In [1]:

```
import pandas as pd
from sklearn import preprocessing
```

In [2]:

```
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
```

```
Using TensorFlow backend.
```

In [3]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:

```
from sklearn.decomposition import PCA
```

In [5]:

```
train_df = pd.read_csv("wat-all.csv")
```

In [6]:

```
train_df.head()
```

Out[6]:

	time	router	outport	inport	packet_address	packet_type	flit_id	flit_type	vnet	vc	src_ni
0	7	0	2	0	0x1dc0	0	0	3	2	8	0
1	7	1	1	0	0xecf40	0	0	3	2	8	1
2	7	0	2	0	0x1dc0	0	0	3	2	8	0
3	11	5	1	3	0xecf40	0	0	3	2	8	1
4	11	1	2	4	0x1dc0	0	0	3	2	8	0

In [7]:

```
train_X = train_df.drop(columns=['packet_address','time','target'])
```

In [8]:

```
train_X
```

Out[8]:

	router	outport	inport	packet_type	flit_id	flit_type	vnet	vc	src_ni	src_router	dst_n
0	0	2	0	0	0	3	2	8	0	0	2
1	1	1	0	0	0	3	2	8	1	1	2
2	0	2	0	0	0	3	2	8	0	0	2
3	5	1	3	0	0	3	2	8	1	1	2
4	1	2	4	0	0	3	2	8	0	0	2
...	...	...	...	...	...	...	...	...	...	...	.
504035	8	3	2	1	2	1	4	16	27	11	
504036	8	3	2	1	3	1	4	16	27	11	
504037	9	4	2	1	4	2	4	16	27	11	
504038	4	3	1	1	0	0	4	16	27	11	
504039	4	3	1	1	1	1	4	16	27	11	

504040 rows × 14 columns

In [9]:

```
x = train_X.values #returns a numpy array
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
train_X = pd.DataFrame(x_scaled)
train_X
```

Out[9]:

	0	1	2	3	4	5	6	7	8	9	
0	0.000000	0.50	0.00	0.000000	0.00	1.000000	0.0	0.000000	0.000000	0.000000	0.741
1	0.066667	0.25	0.00	0.000000	0.00	1.000000	0.0	0.000000	0.032258	0.066667	0.935
2	0.000000	0.50	0.00	0.000000	0.00	1.000000	0.0	0.000000	0.000000	0.000000	0.741
3	0.333333	0.25	0.75	0.000000	0.00	1.000000	0.0	0.000000	0.032258	0.066667	0.935
4	0.066667	0.50	1.00	0.000000	0.00	1.000000	0.0	0.000000	0.000000	0.000000	0.741
...	...	...	...	...	...	...	...	...	...	...	
504035	0.533333	0.75	0.50	0.333333	0.50	0.333333	1.0	0.727273	0.870968	0.733333	0.000
504036	0.533333	0.75	0.50	0.333333	0.75	0.333333	1.0	0.727273	0.870968	0.733333	0.000
504037	0.600000	1.00	0.50	0.333333	1.00	0.666667	1.0	0.727273	0.870968	0.733333	0.000
504038	0.266667	0.75	0.25	0.333333	0.00	0.000000	1.0	0.727273	0.870968	0.733333	0.000
504039	0.266667	0.75	0.25	0.333333	0.25	0.333333	1.0	0.727273	0.870968	0.733333	0.000

504040 rows × 14 columns

In [10]:

```
corr_df = pd.concat([train_X, train_df[['target']]], axis = 1)
corr_df.corr()
```

Out[10]:

	0	1	2	3	4	5	6	7
0	1.000000	0.311228	0.047617	-0.042331	2.395434e-03	-0.003780	0.178998	0.169018
1	0.311228	1.000000	-0.359564	-0.108256	6.052237e-03	-0.009565	0.482558	0.516584
2	0.047617	-0.359564	1.000000	0.111575	-6.101837e-03	0.009632	-0.488682	-0.578135
3	-0.042331	-0.108256	0.111575	1.000000	8.386949e-03	-0.013272	-0.395071	-0.430033
4	0.002395	0.006052	-0.006102	0.008387	1.000000e+00	0.079060	0.107364	0.083671
5	-0.003780	-0.009565	0.009632	-0.013272	7.906039e-02	1.000000	-0.169743	-0.132280
6	0.178998	0.482558	-0.488682	-0.395071	1.073640e-01	-0.169743	1.000000	0.977135
7	0.169018	0.516584	-0.578135	-0.430033	8.367094e-02	-0.132280	0.977135	1.000000
8	0.297251	0.509907	-0.490140	-0.211123	1.244433e-02	-0.019663	0.930521	0.902311
9	0.425807	0.459022	-0.403628	-0.175386	1.037869e-02	-0.016404	0.768846	0.735741
10	-0.110032	-0.485092	0.515266	0.210071	-1.247589e-02	0.019717	-0.931435	-0.914002
11	0.016156	-0.401589	0.464305	0.173351	-1.048132e-02	0.016574	-0.773016	-0.765335
12	0.025157	-0.016820	0.016370	0.045874	-2.262283e-05	-0.000024	-0.041155	-0.045096
13	-0.039097	0.010323	0.000941	-0.010380	6.820495e-07	0.000011	0.009316	0.011594
target	0.002485	0.003407	0.007999	0.007953	-3.565070e-06	0.000019	-0.007137	-0.008493

In [11]:

```
train_Y = train_df['target']
```

In [12]:

```
train_Y
```

Out[12]:

```
0         1
1         0
2         0
3         0
4         0
         ..
504035    1
504036    1
504037    1
504038    1
504039    1
Name: target, Length: 504040, dtype: int64
```

In [13]:

```
pca = PCA(0.95)
```

In [14]:

```
pca.fit(train_X)
```

Out[14]:

```
PCA(copy=True, iterated_power='auto', n_components=0.95, random_state=None,
    svd_solver='auto', tol=0.0, whiten=False)
```

In [15]:

```
principal_components = pca.transform(train_X)
principal_components
```

Out[15]:

```
array([[-0.70976983, -0.30256686,  0.5625034 , ...,  0.59180449,
        -0.2500026 , -0.10373356],
       [-0.92900455, -0.27575105,  0.59820052, ...,  0.63172852,
        -0.3748822 ,  0.54920917],
       [-0.71039771, -0.30538929,  0.51702793, ...,  0.60664799,
        -0.32670836, -0.30636617],
       ...,
       [ 1.09518458,  0.52966508, -0.66684601, ...,  0.10928375,
         0.27870327,  0.14695868],
       [ 1.06930523, -0.50242002, -0.17903966, ...,  0.0156443 ,
         0.11929607,  0.18611297],
       [ 1.06586005, -0.20939815, -0.23565379, ...,  0.07345195,
         0.12245066,  0.18678033]])
```

In [16]:

```
pca.explained_variance_ratio_
```

Out[16]:

```
array([0.52095204, 0.07945799, 0.0733298 , 0.06675942, 0.06053617,
       0.0551926 , 0.04009319, 0.03892328, 0.03432273])
```

In [17]:

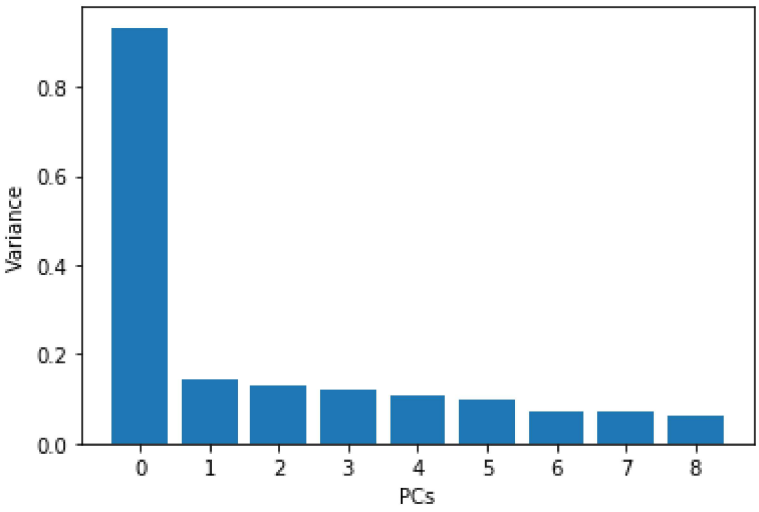
```
features = range(pca.n_components_)
```

In [18]:

```
plt.bar(features, pca.explained_variance_)
plt.xticks(features)
plt.xlabel("PCs")
plt.ylabel("Variance")
```

Out[18]:

```
Text(0, 0.5, 'Variance')
```



In [19]:

```
principal_df = pd.DataFrame(data = principal_components)
principal_df
```

Out[19]:

	0	1	2	3	4	5	6	7	
0	-0.709770	-0.302567	0.562503	0.240608	0.779657	0.003024	0.591804	-0.250003	-0.
1	-0.929005	-0.275751	0.598201	0.391826	0.655477	-0.025954	0.631729	-0.374882	0.
2	-0.710398	-0.305389	0.517028	0.193729	0.775952	0.000873	0.606648	-0.326708	-0.
3	-1.075973	-0.277781	0.275460	0.695061	0.227729	-0.353010	0.646979	-0.060663	0.
4	-0.926192	-0.287472	0.244392	0.482234	0.235926	-0.437892	0.627470	0.183611	-0.
...	...	...	...	...	...	...	...	...	
504035	1.033983	0.020426	-0.465089	-0.181757	-0.229354	-0.052336	0.053601	0.169584	0.
504036	1.038888	0.263618	-0.489906	-0.176130	-0.238567	-0.021640	0.028967	0.168201	0.
504037	1.095185	0.529665	-0.666846	-0.038042	0.065082	-0.071619	0.109284	0.278703	0.
504038	1.069305	-0.502420	-0.179040	-0.396903	-0.273481	0.202173	0.015644	0.119296	0
504039	1.065860	-0.209398	-0.235654	-0.359728	-0.058909	0.024330	0.073452	0.122451	0.

504040 rows × 9 columns

In [20]:

```
final_df = pd.concat([principal_df, train_df[['target']]], axis = 1)
final_df
```

Out[20]:

	0	1	2	3	4	5	6	7	
0	-0.709770	-0.302567	0.562503	0.240608	0.779657	0.003024	0.591804	-0.250003	-0.
1	-0.929005	-0.275751	0.598201	0.391826	0.655477	-0.025954	0.631729	-0.374882	0.
2	-0.710398	-0.305389	0.517028	0.193729	0.775952	0.000873	0.606648	-0.326708	-0.
3	-1.075973	-0.277781	0.275460	0.695061	0.227729	-0.353010	0.646979	-0.060663	0.
4	-0.926192	-0.287472	0.244392	0.482234	0.235926	-0.437892	0.627470	0.183611	-0.
...	...	...	...	...	...	...	...	...	
504035	1.033983	0.020426	-0.465089	-0.181757	-0.229354	-0.052336	0.053601	0.169584	0.
504036	1.038888	0.263618	-0.489906	-0.176130	-0.238567	-0.021640	0.028967	0.168201	0.
504037	1.095185	0.529665	-0.666846	-0.038042	0.065082	-0.071619	0.109284	0.278703	0.
504038	1.069305	-0.502420	-0.179040	-0.396903	-0.273481	0.202173	0.015644	0.119296	0
504039	1.065860	-0.209398	-0.235654	-0.359728	-0.058909	0.024330	0.073452	0.122451	0.

504040 rows × 10 columns

In [21]:

```
final_df.corr()
```

Out[21]:

	0	1	2	3	4	5	
0	1.000000e+00	9.309896e-15	1.074738e-14	-5.663994e-16	-8.803458e-15	-7.881656e-15	2.5
1	9.309896e-15	1.000000e+00	-1.111394e-15	-2.695383e-16	7.452213e-16	3.576386e-15	-
2	1.074738e-14	-1.111394e-15	1.000000e+00	1.951166e-15	-4.377827e-15	-3.385997e-15	1.5
3	-5.663994e-16	-2.695383e-16	1.951166e-15	1.000000e+00	4.672479e-15	4.915293e-15	-
4	-8.803458e-15	7.452213e-16	-4.377827e-15	4.672479e-15	1.000000e+00	-7.156466e-15	1.(
5	-7.881656e-15	3.576386e-15	-3.385997e-15	4.915293e-15	-7.156466e-15	1.000000e+00	4.:
6	2.928622e-15	-1.005026e-15	1.561215e-15	-1.414463e-15	1.037284e-15	4.251616e-15	1.(
7	1.500928e-15	-3.605677e-16	4.434152e-15	-5.697579e-15	1.034679e-14	1.217886e-14	3.5
8	-1.225931e-15	6.994417e-16	2.487012e-16	1.383104e-15	-3.396734e-15	-4.049006e-15	-
target	-6.784181e-03	-4.626786e-03	-6.933698e-02	-6.933465e-02	-2.901904e-03	-1.400564e-03	1.:

In [22]:

```
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2D PCA', fontsize = 20)
targets = [1, 0]
colors = ['r', 'g']
for target, color in zip(targets,colors):
    indicesToKeep = final_df['target'] == target
    ax.scatter(final_df.loc[indicesToKeep, 0]
               , final_df.loc[indicesToKeep, 1]
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
```

```
Error in callback <function install_repl_displayhook.<locals>.post_execute at
0x0000026C34AAA5E8> (for post_execute):
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\Anaconda3\lib\site-packages\matplotlib\pyplot.py in post_execute()
    107             def post_execute():
    108                 if matplotlib.is_interactive():
--> 109                     draw_all()
    110
    111             # IPython >= 2

~\Anaconda3\lib\site-packages\matplotlib\_pylab_helpers.py in draw_all(cls, f
orce)
    126         for f_mgr in cls.get_all_fig_managers():
    127             if force or f_mgr.canvas.figure.stale:
--> 128                 f_mgr.canvas.draw_idle()
    129
    130 atexit.register(Gcf.destroy_all)

~\Anaconda3\lib\site-packages\matplotlib\backend_bases.py in draw_idle(self,
 *args, **kwargs)
   1905         if not self._is_idle_drawing:
   1906             with self._idle_draw_cntx():
-> 1907                 self.draw(*args, **kwargs)
   1908
   1909     def draw_cursor(self, event):

~\Anaconda3\lib\site-packages\matplotlib\backends\backend_agg.py in draw(sel
f)
    386         self.renderer = self.get_renderer(cleared=True)
    387         with RendererAgg.lock:
--> 388             self.figure.draw(self.renderer)
    389             # A GUI class may be need to update a window using this d
raw, so
    390             # don't forget to call the superclass.

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\figure.py in draw(self, renderer)
   1707             self.patch.draw(renderer)
   1708             mimage._draw_list_compositing_images(
-> 1709                 renderer, self, artists, self.suppressComposite)
   1710
   1711             renderer.close_group('figure')

~\Anaconda3\lib\site-packages\matplotlib\image.py in _draw_list_compositing_i
mages(renderer, parent, artists, suppress_composite)
    133     if not_composite or not has_images:
    134         for a in artists:
--> 135             a.draw(renderer)
    136     else:
    137         # Composite any adjacent images together
```

```
~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\axes\_base.py in draw(self, rendere
r, inframe)
   2645             renderer.stop_rasterizing()
   2646
-> 2647         mimage._draw_list_compositing_images(renderer, self, artists)
   2648
   2649         renderer.close_group('axes')

~\Anaconda3\lib\site-packages\matplotlib\image.py in _draw_list_compositing_i
mages(renderer, parent, artists, suppress_composite)
    133     if not_composite or not has_images:
    134         for a in artists:
--> 135             a.draw(renderer)
    136     else:
    137         # Composite any adjacent images together

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\legend.py in draw(self, renderer)
    668             self.legendPatch.draw(renderer)
    669
--> 670         self._legend_box.draw(renderer)
    671
    672         renderer.close_group('legend')

~\Anaconda3\lib\site-packages\matplotlib\offsetbox.py in draw(self, renderer)
    254                                                       renderer)
    255
--> 256         px, py = self.get_offset(width, height, xdescent, ydescent, r
enderer)
    257
    258         for c, (ox, oy) in zip(self.get_visible_children(), offsets):

~\Anaconda3\lib\site-packages\matplotlib\offsetbox.py in get_offset(self, wid
th, height, xdescent, ydescent, renderer)
    194         """
    195         return (self._offset(width, height, xdescent, ydescent, rende
rer)
--> 196                 if callable(self._offset)
    197                 else self._offset)
    198

~\Anaconda3\lib\site-packages\matplotlib\legend.py in _findoffset(self, widt
```

```
h, height, xdescent, ydescent, renderer)
    625
    626         if self._loc == 0:  # "best".
--> 627             x, y = self._find_best_position(width, height, renderer)
    628         elif self._loc in Legend.codes.values():  # Fixed location.
    629             bbox = Bbox.from_bounds(0, 0, width, height)

~\Anaconda3\lib\site-packages\matplotlib\legend.py in _find_best_position(sel
f, width, height, renderer, consider)
   1149                         + legendBox.count_contains(offsets)
   1150                         + legendBox.count_overlaps(bboxes)
-> 1151                         + sum(line.intersects_bbox(legendBox, filled=F
alse)
   1152                               for line in lines))
   1153             if badness == 0:

~\Anaconda3\lib\site-packages\matplotlib\transforms.py in count_contains(sel
f, vertices)
    636         if len(vertices) == 0:
    637             return 0
--> 638         vertices = np.asarray(vertices)
    639         with np.errstate(invalid='ignore'):
    640             return (((self.min < vertices) &

~\Anaconda3\lib\site-packages\numpy\core\_asarray.py in asarray(a, dtype, ord
er)
     83
     84     """
---> 85     return array(a, dtype, copy=False, order=order)
     86
     87

KeyboardInterrupt:


Error in callback <function flush_figures at 0x0000026C34AB48B8> (for post_ex
ecute):
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\Anaconda3\lib\site-packages\ipykernel\pylab\backend_inline.py in flush_figu
res()
    115         # ignore the tracking, just draw and close all figures
    116         try:
--> 117             return show(True)
    118         except Exception as e:
    119             # safely show traceback if in IPython, else raise

~\Anaconda3\lib\site-packages\ipykernel\pylab\backend_inline.py in show(clos
e, block)
     37             display(
     38                 figure_manager.canvas.figure,
---> 39                 metadata=_fetch_figure_metadata(figure_manager.canvas
.figure)
     40             )
     41     finally:

~\Anaconda3\lib\site-packages\IPython\core\display.py in display(include, exc
lude, metadata, transient, display_id, *objs, **kwargs)
    304             publish_display_data(data=obj, metadata=metadata, **kwarg
s)
    305         else:
--> 306             format_dict, md_dict = format(obj, include=include, exclu
de=exclude)
    307             if not format_dict:
    308                 # nothing to display (e.g. _ipython_display_ took ove
r)

~\Anaconda3\lib\site-packages\IPython\core\formatters.py in format(self, obj,
include, exclude)
    178             md = None
    179             try:
--> 180                 data = formatter(obj)
    181             except:
    182                 # FIXME: log the exception

<C:\Users\DELL\Anaconda3\lib\site-packages\decorator.py:decorator-gen-9> in _
_call__(self, obj)

~\Anaconda3\lib\site-packages\IPython\core\formatters.py in catch_format_erro
r(method, self, *args, **kwargs)
    222     """show traceback on failed format call"""
    223     try:
--> 224         r = method(self, *args, **kwargs)
    225     except NotImplementedError:
    226         # don't warn on NotImplementedErrors

~\Anaconda3\lib\site-packages\IPython\core\formatters.py in __call__(self, ob
j)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)
```

```
~\Anaconda3\lib\site-packages\IPython\core\pylabtools.py in <lambda>(fig)
    242
    243     if 'png' in formats:
--> 244         png_formatter.for_type(Figure, lambda fig: print_figure(fig,
'png', **kwargs))
    245     if 'retina' in formats or 'png2x' in formats:
    246         png_formatter.for_type(Figure, lambda fig: retina_figure(fig,
**kwargs))

~\Anaconda3\lib\site-packages\IPython\core\pylabtools.py in print_figure(fig,
fmt, bbox_inches, **kwargs)
    126
    127     bytes_io = BytesIO()
--> 128     fig.canvas.print_figure(bytes_io, **kw)
    129     data = bytes_io.getvalue()
    130     if fmt == 'svg':

~\Anaconda3\lib\site-packages\matplotlib\backend_bases.py in print_figure(sel
f, filename, dpi, facecolor, edgecolor, orientation, format, bbox_inches, **k
wargs)
   2080                     orientation=orientation,
   2081                     bbox_inches_restore=_bbox_inches_restore,
-> 2082                     **kwargs)
   2083             finally:
   2084                 if bbox_inches and restore_bbox:

~\Anaconda3\lib\site-packages\matplotlib\backends\backend_agg.py in print_png
(self, filename_or_obj, metadata, pil_kwargs, *args, **kwargs)
    525
    526         else:
--> 527             FigureCanvasAgg.draw(self)
    528             renderer = self.get_renderer()
    529             with cbook._setattr_cm(renderer, dpi=self.figure.dpi), \

~\Anaconda3\lib\site-packages\matplotlib\backends\backend_agg.py in draw(sel
f)
    386         self.renderer = self.get_renderer(cleared=True)
    387         with RendererAgg.lock:
--> 388             self.figure.draw(self.renderer)
    389             # A GUI class may be need to update a window using this d
raw, so
    390             # don't forget to call the superclass.

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\figure.py in draw(self, renderer)
   1707             self.patch.draw(renderer)
   1708             mimage._draw_list_compositing_images(
-> 1709                 renderer, self, artists, self.suppressComposite)
   1710
   1711             renderer.close_group('figure')
```

```
~\Anaconda3\lib\site-packages\matplotlib\image.py in _draw_list_compositing_i
mages(renderer, parent, artists, suppress_composite)
    133     if not_composite or not has_images:
    134         for a in artists:
--> 135             a.draw(renderer)
    136     else:
    137         # Composite any adjacent images together

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\axes\_base.py in draw(self, rendere
r, inframe)
   2645             renderer.stop_rasterizing()
   2646
-> 2647         mimage._draw_list_compositing_images(renderer, self, artists)
   2648
   2649         renderer.close_group('axes')

~\Anaconda3\lib\site-packages\matplotlib\image.py in _draw_list_compositing_i
mages(renderer, parent, artists, suppress_composite)
    133     if not_composite or not has_images:
    134         for a in artists:
--> 135             a.draw(renderer)
    136     else:
    137         # Composite any adjacent images together

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:

~\Anaconda3\lib\site-packages\matplotlib\collections.py in draw(self, rendere
r)
    864     def draw(self, renderer):
    865         self.set_sizes(self._sizes, self.figure.dpi)
--> 866         Collection.draw(self, renderer)
    867
    868

~\Anaconda3\lib\site-packages\matplotlib\artist.py in draw_wrapper(artist, re
nderer, *args, **kwargs)
     36                 renderer.start_filter()
     37
---> 38             return draw(artist, renderer, *args, **kwargs)
     39         finally:
     40             if artist.get_agg_filter() is not None:
```

```
~\Anaconda3\lib\site-packages\matplotlib\collections.py in draw(self, rendere
r)
    329                 self._offset_position)
    330
--> 331         gc.restore()
    332         renderer.close_group(self.__class__.__name__)
    333         self.stale = False

KeyboardInterrupt:
```

In [ ]:

```
sns.pairplot(final_df.loc[:,final_df.dtypes == 'float64'])
```

In [21]:

```
corr_df[corr_df.duplicated()].shape
```

Out[21]:

```
(0, 15)
```

In [22]:

```
dup_df = train_df.drop(columns=['packet_address_id','packet_address','time'])
dup_df
```

Out[22]:

	router	outport	inport	packet_type	flit_id	flit_type	vnet	vc	src_ni	src_router	dst_n
0	0	2	0	0	0	3	2	8	0	0	2
1	1	1	0	0	0	3	2	8	1	1	2
2	0	2	0	0	0	3	2	8	0	0	2
3	5	1	3	0	0	3	2	8	1	1	2
4	1	2	4	0	0	3	2	8	0	0	2
...	...	...	...	...	...	...	...	...	...	...	...
504035	8	3	2	1	2	1	4	16	27	11	
504036	8	3	2	1	3	1	4	16	27	11	
504037	9	4	2	1	4	2	4	16	27	11	
504038	4	3	1	1	0	0	4	16	27	11	
504039	4	3	1	1	1	1	4	16	27	11	

504040 rows × 14 columns

In [23]:

```
dup_df[dup_df.duplicated()].shape
```

Out[23]:

```
(0, 14)
```

In [24]:

```
dup_df[dup_df.duplicated()].count()
```

Out[24]:

```
router         0
outport        0
inport         0
packet_type    0
flit_id        0
flit_type      0
vnet           0
vc             0
src_ni         0
src_router     0
dst_ni         0
dst_router     0
enq_time       0
target         0
dtype: int64
```

In [25]:

```
principal_df.shape
```

Out[25]:

```
(504040, 9)
```

In [26]:

```
model = Sequential()
```

In [27]:

```
n_cols = principal_df.shape[1]
n_cols
```

Out[27]:

```
9
```

In [28]:

```
model.add(Dense(32, activation='relu', input_shape=(n_cols,)))
model.add(Dense(16, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(4, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

In [29]:

```
model.compile(optimizer='sgd', loss='mean_squared_error', metrics=['accuracy'])
```

In [30]:

```
early_stopping_monitor = EarlyStopping(patience=20)
```

In [31]:

```
model.fit(principal_df, train_Y, epochs=30, validation_split=0.2, callbacks=[early_stoppin
g_monitor])
```

```
Train on 403232 samples, validate on 100808 samples
Epoch 1/30
403232/403232 [==============================] - 32s 80us/step - loss: 0.2470
- accuracy: 0.5567 - val_loss: 0.2425 - val_accuracy: 0.5850
Epoch 2/30
403232/403232 [==============================] - 32s 79us/step - loss: 0.2304
- accuracy: 0.6242 - val_loss: 0.2419 - val_accuracy: 0.6421
Epoch 3/30
403232/403232 [==============================] - 31s 78us/step - loss: 0.2072
- accuracy: 0.6680 - val_loss: 0.2783 - val_accuracy: 0.6457
Epoch 4/30
403232/403232 [==============================] - 33s 81us/step - loss: 0.1892
- accuracy: 0.7131 - val_loss: 0.3057 - val_accuracy: 0.6470
Epoch 5/30
403232/403232 [==============================] - 31s 78us/step - loss: 0.1705
- accuracy: 0.7356 - val_loss: 0.3238 - val_accuracy: 0.6480
Epoch 6/30
403232/403232 [==============================] - 32s 79us/step - loss: 0.1641
- accuracy: 0.7474 - val_loss: 0.3240 - val_accuracy: 0.6460
Epoch 7/30
403232/403232 [==============================] - 31s 76us/step - loss: 0.1588
- accuracy: 0.7581 - val_loss: 0.3243 - val_accuracy: 0.6424
Epoch 8/30
403232/403232 [==============================] - 42s 103us/step - loss: 0.149
9 - accuracy: 0.7786 - val_loss: 0.3210 - val_accuracy: 0.6487
Epoch 9/30
403232/403232 [==============================] - 344s 853us/step - loss: 0.13
87 - accuracy: 0.8010 - val_loss: 0.3139 - val_accuracy: 0.6513
Epoch 10/30
403232/403232 [==============================] - 30s 75us/step - loss: 0.1321
- accuracy: 0.8124 - val_loss: 0.3166 - val_accuracy: 0.6468
Epoch 11/30
403232/403232 [==============================] - 31s 76us/step - loss: 0.1271
- accuracy: 0.8223 - val_loss: 0.3165 - val_accuracy: 0.6490
Epoch 12/30
403232/403232 [==============================] - 31s 77us/step - loss: 0.1219
- accuracy: 0.8327 - val_loss: 0.3259 - val_accuracy: 0.6390
Epoch 13/30
403232/403232 [==============================] - 31s 76us/step - loss: 0.1176
- accuracy: 0.8402 - val_loss: 0.3281 - val_accuracy: 0.6337
Epoch 14/30
403232/403232 [==============================] - 30s 75us/step - loss: 0.1140
- accuracy: 0.8450 - val_loss: 0.3365 - val_accuracy: 0.6206
Epoch 15/30
403232/403232 [==============================] - 30s 74us/step - loss: 0.1111
- accuracy: 0.8499 - val_loss: 0.3226 - val_accuracy: 0.6234
Epoch 16/30
403232/403232 [==============================] - 30s 75us/step - loss: 0.1086
- accuracy: 0.8533 - val_loss: 0.3124 - val_accuracy: 0.5949
Epoch 17/30
403232/403232 [==============================] - 30s 73us/step - loss: 0.1057
- accuracy: 0.8576 - val_loss: 0.2849 - val_accuracy: 0.5937
Epoch 18/30
403232/403232 [==============================] - 29s 71us/step - loss: 0.1031
- accuracy: 0.8610 - val_loss: 0.2422 - val_accuracy: 0.6273
Epoch 19/30
403232/403232 [==============================] - 29s 71us/step - loss: 0.1009
```

```
- accuracy: 0.8643 - val_loss: 0.2419 - val_accuracy: 0.6103
Epoch 20/30
403232/403232 [==============================] - 29s 71us/step - loss: 0.0991
- accuracy: 0.8671 - val_loss: 0.2536 - val_accuracy: 0.6143
Epoch 21/30
403232/403232 [==============================] - 29s 72us/step - loss: 0.0980
- accuracy: 0.8688 - val_loss: 0.2580 - val_accuracy: 0.6509
Epoch 22/30
403232/403232 [==============================] - 28s 69us/step - loss: 0.0968
- accuracy: 0.8714 - val_loss: 0.2604 - val_accuracy: 0.6478
Epoch 23/30
403232/403232 [==============================] - 28s 69us/step - loss: 0.0961
- accuracy: 0.8722 - val_loss: 0.2585 - val_accuracy: 0.6540
Epoch 24/30
403232/403232 [==============================] - 27s 68us/step - loss: 0.0950
- accuracy: 0.8741 - val_loss: 0.2535 - val_accuracy: 0.6492
Epoch 25/30
403232/403232 [==============================] - 28s 69us/step - loss: 0.0942
- accuracy: 0.8751 - val_loss: 0.2587 - val_accuracy: 0.6547
Epoch 26/30
403232/403232 [==============================] - 27s 68us/step - loss: 0.0934
- accuracy: 0.8765 - val_loss: 0.2661 - val_accuracy: 0.6254
Epoch 27/30
403232/403232 [==============================] - 27s 67us/step - loss: 0.0927
- accuracy: 0.8776 - val_loss: 0.2446 - val_accuracy: 0.6767
Epoch 28/30
403232/403232 [==============================] - 27s 67us/step - loss: 0.0922
- accuracy: 0.8784 - val_loss: 0.2519 - val_accuracy: 0.6741
Epoch 29/30
403232/403232 [==============================] - 27s 66us/step - loss: 0.0915
- accuracy: 0.8793 - val_loss: 0.2557 - val_accuracy: 0.6648
Epoch 30/30
403232/403232 [==============================] - 27s 66us/step - loss: 0.0911
- accuracy: 0.8796 - val_loss: 0.2399 - val_accuracy: 0.6830
```

Out[31]:

```
<keras.callbacks.callbacks.History at 0x243b8f502c8>
```

In [32]:

```
pred = model.predict(principal_df)
```

In [33]:

```
for i in range(100):
    print("%s, %s" % (pred[i], train_Y[i]))
```

```
[0.95796204], 1
[3.3756245e-05], 0
[4.0617696e-08], 0
[5.098494e-05], 0
[6.368791e-08], 0
[0.95796204], 1
[5.2459534e-08], 0
[5.824976e-05], 0
[0.95796204], 1
[6.286713e-05], 0
[3.1698743e-08], 0
[0.95796204], 1
[0.95796204], 1
[2.3150506e-08], 0
[4.8920988e-08], 0
[7.154863e-08], 0
[5.025776e-08], 0
[4.086582e-08], 0
[1.6664929e-07], 0
[1.2175097e-08], 0
[0.95796204], 1
[0.95796204], 1
[2.1891141e-07], 0
[1.5518667e-08], 0
[0.95796204], 1
[1.6854997e-08], 0
[1.5711909e-07], 0
[5.2161706e-08], 0
[1.3921041e-07], 0
[1.6388604e-08], 0
[0.95796204], 1
[2.2937643e-08], 0
[3.512227e-07], 0
[0.95796204], 1
[3.040802e-08], 0
[4.6000605e-07], 0
[0.95796204], 1
[3.311335e-07], 0
[3.0667444e-08], 0
[0.95796204], 1
[2.9339265e-07], 0
[3.092885e-08], 0
[1.5959792e-07], 0
[2.2140352e-08], 0
[0.95796204], 1
[0.95796204], 1
[2.6109465e-08], 0
[0.95796204], 1
[1.5229725e-07], 0
[3.4612516e-08], 0
[2.2503005e-07], 0
[0.95796204], 1
[1.5806773e-07], 0
[3.4908016e-08], 0
[0.95796204], 1
[0.95796204], 1
[0.95796204], 1
```

```
[1.2562947e-07], 0
[3.5205495e-08], 0
[4.1001975e-08], 0
[3.3536796e-07], 0
[0.95796204], 1
[2.6404338e-08], 0
[3.5003623e-08], 0
[0.95796204], 1
[3.5302797e-08], 0
[0.95796204], 1
[1.640578e-07], 0
[4.6671204e-08], 0
[3.560317e-08], 0
[0.95796204], 1
[0.95796204], 1
[0.95796204], 1
[3.993698e-08], 0
[3.5206234e-08], 0
[0.95796204], 1
[0.95796204], 1
[4.683511e-08], 0
[0.95796204], 1
[0.95796204], 1
[4.719893e-08], 0
[6.2306036e-08], 0
[0.95796204], 1
[5.4924932e-08], 0
[3.0882813e-08], 0
[2.8646935e-08], 0
[1.7044572e-08], 0
[2.6902924e-05], 0
[0.90480626], 1
[3.3117445e-05], 0
[0.9235403], 1
[1.0243997e-08], 0
[3.6203783e-05], 0
[0.91460073], 1
[4.9414335e-09], 0
[5.651664e-09], 0
[6.2331327e-09], 0
[6.925501e-09], 0
[1.747172e-08], 0
[0.8410988], 1
```

In [ ]: