Tensorboard

NOTE: You must watch the corresponding video to understand this lecture. This notebook can't serve as a full guide. Please watch the video BEFORE posting questions to the QA forum.

Let's explore the built in data visualization capabilities that come with Tensorboard.

Full official tutorial available here: https://www.tensorflow.org/tensorboard/get_started

Data

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv('../DATA/cancer_classification.csv')
```

Train Test Split

In [3]:
```
X = df.drop('benign_0__mal_1',axis=1).values
y = df['benign_0__mal_1'].values
```

In [4]:
```
from sklearn.model_selection import train_test_split
```

In [5]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25,random_state=101)
```

Scaling Data

In [6]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [7]:
```
scaler = MinMaxScaler()
```

In [8]:
```
scaler.fit(X_train)
```

Out[8]:
```
MinMaxScaler(copy=True, feature_range=(0, 1))
```

In [9]:
```
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Creating the Model

In [10]:
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation,Dropout
```

In [11]:
```
from tensorflow.keras.callbacks import EarlyStopping,TensorBoard
```

In [12]:
```
early_stop = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=25)
```

In [13]:
```
pwd
```

Out[13]:
```
'C:\\Users\\Marcial\\Pierian-Data-Courses\\TensorFlow-Two-Bootcamp\\03-ANNs'
```

Creating the Tensorboard Callback

TensorBoard is a visualization tool provided with TensorFlow.

This callback logs events for TensorBoard, including:

- Metrics summary plots

- Training graph visualization
- Activation histograms
- Sampled profiling

If you have installed TensorFlow with pip, you should be able to launch TensorBoard from the command line:

```
tensorboard --logdir=path_to_your_logs
```

You can find more information about TensorBoard here.

```
    Arguments:
        log_dir: the path of the directory where to save the log files to be
          parsed by TensorBoard.
        histogram_freq: frequency (in epochs) at which to compute activation and
          weight histograms for the layers of the model. If set to 0, histograms
          won't be computed. Validation data (or split) must be specified for
          histogram visualizations.
        write_graph: whether to visualize the graph in TensorBoard. The log file
          can become quite large when write_graph is set to True.
        write_images: whether to write model weights to visualize as image in
          TensorBoard.
        update_freq: `'batch'` or `'epoch'` or integer. When using `'batch'`,
          writes the losses and metrics to TensorBoard after each batch. The same
          applies for `'epoch'`. If using an integer, let's say `1000`, the
          callback will write the metrics and losses to TensorBoard every 1000
          samples. Note that writing too frequently to TensorBoard can slow down
          your training.
        profile_batch: Profile the batch to sample compute characteristics. By
          default, it will profile the second batch. Set profile_batch=0 to
          disable profiling. Must run in TensorFlow eager mode.
        embeddings_freq: frequency (in epochs) at which embedding layers will
          be visualized. If set to 0, embeddings won't be visualized.
```

In [14]:
```
from datetime import datetime
```

In [15]:
```
datetime.now().strftime("%Y-%m-%d--%H%M")
```

Out[15]:
```
'2019-11-14--1910'
```

In [16]:
```
# WINDOWS: Use "logs\\fit"
# MACOS/LINUX: Use "logs\fit"

log_directory = 'logs\\fit'

# OPTIONAL: ADD A TIMESTAMP FOR UNIQUE FOLDER
# timestamp = datetime.now().strftime("%Y-%m-%d--%H%M")
# log_directory = log_directory + '\\' + timestamp


board = TensorBoard(log_dir=log_directory,histogram_freq=1,
    write_graph=True,
    write_images=True,
    update_freq='epoch',
    profile_batch=2,
    embeddings_freq=1)
```

Now create the model layers:

In [17]:
```
model = Sequential()
model.add(Dense(units=30,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(units=15,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(units=1,activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam')
```

Train the Model

In [18]:
```
model.fit(x=X_train,
          y=y_train,
          epochs=600,
          validation_data=(X_test, y_test), verbose=1,
          callbacks=[early_stop,board]
          )
```

```
Train on 426 samples, validate on 143 samples
Epoch 1/600
426/426 [==============================] - 1s 2ms/sample - loss: 0.7070 - val_loss: 0.6751
Epoch 2/600
426/426 [==============================] - 0s 204us/sample - loss: 0.6647 - val_loss: 0.6547
```

```
Epoch 3/600
426/426 [==============================] - 0s 192us/sample - loss: 0.6735 - val_loss: 0.6312
Epoch 4/600
426/426 [==============================] - 0s 194us/sample - loss: 0.6284 - val_loss: 0.6052
Epoch 5/600
426/426 [==============================] - 0s 208us/sample - loss: 0.6023 - val_loss: 0.5716
Epoch 6/600
426/426 [==============================] - 0s 191us/sample - loss: 0.5939 - val_loss: 0.5454
Epoch 7/600
426/426 [==============================] - 0s 189us/sample - loss: 0.5745 - val_loss: 0.5210
Epoch 8/600
426/426 [==============================] - 0s 214us/sample - loss: 0.5402 - val_loss: 0.4942
Epoch 9/600
426/426 [==============================] - 0s 206us/sample - loss: 0.5410 - val_loss: 0.4676
Epoch 10/600
426/426 [==============================] - 0s 195us/sample - loss: 0.5075 - val_loss: 0.4402
Epoch 11/600
426/426 [==============================] - 0s 198us/sample - loss: 0.4726 - val_loss: 0.4098
Epoch 12/600
426/426 [==============================] - 0s 204us/sample - loss: 0.4418 - val_loss: 0.3778
Epoch 13/600
426/426 [==============================] - 0s 201us/sample - loss: 0.4381 - val_loss: 0.3497
Epoch 14/600
426/426 [==============================] - 0s 201us/sample - loss: 0.4192 - val_loss: 0.3232
Epoch 15/600
426/426 [==============================] - 0s 204us/sample - loss: 0.4235 - val_loss: 0.3065
Epoch 16/600
426/426 [==============================] - 0s 192us/sample - loss: 0.3875 - val_loss: 0.2917
Epoch 17/600
426/426 [==============================] - 0s 197us/sample - loss: 0.3904 - val_loss: 0.2776
Epoch 18/600
426/426 [==============================] - 0s 194us/sample - loss: 0.3685 - val_loss: 0.2645
Epoch 19/600
426/426 [==============================] - 0s 197us/sample - loss: 0.3247 - val_loss: 0.2463
Epoch 20/600
426/426 [==============================] - 0s 199us/sample - loss: 0.3428 - val_loss: 0.2332
Epoch 21/600
426/426 [==============================] - 0s 199us/sample - loss: 0.3435 - val_loss: 0.2247
Epoch 22/600
426/426 [==============================] - 0s 200us/sample - loss: 0.3304 - val_loss: 0.2188
Epoch 23/600
426/426 [==============================] - 0s 195us/sample - loss: 0.3187 - val_loss: 0.2071
Epoch 24/600
426/426 [==============================] - 0s 195us/sample - loss: 0.2977 - val_loss: 0.1962
Epoch 25/600
426/426 [==============================] - 0s 213us/sample - loss: 0.2878 - val_loss: 0.1892
Epoch 26/600
426/426 [==============================] - 0s 186us/sample - loss: 0.2792 - val_loss: 0.1797
Epoch 27/600
426/426 [==============================] - 0s 189us/sample - loss: 0.2689 - val_loss: 0.1738
Epoch 28/600
426/426 [==============================] - 0s 197us/sample - loss: 0.2746 - val_loss: 0.1703
Epoch 29/600
426/426 [==============================] - 0s 201us/sample - loss: 0.2929 - val_loss: 0.1692
Epoch 30/600
426/426 [==============================] - 0s 215us/sample - loss: 0.2652 - val_loss: 0.1650
Epoch 31/600
426/426 [==============================] - 0s 213us/sample - loss: 0.2448 - val_loss: 0.1558
Epoch 32/600
426/426 [==============================] - 0s 188us/sample - loss: 0.2658 - val_loss: 0.1566
Epoch 33/600
426/426 [==============================] - 0s 210us/sample - loss: 0.2486 - val_loss: 0.1520
Epoch 34/600
426/426 [==============================] - 0s 206us/sample - loss: 0.2584 - val_loss: 0.1507
Epoch 35/600
426/426 [==============================] - 0s 190us/sample - loss: 0.2284 - val_loss: 0.1541
Epoch 36/600
426/426 [==============================] - 0s 190us/sample - loss: 0.2294 - val_loss: 0.1380
Epoch 37/600
426/426 [==============================] - 0s 185us/sample - loss: 0.2064 - val_loss: 0.1336
Epoch 38/600
426/426 [==============================] - 0s 189us/sample - loss: 0.2345 - val_loss: 0.1343
Epoch 39/600
426/426 [==============================] - 0s 188us/sample - loss: 0.2043 - val_loss: 0.1335
Epoch 40/600
426/426 [==============================] - 0s 189us/sample - loss: 0.1871 - val_loss: 0.1249
Epoch 41/600
426/426 [==============================] - 0s 184us/sample - loss: 0.1756 - val_loss: 0.1215
Epoch 42/600
426/426 [==============================] - 0s 185us/sample - loss: 0.1986 - val_loss: 0.1194
Epoch 43/600
426/426 [==============================] - 0s 183us/sample - loss: 0.2050 - val_loss: 0.1209
Epoch 44/600
426/426 [==============================] - 0s 183us/sample - loss: 0.1964 - val_loss: 0.1219
Epoch 45/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1825 - val_loss: 0.1156
Epoch 46/600
426/426 [==============================] - 0s 206us/sample - loss: 0.1911 - val_loss: 0.1127
Epoch 47/600
```

```
426/426 [==============================] - 0s 220us/sample - loss: 0.1897 - val_loss: 0.1142
Epoch 48/600
426/426 [==============================] - 0s 227us/sample - loss: 0.1811 - val_loss: 0.1122
Epoch 49/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1799 - val_loss: 0.1070
Epoch 50/600
426/426 [==============================] - 0s 194us/sample - loss: 0.1737 - val_loss: 0.1111
Epoch 51/600
426/426 [==============================] - 0s 183us/sample - loss: 0.1821 - val_loss: 0.1162
Epoch 52/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1623 - val_loss: 0.1014
Epoch 53/600
426/426 [==============================] - 0s 193us/sample - loss: 0.1563 - val_loss: 0.0996
Epoch 54/600
426/426 [==============================] - 0s 186us/sample - loss: 0.1674 - val_loss: 0.1011
Epoch 55/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1621 - val_loss: 0.1030
Epoch 56/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1833 - val_loss: 0.1028
Epoch 57/600
426/426 [==============================] - 0s 205us/sample - loss: 0.1428 - val_loss: 0.1005
Epoch 58/600
426/426 [==============================] - 0s 202us/sample - loss: 0.1285 - val_loss: 0.0996
Epoch 59/600
426/426 [==============================] - 0s 195us/sample - loss: 0.1368 - val_loss: 0.1043
Epoch 60/600
426/426 [==============================] - 0s 190us/sample - loss: 0.1675 - val_loss: 0.0966
Epoch 61/600
426/426 [==============================] - 0s 186us/sample - loss: 0.1635 - val_loss: 0.0933
Epoch 62/600
426/426 [==============================] - 0s 191us/sample - loss: 0.1388 - val_loss: 0.1001
Epoch 63/600
426/426 [==============================] - 0s 204us/sample - loss: 0.1271 - val_loss: 0.0933
Epoch 64/600
426/426 [==============================] - 0s 204us/sample - loss: 0.1295 - val_loss: 0.0911
Epoch 65/600
426/426 [==============================] - 0s 204us/sample - loss: 0.1386 - val_loss: 0.0917
Epoch 66/600
426/426 [==============================] - 0s 189us/sample - loss: 0.1327 - val_loss: 0.0955
Epoch 67/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1420 - val_loss: 0.1101
Epoch 68/600
426/426 [==============================] - 0s 191us/sample - loss: 0.1181 - val_loss: 0.0886
Epoch 69/600
426/426 [==============================] - 0s 190us/sample - loss: 0.1318 - val_loss: 0.0920
Epoch 70/600
426/426 [==============================] - 0s 196us/sample - loss: 0.1278 - val_loss: 0.1064
Epoch 71/600
426/426 [==============================] - 0s 205us/sample - loss: 0.1161 - val_loss: 0.0901
Epoch 72/600
426/426 [==============================] - 0s 191us/sample - loss: 0.1431 - val_loss: 0.0856
Epoch 73/600
426/426 [==============================] - 0s 189us/sample - loss: 0.1312 - val_loss: 0.0857
Epoch 74/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1042 - val_loss: 0.0957
Epoch 75/600
426/426 [==============================] - 0s 184us/sample - loss: 0.1368 - val_loss: 0.0810
Epoch 76/600
426/426 [==============================] - 0s 190us/sample - loss: 0.1294 - val_loss: 0.0955
Epoch 77/600
426/426 [==============================] - 0s 193us/sample - loss: 0.1278 - val_loss: 0.0894
Epoch 78/600
426/426 [==============================] - 0s 185us/sample - loss: 0.1362 - val_loss: 0.0830
Epoch 79/600
426/426 [==============================] - 0s 183us/sample - loss: 0.1220 - val_loss: 0.0884
Epoch 80/600
426/426 [==============================] - 0s 208us/sample - loss: 0.1277 - val_loss: 0.0870
Epoch 81/600
426/426 [==============================] - 0s 219us/sample - loss: 0.1055 - val_loss: 0.0879
Epoch 82/600
426/426 [==============================] - 0s 215us/sample - loss: 0.1111 - val_loss: 0.0882
Epoch 83/600
426/426 [==============================] - 0s 188us/sample - loss: 0.0943 - val_loss: 0.0930
Epoch 84/600
426/426 [==============================] - 0s 191us/sample - loss: 0.1182 - val_loss: 0.0827
Epoch 85/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1070 - val_loss: 0.0861
Epoch 86/600
426/426 [==============================] - 0s 181us/sample - loss: 0.1110 - val_loss: 0.1013
Epoch 87/600
426/426 [==============================] - 0s 184us/sample - loss: 0.1113 - val_loss: 0.0903
Epoch 88/600
426/426 [==============================] - 0s 185us/sample - loss: 0.1279 - val_loss: 0.0815
Epoch 89/600
426/426 [==============================] - 0s 190us/sample - loss: 0.0966 - val_loss: 0.0841
Epoch 90/600
426/426 [==============================] - 0s 206us/sample - loss: 0.1232 - val_loss: 0.0890
Epoch 91/600
426/426 [==============================] - 0s 205us/sample - loss: 0.1026 - val_loss: 0.0828
```

```
Epoch 92/600
426/426 [==============================] - 0s 190us/sample - loss: 0.0941 - val_loss: 0.0816
Epoch 93/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1079 - val_loss: 0.0825
Epoch 94/600
426/426 [==============================] - 0s 186us/sample - loss: 0.1020 - val_loss: 0.0822
Epoch 95/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1092 - val_loss: 0.0814
Epoch 96/600
426/426 [==============================] - 0s 184us/sample - loss: 0.1014 - val_loss: 0.0919
Epoch 97/600
426/426 [==============================] - 0s 197us/sample - loss: 0.1086 - val_loss: 0.0794
Epoch 98/600
426/426 [==============================] - 0s 199us/sample - loss: 0.1320 - val_loss: 0.0860
Epoch 99/600
426/426 [==============================] - 0s 211us/sample - loss: 0.0911 - val_loss: 0.0830
Epoch 100/600
426/426 [==============================] - 0s 182us/sample - loss: 0.1131 - val_loss: 0.0806
Epoch 101/600
426/426 [==============================] - 0s 205us/sample - loss: 0.1114 - val_loss: 0.0965
Epoch 102/600
426/426 [==============================] - 0s 201us/sample - loss: 0.0980 - val_loss: 0.0843
Epoch 103/600
426/426 [==============================] - 0s 199us/sample - loss: 0.1048 - val_loss: 0.0894
Epoch 104/600
426/426 [==============================] - 0s 187us/sample - loss: 0.0930 - val_loss: 0.0842
Epoch 105/600
426/426 [==============================] - 0s 188us/sample - loss: 0.1038 - val_loss: 0.0909
Epoch 106/600
426/426 [==============================] - 0s 184us/sample - loss: 0.0891 - val_loss: 0.0971
Epoch 107/600
426/426 [==============================] - 0s 187us/sample - loss: 0.0936 - val_loss: 0.0993
Epoch 108/600
426/426 [==============================] - 0s 183us/sample - loss: 0.1042 - val_loss: 0.0891
Epoch 109/600
426/426 [==============================] - 0s 196us/sample - loss: 0.1030 - val_loss: 0.0905
Epoch 110/600
426/426 [==============================] - 0s 190us/sample - loss: 0.0995 - val_loss: 0.0979
Epoch 111/600
426/426 [==============================] - 0s 187us/sample - loss: 0.1167 - val_loss: 0.0841
Epoch 112/600
426/426 [==============================] - 0s 189us/sample - loss: 0.1109 - val_loss: 0.0885
Epoch 113/600
426/426 [==============================] - 0s 186us/sample - loss: 0.0932 - val_loss: 0.0910
Epoch 114/600
426/426 [==============================] - 0s 194us/sample - loss: 0.1033 - val_loss: 0.0810
Epoch 115/600
426/426 [==============================] - 0s 213us/sample - loss: 0.1060 - val_loss: 0.0949
Epoch 116/600
426/426 [==============================] - 0s 209us/sample - loss: 0.0875 - val_loss: 0.0854
Epoch 117/600
426/426 [==============================] - 0s 198us/sample - loss: 0.1012 - val_loss: 0.0817
Epoch 118/600
426/426 [==============================] - 0s 189us/sample - loss: 0.1111 - val_loss: 0.0824
Epoch 119/600
426/426 [==============================] - 0s 185us/sample - loss: 0.0826 - val_loss: 0.0839
Epoch 120/600
426/426 [==============================] - 0s 188us/sample - loss: 0.0872 - val_loss: 0.0863
Epoch 121/600
426/426 [==============================] - 0s 186us/sample - loss: 0.0739 - val_loss: 0.0921
Epoch 122/600
426/426 [==============================] - 0s 184us/sample - loss: 0.1086 - val_loss: 0.0857
Epoch 00122: early stopping
```

Out[18]:

```
<tensorflow.python.keras.callbacks.History at 0x1b1f2eb3788>
```

Running Tensorboard

Running through the Command Line

Watch video to see how to run Tensorboard through a command line call.

Tensorboard will run locally in your browser at http://localhost:6006/

In [19]:

```
print(log_directory)
```

```
logs\fit
```

In [20]:

```
pwd
```

Out[20]:

```
'C:\\Users\\Marcial\\Pierian-Data-Courses\\TensorFlow-Two-Bootcamp\\03-ANNs'
```

Use cd at your command line to change directory to the file path reported back by pwd or your

current .py file location.

Then run this code at your command line or terminal

In [ ]:
```
tensorboard --logdir logs\fit
```