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

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import StratifiedKFold

import torch
import torch.nn as nn
import torch.optim as Adam
import matplotlib.pyplot as plt
from torch import nn
```

1. Baye's Theorem

M is a marker that determines genetic disposition to kidney disease. A chemical test can show if you are positive or negative for M. However, test is not 100% right. y = +ve or -ve x = marker (M) or no marker (no M)

P[+|M] = 0.95

P[-|no M] = 0.95

P[M] = 0.01

1a

P[-|M] = 0.05

P[+|not M] = 0.05

P[not M] = 0.99

1b

In [2]:
```
have_M = 0.01
have_M_test_pos = 0.95

no_M = 0.95
no_M_test_pos = 0.05

actually_have_M = have_M * have_M_test_pos
false_positive = no_M * no_M_test_pos

all_positive_tests = actually_have_M + false_positive
odds_that_Korede_has_M = actually_have_M / all_positive_tests

print(f'The odds that Korede actually has M are {round(odds_that_Korede_has_
```

```
The odds that Korede actually has M are 0.1667.
```

Thanks to Baye's Theorem, I would not be too worried about actually having M. The feature of the data that accounts for this result is the occurence of M in the population (0.01).

1c

New Scenario:

P[M] = 0.10

P[not M] = 0.90

In [3]:
```
have_M = 0.1
have_M_test_pos = 0.95

no_M = 0.90
no_M_test_pos = 0.05

actually_have_M = have_M * have_M_test_pos
false_positive = no_M * no_M_test_pos

all_positive_tests = actually_have_M + false_positive
odds_that_Korede_has_M = actually_have_M / all_positive_tests

print(f'The odds that Korede actually has M are {round(odds_that_Korede_has_
```

```
The odds that Korede actually has M are 0.6786!
```

Omg my odds are so much higher now. I would be worried.

2. Gaussian Naive Bayes

P(cultivar | X) - probability of cultivar given an attribute

P(wine attribute x | cultivar) - probability an attribute given a cultivar

In [22]:
```
wines = pd.read_csv('../wines.csv')
# we don't need start assignment, so drop it
wines = wines.drop('Start assignment', axis = 1)
# remname ranking as Cultivar for clarity
wines = wines.rename(columns = {'ranking': 'Cultivar'})
# shuffle values in Cultivar
shuffled_cultivar = wines['Cultivar'].sample(frac = 1)
wines['Cultivar'] = shuffled_cultivar.values

#wines.head()
```

2a

In [5]:
```
class NaiveBayesClassifier():
    def __init__(self):
        self.type_indices = {}    # store the indices of wines that belong t
        self.type_stats = {}      # store the mean and std of each cultivar
        self.ndata = 0
        self.trained = False

    @staticmethod #static methods are bound to a class and not to instances
    def gaussian(x,mean,std):
        # Gaussian probability density formula
        exponent = -((x - mean)/ 2 * std)**2
        return (1 / (std * np.sqrt(2 * np.pi))) * np.exp(exponent)

    @staticmethod
    def calculate_statistics(x_values):
        # Returns a list with length of input features. Each element is a tu
        n_feats = x_values.shape[1]
        return [(np.average(x_values[:,n]),np.std(x_values[:,n])) for n in r

    @staticmethod
    def calculate_prob(x_input,stats):
        """Calculate the probability that the input features belong to a spe
        x_input: np.array shape(nfeatures)
        stats: list of tuple [(mean1,std1),(means2,std2),...]
        """
        init_prob = 1.0
        for i, (mean, std) in enumerate(stats):
            feature_P = NaiveBayesClassifier.gaussian(x_input[i], mean, std)
            init_prob *= feature_P
        return init_prob

    def fit(self,xs,ys):
        # Train the classifier by calculating the statistics of different fe
        self.ndata = len(ys)
        for y in set(ys):
            type_filter = (ys==y)
            self.type_indices[y] = type_filter
            self.type_stats[y] = self.calculate_statistics(xs[type_filter])
        self.trained = True

    def predict(self,xs):
        # Do the prediction by outputing the class that has highest probabil
        if len(xs.shape) > 1:
            print("Only accepts one sample at a time!")
        if self.trained:
            guess = None
            max_prob = 0
            # P(C|X) = P(X|C)*P(C) / sum_i(P(X|C_i)*P(C_i)) (deniminator for
            for y_type in self.type_stats:
                prob = self.calculate_prob(xs, self.type_stats[y_type])
                if prob > max_prob:
                    max_prob = prob
                    guess = y_type
                    # use to troubleshoot
                    # print (f'max prob {max_prob}, variable prob {prob}')
```

```
            return guess
        else:
            print("Please train the classifier first!")
```

I chose this function form because all the attributes are continuous; it is appopriate to represent them using a gaussian function.

Now to find P(alcohol % 13 | Cultivar 1):

In [6]:
```
# INITIATE CLASSIFIER
Classifier1 = NaiveBayesClassifier()

# SELECT FEATURES AND LABELS FROM DATAFRAME
x_values = wines.iloc[:, :13].values
y_values = wines['Cultivar'].values

Classifier1.fit(x_values, y_values)

# GIVEN MEAN AND STD OF CULTIVAR 1, FIND
# LIKELIHOOD OF IT HAVING ALCOHOL % OF 13
mean, std = Classifier1.type_stats[1][0]

P_of_13_given_cultivar_1 = Classifier1.gaussian(13, mean, std)
print(f'The probability of a wine fron Cultivar 1 having an alcohol % of 13%
```

```
The probability of a wine fron Cultivar 1 having an alcohol % of 13% is 0.50
46355789936335
```

2b

In [7]:
```
# NORMALIZE DATA
def normalize_data(df):
    return df.iloc[:, :-2].apply(lambda x: (x - x.mean()) / x.std())
    # "iloc[:, :-1]" to leave out Cultivar column

wines_normalized = normalize_data(wines.copy())
wines_normalized['Cultivar'] = wines['Cultivar']
#wines_normalized.head()

features = wines.drop(columns='Cultivar').values
scaler = StandardScaler()
x_norm = scaler.fit_transform(features)
```

In the next cell, divide the data into three groups (folds) and predict using Naive Bayes.

In [8]:
```
n_folds = 3     # for 3-fold cross validation
train_sets = []
test_sets = []

# ITERATE THROUGH FOLDS
for i in range(n_folds):
    X_train_df, X_test_df, y_train_series, y_test = train_test_split(wines_r
    # variables on LHS are dataframes and series. Convert to numpy arrays
    X_train = X_train_df.values
```

```
    X_test = X_test_df.values
    y_train = y_train_series.values
    # add variables to their respective sets
    train_sets.append((X_train, y_train))
    test_sets.append((X_test, y_test))

# initiate model. fit and predict
NB_classifier = NaiveBayesClassifier()
NB_classifier.fit(X_train, y_train)
y_predictions = np.array([NB_classifier.predict(X_train[0])])
```

```
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2531548476.p
y:12: RuntimeWarning: divide by zero encountered in scalar divide
  return (1 / (std * np.sqrt(2 * np.pi))) * np.exp(exponent)
```

In [9]:

```
# calculating accuracy of naive bayes
def calculate_accuracy(model,xs,ys):
    y_pred = np.zeros_like(ys)
    for idx,x in enumerate(xs):
        y_pred[idx] = model.predict(x)
    return np.sum(ys==y_pred)/len(ys)

print(f'The accuracy of the Naive Bayes classifier is {calculate_accuracy(NE
```

```
The accuracy of the Naive Bayes classifier is 0.36666666666666664
```

```
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2531548476.p
y:12: RuntimeWarning: divide by zero encountered in scalar divide
  return (1 / (std * np.sqrt(2 * np.pi))) * np.exp(exponent)
```

I heard in lecture that Naive Bayes should be ~0.9 accuracy, so I know that it is more accurate than other methods but unfortunately was not able to demonstrate that in my code.

Naive Bayes outperforms Simulates Annealing (which was used in assignment 2, questions 2d and 2e), Naive Bayes performs much better; Simulated Annealing had an average accuracy of 0.56.

3. Softmax and Cross Entropy Loss

In [10]:

```
# Define a simple neural network with softmax activation
class SimpleNN(torch.nn.Module):
    def __init__(self, input_size, num_classes):
        super(SimpleNN, self).__init__()
        self.fc = torch.nn.Linear(input_size, num_classes)

    def forward(self, x):
        x = self.fc(x)
        return torch.softmax(x, dim=0)  # Apply softmax activation

# Same as above WITHOUT softmax activation
class SimpleNNWithoutSoftmax(torch.nn.Module):
    def __init__(self, input_size, num_classes):
        super(SimpleNNWithoutSoftmax, self).__init__()
```

```
        self.fc = torch.nn.Linear(input_size, num_classes)

    def forward(self, x):
        x = self.fc(x)
        return x  # No softmax activation
```

In [23]:
```
# change featues and labels to tensors
wines_train_X = torch.tensor(x_values, dtype=torch.float32)
wines_train_y = torch.tensor(y_values, dtype=torch.long)
#x_values.shape
```

3a

The softmax activation function transforms the raw model outputs into a probability distribution (all observations add up to 1). Outputs are slightly different with each run, so I have printed each one 3 times.

In [31]:
```
# Define inputs and model
input_size = 13 # number of features
num_classes = 3  # number of cultivars

model_w_softmax = SimpleNN(input_size, num_classes)
# change wine features to tensor!
#train_X = torch.tensor(x_values, dtype=torch.float32)
# Pass the wine features through the network once without backpropagation
with torch.no_grad():
    output = model_w_softmax(wines_train_X)
print(f"Output with softmax activation: {output}")

# Define the model without softmax
model_wo_softmax = SimpleNNWithoutSoftmax(input_size, num_classes)
# Pass the data through the network one without backpropagation and without
with torch.no_grad():
    output_no_softmax = model_wo_softmax(wines_train_X)
```

```
Output with softmax activation: tensor([[0.0000e+00, 1.0807e-12, 4.4635e-0
5],
        [0.0000e+00, 3.6375e-20, 4.8798e-09],
        [0.0000e+00, 2.2876e-15, 3.3388e-09],
        [0.0000e+00, 1.4835e-10, 1.5533e-06],
        [2.9988e-43, 3.9437e-10, 7.0693e-07],
        [7.7071e-44, 2.2690e-08, 3.0147e-03],
        [0.0000e+00, 1.0819e-18, 3.0600e-08],
        [0.0000e+00, 3.1917e-16, 4.5885e-09],
        [0.0000e+00, 1.2986e-19, 1.6006e-10],
        [0.0000e+00, 1.1766e-10, 1.7717e-06],
        [0.0000e+00, 9.8000e-16, 4.5734e-09],
        [0.0000e+00, 1.3319e-08, 2.5198e-02],
        [0.0000e+00, 6.7903e-18, 3.7124e-09],
        [0.0000e+00, 4.9743e-14, 7.8982e-08],
        [0.0000e+00, 6.9512e-14, 5.6375e-08],
        [0.0000e+00, 6.1532e-18, 3.7397e-09],
        [0.0000e+00, 2.3291e-14, 1.0432e-07],
        [0.0000e+00, 7.1436e-11, 1.2159e-06],
        [0.0000e+00, 4.5817e-11, 5.5811e-06],
        [1.1210e-44, 3.5464e-09, 5.5499e-05],
        [0.0000e+00, 4.8530e-23, 2.2897e-12],
        [0.0000e+00, 4.2221e-29, 2.6234e-16],
        [0.0000e+00, 3.6384e-28, 5.8679e-17],
        [0.0000e+00, 8.8607e-19, 2.2434e-09],
        [0.0000e+00, 5.1605e-26, 1.7860e-12],
        [0.0000e+00, 1.5556e-25, 1.9576e-14],
        [0.0000e+00, 8.6442e-27, 2.4998e-14],
        [0.0000e+00, 6.1209e-28, 1.3179e-14],
        [0.0000e+00, 4.2333e-24, 2.0663e-13],
        [0.0000e+00, 2.7439e-26, 1.9300e-15],
        [0.0000e+00, 3.7048e-13, 1.5346e-01],
        [0.0000e+00, 1.2100e-21, 3.9395e-08],
        [0.0000e+00, 9.7544e-30, 4.2501e-16],
        [0.0000e+00, 1.2421e-22, 5.2798e-12],
        [0.0000e+00, 1.1780e-29, 2.6129e-16],
        [0.0000e+00, 5.1168e-25, 6.5429e-12],
        [0.0000e+00, 6.7755e-29, 9.8134e-16],
        [0.0000e+00, 4.2713e-30, 1.4040e-16],
        [0.0000e+00, 6.2989e-30, 5.1329e-16],
        [0.0000e+00, 2.7067e-24, 1.1490e-12],
        [0.0000e+00, 3.6704e-30, 2.4493e-16],
        [0.0000e+00, 1.0473e-30, 3.2620e-16],
        [0.0000e+00, 1.8156e-26, 6.5340e-15],
        [0.0000e+00, 4.2624e-25, 3.3105e-13],
        [0.0000e+00, 7.2982e-24, 4.7079e-12],
        [0.0000e+00, 9.4047e-26, 9.4752e-15],
        [0.0000e+00, 1.9193e-24, 1.5801e-12],
        [0.0000e+00, 3.0449e-26, 7.7249e-14],
        [0.0000e+00, 8.3833e-30, 2.0432e-17],
        [0.0000e+00, 5.7185e-24, 5.7934e-12],
        [0.0000e+00, 2.2548e-26, 1.7041e-12],
        [0.0000e+00, 2.0881e-27, 7.0578e-16],
        [0.0000e+00, 3.4610e-22, 2.0031e-11],
        [0.0000e+00, 5.5740e-21, 1.3770e-10],
        [0.0000e+00, 4.2158e-23, 8.6433e-14],
```

```
[0.0000e+00, 2.3460e-23, 8.4240e-14],
[0.0000e+00, 1.2249e-21, 3.7844e-12],
[0.0000e+00, 8.1094e-21, 6.7188e-11],
[0.0000e+00, 2.5647e-15, 6.4980e-09],
[2.4299e-22, 1.1772e-04, 2.2401e-02],
[6.5685e-26, 1.4962e-05, 6.1352e-03],
[0.0000e+00, 5.0578e-15, 7.7452e-09],
[0.0000e+00, 1.0568e-13, 5.6183e-09],
[0.0000e+00, 8.0470e-12, 1.8826e-05],
[0.0000e+00, 8.5106e-21, 1.0542e-10],
[0.0000e+00, 1.2119e-17, 9.3234e-07],
[0.0000e+00, 3.1985e-17, 1.2084e-08],
[1.4013e-45, 1.2420e-09, 4.7684e-05],
[0.0000e+00, 4.2305e-16, 2.3116e-08],
[0.0000e+00, 2.6556e-13, 2.1810e-06],
[0.0000e+00, 5.6323e-16, 5.8931e-09],
[0.0000e+00, 3.7653e-19, 1.6739e-08],
[0.0000e+00, 1.3583e-22, 9.1214e-12],
[0.0000e+00, 2.0321e-13, 1.5702e-06],
[0.0000e+00, 3.8106e-16, 4.9041e-09],
[0.0000e+00, 1.5734e-13, 1.2635e-08],
[5.5661e-34, 6.1357e-07, 3.8264e-03],
[0.0000e+00, 2.3213e-13, 3.4748e-06],
[0.0000e+00, 9.1969e-28, 3.3047e-16],
[0.0000e+00, 2.3850e-24, 1.5365e-13],
[0.0000e+00, 1.6578e-29, 5.8428e-15],
[0.0000e+00, 6.0171e-28, 2.4001e-16],
[0.0000e+00, 3.4542e-29, 6.5899e-16],
[0.0000e+00, 6.7746e-28, 2.6303e-15],
[0.0000e+00, 5.6101e-30, 3.6411e-17],
[0.0000e+00, 4.1515e-19, 3.0608e-07],
[0.0000e+00, 1.8031e-32, 8.6139e-18],
[0.0000e+00, 3.6796e-23, 3.4208e-13],
[0.0000e+00, 3.9114e-26, 1.1638e-13],
[0.0000e+00, 1.0310e-28, 2.6711e-16],
[0.0000e+00, 2.8440e-28, 4.0246e-16],
[0.0000e+00, 3.5471e-32, 1.8841e-17],
[0.0000e+00, 1.8749e-26, 1.2238e-14],
[0.0000e+00, 4.5383e-28, 1.7532e-14],
[0.0000e+00, 3.3371e-24, 1.5703e-13],
[0.0000e+00, 2.5529e-31, 1.0300e-16],
[0.0000e+00, 2.3842e-23, 2.9195e-12],
[0.0000e+00, 3.5930e-24, 3.5603e-12],
[0.0000e+00, 5.7832e-28, 3.4490e-15],
[0.0000e+00, 5.3298e-31, 4.4784e-18],
[0.0000e+00, 1.2451e-26, 1.9607e-14],
[0.0000e+00, 1.7297e-30, 8.2228e-17],
[0.0000e+00, 1.3550e-27, 5.8253e-14],
[0.0000e+00, 1.6966e-25, 3.0634e-13],
[0.0000e+00, 7.0095e-23, 1.0139e-12],
[0.0000e+00, 9.0812e-25, 2.6044e-13],
[0.0000e+00, 1.9675e-21, 2.0625e-12],
[0.0000e+00, 3.5817e-26, 2.0317e-14],
[0.0000e+00, 1.4063e-19, 4.7341e-10],
[0.0000e+00, 1.4057e-23, 1.9136e-13],
[0.0000e+00, 2.6549e-18, 1.3393e-09],
```

```
[0.0000e+00, 1.0410e-26, 3.0037e-15],
[0.0000e+00, 1.4880e-24, 2.4741e-12],
[0.0000e+00, 1.7226e-22, 1.3251e-11],
[0.0000e+00, 8.5314e-27, 4.6389e-15],
[0.0000e+00, 1.1272e-26, 1.7730e-14],
[0.0000e+00, 9.3781e-28, 2.0887e-16],
[0.0000e+00, 5.5651e-12, 2.1324e-06],
[1.4013e-45, 2.2900e-10, 1.5724e-06],
[3.2832e-42, 2.8065e-08, 3.5581e-03],
[4.2117e-20, 7.5031e-05, 1.2783e-02],
[2.2286e-16, 2.2177e-04, 2.2398e-03],
[1.3878e-41, 2.1094e-08, 3.1207e-04],
[1.0000e+00, 9.9940e-01, 7.3569e-01],
[0.0000e+00, 3.8434e-18, 8.8494e-09],
[0.0000e+00, 2.2467e-15, 2.9028e-08],
[0.0000e+00, 1.7830e-12, 1.0795e-07],
[2.1373e-19, 1.7395e-04, 2.0207e-02],
[0.0000e+00, 1.5630e-17, 3.0036e-09],
[0.0000e+00, 5.4540e-19, 4.1743e-09],
[0.0000e+00, 4.7698e-16, 2.4257e-09],
[0.0000e+00, 1.4462e-14, 5.4366e-08],
[0.0000e+00, 1.5154e-09, 3.2773e-05],
[0.0000e+00, 4.3381e-12, 2.7933e-05],
[0.0000e+00, 3.3892e-15, 3.0371e-07],
[0.0000e+00, 4.3956e-10, 9.0507e-06],
[0.0000e+00, 1.3520e-27, 5.3076e-15],
[0.0000e+00, 5.5732e-23, 2.6415e-12],
[0.0000e+00, 8.9725e-21, 1.4513e-10],
[0.0000e+00, 9.8322e-19, 1.6541e-05],
[0.0000e+00, 4.3760e-13, 1.0902e-02],
[0.0000e+00, 1.9729e-18, 2.3267e-09],
[0.0000e+00, 1.3344e-28, 2.6780e-15],
[0.0000e+00, 2.6834e-27, 1.6354e-13],
[0.0000e+00, 3.1328e-27, 3.4925e-15],
[0.0000e+00, 1.6201e-27, 1.3946e-14],
[0.0000e+00, 9.8730e-29, 1.1201e-15],
[0.0000e+00, 3.9424e-30, 8.0388e-16],
[0.0000e+00, 1.8328e-29, 1.1272e-16],
[0.0000e+00, 6.2016e-24, 1.2840e-13],
[0.0000e+00, 2.7444e-31, 1.4698e-16],
[0.0000e+00, 5.5836e-28, 1.7620e-15],
[0.0000e+00, 3.3168e-31, 6.8134e-17],
[0.0000e+00, 2.5407e-29, 4.8528e-14],
[0.0000e+00, 4.9713e-25, 1.0993e-10],
[0.0000e+00, 3.5204e-29, 5.2348e-14],
[0.0000e+00, 1.6910e-30, 1.1155e-16],
[0.0000e+00, 3.0152e-30, 1.3391e-16],
[0.0000e+00, 2.8550e-22, 3.0684e-10],
[0.0000e+00, 1.2866e-24, 1.4750e-14],
[0.0000e+00, 8.2427e-23, 6.3303e-13],
[0.0000e+00, 2.1420e-26, 1.1650e-13],
[0.0000e+00, 4.7590e-18, 5.7802e-09],
[0.0000e+00, 1.1872e-24, 2.6460e-14],
[0.0000e+00, 3.5614e-24, 4.7763e-11],
[0.0000e+00, 2.4641e-25, 7.1879e-13],
[0.0000e+00, 1.2950e-22, 7.9545e-13],
```

```
        [0.0000e+00, 3.4737e-23, 2.6891e-12],
        [0.0000e+00, 4.6533e-22, 2.6364e-12],
        [0.0000e+00, 2.7027e-22, 1.6401e-12],
        [0.0000e+00, 4.6637e-24, 3.4338e-14],
        [0.0000e+00, 2.5707e-24, 4.2538e-14],
        [0.0000e+00, 2.2735e-20, 1.8427e-10],
        [0.0000e+00, 2.4811e-22, 7.6339e-11],
        [0.0000e+00, 1.3414e-17, 2.6595e-08],
        [0.0000e+00, 1.4304e-17, 3.0881e-08],
        [0.0000e+00, 6.0690e-25, 1.0405e-13],
        [0.0000e+00, 5.5316e-28, 5.1829e-16]])
```

In [13]:
```
print(f"Output with softmax activation: {output}")
```

```
Output with softmax activation: tensor([[4.2039e-44, 4.7292e-33, 0.0000e+0
0],
        [1.6539e-25, 0.0000e+00, 4.3836e-28],
        [2.8026e-44, 1.3507e-30, 5.7341e-42],
        [0.0000e+00, 8.9663e-19, 0.0000e+00],
        [0.0000e+00, 3.3238e-16, 0.0000e+00],
        [0.0000e+00, 1.9144e-21, 0.0000e+00],
        [8.4555e-28, 0.0000e+00, 4.0589e-31],
        [9.2486e-43, 1.0192e-31, 3.2765e-40],
        [7.8513e-33, 5.8742e-40, 1.8481e-31],
        [0.0000e+00, 1.5584e-18, 0.0000e+00],
        [7.4269e-44, 5.6274e-31, 2.4076e-41],
        [0.0000e+00, 4.0563e-25, 0.0000e+00],
        [2.6769e-34, 5.0802e-40, 9.6662e-35],
        [0.0000e+00, 1.2979e-27, 2.8026e-45],
        [0.0000e+00, 1.4065e-28, 5.6052e-45],
        [1.0106e-34, 1.7258e-39, 1.2031e-34],
        [7.0065e-45, 2.6186e-30, 1.7656e-43],
        [0.0000e+00, 2.3130e-19, 0.0000e+00],
        [0.0000e+00, 3.4431e-25, 0.0000e+00],
        [0.0000e+00, 5.1969e-20, 0.0000e+00],
        [2.4178e-23, 0.0000e+00, 1.7405e-23],
        [6.4583e-09, 0.0000e+00, 2.0893e-08],
        [3.8648e-14, 0.0000e+00, 1.2404e-11],
        [5.3257e-34, 1.5050e-39, 1.6959e-33],
        [1.5733e-12, 0.0000e+00, 4.0622e-15],
        [2.6604e-21, 0.0000e+00, 1.7749e-18],
        [2.5110e-14, 0.0000e+00, 3.8103e-14],
        [4.4282e-10, 0.0000e+00, 4.2538e-11],
        [4.7646e-24, 0.0000e+00, 1.1337e-21],
        [2.1216e-21, 0.0000e+00, 2.3315e-17],
        [6.9576e-35, 2.3262e-43, 7.8052e-43],
        [1.7150e-18, 0.0000e+00, 6.6716e-24],
        [4.8560e-08, 0.0000e+00, 1.3342e-07],
        [3.2532e-25, 0.0000e+00, 1.8203e-24],
        [1.0918e-08, 0.0000e+00, 5.3437e-08],
        [3.7898e-16, 0.0000e+00, 1.1330e-17],
        [1.1323e-09, 0.0000e+00, 2.9782e-09],
        [5.0063e-07, 0.0000e+00, 2.5501e-06],
        [2.8031e-08, 0.0000e+00, 1.9392e-07],
        [2.5581e-20, 0.0000e+00, 6.0158e-20],
        [1.0522e-06, 0.0000e+00, 3.2719e-06],
        [3.5392e-04, 0.0000e+00, 2.1233e-04],
        [1.5598e-18, 0.0000e+00, 4.2619e-16],
        [1.4866e-16, 0.0000e+00, 4.2824e-17],
        [1.5986e-18, 0.0000e+00, 5.7014e-20],
        [2.0585e-18, 0.0000e+00, 3.9054e-17],
        [3.1126e-18, 0.0000e+00, 4.5606e-19],
        [2.3932e-14, 0.0000e+00, 5.8462e-15],
        [2.8570e-09, 0.0000e+00, 1.0786e-07],
        [1.2341e-15, 0.0000e+00, 3.0575e-18],
        [6.1849e-08, 0.0000e+00, 5.2361e-12],
        [8.4365e-13, 0.0000e+00, 3.9639e-12],
        [3.2982e-23, 0.0000e+00, 2.4506e-24],
        [3.8354e-24, 0.0000e+00, 9.8442e-26],
        [8.6697e-25, 0.0000e+00, 9.0727e-23],
```

```
        [5.4935e-23, 0.0000e+00, 9.1679e-22],
        [1.9377e-27, 5.6052e-45, 2.4616e-26],
        [1.7591e-27, 4.2039e-45, 1.5319e-27],
        [1.2612e-44, 6.0068e-31, 1.9128e-42],
        [0.0000e+00, 3.4832e-11, 0.0000e+00],
        [0.0000e+00, 1.4501e-12, 0.0000e+00],
        [1.9618e-44, 1.4354e-30, 3.6462e-42],
        [0.0000e+00, 6.3410e-25, 0.0000e+00],
        [0.0000e+00, 4.0280e-28, 0.0000e+00],
        [1.7375e-28, 1.8217e-44, 3.4536e-28],
        [1.5585e-30, 1.2752e-43, 1.1295e-33],
        [2.4924e-36, 8.3169e-38, 2.6008e-36],
        [0.0000e+00, 6.6365e-19, 0.0000e+00],
        [9.0515e-41, 5.6939e-34, 8.3796e-40],
        [0.0000e+00, 2.0413e-29, 0.0000e+00],
        [7.8192e-43, 6.5466e-32, 9.8458e-41],
        [1.1242e-26, 0.0000e+00, 2.9358e-30],
        [3.1744e-23, 0.0000e+00, 1.3017e-23],
        [1.4013e-45, 2.1226e-30, 2.8026e-45],
        [1.0022e-40, 6.8044e-34, 2.8821e-39],
        [0.0000e+00, 4.9228e-25, 0.0000e+00],
        [0.0000e+00, 1.7416e-16, 0.0000e+00],
        [2.9427e-44, 3.6251e-32, 5.6052e-45],
        [7.9878e-15, 0.0000e+00, 1.3327e-13],
        [1.1625e-20, 0.0000e+00, 3.9307e-20],
        [1.8572e-04, 0.0000e+00, 9.0569e-07],
        [1.0574e-14, 0.0000e+00, 3.3383e-12],
        [3.9379e-08, 0.0000e+00, 7.1160e-08],
        [8.0254e-12, 0.0000e+00, 3.0634e-11],
        [1.7581e-07, 0.0000e+00, 8.8486e-07],
        [1.4056e-25, 0.0000e+00, 2.1834e-30],
        [7.8917e-01, 0.0000e+00, 8.5079e-01],
        [7.1060e-26, 2.8026e-45, 1.4506e-23],
        [5.1567e-17, 0.0000e+00, 5.3690e-16],
        [1.0573e-12, 0.0000e+00, 7.7589e-11],
        [1.5462e-13, 0.0000e+00, 1.3675e-11],
        [1.8000e-01, 0.0000e+00, 1.4029e-01],
        [4.7314e-17, 0.0000e+00, 8.3416e-16],
        [1.8052e-09, 0.0000e+00, 1.7806e-10],
        [1.6307e-23, 0.0000e+00, 2.7089e-21],
        [1.7611e-02, 0.0000e+00, 3.7681e-03],
        [2.0665e-23, 0.0000e+00, 1.6426e-22],
        [4.4358e-19, 0.0000e+00, 7.3366e-20],
        [2.7721e-13, 0.0000e+00, 4.1531e-12],
        [9.6459e-07, 0.0000e+00, 3.3579e-05],
        [3.0931e-17, 0.0000e+00, 9.0915e-16],
        [5.7089e-05, 0.0000e+00, 9.7161e-05],
        [3.6363e-11, 0.0000e+00, 1.8742e-11],
        [1.2242e-14, 0.0000e+00, 4.5851e-16],
        [2.5199e-26, 2.8026e-45, 1.7108e-24],
        [4.3191e-19, 0.0000e+00, 6.0371e-19],
        [8.4781e-30, 2.6358e-42, 1.2839e-27],
        [2.0008e-16, 0.0000e+00, 6.0068e-16],
        [7.0013e-32, 1.7879e-41, 1.7302e-31],
        [2.5600e-22, 0.0000e+00, 2.5200e-21],
        [3.5621e-35, 5.2125e-38, 6.4667e-34],
```

```
[4.1993e-15, 0.0000e+00, 4.0199e-14],
[5.8285e-17, 0.0000e+00, 1.7973e-18],
[5.6101e-23, 0.0000e+00, 7.8495e-24],
[5.5394e-15, 0.0000e+00, 4.4320e-14],
[4.5187e-14, 0.0000e+00, 3.2447e-14],
[1.6883e-12, 0.0000e+00, 2.3961e-11],
[0.0000e+00, 4.5195e-24, 0.0000e+00],
[0.0000e+00, 1.3630e-18, 0.0000e+00],
[0.0000e+00, 4.4661e-21, 0.0000e+00],
[0.0000e+00, 9.3463e-09, 0.0000e+00],
[0.0000e+00, 6.9630e-07, 0.0000e+00],
[0.0000e+00, 3.2838e-19, 0.0000e+00],
[0.0000e+00, 1.0000e+00, 0.0000e+00],
[4.4450e-32, 1.6045e-42, 1.6949e-33],
[1.4994e-43, 1.6817e-31, 8.3938e-42],
[0.0000e+00, 9.2287e-23, 0.0000e+00],
[0.0000e+00, 1.7201e-08, 0.0000e+00],
[5.5722e-37, 1.0618e-36, 7.6952e-36],
[4.0659e-29, 5.6052e-45, 5.3477e-31],
[3.9236e-44, 3.4513e-30, 8.1711e-41],
[2.8026e-45, 2.8086e-30, 1.4714e-43],
[0.0000e+00, 4.0151e-21, 0.0000e+00],
[0.0000e+00, 7.4877e-29, 0.0000e+00],
[4.2677e-39, 1.3897e-36, 2.6798e-40],
[0.0000e+00, 4.4854e-20, 0.0000e+00],
[2.6275e-10, 0.0000e+00, 1.8042e-11],
[3.1204e-23, 0.0000e+00, 5.9996e-23],
[8.7160e-27, 0.0000e+00, 4.2966e-28],
[4.3986e-23, 0.0000e+00, 2.8054e-30],
[1.5008e-38, 1.1654e-37, 2.2841e-43],
[5.8749e-35, 1.1796e-38, 3.6278e-34],
[3.6673e-09, 0.0000e+00, 8.6827e-10],
[1.0955e-10, 0.0000e+00, 4.0628e-12],
[1.2876e-14, 0.0000e+00, 2.5066e-13],
[5.1515e-13, 0.0000e+00, 1.2315e-12],
[8.4270e-11, 0.0000e+00, 6.8147e-10],
[3.1100e-04, 0.0000e+00, 1.3486e-05],
[1.6693e-09, 0.0000e+00, 1.8168e-08],
[1.4817e-22, 0.0000e+00, 5.6804e-21],
[8.1175e-03, 0.0000e+00, 3.2065e-03],
[6.4400e-13, 0.0000e+00, 1.2037e-11],
[2.4635e-03, 0.0000e+00, 1.5596e-03],
[1.6790e-03, 0.0000e+00, 1.4106e-06],
[7.4278e-10, 0.0000e+00, 3.7402e-14],
[5.4094e-05, 0.0000e+00, 3.6979e-07],
[6.8061e-07, 0.0000e+00, 1.1720e-05],
[1.3582e-06, 0.0000e+00, 5.7015e-06],
[2.3529e-19, 0.0000e+00, 5.3890e-23],
[1.7032e-22, 0.0000e+00, 2.2709e-20],
[7.9875e-25, 0.0000e+00, 9.5612e-24],
[2.8129e-14, 0.0000e+00, 1.1477e-14],
[2.5881e-33, 1.9491e-41, 6.4096e-34],
[2.9233e-21, 0.0000e+00, 2.0997e-19],
[4.5517e-12, 0.0000e+00, 1.2104e-16],
[1.0093e-11, 0.0000e+00, 1.6032e-14],
[1.0996e-23, 0.0000e+00, 3.1655e-23],
```

```
        [3.6497e-21, 0.0000e+00, 9.0153e-22],
        [1.2191e-26, 1.4013e-45, 2.4231e-25],
        [1.2054e-22, 0.0000e+00, 1.1762e-22],
        [7.9292e-21, 0.0000e+00, 1.4724e-19],
        [1.8667e-20, 0.0000e+00, 2.4638e-19],
        [2.4848e-27, 2.8026e-45, 6.9955e-28],
        [3.2567e-20, 0.0000e+00, 1.9972e-22],
        [7.9044e-32, 3.3911e-43, 7.1137e-34],
        [5.6993e-32, 6.6982e-43, 4.5927e-34],
        [5.9675e-17, 0.0000e+00, 3.4317e-17],
        [1.6312e-14, 0.0000e+00, 2.8051e-12]])
```

In [26]:
```
print(f"Output with softmax activation: {output}")
```

```
Output with softmax activation: tensor([[6.5089e-38, 3.8771e-13, 2.3017e-1
9],
        [0.0000e+00, 1.2393e-17, 5.6340e-29],
        [2.6289e-35, 1.7317e-10, 9.4923e-19],
        [4.8645e-22, 4.2486e-06, 7.4933e-12],
        [5.3045e-19, 7.3404e-05, 1.6193e-10],
        [2.8268e-25, 1.4714e-08, 6.5907e-13],
        [0.0000e+00, 6.3750e-18, 9.1106e-28],
        [1.9912e-37, 1.2959e-10, 1.0247e-19],
        [0.0000e+00, 1.1955e-13, 1.1266e-24],
        [8.4441e-22, 8.5763e-06, 1.1186e-11],
        [5.0702e-36, 1.9008e-10, 4.4512e-19],
        [2.3588e-29, 1.5912e-10, 1.2513e-14],
        [0.0000e+00, 1.6163e-14, 2.4978e-24],
        [1.1999e-32, 2.8682e-09, 3.1186e-17],
        [3.2684e-33, 4.0591e-10, 1.5594e-17],
        [0.0000e+00, 4.9384e-14, 4.8922e-24],
        [1.2504e-35, 1.2730e-10, 8.7443e-19],
        [8.7785e-23, 2.5844e-06, 2.7044e-12],
        [7.4251e-29, 1.7162e-09, 3.3782e-15],
        [2.4376e-23, 2.2459e-07, 2.6048e-12],
        [0.0000e+00, 1.0317e-16, 5.6071e-30],
        [0.0000e+00, 6.5115e-20, 5.0989e-37],
        [0.0000e+00, 9.1569e-18, 3.4608e-34],
        [0.0000e+00, 2.1678e-13, 1.6265e-24],
        [0.0000e+00, 6.4393e-21, 8.0903e-36],
        [0.0000e+00, 4.4038e-15, 1.7624e-30],
        [0.0000e+00, 1.0461e-18, 2.2935e-34],
        [0.0000e+00, 1.4961e-20, 1.8136e-36],
        [0.0000e+00, 6.5490e-15, 2.1471e-29],
        [0.0000e+00, 2.1018e-14, 2.9226e-30],
        [0.0000e+00, 3.9431e-18, 1.9437e-24],
        [0.0000e+00, 1.0491e-20, 5.5666e-33],
        [0.0000e+00, 3.1840e-20, 2.1556e-37],
        [0.0000e+00, 6.7820e-16, 7.4710e-29],
        [0.0000e+00, 1.2800e-19, 3.7905e-37],
        [0.0000e+00, 3.8102e-19, 1.0191e-33],
        [0.0000e+00, 1.8203e-19, 9.2030e-37],
        [0.0000e+00, 4.2703e-20, 5.5814e-38],
        [0.0000e+00, 1.2204e-19, 2.6409e-37],
        [0.0000e+00, 3.2595e-17, 2.0152e-31],
        [0.0000e+00, 1.3519e-20, 2.5040e-38],
        [0.0000e+00, 4.8624e-21, 2.5735e-39],
        [0.0000e+00, 2.4638e-16, 2.5784e-32],
        [0.0000e+00, 2.6572e-18, 7.4231e-34],
        [0.0000e+00, 1.5585e-18, 5.9016e-33],
        [0.0000e+00, 3.3223e-17, 7.7210e-33],
        [0.0000e+00, 3.6921e-18, 5.5578e-33],
        [0.0000e+00, 4.6784e-19, 7.7224e-35],
        [0.0000e+00, 2.8247e-19, 1.6941e-37],
        [0.0000e+00, 3.1139e-20, 3.9775e-35],
        [0.0000e+00, 2.3516e-22, 1.3809e-38],
        [0.0000e+00, 4.0438e-19, 3.5425e-36],
        [0.0000e+00, 2.0923e-17, 1.4130e-30],
        [0.0000e+00, 1.5663e-17, 7.6812e-31],
        [0.0000e+00, 1.7547e-15, 2.8709e-30],
```

```
[0.0000e+00, 3.3538e-16, 7.2052e-31],
[0.0000e+00, 2.7323e-15, 1.0118e-28],
[0.0000e+00, 1.4527e-15, 1.4605e-28],
[2.2338e-35, 9.2119e-11, 8.3146e-19],
[1.0160e-12, 1.2253e-04, 9.3077e-07],
[3.3073e-14, 5.1815e-05, 1.5460e-07],
[1.0693e-35, 1.3192e-10, 4.7334e-19],
[2.4734e-28, 3.6776e-08, 2.3030e-15],
[2.6010e-33, 1.2131e-10, 3.3871e-17],
[0.0000e+00, 1.3747e-15, 1.8729e-27],
[0.0000e+00, 1.9140e-16, 2.5187e-26],
[1.4013e-44, 1.3139e-13, 4.6587e-23],
[5.1286e-23, 2.0434e-06, 5.2495e-12],
[1.0050e-39, 6.5690e-12, 1.0943e-20],
[1.1172e-34, 1.1116e-10, 6.3035e-18],
[2.8954e-37, 8.2945e-11, 1.3173e-19],
[0.0000e+00, 1.1872e-18, 1.4445e-28],
[0.0000e+00, 3.0169e-17, 4.1561e-30],
[1.4942e-35, 2.6663e-11, 1.4913e-18],
[1.7605e-39, 7.2105e-12, 7.9399e-21],
[1.2771e-28, 4.2135e-08, 8.4507e-16],
[4.5075e-19, 1.6393e-06, 5.9235e-10],
[2.6082e-37, 3.1068e-12, 2.5753e-19],
[0.0000e+00, 3.3033e-18, 4.8814e-34],
[0.0000e+00, 4.8174e-17, 5.0934e-31],
[0.0000e+00, 1.2959e-22, 1.1294e-39],
[0.0000e+00, 1.7069e-17, 3.8438e-34],
[0.0000e+00, 5.8676e-20, 4.0394e-37],
[0.0000e+00, 6.6763e-19, 1.5391e-35],
[0.0000e+00, 2.6320e-20, 7.6154e-38],
[0.0000e+00, 7.3662e-19, 4.2603e-29],
[0.0000e+00, 3.6574e-22, 4.4735e-41],
[0.0000e+00, 8.6286e-15, 2.6112e-28],
[0.0000e+00, 2.8926e-17, 5.4830e-33],
[0.0000e+00, 4.6044e-18, 4.1481e-35],
[0.0000e+00, 1.2149e-17, 1.2163e-34],
[0.0000e+00, 2.9299e-22, 7.0875e-41],
[0.0000e+00, 2.1011e-17, 7.7965e-33],
[0.0000e+00, 1.1813e-20, 5.6571e-37],
[0.0000e+00, 2.6523e-15, 1.8536e-29],
[0.0000e+00, 4.6912e-22, 2.3418e-40],
[0.0000e+00, 5.7192e-16, 1.4372e-29],
[0.0000e+00, 2.9269e-18, 2.0777e-32],
[0.0000e+00, 2.7971e-18, 9.6204e-35],
[0.0000e+00, 4.5661e-20, 1.9304e-38],
[0.0000e+00, 2.7942e-17, 8.7803e-33],
[0.0000e+00, 6.9674e-21, 4.7940e-39],
[0.0000e+00, 3.5506e-19, 4.5911e-36],
[0.0000e+00, 1.3354e-19, 6.1765e-35],
[0.0000e+00, 9.9458e-15, 7.7321e-29],
[0.0000e+00, 1.3598e-17, 1.7934e-32],
[0.0000e+00, 6.9519e-14, 6.7490e-27],
[0.0000e+00, 3.2795e-18, 8.6196e-34],
[0.0000e+00, 4.3796e-14, 7.5150e-26],
[0.0000e+00, 2.4336e-16, 3.2813e-31],
[0.0000e+00, 1.2797e-12, 1.2897e-24],
```

```
        [0.0000e+00, 3.6204e-18, 5.9604e-35],
        [0.0000e+00, 4.4723e-19, 7.3880e-34],
        [0.0000e+00, 2.4625e-17, 9.9778e-31],
        [0.0000e+00, 3.3528e-18, 8.9038e-35],
        [0.0000e+00, 4.1364e-19, 2.4758e-35],
        [0.0000e+00, 6.5450e-19, 1.7249e-36],
        [2.3491e-28, 2.9547e-08, 4.7320e-15],
        [2.5183e-21, 3.4775e-06, 1.4639e-11],
        [2.0897e-24, 1.4985e-08, 2.0119e-12],
        [3.5501e-10, 3.5268e-03, 2.0106e-05],
        [4.8470e-07, 1.9295e-02, 2.8058e-04],
        [2.0323e-22, 2.1333e-07, 7.0950e-12],
        [1.0000e+00, 9.7285e-01, 9.9968e-01],
        [0.0000e+00, 6.9665e-16, 1.2263e-25],
        [8.6926e-37, 5.0209e-11, 3.5385e-19],
        [9.4568e-27, 1.9372e-07, 2.4987e-14],
        [3.3791e-10, 4.0583e-03, 1.6562e-05],
        [1.4433e-43, 9.8489e-13, 9.6241e-23],
        [0.0000e+00, 1.1084e-16, 4.2294e-27],
        [1.0821e-35, 7.9274e-10, 4.5470e-19],
        [5.5849e-35, 1.0606e-10, 1.8427e-18],
        [1.0667e-24, 6.8650e-08, 3.0652e-13],
        [5.7199e-34, 8.9534e-11, 1.4946e-17],
        [2.4649e-42, 8.0747e-14, 6.0852e-22],
        [1.4483e-23, 4.9092e-07, 1.3950e-12],
        [0.0000e+00, 1.2697e-20, 5.6369e-37],
        [0.0000e+00, 1.7182e-16, 6.2643e-30],
        [0.0000e+00, 1.8462e-16, 4.2276e-28],
        [0.0000e+00, 6.8773e-21, 1.4139e-30],
        [1.4013e-45, 4.3599e-15, 3.5302e-22],
        [0.0000e+00, 4.3423e-13, 9.1342e-24],
        [0.0000e+00, 1.1348e-20, 2.1055e-37],
        [0.0000e+00, 1.5979e-20, 2.5561e-36],
        [0.0000e+00, 6.1192e-18, 1.5103e-34],
        [0.0000e+00, 1.3323e-18, 5.6468e-35],
        [0.0000e+00, 5.3018e-19, 2.5116e-36],
        [0.0000e+00, 2.8193e-22, 9.2047e-40],
        [0.0000e+00, 1.4513e-19, 1.2592e-36],
        [0.0000e+00, 6.6265e-16, 6.5184e-30],
        [0.0000e+00, 6.8004e-22, 2.5098e-40],
        [0.0000e+00, 3.4654e-18, 5.0889e-35],
        [0.0000e+00, 1.5026e-21, 7.7441e-40],
        [0.0000e+00, 4.6468e-23, 5.7706e-40],
        [0.0000e+00, 2.3967e-22, 3.5615e-37],
        [0.0000e+00, 3.4885e-22, 2.1183e-39],
        [0.0000e+00, 4.4886e-20, 3.6842e-38],
        [0.0000e+00, 2.2176e-20, 3.3359e-38],
        [0.0000e+00, 1.4629e-19, 2.2119e-32],
        [0.0000e+00, 1.7352e-15, 1.4966e-30],
        [0.0000e+00, 9.7651e-16, 8.1790e-30],
        [0.0000e+00, 5.8739e-19, 2.7788e-35],
        [0.0000e+00, 6.8493e-15, 5.3033e-26],
        [0.0000e+00, 2.9133e-16, 8.2248e-32],
        [0.0000e+00, 5.3779e-22, 7.9277e-37],
        [0.0000e+00, 2.7742e-21, 2.9143e-37],
        [0.0000e+00, 1.4223e-16, 6.9633e-31],
```

```
        [0.0000e+00, 2.1267e-17, 9.0420e-32],
        [0.0000e+00, 3.8180e-15, 3.9218e-29],
        [0.0000e+00, 9.9503e-17, 2.6104e-31],
        [0.0000e+00, 1.6542e-16, 3.8480e-32],
        [0.0000e+00, 8.2706e-17, 3.1994e-32],
        [0.0000e+00, 7.7279e-16, 1.0356e-28],
        [0.0000e+00, 9.3393e-19, 1.1160e-32],
        [0.0000e+00, 4.7548e-16, 4.8223e-27],
        [0.0000e+00, 7.8268e-16, 1.0860e-26],
        [0.0000e+00, 2.4519e-18, 4.1473e-34],
        [0.0000e+00, 1.3344e-17, 3.5574e-34]])
```

Below are the outputs without softmax activation:

In [15]:
```
print(f"\n Output without softmax activation:{output_no_softmax}")
```

```
 Output without softmax activation:tensor([[ -84.1552,  267.8842, -149.722
8],
        [ -57.6226,  182.7806, -102.8117],
        [ -82.6024,  264.5943, -146.5335],
        [-101.4118,  325.2742, -179.6516],
        [-104.7699,  336.0167, -185.4267],
        [-101.2071,  323.3951, -179.8936],
        [ -61.4611,  194.2248, -109.5226],
        [ -80.2996,  257.0416, -142.1422],
        [ -66.8265,  213.1294, -118.0160],
        [-101.4486,  326.5923, -180.1434],
        [ -82.0204,  262.0371, -145.0941],
        [ -97.2235,  311.1270, -173.4145],
        [ -69.1328,  221.0978, -123.2735],
        [ -87.1696,  279.8150, -154.8227],
        [ -86.5504,  277.4482, -153.6354],
        [ -69.9119,  222.4422, -124.1565],
        [ -84.1641,  267.6179, -148.9594],
        [-100.6126,  321.6613, -177.7326],
        [ -94.1225,  301.0637, -167.3854],
        [-101.7890,  325.6911, -180.6520],
        [ -52.9462,  169.4838,  -94.8231],
        [ -33.1173,  103.6467,  -58.1725],
        [ -40.1736,  125.5261,  -70.1477],
        [ -68.4525,  218.2208, -121.7626],
        [ -38.6058,  122.1901,  -69.7630],
        [ -49.8519,  158.3549,  -87.3256],
        [ -41.0326,  129.2044,  -72.2556],
        [ -35.1473,  110.7885,  -62.2540],
        [ -53.7157,  170.7140,  -94.8161],
        [ -49.5211,  157.6895,  -86.5050],
        [ -72.7201,  232.2763, -131.2282],
        [ -48.2772,  152.9754,  -87.1525],
        [ -31.6399,  100.1263,  -56.2228],
        [ -55.7925,  177.7622,  -99.3202],
        [ -32.2041,  102.1342,  -57.1791],
        [ -43.8244,  138.5972,  -78.4960],
        [ -34.0651,  107.0172,  -59.9594],
        [ -30.3137,   94.7467,  -53.0000],
        [ -31.9620,  100.5085,  -56.1445],
        [ -49.1299,  156.0126,  -87.1921],
        [ -29.4778,   92.8272,  -52.7836],
        [ -26.8201,   83.3112,  -46.8461],
        [ -45.5091,  144.7788,  -80.8404],
        [ -44.4135,  137.6459,  -78.5780],
        [ -46.8820,  147.5162,  -84.1481],
        [ -45.9909,  143.4419,  -81.5183],
        [ -46.4465,  145.3275,  -82.4546],
        [ -40.8922,  127.6204,  -72.5469],
        [ -32.6815,  101.1060,  -58.2297],
        [ -44.0639,  133.3786,  -78.4186],
        [ -33.6768,  101.3989,  -59.6049],
        [ -38.8488,  116.9735,  -68.2590],
        [ -53.5682,  168.1243,  -95.5499],
        [ -56.0717,  170.7635,  -98.9841],
        [ -55.5159,  169.8673,  -97.5256],
```

```
        [ -53.3876,  163.4633,  -93.5292],
        [ -58.8823,  183.7119, -104.7687],
        [ -59.6626,  186.0260, -105.8151],
        [ -82.6974,  265.9222, -147.3480],
        [-117.4304,  375.5955, -208.3207],
        [-114.7476,  368.0522, -203.9801],
        [ -83.3167,  264.3034, -147.0679],
        [ -91.0718,  292.4468, -161.4199],
        [ -89.4616,  284.9536, -158.8417],
        [ -60.8211,  192.8332, -108.1008],
        [ -65.1981,  207.0914, -115.9319],
        [ -72.2051,  230.3942, -128.0443],
        [-102.0032,  326.0322, -180.3015],
        [ -78.0269,  249.9266, -138.6540],
        [ -86.2952,  276.5113, -153.4552],
        [ -80.2681,  258.0846, -142.8423],
        [ -59.3128,  188.7683, -106.8619],
        [ -53.3153,  169.3797,  -95.3589],
        [ -85.0668,  272.1404, -151.7499],
        [ -78.1466,  248.7080, -138.3756],
        [ -91.7087,  291.8239, -162.1553],
        [-108.7494,  348.3441, -193.3555],
        [ -83.7295,  266.8459, -149.0232],
        [ -39.9766,  129.1788,  -72.0264],
        [ -49.4678,  157.1665,  -87.7806],
        [ -27.4061,   85.5526,  -49.0651],
        [ -40.8857,  127.2424,  -71.2193],
        [ -32.4410,  101.1820,  -56.3133],
        [ -37.4136,  116.5978,  -65.4507],
        [ -31.1061,   96.2131,  -54.6043],
        [ -58.4622,  185.5748, -105.1336],
        [ -21.7124,   67.2068,  -37.9736],
        [ -56.6340,  179.6666,  -99.8261],
        [ -44.5573,  140.0082,  -78.0242],
        [ -37.7560,  119.3244,  -66.6111],
        [ -39.2285,  123.0338,  -68.7813],
        [ -22.4528,   70.0865,  -39.8082],
        [ -44.2413,  140.1062,  -78.1455],
        [ -34.1606,  107.4269,  -60.6541],
        [ -53.2215,  168.8267,  -93.6575],
        [ -24.4274,   75.3500,  -42.9055],
        [ -53.5943,  170.7255,  -94.6474],
        [ -47.6796,  150.5603,  -85.0040],
        [ -38.9658,  123.0370,  -68.5350],
        [ -28.9863,   90.9782,  -51.7826],
        [ -44.0051,  140.6882,  -78.3705],
        [ -27.8534,   86.0212,  -48.6514],
        [ -36.8569,  114.6485,  -64.5568],
        [ -41.6173,  129.6906,  -74.4473],
        [ -56.4516,  179.5520, -100.9795],
        [ -47.3826,  148.4976,  -83.9884],
        [ -62.0369,  195.3715, -109.5570],
        [ -43.0627,  135.6135,  -77.0277],
        [ -65.2045,  207.2862, -116.3725],
        [ -52.1652,  160.7841,  -91.8998],
        [ -70.3569,  219.8394, -124.0377],
```

```
[ -41.5474,  127.5102,  -73.4813],
[ -45.0541,  139.7645,  -80.1233],
[ -53.0720,  166.9733,  -94.9526],
[ -41.6172,  127.5517,  -73.3863],
[ -40.3250,  124.9241,  -71.7680],
[ -38.4433,  114.5267,  -67.0225],
[ -94.1287,  300.3153, -166.5937],
[-102.3367,  328.0020, -181.1877],
[-102.1407,  327.4783, -181.8927],
[-119.6794,  384.0856, -212.2661],
[-122.9124,  393.9369, -217.9223],
[-103.8947,  331.6310, -184.4620],
[-133.5766,  427.5669, -236.8176],
[ -66.3773,  211.5364, -118.7111],
[ -81.7676,  262.0645, -144.9740],
[ -94.5310,  303.6638, -167.6022],
[-120.3013,  384.9980, -213.2020],
[ -73.0171,  231.9160, -128.9779],
[ -62.8841,  198.5274, -111.7440],
[ -81.7334,  261.9782, -145.2618],
[ -83.9996,  269.2909, -149.4884],
[-100.5992,  318.8259, -177.8946],
[ -88.7946,  282.2079, -157.3986],
[ -76.6801,  243.5656, -136.3125],
[-100.5284,  322.2060, -178.4920],
[ -35.3699,  109.5982,  -63.0811],
[ -53.5752,  169.4644,  -94.8499],
[ -58.2405,  187.0788, -104.4034],
[ -55.7039,  176.2780, -100.5746],
[ -77.2317,  246.0052, -137.5150],
[ -69.5959,  223.0546, -123.4990],
[ -33.4581,  104.5882,  -59.8349],
[ -35.9936,  113.5624,  -64.2385],
[ -40.9402,  127.1858,  -72.2270],
[ -38.8879,  122.5026,  -68.5939],
[ -35.8862,  110.9512,  -62.6449],
[ -26.6128,   83.4483,  -47.6220],
[ -33.4863,  106.0098,  -59.2583],
[ -52.1003,  165.7383,  -92.0335],
[ -24.8725,   76.0349,  -43.5653],
[ -38.7775,  120.9540,  -67.8814],
[ -24.9931,   78.9001,  -44.4451],
[ -26.6547,   82.8012,  -47.4406],
[ -37.0854,  113.1590,  -65.2299],
[ -28.2633,   87.9298,  -50.4188],
[ -29.9771,   93.5440,  -52.7549],
[ -29.6446,   92.8961,  -52.1825],
[ -48.8760,  154.8262,  -88.0632],
[ -51.5758,  162.1170,  -91.0384],
[ -55.3940,  172.6788,  -98.0870],
[ -40.5277,  125.7700,  -72.3762],
[ -68.0509,  212.5594, -121.2664],
[ -50.0348,  154.6948,  -88.4284],
[ -39.8426,  120.0310,  -70.9635],
[ -39.0432,  115.2909,  -68.7310],
[ -54.6376,  166.6015,  -96.1938],
```

```
        [ -51.2495,  157.8084,  -90.3697],
        [ -57.9209,  179.8298, -102.6924],
        [ -54.4317,  162.5723,  -93.9853],
        [ -50.9576,  153.0278,  -88.1539],
        [ -49.9042,  151.7486,  -87.2841],
        [ -60.2706,  185.8547, -106.0103],
        [ -50.2323,  154.1531,  -89.2080],
        [ -66.7264,  206.3019, -118.8399],
        [ -67.0831,  208.0101, -119.0985],
        [ -45.1783,  136.9708,  -79.3436],
        [ -40.6196,  127.2582,  -71.1206]])
```

In [28]:
```
print(f"\n Output without softmax activation:{output_no_softmax}")
```

```
 Output without softmax activation:tensor([[ 192.1592,  168.8958, -274.339
9],
        [ 130.3218,  113.4627, -197.5657],
        [ 187.8321,  166.5602, -263.0306],
        [ 230.6033,  205.6793, -318.6050],
        [ 238.0447,  212.8813, -326.3802],
        [ 230.4151,  204.7302, -324.0532],
        [ 139.7504,  121.2977, -207.7227],
        [ 182.0681,  161.1645, -257.6034],
        [ 150.1446,  132.1472, -218.6964],
        [ 231.5585,  206.4053, -319.9852],
        [ 185.8378,  164.6340, -261.4806],
        [ 222.8770,  196.7199, -316.2273],
        [ 158.1201,  138.7063, -228.1807],
        [ 198.4631,  176.3081, -279.0078],
        [ 197.2301,  174.7471, -275.9252],
        [ 158.4207,  139.3878, -228.9405],
        [ 189.8369,  168.7596, -269.0173],
        [ 227.7114,  203.3489, -314.8707],
        [ 215.2217,  190.8535, -299.3351],
        [ 231.8562,  206.4780, -321.6269],
        [ 121.4187,  105.5765, -180.4903],
        [  72.1608,   62.0635, -116.9619],
        [  87.7058,   76.6318, -132.9883],
        [ 155.0258,  136.7527, -226.5262],
        [  88.1581,   74.9494, -141.7902],
        [ 109.5472,   96.4642, -166.5329],
        [  91.0048,   78.6360, -142.0664],
        [  78.2899,   66.4400, -126.8907],
        [ 119.6021,  105.2967, -178.8978],
        [ 108.4934,   95.6522, -163.7616],
        [ 168.4594,  145.9808, -253.0925],
        [ 110.7487,   94.6986, -176.0833],
        [  69.4552,   59.7520, -114.4470],
        [ 126.1407,  110.4247, -187.1092],
        [  71.2170,   60.9290, -116.7159],
        [  98.2400,   85.3315, -154.9058],
        [  74.6339,   64.1845, -121.7093],
        [  65.1918,   55.5114, -110.0227],
        [  69.3114,   59.6018, -115.4164],
        [ 110.0250,   95.8981, -167.7165],
        [  63.9702,   54.9354, -109.2730],
        [  56.4448,   47.8232, -101.3106],
        [ 100.9165,   88.7173, -155.0745],
        [  97.4902,   85.2448, -153.6957],
        [ 105.5275,   91.9779, -164.1280],
        [ 101.9763,   89.3995, -155.2673],
        [ 103.2318,   89.7854, -161.4778],
        [  90.2781,   78.1068, -144.4302],
        [  71.2197,   61.9992, -115.6573],
        [  96.8860,   84.5406, -153.1356],
        [  72.7515,   62.1068, -125.4655],
        [  83.7453,   73.5431, -131.3354],
        [ 120.7253,  105.4227, -182.3855],
        [ 123.2529,  108.8489, -186.6247],
        [ 121.8781,  108.3289, -179.0207],
```

```
[ 116.8290,  103.3835, -173.5144],
[ 131.4573,  116.4726, -194.1105],
[ 132.7420,  117.3924, -198.2428],
[ 190.0182,  167.8153, -264.7856],
[ 267.8001,  238.8720, -367.8916],
[ 262.5514,  234.0766, -360.5877],
[ 187.9437,  166.8300, -263.6252],
[ 207.3062,  184.8084, -285.3925],
[ 202.9044,  179.9852, -288.0244],
[ 137.0500,  120.3794, -202.2297],
[ 146.9972,  128.3823, -222.1990],
[ 163.3432,  143.8036, -236.6729],
[ 230.2101,  205.7560, -322.0893],
[ 177.3620,  156.7650, -253.1496],
[ 196.6129,  173.9724, -279.3597],
[ 183.3026,  162.2467, -258.6068],
[ 136.7548,  118.4080, -203.1035],
[ 120.6413,  105.3276, -181.0148],
[ 193.9763,  171.5521, -276.0218],
[ 176.6947,  156.4516, -250.3290],
[ 207.6744,  185.5637, -285.9522],
[ 248.6987,  221.2546, -344.5927],
[ 191.0207,  168.4613, -271.6561],
[  92.7403,   79.5645, -139.2366],
[ 111.1825,   96.7748, -167.5386],
[  60.9939,   50.7910, -106.2122],
[  88.7874,   77.3816, -136.9309],
[  68.6257,   59.0199, -116.2176],
[  80.6515,   69.7869, -131.0311],
[  67.7433,   57.9626, -109.6968],
[ 134.7167,  116.4295, -203.1223],
[  46.2031,   38.0693,  -84.5026],
[ 126.1777,  111.2106, -185.8546],
[  97.0963,   84.7350, -153.4426],
[  83.2867,   72.3688, -130.7497],
[  85.4156,   74.5976, -134.3848],
[  48.5253,   40.1994,  -87.6417],
[  98.4643,   85.5456, -151.1913],
[  75.5378,   64.1552, -124.8944],
[ 118.1747,  103.6531, -176.6608],
[  52.4201,   43.5911,  -93.4998],
[ 119.7952,  104.9751, -179.9701],
[ 107.2421,   93.1843, -165.2720],
[  85.6899,   73.9901, -135.9498],
[  64.1684,   54.8604, -104.6687],
[  98.5827,   85.8114, -152.0604],
[  59.0427,   50.2849, -101.9172],
[  78.7315,   68.3857, -132.2328],
[  93.0150,   80.5978, -146.9386],
[ 127.4113,  112.6344, -189.2605],
[ 105.6555,   91.9985, -162.2424],
[ 138.7394,  122.4810, -202.2681],
[  96.2472,   83.9060, -149.5054],
[ 147.5589,  130.1671, -217.0375],
[ 114.6608,  101.3810, -172.2053],
[ 155.9160,  138.8121, -228.7602],
```

```
[  90.9680,   79.9117, -141.3024],
[ 100.1983,   86.9104, -157.3731],
[ 120.1080,  104.9124, -180.9119],
[  90.2664,   79.2975, -142.0128],
[  89.7442,   77.7951, -140.6923],
[  82.4216,   72.2761, -128.4592],
[ 212.8554,  189.8338, -298.0148],
[ 232.9581,  207.5229, -320.4785],
[ 234.0649,  207.1660, -327.7354],
[ 272.5594,  243.7599, -374.2411],
[ 280.6192,  251.2811, -380.0307],
[ 236.8509,  210.8084, -328.7479],
[ 304.5223,  272.9181, -413.3990],
[ 151.7913,  132.7794, -221.1477],
[ 185.8423,  164.1705, -262.9953],
[ 215.3004,  191.6040, -298.3022],
[ 273.5887,  244.7964, -375.7738],
[ 163.8829,  144.9046, -236.6877],
[ 142.2823,  124.3399, -209.3704],
[ 185.1090,  164.7387, -261.5698],
[ 191.2531,  169.5593, -269.6184],
[ 227.3347,  202.8032, -315.8458],
[ 200.9370,  178.1806, -286.0938],
[ 174.3933,  153.3947, -250.2581],
[ 229.1520,  204.2013, -317.4998],
[  79.0544,   67.6436, -126.2900],
[ 120.0541,  105.3166, -179.3371],
[ 134.0140,  116.3735, -198.5432],
[ 128.9349,  110.1508, -199.5436],
[ 174.4857,  152.8080, -262.4993],
[ 157.7685,  138.8968, -229.4365],
[  75.0042,   64.0274, -121.0330],
[  79.3792,   68.0524, -131.8797],
[  88.9312,   78.1005, -140.4451],
[  85.2245,   73.9404, -137.1126],
[  77.3579,   66.9686, -125.5704],
[  59.2512,   49.3782, -102.1811],
[  74.2114,   63.7085, -118.2482],
[ 116.1876,  102.1417, -172.9053],
[  52.5106,   43.8320,  -95.1767],
[  83.8212,   73.0418, -134.0304],
[  54.2748,   45.4098,  -96.5173],
[  58.0890,   47.9949, -105.6508],
[  79.3341,   67.8019, -136.6559],
[  60.7992,   51.2244, -110.6053],
[  64.0678,   55.2906, -108.1458],
[  64.0841,   54.5348, -108.2365],
[ 112.0443,   96.5121, -173.2164],
[ 115.2256,  101.4170, -170.1119],
[ 123.7370,  109.0708, -182.7369],
[  89.1024,   77.8316, -143.7396],
[ 153.7354,  135.7596, -223.7816],
[ 110.1152,   97.4299, -165.7818],
[  87.1523,   75.1451, -144.0363],
[  84.1195,   73.0923, -136.3203],
[ 120.1899,  106.4843, -178.0543],
```

```
        [ 113.4346,   99.4339, -171.7781],
        [ 128.2612,  114.0084, -190.4959],
        [ 115.7800,  103.0705, -175.0315],
        [ 108.9426,   96.9553, -163.7062],
        [ 108.1705,   95.7391, -163.6051],
        [ 132.5930,  117.2755, -198.8430],
        [ 110.7972,   97.0630, -172.7193],
        [ 149.8617,  132.3183, -220.5213],
        [ 150.6796,  132.7487, -221.6013],
        [  97.5183,   85.9160, -152.6548],
        [  88.7078,   77.3308, -137.5005]])
```

In [32]:
```
print(f"\n Output without softmax activation:{output_no_softmax}")
```

```
 Output without softmax activation:tensor([[ 261.2544, -245.4799,  140.127
4],
        [ 177.4916, -173.6684,   93.1411],
        [ 258.9286, -237.3772,  140.9150],
        [ 318.0180, -287.2196,  174.8949],
        [ 328.8492, -295.0155,  181.0653],
        [ 315.6892, -290.2225,  171.4373],
        [ 189.1180, -184.3628,   98.9298],
        [ 250.8167, -230.6913,  136.4422],
        [ 207.7969, -194.5613,  111.8484],
        [ 319.1204, -288.1627,  175.8285],
        [ 256.2358, -235.1284,  139.5353],
        [ 303.0523, -282.2583,  163.5945],
        [ 215.2545, -203.7039,  115.1919],
        [ 273.0750, -250.1138,  148.9745],
        [ 271.1752, -248.6146,  147.2590],
        [ 217.1216, -204.2336,  116.1072],
        [ 261.7100, -241.0446,  141.6871],
        [ 314.6559, -283.9193,  172.4401],
        [ 294.4597, -270.0367,  159.3409],
        [ 318.5303, -289.7024,  173.2775],
        [ 164.5207, -159.5948,   87.5012],
        [ 101.1298, -102.1456,   50.5883],
        [ 123.2474, -118.6156,   63.9233],
        [ 212.2067, -200.2009,  114.8603],
        [ 117.7069, -122.7173,   60.6679],
        [ 154.1270, -145.9114,   82.3905],
        [ 125.5578, -124.3897,   64.8274],
        [ 107.2663, -110.3267,   54.5338],
        [ 166.1023, -157.4362,   89.1277],
        [ 153.7661, -143.9375,   82.8272],
        [ 224.8987, -222.1607,  117.5840],
        [ 147.1709, -151.9007,   75.8667],
        [  97.3474,  -99.7758,   49.3058],
        [ 173.0132, -165.6411,   92.3199],
        [  99.1068, -101.0149,   50.9080],
        [ 134.8527, -135.1538,   69.6191],
        [ 103.7414, -105.0900,   52.5727],
        [  91.7984,  -94.3806,   46.2783],
        [  97.3340,  -99.2728,   49.1241],
        [ 151.6787, -147.2943,   79.6281],
        [  90.1214,  -94.1884,   46.6091],
        [  80.5676,  -85.5690,   39.7702],
        [ 140.8050, -135.8371,   75.7196],
        [ 133.6163, -133.0214,   70.1574],
        [ 143.0042, -142.9196,   75.4294],
        [ 139.8854, -136.5855,   74.8948],
        [ 141.1158, -140.2467,   74.2101],
        [ 123.6534, -124.9290,   65.0851],
        [  98.3430, -100.4490,   52.6791],
        [ 129.6704, -132.7840,   66.7466],
        [  98.1243, -106.2227,   47.9095],
        [ 114.5825, -115.0528,   59.7603],
        [ 163.4756, -160.3657,   86.9966],
        [ 166.6462, -163.4872,   87.4942],
        [ 166.2811, -158.5079,   88.9396],
```

```
[ 160.1442, -153.5442,   85.1878],
[ 179.3389, -171.7494,   97.1784],
[ 181.4273, -174.3769,   97.2293],
[ 259.6352, -239.0003,  141.4188],
[ 367.6327, -332.1664,  200.9447],
[ 359.9551, -325.6713,  196.9930],
[ 258.7263, -237.2569,  139.9534],
[ 286.5917, -259.1606,  156.5354],
[ 278.1611, -257.4204,  149.9733],
[ 187.7834, -179.4000,  100.5037],
[ 200.8714, -194.6490,  105.8571],
[ 224.6163, -210.8888,  120.4220],
[ 318.7778, -288.7383,  173.8683],
[ 243.8253, -226.4805,  131.9825],
[ 269.5133, -249.5806,  146.3041],
[ 251.8274, -232.0644,  137.0258],
[ 183.5465, -180.7994,   96.6520],
[ 164.8935, -160.2556,   87.6227],
[ 265.4688, -246.8569,  144.5008],
[ 243.3293, -224.8611,  131.2088],
[ 286.0038, -258.8629,  155.8246],
[ 340.2413, -310.0373,  185.7894],
[ 260.2996, -243.0393,  140.2961],
[ 125.5601, -123.8867,   66.8622],
[ 153.0337, -148.0969,   81.2802],
[  82.0795,  -90.5613,   39.9619],
[ 124.1888, -120.2980,   65.1410],
[  98.4724, -100.0613,   49.5794],
[ 113.3417, -113.2696,   58.9231],
[  93.8083,  -96.0214,   47.5291],
[ 179.9201, -178.8259,   93.7183],
[  64.8908,  -71.8356,   30.8039],
[ 175.3216, -165.0657,   93.9120],
[ 135.7474, -132.6818,   71.2431],
[ 115.9408, -114.3516,   60.9731],
[ 119.7616, -117.2374,   63.1452],
[  67.6848,  -74.9887,   32.9123],
[ 136.2094, -132.7585,   72.3299],
[ 103.9060, -107.6898,   52.8632],
[ 164.5900, -155.9964,   88.1761],
[  72.9897,  -79.7377,   35.2672],
[ 166.1590, -158.6963,   87.6188],
[ 145.6864, -144.3935,   76.3337],
[ 119.4596, -118.1822,   62.5234],
[  88.3525,  -91.3496,   46.2189],
[ 136.6760, -133.4184,   73.0080],
[  83.6553,  -87.3979,   41.1322],
[ 110.9498, -112.4120,   56.6038],
[ 125.6459, -127.6768,   65.5814],
[ 174.5504, -166.7119,   95.1227],
[ 144.2031, -141.8737,   76.6869],
[ 190.5867, -179.3861,  103.5800],
[ 131.8987, -131.0510,   70.4927],
[ 201.8145, -191.7178,  109.6604],
[ 157.2199, -151.5806,   83.6656],
[ 214.5180, -201.2354,  115.7965],
```

```
        [ 124.4863, -123.3980,   65.2690],
        [ 135.5478, -136.6327,   70.9508],
        [ 162.2115, -159.1131,   86.6287],
        [ 124.3929, -123.6606,   66.1033],
        [ 121.1829, -122.4310,   63.6279],
        [ 112.0520, -112.1501,   58.0479],
        [ 293.6081, -267.5941,  159.8060],
        [ 320.8979, -289.7262,  176.4538],
        [ 319.3297, -293.9443,  174.1693],
        [ 375.7021, -337.6774,  206.5787],
        [ 386.0020, -345.2137,  212.1482],
        [ 323.9563, -295.6815,  176.4628],
        [ 418.5584, -374.1424,  230.0150],
        [ 205.9368, -197.2069,  109.8518],
        [ 255.7790, -235.9395,  138.9967],
        [ 296.8044, -268.9040,  162.6077],
        [ 376.5469, -338.5610,  206.7117],
        [ 226.4258, -210.8340,  121.8060],
        [ 193.6814, -186.2656,  101.7863],
        [ 255.9273, -234.4585,  140.3728],
        [ 263.2762, -242.5852,  143.3402],
        [ 312.0739, -284.2850,  169.4163],
        [ 275.5997, -255.3719,  148.6046],
        [ 237.8463, -223.8301,  127.1515],
        [ 314.8726, -285.9243,  171.9890],
        [ 106.3893, -110.0079,   54.0412],
        [ 165.0800, -158.6220,   86.8019],
        [ 181.7182, -175.7395,   96.7633],
        [ 170.2167, -174.0194,   87.6040],
        [ 238.7306, -229.4574,  126.1649],
        [ 217.3938, -203.5164,  116.6012],
        [ 101.1561, -105.1829,   51.5373],
        [ 109.9250, -113.6087,   56.1291],
        [ 123.9091, -122.2897,   65.4940],
        [ 118.7402, -118.5567,   61.6980],
        [ 107.6836, -108.3479,   55.6082],
        [  80.5072,  -87.7295,   39.0648],
        [ 103.1632, -103.8276,   52.8152],
        [ 161.9782, -153.5199,   85.0958],
        [  73.3046,  -80.3619,   36.1459],
        [ 117.6046, -116.2317,   61.5862],
        [  76.1839,  -82.2378,   37.8354],
        [  79.4193,  -88.8763,   38.2537],
        [ 109.0956, -115.6094,   52.1537],
        [  84.3888,  -92.7257,   41.9657],
        [  91.0431,  -93.4219,   46.2685],
        [  90.0799,  -93.1314,   45.4539],
        [ 149.7417, -151.2269,   78.0396],
        [ 158.3690, -150.7532,   84.6507],
        [ 168.4999, -161.5394,   90.1129],
        [ 121.9123, -123.7724,   64.3051],
        [ 207.0673, -198.0967,  111.3182],
        [ 151.0318, -145.7818,   81.1163],
        [ 116.1917, -123.2595,   58.2875],
        [ 112.2027, -117.3510,   56.1187],
        [ 163.2281, -157.5600,   86.3269],
```

```
        [ 153.9026, -150.5466,   80.8060],
        [ 175.7242, -168.0080,   94.7975],
        [ 160.0565, -153.8527,   82.8350],
        [ 150.5745, -144.2698,   78.3439],
        [ 148.5076, -143.7331,   78.7298],
        [ 181.3493, -174.4592,   96.3088],
        [ 149.8724, -150.0229,   78.4784],
        [ 201.2024, -194.6788,  107.2868],
        [ 202.8279, -195.5919,  107.7739],
        [ 133.8861, -132.9262,   70.1051],
        [ 123.9493, -120.6476,   65.2891]])
```

3b

In [18]:
```python
def train_and_val(model, train_X, train_y, epochs, draw_curve = True):
    """
    Parameters
    --------------
    model: a PyTorch model
    train_X: np.array shape(ndata,nfeatures)
    train_y: np.array shape(ndata)
    epochs: int
    draw_curve: bool
    """
    ### Define your loss function, optimizer. Convert data to torch tensor #
    optimizer = torch.optim.Adam(model.parameters(), lr = 0.005)
    loss_func = nn.CrossEntropyLoss()
    train_X = torch.tensor(train_X, dtype=torch.float)
    train_y = torch.tensor(train_y, dtype=torch.long)

    ### Split training examples further into training and validation ###
    X_train, X_val, y_train, y_val = train_test_split(train_X, train_y)

    val_array=[]
    lowest_val_loss = np.inf
    model_param = model.state_dict()

    for i in range(epochs):
        ### Compute the loss and do backpropagation ###
        optimizer.zero_grad()
        y_pred = model(X_train)
        loss = loss_func(y_pred, y_train-1)
        loss.backward()
        optimizer.step()

        ### compute validation loss and keep track of the lowest val loss ##
        with torch.no_grad():
            val_pred = model(X_val)
            val_loss = loss_func(val_pred, y_val-1).detach().numpy()
            val_array.append(val_loss)

            if val_loss < lowest_val_loss:
                lowest_val_loss = val_loss
                model_param = model.state_dict()
```

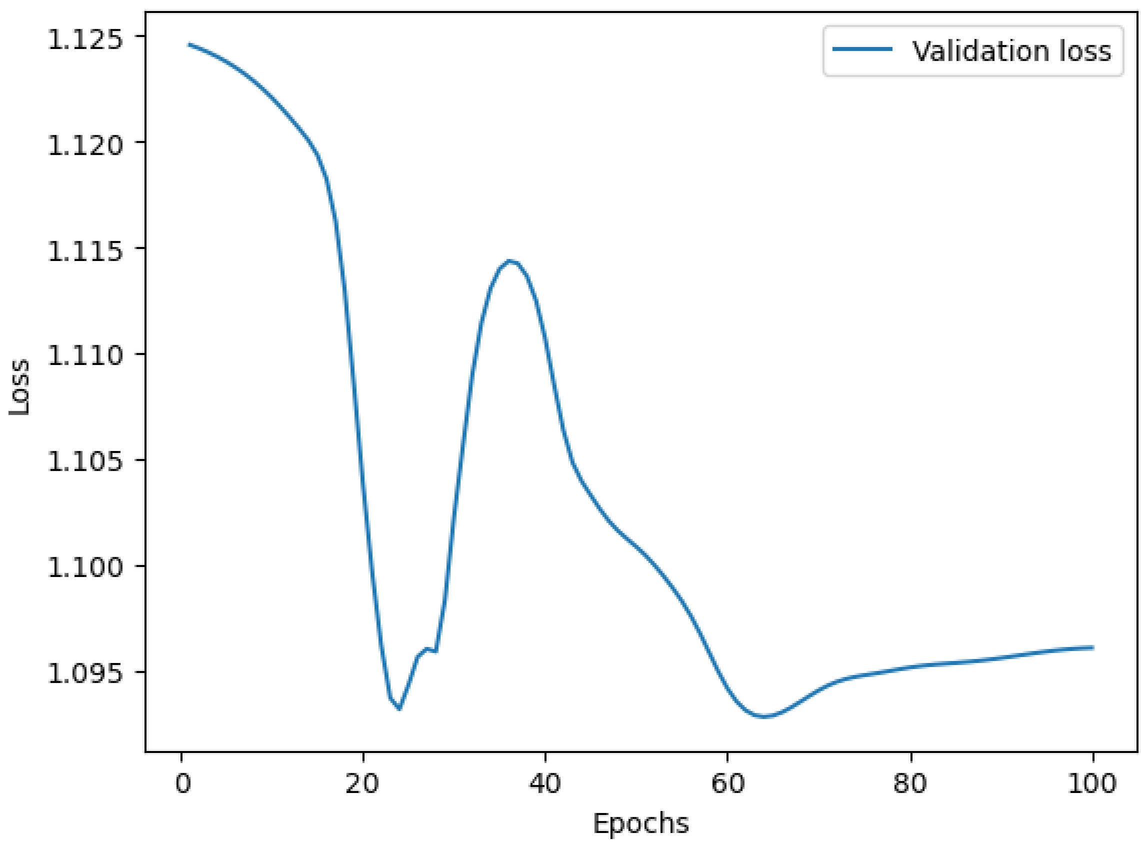
```
    # The final number of epochs is when the minimum error in validation se
    final_epochs=np.argmin(val_array)+1
    print("Number of epochs with lowest validation:",final_epochs)
    ### Recover the model weight ###
    model.load_state_dict(model_param)

    if draw_curve:
        plt.figure()
        plt.plot(np.arange(len(val_array))+1,val_array,label='Validation los
        plt.xlabel('Epochs')
        plt.ylabel('Loss')
        plt.legend()
```

In [19]:
```
train_and_val(model_w_softmax, wines_train_X, wines_train_y, 100)
```

```
Number of epochs with lowest validation: 64
```

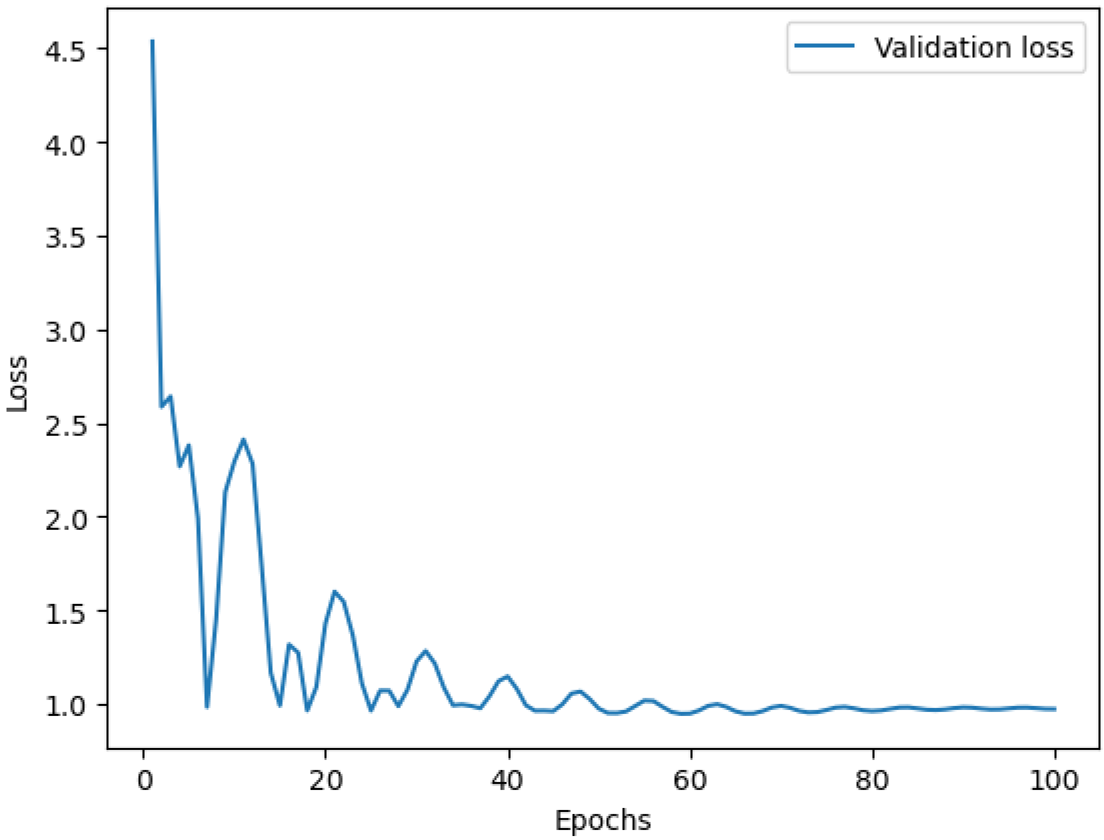
```
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2244072975.p
y:14: UserWarning: To copy construct from a tensor, it is recommended to use
sourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad
_(True), rather than torch.tensor(sourceTensor).
  train_X = torch.tensor(train_X, dtype=torch.float)
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2244072975.p
y:15: UserWarning: To copy construct from a tensor, it is recommended to use
sourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad
_(True), rather than torch.tensor(sourceTensor).
  train_y = torch.tensor(train_y, dtype=torch.long)
```



The loss starts to increase again at ~25, but the lowest validation is at 64

In [36]:
```
train_and_val(model_wo_softmax, wines_train_X, wines_train_y, 100)
```

```
Number of epochs with lowest validation: 59
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2244072975.p
y:14: UserWarning: To copy construct from a tensor, it is recommended to use
sourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad
_(True), rather than torch.tensor(sourceTensor).
  train_X = torch.tensor(train_X, dtype=torch.float)
/var/folders/m8/skfw9g2x4_g4pq5cv80_g24w0000gn/T/ipykernel_8594/2244072975.p
y:15: UserWarning: To copy construct from a tensor, it is recommended to use
sourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad
_(True), rather than torch.tensor(sourceTensor).
  train_y = torch.tensor(train_y, dtype=torch.long)
```



In [ ]:

```python
### CALCULATE ACCURACY OF EACH
# calculate_accuracy(model,xs,ys)
def calculate_accuracy_nn(model,xs,ys):
    y_pred=np.zeros_like(ys)
    for idx,x in enumerate(xs):
        y_pred[idx]=model.forward(x)
    return np.sum(ys==y_pred)/len(ys)

calculate_accuracy_nn(model_w_softmax, wines_train_X, wines_train_y)
calculate_accuracy_nn(model_wo_softmax, wines_train_X, wines_train_y)
```