علی آرمان ۴۰۲۳۶۲۴۰۰۲

در پیاده سازی ابتدا یک بار کد ها نوشته شد و در نهایت مرتب شد و به صورت بهینه کنار هم قرار گرفت.

در ابتدا نگارش کد ها گزارش شده و بعد از آن اجرای کد و در هر قسمت با رنگ قرمز شماره گزارش مشخص شده.

```
import torch ···

class KNNModel(nn.Module): ···

class MLPModel(nn.Module): ···

#step 1: opening file
def read_dataset(file_path): ···

# Step 2: Encode categorical data using LabelEncoder
def encode_categorical_data(data): ···

# Step 3: Handle missing values by replacing with the mean
def handle_missing_values(data): ···

# Step 4: Plot the Success Rate
def plot_success_rate(success_rate_train, success_rate_val, hidden_layers, hidden_neurons, activation_function): ···

# Step 5: Train and Evaluate the KNN Model
def train_and_evaluate_knn_model(k_values, distance_metrics): ···

# Step 6: Train the MLP Model
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=20): ···

# Step 7: Train and Evaluate the MLP Model
def train_and_evaluate_mlp_model(hidden_layers_list, hidden_neurons_list, activation_functions): ···


file_path = 'Telecust1.csv'
data = read_dataset(file_path)
data_encoded = encode_categorical_data(data)
data_filled = handle_missing_values(data_encoded)

# split atribut and labels
X = data_filled.iloc[:, 1:-1]
y = data_filled['custcat']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=17, shuffle=False)

k_values = [1, 3, 5, 7]
distance_metrics = ['euclidean', 'manhattan']
train_and_evaluate_knn_model(k_values, distance_metrics)

# Prepare data for MLP
```

در کلاس KNNModel ما طبق اموزش tourch تابع سازنده، تابع فیت و تابع پریدیکت داریم:

```python
class KNNModel(nn.Module):
    def __init__(self, k=1, distance_metric='euclidean'):
        super(KNNModel, self).__init__()
        self.k = k
        self.distance_metric = distance_metric
        self.X_train = None
        self.y_train = None

    def fit(self, X_train, y_train):
        self.X_train = X_train
        self.y_train = y_train

    def predict(self, X_test):
        predictions = []
        for x in X_test:
            distances = self.calculate_distances(x)
            _, indices = torch.topk(distances, self.k, largest=False)
            k_nearest_labels = self.y_train[indices]
            unique_labels, counts = torch.unique(k_nearest_labels, return_counts=True)
            predicted_label = unique_labels[torch.argmax(counts)]
            predictions.append(predicted_label.item())
        return torch.tensor(predictions)

    def calculate_distances(self, x):
        if self.distance_metric == 'euclidean':
            distances = torch.norm(self.X_train - x, dim=1, p=2)
        elif self.distance_metric == 'manhattan':
            distances = torch.norm(self.X_train - x, dim=1, p=1)
        return distances
```

تابع calculate_distances برای مشخص کردن نوع فاصله هست

در کلاس MLPModel مشابه کلاس قبلی طبق آموزش پیاده سازی شده :

```python
class MLPModel(nn.Module):
    def __init__(self, input_size, output_size, hidden_layers, hidden_neurons, activation_function):
        super(MLPModel, self).__init__()
        layers = []
        layers.append(nn.Linear(input_size, hidden_neurons))
        layers.append(activation_function())

        for _ in range(hidden_layers - 1):
            layers.append(nn.Linear(hidden_neurons, hidden_neurons))
            layers.append(activation_function())

        layers.append(nn.Linear(hidden_neurons, output_size))
        self.model = nn.Sequential(*layers)

    def forward(self, x):
        return self.model(x)
```

در تابع سازنده، سایز ورودی و خروجی، تعداد لایه ها و نورون ها و همچنین تابع فعال ساز را مشخص کردیم.

در تابع read_dataset فایل را باز میکنیم:

```python
#step 1: opening file
def read_dataset(file_path):
    return pd.read_csv(file_path)
```

در تابع encode_categorical_data ما داده را انکود میکنیم تا قابل پردازش کنیم:

```python
# Step 2: Encode categorical data using LabelEncoder
def encode_categorical_data(data):
    label_encoder = LabelEncoder()
    for column in data.select_dtypes(include=['object']).columns:
        data[column] = label_encoder.fit_transform(data[column])
    return data
```

در تابع handle_missing_values مقادیری که null هستند را با میانگین پر میکنیم:

```python
# Step 3: Handle missing values by replacing with the mean
def handle_missing_values(data):
    data_filled = data.fillna(data.mean())
    return data_filled
```

در تابع plot_success_rate ما خروجی شبکه عصبی را با نمودار نشان میدهیم. که تعداد لایه و نورون و تابع فعال ساز در تایتل آن مشخص است.

```python
# Step 4: Plot the Success Rate
def plot_success_rate(success_rate_train, success_rate_val, hidden_layers, hidden_neurons, activation_function):
    plt.plot(success_rate_train, label=f'Training Success Rate')
    plt.plot(success_rate_val, label=f'Validation Success Rate')
    plt.xlabel('Epochs')
    plt.ylabel('Success Rate')
    plt.title(f'Hidden Layers: {hidden_layers} hidden neurons: {hidden_neurons}, activation function: {activation_function}')
    plt.legend()
    plt.show()
```

در تابع train_and_evaluate_knn_model مدل knn خود را با ترکیب تمام حالات نوع فاصله و تعداد k آموزش دادیم و در نهایت با پلات خروجی را برای دو نوع فاصله نمایش دادیم.

```python
# Step 5: Train and Evaluate the KNN Model
def train_and_evaluate_knn_model(k_values, distance_metrics):
    for distance_metric in distance_metrics:
        train_accuracy = []
        test_accuracy = []

        for k in k_values:
            knn_model = KNNModel(k=k, distance_metric=distance_metric)
            knn_model.fit(torch.Tensor(X_train.values), torch.LongTensor(y_train.values))

            y_pred_train = knn_model.predict(torch.Tensor(X_train.values))
            y_pred_test = knn_model.predict(torch.Tensor(X_test.values))

            train_accuracy.append(accuracy_score(y_train, y_pred_train))
            test_accuracy.append(accuracy_score(y_test, y_pred_test))

            print(f"\nKNN with k={k} and distance metric={distance_metric}:")
            print("Training Classification Report:\n", classification_report(y_train, y_pred_train))
            print("Testing Classification Report:\n", classification_report(y_test, y_pred_test))

        # Plot accuracy vs. k for both training and testing sets
        plt.plot(k_values, train_accuracy, label=f'Training Accuracy (Distance Metric: {distance_metric})')
        plt.plot(k_values, test_accuracy, label=f'Testing Accuracy (Distance Metric: {distance_metric})')

        plt.xlabel('k')
        plt.ylabel('Accuracy')
        plt.title(f'KNN Model Accuracy for different K Values and {distance_metric}')
        plt.legend()
        plt.show()
```

در تابع train_mlp_model طبق آموزش های موجود به تعداد دور اموزش(epochs) مدل را آموزش میدهیم، در هر بار مقدار خطا را حساب میکنیم و به مقدار های قبلی لیست اضافه میکنیم. و در نهایت خروجی را پرینت میکنیم:

```python
# Step 6: Train the MLP Model
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=20):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            _, predicted = torch.max(outputs.data, 1)
            total_train += labels.size(0)
            correct_train += (predicted == labels).sum().item()

        success_rate_train.append(correct_train / total_train)

        model.eval()
        total_val = 0
        correct_val = 0

        with torch.no_grad():
            for inputs, labels in val_loader:
                outputs = model(inputs)
                _, predicted = torch.max(outputs.data, 1)
                total_val += labels.size(0)
                correct_val += (predicted == labels).sum().item()

            success_rate_val.append(correct_val / total_val)

        print(f'Epoch {epoch + 1}/{epochs}, Training Success Rate: {success_rate_train[-1]}, Validation Success Rate: {success_rate_val[-1]}')

    return success_rate_train, success_rate_val
```

در تابع train_and_evaluate_mlp_model ما شبکه را با تعداد لایه و تعداد نورون و توابع فعال ساز مختلف در تمام حالت ها آموزش میدهیم و در هر بار بررسی میکنیم اگر جواب به دست آمده بهتر باشد با جواب بهتر قبلی جایگزین میکنیم.

همچنین از plot_success_rate برای نمایش نتیجه این آموزش استفاده شده و در نهایت بهترین حالت در خروجی چاپ شده.

دلیل اینکه تمام حالت های ممکن برای تعداد لایه و تعداد نورون و توابع فعال ساز مختلف باهم تست شده این بوده که ممکن است از بین تعداد لایه ۵ بهترین حالت باشد(عدد ۵ فرضی است) ولی با تغییر تعداد نورون با ۲ لایه جواب بهتری برسیم، این ترکیب برای تابع فعال ساز هم می باشد.

```python
# Step 7: Train and Evaluate the MLP Model
def train_and_evaluate_mlp_model(hidden_layers_list, hidden_neurons_list, activation_functions):
    best_model = None
    best_val_accuracy = 0
    best_hidden_layers = 0
    best_hidden_neurons = 0
    best_activation_function = None

    for hidden_layers in hidden_layers_list:
        for hidden_neurons in hidden_neurons_list:
            for activation_function in activation_functions:
                # Build and train the MLP model
                mlp_model = MLPModel(input_size=len(X_train.columns), output_size=len(y_train.unique()), hidden_layers=hidden_layers, hidden_neurons=hidden_neurons, activation_function=activation
                criterion = nn.CrossEntropyLoss()
                optimizer = optim.Adam(mlp_model.parameters(), lr=0.001)

                success_rate_train, success_rate_val = train_mlp_model(mlp_model, criterion, optimizer, train_loader, val_loader)

                # plot_success_rate(success_rate_train, success_rate_val, hidden_layers, hidden_neurons, activation_function.__name__)

                # Evaluate the MLP Model on validation set
                mlp_model.eval()
                with torch.no_grad():
                    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor), axis=1)
                    val_accuracy = torch.sum(y_pred_val_mlp == y_val_tensor) / len(y_val_tensor)

                    print(f"\nMLP Model Classification Report - Validation Set (Hidden Layers: {hidden_layers}, Hidden Neurons: {hidden_neurons}, Activation Function: {activation_function.__name_

                if val_accuracy > best_val_accuracy:
                    best_val_accuracy = val_accuracy
                    best_model = mlp_model
                    best_hidden_layers = hidden_layers
                    best_hidden_neurons = hidden_neurons
                    best_activation_function = activation_function

    print(f"\nBest MLP Model was trained with {best_hidden_layers} hidden layers, {best_hidden_neurons} hidden neurons, and {best_activation_function.__name__} activation function.")

    return best_model
```

قسمت های اجرایی کد:

ابتدا نام فایل مشخص شده و فایل خوانده شده و انکود کردن و هندل کردن داده های null صورت گرفته.

سپس ویژگی ها و دسته مشتریان از هم جدا شده اند و با train_test_split داده های اموزشی و تست تخصیص داده شده اند.

```python
file_path = 'Telecust1.csv'
data = read_dataset(file_path)
data_encoded = encode_categorical_data(data)
data_filled = handle_missing_values(data_encoded)

# split atribut and labels
X = data_filled.iloc[:, 1:-1]
y = data_filled['custcat']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=17, shuffle=False)
```

در ادامه تابع train_and_evaluate_knn_model با مقدار k های مختلف ۱، ۳، ۵ و ۷ و همچنین فاصله های Euclidean و Manhattan فراخوانی شده.

```python
k_values = [1, 3, 5, 7]
distance_metrics = ['euclidean', 'manhattan']
train_and_evaluate_knn_model(k_values, distance_metrics)
```

در ادامه داده ها با استفاده از tourch برای استفاده در شبکه عصبی آماده شده اند تابع train_and_evaluate_mlp_model با تعداد لایه ۲ تا ۶ و تعداد نورون ۳۲ تا ۵۱۲ و ۴ تابع فعال ساز مختلف فراخونی شد و در نهایت گزارش بهترین کلاس چاپ شده:

```
# Prepare data for MLP
X_train_tensor = torch.Tensor(X_train.values)
y_train_tensor = torch.LongTensor(y_train.values)
X_val_tensor = torch.Tensor(X_test.values)
y_val_tensor = torch.LongTensor(y_test.values)

train_dataset = TensorDataset(X_train_tensor, y_train_tensor)
val_dataset = TensorDataset(X_val_tensor, y_val_tensor)

train_loader = DataLoader(train_dataset, batch_size=32, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=32, shuffle=False)

# Hidden layers, neurons, and activation functions to try
hidden_layers_to_try = [2, 3, 4, 5, 6]
hidden_neurons_to_try = [32, 64, 128, 256, 512]
activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU, nn.Sigmoid]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_try, hidden_neurons_to_try, activation_functions_to_try)

# Evaluate the best MLP model on the testing set
best_mlp_model.eval()
with torch.no_grad():
    y_pred_test_mlp = torch.argmax(best_mlp_model(X_val_tensor), axis=1)
    print("\nBest MLP Model Classification Report - Testing Set:\n", classification_report(y_val_tensor, y_pred_test_mlp))
```

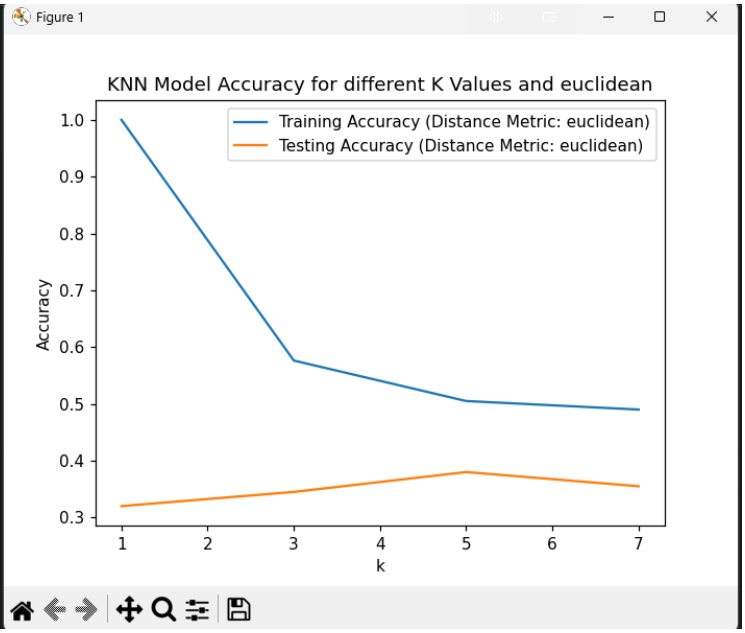
اجرای کد:

گزارش ۱

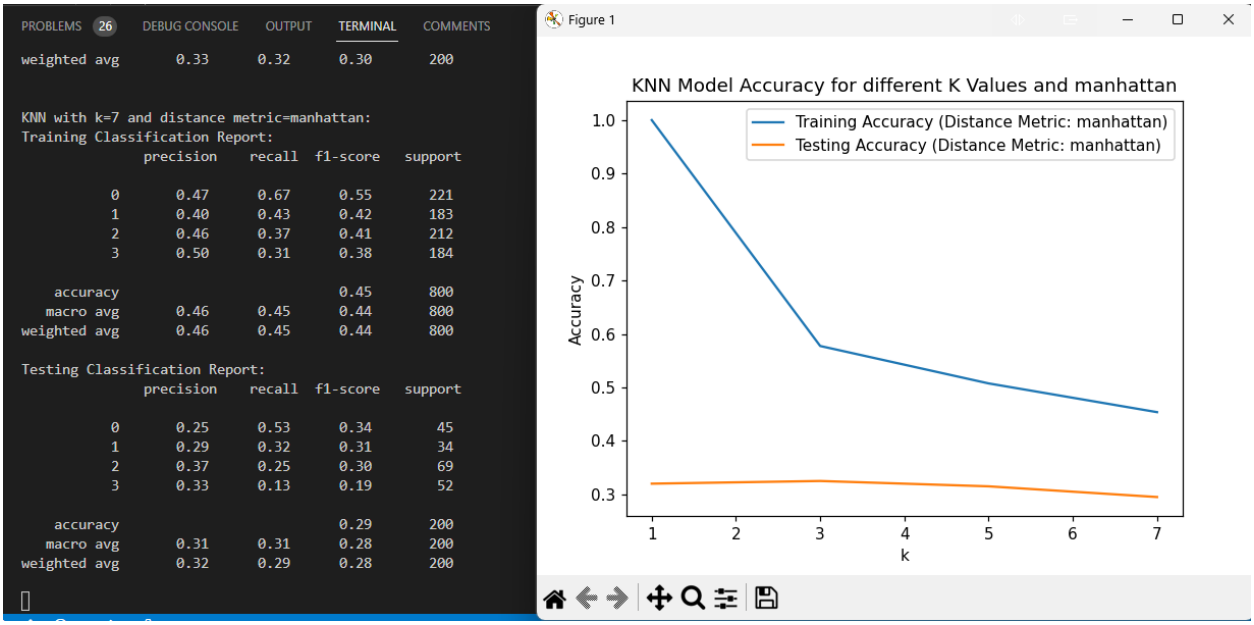
نمایش نمودار و classification_report مربوط به اجرا knn با k های مختلف و Euclidean:

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
KNN with k=1 and distance metric=euclidean:
Training Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       221
           1       1.00      1.00      1.00       183
           2       1.00      1.00      1.00       212
           3       1.00      1.00      1.00       184

    accuracy                           1.00       800
   macro avg       1.00      1.00      1.00       800
weighted avg       1.00      1.00      1.00       800

Testing Classification Report:
              precision    recall  f1-score   support

           0       0.29      0.40      0.34        45
           1       0.23      0.29      0.26        34
           2       0.44      0.35      0.39        69
           3       0.29      0.23      0.26        52

    accuracy                           0.32       200
   macro avg       0.32      0.32      0.31       200
weighted avg       0.33      0.32      0.32       200


KNN with k=3 and distance metric=euclidean:
Training Classification Report:
```



نمایش نمودار و classification_report مربوط به اجرا knn با k های مختلف و manhattan:



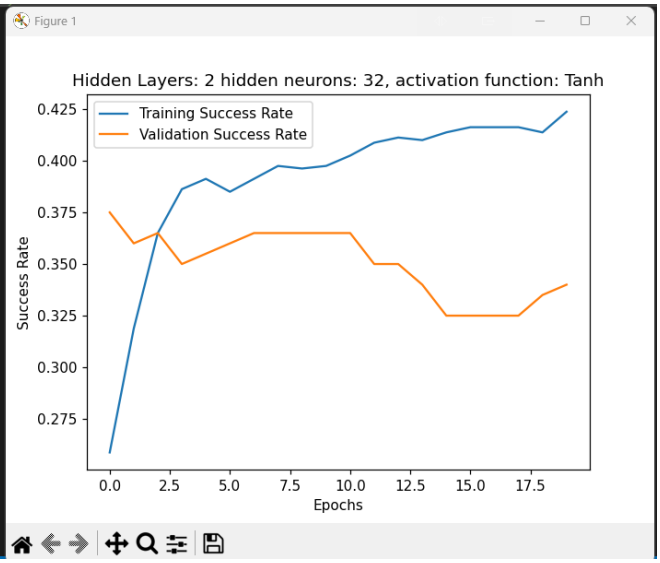
همان طور که پیداست فاصله منهتن در شروع کار در تست بهتر بوده ولی فاصله اقلیدسی در نهایت در تست بهتر بوده. و هر دو در ترین تقریبا مشابه هم دیگر بوده اند. و همان طور که از نتایج پیداست دقت فاصله اقلیدسی بهتر بوده.

گزارش ۲ و ۳ و ۴ و ۵

نمایش نمودار و classification_report مربوط به اجرا mlp با تعداد لایه و تعداد نورون و توابع فعال ساز مختلف انجام شده(تعداد کل حالت ها زیاد می باشد اسکرین اجرا چند حالت در ادامه اورده شده):

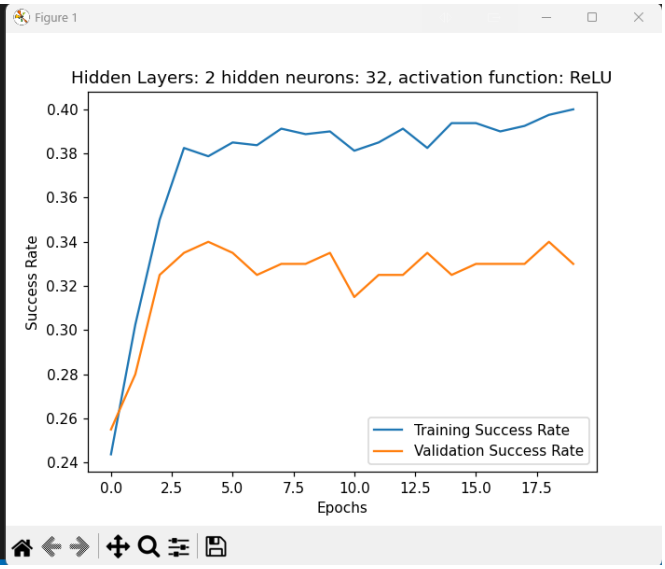
PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
           1       0.28      0.65      0.39        34
           2       0.62      0.23      0.34        69
           3       0.25      0.04      0.07        52

    accuracy                           0.34       200
   macro avg       0.36      0.38      0.30       200
weighted avg       0.39      0.34      0.29       200

Epoch 1/20, Training Success Rate: 0.25875, Validation Success Rate: 0.375
Epoch 2/20, Training Success Rate: 0.31875, Validation Success Rate: 0.36
Epoch 3/20, Training Success Rate: 0.365, Validation Success Rate: 0.365
Epoch 4/20, Training Success Rate: 0.38625, Validation Success Rate: 0.35
Epoch 5/20, Training Success Rate: 0.39125, Validation Success Rate: 0.355
Epoch 6/20, Training Success Rate: 0.385, Validation Success Rate: 0.36
Epoch 7/20, Training Success Rate: 0.39125, Validation Success Rate: 0.365
Epoch 8/20, Training Success Rate: 0.3975, Validation Success Rate: 0.365
Epoch 9/20, Training Success Rate: 0.39625, Validation Success Rate: 0.365
Epoch 10/20, Training Success Rate: 0.3975, Validation Success Rate: 0.365
Epoch 11/20, Training Success Rate: 0.4025, Validation Success Rate: 0.365
Epoch 12/20, Training Success Rate: 0.40875, Validation Success Rate: 0.35
Epoch 13/20, Training Success Rate: 0.41125, Validation Success Rate: 0.35
Epoch 14/20, Training Success Rate: 0.41, Validation Success Rate: 0.34
Epoch 15/20, Training Success Rate: 0.41375, Validation Success Rate: 0.325
Epoch 16/20, Training Success Rate: 0.41625, Validation Success Rate: 0.325
Epoch 17/20, Training Success Rate: 0.41625, Validation Success Rate: 0.325
Epoch 18/20, Training Success Rate: 0.41625, Validation Success Rate: 0.325
Epoch 19/20, Training Success Rate: 0.41375, Validation Success Rate: 0.335
Epoch 20/20, Training Success Rate: 0.42375, Validation Success Rate: 0.34
```



PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

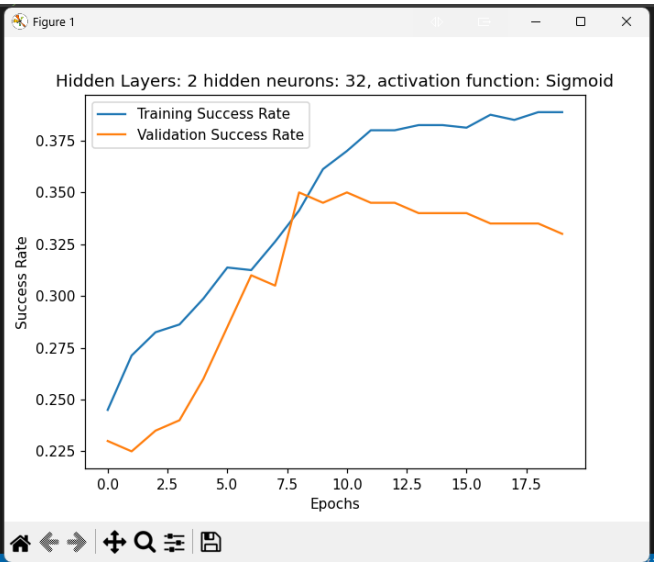
```
           1       0.28      0.24      0.25        34
           2       0.45      0.32      0.37        69
           3       0.33      0.27      0.29        52

    accuracy                           0.34       200
   macro avg       0.34      0.34      0.33       200
weighted avg       0.35      0.34      0.34       200

Epoch 1/20, Training Success Rate: 0.24375, Validation Success Rate: 0.255
Epoch 2/20, Training Success Rate: 0.3025, Validation Success Rate: 0.28
Epoch 3/20, Training Success Rate: 0.35, Validation Success Rate: 0.325
Epoch 4/20, Training Success Rate: 0.3825, Validation Success Rate: 0.335
Epoch 5/20, Training Success Rate: 0.37875, Validation Success Rate: 0.34
Epoch 6/20, Training Success Rate: 0.385, Validation Success Rate: 0.335
Epoch 7/20, Training Success Rate: 0.38375, Validation Success Rate: 0.325
Epoch 8/20, Training Success Rate: 0.39125, Validation Success Rate: 0.33
Epoch 9/20, Training Success Rate: 0.38875, Validation Success Rate: 0.33
Epoch 10/20, Training Success Rate: 0.39, Validation Success Rate: 0.335
Epoch 11/20, Training Success Rate: 0.38125, Validation Success Rate: 0.315
Epoch 12/20, Training Success Rate: 0.385, Validation Success Rate: 0.325
Epoch 13/20, Training Success Rate: 0.39125, Validation Success Rate: 0.325
Epoch 14/20, Training Success Rate: 0.3825, Validation Success Rate: 0.335
Epoch 15/20, Training Success Rate: 0.39375, Validation Success Rate: 0.325
Epoch 16/20, Training Success Rate: 0.39375, Validation Success Rate: 0.33
Epoch 17/20, Training Success Rate: 0.39, Validation Success Rate: 0.33
Epoch 18/20, Training Success Rate: 0.3925, Validation Success Rate: 0.33
Epoch 19/20, Training Success Rate: 0.3975, Validation Success Rate: 0.34
Epoch 20/20, Training Success Rate: 0.4, Validation Success Rate: 0.33
```



PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

1	0.32	0.35	0.33	34
2	0.59	0.14	0.23	69
3	0.30	0.29	0.29	52
accuracy			0.33	200
macro avg	0.38	0.36	0.32	200
weighted avg	0.40	0.33	0.31	200

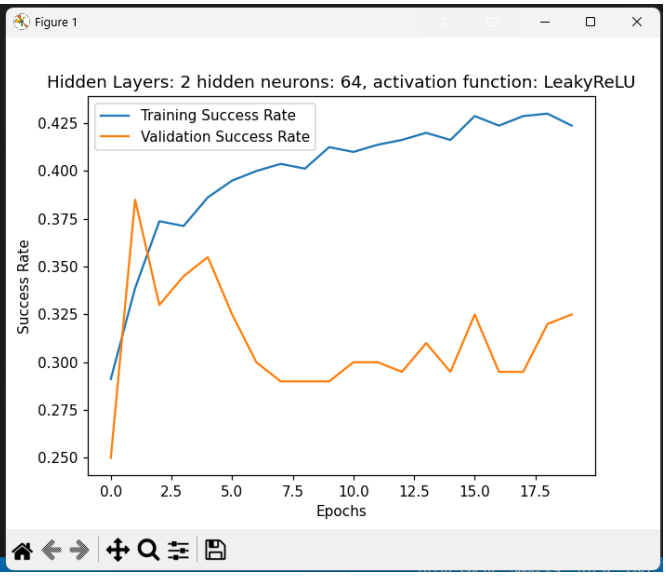
```
Epoch 1/20, Training Success Rate: 0.245, Validation Success Rate: 0.23
Epoch 2/20, Training Success Rate: 0.27125, Validation Success Rate: 0.225
Epoch 3/20, Training Success Rate: 0.2825, Validation Success Rate: 0.235
Epoch 4/20, Training Success Rate: 0.28625, Validation Success Rate: 0.24
Epoch 5/20, Training Success Rate: 0.29875, Validation Success Rate: 0.26
Epoch 6/20, Training Success Rate: 0.31375, Validation Success Rate: 0.285
Epoch 7/20, Training Success Rate: 0.3125, Validation Success Rate: 0.31
Epoch 8/20, Training Success Rate: 0.32625, Validation Success Rate: 0.305
Epoch 9/20, Training Success Rate: 0.34125, Validation Success Rate: 0.35
Epoch 10/20, Training Success Rate: 0.36125, Validation Success Rate: 0.345
Epoch 11/20, Training Success Rate: 0.37, Validation Success Rate: 0.35
Epoch 12/20, Training Success Rate: 0.38, Validation Success Rate: 0.345
Epoch 13/20, Training Success Rate: 0.38, Validation Success Rate: 0.345
Epoch 14/20, Training Success Rate: 0.3825, Validation Success Rate: 0.34
Epoch 15/20, Training Success Rate: 0.3825, Validation Success Rate: 0.34
Epoch 16/20, Training Success Rate: 0.38125, Validation Success Rate: 0.34
Epoch 17/20, Training Success Rate: 0.3875, Validation Success Rate: 0.335
Epoch 18/20, Training Success Rate: 0.385, Validation Success Rate: 0.335
Epoch 19/20, Training Success Rate: 0.38875, Validation Success Rate: 0.335
Epoch 20/20, Training Success Rate: 0.38875, Validation Success Rate: 0.33
```

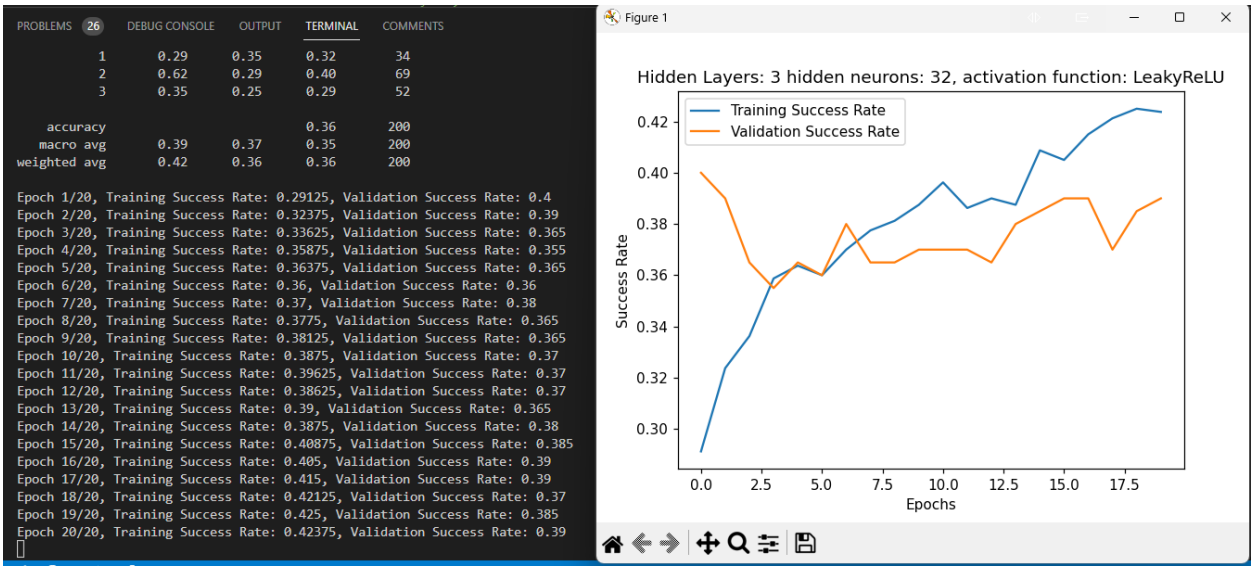


PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

1	0.29	0.41	0.34	34
2	0.54	0.22	0.31	69
3	0.29	0.19	0.23	52
accuracy			0.33	200
macro avg	0.35	0.36	0.32	200
weighted avg	0.38	0.33	0.32	200

```
Epoch 1/20, Training Success Rate: 0.29125, Validation Success Rate: 0.25
Epoch 2/20, Training Success Rate: 0.33875, Validation Success Rate: 0.385
Epoch 3/20, Training Success Rate: 0.37375, Validation Success Rate: 0.33
Epoch 4/20, Training Success Rate: 0.37125, Validation Success Rate: 0.345
Epoch 5/20, Training Success Rate: 0.38625, Validation Success Rate: 0.355
Epoch 6/20, Training Success Rate: 0.395, Validation Success Rate: 0.325
Epoch 7/20, Training Success Rate: 0.4, Validation Success Rate: 0.3
Epoch 8/20, Training Success Rate: 0.40375, Validation Success Rate: 0.29
Epoch 9/20, Training Success Rate: 0.40125, Validation Success Rate: 0.29
Epoch 10/20, Training Success Rate: 0.4125, Validation Success Rate: 0.29
Epoch 11/20, Training Success Rate: 0.41, Validation Success Rate: 0.3
Epoch 12/20, Training Success Rate: 0.41375, Validation Success Rate: 0.3
Epoch 13/20, Training Success Rate: 0.41625, Validation Success Rate: 0.295
Epoch 14/20, Training Success Rate: 0.42, Validation Success Rate: 0.31
Epoch 15/20, Training Success Rate: 0.41625, Validation Success Rate: 0.295
Epoch 16/20, Training Success Rate: 0.42875, Validation Success Rate: 0.325
Epoch 17/20, Training Success Rate: 0.42375, Validation Success Rate: 0.295
Epoch 18/20, Training Success Rate: 0.42875, Validation Success Rate: 0.295
Epoch 19/20, Training Success Rate: 0.43, Validation Success Rate: 0.32
Epoch 20/20, Training Success Rate: 0.42375, Validation Success Rate: 0.325
```





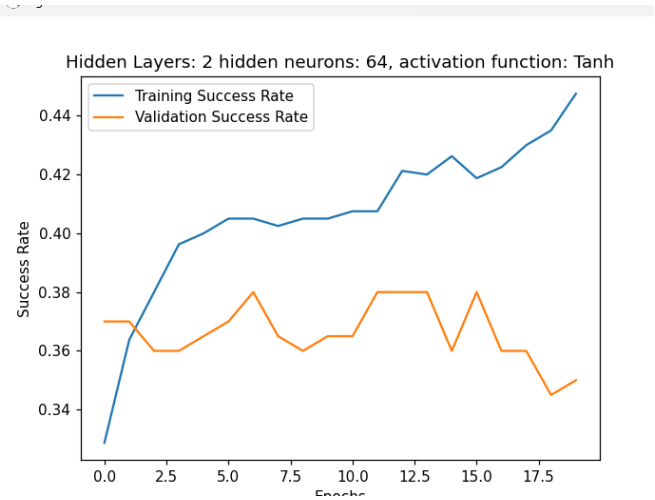
که بهترین جواب طبق اجرا کد به صورت زیر می باشد:

```
Best MLP Model was trained with 2 hidden layers, 64 hidden neurons, and Tanh activation function.

Best MLP Model Classification Report - Testing Set:
              precision    recall  f1-score   support

           0       0.31      0.58      0.40        45
           1       0.26      0.29      0.28        34
           2       0.49      0.32      0.39        69
           3       0.36      0.23      0.28        52

    accuracy                           0.35       200
   macro avg       0.36      0.36      0.34       200
weighted avg       0.38      0.35      0.34       200
```

برای نمایش نمودار حالت برگزیده فقط مقادیری که گزارش شده به عنوان ورودی تابع در نظر گرفته شده:

```
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU, nn.Sigmoid]
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
```

```
185                 best_model = mlp_model
186                 best_hidden_layers = hidden_layers

PROBLEMS 26  DEBUG CONSOLE  OUTPUT  TERMINAL  COMMENTS

(env_hw_2) PS D:\sku\machine learning\hw_2_ml> python code_3.py
Epoch 1/20, Training Success Rate: 0.32875, Validation Success Rate: 0.37
Epoch 2/20, Training Success Rate: 0.36375, Validation Success Rate: 0.37
Epoch 3/20, Training Success Rate: 0.38, Validation Success Rate: 0.36
Epoch 4/20, Training Success Rate: 0.39625, Validation Success Rate: 0.36
Epoch 5/20, Training Success Rate: 0.4, Validation Success Rate: 0.365
Epoch 6/20, Training Success Rate: 0.405, Validation Success Rate: 0.37
Epoch 7/20, Training Success Rate: 0.405, Validation Success Rate: 0.38
Epoch 8/20, Training Success Rate: 0.4025, Validation Success Rate: 0.365
Epoch 9/20, Training Success Rate: 0.405, Validation Success Rate: 0.36
Epoch 10/20, Training Success Rate: 0.405, Validation Success Rate: 0.365
Epoch 11/20, Training Success Rate: 0.4075, Validation Success Rate: 0.365
Epoch 12/20, Training Success Rate: 0.4075, Validation Success Rate: 0.38
Epoch 13/20, Training Success Rate: 0.42125, Validation Success Rate: 0.38
Epoch 14/20, Training Success Rate: 0.42, Validation Success Rate: 0.38
Epoch 15/20, Training Success Rate: 0.42625, Validation Success Rate: 0.36
Epoch 16/20, Training Success Rate: 0.41875, Validation Success Rate: 0.38
Epoch 17/20, Training Success Rate: 0.4225, Validation Success Rate: 0.36
Epoch 18/20, Training Success Rate: 0.43, Validation Success Rate: 0.36
Epoch 19/20, Training Success Rate: 0.435, Validation Success Rate: 0.345
Epoch 20/20, Training Success Rate: 0.4475, Validation Success Rate: 0.35
```



هنگامی که shuffle برابر با True باشد مقادیر مختلفی برای بهترین جواب ارائه میشود:

```
# Best MLP Model was trained with 5 hidden layers, 512 hidden neurons, and Tanh activation function.
# Best MLP Model Classification Report - Testing Set:
#               precision    recall  f1-score   support

#            0       0.47      0.58      0.52        53
#            1       0.35      0.39      0.37        41
#            2       0.51      0.50      0.50        64
#            3       0.32      0.19      0.24        42

#     accuracy                           0.43       200
#    macro avg       0.41      0.42      0.41       200
# weighted avg       0.43      0.43      0.42       200

# Best MLP Model was trained with 2 hidden layers, 256 hidden neurons, and LeakyReLU activation function.

# Best MLP Model Classification Report - Testing Set:
#               precision    recall  f1-score   support

#            0       0.40      0.68      0.50        53
#            1       0.80      0.10      0.17        41
#            2       0.48      0.55      0.51        64
#            3       0.31      0.24      0.27        42

#     accuracy                           0.42       200
#    macro avg       0.50      0.39      0.36       200
# weighted avg       0.49      0.42      0.39       200
```

با شافل برابر True جواب های بهتری هم به دست اومده!

گزارش ۶

برای قسمت تست بهینه ساز Adam شبکه را با بهترین حالت گزارش شده آموزش میدهیم و در هر بار نرخ بهینه گر Adam را به صورت دستی در کد عوض میکنیم:

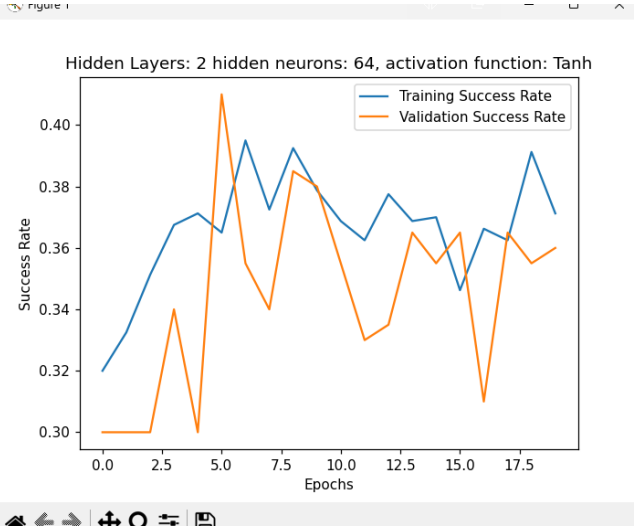
```
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(mlp_model.parameters(), lr=0.001)
```

اجرا با ۰.۰۱:

```
optimizer = optim.Adam(mlp_model.parameters(), lr=0.01)

success_rate_train, success_rate_val = train_mlp_model(m

plot_success_rate(success_rate_train, success_rate_val,

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor
    val_accuracy = torch.sum(y_pred_val_mlp == y_val_ten

    print(f"\nMLP Model Classification Report - Validati
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 10/20, Training Success Rate: 0.37875, Validation Success Rate: 0.38
Epoch 11/20, Training Success Rate: 0.36875, Validation Success Rate: 0.355
Epoch 12/20, Training Success Rate: 0.3625, Validation Success Rate: 0.33
Epoch 13/20, Training Success Rate: 0.3775, Validation Success Rate: 0.335
Epoch 14/20, Training Success Rate: 0.36875, Validation Success Rate: 0.365
Epoch 15/20, Training Success Rate: 0.37, Validation Success Rate: 0.355
Epoch 16/20, Training Success Rate: 0.34625, Validation Success Rate: 0.365
Epoch 17/20, Training Success Rate: 0.36625, Validation Success Rate: 0.31
Epoch 18/20, Training Success Rate: 0.3625, Validation Success Rate: 0.365
Epoch 19/20, Training Success Rate: 0.39125, Validation Success Rate: 0.355
Epoch 20/20, Training Success Rate: 0.37125, Validation Success Rate: 0.36
```



نتیجه گزارش:

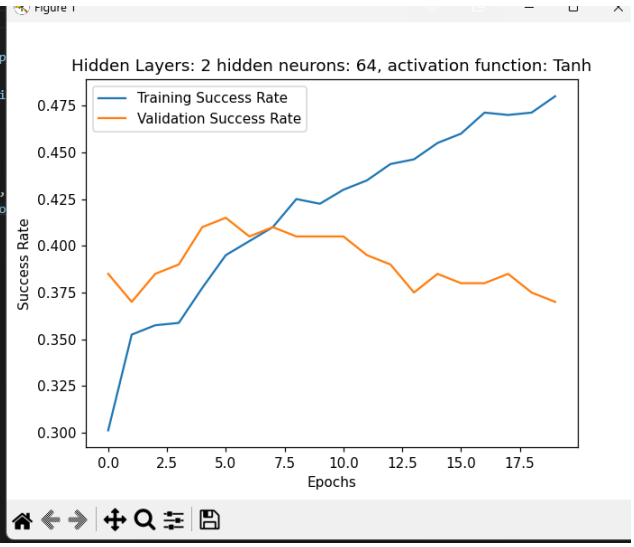
	precision	recall	f1-score	support
0	0.32	0.58	0.41	45
1	0.27	0.32	0.29	34
2	0.55	0.32	0.40	69
3	0.35	0.25	0.29	52
accuracy			0.36	200
macro avg	0.37	0.37	0.35	200
weighted avg	0.40	0.36	0.36	200

اجرا با ۰.۰۰۱:

```
optimizer = optim.Adam(mlp_model.parameters(), lr=0.001)

success_rate_train, success_rate_val = train_mlp_model(mlp

plot_success_rate(success_rate_train, success_rate_val, hi

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor),
    val_accuracy = torch.sum(y_pred_val_mlp == y_val_tenso
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.3775, Validation Success Rate: 0.41
Epoch 6/20, Training Success Rate: 0.395, Validation Success Rate: 0.415
Epoch 7/20, Training Success Rate: 0.4025, Validation Success Rate: 0.405
Epoch 8/20, Training Success Rate: 0.41, Validation Success Rate: 0.41
Epoch 9/20, Training Success Rate: 0.425, Validation Success Rate: 0.405
Epoch 10/20, Training Success Rate: 0.4225, Validation Success Rate: 0.405
Epoch 11/20, Training Success Rate: 0.43, Validation Success Rate: 0.405
Epoch 12/20, Training Success Rate: 0.435, Validation Success Rate: 0.395
Epoch 13/20, Training Success Rate: 0.44375, Validation Success Rate: 0.39
Epoch 14/20, Training Success Rate: 0.44625, Validation Success Rate: 0.375
Epoch 15/20, Training Success Rate: 0.455, Validation Success Rate: 0.385
Epoch 16/20, Training Success Rate: 0.46, Validation Success Rate: 0.38
Epoch 17/20, Training Success Rate: 0.47125, Validation Success Rate: 0.38
Epoch 18/20, Training Success Rate: 0.47, Validation Success Rate: 0.385
Epoch 19/20, Training Success Rate: 0.47125, Validation Success Rate: 0.375
Epoch 20/20, Training Success Rate: 0.48, Validation Success Rate: 0.37
```



نتیجه:

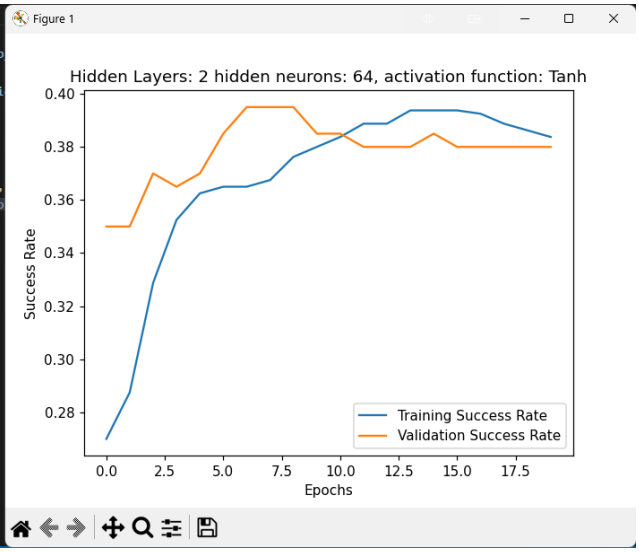
	precision	recall	f1-score	support
0	0.33	0.51	0.40	45
1	0.28	0.32	0.30	34
2	0.53	0.41	0.46	69
3	0.32	0.23	0.27	52
accuracy			0.37	200
macro avg	0.36	0.37	0.36	200
weighted avg	0.39	0.37	0.37	200

اجرا با ۰.۰۰۰۱:

```
optimizer = optim.Adam(mlp_model.parameters(), lr=0.0001)

success_rate_train, success_rate_val = train_mlp_model(mlp

plot_success_rate(success_rate_train, success_rate_val, hi

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor),
    val_accuracy = torch.sum(y_pred_val_mlp == y_val_tenso
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.3625, Validation Success Rate: 0.37
Epoch 6/20, Training Success Rate: 0.365, Validation Success Rate: 0.385
Epoch 7/20, Training Success Rate: 0.365, Validation Success Rate: 0.395
Epoch 8/20, Training Success Rate: 0.3675, Validation Success Rate: 0.395
Epoch 9/20, Training Success Rate: 0.37625, Validation Success Rate: 0.395
Epoch 10/20, Training Success Rate: 0.38, Validation Success Rate: 0.385
Epoch 11/20, Training Success Rate: 0.38375, Validation Success Rate: 0.385
Epoch 12/20, Training Success Rate: 0.38875, Validation Success Rate: 0.38
Epoch 13/20, Training Success Rate: 0.38875, Validation Success Rate: 0.38
Epoch 14/20, Training Success Rate: 0.39375, Validation Success Rate: 0.38
Epoch 15/20, Training Success Rate: 0.39375, Validation Success Rate: 0.385
Epoch 16/20, Training Success Rate: 0.39375, Validation Success Rate: 0.38
Epoch 17/20, Training Success Rate: 0.3925, Validation Success Rate: 0.38
Epoch 18/20, Training Success Rate: 0.38875, Validation Success Rate: 0.38
Epoch 19/20, Training Success Rate: 0.38625, Validation Success Rate: 0.38
Epoch 20/20, Training Success Rate: 0.38375, Validation Success Rate: 0.38
```



نتیجه:

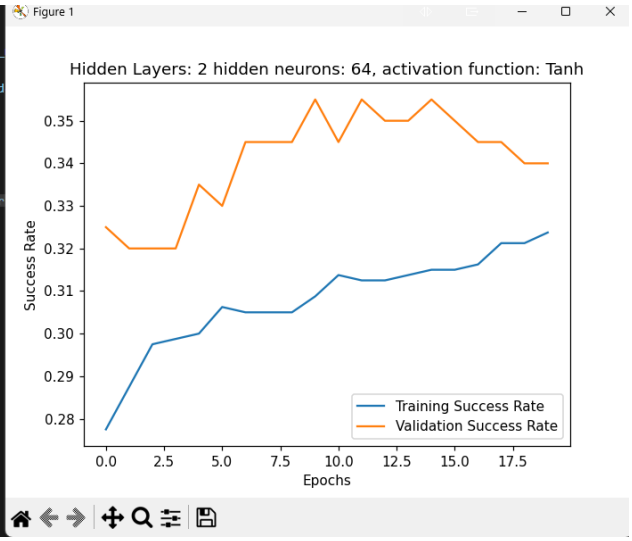
	precision	recall	f1-score	support
0	0.36	0.67	0.47	45
1	0.29	0.38	0.33	34
2	0.64	0.30	0.41	69
3	0.31	0.23	0.26	52
accuracy			0.38	200
macro avg	0.40	0.40	0.37	200
weighted avg	0.43	0.38	0.37	200

اجرا با ۰.۰۰۰۰۱:

```
169         optimizer = optim.Adam(mlp_model.parameters(), lr=0.00001)
170
171         success_rate_train, success_rate_val = train_mlp_model(mlp_
172
173         plot_success_rate(success_rate_train, success_rate_val, hid
174
175         # Evaluate the MLP Model on validation set
176         mlp_model.eval()
177         with torch.no_grad():
178             y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor),
179             val_accuracy = torch.sum(y_pred_val_mlp == y_val_tensor
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.3, Validation Success Rate: 0.335
Epoch 6/20, Training Success Rate: 0.30625, Validation Success Rate: 0.33
Epoch 7/20, Training Success Rate: 0.305, Validation Success Rate: 0.345
Epoch 8/20, Training Success Rate: 0.305, Validation Success Rate: 0.345
Epoch 9/20, Training Success Rate: 0.305, Validation Success Rate: 0.345
Epoch 10/20, Training Success Rate: 0.30875, Validation Success Rate: 0.355
Epoch 11/20, Training Success Rate: 0.31375, Validation Success Rate: 0.345
Epoch 12/20, Training Success Rate: 0.3125, Validation Success Rate: 0.355
Epoch 13/20, Training Success Rate: 0.3125, Validation Success Rate: 0.35
Epoch 14/20, Training Success Rate: 0.31375, Validation Success Rate: 0.35
Epoch 15/20, Training Success Rate: 0.315, Validation Success Rate: 0.355
Epoch 16/20, Training Success Rate: 0.315, Validation Success Rate: 0.35
Epoch 17/20, Training Success Rate: 0.31625, Validation Success Rate: 0.345
Epoch 18/20, Training Success Rate: 0.32125, Validation Success Rate: 0.345
Epoch 19/20, Training Success Rate: 0.32125, Validation Success Rate: 0.34
Epoch 20/20, Training Success Rate: 0.32375, Validation Success Rate: 0.34
```



نتیجه:

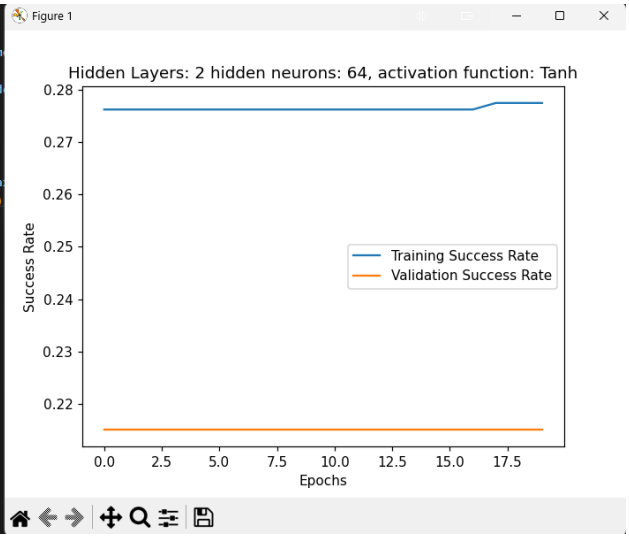
	precision	recall	f1-score	support
0	0.30	0.64	0.41	45
1	0.17	0.03	0.05	34
2	0.41	0.39	0.40	69
3	0.37	0.21	0.27	52
accuracy			0.34	200
macro avg	0.31	0.32	0.28	200
weighted avg	0.33	0.34	0.31	200

اجرا با ۰.۰۰۰۰۰۱:

```
169         optimizer = optim.Adam(mlp_model.parameters(), lr=0.000001)
170
171         success_rate_train, success_rate_val = train_mlp_model(mlp_m
172
173         plot_success_rate(success_rate_train, success_rate_val, hidd
174
175         # Evaluate the MLP Model on validation set
176         mlp_model.eval()
177         with torch.no_grad():
178             y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor), a
179             val_accuracy = torch.sum(y_pred_val_mlp == y_val_tensor)
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 6/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 7/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 8/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 9/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 10/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 11/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 12/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 13/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 14/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 15/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 16/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 17/20, Training Success Rate: 0.27625, Validation Success Rate: 0.215
Epoch 18/20, Training Success Rate: 0.2775, Validation Success Rate: 0.215
Epoch 19/20, Training Success Rate: 0.2775, Validation Success Rate: 0.215
Epoch 20/20, Training Success Rate: 0.2775, Validation Success Rate: 0.215
```



نتیجه:

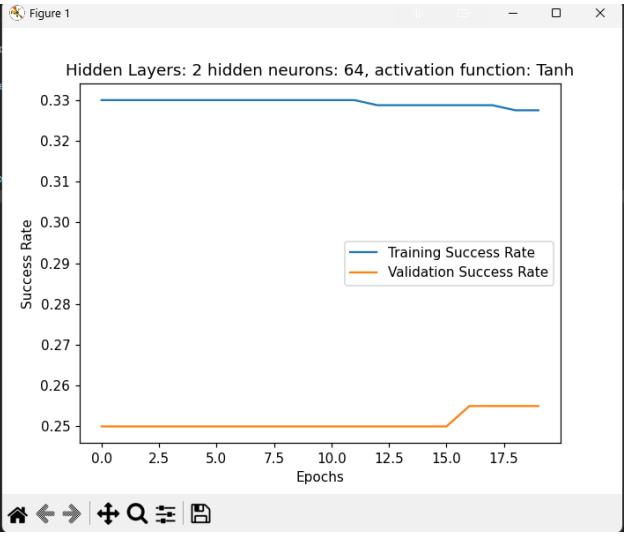
	precision	recall	f1-score	support
0	0.22	0.96	0.36	45
1	0.00	0.00	0.00	34
2	0.00	0.00	0.00	69
3	0.00	0.00	0.00	52
accuracy			0.21	200
macro avg	0.05	0.24	0.09	200
weighted avg	0.05	0.21	0.08	200

اجرا با ۰.۰۰۰۰۰۰۱:

```
optimizer = optim.Adam(mlp_model.parameters(), lr=0.0000001)

success_rate_train, success_rate_val = train_mlp_model(mlp_m

plot_success_rate(success_rate_train, success_rate_val, hidde

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tensor), a
    val_accuracy = torch.sum(y_pred_val_mlp == y_val_tensor)
```

PROBLEMS 26 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 6/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 7/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 8/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 9/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 10/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 11/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 12/20, Training Success Rate: 0.33, Validation Success Rate: 0.25
Epoch 13/20, Training Success Rate: 0.32875, Validation Success Rate: 0.25
Epoch 14/20, Training Success Rate: 0.32875, Validation Success Rate: 0.25
Epoch 15/20, Training Success Rate: 0.32875, Validation Success Rate: 0.25
Epoch 16/20, Training Success Rate: 0.32875, Validation Success Rate: 0.25
Epoch 17/20, Training Success Rate: 0.32875, Validation Success Rate: 0.255
Epoch 18/20, Training Success Rate: 0.32875, Validation Success Rate: 0.255
Epoch 19/20, Training Success Rate: 0.3275, Validation Success Rate: 0.255
Epoch 20/20, Training Success Rate: 0.3275, Validation Success Rate: 0.255
```



نتیجه:

	precision	recall	f1-score	support
0	0.24	0.71	0.36	45
1	0.28	0.56	0.38	34
2	0.00	0.00	0.00	69
3	0.00	0.00	0.00	52
accuracy			0.26	200
macro avg	0.13	0.32	0.18	200
weighted avg	0.10	0.26	0.14	200

که بهترین دقت برای اجرا با ۰.۰۰۰۱ می باشد که برابر ۰.۳۸ است.

گزارش ۷

برای آموزش با SGD نیز همین روال انجام شده:

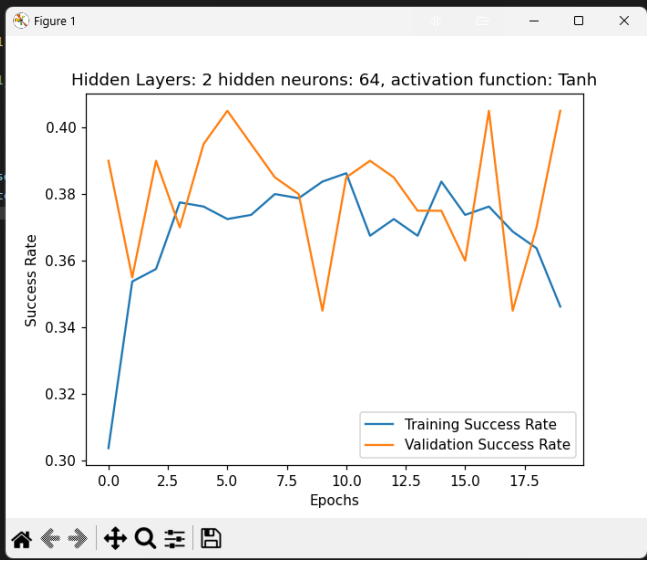
```
# optimizer = optim.Adam(mlp_model.parameters(), lr=0.0000001)
optimizer = optim.SGD(mlp_model.parameters(), lr=0.01, momentum=0.9)
```

اجرا با ۰.۰۱:

```
optimizer = optim.SGD(mlp_model.parameters(), lr=0.01, momentum=0.9)


success_rate_train, success_rate_val = train_mlp_model

plot_success_rate(success_rate_train, success_rate_val

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_tens
    val_accuracy = torch.sum(y_pred_val_mlp == y_val_t
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.37625, Validation Success Rate: 0.395
Epoch 6/20, Training Success Rate: 0.3725, Validation Success Rate: 0.405
Epoch 7/20, Training Success Rate: 0.37375, Validation Success Rate: 0.395
Epoch 8/20, Training Success Rate: 0.38, Validation Success Rate: 0.385
Epoch 9/20, Training Success Rate: 0.37875, Validation Success Rate: 0.38
Epoch 10/20, Training Success Rate: 0.38375, Validation Success Rate: 0.345
Epoch 11/20, Training Success Rate: 0.38625, Validation Success Rate: 0.385
Epoch 12/20, Training Success Rate: 0.3675, Validation Success Rate: 0.39
Epoch 13/20, Training Success Rate: 0.3725, Validation Success Rate: 0.385
Epoch 14/20, Training Success Rate: 0.3675, Validation Success Rate: 0.375
Epoch 15/20, Training Success Rate: 0.38375, Validation Success Rate: 0.375
Epoch 16/20, Training Success Rate: 0.37375, Validation Success Rate: 0.36
Epoch 17/20, Training Success Rate: 0.37625, Validation Success Rate: 0.405
Epoch 18/20, Training Success Rate: 0.36875, Validation Success Rate: 0.345
Epoch 19/20, Training Success Rate: 0.36375, Validation Success Rate: 0.37
Epoch 20/20, Training Success Rate: 0.34625, Validation Success Rate: 0.405
```



نتیجه:

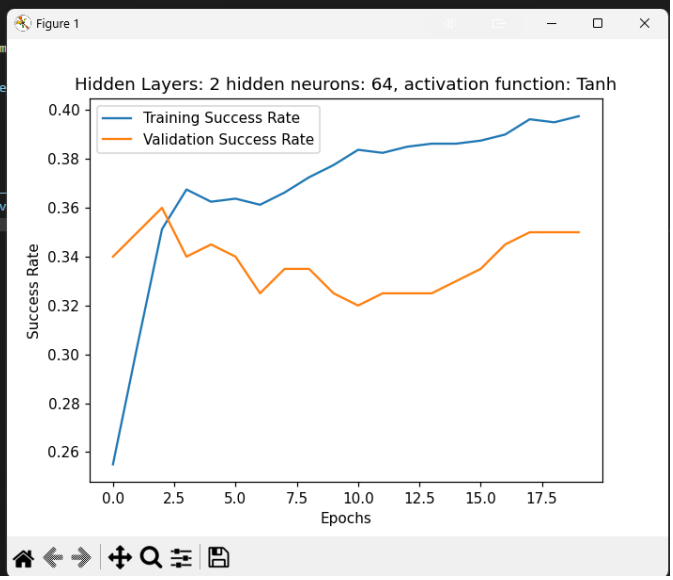
	precision	recall	f1-score	support
0	0.38	0.67	0.48	45
1	0.34	0.29	0.32	34
2	0.54	0.39	0.45	69
3	0.34	0.27	0.30	52
accuracy			0.41	200
macro avg	0.40	0.41	0.39	200
weighted avg	0.42	0.41	0.40	200

اجرا با ۰.۰۰۱:

```
170         optimizer = optim.SGD(mlp_model.parameters(), lr=0.001, momentum=0.9)
171
172
173         success_rate_train, success_rate_val = train_mlp_m
174
175         plot_success_rate(success_rate_train, success_rate
176
177         # Evaluate the MLP Model on validation set
178         mlp_model.eval()
179         with torch.no_grad():
180             y_pred_val_mlp = torch.argmax(mlp_model(X_val_
181             val_accuracy = torch.sum(y_pred_val_mlp == y_v
182
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.3625, Validation Success Rate: 0.345
Epoch 6/20, Training Success Rate: 0.36375, Validation Success Rate: 0.34
Epoch 7/20, Training Success Rate: 0.36125, Validation Success Rate: 0.325
Epoch 8/20, Training Success Rate: 0.36625, Validation Success Rate: 0.335
Epoch 9/20, Training Success Rate: 0.3725, Validation Success Rate: 0.335
Epoch 10/20, Training Success Rate: 0.3775, Validation Success Rate: 0.325
Epoch 11/20, Training Success Rate: 0.38375, Validation Success Rate: 0.32
Epoch 12/20, Training Success Rate: 0.3825, Validation Success Rate: 0.325
Epoch 13/20, Training Success Rate: 0.385, Validation Success Rate: 0.325
Epoch 14/20, Training Success Rate: 0.38625, Validation Success Rate: 0.325
Epoch 15/20, Training Success Rate: 0.38625, Validation Success Rate: 0.33
Epoch 16/20, Training Success Rate: 0.3875, Validation Success Rate: 0.335
Epoch 17/20, Training Success Rate: 0.39, Validation Success Rate: 0.345
Epoch 18/20, Training Success Rate: 0.39625, Validation Success Rate: 0.35
Epoch 19/20, Training Success Rate: 0.395, Validation Success Rate: 0.35
Epoch 20/20, Training Success Rate: 0.3975, Validation Success Rate: 0.35
```



نتیجه:

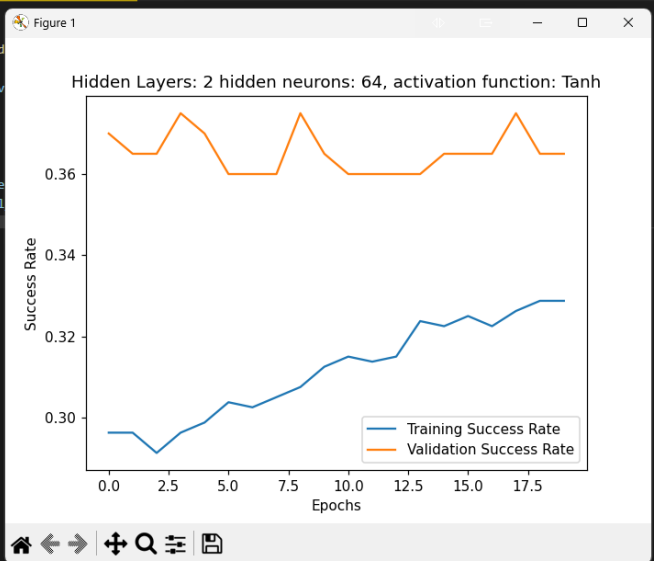
	precision	recall	f1-score	support
0	0.35	0.64	0.46	45
1	0.26	0.35	0.30	34
2	0.50	0.26	0.34	69
3	0.31	0.21	0.25	52
accuracy			0.35	200
macro avg	0.36	0.37	0.34	200
weighted avg	0.38	0.35	0.34	200

اجرا با ۰.۰۰۰۱:

```
optimizer = optim.SGD(mlp_model.parameters(), lr=0.0001, momentum=0.9)


success_rate_train, success_rate_val = train_mlp_mod

plot_success_rate(success_rate_train, success_rate_v

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_te
    val_accuracy = torch.sum(y_pred_val_mlp == y_val
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.29875, Validation Success Rate: 0.37
Epoch 6/20, Training Success Rate: 0.30375, Validation Success Rate: 0.36
Epoch 7/20, Training Success Rate: 0.3025, Validation Success Rate: 0.36
Epoch 8/20, Training Success Rate: 0.305, Validation Success Rate: 0.36
Epoch 9/20, Training Success Rate: 0.3075, Validation Success Rate: 0.375
Epoch 10/20, Training Success Rate: 0.3125, Validation Success Rate: 0.365
Epoch 11/20, Training Success Rate: 0.315, Validation Success Rate: 0.36
Epoch 12/20, Training Success Rate: 0.31375, Validation Success Rate: 0.36
Epoch 13/20, Training Success Rate: 0.315, Validation Success Rate: 0.36
Epoch 14/20, Training Success Rate: 0.32375, Validation Success Rate: 0.36
Epoch 15/20, Training Success Rate: 0.3225, Validation Success Rate: 0.365
Epoch 16/20, Training Success Rate: 0.325, Validation Success Rate: 0.365
Epoch 17/20, Training Success Rate: 0.3225, Validation Success Rate: 0.365
Epoch 18/20, Training Success Rate: 0.32625, Validation Success Rate: 0.375
Epoch 19/20, Training Success Rate: 0.32875, Validation Success Rate: 0.365
Epoch 20/20, Training Success Rate: 0.32875, Validation Success Rate: 0.365
```



نتیجه:

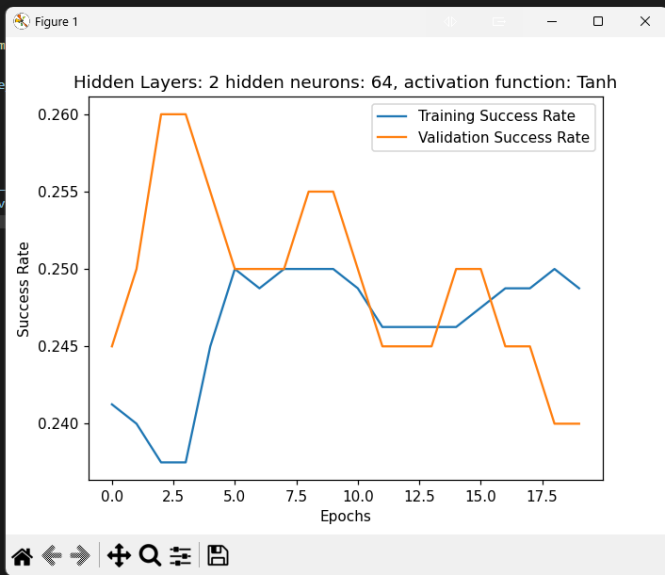
	precision	recall	f1-score	support
0	0.36	0.76	0.49	45
1	0.25	0.03	0.05	34
2	0.37	0.51	0.43	69
3	0.43	0.06	0.10	52
accuracy			0.36	200
macro avg	0.35	0.34	0.27	200
weighted avg	0.36	0.36	0.29	200

اجرا با ۰.۰۰۰۰۱:

```
optimizer = optim.SGD(mlp_model.parameters(), lr=0.00001, momentum=0.9)


success_rate_train, success_rate_val = train_mlp_m

plot_success_rate(success_rate_train, success_rate

# Evaluate the MLP Model on validation set
mlp_model.eval()
with torch.no_grad():
    y_pred_val_mlp = torch.argmax(mlp_model(X_val_
    val_accuracy = torch.sum(y_pred_val_mlp == y_v
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.245, Validation Success Rate: 0.255
Epoch 6/20, Training Success Rate: 0.25, Validation Success Rate: 0.25
Epoch 7/20, Training Success Rate: 0.24875, Validation Success Rate: 0.25
Epoch 8/20, Training Success Rate: 0.25, Validation Success Rate: 0.25
Epoch 9/20, Training Success Rate: 0.25, Validation Success Rate: 0.255
Epoch 10/20, Training Success Rate: 0.25, Validation Success Rate: 0.255
Epoch 11/20, Training Success Rate: 0.24875, Validation Success Rate: 0.25
Epoch 12/20, Training Success Rate: 0.24625, Validation Success Rate: 0.245
Epoch 13/20, Training Success Rate: 0.24625, Validation Success Rate: 0.245
Epoch 14/20, Training Success Rate: 0.24625, Validation Success Rate: 0.245
Epoch 15/20, Training Success Rate: 0.24625, Validation Success Rate: 0.25
Epoch 16/20, Training Success Rate: 0.2475, Validation Success Rate: 0.25
Epoch 17/20, Training Success Rate: 0.24875, Validation Success Rate: 0.245
Epoch 18/20, Training Success Rate: 0.24875, Validation Success Rate: 0.245
Epoch 19/20, Training Success Rate: 0.25, Validation Success Rate: 0.24
Epoch 20/20, Training Success Rate: 0.24875, Validation Success Rate: 0.24
```



نتیجه:

	precision	recall	f1-score	support
0	0.17	0.13	0.15	45
1	0.15	0.09	0.11	34
2	0.17	0.06	0.09	69
3	0.29	0.67	0.40	52
accuracy			0.24	200
macro avg	0.20	0.24	0.19	200
weighted avg	0.20	0.24	0.19	200

همان طور که مشاهده شد بهترین جواب برای اجرا با ۰.۰۱ بود و مقدار دقت آن ۰.۴۱ بود.

گزارش ۸

طبق اجرا بهینه گر SGD نسبت به بهینه گر Adam جواب بهتری داد البته ممکن است با ترکیب بهینه گر با تعداد لایه و نورون و فعال ساز جواب بهتری به دست بیاید.

گزارش ۹

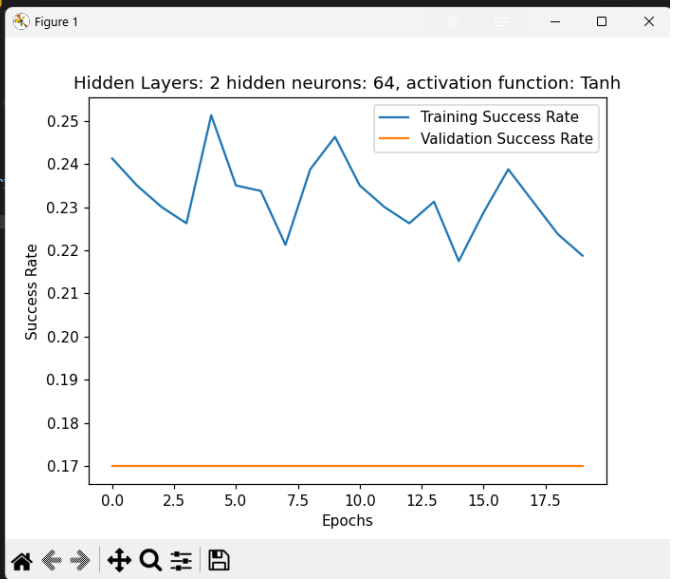
برای تست batch_size هم به همین شکل عمل میکنیم:

اجرا با ۱:

```
train_loader = DataLoader(train_dataset, batch_size=1, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=1, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU,
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_tr

# Evaluate the best MLP model on the testing set
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.25125, Validation Success Rate: 0.17
Epoch 6/20, Training Success Rate: 0.235, Validation Success Rate: 0.17
Epoch 7/20, Training Success Rate: 0.23375, Validation Success Rate: 0.17
Epoch 8/20, Training Success Rate: 0.22125, Validation Success Rate: 0.17
Epoch 9/20, Training Success Rate: 0.23875, Validation Success Rate: 0.17
Epoch 10/20, Training Success Rate: 0.24625, Validation Success Rate: 0.17
Epoch 11/20, Training Success Rate: 0.235, Validation Success Rate: 0.17
Epoch 12/20, Training Success Rate: 0.23, Validation Success Rate: 0.17
Epoch 13/20, Training Success Rate: 0.22625, Validation Success Rate: 0.17
Epoch 14/20, Training Success Rate: 0.23125, Validation Success Rate: 0.17
Epoch 15/20, Training Success Rate: 0.2175, Validation Success Rate: 0.17
Epoch 16/20, Training Success Rate: 0.22875, Validation Success Rate: 0.17
Epoch 17/20, Training Success Rate: 0.23875, Validation Success Rate: 0.17
Epoch 18/20, Training Success Rate: 0.23125, Validation Success Rate: 0.17
Epoch 19/20, Training Success Rate: 0.22375, Validation Success Rate: 0.17
Epoch 20/20, Training Success Rate: 0.21875, Validation Success Rate: 0.17
```



نتیجه:

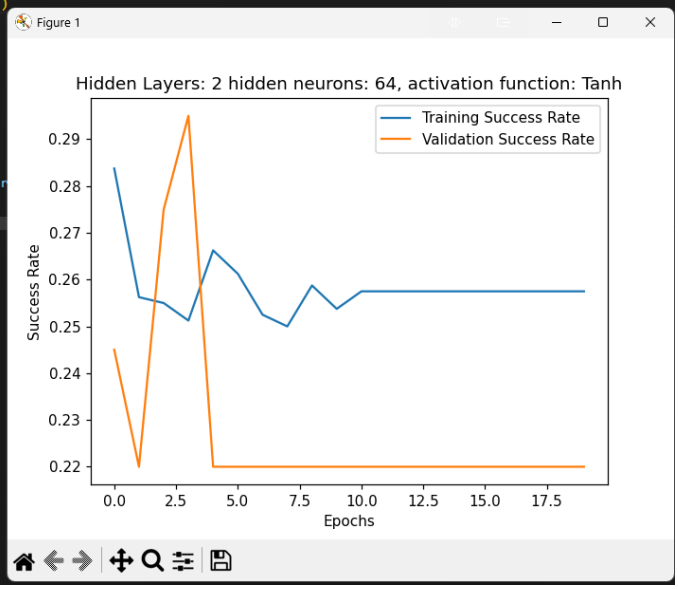
	precision	recall	f1-score	support
0	0.00	0.00	0.00	45
1	0.17	1.00	0.29	34
2	0.00	0.00	0.00	69
3	0.00	0.00	0.00	52
accuracy			0.17	200
macro avg	0.04	0.25	0.07	200
weighted avg	0.03	0.17	0.05	200

اجرا با ۴:

```
train_loader = DataLoader(train_dataset, batch_size=4, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=4, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU,
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_tr

# Evaluate the best MLP model on the testing set
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.26625, Validation Success Rate: 0.22
Epoch 6/20, Training Success Rate: 0.26125, Validation Success Rate: 0.22
Epoch 7/20, Training Success Rate: 0.2525, Validation Success Rate: 0.22
Epoch 8/20, Training Success Rate: 0.25, Validation Success Rate: 0.22
Epoch 9/20, Training Success Rate: 0.25875, Validation Success Rate: 0.22
Epoch 10/20, Training Success Rate: 0.25375, Validation Success Rate: 0.22
Epoch 11/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 12/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 13/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 14/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 15/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 16/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 17/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 18/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 19/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
Epoch 20/20, Training Success Rate: 0.2575, Validation Success Rate: 0.22
```



نتیجه:

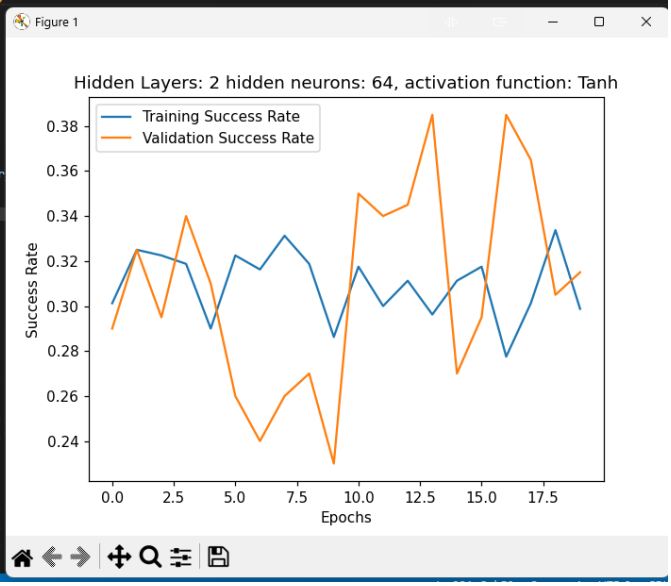
	precision	recall	f1-score	support
0	0.22	0.98	0.36	45
1	0.00	0.00	0.00	34
2	0.00	0.00	0.00	69
3	0.00	0.00	0.00	52
accuracy			0.22	200
macro avg	0.06	0.24	0.09	200
weighted avg	0.05	0.22	0.08	200

اجرا با ۸:

```
train_loader = DataLoader(train_dataset, batch_size=8, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=8, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU,
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_tr

# Evaluate the best MLP model on the testing set
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.29, Validation Success Rate: 0.31
Epoch 6/20, Training Success Rate: 0.3225, Validation Success Rate: 0.26
Epoch 7/20, Training Success Rate: 0.31625, Validation Success Rate: 0.24
Epoch 8/20, Training Success Rate: 0.33125, Validation Success Rate: 0.26
Epoch 9/20, Training Success Rate: 0.31875, Validation Success Rate: 0.27
Epoch 10/20, Training Success Rate: 0.28625, Validation Success Rate: 0.23
Epoch 11/20, Training Success Rate: 0.3175, Validation Success Rate: 0.35
Epoch 12/20, Training Success Rate: 0.3, Validation Success Rate: 0.34
Epoch 13/20, Training Success Rate: 0.31125, Validation Success Rate: 0.345
Epoch 14/20, Training Success Rate: 0.29625, Validation Success Rate: 0.385
Epoch 15/20, Training Success Rate: 0.31125, Validation Success Rate: 0.27
Epoch 16/20, Training Success Rate: 0.3175, Validation Success Rate: 0.295
Epoch 17/20, Training Success Rate: 0.2775, Validation Success Rate: 0.385
Epoch 18/20, Training Success Rate: 0.30125, Validation Success Rate: 0.365
Epoch 19/20, Training Success Rate: 0.33375, Validation Success Rate: 0.305
Epoch 20/20, Training Success Rate: 0.29875, Validation Success Rate: 0.315
```



نتیجه:

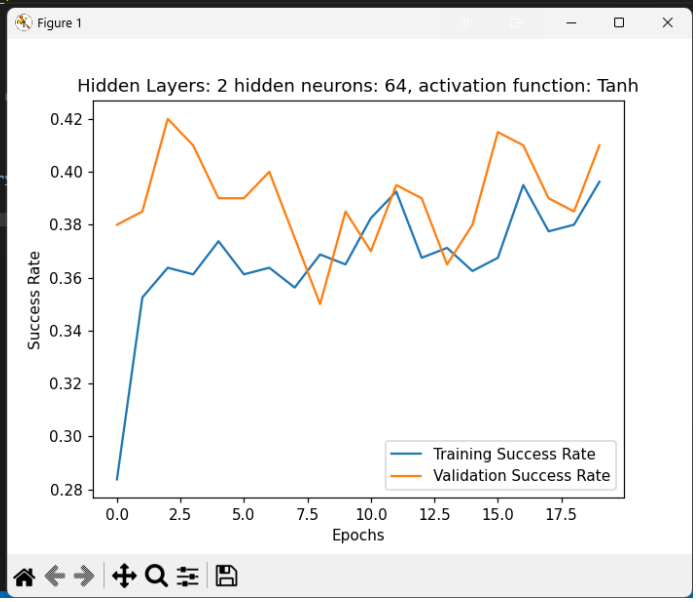
	precision	recall	f1-score	support
0	0.38	0.42	0.40	45
1	0.24	0.68	0.36	34
2	0.38	0.30	0.34	69
3	0.00	0.00	0.00	52
accuracy			0.32	200
macro avg	0.25	0.35	0.27	200
weighted avg	0.26	0.32	0.27	200

اجرا با ۳۲:

```
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=32, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU,
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_tr

# Evaluate the best MLP model on the testing set
```

LEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
h 5/20, Training Success Rate: 0.37375, Validation Success Rate: 0.39
h 6/20, Training Success Rate: 0.36125, Validation Success Rate: 0.39
h 7/20, Training Success Rate: 0.36375, Validation Success Rate: 0.4
h 8/20, Training Success Rate: 0.35625, Validation Success Rate: 0.375
h 9/20, Training Success Rate: 0.36875, Validation Success Rate: 0.35
h 10/20, Training Success Rate: 0.365, Validation Success Rate: 0.385
h 11/20, Training Success Rate: 0.3825, Validation Success Rate: 0.37
h 12/20, Training Success Rate: 0.3925, Validation Success Rate: 0.395
h 13/20, Training Success Rate: 0.3675, Validation Success Rate: 0.39
h 14/20, Training Success Rate: 0.37125, Validation Success Rate: 0.365
h 15/20, Training Success Rate: 0.3625, Validation Success Rate: 0.38
h 16/20, Training Success Rate: 0.3675, Validation Success Rate: 0.415
h 17/20, Training Success Rate: 0.395, Validation Success Rate: 0.41
h 18/20, Training Success Rate: 0.3775, Validation Success Rate: 0.39
h 19/20, Training Success Rate: 0.38, Validation Success Rate: 0.385
h 20/20, Training Success Rate: 0.39625, Validation Success Rate: 0.41
```



نتیجه:

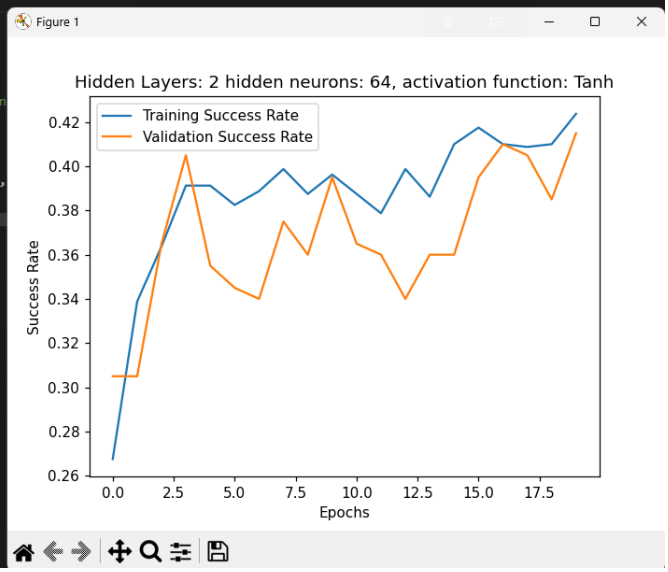
	precision	recall	f1-score	support
0	0.38	0.56	0.45	45
1	0.29	0.44	0.35	34
2	0.51	0.51	0.51	69
3	0.44	0.13	0.21	52
accuracy			0.41	200
macro avg	0.41	0.41	0.38	200
weighted avg	0.43	0.41	0.39	200

اجرا با ۶۴:

```
train_loader = DataLoader(train_dataset, batch_size=64, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=64, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU, nn
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_try,

# Evaluate the best MLP model on the testing set
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.39125, Validation Success Rate: 0.355
Epoch 6/20, Training Success Rate: 0.3825, Validation Success Rate: 0.345
Epoch 7/20, Training Success Rate: 0.38875, Validation Success Rate: 0.34
Epoch 8/20, Training Success Rate: 0.39875, Validation Success Rate: 0.375
Epoch 9/20, Training Success Rate: 0.3875, Validation Success Rate: 0.36
Epoch 10/20, Training Success Rate: 0.39625, Validation Success Rate: 0.395
Epoch 11/20, Training Success Rate: 0.3875, Validation Success Rate: 0.365
Epoch 12/20, Training Success Rate: 0.37875, Validation Success Rate: 0.36
Epoch 13/20, Training Success Rate: 0.39875, Validation Success Rate: 0.34
Epoch 14/20, Training Success Rate: 0.38625, Validation Success Rate: 0.36
Epoch 15/20, Training Success Rate: 0.41, Validation Success Rate: 0.36
Epoch 16/20, Training Success Rate: 0.4175, Validation Success Rate: 0.395
Epoch 17/20, Training Success Rate: 0.41, Validation Success Rate: 0.41
Epoch 18/20, Training Success Rate: 0.40875, Validation Success Rate: 0.405
Epoch 19/20, Training Success Rate: 0.41, Validation Success Rate: 0.385
Epoch 20/20, Training Success Rate: 0.42375, Validation Success Rate: 0.415
```



نتیجه:

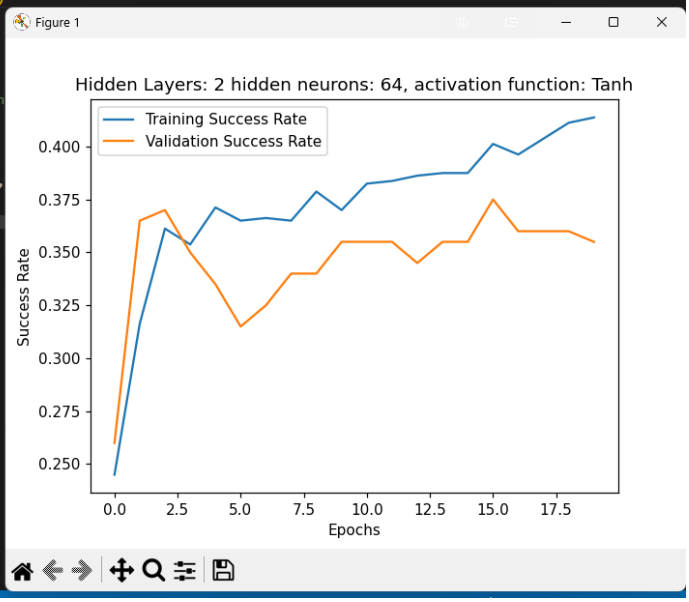
	precision	recall	f1-score	support
0	0.35	0.51	0.42	45
1	0.31	0.47	0.37	34
2	0.55	0.45	0.50	69
3	0.48	0.25	0.33	52
accuracy			0.41	200
macro avg	0.42	0.42	0.40	200
weighted avg	0.45	0.41	0.41	200

اجرا با ۱۲۸:

```
train_loader = DataLoader(train_dataset, batch_size=128, shuffle=False)
val_loader = DataLoader(val_dataset, batch_size=128, shuffle=False)

# Hidden layers, neurons, and activation functions to try
# hidden_layers_to_try = [2, 3, 4, 5, 6]
# hidden_neurons_to_try = [32, 64, 128, 256, 512]
# activation_functions_to_try = [nn.LeakyReLU, nn.Tanh, nn.ReLU, nn
hidden_layers_to_try = [2]
hidden_neurons_to_try = [64]
activation_functions_to_try = [nn.Tanh]
best_mlp_model = train_and_evaluate_mlp_model(hidden_layers_to_try,

# Evaluate the best MLP model on the testing set
```

EMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
h 5/20, Training Success Rate: 0.37125, Validation Success Rate: 0.335
h 6/20, Training Success Rate: 0.365, Validation Success Rate: 0.315
h 7/20, Training Success Rate: 0.36625, Validation Success Rate: 0.325
h 8/20, Training Success Rate: 0.365, Validation Success Rate: 0.34
h 9/20, Training Success Rate: 0.37875, Validation Success Rate: 0.34
h 10/20, Training Success Rate: 0.37, Validation Success Rate: 0.355
h 11/20, Training Success Rate: 0.3825, Validation Success Rate: 0.355
h 12/20, Training Success Rate: 0.38375, Validation Success Rate: 0.355
h 13/20, Training Success Rate: 0.38625, Validation Success Rate: 0.345
h 14/20, Training Success Rate: 0.3875, Validation Success Rate: 0.355
h 15/20, Training Success Rate: 0.3875, Validation Success Rate: 0.355
h 16/20, Training Success Rate: 0.40125, Validation Success Rate: 0.375
h 17/20, Training Success Rate: 0.39625, Validation Success Rate: 0.36
h 18/20, Training Success Rate: 0.40375, Validation Success Rate: 0.36
h 19/20, Training Success Rate: 0.41125, Validation Success Rate: 0.36
h 20/20, Training Success Rate: 0.41375, Validation Success Rate: 0.355
```



نتیجه:

	precision	recall	f1-score	support
0	0.36	0.60	0.45	45
1	0.30	0.47	0.37	34
2	0.45	0.22	0.29	69
3	0.34	0.25	0.29	52
accuracy			0.36	200
macro avg	0.36	0.38	0.35	200
weighted avg	0.38	0.35	0.34	200

گزارش ۱۰

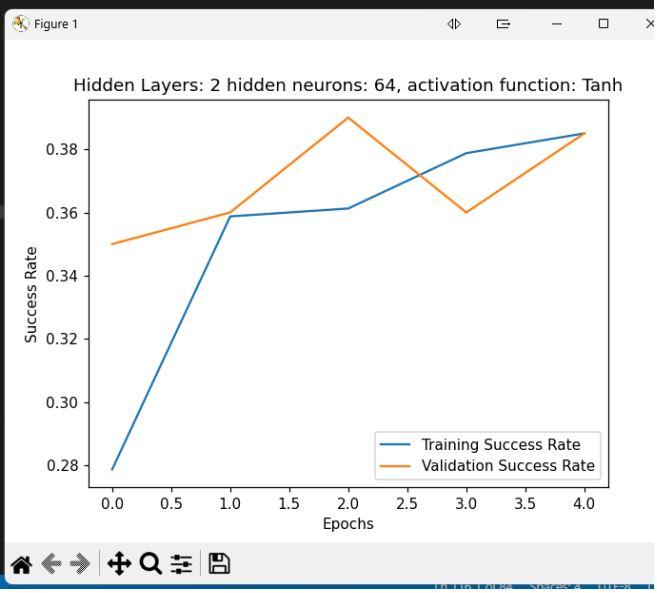
آموزش با تعداد دور های متفاوت:

اجرا با ۵:

```python
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=5):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
           0       0.36      0.60      0.45        45
           1       0.30      0.47      0.37        34
           2       0.45      0.22      0.29        69
           3       0.34      0.25      0.29        52

    accuracy                           0.36       200
   macro avg       0.36      0.38      0.35       200
weighted avg       0.38      0.35      0.34       200

(env_hw_2) PS D:\sku\machine learning\hw_2_ml> python code_3.py
Epoch 1/5, Training Success Rate: 0.27875, Validation Success Rate: 0.35
Epoch 2/5, Training Success Rate: 0.35875, Validation Success Rate: 0.36
Epoch 3/5, Training Success Rate: 0.36125, Validation Success Rate: 0.39
Epoch 4/5, Training Success Rate: 0.37875, Validation Success Rate: 0.36
Epoch 5/5, Training Success Rate: 0.385, Validation Success Rate: 0.385
```



نتیجه:

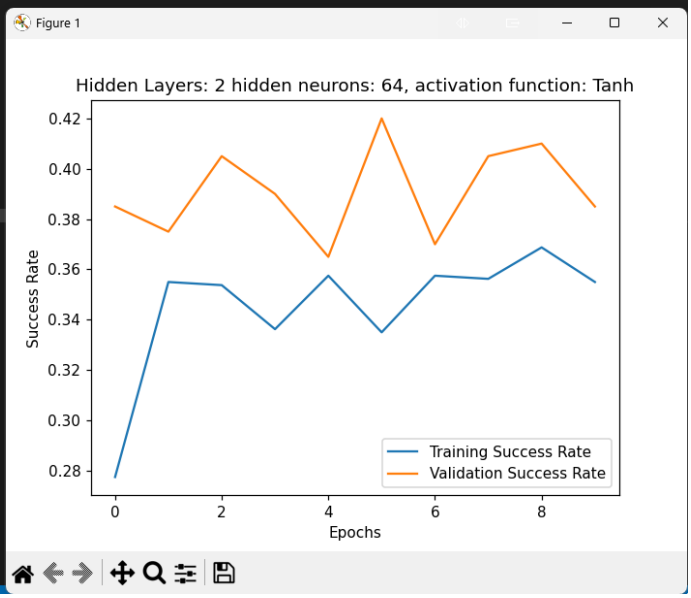
	precision	recall	f1-score	support
0	0.36	0.62	0.46	45
1	0.28	0.35	0.31	34
2	0.55	0.39	0.46	69
3	0.32	0.19	0.24	52
accuracy			0.38	200
macro avg	0.38	0.39	0.37	200
weighted avg	0.40	0.39	0.38	200

اجرا با ۱۰:

```python
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=10):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
 accuracy                           0.38       200
 macro avg       0.38      0.39     0.37       200
ghted avg        0.40      0.39     0.38       200

hw_hw_2) PS D:\sku\machine learning\hw_2_ml> python code_3.py
och 1/10, Training Success Rate: 0.2775, Validation Success Rate: 0.385
och 2/10, Training Success Rate: 0.355, Validation Success Rate: 0.375
och 3/10, Training Success Rate: 0.35375, Validation Success Rate: 0.405
och 4/10, Training Success Rate: 0.33625, Validation Success Rate: 0.39
och 5/10, Training Success Rate: 0.3575, Validation Success Rate: 0.365
och 6/10, Training Success Rate: 0.335, Validation Success Rate: 0.42
och 7/10, Training Success Rate: 0.3575, Validation Success Rate: 0.37
och 8/10, Training Success Rate: 0.35625, Validation Success Rate: 0.405
och 9/10, Training Success Rate: 0.36875, Validation Success Rate: 0.41
och 10/10, Training Success Rate: 0.355, Validation Success Rate: 0.385
```



نتیجه:

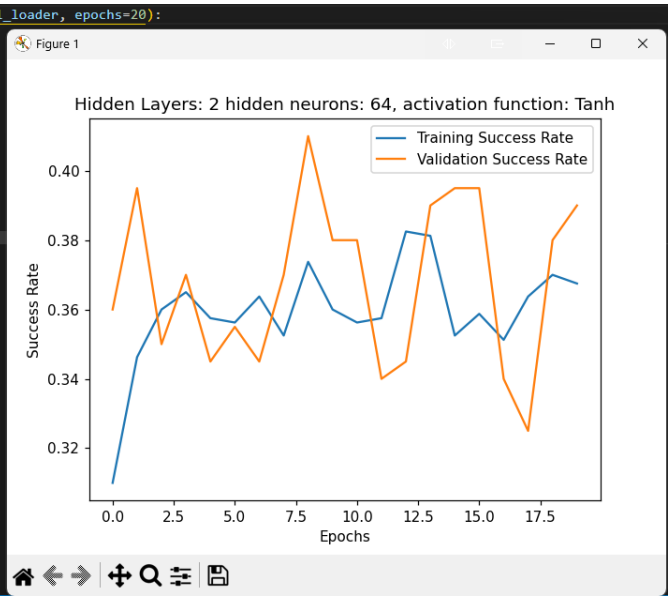
	precision	recall	f1-score	support
0	0.36	0.47	0.41	45
1	0.37	0.29	0.33	34
2	0.44	0.61	0.51	69
3	0.20	0.08	0.11	52
accuracy			0.38	200
macro avg	0.34	0.36	0.34	200
weighted avg	0.35	0.39	0.35	200

اجرا با ۲۰:

```python
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=20):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
Epoch 5/20, Training Success Rate: 0.3575, Validation Success Rate: 0.345
Epoch 6/20, Training Success Rate: 0.35625, Validation Success Rate: 0.355
Epoch 7/20, Training Success Rate: 0.36375, Validation Success Rate: 0.345
Epoch 8/20, Training Success Rate: 0.3525, Validation Success Rate: 0.37
Epoch 9/20, Training Success Rate: 0.37375, Validation Success Rate: 0.41
Epoch 10/20, Training Success Rate: 0.36, Validation Success Rate: 0.38
Epoch 11/20, Training Success Rate: 0.35625, Validation Success Rate: 0.38
Epoch 12/20, Training Success Rate: 0.3575, Validation Success Rate: 0.34
Epoch 13/20, Training Success Rate: 0.3825, Validation Success Rate: 0.345
Epoch 14/20, Training Success Rate: 0.38125, Validation Success Rate: 0.39
Epoch 15/20, Training Success Rate: 0.3525, Validation Success Rate: 0.395
Epoch 16/20, Training Success Rate: 0.35875, Validation Success Rate: 0.395
Epoch 17/20, Training Success Rate: 0.35125, Validation Success Rate: 0.34
Epoch 18/20, Training Success Rate: 0.36375, Validation Success Rate: 0.325
Epoch 19/20, Training Success Rate: 0.37, Validation Success Rate: 0.38
Epoch 20/20, Training Success Rate: 0.3675, Validation Success Rate: 0.39
```



نتیجه:

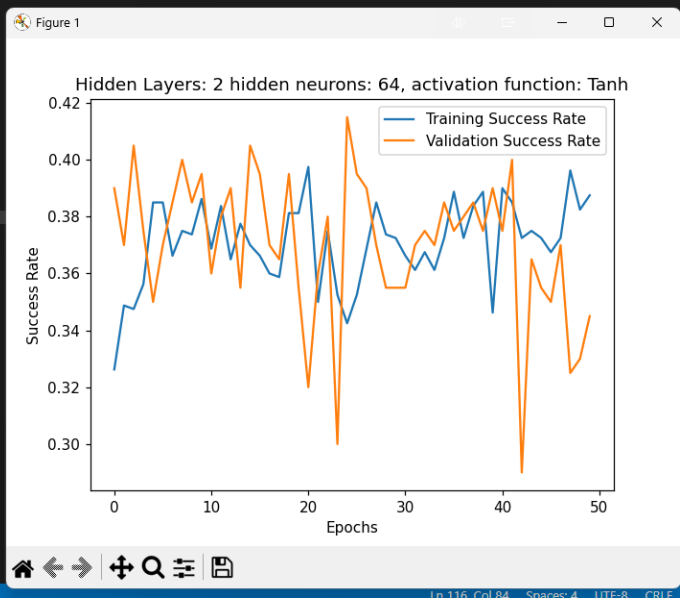
	precision	recall	f1-score	support
0	0.34	0.64	0.45	45
1	0.24	0.15	0.18	34
2	0.47	0.51	0.49	69
3	0.45	0.17	0.25	52
accuracy			0.39	200
macro avg	0.38	0.37	0.34	200
weighted avg	0.40	0.39	0.37	200

اجرا با ۵۰:

```python
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=50):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
```

PROBLEMS 27 DEBUG CONSOLE OUTPUT TERMINAL COMMENTS

```
ch 35/50, Training Success Rate: 0.3725, Validation Success Rate: 0.385
ch 36/50, Training Success Rate: 0.38875, Validation Success Rate: 0.375
ch 37/50, Training Success Rate: 0.3725, Validation Success Rate: 0.38
ch 38/50, Training Success Rate: 0.38375, Validation Success Rate: 0.385
ch 39/50, Training Success Rate: 0.38875, Validation Success Rate: 0.375
ch 40/50, Training Success Rate: 0.34625, Validation Success Rate: 0.39
ch 41/50, Training Success Rate: 0.39, Validation Success Rate: 0.375
ch 42/50, Training Success Rate: 0.385, Validation Success Rate: 0.4
ch 43/50, Training Success Rate: 0.3725, Validation Success Rate: 0.29
ch 44/50, Training Success Rate: 0.375, Validation Success Rate: 0.365
ch 45/50, Training Success Rate: 0.3725, Validation Success Rate: 0.355
ch 46/50, Training Success Rate: 0.3675, Validation Success Rate: 0.35
ch 47/50, Training Success Rate: 0.3725, Validation Success Rate: 0.37
ch 48/50, Training Success Rate: 0.39625, Validation Success Rate: 0.325
ch 49/50, Training Success Rate: 0.3825, Validation Success Rate: 0.33
ch 50/50, Training Success Rate: 0.3875, Validation Success Rate: 0.345
```



نتیجه:

	precision	recall	f1-score	support
0	0.31	0.58	0.40	45
1	0.26	0.29	0.27	34
2	0.47	0.43	0.45	69
3	0.25	0.06	0.09	52
accuracy			0.34	200
macro avg	0.32	0.34	0.30	200
weighted avg	0.34	0.34	0.32	200

اجرا با ۱۰۰:

```python
def train_mlp_model(model, criterion, optimizer, train_loader, val_loader, epochs=100):
    success_rate_train = []
    success_rate_val = []

    for epoch in range(epochs):
        model.train()
        total_train = 0
        correct_train = 0

        for inputs, labels in train_loader:
            optimizer.zero_grad()
            outputs = model(inputs)
```

```
EMS 27  DEBUG CONSOLE  OUTPUT  TERMINAL  COMMENTS

85/100, Training Success Rate: 0.40125, Validation Success Rate: 0.355
86/100, Training Success Rate: 0.4025, Validation Success Rate: 0.38
87/100, Training Success Rate: 0.38, Validation Success Rate: 0.335
88/100, Training Success Rate: 0.38, Validation Success Rate: 0.34
89/100, Training Success Rate: 0.3725, Validation Success Rate: 0.31
90/100, Training Success Rate: 0.375, Validation Success Rate: 0.35
91/100, Training Success Rate: 0.36375, Validation Success Rate: 0.335
92/100, Training Success Rate: 0.37875, Validation Success Rate: 0.385
93/100, Training Success Rate: 0.36625, Validation Success Rate: 0.345
94/100, Training Success Rate: 0.3725, Validation Success Rate: 0.35
95/100, Training Success Rate: 0.3625, Validation Success Rate: 0.35
96/100, Training Success Rate: 0.37, Validation Success Rate: 0.32
97/100, Training Success Rate: 0.3575, Validation Success Rate: 0.355
98/100, Training Success Rate: 0.37625, Validation Success Rate: 0.34
99/100, Training Success Rate: 0.38875, Validation Success Rate: 0.365
100/100, Training Success Rate: 0.375, Validation Success Rate: 0.365
```

نتیجه:

	precision	recall	f1-score	support
0	0.34	0.49	0.40	45
1	0.35	0.24	0.28	34
2	0.45	0.36	0.40	69
3	0.32	0.35	0.33	52
accuracy			0.36	200
macro avg	0.36	0.36	0.35	200
weighted avg	0.37	0.36	0.36	200

گزارش ۱۱

رابطه‌ای بین batch_size و epochs

Batch Size (اندازه دسته): این پارامتر نشان دهنده تعداد نمونه‌های داده که به عنوان ورودی به مدل داده می‌شود در هر مرحله است. بزرگتر کردن batch_size منجر به افزایش سرعت آموزش می‌شود، زیرا محاسبات موازی‌تر انجام می‌شود. اما افزایش این مقدار ممکن است منجر به حافظه‌گیری زیاد و کاهش تنوع در به‌روزرسانی وزن‌ها شود.

Epochs (تعداد دوره‌ها): یک epoch به مرور کامل دیتاست آموزشی می‌پردازد. افزایش تعداد epochs به معنای افزایش تعداد مراتبی است که مدل داده‌ها را مشاهده و به‌روزرسانی وزن‌ها را انجام می‌دهد. افزایش تعداد epochs می‌تواند منجر به بهبود دقت مدل و یادگیری الگوهای پیچیده‌تر شود.

رابطه بین این دو پارامتر به این صورت است که با افزایش اندازه دسته (batch_size)، معمولاً تعداد دفعاتی که مدل به تمامی داده‌های آموزشی دسترسی پیدا می‌کند (تعداد epochs) کاهش می‌یابد. این امر معمولاً به معنای این است که مدل با دیدن هر دسته داده بهتر یاد می‌گیرد و نیاز به تعداد کمتری epochs دارد.

با این حال، این قاعده ممکن است در موارد خاصی تغییر کند، و برخی مدل‌ها یا دیتاست‌ها ممکن است به بهینه‌ترین ترکیب از این دو پارامتر نیاز داشته باشند. در عمل، بررسی و تنظیم این پارامترها نیاز به آزمون و خطا و تجربه دارد.

با تغییر مقادیر بهترین جوابی که حاصل شد دقت ۰.۴۳ بود:

```
Best MLP Model was trained with 4 hidden layers, 256 hidden neurons, and ReLU activation function.

Best MLP Model Classification Report - Testing Set:
              precision    recall  f1-score   support

           0       0.35      0.60      0.44        45
           1       0.37      0.29      0.33        34
           2       0.54      0.70      0.61        69
           3       0.17      0.02      0.03        52

    accuracy                           0.43       200
   macro avg       0.36      0.40      0.35       200
weighted avg       0.37      0.43      0.37       200
```