```
import pandas as pd
import seaborn as sns
```

```
dt = pd.read_csv('/content/Iris.csv')
```

```
dt
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

Next steps: Generate code with dt | View recommended plots | New interactive sheet

```
dt.describe()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

```
dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
dt['Species'].unique()
```

```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

```
from sklearn.preprocessing import LabelEncoder
```

```
l = LabelEncoder()
```

```
dt['Species'] = l.fit_transform(dt['Species'])
```

```
dt['Species'].unique()
```

```
array([0, 1, 2])
```

```
dt
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	0
1	2	4.9	3.0	1.4	0.2	0
2	3	4.7	3.2	1.3	0.2	0
3	4	4.6	3.1	1.5	0.2	0
4	5	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	2
146	147	6.3	2.5	5.0	1.9	2
147	148	6.5	3.0	5.2	2.0	2
148	149	6.2	3.4	5.4	2.3	2
149	150	5.9	3.0	5.1	1.8	2

150 rows × 6 columns

Next steps: Generate code with dt | View recommended plots | New interactive sheet

```
from sklearn.model_selection import train_test_split
```

```
x = dt.drop(['Species','Id'],axis = 1)
y = dt['Species']
```

```
x
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

Next steps: Generate code with x | View recommended plots | New interactive sheet

y

	Species
0	0
1	0
2	0
3	0
4	0
...	...
145	2
146	2
147	2
148	2
149	2

150 rows × 1 columns

dtype: int64

```
dt.corr()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
Id	1.000000	0.716676	-0.397729	0.882747	0.899759	0.942830
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954	0.782561
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544	-0.419446
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757	0.949043
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000	0.956464
Species	0.942830	0.782561	-0.419446	0.949043	0.956464	1.000000

```
xtrain, xtest, ytrain, ytest = train_test_split(x,y, test_size=0.25)
```

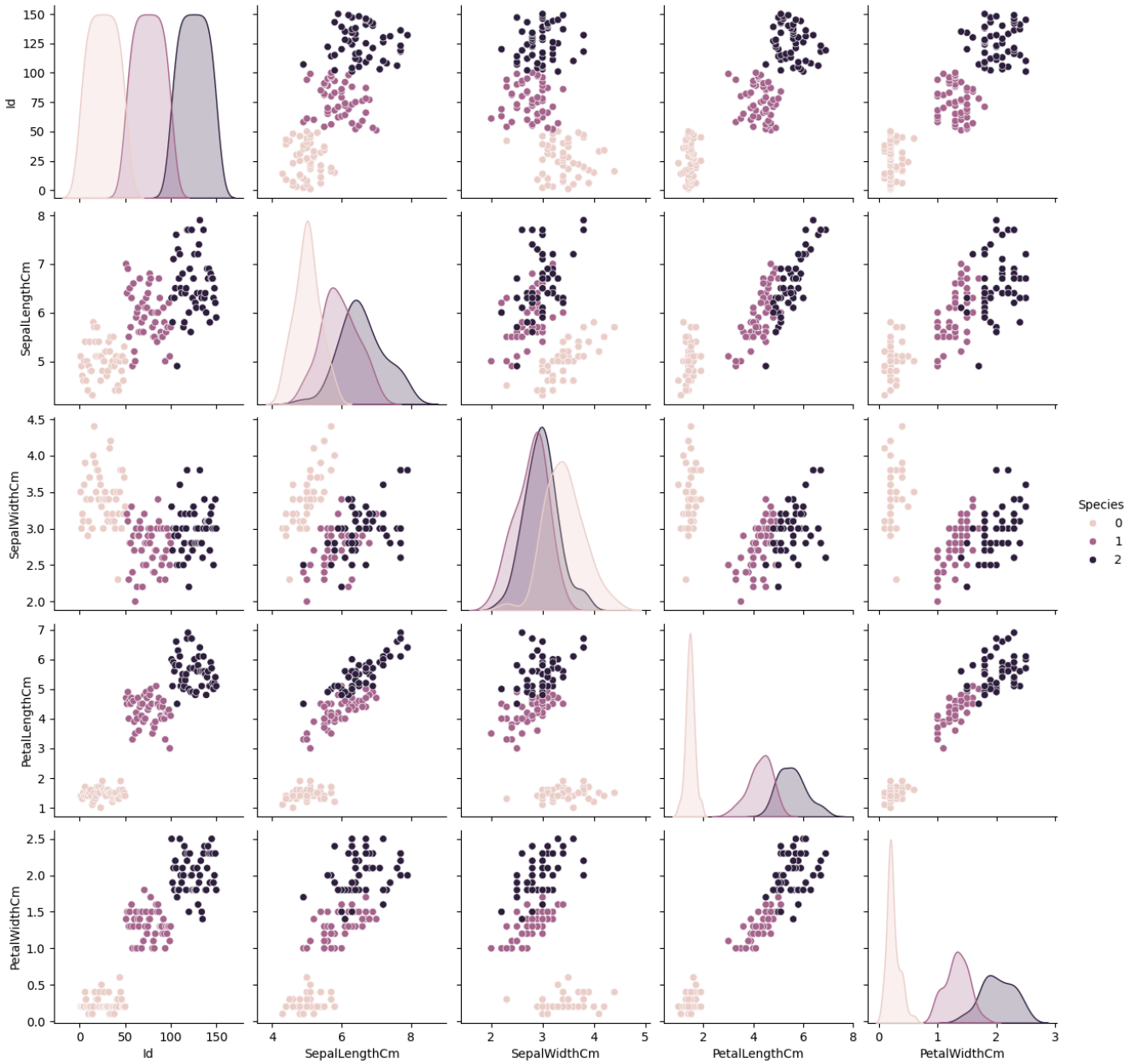
```
xtest
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
101	5.8	2.7	5.1	1.9
34	4.9	3.1	1.5	0.1
37	4.9	3.1	1.5	0.1
123	6.3	2.7	4.9	1.8
21	5.1	3.7	1.5	0.4
87	6.3	2.3	4.4	1.3
61	5.9	3.0	4.2	1.5
106	4.9	2.5	4.5	1.7
59	5.2	2.7	3.9	1.4
124	6.7	3.3	5.7	2.1
107	7.3	2.9	6.3	1.8
1	4.9	3.0	1.4	0.2
60	5.0	2.0	3.5	1.0
18	5.7	3.8	1.7	0.3
135	7.7	3.0	6.1	2.3
125	7.2	3.2	6.0	1.8
78	6.0	2.9	4.5	1.5
132	6.4	2.8	5.6	2.2
15	5.7	4.4	1.5	0.4
83	6.0	2.7	5.1	1.6
42	4.4	3.2	1.3	0.2
96	5.7	2.9	4.2	1.3
46	5.1	3.8	1.6	0.2
23	5.1	3.3	1.7	0.5
56	6.3	3.3	4.7	1.6
136	6.3	3.4	5.6	2.4
116	6.5	3.0	5.5	1.8
30	4.8	3.1	1.6	0.2
0	5.1	3.5	1.4	0.2
143	6.8	3.2	5.9	2.3
92	5.8	2.6	4.0	1.2
68	6.2	2.2	4.5	1.5
43	5.0	3.5	1.6	0.6
12	4.8	3.0	1.4	0.1
31	5.4	3.4	1.5	0.4
84	5.4	3.0	4.5	1.5
121	5.6	2.8	4.9	2.0
19	5.1	3.8	1.5	0.3

Next steps: Generate code with `xtest` View recommended plots New interactive sheet

```
sns.pairplot(dt, hue='Species')
```

```
<seaborn.axisgrid.PairGrid at 0x7cd645ca4590>
```




```
from sklearn.linear_model import LogisticRegression
```

```
lr = LogisticRegression()
```

```
lr.fit(xtrain,ytrain)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
ypred = lr.predict(xtest)
```

```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(ytest,ypred)
```

```
0.9473684210526315
```

```
import pickle

pickle.dump(lr,open('iris.pkl','wb'))
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.