```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
```

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

```
Mounted at /content/drive
```

```python
file_path='/content/drive/My Drive/machine
learning/BankNoteAuthentication.csv'
df=pd.read_csv(file_path)
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 1372,\n  \"fields\":
[\n    {\n      \"column\": \"variance\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 2.842762586278562,\n
\"min\": -7.0421,\n        \"max\": 6.8248,\n
\"num_unique_values\": 1338,\n        \"samples\": [\n          -
2.286,\n          -0.539,\n          0.89512\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"skewness\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
5.869046743695522,\n        \"min\": -13.7731,\n        \"max\":
12.9516,\n        \"num_unique_values\": 1256,\n        \"samples\":
[\n          11.2217,\n          -4.6145,\n          6.1499\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"curtosis\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 4.310030090106595,\n        \"min\": -
5.2861,\n        \"max\": 17.9274,\n        \"num_unique_values\":
1270,\n        \"samples\": [\n          7.8981,\n          9.8208,\n
0.20021\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"entropy\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 2.1010131373596095,\n        \"min\": -8.5482,\n
\"max\": 2.4495,\n        \"num_unique_values\": 1156,\n
\"samples\": [\n          1.0271,\n          0.90946,\n          -
0.026738\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"class\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```python
X = df.iloc[:,:-1]
y = df.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)


mlp = MLPClassifier(hidden_layer_sizes=(10,10),
activation='relu',solver='adam', max_iter=500,
early_stopping=True,validation_fraction=0.1, random_state=42)

mlp.fit(X_train, y_train)
```

```
MLPClassifier(early_stopping=True, hidden_layer_sizes=(10, 10),
max_iter=500,
              random_state=42)
```

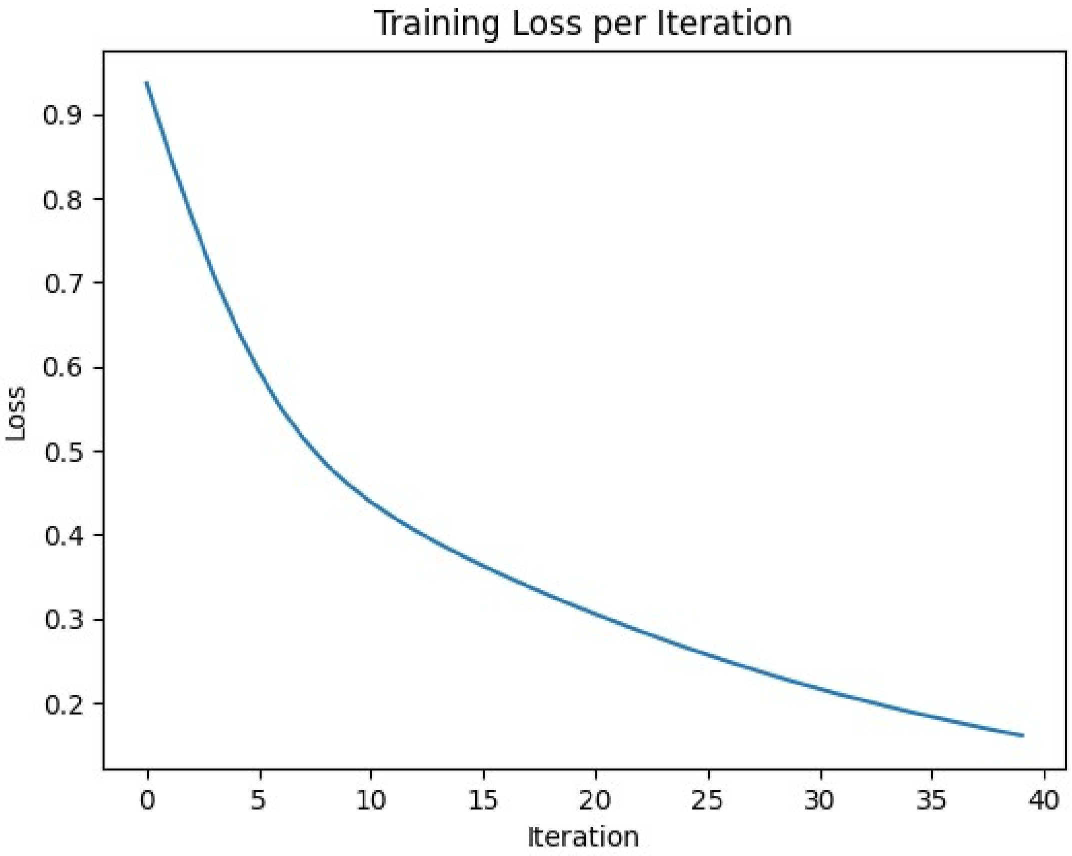
```python
y_pred = mlp.predict(X_test)
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Accuracy:", accuracy_score(y_test, y_pred))
```

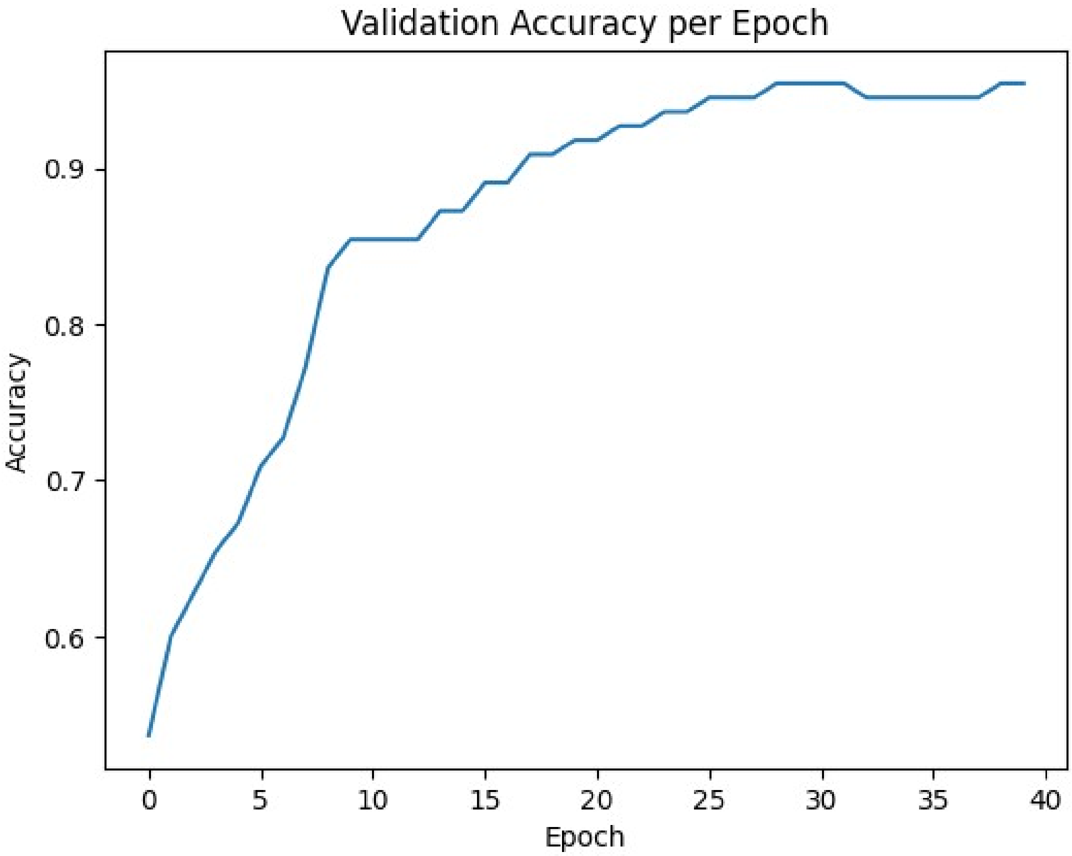
```
Confusion Matrix:
 [[147   1]
 [ 26 101]]
Accuracy: 0.9018181818181819
```

```python
import matplotlib.pyplot as plt

plt.plot(mlp.loss_curve_)
plt.title("Training Loss per Iteration")
plt.xlabel("Iteration")
plt.ylabel("Loss")
plt.show()

if hasattr(mlp, "validation_scores_"):
   plt.plot(mlp.validation_scores_)
   plt.title("Validation Accuracy per Epoch")
   plt.xlabel("Epoch")
   plt.ylabel("Accuracy")
   plt.show()
```


Training Loss per Iteration
Loss
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
Iteration
0
5
10
15
20
25
30
35
40



```
activations = ['tanh', 'logistic', 'identity']

for act in activations:
    model = MLPClassifier(hidden_layer_sizes=(10,10),
activation=act,solver='adam',max_iter=500,early_stopping=True,validati
on_fraction=0.1,random_state=42)


    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"\nActivation: {act}")
    print("Accuracy:", accuracy_score(y_test, y_pred))


Activation: tanh
Accuracy: 0.9272727272727272

Activation: logistic
Accuracy: 0.5381818181818182

Activation: identity
Accuracy: 0.8909090909090909
```