CxAODReader setup on lxplus

```
[schioppa@lxplus799:~/Reader4/CxAODReader_VVSemileptonic] stat -c %y .git/FETCH_HEAD
2021-07-13 17:57:51.000000001 +0200
```

Code version from 13-07-2021

```
setupATLAS
asetup AnalysisBase, 21.2.150, here
source build/x86_64-centos7-gcc8-opt/setup.sh
```

Environment settings. In particular, no local installation (e.g. lwtnn) → **compile**

```
# N events
int maxEvents        = -1 # -1 = all events

#Apply pile-up reweighting
bool applyPUWeight             = true
vector<string> ilumicalcFiles = GoodRunsLists/data15_13TeV/20170619/PHYS_StandardGRL_All_Good_25ns_276262-284484_
_297730-311481_OflLumi-13TeV-009.root
vector<string> configFiles     = $WorkDir_DIR/data/CxAODOperations_VHbb/PRW/PRW.VHbb.mc16a.root

# input directory where the CxAODs are stored
# HIGG5D1 = 0 leptons, HIGG5D2 = 1 lepton, HIGG2D4 = 2 leptons
#string dataset_dir = /eos/user/m/mmazza/samples/Wh_signal_samples/MC16a/
string dataset_dir = /eos/user/r/rles/CxAOD/r33-08/TCC_EMTopo/HIGG2D4/

# List of samples to run on:
#vector<string> samples = group.phys-exotics.mc16_13TeV.302341.CAOD_HIGG5D2.e7767_e5984_s3126_r9364_r9315_p3641.3
_s3126_r9364_r9315_p3641_32-16-ITADFIX-19_CxAOD.root
vector<string> samples = Diboson Signal
```

Full statistics on signal and diboson

Not going to touch this config file anymore

```
[schioppa@lxplus799:~/Reader4/run] ls -lrth output_2021-09-08_test/fetch/data-MVATree/
total 433M
-rw-r--r--. 1 schioppa zp  65K  8 set 22.51 Diboson-2.root
-rw-r--r--. 1 schioppa zp  21M  8 set 22.54 Signal-0.root
-rw-r--r--. 1 schioppa zp 147M  8 set 23.12 Diboson-1.root
-rw-r--r--. 1 schioppa zp 266M  8 set 23.23 Diboson-0.root
```

Output

Building the DNN

In [2]:
```
inputFiles = ['Signal-0','Diboson-0','Diboson-1','Diboson-2']
for i in inputFiles:
    inFile = '../ntuples/original/'+i+'.root'
    theFile = uproot3.open(inFile)
    tree = uproot3.open(inFile)['Nominal']
    Nevents = uproot3.open(inFile)['Nominal'].numentries
    print ('Number of events in '+inFile,'\t'+str(Nevents))
```

```
Number of events in ../ntuples/original/Signal-0.root   67836
Number of events in ../ntuples/original/Diboson-0.root  1184019
Number of events in ../ntuples/original/Diboson-1.root  621719
Number of events in ../ntuples/original/Diboson-2.root  100
```

Number of events

In [3]:
```
DF_Signal = uproot3.open('../ntuples/original/Signal-0.root')['Nominal'].pandas.df()
```

In [4]:
```
DF_Diboson = uproot3.open('../ntuples/original/Diboson-1.root')['Nominal'].pandas.df()
DF_Diboson.shape
```

Out[4]:
```
(621719, 225)
```

I consider just one background file (Diboson-1.root)

In [5]:
```
import sklearn.utils
def Shuffling(df):
    df = sklearn.utils.shuffle(df,random_state=123)
    df = df.reset_index(drop=True)
    return df

DF_Signal = Shuffling(DF_Signal)
DF_Diboson= Shuffling(DF_Diboson)
```

In [6]:
```
DF_Signal.insert(len(DF_Signal.columns), 'isSignal', np.ones(DF_Signal.shape[0]), True)
DF_Diboson.insert(len(DF_Diboson.columns), 'isSignal', np.zeros(DF_Diboson.shape[0]), True)

signalList = [DF_Signal[:50000] ]
dibosonList = [DF_Diboson[:50000] ]

totalPD_sig = pd.concat(signalList,ignore_index=True)
totalPD_bkg = pd.concat(dibosonList,ignore_index=True)

print ('Signal events:',totalPD_sig.shape)
print ('Background events:',totalPD_bkg.shape)

df_total = pd.concat([totalPD_sig,totalPD_bkg],ignore_index=True)
df_total = Shuffling(df_total)
print ("Total events after shuffling: ",df_total.shape)
```

I take 50k sig + 50k bkg

Building the DNN

In [7]:
```
InputFeatures = ['lep1_m', 'lep1_pt', 'lep1_eta', 'lep1_phi']#, 'lep2_m', 'lep2_pt', 'lep2_eta', 'lep2_phi', 'fatjet_
X=df_total[InputFeatures].values
y_tmp = df_total['isSignal']
eventID = df_total['eventNumber']
w = np.ones(X.shape[0])
```

In [8]:
```
from sklearn.preprocessing import StandardScaler,LabelEncoder
scaler = StandardScaler()
X = scaler.fit_transform(X)
le = LabelEncoder()
y = le.fit_transform(y_tmp)
```

In [9]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test, w_train, w_test = train_test_split(X, y, w, train_size=0.7,random_state=123)
```

In [10]:
```
from keras.models import Model, Sequential
from keras.layers import Dense, Dropout, Input, BatchNormalization
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.layers.core import Dense, Activation

def BuildDNN(N_input,width,depth):
    model = Sequential()
    model.add(Dense(units=width, input_dim=N_input))
    model.add(Activation('relu'))
    model.add(Dropout(0.2))
    # First layer
    for i in range(0, depth):
        model.add(Dense(width))
        model.add(Activation('relu'))
        # Dropout randomly sets a fraction of input units to 0 at each update during training time
        # which helps prevent overfitting.
        model.add(Dropout(0.2))
        model.add(Dense(1, activation='sigmoid'))
    # Output layer/node
    return model

n_dim=X_train.shape[1]
n_nodes = 32
n_depth = 2 # number of additional hidden layers
```

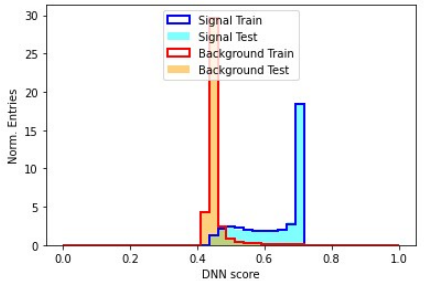
In [11]:
```
model=BuildDNN(n_dim,n_nodes,n_depth)
model.compile(loss='binary_crossentropy',optimizer='rmsprop',metrics=['accuracy'])
```

In [12]:
```
modelMetricsHistory = model.fit(X_train, y_train, epochs=4,batch_size=2048,validation_split=0.2, verbose=1)
```

Purposely downgrading the DNN (I want to have a large variation of scores)

Building the DNN

```
plt.hist(yhat_train_signal,bins=bins, histtype='step',lw=2, color='blue', label=[r'Signal Train'], density=True)
plt.hist(yhat_test_signal,bins=bins, histtype='stepfilled', lw=2, color='cyan', alpha=0.5, label=[r'Signal Test'], de
plt.hist(yhat_train_background, bins=bins, histtype='step',lw=2, color='red', label=[r'Background Train'], density=Tr
plt.hist(yhat_test_background, bins=bins, histtype='stepfilled', lw=2, color='orange', alpha=0.5, label=[r'Background
plt.ylabel('Norm. Entries')
plt.xlabel('DNN score')
# plt.yscale('log')
# plt.ylim((0,1000))
plt.legend(loc="upper center")
plt.savefig("output/scores.pdf")
plt.show()
plt.clf()
```




```
<Figure size 432x288 with 0 Axes>
```

In [20]:
```
def SaveArchAndWeights(model, outputDir):
    arch = model.to_json()
    outputArch = outputDir + '/architecture.json'
    with open(outputArch, 'w') as arch_file:
        arch_file.write(arch)
    outputWeights = outputDir + '/weights.h5'
    model.save_weights(outputWeights)

SaveArchAndWeights(model, 'output')
```

Saving the model and architecture

Building the DNN

```
X=df_total[InputFeatures].values
y_tmp = df_total['isSignal']
eventID = df_total['eventNumber']
w = np.ones(X.shape[0])
```

In [8]:
```
with open('output/variables.json', 'w') as var_file:
        var_file.write("{\n")
        var_file.write("  \"inputs\": [\n")
        index = 0
        for col in X.transpose():
            offset = -1. * np.mean(col)
            scale = 1. / np.std(col)
            print(-offset)
            var_file.write("    {\n")
            var_file.write("      \"name\": \"%s\",\n" % InputFeatures[index])
            var_file.write("      \"offset\": %lf,\n" % offset)
            var_file.write("      \"scale\": %lf\n" % scale)
            var_file.write("    }")
            if index != (len(InputFeatures)-1):
                var_file.write(",\n")
            else:
                var_file.write("\n")
            index = index + 1
        var_file.write("  ],\n")
        var_file.write("  \"class_labels\": [\"BinaryClassificationOutputName\"]\n")
        var_file.write("}\n")
```

```
0.054762840270996094
336.2005615234375
0.0004946031258441508
0.01665002666413784
```

In [9]:
```
from sklearn.preprocessing import StandardScaler,LabelEncoder
scaler = StandardScaler()
X = scaler.fit_transform(X)
le = LabelEncoder()
y = le.fit_transform(y_tmp)
print(scaler.mean_)
```

```
[5.47628420e-02 3.36200609e+02 4.94603151e-04 1.66500305e-02]
```

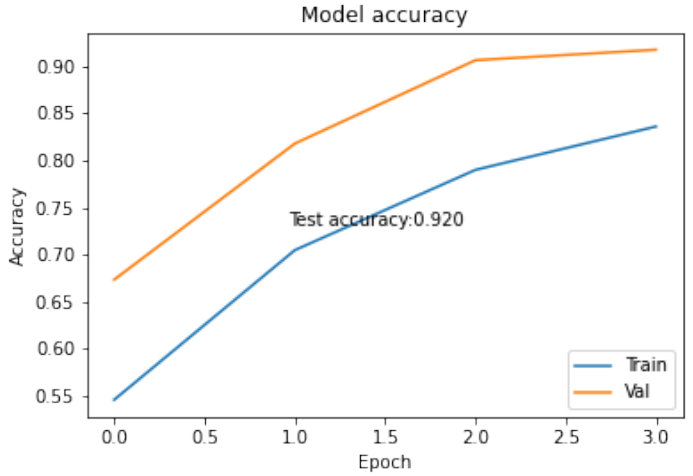
In [ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test, w_train, w_test = train_test_split(X, y, w, train_size=0.7,random_state=123)
```

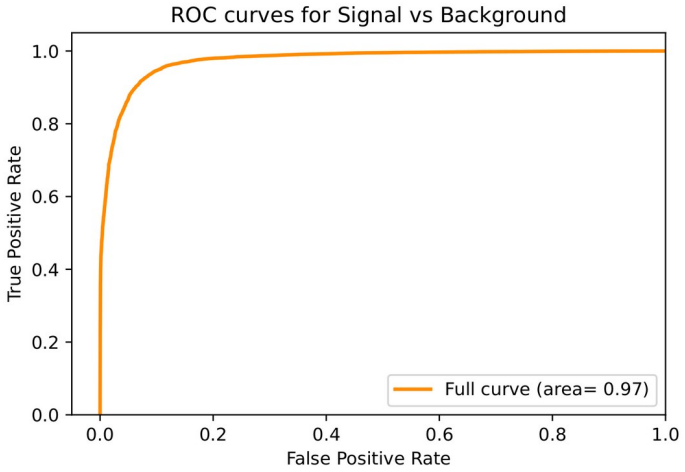
In [ ]:
```
from keras.models import Model, Sequential
from keras.layers import Dense, Dropout, Input, BatchNormalization
```

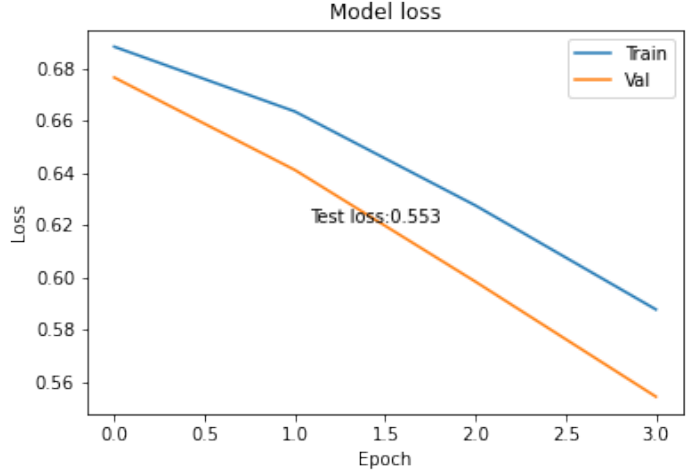
Saving the variables file

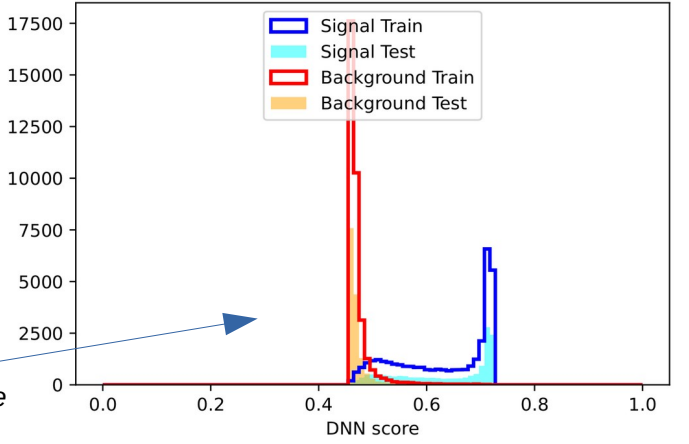
Sanity-checked that the mean and std calculated by the StandardScaler are the same that are saved in the variables file

Results



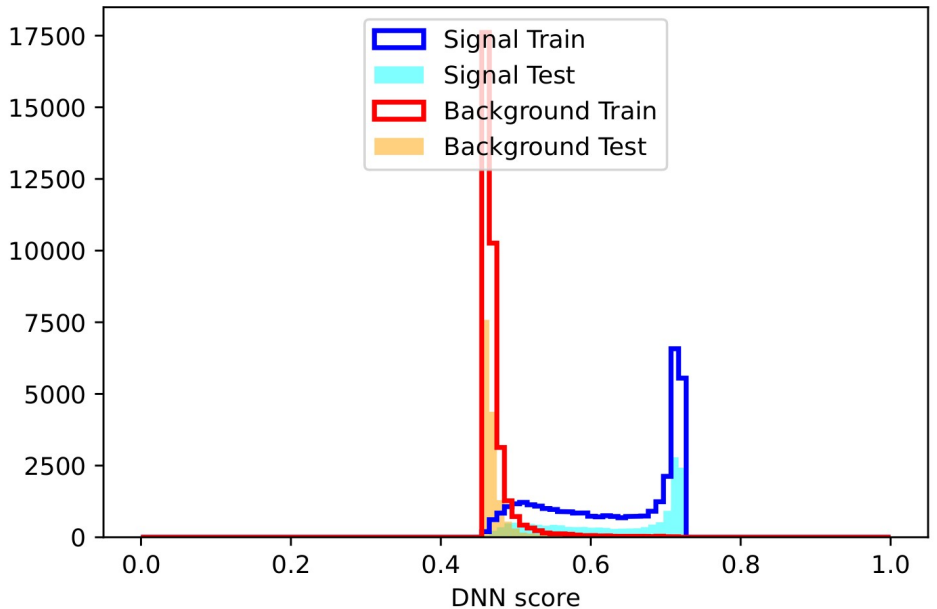




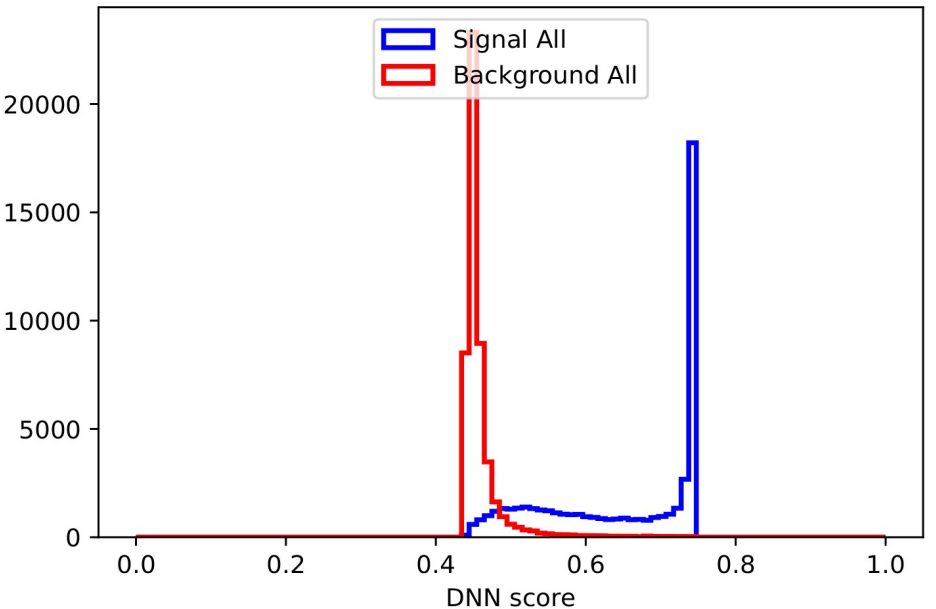


I don't want to normalize the score distributions, I want to see the absolute scales

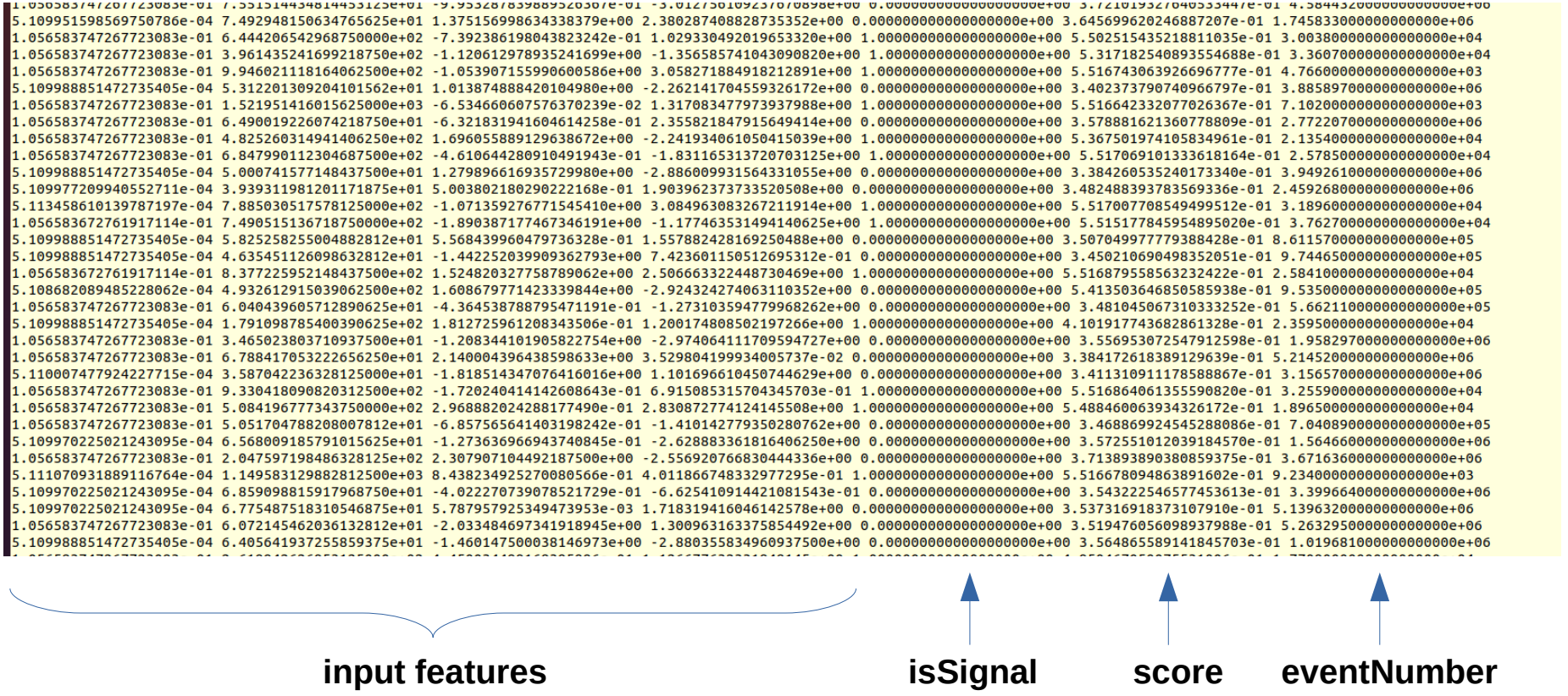
Results



Full set

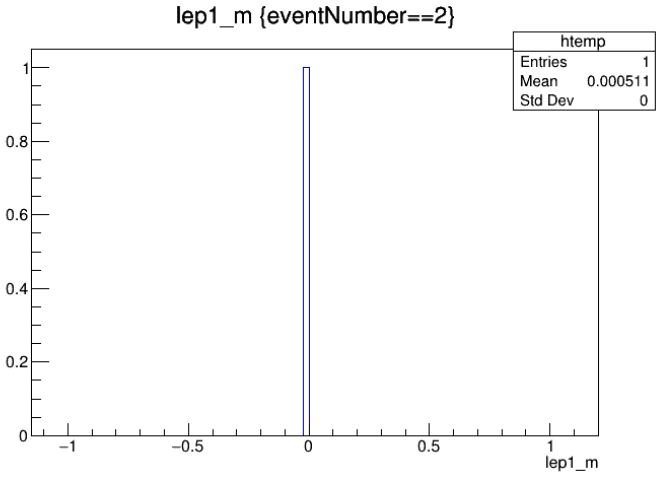


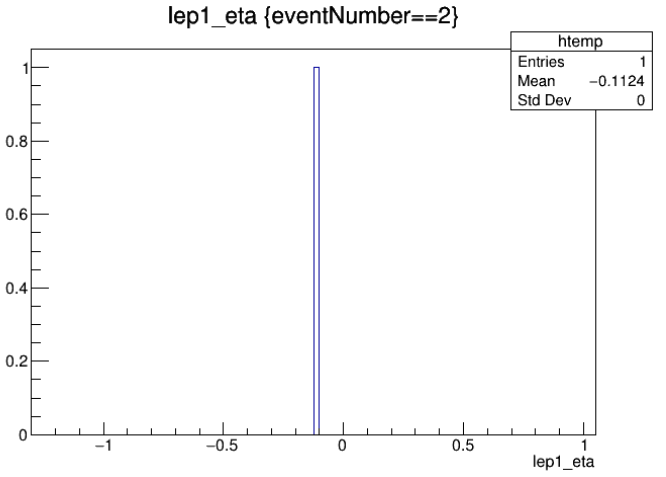
Keeping eventNumber

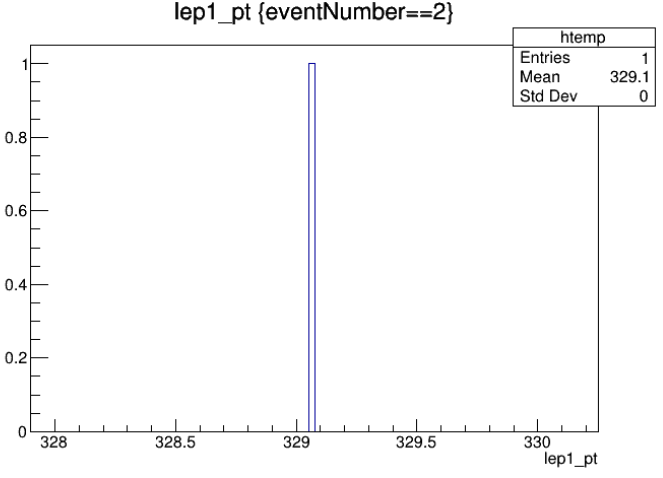


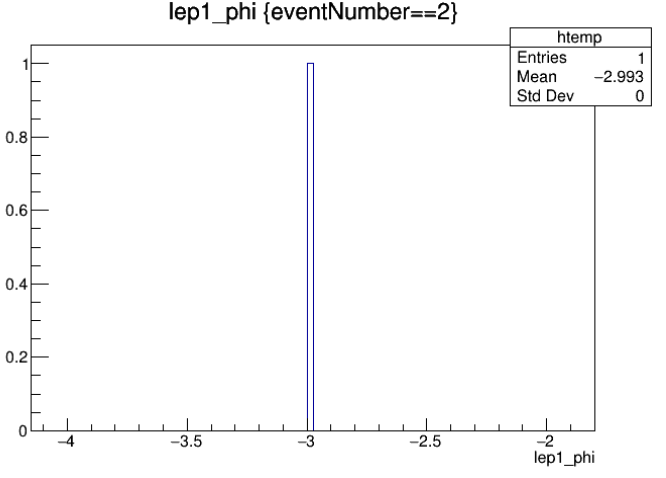
Keeping eventNumber: sanity check

```
5.10995159856975078 6e-04 1.67590866088867187 5e+02 6.76755249500274658 2e-01 8.21119323372840881 3e-02 0.00000000000000000 0e+00 4.00712907314300537 1e-01 5.09753800000000000 0e+06
5.10987651068717241 3e-04 3.29071777343750000 0e+02 -1.12388715147972106 9e-01 -2.99302196502685546 9e+00 1.00000000000000000 0e+00 5.03355145454406738 3e-01 2.00000000000000000 0e+00
5.10995159856975078 6e-04 9.72563934326171875 0e+01 1.05068588256835937 5e+00 7.68789350986480712 9e-01 0.00000000000000000 0e+00 3.58882009983062744 1e-01 3.23552200000000000 0e+06
1.05658374726772308 3e-01 2.15303405761718750 0e+02 1.81606590747833252 0e+00 -2.04437136650085449 2e+00 0.00000000000000000 0e+00 3.86630564928054809 6e-01 1.44797200000000000 0e+06
```

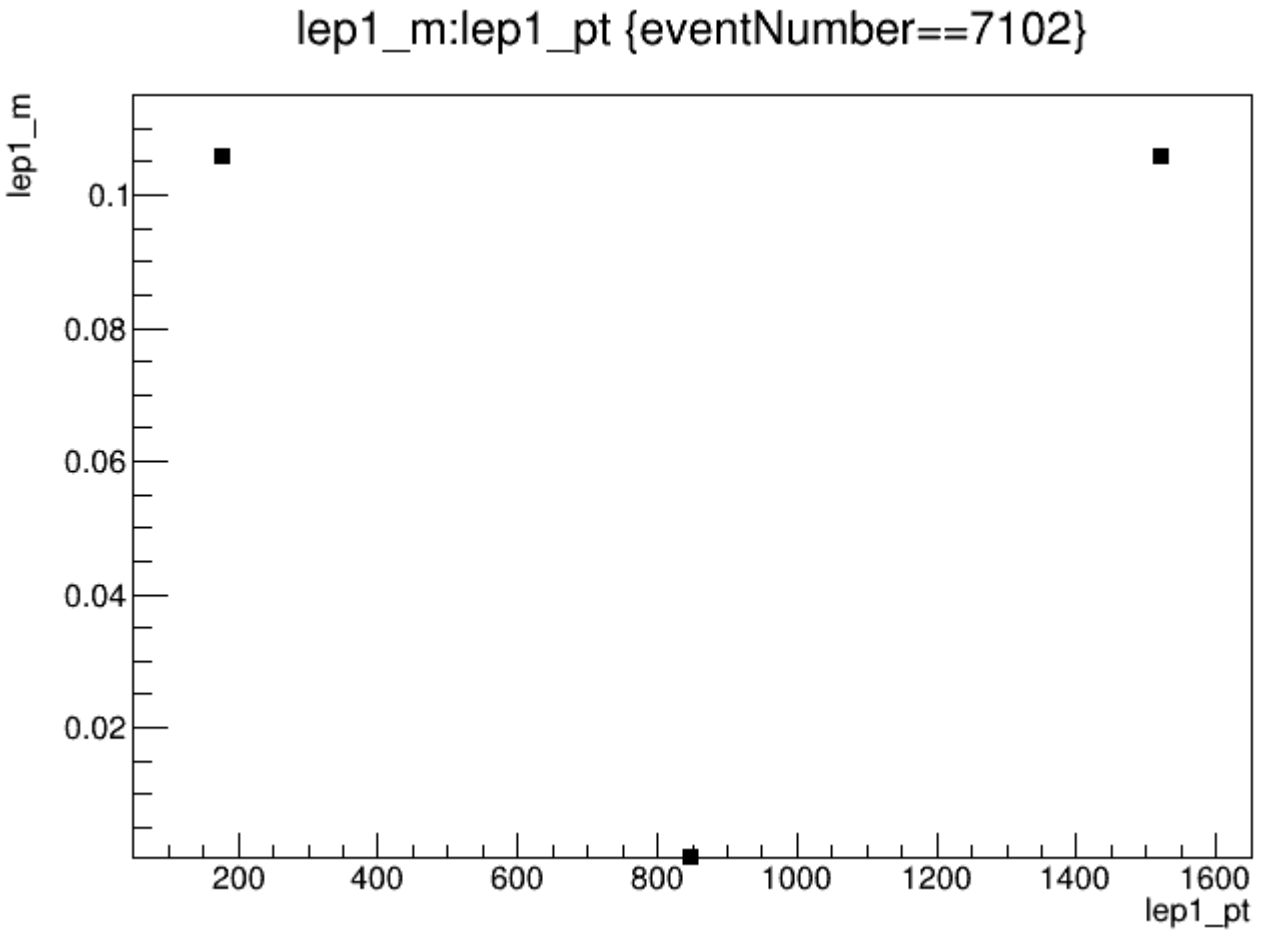








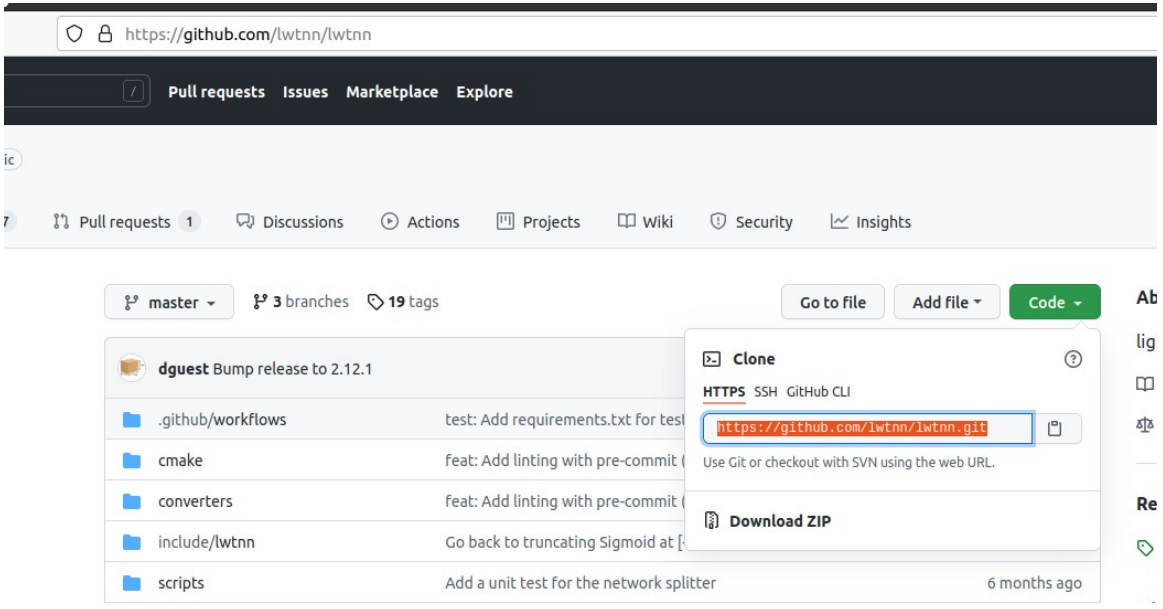
Keeping eventNumber: observation



EventNumber is not a unique identifier

From this moment on, I’m not touching python anymore (lest I re-initialize the random generators and produce slightly different results)

Running the lwtnn converters



Cloned a local copy of lwtnn

```
enricojr@enricojr-ET2321I:~/cernbox/analysis/development/2021-09-09_lwtnnTest$ ll
total 20
drwxrwxr-x  5 enricojr enricojr 4096 set 10 15:58 ./
drwxrwxr-x  9 enricojr enricojr 4096 set  9 10:20 ../
drwxrwxr-x  4 enricojr enricojr 4096 set 10 15:54 AI/
drwxrwxr-x 10 enricojr enricojr 4096 set 10 15:58 lwtnn/
drwxrwxr-x  3 enricojr enricojr 4096 set  9 10:20 ntuples/
```

Running the lwtnn converters

```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
/home/enricojr/cernbox/analysis/development/2021-09-09_lwtnnTest/lwtnn/converters/keras2json.py architecture.json variables.json weights.h5 > neural_net.json
```

```
(AI) enricojr@enricojr-ET2321I:~/cernbox/analysis/development/2021-09-09_lwtnnTest/AI/output$ ll
total 2620
drwxrwxr-x 2 enricojr enricojr   4096 set 10 16:02 ./
drwxrwxr-x 4 enricojr enricojr   4096 set 10 15:54 ../
-rw-rw-r-- 1 enricojr enricojr  16823 set 10 15:53 Accuracy.png
-rw-rw-r-- 1 enricojr enricojr   3254 set 10 15:53 architecture.json
-rwxrwxr-x 1 enricojr enricojr    170 set 10 16:02 convert*
-rw-rw-r-- 1 enricojr enricojr    142 set 10 16:01 convert~
-rw-rw-r-- 1 enricojr enricojr  17188 set 10 15:53 Loss.png
-rw-rw-r-- 1 enricojr enricojr  40573 set 10 16:02 neural_net.json
-rw-rw-r-- 1 enricojr enricojr  19353 set 10 15:53 ROC.pdf
-rw-rw-r-- 1 enricojr enricojr  13777 set 10 15:53 scores.pdf
-rw-rw-r-- 1 enricojr enricojr 374275 set 10 15:53 scores_test_background.txt
-rw-rw-r-- 1 enricojr enricojr 375725 set 10 15:53 scores_test_signal.txt
-rw-rw-r-- 1 enricojr enricojr 875725 set 10 15:53 scores_train_background.txt
-rw-rw-r-- 1 enricojr enricojr 874275 set 10 15:53 scores_train_signal.txt
-rw-rw-r-- 1 enricojr enricojr    435 set 10 15:53 variables.json
-rw-rw-r-- 1 enricojr enricojr  30960 set 10 15:53 weights.h5
```

Re-running the CxAODReader

Copied the converted file to the PDNNModel folder:

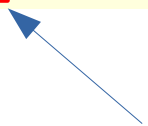
```
[schioppa@lxplus774:~/Reader4/build/x86_64-centos7-gcc8-opt/data/CxAODReader_VVSemileptonic/PDNNModel] ll
total 42
-rw-r--r--. 1 schioppa zp  1873 Jul 15 09:51 FeatureScaling.dat
-rw-r--r--. 1 schioppa zp 40573 Sep 10 16:06 neural_net.json
```

Notice that this folder already contained a FeatureScaling.dat file too
According to previous experience, this file is useless (feature scaling is applied by lwtnn internally using all the information inside the json file)

Re-running the CxAODReader

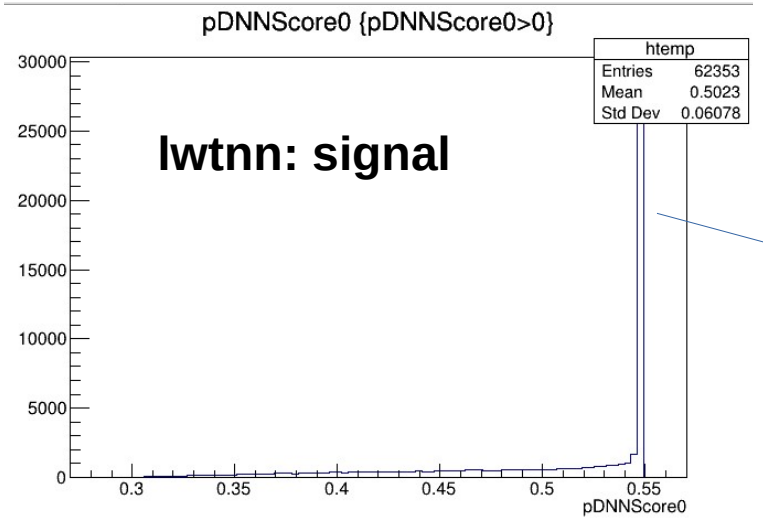
Configuration file:

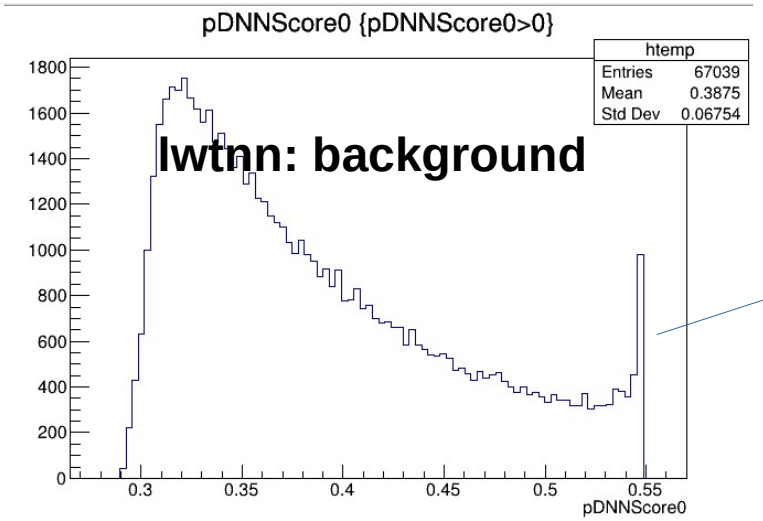
```
# PDNN configuration
bool LoadPDNN = true
string PDNNName = pDNN_ggF_Merged
string PDNNModelPath = /data/CxAODReader_VVSemileptonic/PDNNModel/neural_net.json
string PDNNScalingPath = /data/CxAODReader_VVSemileptonic/PDNNModel/FeatureScaling.dat
vector<int> PDNNMassHypothesis = 0 #700 1000 3000 5000
```

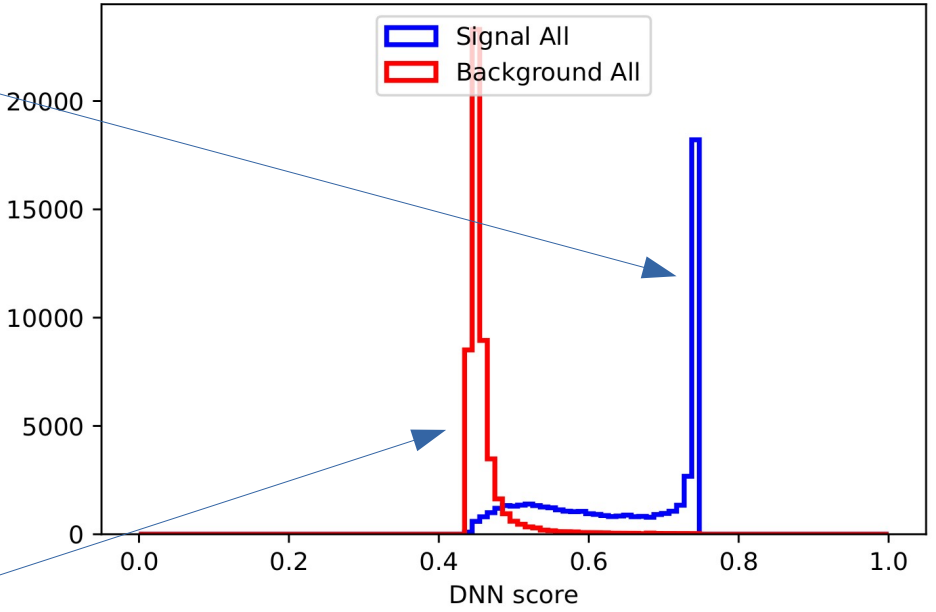


One single (dummy) mass hypothesis should collapse the pDNN into a DNN

Comparing scores

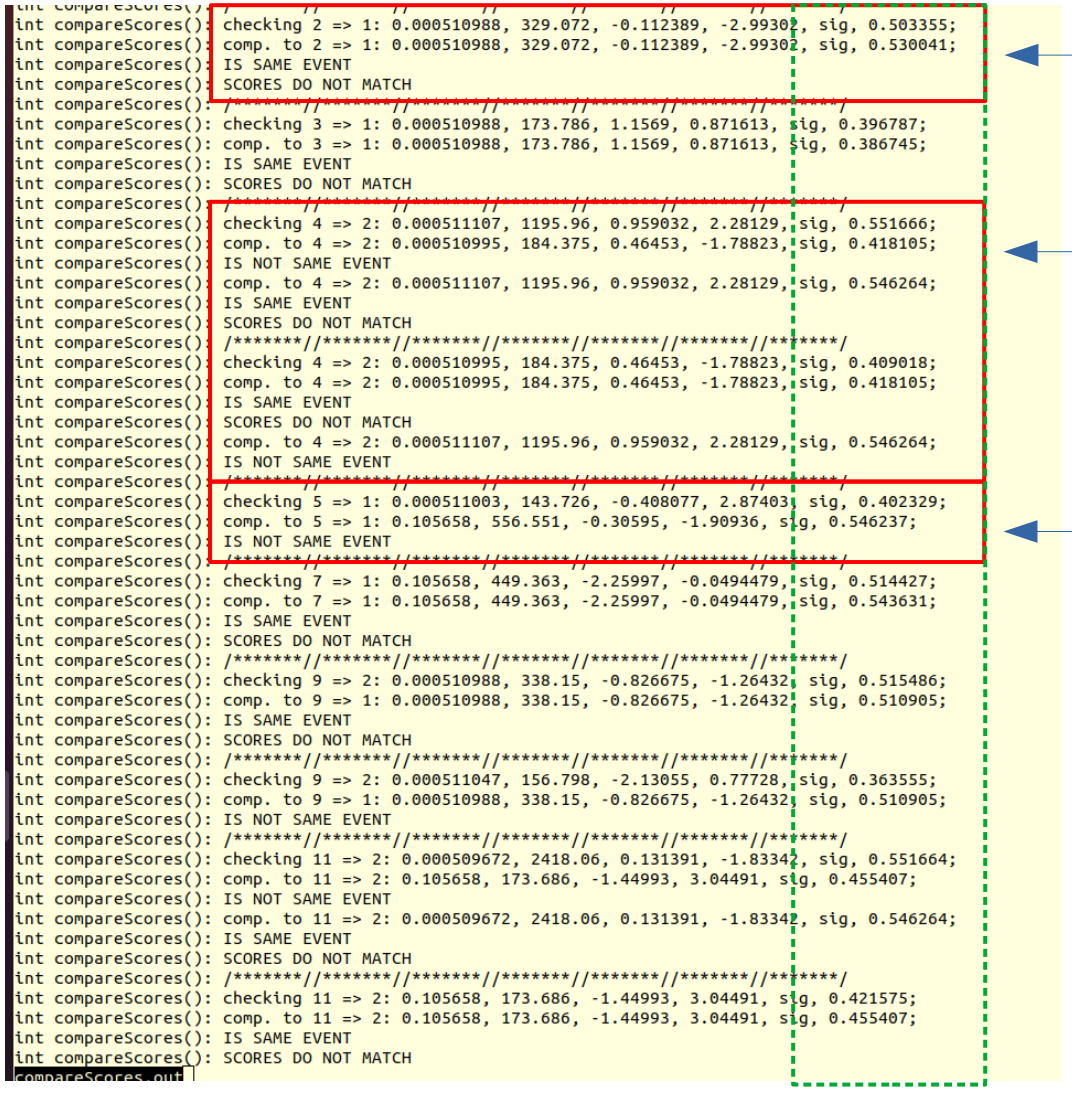






There are arguments why at this level it could be ok that they don't match.
However ...

Comparing scores

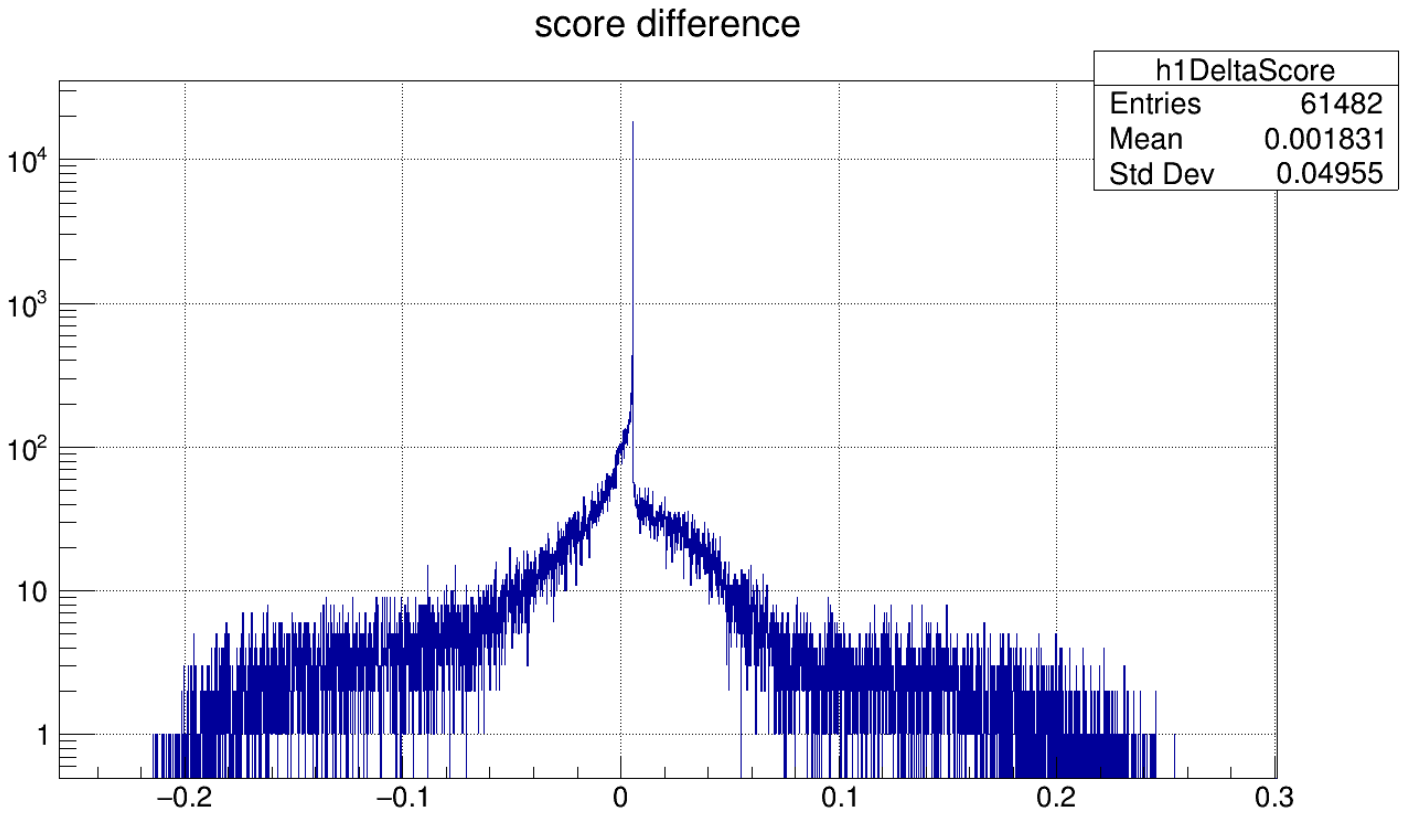


Example of single event

Example of double event

Example of non-selected event (-99)

Comparing scores



Here do the same plot for sig and bkg separately

BONUS

An interesting observation

```
In [1]: import uproot3
        import numpy as np
        import pandas as pd

In [2]: DF_Signal = uproot3.open('../ntuples/original/Signal-0.root')['Nominal'].pandas.df()

In [3]: DF_Diboson = uproot3.open('../ntuples/original/Diboson-1.root')['Nominal'].pandas.df()
        DF_Diboson.shape

Out[3]: (621719, 225)

In [4]: import sklearn.utils
        def Shuffling(df):
            df = sklearn.utils.shuffle(df,random_state=123)
            df = df.reset_index(drop=True)
            return df

        DF_Signal = Shuffling(DF_Signal)
        DF_Diboson = Shuffling(DF_Diboson)

In [5]: DF_Signal.insert(len(DF_Signal.columns), 'isSignal', np.ones(DF_Signal.shape[0]), True)
        DF_Diboson.insert(len(DF_Diboson.columns), 'isSignal', np.zeros(DF_Diboson.shape[0]), True)

        signalList = [DF_Signal[:50000] ]
        dibosonList = [DF_Diboson[:50000] ]

        totalPD_sig = pd.concat(signalList,ignore_index=True)
        totalPD_bkg = pd.concat(dibosonList,ignore_index=True)

        print ('Signal events:',totalPD_sig.shape)
        print ('Background events:',totalPD_bkg.shape)

        df_total = pd.concat([totalPD_sig,totalPD_bkg],ignore_index=True)
        df_total = Shuffling(df_total)
        print ("Total events after shuffling: ",df_total.shape)

        Signal events: (50000, 226)
        Background events: (50000, 226)
        Total events after shuffling:  (100000, 226)

In [6]: InputFeatures = ['lep1_m', 'lep1_pt', 'lep1_eta', 'lep1_phi']#, 'lep2_m', 'lep2_pt', 'lep2_eta', 'lep2_phi', 'fatjet_
        X=df_total[InputFeatures].values
        y_tmp = df_total['isSignal']
```

If I don’t do this, the DNN breaks down (!?)