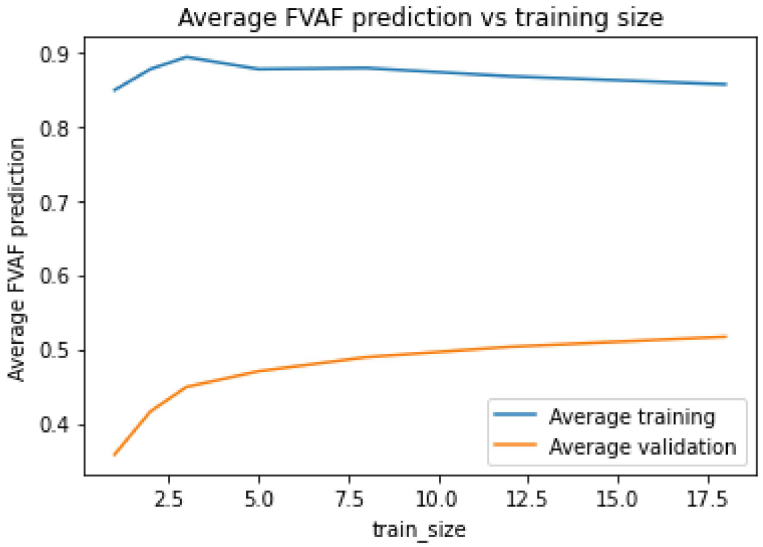
In [41]:
```
import matplotlib.pyplot as plt
import tensorflow as tf
import pandas as pd
import numpy as np
from tensorflow import keras
import os
import fnmatch
import time
import pickle
from tensorflow.keras.layers import InputLayer, Dense
from tensorflow.keras.models import Sequential

#opening the file
file_1 = open("bmi_dataset.pkl", "rb")
f_open_1 = pickle.load(file_1)
file_1.close()

#print(list(f_open_1.keys()))

# finding the actual testing labels
ins = f_open_1['MI']
Nfolds = len(ins)
folds_testing = (np.array([Nfolds-1]) + 0) % Nfolds
outs = f_open_1['torque']
outs_testing = np.concatenate(np.take(outs, folds_testing))
actual_testing = outs_testing[:,[0]]

# Getting the predicted testing labels
predict_testing = f_open_2['predict_testing']

## Getting the timestamp for predicted labels
timestamp = f_open_2['time_testing']


new_results = []

def read_all_rotations(dirname, filebase):
    '''Read results from dirname from files matching filebase'''

    # The set of files in the directory
    files = fnmatch.filter(os.listdir(dirname), filebase)
    files.sort()
    results = []


    # Loop over matching files
    for f in files:
        fp = open("%s/%s"%(dirname,f), "rb")
        r = pickle.load(fp)
        fp.close()
        results.append(r)
    return results

# matching the files
train = [1, 2, 3, 5, 8, 12, 18]
for t in train:
    filebase = "bmi_torque_1_hidden_200_100_50_25_10_5_JI_Ntraining_"+str(t)+"_rotatio
    new_results.append( read_all_rotations("results", filebase))
```

```
avg_train = []
avg_validate = []


temp_1 = []
temp_2 = []


# calculating the average
for i in range(len(train)):
    for j in range(len(new_results[0][0])):
        temp_1.append(np.mean(new_results[i][j]['predict_training_eval'][1]))
        temp_2.append(np.mean(new_results[i][j]['predict_validation_eval'][1]))

    avg_train.append(sum(temp_1)/len(temp_1))
    avg_validate.append(sum(temp_2)/len(temp_2))

print(len(avg_train))
#plotting figure 1

plt.plot(train,avg_train)
plt.plot(train,avg_validate)


plt.ylabel('Average FVAF prediction')
plt.xlabel('train_size')
plt.title('Average FVAF prediction vs training size')
plt.legend(['Average training','Average validation'])
#saving the figure2
plt.savefig("figure1.png")
plt.show()
plt.close()
```

```
7
```



In [ ]:

In [ ]: