Sprint 4

	Internal dataset: Stanford	External dataset: Intermountain
Metric (AUROC) [95% CI]		
PENet kinetics pretrained	0.84 [0.82–0.87]	0.85 [0.81–0.88]
PENet no pretraining	0.69 [0.74–0.65]	0.62 [0.57–0.88]
ResNet3D-50 kinetics pretrained	0.78 [0.74–0.81]	0.77 [0.74–0.80]
ResNeXt3D-101 kinetics pretrained	0.80 [0.77–0.82]	0.83 [0.81–0.85]
DenseNet3D-121 kinetics pretrained	0.69 [0.64–0.73]	0.67 [0.63–0.71]

https://www.nature.com/articles/s41746-020-0266-y

Accuracy = $\frac{\text{(true positives + true negatives)}}{\text{(true positives + true negatives + false positives + false negatives)}}$

Precision

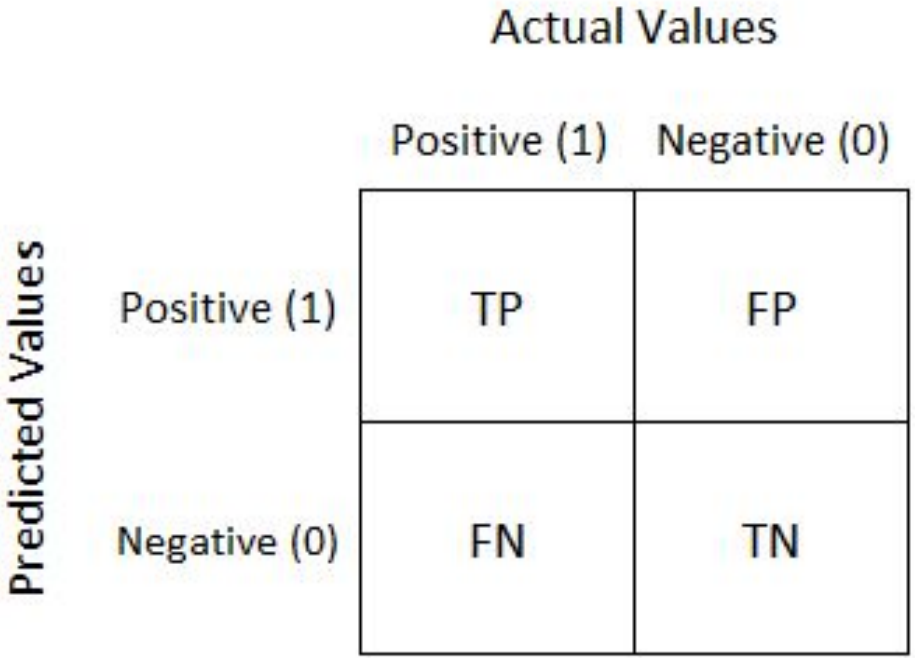
(true positives)/(true positives + false positives)

Recall

(true positives)/(true positives + false negatives)

Specificity

(true negatives)/(true negatives + false positives)



https://www.mygreatlearning.com/blog/confusion-matrix-an-overview-with-python-and-r/

Results Summary

P = positive for PE
N = negative for PE

P+N = Total

		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	0	0
Predicted Values	Negative (0)	P	N

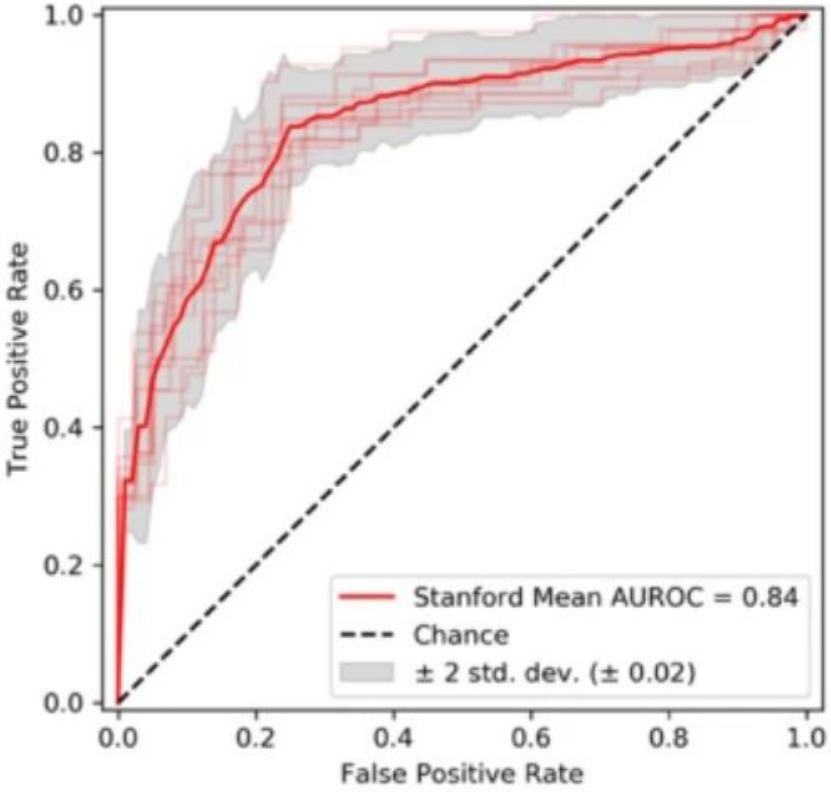
Generate Pkl file and hdf5 file from RSNA dataset

```
[meganmp@scc-c13 meganmp]$ cd test
[meganmp@scc-c13 test]$ ls
archive  data.hdf5  series_list.pkl
```

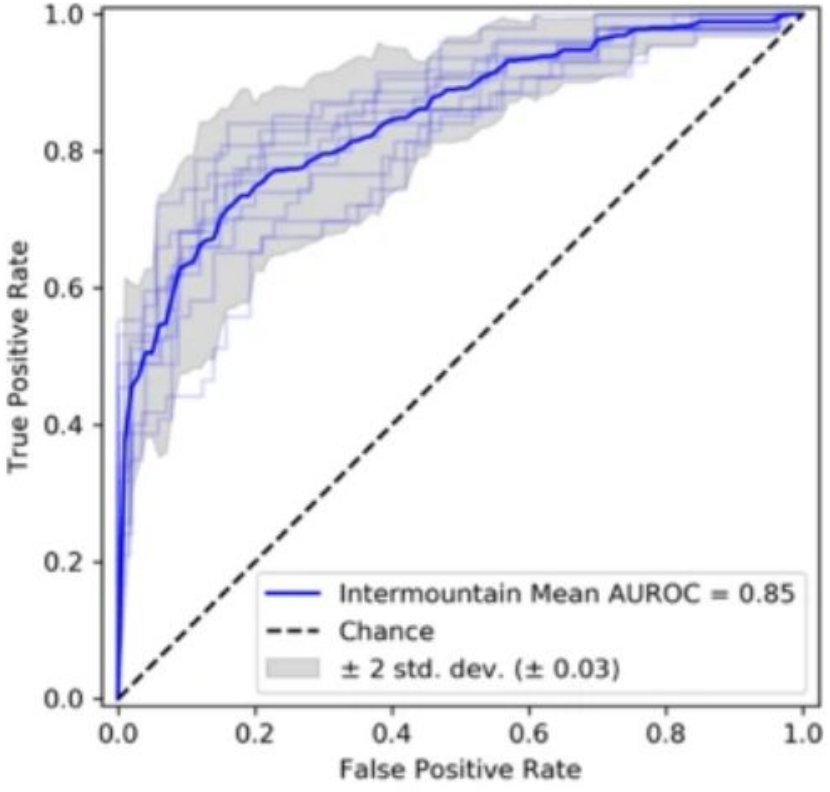
Produce preds.pkl

```
>>> for key in data.keys():
...     print('{}\t{}'.format(key, data[key]))
...
b7790ec38aaf    {'label': 0, 'pred': 0.8295624256134033}
0402afec0b08    {'label': 0, 'pred': 0.2151593267917633}
9b555f06f486    {'label': 0, 'pred': 0.3412463068962097}
af5cf805065e    {'label': 0, 'pred': 0.37278223037719727}
4e28d73a20b7    {'label': 0, 'pred': 0.3647671043872833}
e691f739c418    {'label': 0, 'pred': 0.511766791343689}
cf08033f31ea    {'label': 0, 'pred': 0.1854744553565979}
557199635c70    {'label': 0, 'pred': 0.24036115407943726}
93679ed29ca8    {'label': 0, 'pred': 0.5432394742965698}
90b012397be4    {'label': 0, 'pred': 0.4250437915325165}
69b3e1e314f5    {'label': 0, 'pred': 0.4437997043132782}
7153d75d63f2    {'label': 0, 'pred': 0.28959113359451294}
```

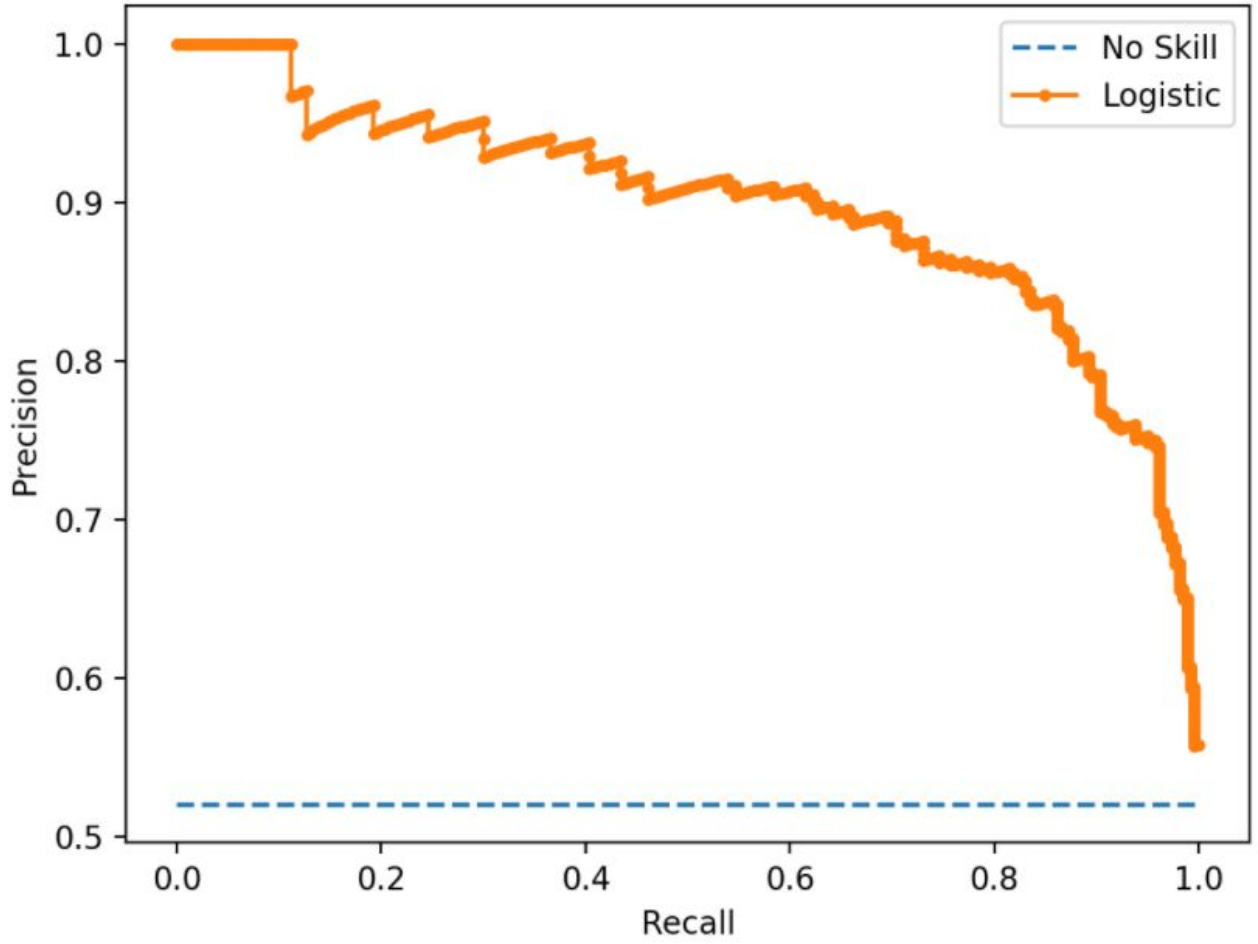
Fig. 1: PENet performance on independent test datasets.



(a)



(b)



https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

Overview of Progress and Plans

- PENet is state-of-the-art (done)
  - Use PENet to predict PE from raw DICOMs
  - Use PENet algorithm to generate probabilities using specially-preprocessed input
  - Use PENet with **minimally-processed** inputs
  - Goal: Use PENet to generate a datapoint for use in algorithm

- Comparisons using alternate algorithms (in-progress)
  - Means of comparison (ROC AUC, Precision-Recall AUC)
  - Statistical methods (from literature)

- Generate CAMs (in-progress)
  - Visualization

- **Goal**: Use PENet output as datapoint, build algorithm that gives study-level predictions about PE (acute, chronic, etc) given clinically-relevant inputs, and provide a visualization to clinicians