

```
In [8]: import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_mat

data = pd.read_csv("bank.csv", sep=';')
```

```
In [9]: data.head()
```

```
Out[9]:
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	unemployed	married	primary	no	1787	no	no	cellular								
1	33	services	married	secondary	no	4789	yes	yes	cellular								
2	35	management	single	tertiary	no	1350	yes	no	cellular								
3	30	management	married	tertiary	no	1476	yes	yes	unknown								
4	59	blue-collar	married	secondary	no	0	yes	no	unknown								

```
In [11]: data.isnull().sum()
```

```
Out[11]: age          0
job            0
marital        0
education      0
default        0
balance        0
housing        0
loan           0
contact        0
day            0
month          0
duration       0
campaign       0
pdays         0
previous       0
poutcome       0
y              0
dtype: int64
```

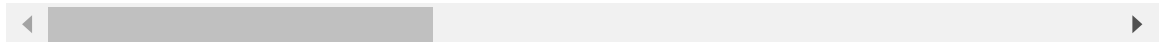
```
In [12]: data_encoded = pd.get_dummies(data)
```

```
In [13]: data_encoded.head()
```

Out[13]:

	age	balance	day	duration	campaign	pdays	previous	job_admin.	job_blue-collar	job
0	30	1787	19	79	1	-1	0	False	False	
1	33	4789	11	220	1	339	4	False	False	
2	35	1350	16	185	1	330	1	False	False	
3	30	1476	3	199	4	-1	0	False	False	
4	59	0	5	226	1	-1	0	False	True	

5 rows × 53 columns



```
In [19]: X = data_encoded.drop(columns=['y_yes', 'y_no'])
y = data_encoded['y_yes']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

```
In [20]: clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
```

```
Out[20]: ▼ DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

```
In [21]: y_pred = clf.predict(X_test)
```

```
In [22]: accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
In [23]: print(f"Accuracy: {accuracy}")
print("Classification Report:")
print(classification_rep)
print("Confusion Matrix:")
print(conf_matrix)
```

Accuracy: 0.8917127071823204

Classification Report:

	precision	recall	f1-score	support
False	0.94	0.93	0.94	807
True	0.50	0.54	0.52	98
accuracy			0.89	905
macro avg	0.72	0.74	0.73	905
weighted avg	0.90	0.89	0.89	905

Confusion Matrix:

```
[[754  53]
 [ 45  53]]
```