説明変数を入れると係数が不安定（多重共線性）

決定木モデル



- ✓ あくまで学習データを基準に目的変数の値域が決まる為、学習データの値域を超えた予測はできない
- ✓ 交互作用がモデル内に内包されている為、交互作用に関する特徴量を作る必要がない
- ✓ 多重共線性をあまり考慮しなくてよい

課題

- グループに分かれて分析コンペティションに挑戦いただきます
- グループで協力し、下記課題に挑戦下さい
  A) 精度の高い予測モデルを作成する
  B) コンペ配布データから得られるインサイトを分析する

- 講座５日目に上記課題に関する発表をしていただきます
  ① コンペの解法を発表する
  ② データ分析で得られたインサイトを発表する

今日の課題

① データから気づいたことを10個以上列挙する

② どういう時で観客が増加or減少しそうかを8個以上列挙する

③ グループで話し合いをし、仮説を3つ選ぶ

④ 選んだ仮説を元にモデリングをし、コンペに投稿する

グループワークの進め方

#	目安時間	内容
1	5	グループ内で自己紹介をしましょう。
2	5	チーム名と意気込みと、役割（リーダーとタイムキーパー）を決めましょう
3	15	各自でデータを確認し、気になるポイントを調査しましょう
4	15	グループで気づいたことを出し合い、仮説共有フォーマットにまとめましょう
5	20	出てきた気づきを元に、話し合いで仮説を出し合い、仮説共有フォーマットにまとめましょう
6	15	出てきた仮説から有力そうなものを３つ選び、Teamsに投稿し、OKをもらいましょう
7	45	各自で選んだ仮説を元にモデリングをし、コンペに投稿しましょう。先に投稿できた人は他のグループメンバー全員が投稿できるように助けましょう。
8	20	次に試してみる仮説を話し合い、試してみる優先順位をつけましょう。また誰がどの仮説を試してみるかの役割分担までしましょう。
9	10	次回までの間に作業内容を共有し合うwebMTGをする日程を1日決めましょう。

本日の発表

- 下記について発表いただきます（仮説共有フォーマット.pptxがテンプレートです）
  - チーム名
  - 意気込み
  - データから気づいたこと
  - 観客動員数予測に寄与すると考えられる仮説と選んだ３つ
  - 投稿結果のスコア

課題A)　精度の高い予測モデルを作成する

- 分析コンペに挑戦し、精度が高い予測モデルを作成いただきます
- 配布しているnotebookをぜひ活用ください
  - Sample_EDA.ipynb
  - Preprocessing.ipynb
  - Sample_modeling.ipynb
- データは細かく分割されているため、本格的にモデリングをする際にはjupyter notebook上でデータをマージしてから、データ分析を始める必要があります
- いきなりプログラミングは始めずに、まずは何をすべきかの手順や仮説を話しあい、その後は配布データのサンプルコードの確認などからはじめることをお薦めします
- サンプルコードをコピペしながらプログラムを組んでいくのも良いでしょう
- チームで話し合った観客動員数の増減に寄与しそうな仮説・アイディアを反映した特徴量をデータから作成し、予測モデルを作成しましょう。また得られたインサイトについてグループで情報共有しながら進めましょう
- プログラミングが苦手なメンバーとどう連携すべきかも考えながら課題を進めましょう
- 配布データ以外の情報を利用しても構いません
- RMSEを改善するにはよくデータを考察する必要があります。RMSEが3000を下回ることを目標に取り組んでください
- 講座５日目の最終レポートについては、最終レポート例を参考にしながら自由に作成いただき、発表ください

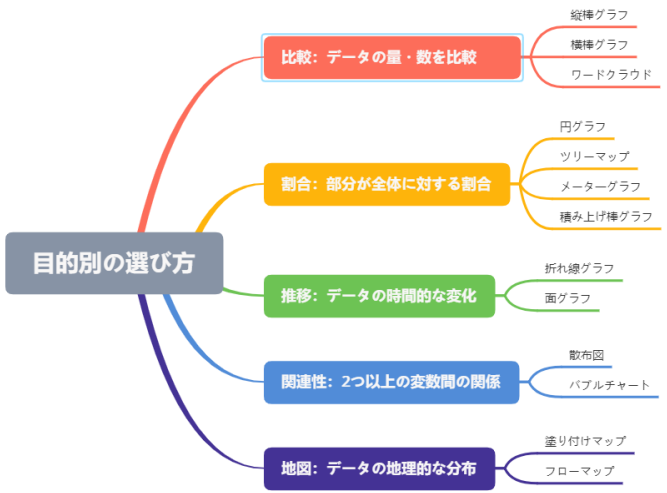
課題B)　データ分析で得られたインサイトを分析する

- 実際のデータを集計したり、分析をしながら、データから読み取れるインサイトを分析してもらいます
- Excelを使ってデータを眺めてみるのも良いでしょう
- グループで話し合った仮説・アイディアを元に、まずは分析テーマの候補を考えてみるところから始めてみましょう
  - 観客動員数が多いチームの共通点とは？
  - 私の好きなチーム〇〇はこんな特徴があります
  - 観客動員数が多いと本当に儲かっている？

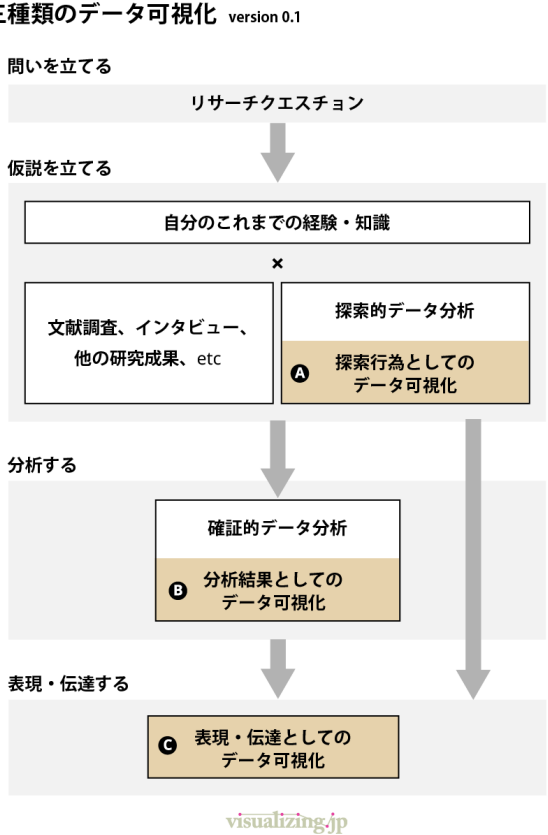
- 分析テーマが決まりましたら、実際にその仮説やアイディアをデータや分析手法を利用し、証明・説明ください。最低限必要な項目は下記の通りです
  - 分析テーマ
  - 分析方法・可視化
  - 結論
- 配布データ以外の情報を利用しても構いません
- 発表資料では表やグラフを必ず用いて、得られたインサイトを説明ください

インサイトのアウトプット例

- 仮説を立てる・証明する・補強する
- 比較をする・差をみる・差を見つける
- グラフを描く・地図にマッピングする・時系列でみる
- シミュレーションする・モデルで予測する



https://www.finereport.com/jp/analysis/datavisualgraph/



https://visualizing.jp/viz-for-whom/

次回について

次回について

次回、中間発表があります！

次回の宿題

①モデルを改良してコンペページに投稿

・サンプルコードをベースにして、モデルを作成し、コンペページに投稿してください。（最低2回以上）

・投稿した内容については下記事項をまとめてteamsのグループチャンネルに投稿してください。

- 投稿スコア
- 何をしたか？（モデル／特徴量など）

②中間発表の準備

次回は中間発表がありますので、事前にグループで打ち合わせをして当日慌てないように準備を進めてください。（1回以上はweb mtgを実施してください）

Day４までのお願い

- Day4に中間発表をして頂きます
- 発表内容は課題Aのみです（課題Bは最終発表のみ）
- 中間発表フォーマット.pptxに従い、チーム毎に記入の上、発表をお願い致します
- 項目は下記の通りです
  - チーム名
  - 最高スコア（SIGNATE投稿スコア）
  - 分析方針（チームにおけるモデル作成・取り組み方の方針）
  - 分析手順（どんな作業をどんな手順で実施しているか？）
  - 採用モデル（モデル手法）
  - 出された仮説（現在挙げられている予測に寄与すると考えられる仮説）
  - 悩み・疑問点

おしまい

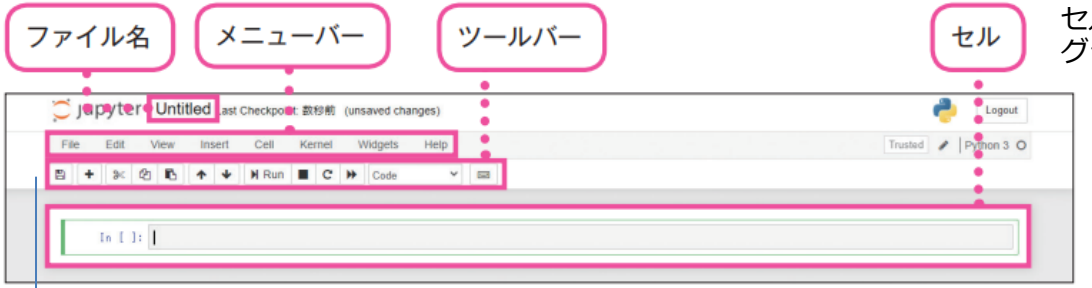
Let's Enjoy DataScience!!!

APPENDIX

Jupyter Notebookの使い方

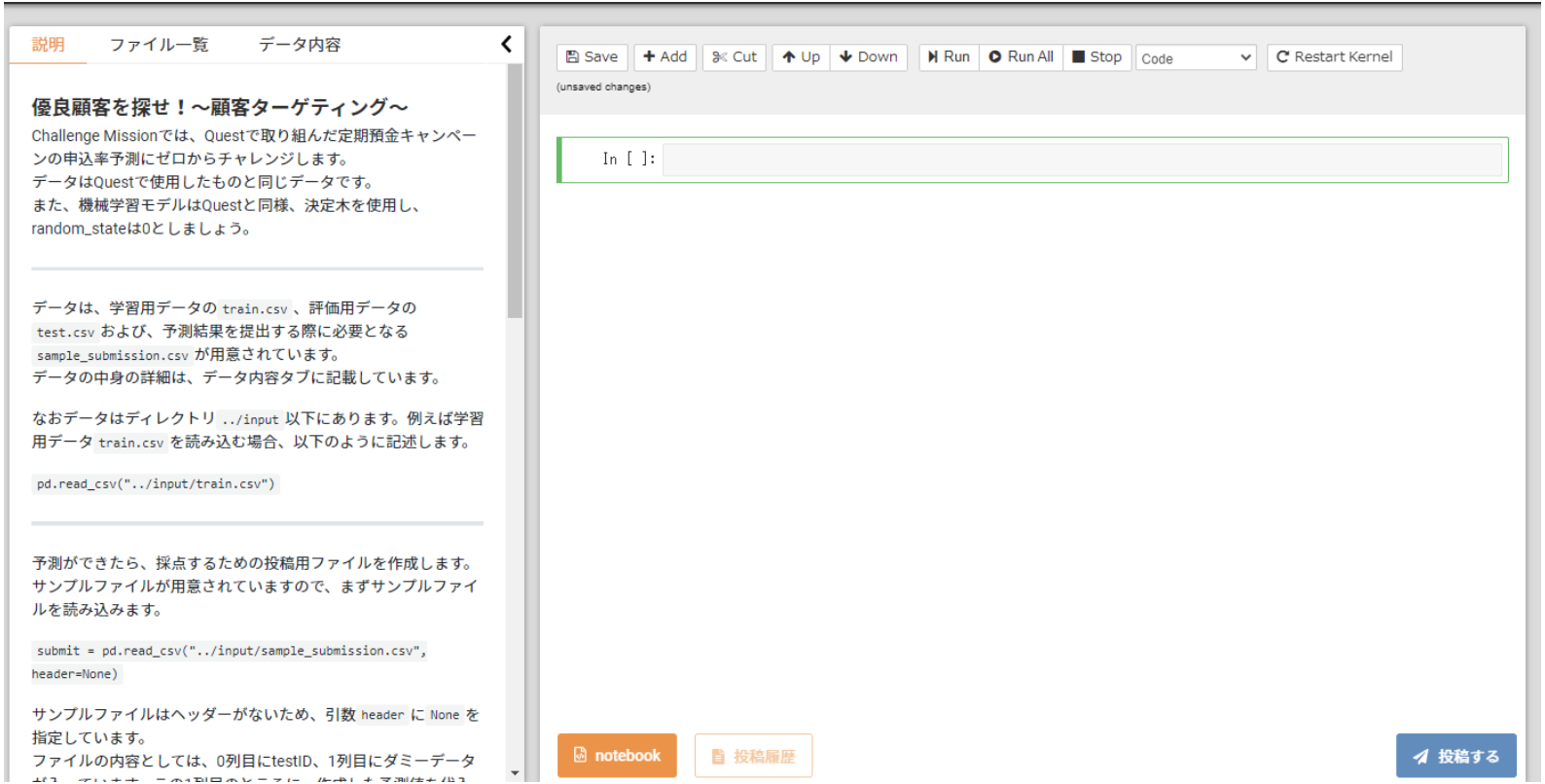
Jupyter Notebook

NotebookのUIは次のようになっています。



よく使うツールバーのアイコン

Challenge MissionのUI

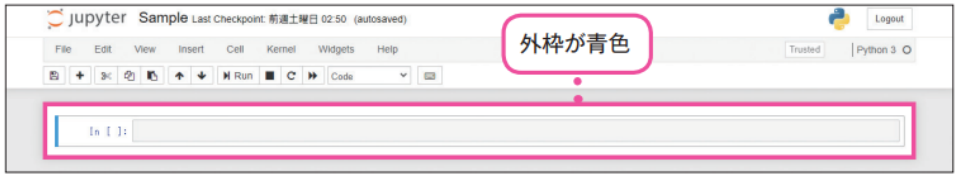


Jupyter Notebook

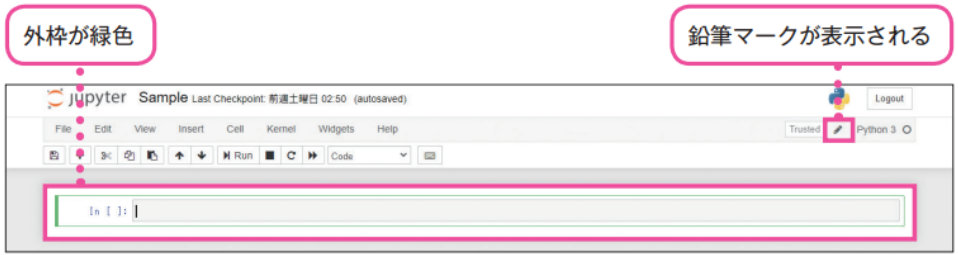
Notebookを利用するうえで重要になるのは「セル」です。
In[]の右にある入力欄を「セル」と呼び、このセルには「コマンドモード」と「編集モード」というモードが2種類あります。

「コマンドモード」・・・セルを増やしたり減らしたり、操作するときのモード。外枠が青色。
「編集モード」・・・プログラムを入力するときのモード。外枠が緑色＋鉛筆マークが表示。

●コマンドモード

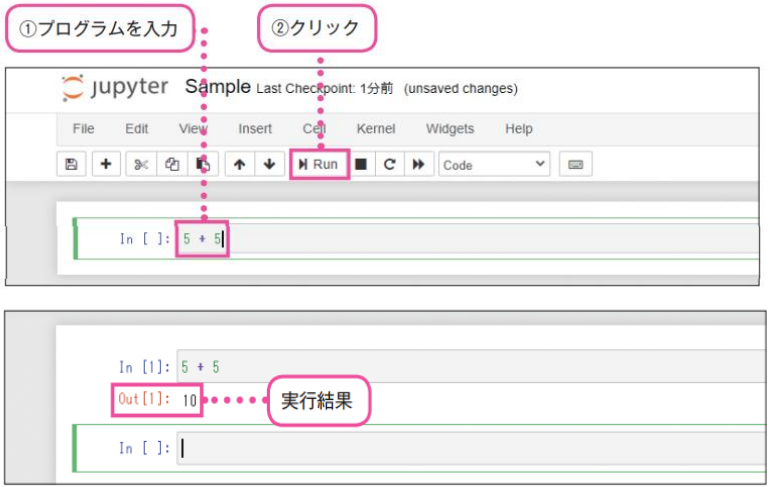


●編集モード



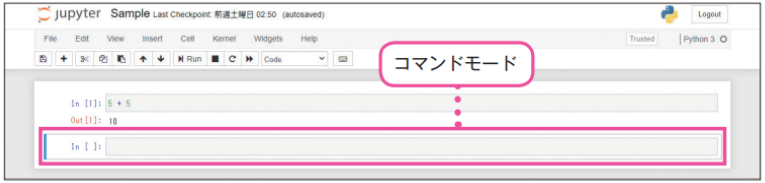
Jupyter Notebook

実際に動作の確認をしてみます。
セルをクリックすると、カーソルが表示され「編集モード」に切り替わります。
セル内に「5+5」を入力し、プログラムを実行してみてください。
プログラムの実行はツールバーの「Run」アイコンか、Shiftキーを押しながらEnterキーを押します。
後者の実行のショートカットキーは頻繁に利用するので、覚えておくと良いでしょう。



プログラムを実行すると、Out[1]の右側に実行結果が表示される。In[1]やOut[1]の数字はプログラムが何回目に実行されたかの履歴番号を表す。

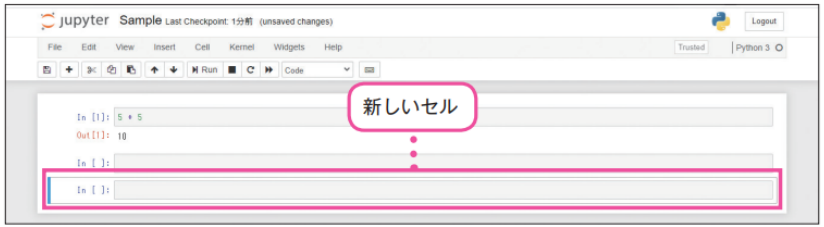
実行後、自動的に新規セルが作成されます。
この状態で入力欄以外をクリックするか、Escキーを押すと、コマンドモードに切り替わります



Jupyter Notebook

コマンドモードの状態ではセルを増やしたり減らしたりすることができます。
「B」キーを1回押すとセルが増え、「D」キーを2回押すとセルが削除されます。
「H」キーを押すと、ショートカットキーの一覧が表示されます。

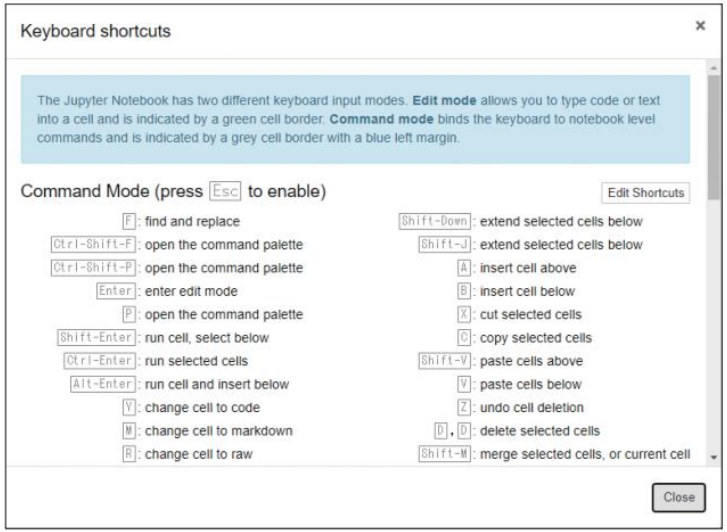
●図3-14　Bキーによる新規セルの作成



今度はDキーを2回押すと、セルが1つ削除されます。

●図3-15　Dキーの2回押しによるセルの削除

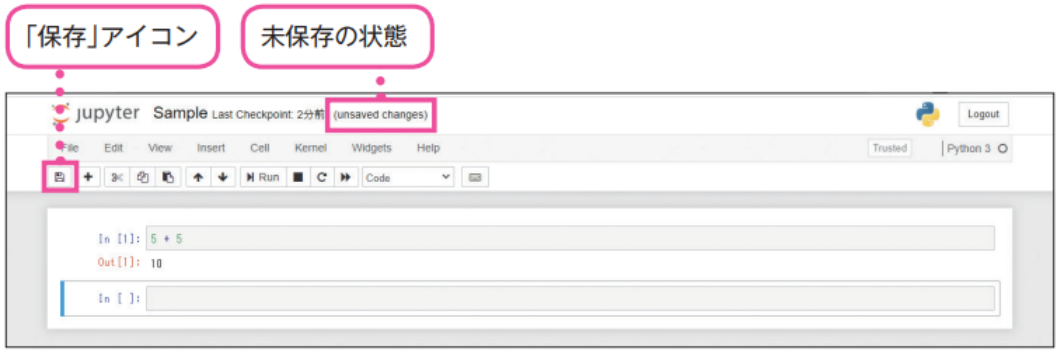
●図3-16　Hキーで表示されるショートカットキーの一覧



Jupyter Notebook

Notebookファイルの保存は、「保存」アイコンをクリックします。
Notebookには自動保存機能もあるため、画面の上部に「autosaved」が表示されていれば手動保存は必要ありません。
未保存の場合には、「unsaved changes」と表示されますので、この時は手動保存しましょう。

なお、ブラウザとターミナルを終了すればJupyter Notebookは終了します。
Notebookファイルはプログラムのソースコードと実行結果については保存されますが、**一度終了したnotebookを再開したいときには、最初からプログラムを実行しなおす必要があります**。つまり、Notebookを終了すると、プログラムのソースコードや実行結果は残っていますが、ライブラリの宣言や変数の値はすべてリセットされてますので、再度の実行が必要です。その点についてよく注意しましょう。



Jupyter Notebook

覚えてほしいショートカットキー

ショートカット内容	利用できるモード	ショートカットキー
セルを実行する	編集モード	Shift + Enter
編集モードからコマンドモードに切り替える	編集モード	ESC
コマンドモードから編集モードに切り替える	コマンドモード	Enter
セルを増やす	コマンドモード	B
セルを削除する	コマンドモード	D（2回押し）
削除したセルを元に戻す	コマンドモード	Z
ショートカットキー一覧を表示する	コマンドモード	H

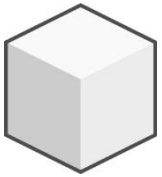
データ前処理に役立つ関数

データ前処理に役立つ関数

- split関数
  - 文字列を分割する関数
- apply関数
  - データの各値に数式や関数を適用する関数
- concat関数
  - ２つのデータを単純結合する
- merge関数
  - ２つのデータをkeyを元に結合する

split関数

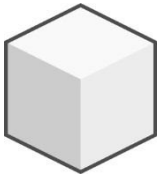
- 文字列を指定した文字で分割する関数



.　split（”区切り文字”）

split関数

- 文字列を指定した文字で分割する関数



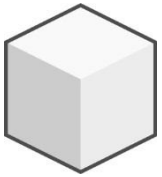
. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

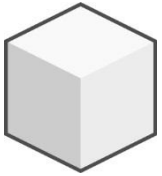
terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

⇒colors = terms.split(“,”)

カンマ

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

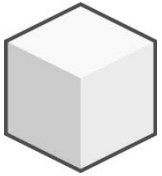
⇒colors = terms.split(“,”)

★colorsの中身を取り出したい

[“red”, “blue”, “green”, “yellow”]

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

⇒colors = terms.split(“,”)

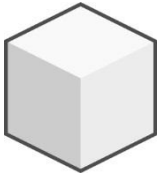
★colorsの中身を取り出したい

[“red”, “blue”, “green”, “yellow”]

0 1 2 3

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

⇒colors = terms.split(“,”)

★colorsの中身を取り出したい

[“red”, “blue”, “green”, “yellow”]

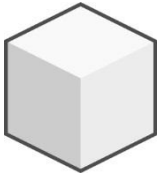
0 1 2 3

redを取り出したい

yellowを取り出したい

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

⇒colors = terms.split(“,”)

★colorsの中身を取り出したい

[“red”, “blue”, “green”, “yellow”]

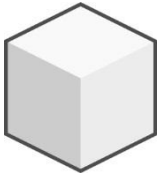
0 1 2 3

redを取り出したい⇒colors[**0**]

yellowを取り出したい

split関数

- 文字列を指定した文字で分割する関数



. split(”区切り文字”)

<E.g>

terms = “red,blue,green,yellow”を**カンマ**で区切り、結果を変数colorsに代入したい

⇒colors = terms.split(“,”)

★colorsの中身を取り出したい

[“red”, “blue”, “green”, “yellow”]

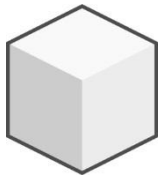
0 1 2 3

redを取り出したい⇒colors[**0**]

yellowを取り出したい⇒colors[**3**]

apply関数

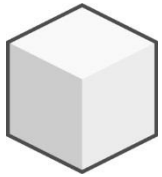
- 各値に数式や関数を適用する為の関数（pandas）



. apply（lambda x : 〇〇）

apply関数

- 各値に数式や関数を適用する為の関数（pandas）



.　apply（lambda x : ○○）

数式 or 関数

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat[“value”].apply(lambda **x** : x*2+1)

dat =

Index	value
1	4
2	7
3	3
4	8

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat[“value”].apply(lambda x : x*2+1)

4*2+1

dat =

Index	value
1	4
2	7
3	3
4	8

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

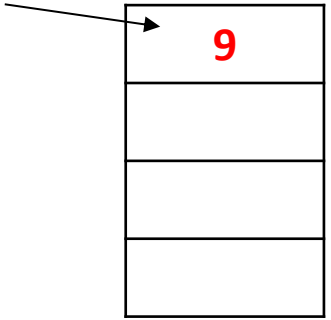
<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

4*2+1



apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat["value"].apply(lambda **x** : x*2+1)

dat =

Index	value
1	4
2	7
3	3
4	8

9

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat["value"].apply(lambda x : x*2+1)

dat =

Index	value
1	4
2	7
3	3
4	8

7*2+1

9

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

7*2+1

9
15

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat["value"].apply(lambda **x** : x*2+1)

dat =

Index	value
1	4
2	7
3	3
4	8

9
15

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

3*2+1

9
15

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

3*2+1

9
15
7

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

dat["value"].apply(lambda **x** : x*2+1)

dat =

Index	value
1	4
2	7
3	3
4	8

9
15
7

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

8*2+1

dat =

Index	value
1	4
2	7
3	3
4	8

9
15
7

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

8*2+1

9
15
7
17

apply関数

- 各値に数式や関数を適用する為の関数（pandas）

<E.g> datのvalueの各値を2倍して1を足したい

```
dat["value"].apply(lambda x : x*2+1)
```

dat =

Index	value
1	4
2	7
3	3
4	8

各値に「x*2+1」という
数式が適用された

9
15
7
17

concat関数

- データフレームを縦に結合する

[ ]が必要なので注意。良く忘れます

```
dfnew = pd.concat([A,B],sort=False)
```

データフレームA

	val1	val2
0	A1	B1
1	A2	B2
2	A3	B3
3	A4	B4

+

データフレームB

	val1	val2
4	A5	B5
5	A6	B6
6	A7	B7
7	A8	B8

dfnew

	val1	val2
0	A1	B1
1	A2	B2
2	A3	B3
3	A4	B4
4	A5	B5
5	A6	B6
6	A7	B7
7	A8	B8

merge関数

- キーをヒントとして結合する

```
dfnew = pd.merge(A,B,on="id")
```

ヒントとなるカラムをオプションで設定

データフレームA

	val1	id
0	JP	01
1	US	02
2	CN	03
3	GB	04

+

データフレームB

	id	val2
0	04	イギリス
1	03	中国
2	02	アメリカ
3	01	日本

dfnew

	val1	id	val2
0	JP	01	日本
1	US	02	アメリカ
2	CN	03	中国
3	GB	04	イギリス

カラムidをヒントに結合する

カラムidで同一の値があるもの同士が横に結合される

ヒントとするキーのカラム名が異なる場合

dfnew = pd.merge（A,B,left_on=“X”,right_on=“Y”）

左のデータフレームはXを、右はYをヒントで結合する

データフレームA

	val1	X
0	JP	01
1	US	02
2	CN	03
3	GB	04

+

データフレームB

	Y	val2
0	04	イギリス
1	03	中国
2	02	アメリカ
3	01	日本

dfnew

	val1	X	Y	val2
0	JP	01	01	日本
1	US	02	02	アメリカ
2	CN	03	03	中国
3	GB	04	04	イギリス

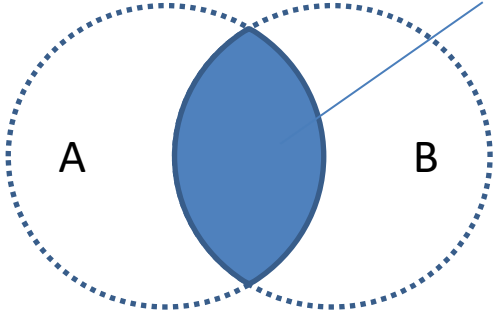
名前が異なる場合は結合時、
どちらのカラムも残る

もっと詳しく知りたい方は
http://pandas.pydata.org/pandas-docs/stable/generated/pandas.concat.html
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.merge.html
http://sinhrks.hatenablog.com/entry/2015/01/28/073327

dfnew = pd.merge（A,B,on=“X”,how=“?”）

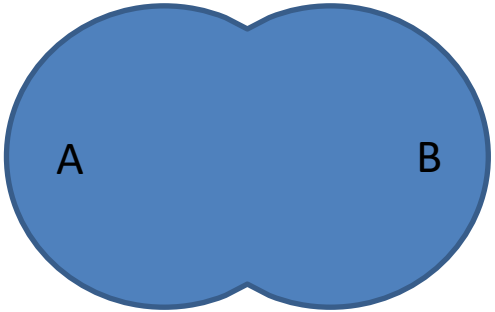
mergeのオプション

▼inner（デフォルト）

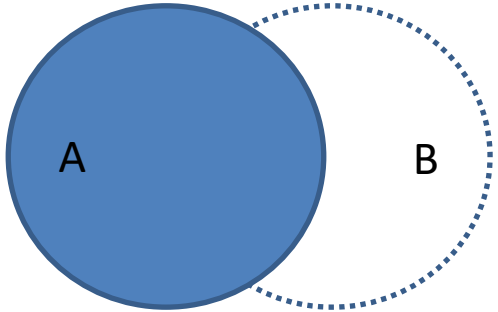


それぞれのDFのキーの集合を考えた場合、オプションにより結合方法が変わる

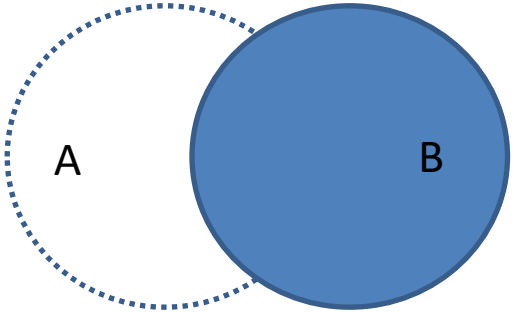
▼outer



▼left



▼right



Inner Join

dfnew = pd.merge（A,B,on=“id”,how=“**inner**”）

データフレームA

	val1	id
0	JP	01
1	US	02
2	CN	03
3	ES	06

+

データフレームB

	id	val2
0	05	韓国
1	03	中国
2	02	アメリカ
3	01	日本

=

dfnew

	id	val1	val2
0	01	JP	日本
1	02	US	アメリカ
2	03	CN	中国

両方のデータフレームに一致するidのみを結合

Outer Join

dfnew = pd.merge（A,B,on=“id”,how=“**outer**”）

データフレームA

	val1	id
0	JP	01
1	US	02
2	CN	03
3	ES	06

+

データフレームB

	id	val2
0	05	韓国
1	03	中国
2	02	アメリカ
3	01	日本

=

dfnew

	id	val1	val2
0	01	JP	日本
1	02	US	アメリカ
2	03	CN	中国
3	06	ES	NaN
4	05	NaN	韓国

両方のデータフレームに一致するidがない
行についても結合を行う

対応する値が存在しない為、
NaN（欠損）となる

Left Join

dfnew = pd.merge（A,B,on=“id”,how=“**left**”）

データフレームA

	val1	id
0	JP	01
1	US	02
2	CN	03
3	ES	06

+

データフレームB

	id	val2
0	05	韓国
1	03	中国
2	02	アメリカ
3	01	日本

=

dfnew

	id	val1	val2
0	01	JP	日本
1	02	US	アメリカ
2	03	CN	中国
3	06	ES	**NaN**

左のデータフレームは全て残した状態で、
右のデータフレームと一致するidがある行のみ結合する

id=06は右のデータフレームBには存在しない為、
val2の列値はNaN（欠損）となる

Right Join

dfnew = pd.merge（A,B,on=“id”,how=“**right**”）

データフレームA

	val1	id
0	JP	01
1	US	02
2	CN	03
3	ES	06

+

データフレームB

	id	val2
0	05	韓国
1	03	中国
2	02	アメリカ
3	01	日本

=

dfnew

	id	val1	val2
0	01	JP	日本
1	02	US	アメリカ
2	03	CN	中国
3	05	**NaN**	韓国

右のデータフレームは全て残した状態で、
左のデータフレームと一致するidがある行のみ結合する

id=05は左のデータフレームAには存在しない為、
val1の列値はNaN（欠損）となる