

# A I 基礎セミナー

## 第 1 回 カリキュラムと環境作成

### 改訂履歴

日付	担当者	内容
2021/05/08	M. Takeda	Git 公開
2021/06/14	M. Takeda	「(3) 開発環境の作成 (Anaconda)」を「2021/06」時点のもので更新

### 目次

- (1) はじめに
- (2) カリキュラム
  - (2.1) セミナーの目標
  - (2.2) セミナーのカリキュラム
  - (2.3) 機械学習ライブラリ
  - (2.4) 言語
  - (2.5) OS
  - (2.6) 教材
- (3) 開発環境の作成 (Anaconda)
  - (3.1) インストール対象物
  - (3.2) インストール
- (4) 開発環境の作成 (Anaconda以外)
  - (4.1) Google Colaboratory
  - (4.2) Amazon SageMaker
- (参考) pip と conda
  - (参考.1) conda と pip について

## (1) はじめに

- ・ 第 1 回では、カリキュラムと環境作成について解説します。
- ・ 第 2 回以降に掲載した Pythonコードは全て JupyterNotebook で動作確認済です。
- ・ Pythonコードのローカルの開発環境として、「Anaconda」を紹介します。  
サーバ上で公開している開発環境として、Google が提供している「Google Colaboratory」を紹介します。  
後者は、参照するライブラリのバージョンの齟齬などを気にすることなく開発でき、  
実装の検証がお手軽にできます。
- ・ A I サービスは様々な企業がクラウドで提供しています。  
Amazon が提供しているAWS(Amazon Web Services) もその一つで、本セミナーでは少しだけ触れます。  
こうした環境では自作の A I サービスを開発・公開するだけでなく、  
出来合いの A I サービスも利用可能で、それほど作りこみをしなくても利用できるようになっています。

## (2) カリキュラム

### (2.1) セミナーの目標

- ・本セミナーでは、以下の2点を目標とします：
  - (a) 機械学習の基礎的な理論が理解できるようになる。
  - (b) 機械学習の実装ができるようになる。

### (2.2) セミナーのカリキュラム

- ・本セミナーでは、概ね、以下のような流れで進めます：
  - (a) 環境作成
  - (b) 機械学習の全貌
  - (c) Python の言語仕様
  - (d) 数学の基礎と実装
  - (e) 機械学習の理論と実装

### (2.3) 機械学習ライブラリ

- ・機械学習ライブラリとして、以下の例のように様々なものが提供されています。

・ Theano (テアノ)	: カナダ モントリオール大学
・ Pylearn2 (パイラーンツー)	: カナダ モントリオール大学 (「Theano」ベース)
・ Caffe (カフエ)	: アメリカ UCLB
・ Chainer (チェイナ-)	: 日本 (株)Preferred Networks (2015 OSSとして公開、2019開発終了)
・ TensorFlow (テンソルフロー)	: アメリカ Google
・ PyTorch (パイトーチ)	: アメリカ Facebook (Chainerを参考に開発)
・ CNTK	: アメリカ Microsoft (CNTK は Microsoft Cognitive Toolkit)
- ・本セミナーでは、  
様々な機械学習ライブラリのうち、世界的にも広く支持されている TensorFlow を扱います。  
これは、以下のように対応環境が多様で、様々な論文やサポートがあります。

・ 対応OS	: Windows、Linux、MacOS
・ 対応プログラミング言語	: C, C++, Python, Java, Go
・ 対応ハードウェア	: CPU, GPU(Graphics Processing Unit), TPU(Tensor processing unit)

### (2.4) 言語

- ・TensorFlow は様々な言語に対応していますが、Python のAPIが最も完成されているため、  
使用言語は Python とします。使用するライブラリの関係からバージョンは 3.7 以上とします。  
(TensorFlow に限らず、多くの機械学習の本が Python で解説しています。)

### (2.5) OS

- ・TensorFlow は様々なOSに対応していますが、本セミナーでは、  
パソコンOSとして普及している Windows を使用OSとします。

## (2.6) 教材

- ・本セミナーでは、主に以下の書籍を教材として使用します。

### (教材 1)

「Pythonで動かして学ぶ！あたらしい機械学習の教科書」  
(2018年01月 翔泳社 伊藤真著)

この書籍では、AI分野で必須となる用語や手法が、わかりやすく紹介されています。  
機械学習について、Pythonでの実装／関連する数学の基本／機械学習のアルゴリズム  
／TensorFlowを用いた実装、などについて、初学者向けに記された良書です。  
まずは、この書籍で、AI分野に取り組む為の基礎を築くのが良いかと思います。  
添付の Python ファイルで、実際に実装を見ながら動作確認できる、という利点があります。

### (教材 2)

「現場で使える！TensorFlow開発入門 Kerasによる深層学習モデル構築手法」  
(2018年04月 翔泳社 太田満久、須藤広大、黒澤匠雅、小田大輔 共著)

この本は、TensorFlowの導入から、高レベルAPIであるKerasを利用した実践的な  
深層学習モデルまで解説した、エンジニア向けの入門書です。  
主に画像関係のソリューションを扱っていて、  
数式は殆ど使わない TensorFlow 初学者向の書籍でもあります。  
この書籍で、機械学習のモデルがどのようなものかのイメージを築けるかと思います。  
添付の Python ファイルで、実際に実装を見ながら動作確認できる、という利点があります。

### (教材 3)

「ITエンジニアのための強化学習理論入門」(2020年07月 技術評論社 中井悦司)  
「現場で使える！Python 深層強化学習入門」(2019年07月 翔泳社 伊藤多一、他)  
「強化学習」(2018年11月 第1版第11刷 森北出版 Richard S. Sutton and Andrew G. Barto)

これらの本は、強化学習で参考になります。  
「ITエンジニアのための強化学習理論入門」は、実装を交えながらの解説例が多くて理解しやすいです。  
「現場で使える！Python 深層強化学習入門」は、理論と実装についての参考になります。  
「Sutton」本は強化学習入門者のバイブルと呼ばれていて、本資料の表記はこれに合わせています。  
「Sutton」本の第2版は、以下のサイトでpdfファイルをダウンロードできます(英語)：  
サイト⇒ <https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf>

### 【参照URL】

TensorFlow⇒ <https://www.TensorFlow.org/>  
Chainer開発終了⇒ <https://xtech.nikkei.com/atcl/nxt/column/18/00001/03341/>

### (3) 開発環境の作成 (Anaconda)

- ・ Pythonコードのローカルの開発環境として「Anaconda」を紹介します。

#### (3.1) インストール対象物

##### (3.1.1) Python (パイソン)

- ・ TensorFlow は様々な言語に対応していますが、Python のAPIが最も完成されているため、Python から使用することが一般的です。そのため Python の開発環境を構築します。
- バージョンは、2021年6月時点での最新公開バージョンである 3.8 とします。

##### (3.1.2) Anaconda (アナコンダ)

- ・ Anaconda は、Anaconda 社 (旧 Continuum Analytics 社) によって提供されている、Python のディストリビューションです。
- ・ Anaconda は、Python 本体に加え、科学技術、数学、エンジニアリング、データ分析など、よく利用される Python パッケージを一括でインストール可能にしたパッケージです。
- ・ なお、Anaconda は商用目的にも利用可能です。

##### (3.1.3) Jupyter Notebook (ジュプターノートブック)

- ・ Python のエディッターとして「Jupyter Notebook」をインストールします。
- ・ インタリセンス機能はないので少々不便ですが、実行確認が容易にできます。

##### (3.1.4) TensorFlow (テンソルフロー)

- ・ TensorFlow は Google が無償で提供している機械学習ライブラリです。

##### (3.1.5) Keras (ケラス)

- ・ Kerasは、Pythonで書かれた、TensorFlow/CNTK/Theano上で実行可能な高水準のニューラルネットワークライブラリです。
- 元々は TensorFlow とは独立していましたが、2017年初めに TensorFlow と統合されました。
- TensorFlow と統合された系統と、複数のバックエンドを選べるAPIという別の系統があります。
- ・ TensorFlow を用いる開発者から見ると、Keras は TensorFlow の使い勝手を良くしたラッパーライブラリでもあり、TensorFlow を統合した API仕様 となっています。

##### (3.1.6) その他のパッケージ

- ・ 上記以外に、科学技術計算用やグラフ表示用などの各種パッケージを、必要に応じてインストールします。
- 以下はその一例です。

パッケージ	説明
h5py	HDF5形式のファイルを取り扱うライブラリで、Kerasのモデルを保存する際に利用する
matplotlib	標準的な可視化ライブラリで、学習結果の可視化などで利用している
opencv	広く使われている画像処理ライブラリ
pandas	データ解析ライブラリ
pillow	標準的な画像処理ライブラリで、Kerasが内部的に利用している
scipy	科学技術計算ライブラリ

#### 【参照URL・出典】

「現場で使える！TensorFlow開発入門 Kerasによる深層学習モデル構築手法」 (2018年04月 翔泳社)

「<https://keras.io/ja/>」

「<https://pythondatascience.plavox.info/pythonのインストール/pythonのインストール-windows>」

「<https://deepblue-ts.co.jp/python/conda-command-basic/>」

「<https://anaconda.cloud/tutorials/getting-started-with-anaconda-individual-edition?source=download>」

### (3.2) インストール

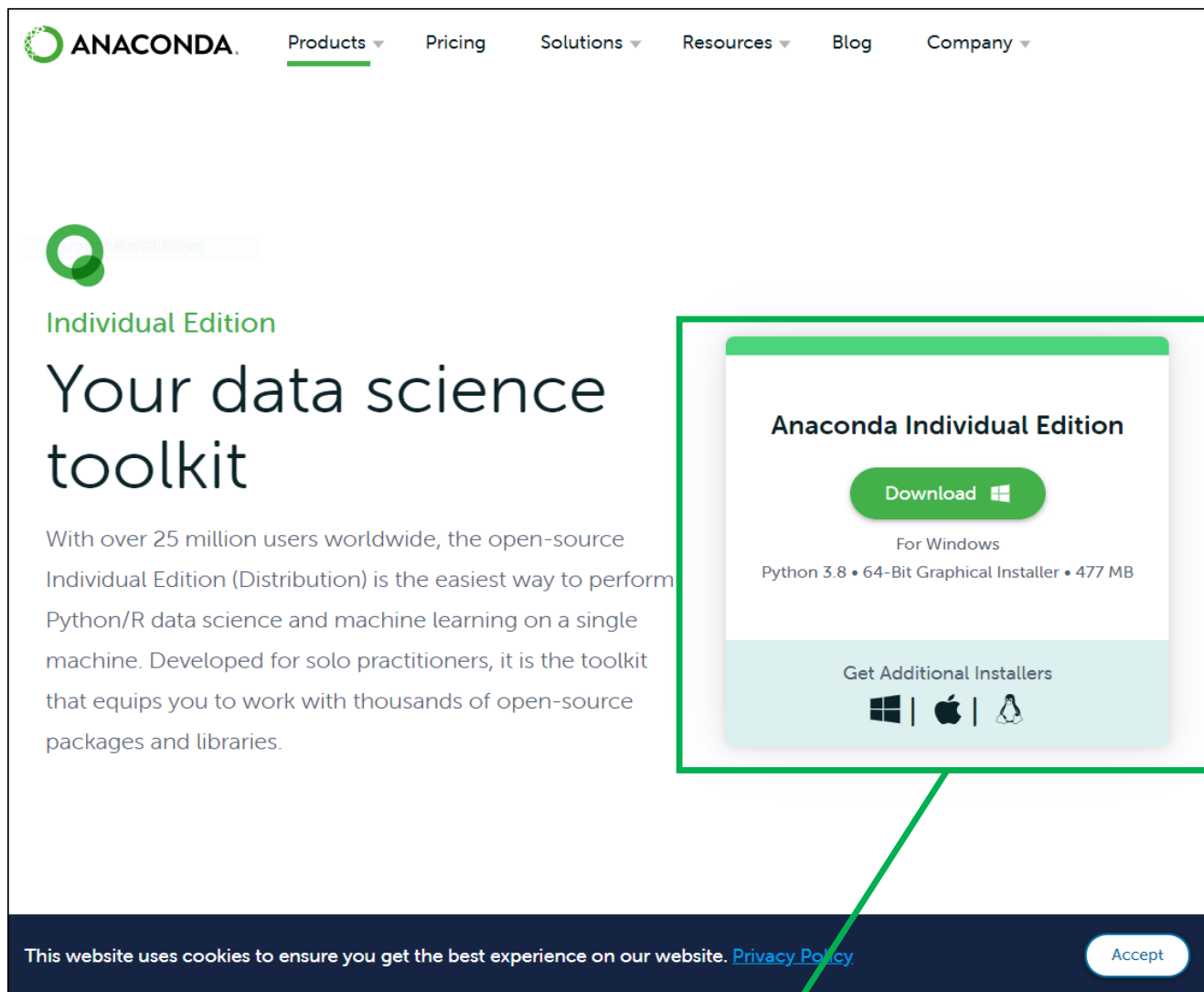
(※以下では「Anaconda Navigator」から仮想環境を指定の上でインストールする、という方法で説明しますが、「Anaconda Navigator」を使用しないコマンドラインを用いる方法等もあります。)

#### (3.2.1) Anaconda をインストール

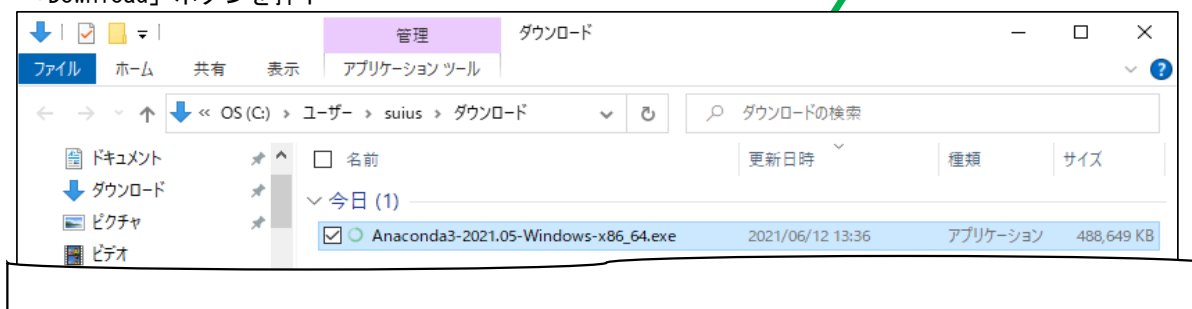
- Anaconda は、以下のサイトからダウンロードしてインストールします。  
インストールは、特に問題なければデフォルト値のままでよいでしょう。

`「https://www.anaconda.com/products/individual」`

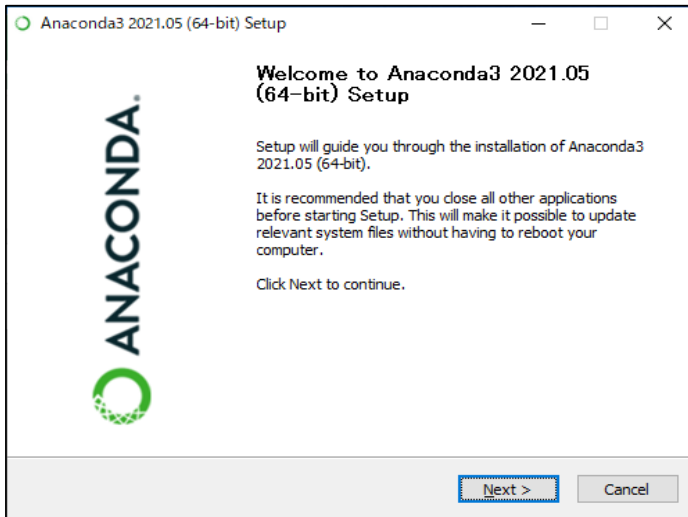
- 以下、「https://www.python.jp/install/anaconda/windows/install.html」を参考にして、ダウンロードからインストールまでの作業を行った経過を示します。



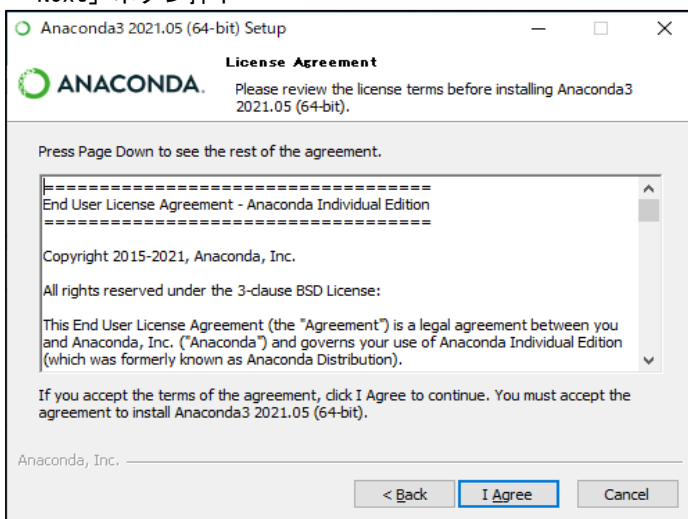
「Download」ボタンを押下



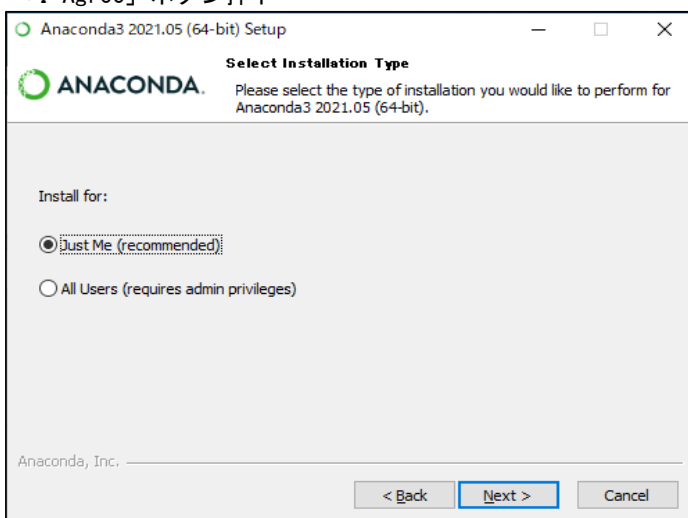
ダウンロードしたインストーラ「Anaconda3-2021.05-Windows-x86\_64.exe」を実行。



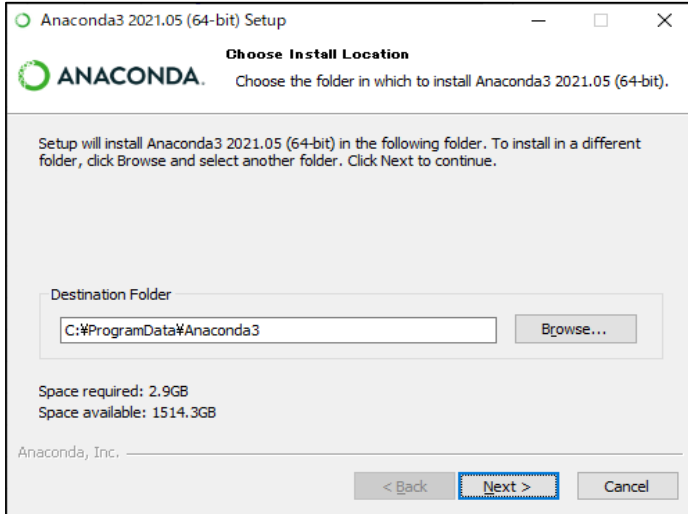
「Next」ボタン押下



「I Agree」ボタン押下



## 「Next」ボタン押下



筆者の環境では、

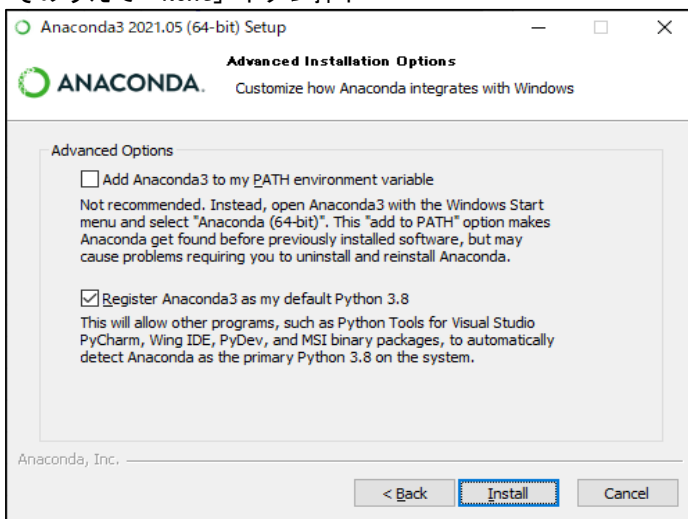
「Destination Folder」として、デフォルトの「C:\ProgramingData\Anaconda3」は使用済だったので  
「Destination Folder」として、デフォルトの「C:\ProgramingData\Anaconda3\_suiu」で指定しました。

尚、インストール先のフォルダーパスは、英数字のみで構成されているものを指定します。

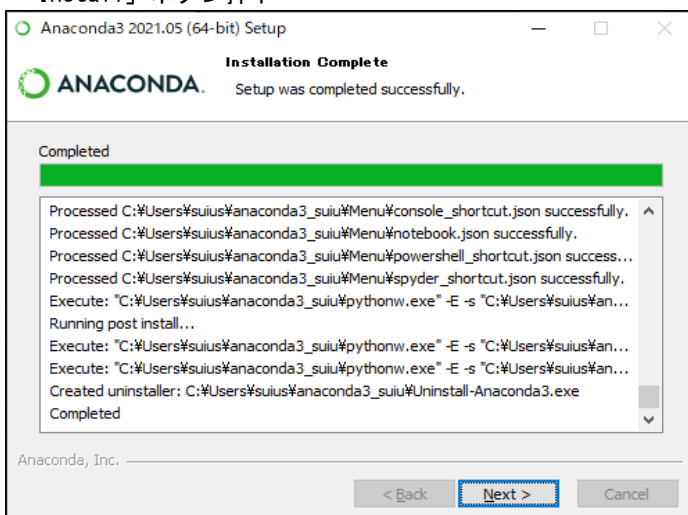
(日本語を含むフォルダーにはインストールしないように、注意してください！！

そうしないと、例えば TensorBoard による可視化がうまく機能しません。)

## そのうえで「Next」ボタン押下

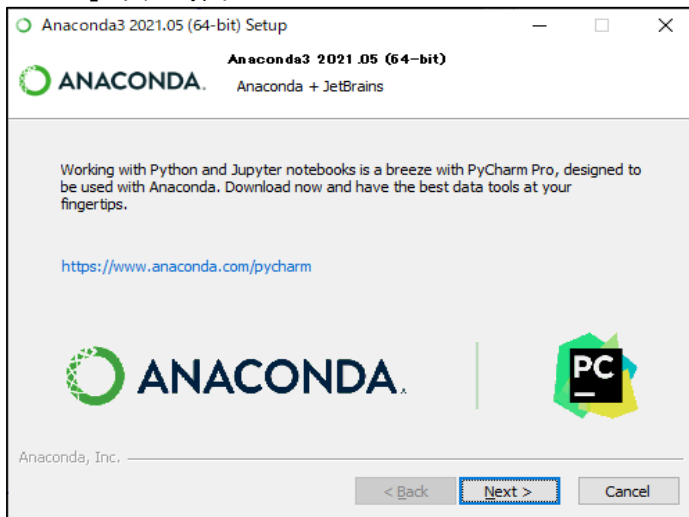


## 「Install」ボタン押下

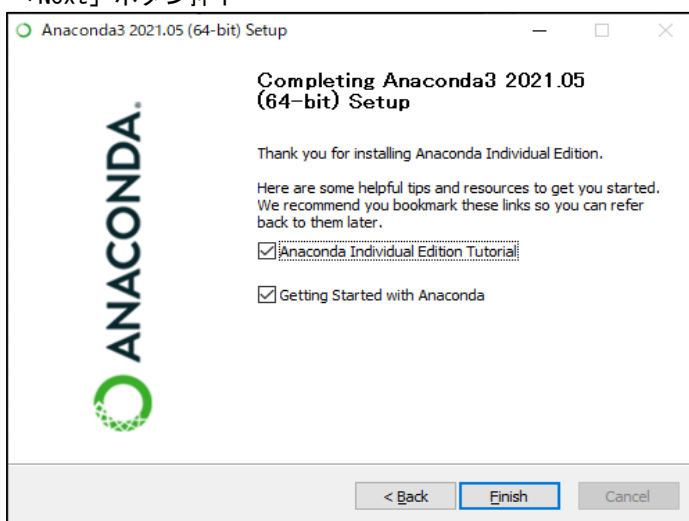




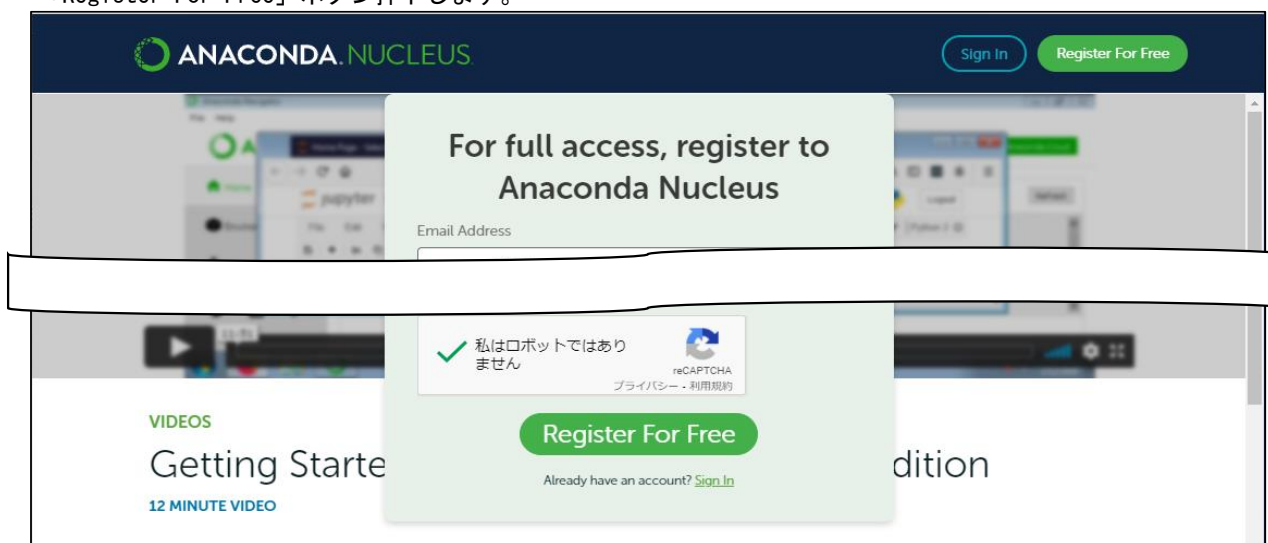
「Next」 ボタン押下



「Next」 ボタン押下



「Finish」 ボタン押下でブラウザ上に「ANACONDA. NUCLEUS」の画面が出てくるので必須項目を入力して「Register For Free」 ボタン押下します。




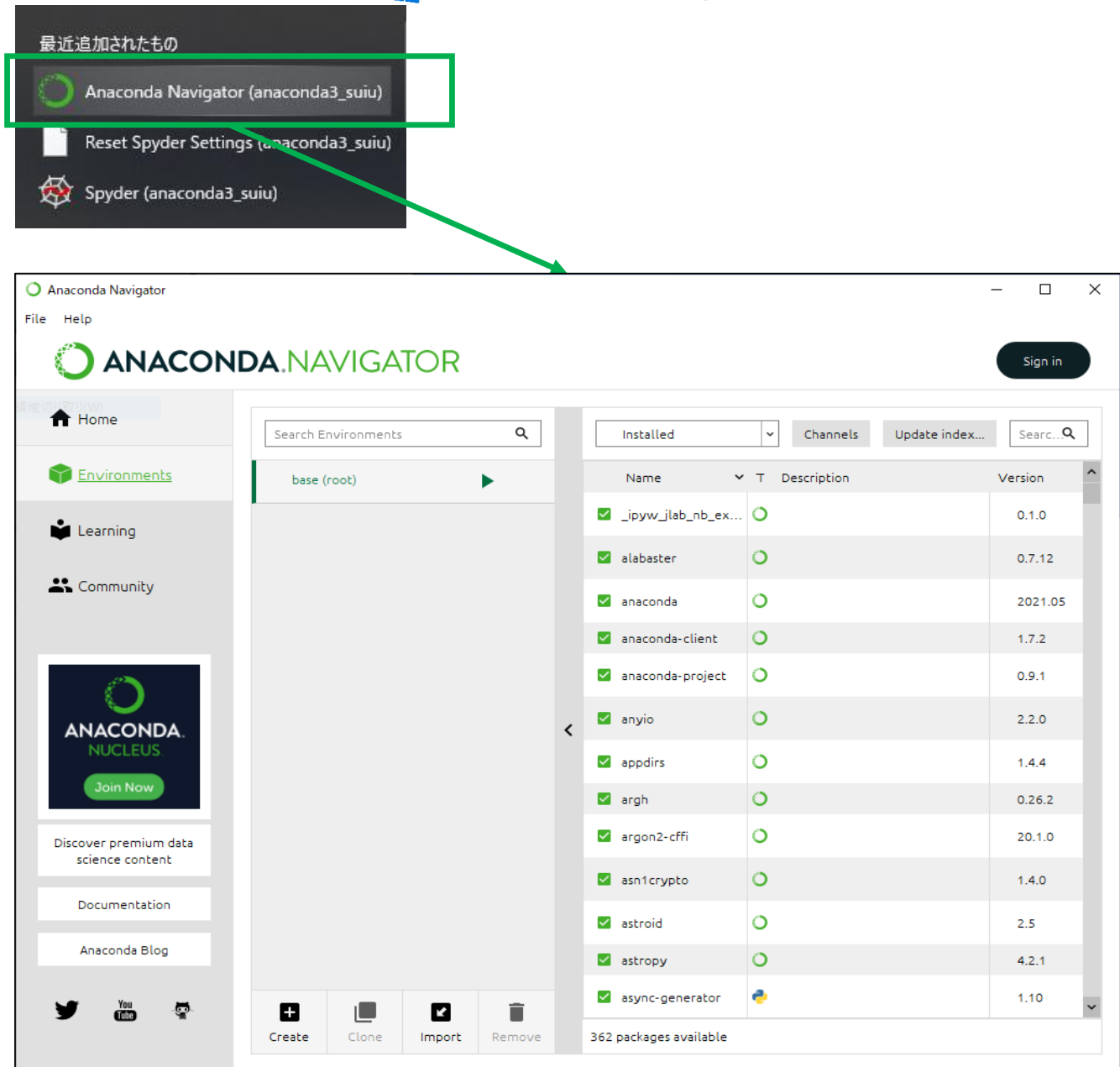
以上で、インストール作業は終了です。

「スタート」メニューに登録した「Anaconda Navigator」が表示されているのが確認できます。

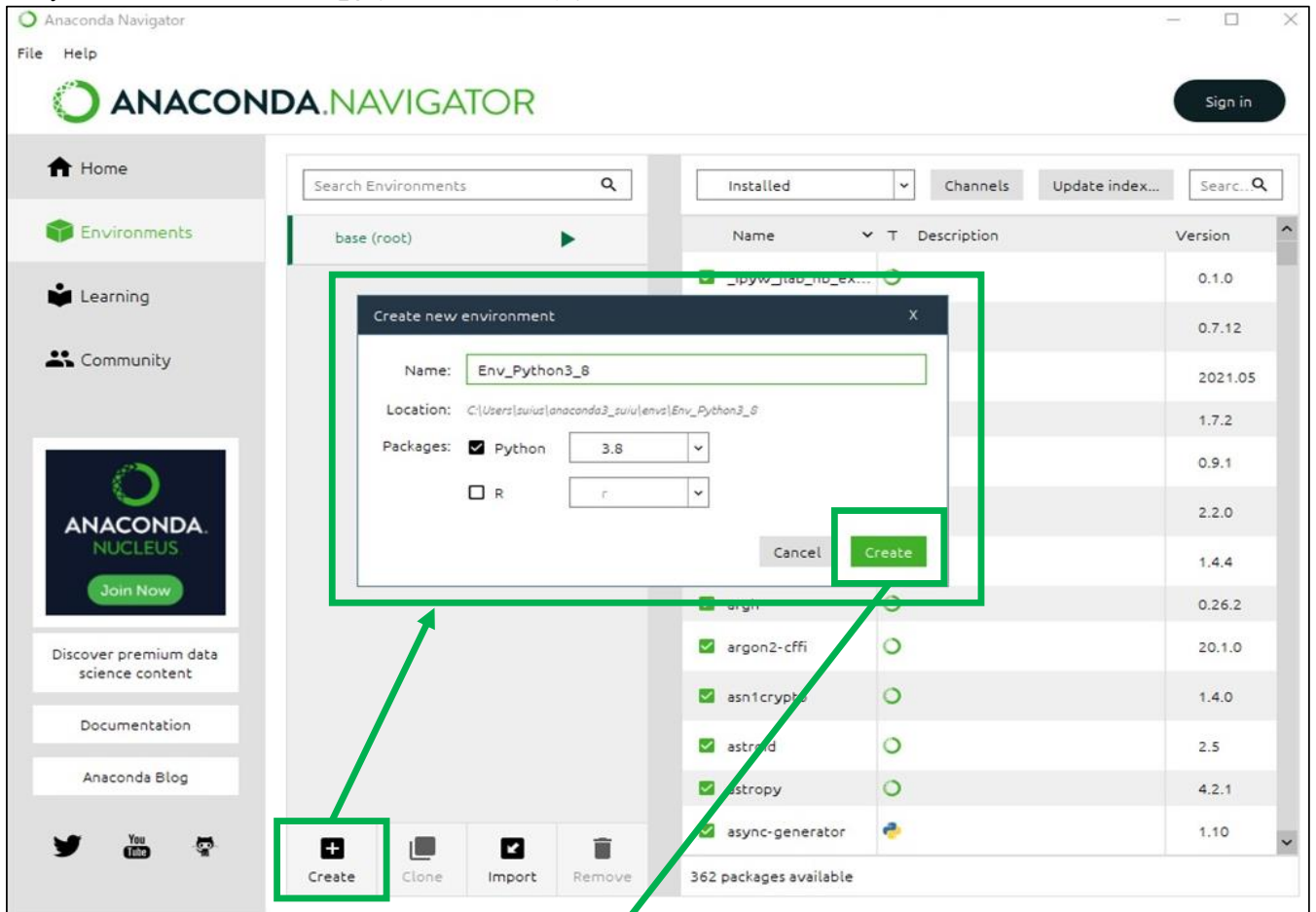
### (3.2.2) Python のバージョンを指定して Anaconda 仮想環境を作成

- ・ Anaconda をインストールすると、「Anaconda Navigator」もインストールされます。  
「Anaconda Navigator」から、バージョンを指定した開発環境（仮想環境）が、幾つでも作成できます。  
その手順を以下に述べます。

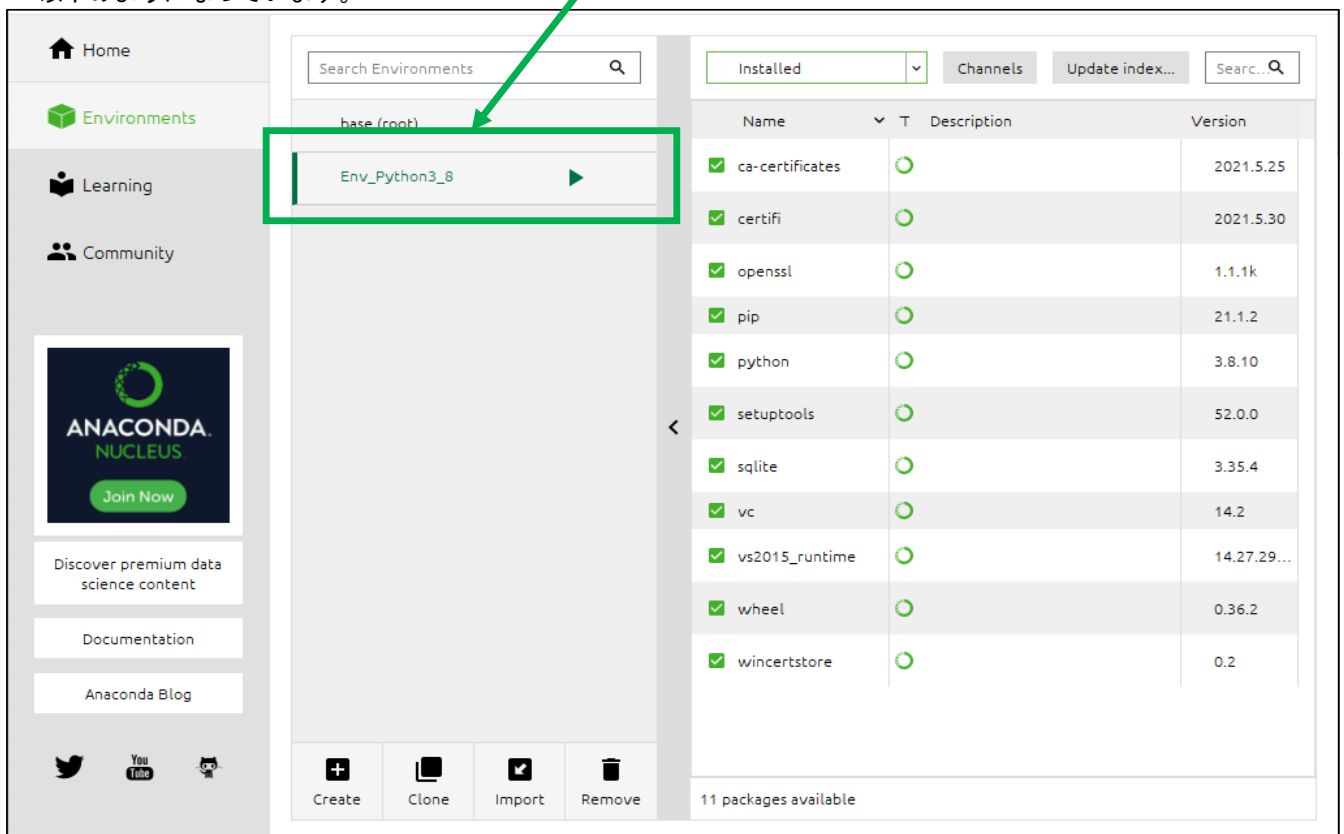
- ・ はじめに、タスクバーのスタート  メニューで、Anaconda Navigator を起動します。



- ・「Anaconda Navigator」から、「Environments -> Create」で仮想環境作成用のダイアログを表示し、Python のバージョンを指定した上で、仮想環境を作成します。
- ・以下の例では、「Env\_Python3\_8」という名前の仮想環境を、Python のバージョン 3.8 を指定して作成します。



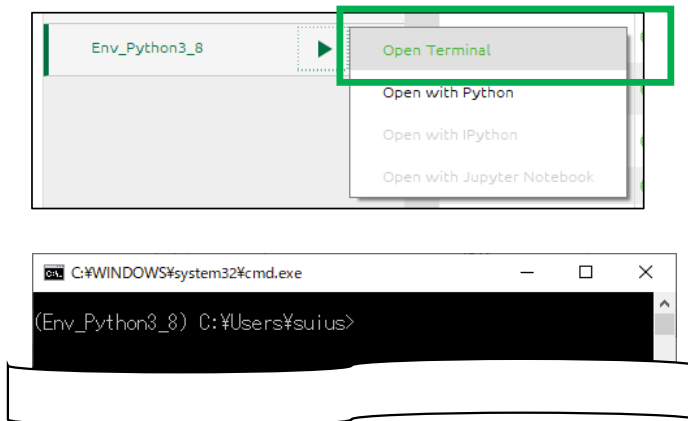
- ・上記例で、仮想環境「Env\_Python3\_8」を作成した直後の、インストール済みライブラリは以下になっています。



### (3.2.3) Anaconda 仮想環境でのインストール

- ・「Anaconda Navigator」から仮想環境を指定し、様々な環境をインストールします。  
以下の手順で行います。

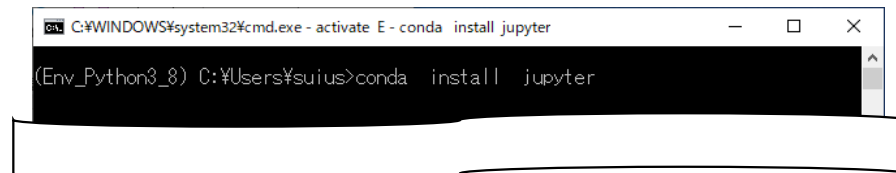
(1st) 仮想環境「Env\_Python3\_8」で「Open Terminal」を指定し、コマンドプロンプトを立ち上げます。  
これ以降のインストール作業は、この仮想環境のコマンドプロンプトで行います。



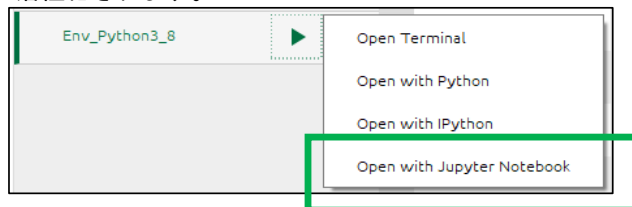
### (3.2.4) Jupyter Notebook (ジュパターノートブック)

- ・仮想環境のコマンドプロンプトから次のコマンドで、  
Jupyter Notebook をインストールします。

```
conda install jupyter
```



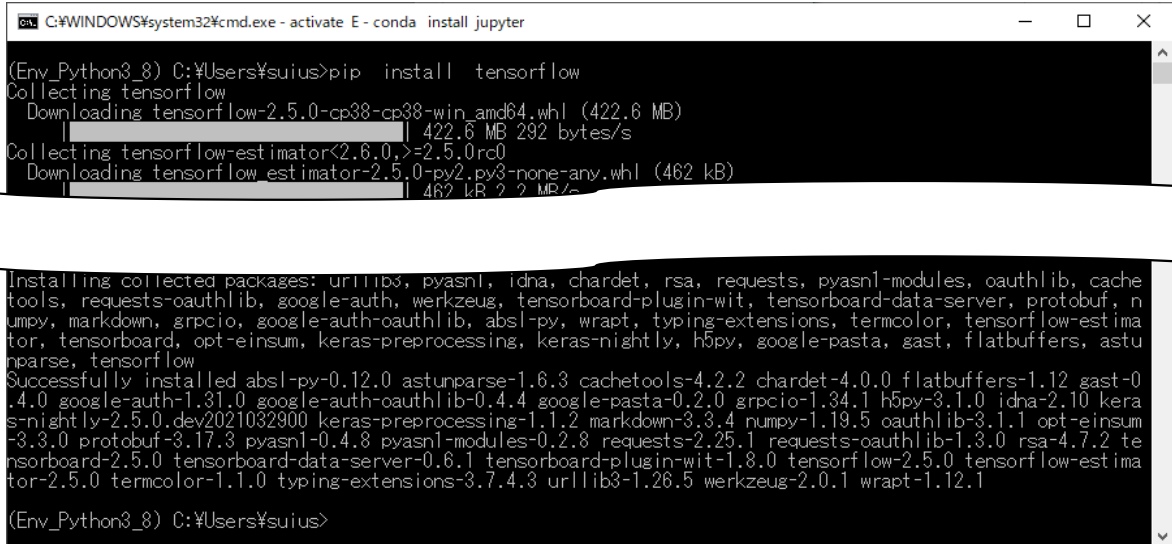
インストールにより、仮想環境「Env\_Python3\_8」で「Open With Jupyter Notebook」のメニューが  
活性化されます。



### (3.2.5) TensorFlow (テンソルフロー)

- ・仮想環境のコマンドプロンプトから次のコマンドで、TensorFlow をインストールします。
- さらにオプション指定で、tensorflow のバージョンを指定をすることもできます。

```
pip install tensorflow
```



```
C:\WINDOWS\system32\cmd.exe - activate E - conda install jupyter

(Env_Python3_8) C:\Users\suius>pip install tensorflow
Collecting tensorflow
  Downloading tensorflow-2.5.0-cp38-cp38-win_amd64.whl (422.6 MB)
    | 422.6 MB 292 bytes/s
Collecting tensorflow-estimator<2.6.0,>=2.5.0rc0
  Downloading tensorflow_estimator-2.5.0-py2.py3-none-any.whl (462 kB)
    | 462 kB 2.2 MB/s
Installing collected packages: urllib3, pyasn1, idna, chardet, rsa, requests, pyasn1-modules, oauthlib, cache
tools, requests-oauthlib, google-auth, werkzeug, tensorboard-plugin-wit, tensorboard-data-server, protobuf, n
umpy, markdown, grpcio, google-auth-oauthlib, absl-py, wrapt, typing-extensions, termcolor, tensorflow-estima
tor, tensorboard, opt-einsum, keras-preprocessing, keras-nightly, h5py, google-pasta, gast, flatbuffers, astu
mparse, tensorflow
Successfully installed absl-py-0.12.0 astunparse-1.6.3 cachetools-4.2.2 chardet-4.0.0 flatbuffers-1.12 gast-0
.4.0 google-auth-1.31.0 google-auth-oauthlib-0.4.4 google-pasta-0.2.0 grpcio-1.34.1 h5py-3.1.0 idna-2.10 kera
s-nightly-2.5.0.dev20210329000 keras-preprocessing-1.1.2 markdown-3.3.4 numpy-1.19.5 oauthlib-3.1.1 opt-einsum
-3.3.0 protobuf-3.17.3 pyasn1-0.4.8 pyasn1-modules-0.2.8 requests-2.25.1 requests-oauthlib-1.3.0 rsa-4.7.2 te
nsorboard-2.5.0 tensorboard-data-server-0.6.1 tensorboard-plugin-wit-1.8.0 tensorflow-2.5.0 tensorflow-estima
tor-2.5.0 termcolor-1.1.0 typing-extensions-3.7.4.3 urllib3-1.26.5 werkzeug-2.0.1 wrapt-1.12.1

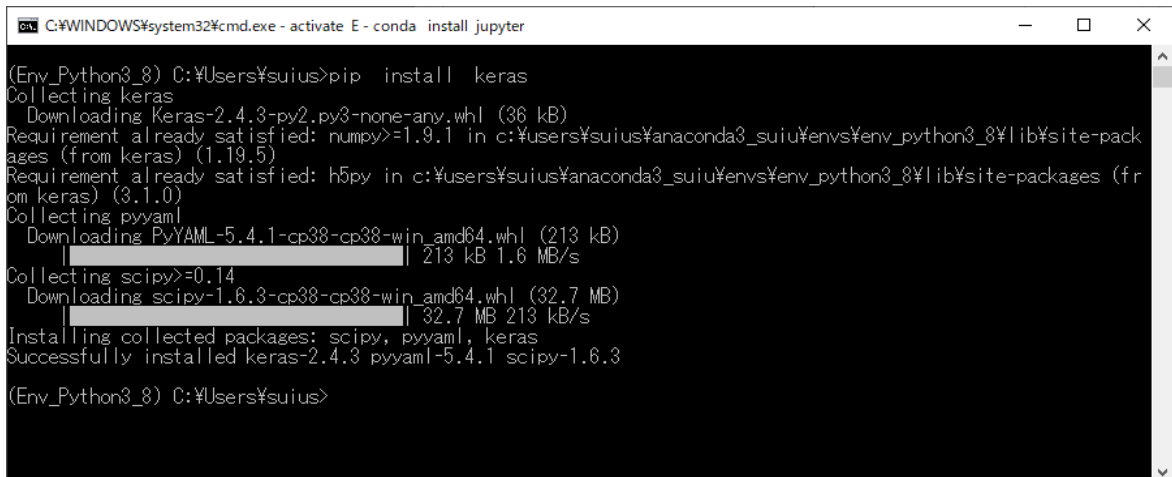
(Env_Python3_8) C:\Users\suius>
```

- ・場合によっては、インストール時に推奨メッセージが出ますので、その推奨に従ってコマンドを発行します。

### (3.2.6) Keras (ケラス)

- ・仮想環境のコマンドプロンプトから次のコマンドで Keras をインストールします。

```
pip install keras
```



```
C:\WINDOWS\system32\cmd.exe - activate E - conda install jupyter

(Env_Python3_8) C:\Users\suius>pip install keras
Collecting keras
  Downloading Keras-2.4.3-py2.py3-none-any.whl (36 kB)
Requirement already satisfied: numpy>=1.9.1 in c:\users\suius\anaconda3_suius\envs\env_python3_8\lib\site-pack
ages (from keras) (1.19.5)
Requirement already satisfied: h5py in c:\users\suius\anaconda3_suius\envs\env_python3_8\lib\site-packages (fr
om keras) (3.1.0)
Collecting pyyaml
  Downloading PyYAML-5.4.1-cp38-cp38-win_amd64.whl (213 kB)
    | 213 kB 1.6 MB/s
Collecting scipy>=0.14
  Downloading scipy-1.6.3-cp38-cp38-win_amd64.whl (32.7 MB)
    | 32.7 MB 213 kB/s
Installing collected packages: scipy, pyyaml, keras
Successfully installed keras-2.4.3 pyyaml-5.4.1 scipy-1.6.3

(Env_Python3_8) C:\Users\suius>
```

### (3.2.7) その他のパッケージ

- ・ 仮想環境のコマンドプロンプトから次のコマンドで、インストール済みのパッケージ一覧を参照できます。

```
conda list [-n 仮想環境名] (凡例: [ ] 内は省略可能)
```

以下は仮想環境 “Env\_Python3\_8” について、インストール済みのパッケージ一覧を参照する例です。

```
conda list -n Env_Python3_8
```

```
(Env_Python3_8) C:\Users\suius>conda list -n Env_Python3_8
# packages in environment at C:\Users\suius\anaconda3_suiu\envs\Env_Python3_8:
#
# Name                                Version                                Build                                Channel
abs1-py                               0.12.0                                pypi_0                               pypi
argon2-cffi                           20.1.0                                py38h2bbff1b_1                       pypi
astunparse                            1.6.3                                  pypi_0                               pypi
async_generator                       1.10                                   pyhd3eb1b0_0                         pypi
attrs                                 21.2.0                                pyhd3eb1b0_0                         pypi
backcall                              0.2.0                                  pyhd3eb1b0_0                         pypi
bleach                                3.3.0                                  pyhd3eb1b0_0                         pypi
ca-certificates                       2021.5.25                             haa95532_1                           pypi
werkzeug                              2.0.1                                  pypi_0                               pypi
wheel                                 0.36.2                                pyhd3eb1b0_0                         pypi
widgetsnbextension                    3.5.1                                  py38_0                               pypi
wincertstore                          0.2                                    py38_0                               pypi
winpty                                0.4.3                                  4                                     pypi
wrapt                                  1.12.1                                pypi_0                               pypi
zeromq                                4.3.3                                  ha925a81_3                           pypi
zipp                                   3.4.1                                  pyhd3eb1b0_0                         pypi
zlib                                   1.2.11                                h62dcd97_4                           pypi

(Env_Python3_8) C:\Users\suius>
```

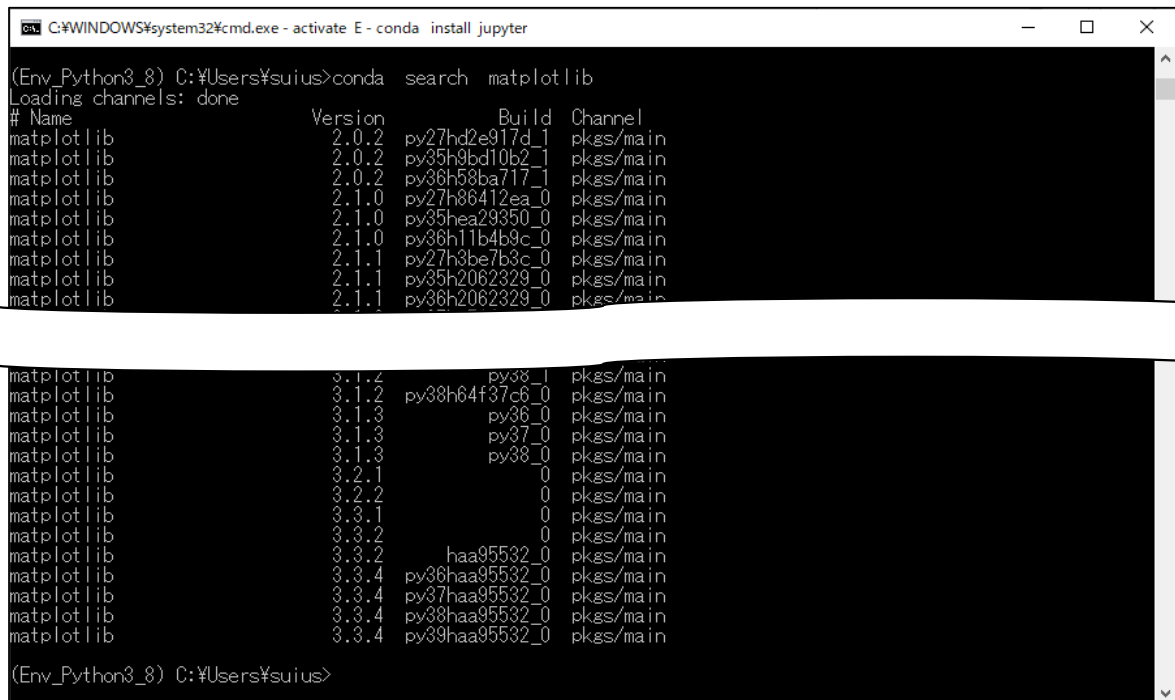
### (3.2.7.1) Anaconda で提供しているパッケージ

- Anaconda で提供しているパッケージについて、  
仮想環境のコマンドプロンプトから、次のコマンドで、パッケージのバージョン一覧を参照できます：

```
conda search パッケージ名
```

以下の例では、パッケージ “matplotlib” のバージョン一覧を参照します。

```
conda search matplotlib
```



```
(Env_Python3_8) C:\Users\suius>conda search matplotlib
Loading channels: done
# Name          Version      Build      Channel
matplotlib      2.0.2       py27hd2e917d_1 pkgs/main
matplotlib      2.0.2       py35h9bd10b2_1 pkgs/main
matplotlib      2.0.2       py36h58ba717_1 pkgs/main
matplotlib      2.1.0       py27h86412ea_0 pkgs/main
matplotlib      2.1.0       py35hea29350_0 pkgs/main
matplotlib      2.1.0       py36h11b4b9c_0 pkgs/main
matplotlib      2.1.1       py27h3be7b3c_0 pkgs/main
matplotlib      2.1.1       py35h2062329_0 pkgs/main
matplotlib      2.1.1       py36h2062329_0 pkgs/main
matplotlib      3.1.2       py38_1      pkgs/main
matplotlib      3.1.2       py38h64f37c6_0 pkgs/main
matplotlib      3.1.3       py36_0      pkgs/main
matplotlib      3.1.3       py37_0      pkgs/main
matplotlib      3.1.3       py38_0      pkgs/main
matplotlib      3.2.1       0           pkgs/main
matplotlib      3.2.2       0           pkgs/main
matplotlib      3.3.1       0           pkgs/main
matplotlib      3.3.2       0           pkgs/main
matplotlib      3.3.2       haa95532_0  pkgs/main
matplotlib      3.3.4       py36haa95532_0 pkgs/main
matplotlib      3.3.4       py37haa95532_0 pkgs/main
matplotlib      3.3.4       py38haa95532_0 pkgs/main
matplotlib      3.3.4       py39haa95532_0 pkgs/main

(Env_Python3_8) C:\Users\suius>
```

- インストール済みでないパッケージのうち、Anaconda で提供しているパッケージについては、  
仮想環境のコマンドプロンプトから次のコマンドで、  
バージョンを指定してパッケージをインストールします。

```
conda install パッケージ名[==バージョン名] (凡例：[ ] 内は省略可能)
```

- 上記コマンドで、バージョンを指定しないでインストールする場合、  
最新バージョンがインストールされます。  
他のパッケージのバージョンとの整合性を考慮する場合は、バージョンを指定します。

- ここでは、以下の各コマンドで各パッケージを最新のものでインストールします。

```
conda install matplotlib
conda install pandas
conda install pillow
```

- インストール済みのパッケージのバージョンが目的のものでない場合は、“conda uninstall パッケージ名”  
コマンドで、アンインストールした後で、バージョンを指定してパッケージを再インストールします。  
以下は、パッケージ ‘pandas’ をバージョン ‘0.22.0’ で再インストールする例です。

```
conda uninstall pandas
conda install pandas==0.22.0
```

### (3.2.7.2) Anaconda で提供していないパッケージ

- Anaconda で提供していないパッケージの場合、仮想環境のコマンドプロンプトから、次のコマンドでパッケージ提供しているチャンネルがあるか探します。

anaconda search パッケージ名

以下は、パッケージ "opencv" を提供しているチャンネルがあるか探す例です。

```
anaconda search opencv
```

[illegible]

- ・ パッケージを提供しているチャンネルの検索で複数の検索結果がある場合、バージョンと適用環境などから判断して、提供チャンネルを選択します。この場合もチャンネルの優先順位など、様々な注意が必要です。詳しくは、以下の公式ドキュメントを参照してください。

「<https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-channels.html>」

- ・提供チャンネルが決定したら、次のコマンドでインストールします。  
最新バージョンをインストールする場合は、チャンネルの指定は不要です。

```
conda install [-c チャネル名] パッケージ名 (凡例: [ ] 内は省略可能)
```

- ・ 以下は、パッケージ “opencv” の最新版をインストールする例です：

```
conda install opencv
```

```
C:\WINDOWS\system32\cmd.exe - conda install opencv

(Env_Python3_8) C:\Users\ysuius>conda install opencv
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\ysuius\anaconda3_suiu\envs\Env_Python3_8

  opencv=4.0.1 | 22 KB | ##### 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(Env_Python3_8) C:\Users\ysuius>
```



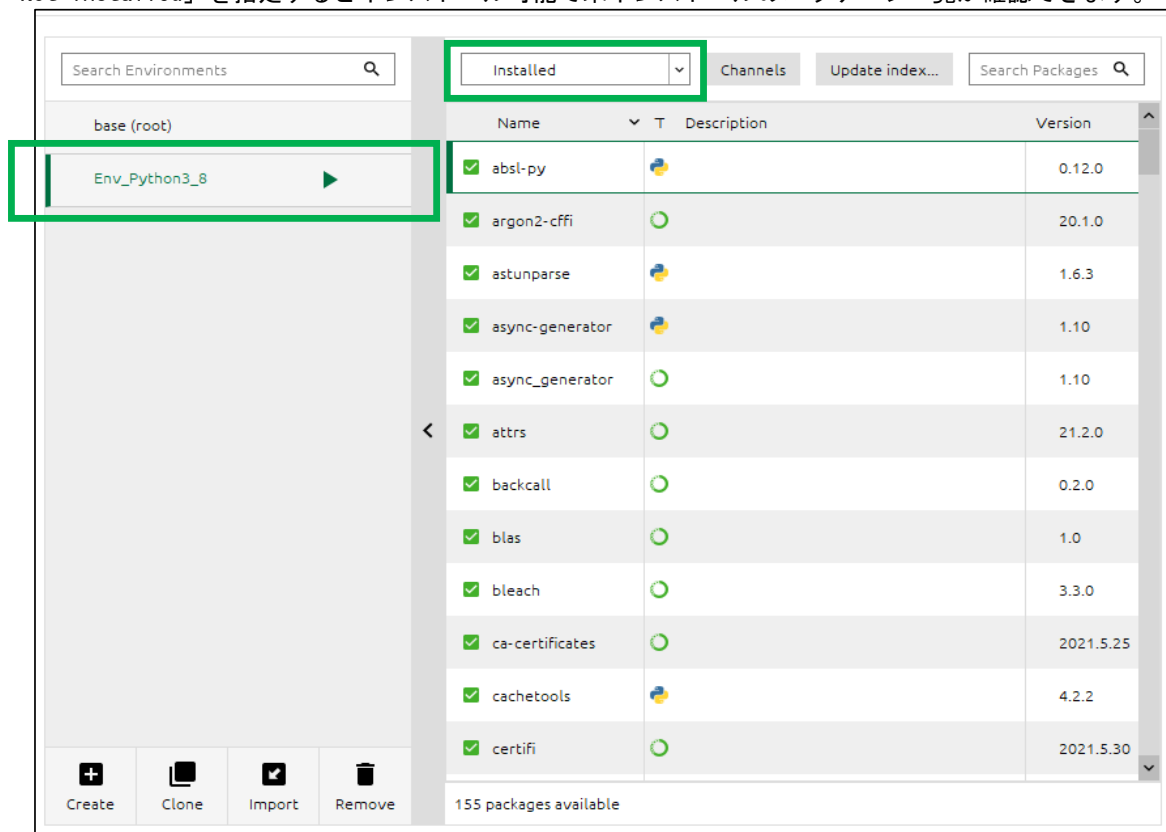
### (3.2.8) インストールしたパッケージ

- ・ (3.2.7) で既に述べたとおり、仮想環境のコマンドプロンプトから次のコマンドで、インストール済みのパッケージ一覧を参照できます。

```
conda list [ -n 仮想環境名 ]
```

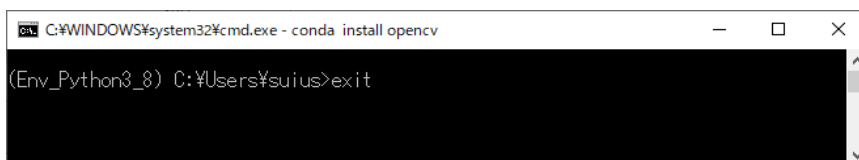
(凡例 : [ ] 内は省略可能)

- ・ 「Anaconda Navigator」で仮想環境の利用可能パッケージ一覧から、「Installed」を指定するとインストール済みのパッケージ一覧が確認できます。「Not installed」を指定するとインストール可能で未インストールのパッケージ一覧が確認できます。



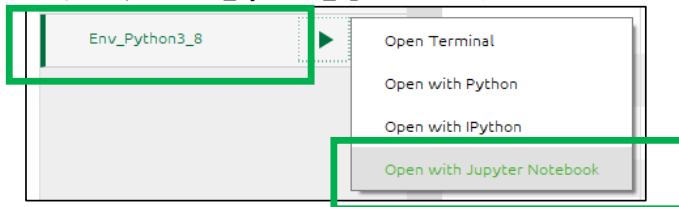
- ・ コマンドプロンプトを閉じるには、「exit」コマンドを実行します。

```
exit
```

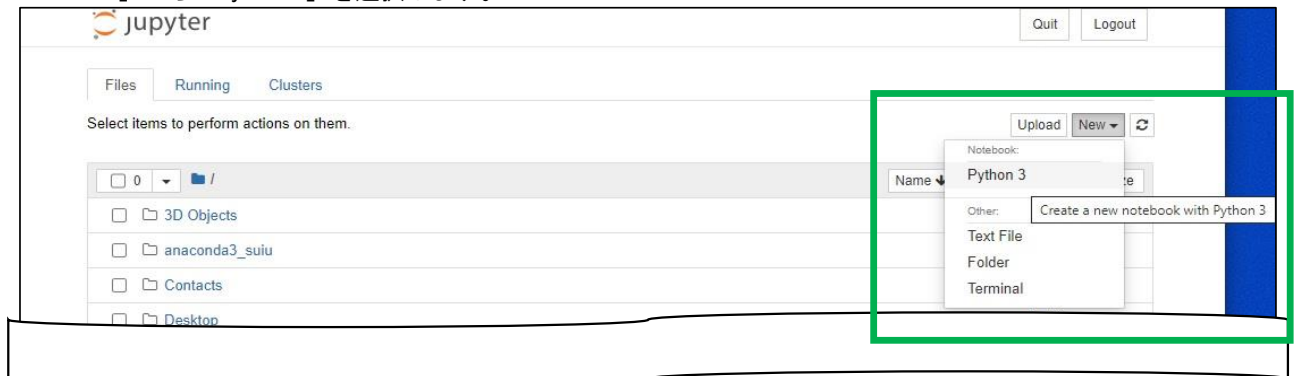


### (3.2.9) Jupyter Notebook による動作確認

- ・作成した仮想環境で、「Open with Jupyter Notebook」のメニューを選択します。  
以下は仮想環境「Env\_Python3\_8」での例です：

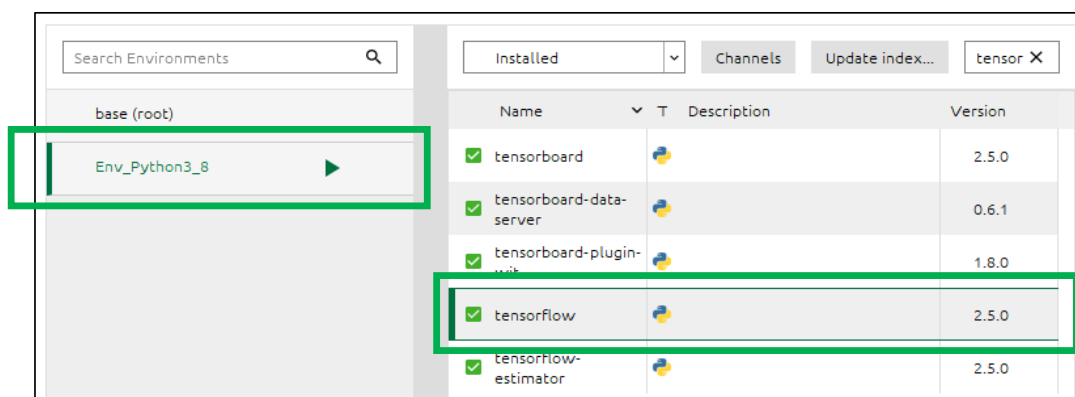
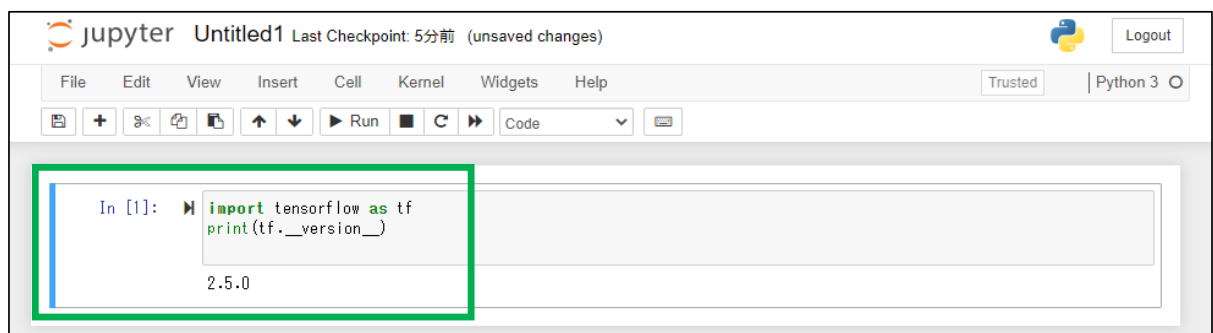


- ・するとブラウザが起動され、「Jupyter Notebook」の画面が表示されるので、右上の「New」から「Python3」を選択します。



- ・すると「Jupyter Notebook」が表示されます。  
以下のコードを入力して、[Shift]+[Enter]キー（あるいは、[Ctrl]+[Enter]キー）を押すと、  
importした TensorFlow のバージョンが表示されます。  
これが、「(3.2.5) TensorFlow (テンソフロー)」でインストールしたバージョンと同じであれば、  
「Jupyter Notebook」でバージョンの確認がとれたことになります。

```
import tensorflow as tf
print(tf.__version__)
```



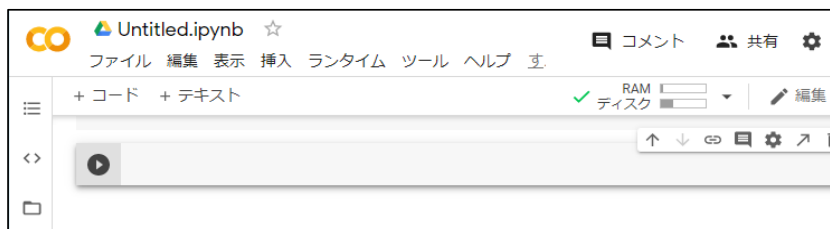
## (4) 開発環境の作成 (Anaconda以外)

- ・サーバー上に公開されている Pythonコードの開発環境として、「Google Colaboratory」と「AWS (特に “Amazon SageMaker”)」を紹介します。

### (4.1) Google Colaboratory

- ・ Pythonコードの開発環境として、Google は特別な設定なしで無料で使用できる「Google Colaboratory (グーグル コラボラトリ)」という環境をサーバー上に公開しています。以下の手順で用意します：

- (1) 予め以下のサイトでご自分の Google アカウントを用意しておきます：  
(Gmail のアカウントをお持ちの場合、Google アカウントは既にお持ちの状態です)  
「Google アカウントの作成 (<https://support.google.com/accounts/answer/27441>)」
- (2) 「Google Colaboratory」へ手順(1)で作成したアカウントでログインします：  
「Google Colaboratory (<https://colab.research.google.com>)」
- (3) ログインすると、以下のような画面が表示されるので、「ノートブックを新規作成」を選択すると、別のウィンドウで「Colab ノートブック」という、コードを記述して実行できる環境が開きます。  
あるいは「Colaboratory へようこそ」画面で、「ファイル」メニューから「ノートブックを新規作成」でも同じことができます：



- (4) これより後は、Anaconda 環境で Jupyter Notebook を開いた時と同様な操作になります。ライブラリのインストールなどの手間は不要です。
- (5) ご自分で作成した Anaconda 環境で実行した場合、機械学習の P C リソース占拠で P C 環境が使用できなくなることがありますので、上記環境での試行をお勧めします。尚、ファイルの保存や、接続時間などの制限がありますのでご注意ください。

## (4.2) Amazon SageMaker

- ・ Amazon が提供しているAWS (Amazon Web Services) では、AIのサービスも提供しており、機械学習サービス開発と公開や、公開されている出来合いのAPI利用が出来ます：

(自作サービスの開発と公開用)

Amazon SageMaker : 機械学習モデルの開発と提供

(出来合いのサービス利用・・・AWS提供)

Amazon CodeGuru	: コードレビューを自動化する
Amazon Comprehend	: 有用な情報を発見・分析する為の自然言語処理
Amazon Forecast	: 過去の履歴から将来を予測する時系列データ予測サービス
Amazon Lex	: 音声やテキストを使用して対話型インタフェースを構築するサービス
Amazon Personalize	: ユーザ向けにパーソナライズした推奨をするための機械学習サービス
Amazon Polly	: テキストを音声に変換する、音声認識サービス
Amazon Rekognition	: 画像・動画の認識サービス
Amazon Textract	: PDFや画像からテキストを抽出するサービス
Amazon Transcribe	: 音声をテキストに変換する、音声認識サービス
Amazon Translate	: テキストを言語間で翻訳するサービス

(出来合いのサービス利用・・・3rdベンダー提供)

3rdベンダー提供サービスをAWS上で利用することも出来ます。

### 【参照URL】

AWS機械学習サービス⇒<https://aws.amazon.com/jp/machine-learning/>

## (参考) pip と conda

### (参考.1) conda と pip について

インストール時に、pip と conda コマンドを混在して使用するにあたり、様々な混乱があるとのことなので、少し記事を集めてみました。

以下の記事は、下記URLからの要約です。

【参照URL】 [「https://code.i-harness.com/ja/q/1405a9c」](https://code.i-harness.com/ja/q/1405a9c)

【参照URL】 [「https://teratail.com/questions/14133」](https://teratail.com/questions/14133)

【参照URL】 [「https://pypi.org/」](https://pypi.org/)

【参照URL】 [「http://onoz000.hatenablog.com/entry/2018/02/11/142347」](http://onoz000.hatenablog.com/entry/2018/02/11/142347)

- ・ Condaは「Continuum Analytics」によって提供される、Anacondaのパッケージマネージャで、Anacondaの外部でも使用できます。Conda を使用して任意の言語のパッケージを管理できます。Condaは、言語に依存しない環境をネイティブに作成します。

- ・ pipは、Python環境で標準のパッケージマネージャです。  
pipは「virtualenv」に依存して、Python環境のみを管理します。

- ・ Anaconda下では基本的に「conda」を使ってパッケージをインストールするのですが、一部のパッケージはAnaconda社のレポジトリからは提供されていません。  
そのような場合にとるべきアプローチはいくつかあります。

1. デフォルト以外のレポジトリ（チャンネル）からインストール  
（例：「conda install -c matsci pymatgen」）
2. 自分でconda用のパッケージを作る
3. pipを使ってインストール

- ・ このうち最後の「pipを使ってインストール」をすると、condaとpipのパッケージが混ざって厄介なことになります。  
condaから入れたパッケージはpipからも認識されるものの、
  1. 依存関係のバージョン違い
  2. condaとpipのパッケージ名の違い（例：pyqt (conda) vs. PyQt5 (pip)）

等から予期せずcondaのパッケージが上書きされてしまうことがあります。

その結果、パッケージ1つのインストールでAnaconda環境が壊れてしまい、

Anacondaそのものを再インストールしない限り修復困難になってしまうことがあります。

また、condaがハードリンクを用いてパッケージを共有している関係から一つの環境で失敗してしまったが最後、他の仮想環境まで破壊されることもあります。

- ・ 「conda install パッケージ名」でパッケージが見つからなかった場合に、安易にpipから入れるのは危険です。

リスクを減らすためには例えば次の様な手順を踏みます。

1. 「anaconda search パッケージ名」でパッケージを提供しているチャンネルがないか探します。  
あれば「conda install -c チャンネル名 パッケージ名」等の方法でインストールします。  
（この場合もチャンネルの優先順位など、様々な注意が必要。詳しくは公式ドキュメント参照  
「<https://conda.io/docs/user-guide/tasks/manage-channels.html>」）。
2. pipから入れたい場合、まずPyPI (パイ、パイ) のサイト（「<https://pypi.org/>」）から該当するパッケージを探し、依存関係を調べておきます。  
依存するパッケージのうち、condaからインストール可能なものは予めインストールしておきます。
3. 依存関係を満たしたら「pip install --no-deps パッケージ名」でパッケージをインストールし、動作確認します。

- ・あるいは別の選択肢として、以下のような対応があります。
  1. pipからしか入れられないパッケージを入れたい場合、新しいcondaの環境を作る。  
(「conda create -n env python」)その環境内では「conda install」は一切使いません。
  2. Anacondaを使うのをやめます。  
Python公式サイト Pythonを使い、パッケージはpipで導入します。  
仮想環境については「venv」や「virtualenv」を用います。