不正侵入パケットに対する外れ値検知

1　目的

- 不正侵入パケットに対する外れ値検知の方法を学びます。
- Jubatus のインストール方法を学びます。
- Jubatus の使い方を学びます。

2　アマゾン EC2 へのログイン

アマゾン EC2 にログインします。

`https://aws.amazon.com/jp/ec2/`

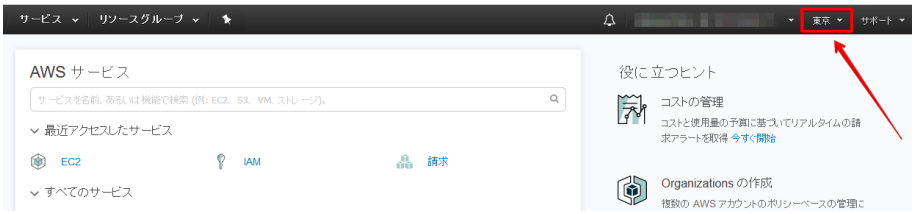
すでにアカウントがある場合

新規にアカウントをつくる場合

今回の授業では、すでにアカウントが作成されています。各自の課題提出フォルダの下に、各自の ID とパスワードが記載されたファイルがあります。そちらに記載された ID とパスワードを使ってログインして下さい。

2.1　リージョンの選択

利用するサーバのリージョンを選択します。

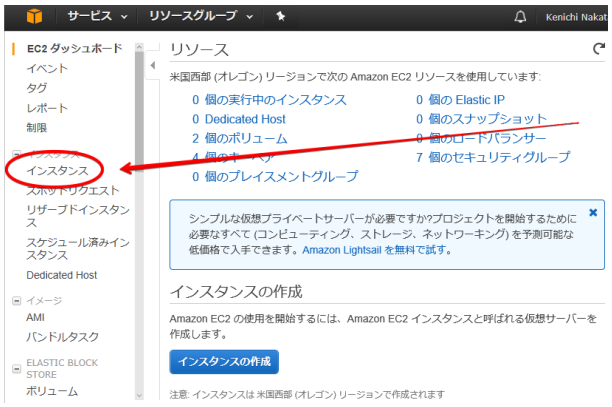


授業では、クリックして「米国東部（バージニア北部）」を選択して下さい。

EC2 サービスを選択します

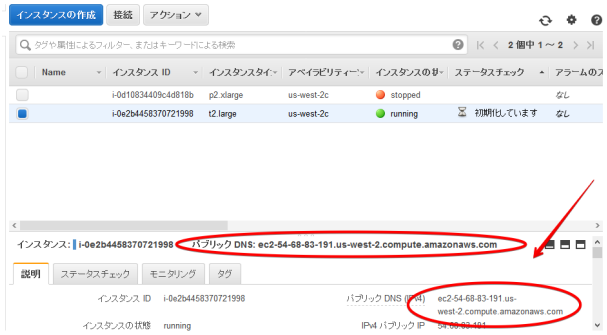
3　インスタンスの起動

以前の授業で作成した GPU 付インスタンスを作成します。授業を欠席して、作成していない人は、テキスト「アマゾン EC2 インスタンスの設定」の「インスタンスの作成」の章をみてインスタンスを作成して下さい。Putty や WinSCP のインストールや設定等を行っていない場合には、同じテキストを参照して、インストール及び設定を行って下さい。



「インスタンス」をクリックします

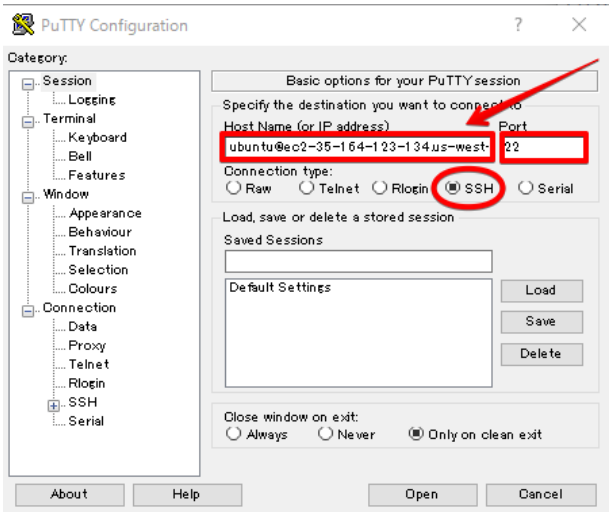
インスタンスが立ち上がっていない場合、「アクション」から「インスタンスの状態/開始」を選択し、インスタンスを立ち上げます。



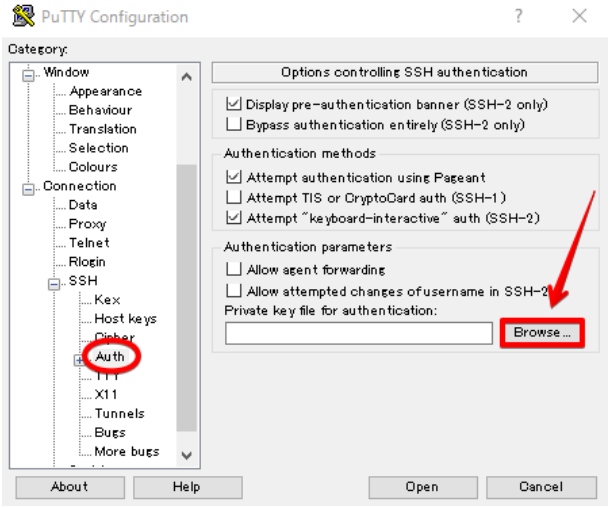
インスタンスの状態が「running」になったら、インスタンスの「パブリック DNS」を確認します。

4　PuTTY セッションの開始

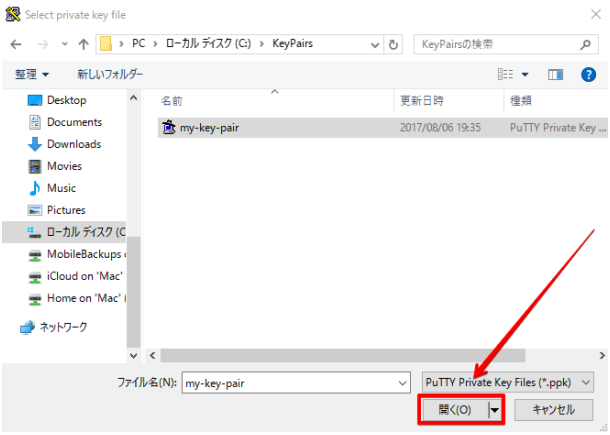
[スタート] メニューで [All Programs]-[PuTTY]-[PuTTY] を選択し、PuTTY を開始します。



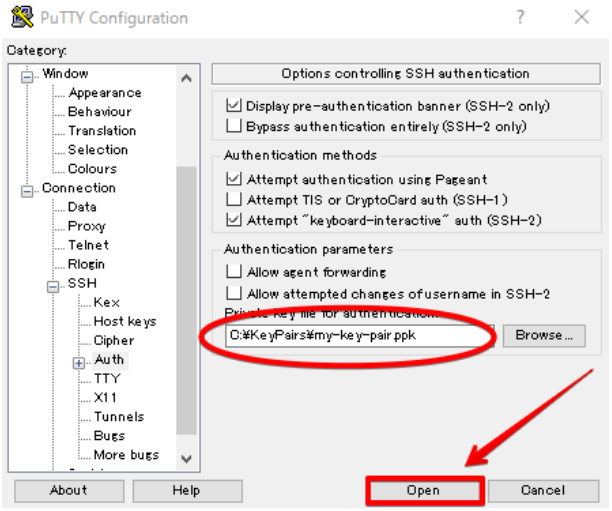
①「Host Name」に「ubuntu@(パブリック DNS)」を入力します。
②「Connection Type」を「SSH」とします。
③「Port」が「22」となっていることを確認します。



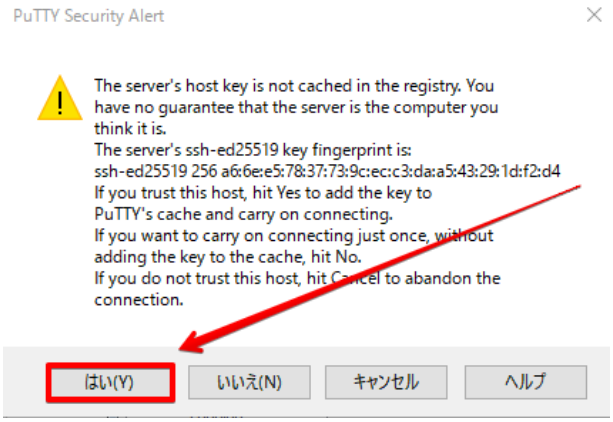
① [Category] ペインで、[Connection]、[SSH] の順に展開し、[Auth] を選択します。
②「Browse」をクリックします。



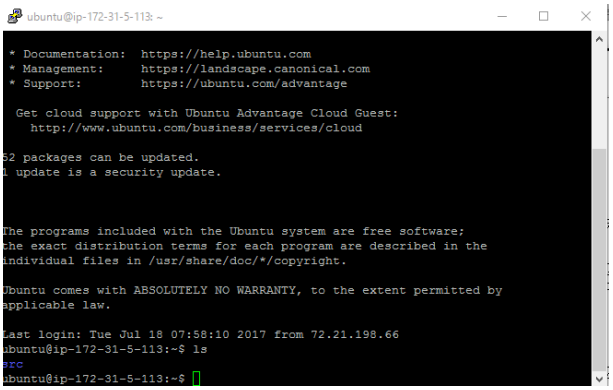
①先ほど保存した ppk ファイルを選択します。
②「開く」を押します。



①選択した ppk ファイルのパスが入力されていることを確認します。
②「Open」を押します。



「はい (Y)」を押します。



SSH クライアントでインスタンスに接続しました

5 ツールの取得

今回は、AWS でのデフォルトの環境で作業を行いますので、環境の切り替えの必要はありません。

```
cd ~/Lesson/tools
git pull
```

（参考　前回の事業を欠席している人は上記の変わりに、以下の通り行って下さい）

```
cd ~
mkdir Lesson
cd Lesson
git clone https://github.com/k1nk/tools.git
cd tools
```

6 Jubatus とは

データとその分析手法は、大きく 2 つのタイプに分けることができます。

- 蓄積されたデータ（ストック型）に対する一括分析（バッチ処理）
- 連像的に発生しているデータの流れ（フロー型）に対する逐次分析（リアルタイム処理）

Jubatus は、後者のフロー型データのリアルタイム分析を対象とし、オンライン機械学習による深い分析という付加価値を提供するフレームワークです。

Jubatus は、以下の 3 つの特徴を有してます。

深い分析

Jubatus は、分類、回帰、統計、近傍探索、推薦、異常検知、クラスタリング、クラスタ分析、統計量、グラフマイニングなど多くの深い分析を実現しています。またバッチ型機械学習と同程度の学習精度を実現しています。

スケーラビリティ

Jubatus は、廉価なコモディティサーバを多数並べた分散処理（スケールアウト）が可能です。理論的には 100 台程度までスケールアウトが可能です。Jubatus では、コモディティサーバで 10 万 qps を超えるスケーラブルな機械学習を実現しています。

リアルタイム

Jubatus は、データをためることなく瞬時に学習を行います

今回は、Jubatus を使って、不正侵入パケットに対する外れ値検知を行います。

7 Jubatus のインストール

以下の行を /etc/apt/sources.list.d/jubatus.list に記述して、Jubatus の Apt リポジトリをシステムに登録します。

```
deb http://download.jubat.us/apt/ubuntu/xenial binary/
```

```
sudo nano /etc/apt/sources.list.d/jubatus.list
上記の行を記載してファイルを(Ctrl+oで)保存します。
Ctrl+x でnanoを終了します。
```

jubatus のパッケージをインストールします。

```
sudo apt-get update
sudo apt-get install jubatus
```

以下の警告メッセージが表示された場合は、 y を入力してください。

```
Install these packages without verification [y/N]? y
```

これで、Jubatus が /opt/jubatus にインストールされました。

Jubatus を使う前に、毎回 profile スクリプトから環境変数を読み込む必要があります

```
source /opt/jubatus/profile
```

8 Jubatus クライアントのインストール

```
pip install jubatus
```

今回は、サーバとクライアントを同じマシンで動かしますが、別のマシンで実行することが一般的です。

9 データのダウンロード

```
cd ~/Lesson/tools/jubatus
wget http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.
   gz
gunzip kddcup.data_10_percent.gz
mv kddcup.data_10_percent kddcup.data_10_percent.txt
```

10 プログラムの実行

10.1 Jubatus サーバ

```
jubaanomaly --configpath config.json &
```

設定ファイル（config.json）

```
{
 "method" : "lof",
 "parameter" : {
  "nearest_neighbor_num" : 10,
  "reverse_nearest_neighbor_num" : 30,
  "method" : "euclid_lsh",
  "parameter" : {
   "hash_num" : 8,
   "table_num" : 16,
   "probe_num" : 64,
   "bin_width" : 10,
   "seed" : 1234
  }
 },

 "converter" : {
  "string_filter_types": {},
  "string_filter_rules": [],
  "num_filter_types": {},
  "num_filter_rules": [],
  "string_types": {},
  "string_rules": [{"key":"*", "type":"str", "global_weight" : "bin", "
     sample_weight" : "bin"}],
  "num_types": {},
  "num_rules": [{"key" : "*", "type" : "num"}]
 }
}
```

10.2 Jubatus クライアント

```
python anomaly.py
```

LOF が 1 又は Inf でないものを表示します。

クライアントプログラム（anomaly.py）

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
```

```
3
4  import signal
5  import sys, json
6  from jubatus.anomaly import client
7  from jubatus.common import Datum
8
9  NAME = "anom_kddcup";
10
11 # handle keyboard interruption"
12 def do_exit(sig, stack):
13     print('You pressed Ctrl+C.')
14     print('Stop running the job.')
15     sys.exit(0)
16
17 if __name__ == '__main__':
18     # 0. set KeyboardInterrupt handler
19     signal.signal(signal.SIGINT, do_exit)
20
21     # 1. set jubatus server
22     anom = client.Anomaly("127.0.0.1", 9199, NAME)
23
24     # 2. prepare training data
25     with open('kddcup.data_10_percent.txt', mode='r') as file:
26         for line in file:
27             duration, protocol_type, service, flag, src_bytes,
                   dst_bytes, land, wrong_fragment, urgent, hot,
                   num_failed_logins, logged_in, num_compromised,
                   root_shell, su_attempted, num_root, num_file_creations,
                   num_shells, num_access_files, num_outbound_cmds,
                   is_host_login, is_guest_login, count, srv_count,
                   serror_rate, srv_serror_rate, rerror_rate,
                   srv_rerror_rate, same_srv_rate, diff_srv_rate,
                   srv_diff_host_rate, dst_host_count, dst_host_srv_count,
                   dst_host_same_srv_rate, dst_host_diff_srv_rate,
                   dst_host_same_src_port_rate, dst_host_srv_diff_host_rate
                   , dst_host_serror_rate, dst_host_srv_serror_rate,
                   dst_host_rerror_rate, dst_host_srv_rerror_rate, label =
                   line[:-1].split(",")
28
29             datum = Datum()
```

```
30              for (k, v) in [
31                      ["protocol_type", protocol_type],
32                      ["service", service],
33                      ["flag", flag],
34                      ["land", land],
35                      ["logged_in", logged_in],
36                      ["is_host_login", is_host_login],
37                      ["is_guest_login", is_guest_login],
38                      ]:
39                  datum.add_string(k, v)
40
41              for (k, v) in [
42                      ["duration",float(duration)],
43                      ["src_bytes", float(src_bytes)],
44                      ["dst_bytes", float(dst_bytes)],
45                      ["wrong_fragment", float(wrong_fragment)],
46                      ["urgent", float(urgent)],
47                      ["hot", float(hot)],
48                      ["num_failed_logins", float(num_failed_logins)],
49                      ["num_compromised", float(num_compromised)],
50                      ["root_shell", float(root_shell)],
51                      ["su_attempted", float(su_attempted)],
52                      ["num_root", float(num_root)],
53                      ["num_file_creations", float(num_file_creations)],
54                      ["num_shells", float(num_shells)],
55                      ["num_access_files", float(num_access_files)],
56                      ["num_outbound_cmds",float(num_outbound_cmds)],
57                      ["count", float(count)],
58                      ["srv_count",float(srv_count)],
59                      ["serror_rate", float(serror_rate)],
60                      ["srv_serror_rate", float(srv_serror_rate)],
61                      ["rerror_rate", float(rerror_rate)],
62                      ["srv_rerror_rate",float( srv_rerror_rate)],
63                      ["same_srv_rate", float(same_srv_rate)],
64                      ["diff_srv_rate", float(diff_srv_rate)],
65                      ["srv_diff_host_rate", float(srv_diff_host_rate)],
66                      ["dst_host_count",float( dst_host_count)],
67                      ["dst_host_srv_count", float(dst_host_srv_count)],
68                      ["dst_host_same_srv_rate",float(
                           dst_host_same_srv_rate)],
```

```
69                      ["dst_host_same_src_port_rate",float(
                            dst_host_same_src_port_rate)],
70                      ["dst_host_diff_srv_rate", float(
                            dst_host_diff_srv_rate)],
71                      ["dst_host_srv_diff_host_rate",float(
                            dst_host_srv_diff_host_rate)],
72                      ["dst_host_serror_rate",float(dst_host_serror_rate
                            )],
73                      ["dst_host_srv_serror_rate",float(
                            dst_host_srv_serror_rate)],
74                      ["dst_host_rerror_rate",float(dst_host_rerror_rate
                            )],
75                      ["dst_host_srv_rerror_rate",float(
                            dst_host_srv_rerror_rate)],
76                      ]:
77                  datum.add_number(k, v)
78
79              # 3. train data and update jubatus model
80              ret = anom.add(datum)
81
82              # 4. output results
83              if (ret.score != float('Inf')) and (ret.score!= 1.0):
84                  print (ret, label)
```

10.3 プログラムの終了方法

Jubatus のクライアント用プログラムは、適用なところで、Ctrl+c で終了します。Jubatus サーバプログラムはバックグラウンドで実行されています。

```
jobs
```

として番号を確認し、

```
kill #1
```

として、終了するジョブの番号を先頭に#をつけて指定し、終了します。