**GLOBAL VARIABLE**

def get\_probability\_trustworthy(nodes\_and\_likelihood, hypothesis, current\_node=None ):

**global net**

return d

**MONGODB QUERY**

cursor = db.assets.find({'security.tenant': 'jpmc'})

**ITERATING THROUGH CURSOR**

cursor = db.assets.find({'security.tenant': 'jpmc'})

**for doc in cursor:**

1. **DICTIONARY WITH VALUES AS LIST**
2. **EXAMPLE OF EXTRACTING INFO IN MONGO**
3. **ITER ITEMS EXAMPLE**
4. **ADD LIST TO VALUE TO DICTIONARY**
5. **DEFAULTDICT**
6. **ENUMERATE**

for doc in cursor:

pid = doc['pid']

load\_default\_network()

**node\_and\_likelihood = defaultdict(list)**

trustworthy\_posterior = **doc['model\_results']['carbon']['results']['Trustworthy']['states']['true']['posterior']**

**model\_results = doc['model\_results']['carbon']['results']**

**for key, result in model\_results.iteritems():**

if len(applied\_ingestors) > 0:

for ingestor in applied\_ingestors:

netica\_node\_name = ingestor['netica\_node\_name']

likelihood = ingestor['lh'][0]**GROUPBY PANDAS CALCULATE MEAN AND SORT VALUES**

**node\_and\_likelihood[netica\_node\_name].append(likelihood)**

**for i, result in enumerate(list\_netica\_node\_likelihoods):**

ratio = a/result[0]

netica\_node = result[1]

df\_temp.loc[i]= [pid, netica\_node,ratio]

**df = pd.concat([df, df\_temp], axis=0)**

**CREATE MULTIPLE HISTOGRAMS BY FIELD AND FIGSIZE AND BINS**

%matplotlib inline

import matplotlib.pyplot as plt

df.hist(by=df['netica\_node'],figsize=(12,14),bins=7, color='red')

g = df.groupby('netica\_node')

(g.ratio.mean() \* g.netica\_node.count()).sort\_values(ascending=False)

**ADD ADDITIONAL FILTER**

df = df[df['ratio'] < 1.0]

g = df.groupby('netica\_node')

((1.- g.ratio.mean()) \* g.netica\_node.count()).sort\_values(ascending=False)

**HOW TO STORE DATA FRAME**

df.to\_pickle(file\_name) # where to save it, usually as a .pkl  
Then you can load it back using:  
  
df = pd.read\_pickle(file\_name)

**FILL DATA AND FORMAT DATE**

dates = ['2016-1-{}'.format(i)for i in range(1,21)]

df['Date'] = pd.to\_datetime(df['Date'])

**REMEMBER TO ADD VALUES IF YOU HAVE NULL**

ts = pd.Series(df['Value'].values, index=df['Date'])

**ASSIGN, SORT\_VALUES, GROUPBY, DIFF, FILLNA**

<http://stackoverflow.com/questions/41929772/time-difference-within-group-by-objects-in-python-pandas/41929958#41929958>

df.assign(  
 timediff=df.sort\_values(  
 'datetime', ascending=False  
 ).groupby(['from', 'to']).datetime.diff(-1).dt.seconds.div(60).fillna(0))

**CHECK IF ONE DATA FRAME ISN’T WITHIN THE OTHER COLUMNS**

df1[~df1.sym1.isin(pd.concat([df2, df3, df4, df5]).sym2)]

http://stackoverflow.com/questions/41930333/how-to-check-one-dataframe-column-available-or-not-in-other-four-dataframes-col/41930378#41930378

**IPYTHON NOTEBOOK - RUN SCRIPT**

Cell - Run All

**D3 - Server Reading CSV FILE**

In directory run: python -m SimpleHTTPServer

http://localhost:8000/index3.html

**GIT TO GET UPDATED API MODEL.PY**

git checkout api/model.py

**WHERE IS THE ERROR LOG DIRECTORY**

/usr/local/var/log/apache2

**OPEN SUBLIME FILE FROM COMMAND LINE**

open . -a 'Sublime Text 2.app'

**CLEAR CACHE BACKUP RESTART - TOTAL TIME - 5 MINS**

[**D3 Template**](https://docs.google.com/document/d/1l-CczC-4L8Xw9zb3qmZy_YV6PpCAKNdiuWpJf5eY78k/edit)

**D3 Reading from JSON File**

{

"links": [

{"source": "Commits\_Crime", "target": 7},

{"source": "Drinks", "target": 4},

{"source": "Doesn't Drink", "target": 3}]

}

d3.json("netica.json", function(error, graph) {

if (error) {

console.log("error reading file");

}

console.log(graph)

var linkedByIndex = {};

graph.links.forEach(function(d) {

console.log(d.source)

console.log(d.target)

// linkedByIndex[d.source + "," + d.target] = true;

});

});

**SUBLIME - CTRL G TO LINE NUMBER**

**APP.JS - Routing**

when('/dashboard', {

templateUrl: 'dashboard/dashboard.html',

controller: 'DashboardCtrl',

label: 'Dashboard',

authorized\_key : authorized\_keys.dashboard,

menuid: 'app\_link\_li\_dashboard',

apiaccess: { 'service': 'Dashboard', 'verb': 'GET' },

resolve: {//authorized variable can be accessed from next route controller (see ThreatStreamCtrl)

authorized: ["$rootScope", function ($rootScope) {

return $rootScope.isauthorized("Dashboard");

}]

}

})

**HUGELY IMPORTANT GREP COMMAND**

**grep "def get\_model\_json" . -r --color**

**FLASK**

@app.route('/assets/impacts', methods=['GET'])

@mimerender(

default = 'json',

override\_input\_key = 'format',

xml = lambda \*\*kwargs: util.render\_xml('asset', kwargs),

json = util.render\_json)

def assets\_model\_impacts():

**find /path/to/folder -iname "\*.bin" -o -iname "\*.sh"**

find . -iname "\*.json"

**find . -type f -name "TOP IMPACTS\*"**

<**script** src="https://d3js.org/d3.v4.js"></**script**>

**`**