Installations/Imports

```
pip install pyedflib
```

```
Requirement already satisfied: pyedflib in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.7/dist-pack
```

```
pip install mne
```

```
Requirement already satisfied: mne in /usr/local/lib/python3.7/dist-packages (1.
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: pooch>=1.5 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: numpy>=1.15.4 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: decorator in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.7/dist
Requirement already satisfied: appdirs>=1.3.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dis
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/di
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/l
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dis
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dis
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages
```

```
#IMPORTS
import numpy as np
import sklearn
import pyedflib
from pyedflib import highlevel
import pandas as pd
import matplotlib.pyplot as plt
import mne
import random
from sklearn.metrics import confusion_matrix
import seaborn as sn
from sklearn import svm
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn import svm
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call dr
```

Data Preprocessing

```
filenames_03 = []
y_03 = []
# file_group = 'chb03-summary.txt'
with open('/content/drive/MyDrive/ML Final Project/EEG_DATA/' + 'chb03-summary.txt') a
    lines = f.readlines()

for line in lines:
    # get each filename and store in a filename array
    if (line[0:10] == 'File Name:'):
        # add the filename to the array
        filenames_03.append(line[11:-1])
    elif (line[0:27] == 'Number of Seizures in File:'):
        y_03.append(int(line[28]))


# formatting data for input into svm


X_03 = np.empty((0,8280000), float)
for file in filenames_03:
    edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' + fi
    header = ','.join(edf.ch_names)

    # downsample
    raw_downsampled = edf.copy().resample(sfreq=100)

    # add to np array
    print(file)
    print(raw_downsampled._data.flatten().shape)
    X_03 = np.append(X_03, raw_downsampled._data.flatten().reshape(1,8280000), axis =
```

```
chb03_31.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique,
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/'
chb03_32.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
```

```
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_33.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_34.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_35.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_36.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_37.edf
(8280000,)
Extracting EDF parameters from /content/drive/MyDrive/ML Final Project/EEG_DATA/
EDF file detected
Setting channel info structure...
Creating raw.info structure...
<ipython-input-6-b5868d555081>:6: RuntimeWarning: Channel names are not unique, 
  edf = mne.io.read_raw_edf('/content/drive/MyDrive/ML Final Project/EEG_DATA/' 
chb03_38.edf
(8280000,)
```

Train test split

```
X_train, X_test, y_train, y_test = train_test_split(X_03, y_03, random_state=1, test_s
```

SVM

```
svm1 = svm.SVC()
svm1.fit(X_train, y_train)
```

```
SVC()
```

```
print(svm1.score(X_test, y_test))
```

```
0.75
```

```
lin_ker = svm.SVC(kernel='linear')
lin_ker.fit(X_train, y_train)
print(lin_ker.score(X_test, y_test))
```

```
0.75
```

```
poly_ker = svm.SVC(kernel='poly')
poly_ker.fit(X_train, y_train)
print(poly_ker.score(X_test, y_test))
```

```
0.75
```

```
sig_ker = svm.SVC(kernel='sigmoid')
sig_ker.fit(X_train, y_train)
print(sig_ker.score(X_test, y_test))
```

```
0.75
```

```
# make a confusion matrix
pred_vals = []
for val in X_test:
    pred_vals.append(svm1.predict(val.reshape(1,-1)))

cm = confusion_matrix(y_test, pred_vals)

class_names = ["no seizure", "seizure"]

fig = plt.figure(figsize=(16, 14))
ax= plt.subplot()
sn.heatmap(cm, annot=True, ax = ax, fmt = 'g');

ax.set_xlabel('Predicted', fontsize=20)
ax.xaxis.set_label_position('bottom')
plt.xticks(rotation=90)
ax.xaxis.set_ticklabels(class_names, fontsize = 10)
```

```
ax.xaxis.set_ticklabels(class_names, fontsize = 10)
ax.xaxis.tick_bottom()

ax.set_ylabel('True', fontsize=20)
ax.yaxis.set_ticklabels(class_names, fontsize = 10)
plt.yticks(rotation=0)

plt.title('SVM Confusion Matrix', fontsize=20)
plt.show()
```

Logistic Regression

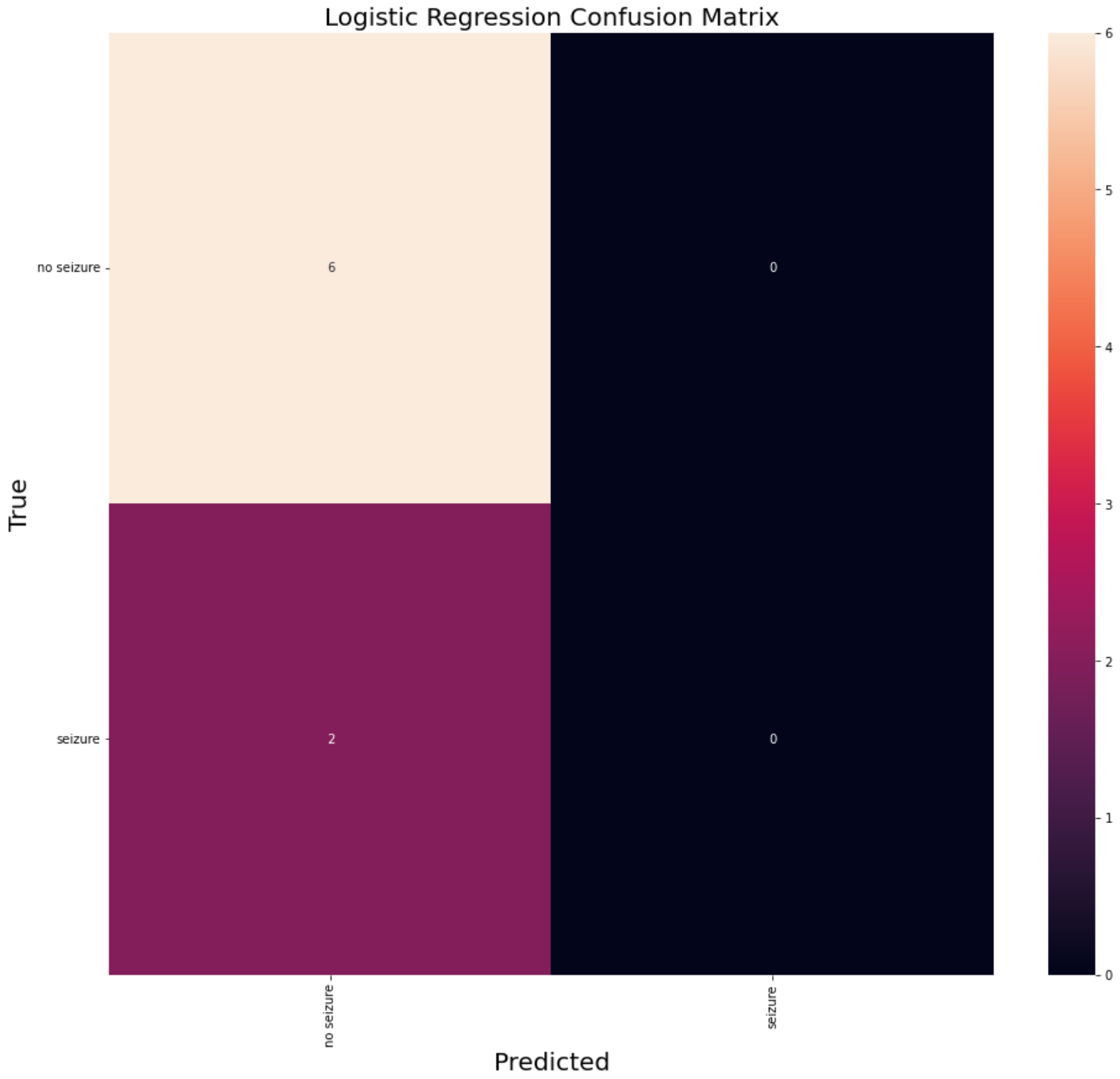

```
logisticRegr = LogisticRegression()
logisticRegr.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
score = logisticRegr.score(X_test, y_test)
print(score)
```

```
0.75
```

```
# make a confusion matrix
pred_vals = []
for val in X_test:
    pred_vals.append(logisticRegr.predict(val.reshape(1,-1)))

cm = confusion_matrix(y_test, pred_vals)

class_names = ["no seizure", "seizure"]

fig = plt.figure(figsize=(16, 14))
ax= plt.subplot()
sn.heatmap(cm, annot=True, ax = ax, fmt = 'g');

ax.set_xlabel('Predicted', fontsize=20)
ax.xaxis.set_label_position('bottom')
plt.xticks(rotation=90)
ax.xaxis.set_ticklabels(class_names, fontsize = 10)
ax.xaxis.tick_bottom()

ax.set_ylabel('True', fontsize=20)
ax.yaxis.set_ticklabels(class_names, fontsize = 10)
plt.yticks(rotation=0)

plt.title('Logistic Regression Confusion Matrix', fontsize=20)
plt.show()
```


Logistic Regression Confusion Matrix
no seizure
seizure
6
0
2
0
True
no seizure
seizure
Predicted
6
5
4
3
2
1
0

✓ 0s completed at 11:13 PM