Machine Learning Lab

Section 2: Data Preprocessing

Section-1 ..

Python programming

Python for data science/ Machine learning

Section-2 ..

Data preprocessing for Machine learning

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Importing Libraries

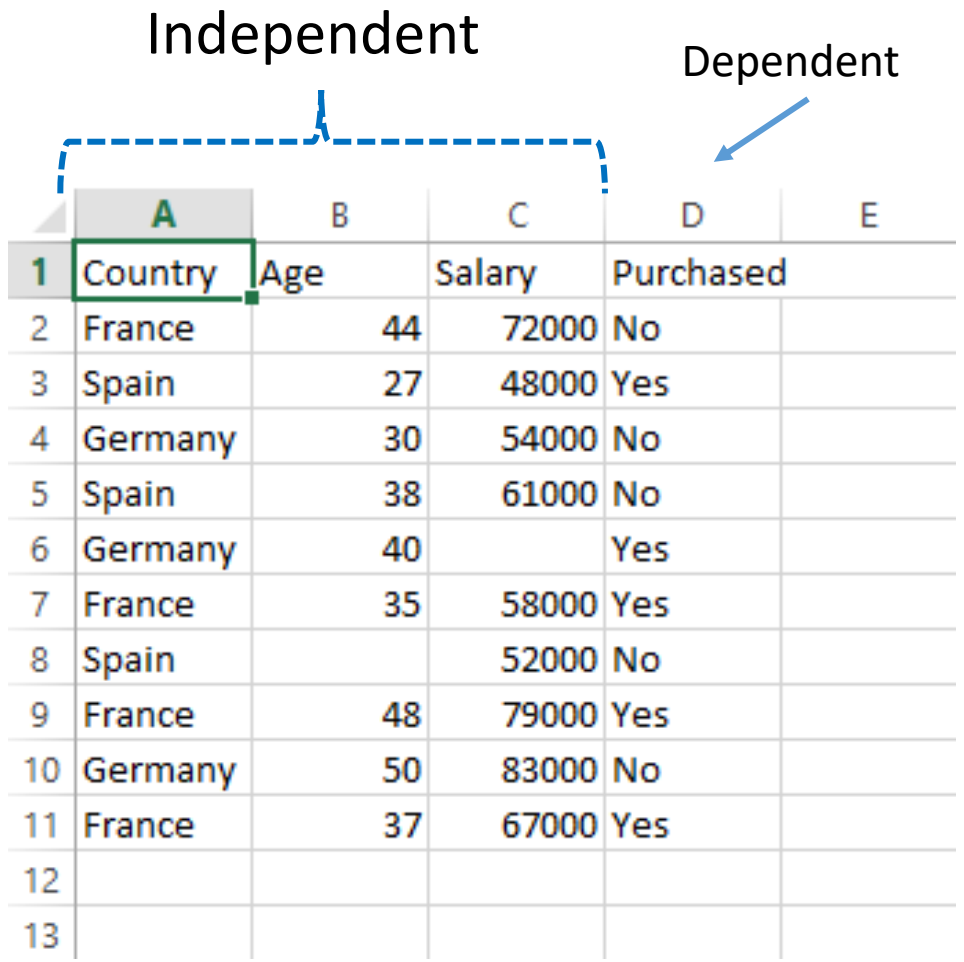
- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Importing Dataset

- Data Set : Data.csv



	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

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

```
import pandas as pd
data = pd.read_csv("drive/My Drive/Colab Notebooks/DataSets/Data.csv")
dataset = data
```

Importing Dataset

- Data Set : Data.csv

- Create matrix of all independent variables

```
dataset
X = dataset.iloc[:, :-1].values
```

Independent

	A	B	C	D
1	Country	Age	Salary	Purchased
2	France	44	72000	No
3	Spain	27	48000	Yes
4	Germany	30	54000	No
5	Spain	38	61000	No
6	Germany	40		Yes
7	France	35	58000	Yes
8	Spain		52000	No
9	France	48	79000	Yes
10	Germany	50	83000	No
11	France	37	67000	Yes
12				

Importing Dataset

- Data Set : Data.csv

- Create matrix of all independent variables

```
dataset
X = dataset.iloc[:, :-1].values
```

Independent

	A	B	C
1	Country	Age	Salary
2	France	44	72000
3	Spain	27	48000
4	Germany	30	54000
5	Spain	38	61000
6	Germany	40	
7	France	35	58000
8	Spain		52000
9	France	48	79000
10	Germany	50	83000
11	France	37	67000
12			

Importing Dataset

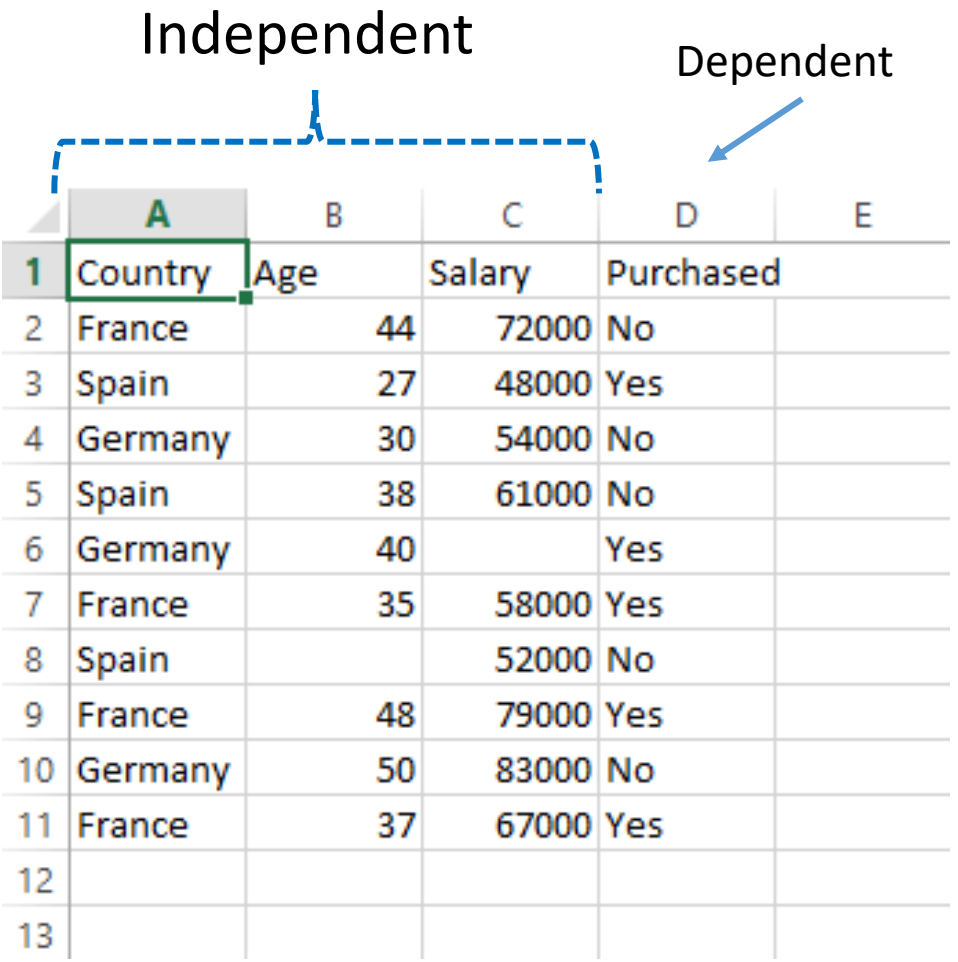
- Data Set : Data.csv

- Create matrix of all independent variables

```
X = dataset.iloc[:, :-1].values
```

- Create matrix of dependent variable

```
y = dataset.iloc[:, -1].values
```



	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

Importing Dataset

- Data Set : Data.csv

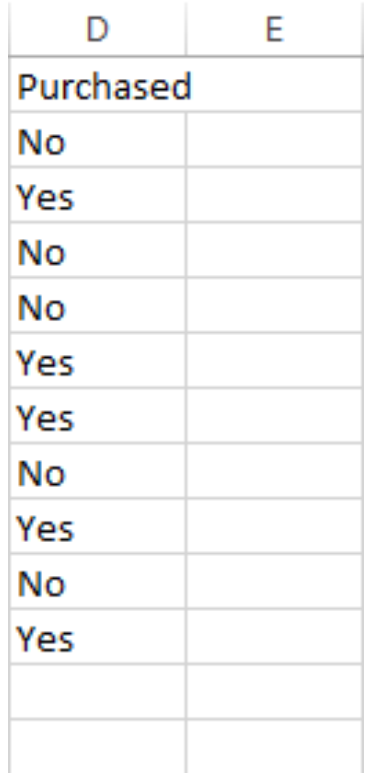
- Create matrix of all independent variables

```
X = dataset.iloc[:, :-1].values
```

- Create matrix of dependent variable

```
y = dataset.iloc[:, 3].values
```

Dependent



D	E
Purchased	
No	
Yes	
No	
No	
Yes	
Yes	
No	
Yes	
No	
Yes	

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

Handling missing values

```
# Taking care of missing data
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan,
 strategy='mean')
imputer = imputer.fit(X[:, 1:3])
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, nan],
       ['France', 35.0, 58000.0],
       ['Spain', nan, 52000.0],
       ['France', 48.0, 79000.0],
       ['Germany', 50.0, 83000.0],
       ['France', 37.0, 67000.0]],
```

sklearn → contains large number of libraries for machine learning applications
impute → library contains methods, classes for preprocessing the data set

Handling missing values

```
# Taking care of missing data
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan,
 strategy='mean')
imputer = imputer.fit(X[:, 1:3])
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, nan],
       ['France', 35.0, 58000.0],
       ['Spain', nan, 52000.0],
       ['France', 48.0, 79000.0],
       ['Germany', 50.0, 83000.0],
       ['France', 37.0, 67000.0]],
```

SimpleImputer → class, allow as to take care of missing values in the data set

strategy = 'mean' – default value (let we specify here)

imputer.fit(X[:, 1:3]) – fit all rows, 1 and 2nd col

imputer.transform(X[:, 1:3]) –transform (replace) the missing values by mean

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Categorical data

	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

```
# Encoding categorical data
# Encoding the Independent Variable
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])
print(X)
```

	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

```
[[0 44.0 72000.0]
[2 27.0 48000.0]
[1 30.0 54000.0]
[2 38.0 61000.0]
[1 40.0 63777.77777777778]
[0 35.0 58000.0]
[2 38.77777777777778 52000.0]
[0 48.0 79000.0]
[1 50.0 83000.0]
[0 37.0 67000.0]]
```

```
# Encoding categorical data
# Encoding the Independent Variable
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])
print(X)
```

Categorical data

	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

```
from sklearn.compose import ColumnTransformer
ct = ColumnTransformer([("Country", OneHotEncoder(),[0])], remainder="passthrough")

X = ct.fit_transform(X)

X.astype(int)
```

Categorical data

```
array([
[ 1, 0, 0, 44, 72000],
[ 0, 0, 1, 27, 48000],
[ 0, 1, 0, 30, 54000],
[ 0, 0, 1, 38, 61000],
[ 0, 1, 0, 40, 63777],
[ 1, 0, 0, 35, 58000],
[ 0, 0, 1, 38, 52000],
[ 1, 0, 0, 48, 79000],
[ 0, 1, 0, 50, 83000],
[ 1, 0, 0, 37, 67000]])
```

	A	B	C	D	E
1	Country	Age	Salary	Purchased	
2	France	44	72000	No	
3	Spain	27	48000	Yes	
4	Germany	30	54000	No	
5	Spain	38	61000	No	
6	Germany	40		Yes	
7	France	35	58000	Yes	
8	Spain		52000	No	
9	France	48	79000	Yes	
10	Germany	50	83000	No	
11	France	37	67000	Yes	
12					
13					

```
# Encoding the Dependent Variable
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
y
```

```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Splitting Training and Test Data

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
X_test.astype(int)
```

```
X_train
```

```
array([
[ 0, 1, 0, 40, 63777],
[ 1, 0, 0, 37, 67000],
[ 0, 0, 1, 27, 48000],
[ 0, 0, 1, 38, 52000],
[ 1, 0, 0, 48, 79000],
[ 0, 0, 1, 38, 61000],
[ 1, 0, 0, 44, 72000],
[ 1, 0, 0, 35, 58000]])
```

```
X_test
```

```
array([
[ 0, 1, 0, 30, 54000],
[ 0, 1, 0, 50, 83000]])
```

Data Preprocessing Steps

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

```
array([
[-1. , 2.64575131, -0.77459667, 0.26306757, 0.12381479],
[ 1. , -0.37796447, -0.77459667, -0.25350148, 0.46175632],
[-1. , -0.37796447, 1.29099445, -1.97539832, -1.53093341],
[-1. , -0.37796447, 1.29099445, 0.05261351, -1.11141978],
[ 1. , -0.37796447, -0.77459667, 1.64058505, 1.7202972 ],
[-1. , -0.37796447, 1.29099445, -0.0813118 , -0.16751412],
[ 1. , -0.37796447, -0.77459667, 0.95182631, 0.98614835],
[ 1. , -0.37796447, -0.77459667, -0.59788085, -0.48214934]])
```

Feature Scaling

```
array([
[-1. , 2.64575131, -0.77459667, -1.45882927, -0.90166297],
[-1. , 2.64575131, -0.77459667, 1.98496442, 2.13981082]])
```

Data Preprocessing

- Get the data set
- Importing the Libraries
- Importing the data set
- Handling Missing data
- Categorical data
- Splitting the data set
- Feature scaling

Thank you