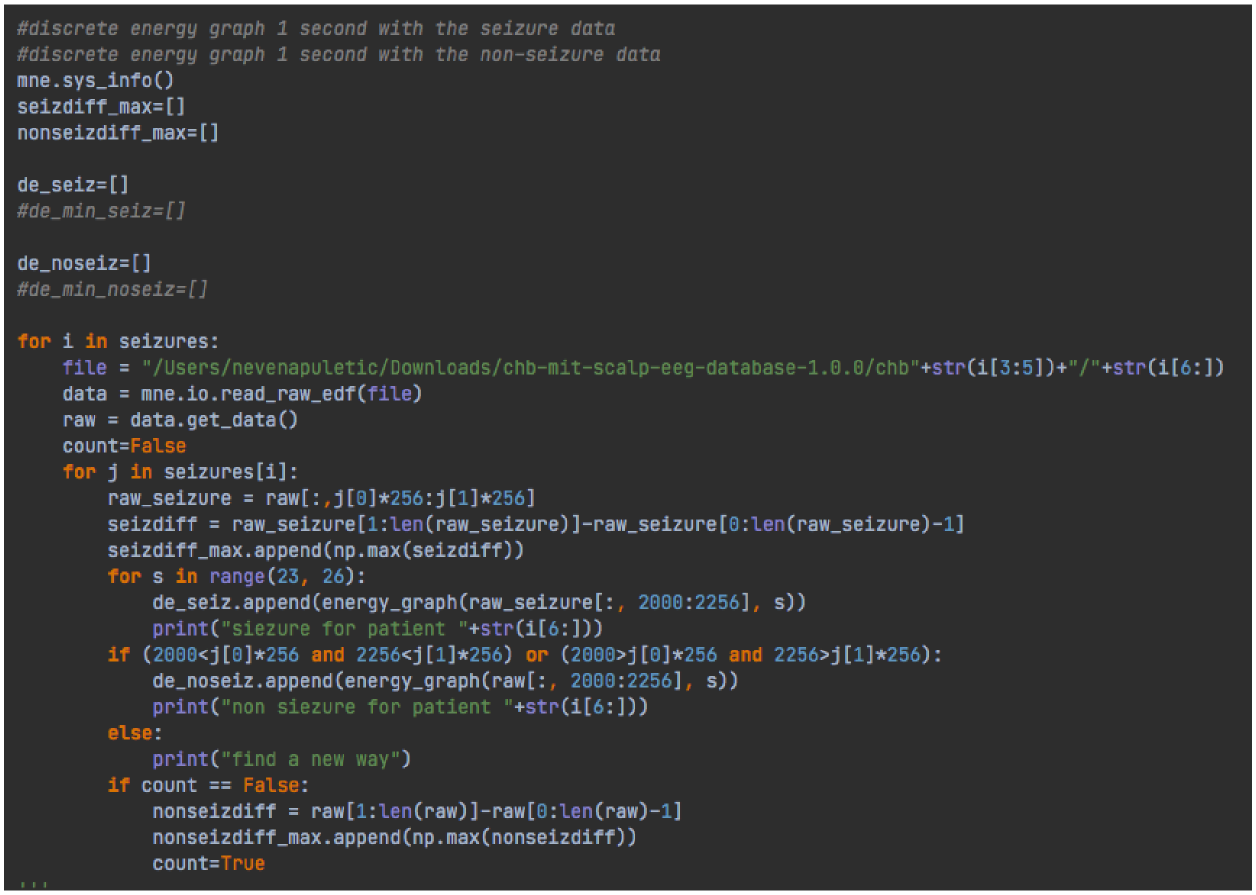
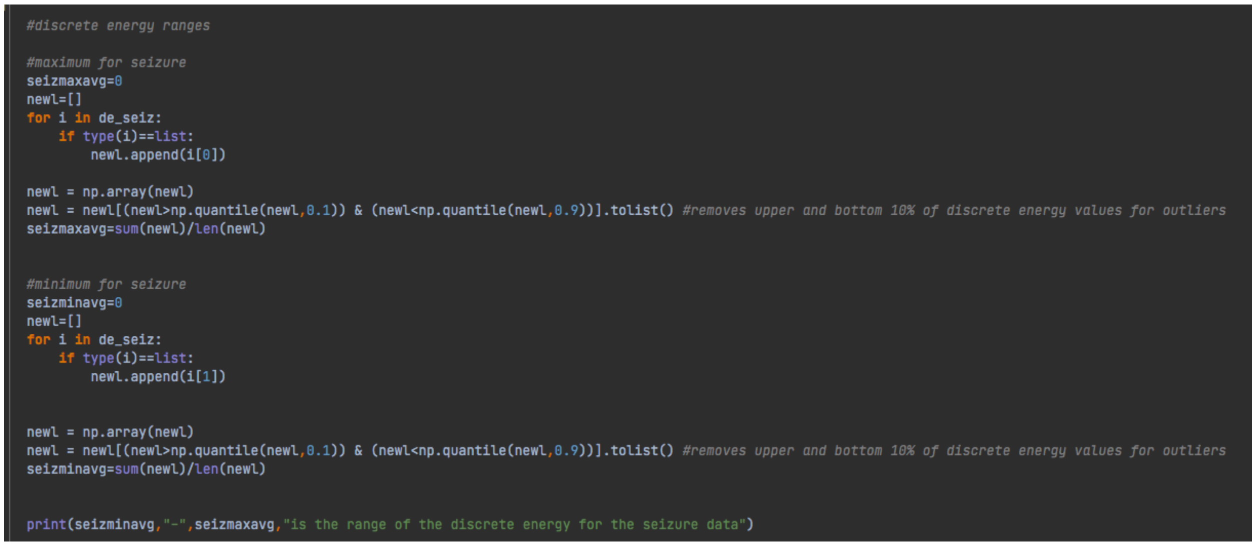
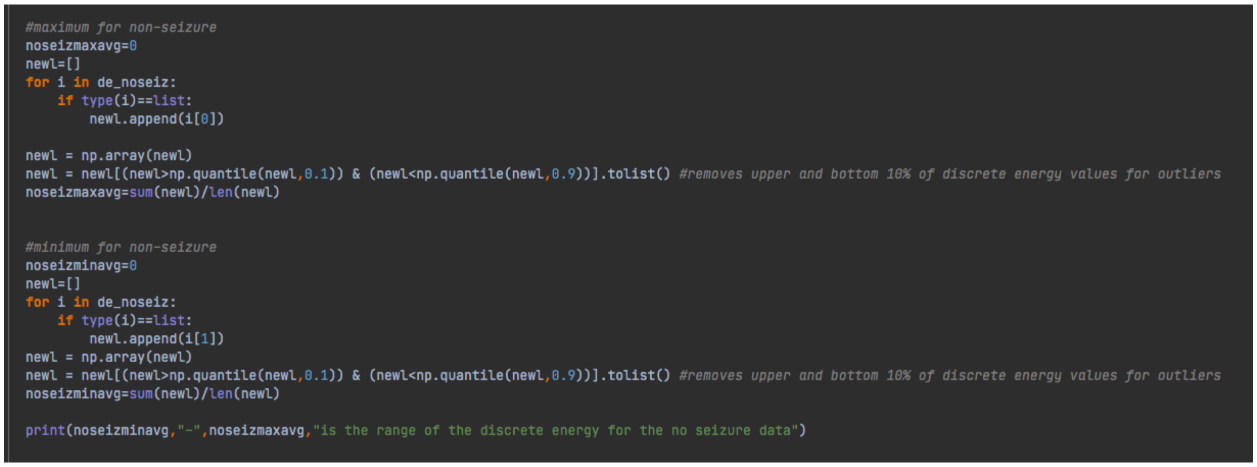


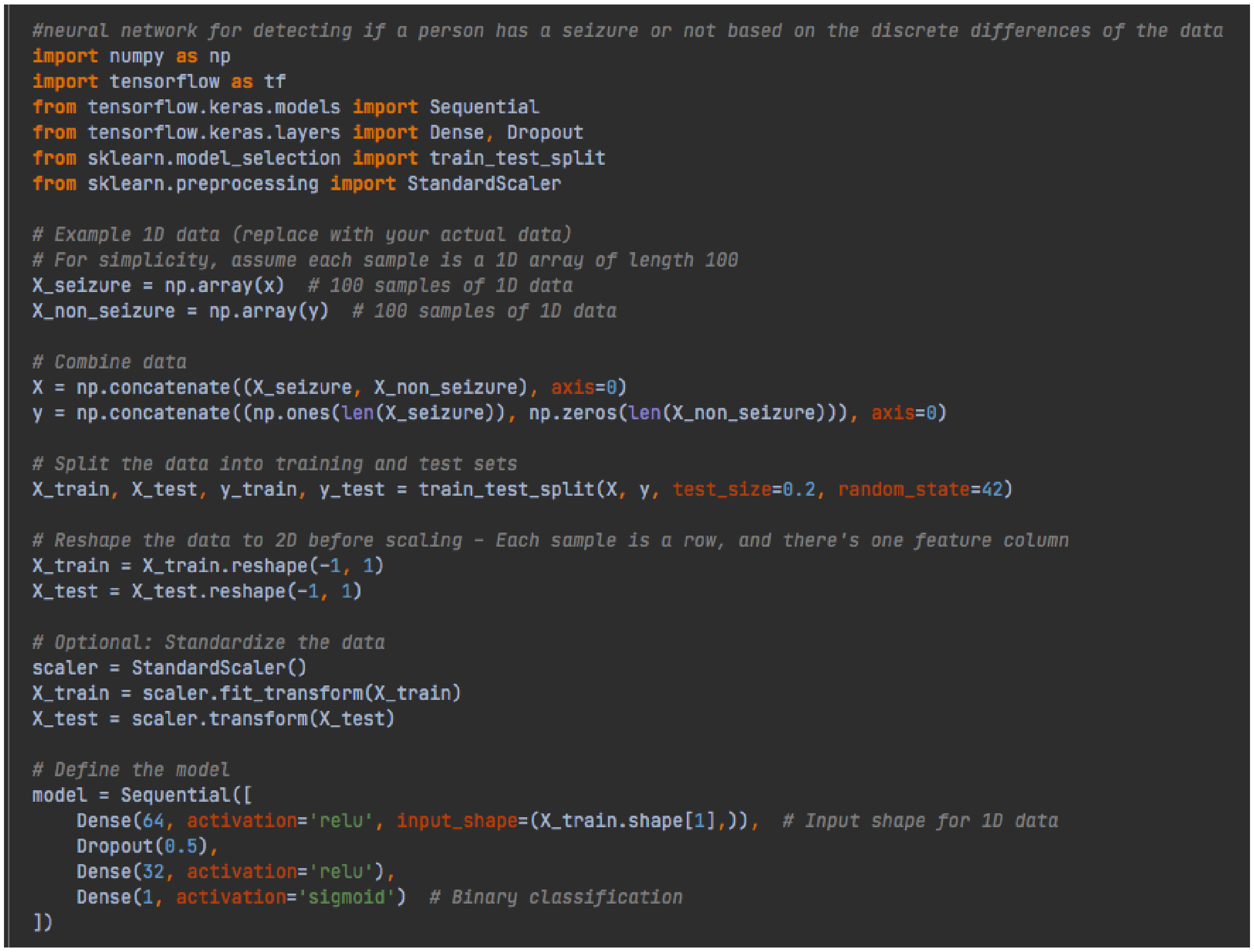
```python
#discrete energy graph 1 second with the seizure data
#discrete energy graph 1 second with the non-seizure data
mne.sys_info()
seizdiff_max=[]
nonseizdiff_max=[]

de_seiz=[]
#de_min_seiz=[]

de_noseiz=[]
#de_min_noseiz=[]

for i in seizures:
    file = "/Users/nevenapuletic/Downloads/chb-mit-scalp-eeg-database-1.0.0/chb"+str(i[3:5])+"/"+str(i[6:])
    data = mne.io.read_raw_edf(file)
    raw = data.get_data()
    count=False
    for j in seizures[i]:
        raw_seizure = raw[:,j[0]*256:j[1]*256]
        seizdiff = raw_seizure[1:len(raw_seizure)]-raw_seizure[0:len(raw_seizure)-1]
        seizdiff_max.append(np.max(seizdiff))
        for s in range(23, 26):
            de_seiz.append(energy_graph(raw_seizure[:, 2000:2256], s))
            print("siezure for patient "+str(i[6:]))
        if (2000<j[0]*256 and 2256<j[1]*256) or (2000>j[0]*256 and 2256>j[1]*256):
            de_noseiz.append(energy_graph(raw[:, 2000:2256], s))
            print("non siezure for patient "+str(i[6:]))
        else:
            print("find a new way")
        if count == False:
            nonseizdiff = raw[1:len(raw)]-raw[0:len(raw)-1]
            nonseizdiff_max.append(np.max(nonseizdiff))
            count=True
'''
```



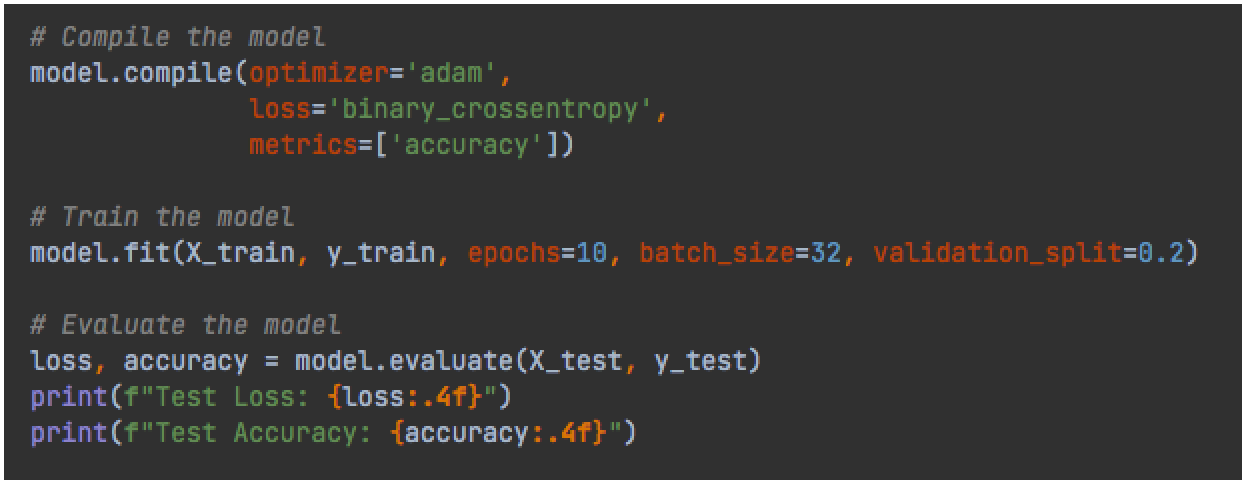
```python
#discrete energy ranges

#maximum for seizure
seizmaxavg=0
newl=[]
for i in de_seiz:
    if type(i)==list:
        newl.append(i[0])

newl = np.array(newl)
newl = newl[(newl>np.quantile(newl,0.1)) & (newl<np.quantile(newl,0.9))].tolist() #removes upper and bottom 10% of discrete energy values for outliers
seizmaxavg=sum(newl)/len(newl)


#minimum for seizure
seizminavg=0
newl=[]
for i in de_seiz:
    if type(i)==list:
        newl.append(i[1])


newl = np.array(newl)
newl = newl[(newl>np.quantile(newl,0.1)) & (newl<np.quantile(newl,0.9))].tolist() #removes upper and bottom 10% of discrete energy values for outliers
seizminavg=sum(newl)/len(newl)


print(seizminavg,"-",seizmaxavg,"is the range of the discrete energy for the seizure data")
```



```python
#maximum for non-seizure
noseizmaxavg=0
newl=[]
for i in de_noseiz:
    if type(i)==list:
        newl.append(i[0])

newl = np.array(newl)
newl = newl[(newl>np.quantile(newl,0.1)) & (newl<np.quantile(newl,0.9))].tolist() #removes upper and bottom 10% of discrete energy values for outliers
noseizmaxavg=sum(newl)/len(newl)


#minimum for non-seizure
noseizminavg=0
newl=[]
for i in de_noseiz:
    if type(i)==list:
        newl.append(i[1])
newl = np.array(newl)
newl = newl[(newl>np.quantile(newl,0.1)) & (newl<np.quantile(newl,0.9))].tolist() #removes upper and bottom 10% of discrete energy values for outliers
noseizminavg=sum(newl)/len(newl)

print(noseizminavg,"-",noseizmaxavg,"is the range of the discrete energy for the no seizure data")
```



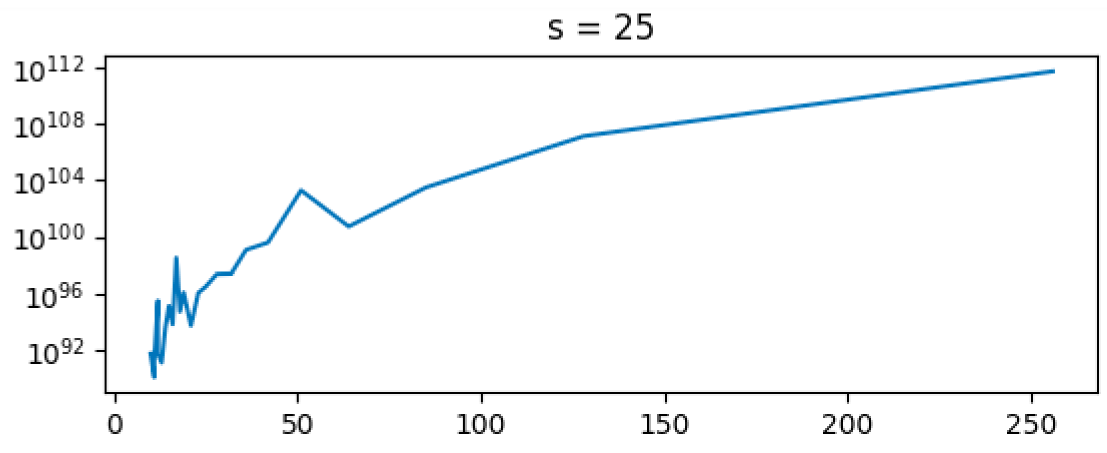
```python
#neural network for detecting if a person has a seizure or not based on the discrete differences of the data
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Example 1D data (replace with your actual data)
# For simplicity, assume each sample is a 1D array of length 100
X_seizure = np.array(x)  # 100 samples of 1D data
X_non_seizure = np.array(y)  # 100 samples of 1D data

# Combine data
X = np.concatenate((X_seizure, X_non_seizure), axis=0)
y = np.concatenate((np.ones(len(X_seizure)), np.zeros(len(X_non_seizure))), axis=0)

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Reshape the data to 2D before scaling - Each sample is a row, and there's one feature column
X_train = X_train.reshape(-1, 1)
X_test = X_test.reshape(-1, 1)

# Optional: Standardize the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Define the model
model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train.shape[1],)),  # Input shape for 1D data
    Dropout(0.5),
    Dense(32, activation='relu'),
    Dense(1, activation='sigmoid')  # Binary classification
])
```



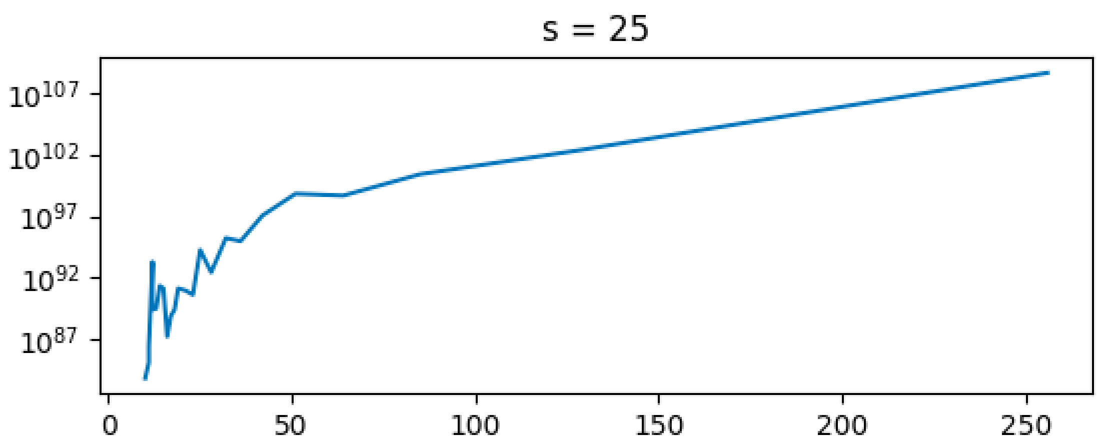
```python
# Compile the model
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2)

# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Loss: {loss:.4f}")
print(f"Test Accuracy: {accuracy:.4f}")
```

```
Test Loss: 0.4335                Test Loss: 0.3648
Test Accuracy: 0.8178            Test Accuracy: 0.8679
```

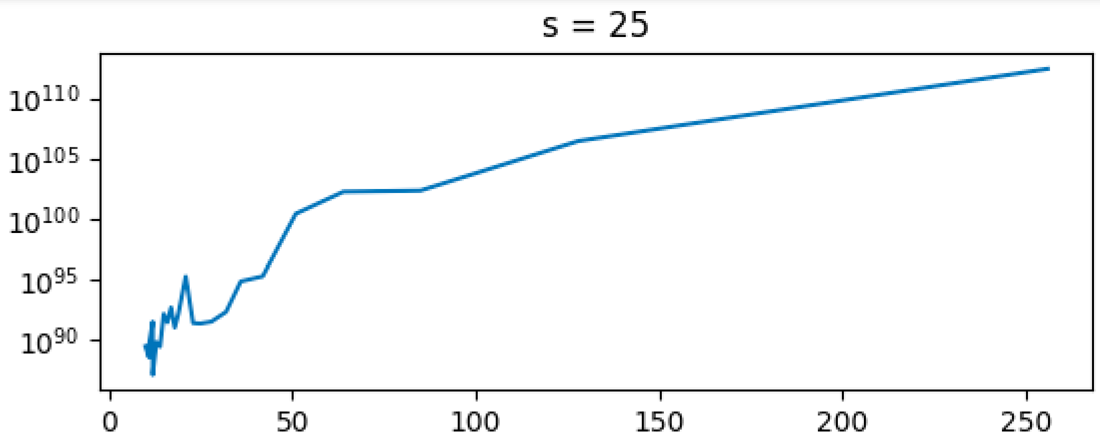
```
9.001246757345378e+96 - 8.777881417578864e+116 is the range of the discrete energy for the seizure data
2.4790236109105207e+104 - 1.7909608544528335e+124 is the range of the discrete energy for the no seizure data
2.3289886727245104e-05 seizure vs non seizure minimum comparisons
2.8235444161872268e-08 seizure vs non seizure maximum comparisons
```



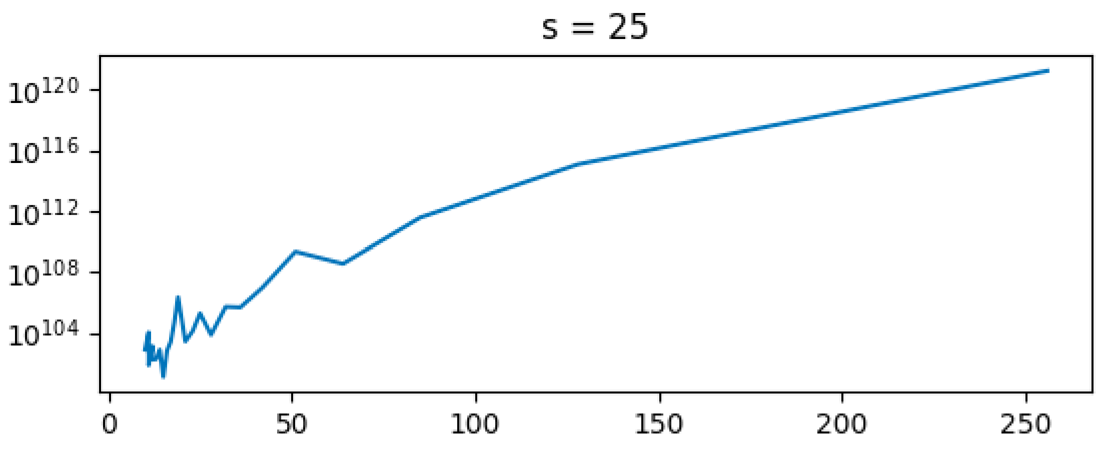
```
siezure for patient chb02_19.edf
```



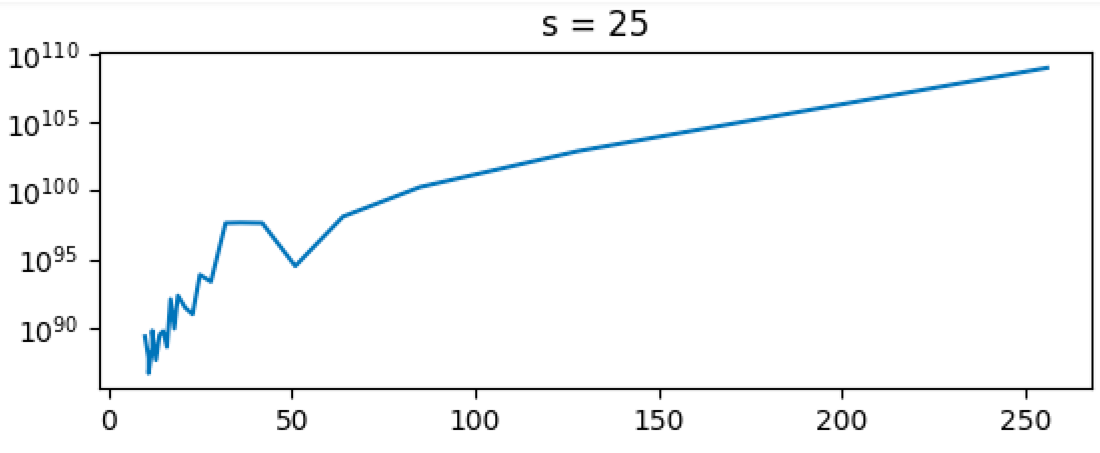
non siezure for patient chb02_19.edf



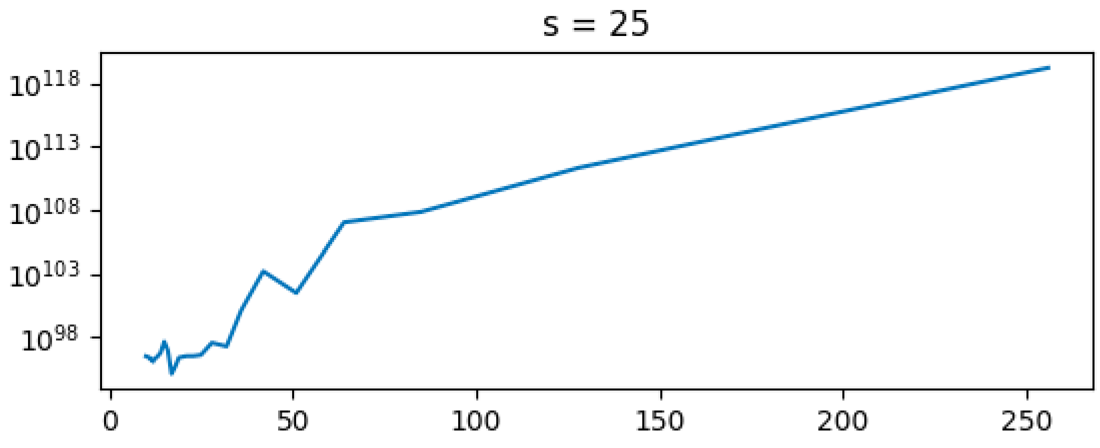
siezure for patient chb02_16.edf



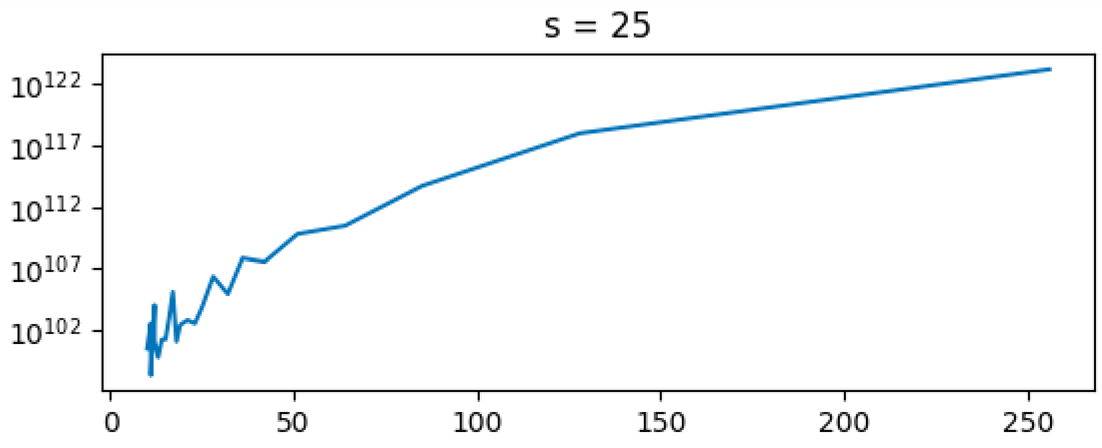
non siezure for patient chb02_16.edf



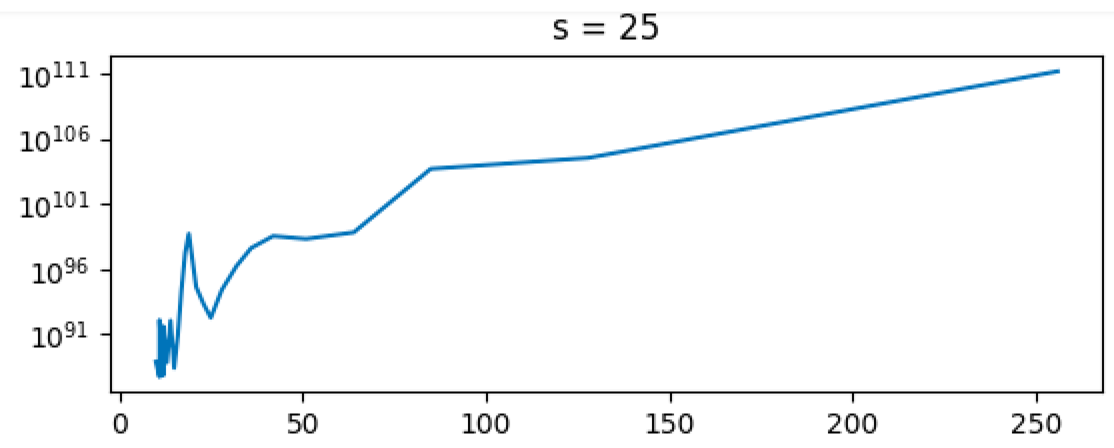
```
siezure for patient chb01_21.edf
```



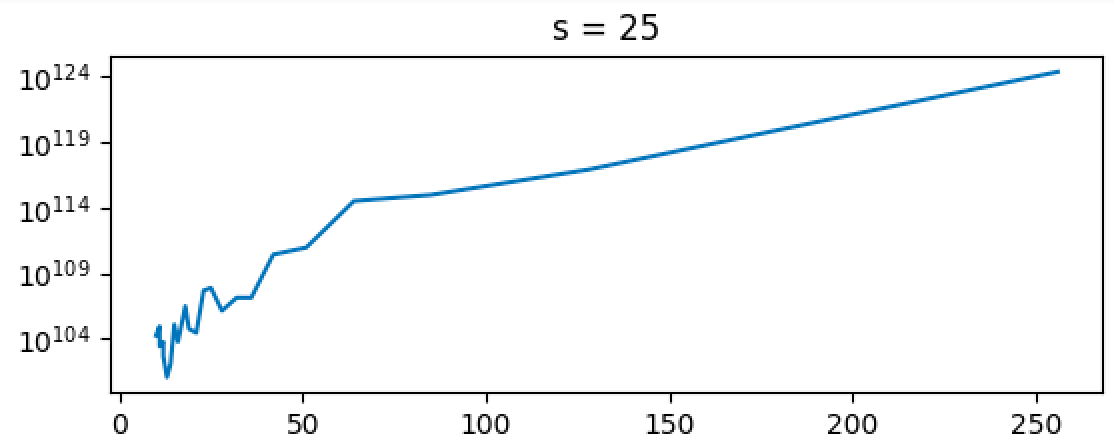
```
non siezure for patient chb01_21.edf
```



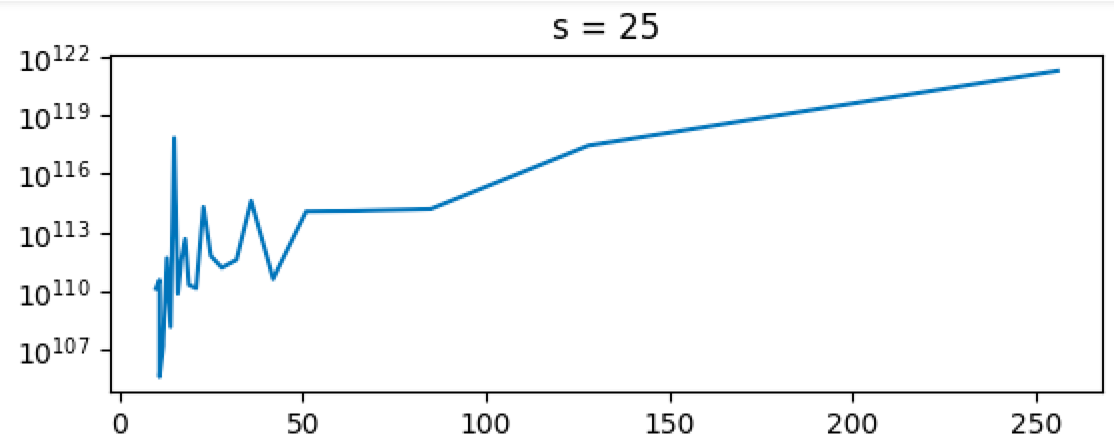
```
siezure for patient chb04_28.edf
```



```
siezure for patient chb01_04.edf
```



```
non siezure for patient chb01_04.edf
```



```
non siezure for patient chb04_28.edf
```