Primeira Questão

Destrinchada:

```
import numpy as np

def calculate_eqm(y_prediction, y_i):
    quadrado_das_subtracoes = np.square(y_prediction - y_i)
    soma_dos_quadrados = np.sum(quadrado_das_subtracoes)
    tamanho_da_array = np.size(y_prediction)
    media = soma_dos_quadrados / tamanho_da_array
    return media

y_prediction = np.array([1,2,3])
y_i = np.array([0,0,3])

print(calculate_eqm(y_prediction, y_i))
```

Otimizada:

```
import numpy as np

def calculate_eqm(y_prediction, y_i):
    return np.sum(np.square(y_prediction - y_i)) / np.size(y_prediction)

y_prediction = np.array([1,2,3])
y_i = np.array([0,0,3])

print(calculate_eqm(y_prediction, y_i))
```

Segunda Questão

Destrinchada:

```
import numpy as np

x = np.random.randn(64, 512)

def process_EEG_signal(matriz):
    media = np.mean(matriz)
    matriz_processada = matriz - media
    return np.array(matriz_processada)

print(process_EEG_signal(x))
```

Otimizada:

```
import numpy as np

x = np.random.randn(64, 512)

def process_EEG_signal(matriz):
    return np.array(matriz - np.mean(matriz))

print(process_EEG_signal(x))
```

Opcional para verificação (adicionar ao final do código substituindo “print(process_EEG_signal(x))”:

```
print(x)
print(x.shape)
processed = process_EEG_signal(x)
print(processed)
print(processed.shape)
```

Terceira Questão

Destrinchada:

```
import numpy as np

def outliers(matriz):
    Q1 = np.quantile(matriz, .25)
    Q3 = np.quantile(matriz, .75)
    IQR = Q3 - Q1
    limite_acima = 1.5 * IQR + Q3
    limite_abaixo = 1.5 * IQR - Q1
    outlier_bool = (matriz > limite_acima) | (matriz < limite_abaixo)
    outlier_count = np.sum(outlier_bool)
    outlier_numbers = matriz[(matriz > limite_acima) | (matriz < limite_abaixo)]
    return f"Limit Above: {limite_acima}\n\nLimit Below: {limite_abaixo}\n\nIs
Outlier Bool:\n\n{outlier_bool}\n\nOutliers Count:
{outlier_count}\n\nOutliers:\n\n{outlier_numbers}"

x = np.random.randn(300,15)
print(outliers(x))
```