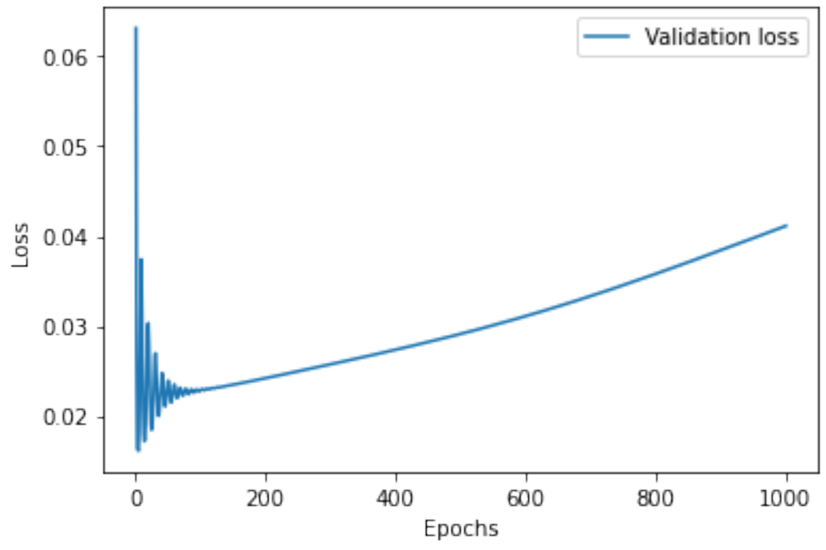
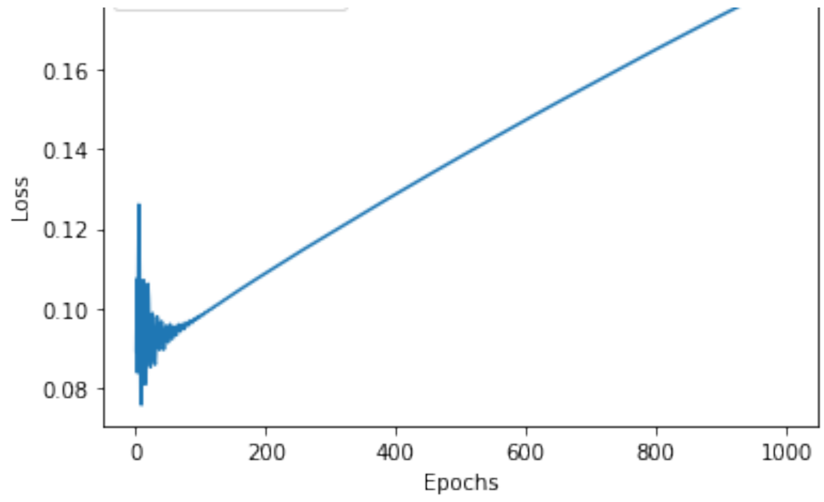
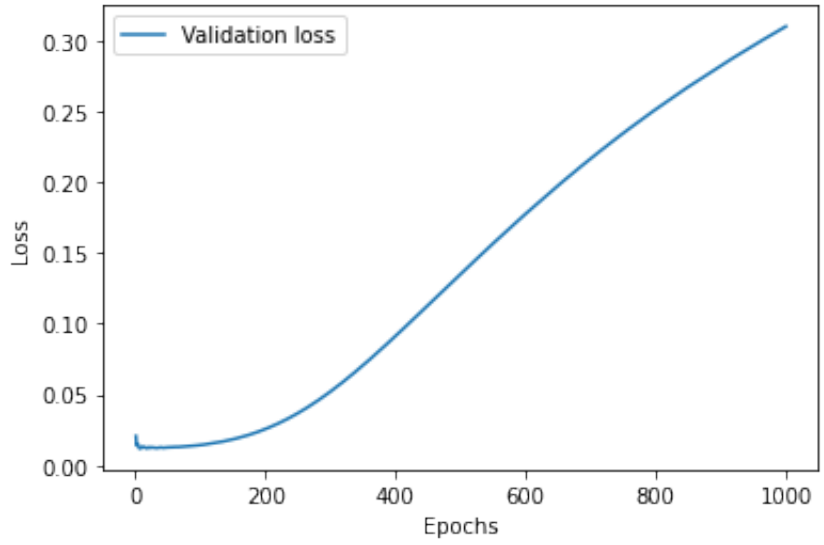
```
        # Plot the validation loss curve
        if draw_curve:
            plt.figure()
            plt.plot(np.arange(len(val_ar
            plt.xlabel('Epochs')
            plt.ylabel('Loss')
            plt.legend()

    return lowest_val_loss.item()
```

```
train_and_val(model_no_softmax,·pytorch_f
```

```
<ipython-input-97-f0eea764c926>:24: UserWarning: To copy construct from a ten
  Xs = torch.tensor(train_X).float()
<ipython-input-97-f0eea764c926>:25: UserWarning: To copy construct from a ten
  ys = torch.tensor(train_y).long()
Number of epochs with lowest validation: 5
Test accuracy: 0.9666666666666667
Number of epochs with lowest validation: 9
Test accuracy: 0.9830508474576272
Number of epochs with lowest validation: 7
Test accuracy: 1.0
0.011702748946845531
```

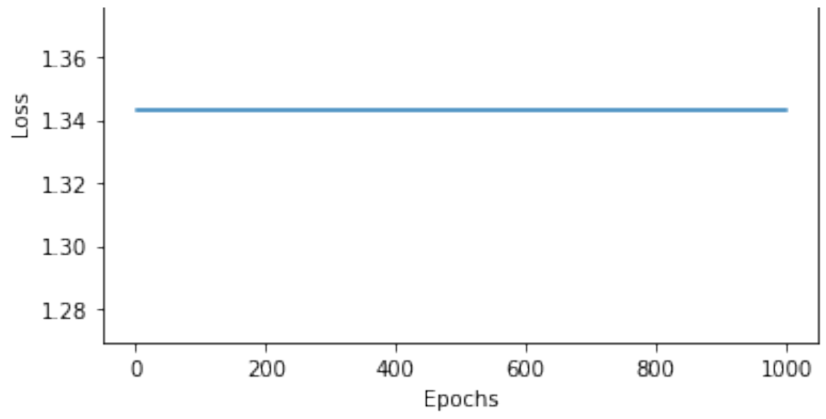


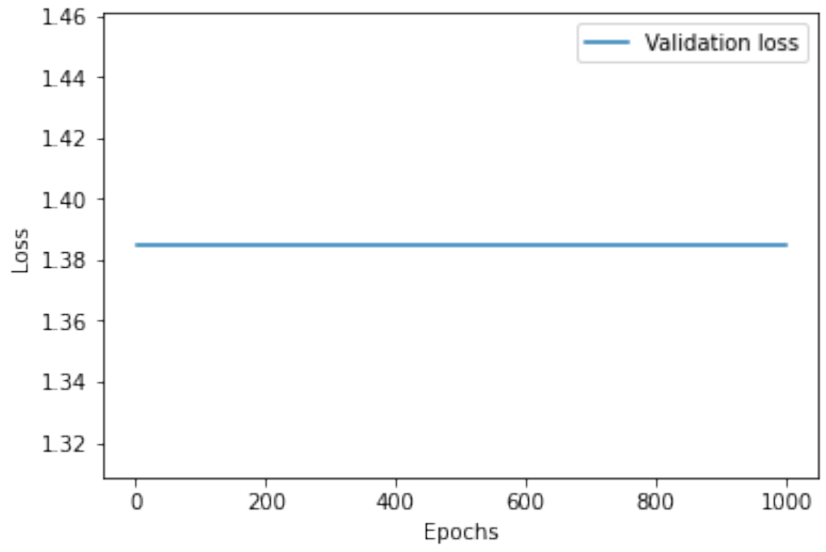


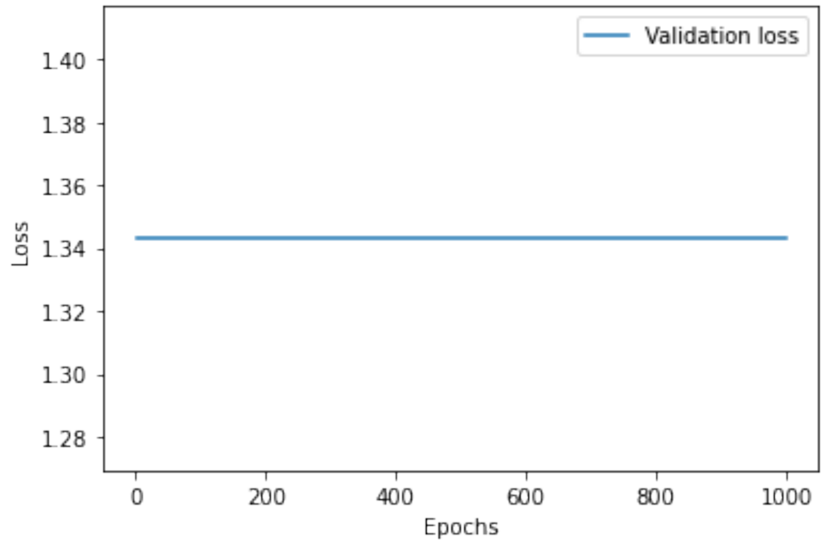


```
train_and_val(model_softmax,·pytorch_feat
```

```
<ipython-input-71-f0eea764c926>:24: UserWarning: To copy construct from a ten
  Xs = torch.tensor(train_X).float()
<ipython-input-71-f0eea764c926>:25: UserWarning: To copy construct from a ten
  ys = torch.tensor(train_y).long()
Number of epochs with lowest validation: 1
Test accuracy: 0.36666666666666664
Number of epochs with lowest validation: 1
Test accuracy: 0.2542372881355932
Number of epochs with lowest validation: 1
Test accuracy: 0.1864406779661017
1.3431113958358765
```


Loss
1.36
1.34
1.32
1.30
1.28
0
200
400
600
800
1000
Epochs


Validation loss
Loss
1.46
1.44
1.42
1.40
1.38
1.36
1.34
1.32
0
200
400
600
800
1000
Epochs

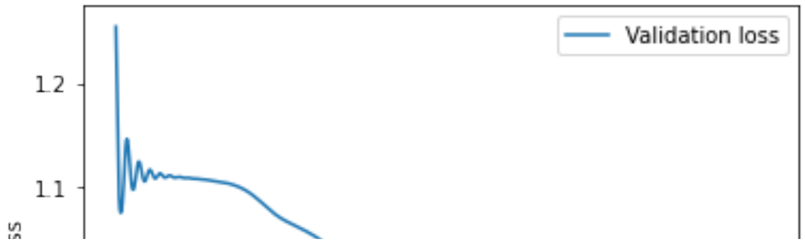

Validation loss
Loss
1.40
1.38
1.36
1.34
1.32
1.30
1.28
0
200
400
600
800
1000
Epochs

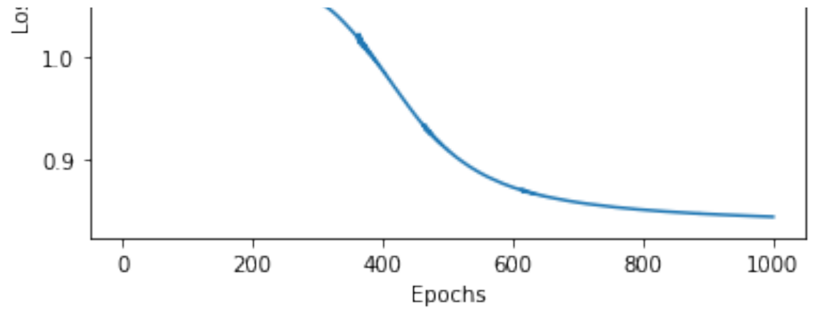
```
class Classifier_softmax(nn.Module):
    def __init__(self, input_dim, hidden_
        super(Classifier_softmax, self)._
        self.fc1 = nn.Linear(input_dim, h
        self.fc2 = nn.Linear(hidden_dim, 

    def forward(self, x):
        out = self.fc1(x)
        out = nn.functional.relu(out)
        out = self.fc2(out)
        out = nn.functional.softmax(out, 
        return out
```

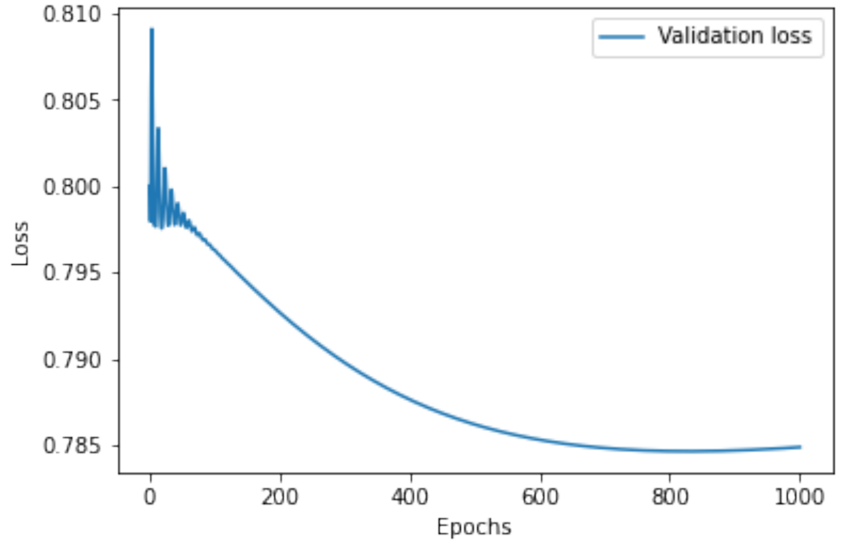
```
wine_classifier = Classifier_softmax(pyto
```

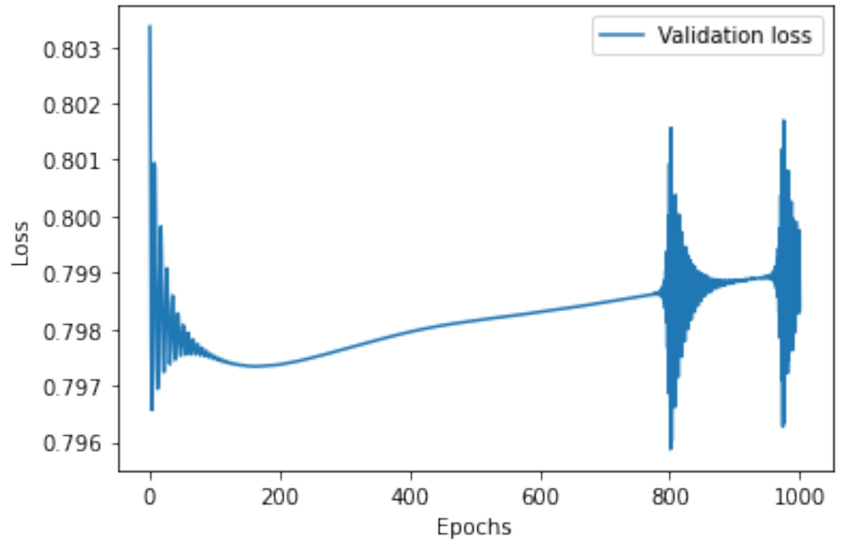
```
train_and_val(wine_classifier,·pytorch_fe
```

```
<ipython-input-71-f0eea764c926>:24: UserWarning: To copy construct from a ten
  Xs = torch.tensor(train_X).float()
<ipython-input-71-f0eea764c926>:25: UserWarning: To copy construct from a ten
  ys = torch.tensor(train_y).long()
Number of epochs with lowest validation: 1000
Test accuracy: 0.6166666666666667
Number of epochs with lowest validation: 828
Test accuracy: 0.6101694915254238
Number of epochs with lowest validation: 801
Test accuracy: 0.7288135593220338
0.7958707809448242
```




Lo
1.0
0.9
0
200
400
600
800
1000
Epochs


Validation loss
Loss
0.810
0.805
0.800
0.795
0.790
0.785
0
200
400
600
800
1000
Epochs


Validation loss
Loss
0.803
0.802
0.801
0.800
0.799
0.798
0.797
0.796
0
200
400
600
800
1000
Epochs

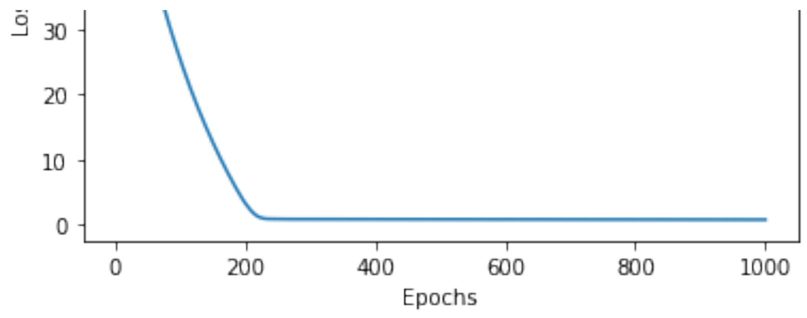
```
class Classifier_Relu(nn.Module):
    def __init__(self, input_dim, hidden_
        super(Classifier_Relu, self).__in
        self.fc1 = nn.Linear(input_dim, h
        self.fc2 = nn.Linear(hidden_dim,
        self.relu = nn.ReLU()

    def forward(self, x):
        out = self.fc1(x)
        out = self.relu(out)
        out = self.fc2(out)
        return out
```

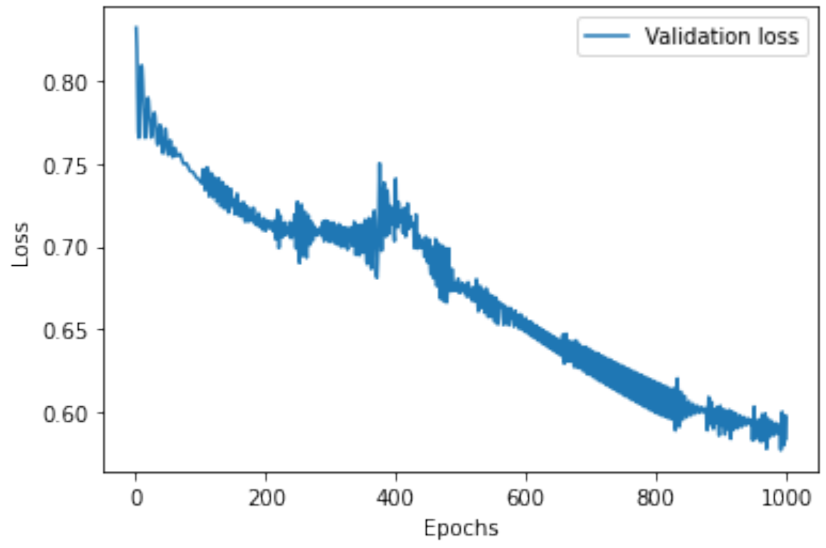
```
wine_classifier_2 = Classifier_Relu(pytor
```

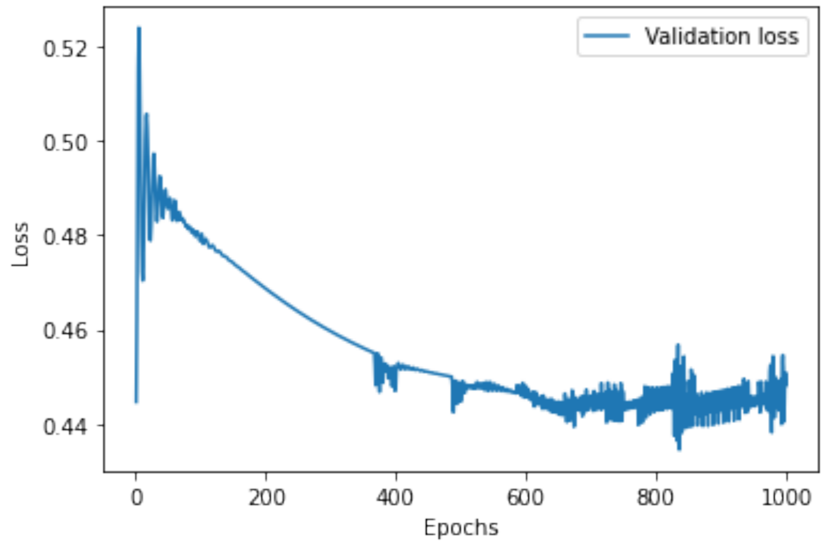
```
train_and_val(wine_classifier_2, pytorch_
```

```
<ipython-input-71-f0eea764c926>:24: UserWarning: To copy construct from a ten
  Xs = torch.tensor(train_X).float()
<ipython-input-71-f0eea764c926>:25: UserWarning: To copy construct from a ten
  ys = torch.tensor(train_y).long()
Number of epochs with lowest validation: 1000
Test accuracy: 0.6166666666666667
Number of epochs with lowest validation: 992
Test accuracy: 0.6949152542372882
Number of epochs with lowest validation: 836
Test accuracy: 0.711864406779661
0.4345228970050812
```

Validation loss
60
50
40


Lo
30
20
10
0
0
200
400
600
800
1000
Epochs


Validation loss
0.80
0.75
0.70
0.65
0.60
Loss
0
200
400
600
800
1000
Epochs


Validation loss
0.52
0.50
0.48
0.46
0.44
Loss
0
200
400
600
800
1000
Epochs