(1) For running code for the trade-off between privacy and feature importance:

Run featimp_vs_privacy.py

The default setup will train a non-private classifier, under which the feature importance will be learned five times with different seeds, to see if there are variabilities among the learned features.

If you want to change to private classifiers, change the —dp-sigma argument to one of these:

```
parser.add_argument('--dp-sigma', type=float, default=1.35)
parser.add_argument('--dp-sigma', type=float, default=2.3)
parser.add_argument('--dp-sigma', type=float, default=4.4)
parser.add_argument('--dp-sigma', type=float, default=8.4)
parser.add_argument('--dp-sigma', type=float, default=17.)
```

```
where each noise level corresponds to these privacy (epsilon) level

# sig = 1.35 -> eps 8.07 ~= 8
# sig = 2.3 -> eps 4.01  ~= 4
# sig = 4.4 -> eps 1.94  ~= 2
# sig = 8.4 -> eps 0.984 ~= 1
# sig = 17. -> eps 0.48  ~= 0.5
```

(2) For running code for the trade-off between fairness and feature importance:

First run vfairness_weight_readout.py to train a classifier with/without fairness constraints. In the main function of the script, choose either of the following choices.

```
# for baseline classifier training
train_baseline(*data)

# for fair classifier training with varying fairness
# train_fair(*data, T_iter=60)
# train_fair(*data, T_iter=125)
# train_fair(*data, T_iter=185)
# train_fair(*data, T_iter=250)
```

Then, run pytorch_fair_models.py to learn the feature importance under the classifier that was trained with/without fairness constraints in vfairness_weight_readout.py. In the main function of the script, choose either of the following choices. We learn the feature importance under the same classifier five times with different seeds.

If the baseline classifier is used, then set baseline = True
Otherwise, set it to False, then choose T_iter either of these [60, 125, 185, 250]

```
baseline = False  # for baseline classifier
if baseline:
  classifier = ImportedClassifier(d_in=input_dim,
weights_file='baseline_clf.npz')
else:
  T_iter = 250 # either 60, 125, 185, or 250
  filename = 'fair_clf_' + str(T_iter) + '.npz'
  classifier = ImportedClassifier(d_in=input_dim, weights_file=filename)
```

(3) Dependencies:

```
Package                 Version
----------------------- -------------------
absl-py                 0.9.0
appnope                 0.1.0
astunparse              1.6.3
autodp                  0.1.1
backcall                0.1.0
backpack                0.1
backpack-for-pytorch    1.1.1
cachetools              4.1.0
certifi                 2020.4.5.1
cffi                    1.14.0
chardet                 3.0.4
cycler                  0.10.0
decorator               4.4.2
future                  0.18.2
gast                    0.3.3
google-auth             1.16.0
google-auth-oauthlib    0.4.1
google-pasta            0.2.0
grpcio                  1.29.0
h5py                    2.10.0
idna                    2.9
importlib-metadata      1.6.0
ipython                 7.14.0
ipython-genutils        0.2.0
jedi                    0.17.0
joblib                  0.14.1
Keras                   2.3.1
Keras-Applications      1.0.8
Keras-Preprocessing     1.1.2
kiwisolver              1.2.0
Markdown                3.2.2
matplotlib              3.1.3
mkl-fft                 1.0.15
mkl-random              1.1.0
mkl-service             2.3.0
networkx                2.4
numpy                   1.18.1
oauthlib                3.1.0
```

```
opt-einsum             3.2.1
pandas                 1.0.3
parso                  0.7.0
pexpect                4.8.0
pickleshare            0.7.5
Pillow                 7.1.2
pip                    20.0.2
pomegranate            0.11.2
prompt-toolkit         3.0.5
protobuf               3.12.2
ptyprocess             0.6.0
pyasn1                 0.4.8
pyasn1-modules         0.2.8
pycparser              2.20
Pygments               2.6.1
pyparsing              2.4.7
python-dateutil        2.8.1
pytz                   2020.1
PyYAML                 5.3.1
requests               2.23.0
requests-oauthlib      1.3.0
rsa                    4.0
scikit-learn           0.21.3
scipy                  1.4.1
sdgym                  0.2.0
seaborn                0.10.1
setuptools             46.1.3.post20200330
shap                   0.35.0
simplejson             3.17.0
six                    1.14.0
sklearn                0.0
tensorboard            2.2.2
tensorboard-plugin-wit 1.6.0.post3
tensorflow             2.2.0
tensorflow-estimator   2.2.0
termcolor              1.1.0
torch                  1.5.0
torchvision            0.6.0
tornado                6.0.4
tqdm                   4.46.0
traitlets              4.3.3
urllib3                1.25.9
wcwidth                0.1.9
Werkzeug               1.0.1
wheel                  0.34.2
wrapt                  1.12.1
xgboost                1.1.0
zipp                   3.1.0
```