```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
import seaborn as sns
import warnings
warnings.simplefilter("ignore")
```

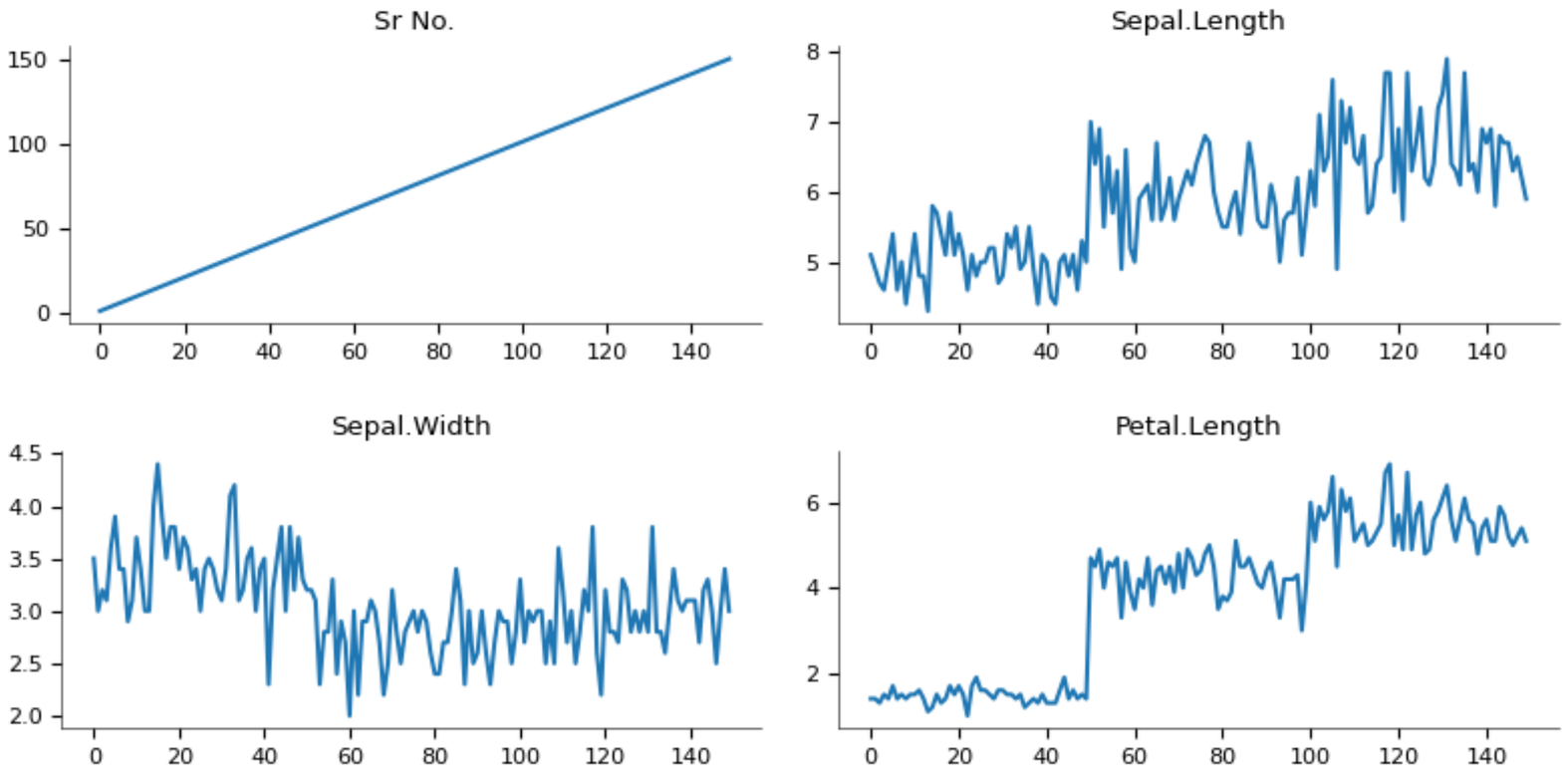
```
data=pd.read_csv("/content/iris.csv")
```

```
data
```

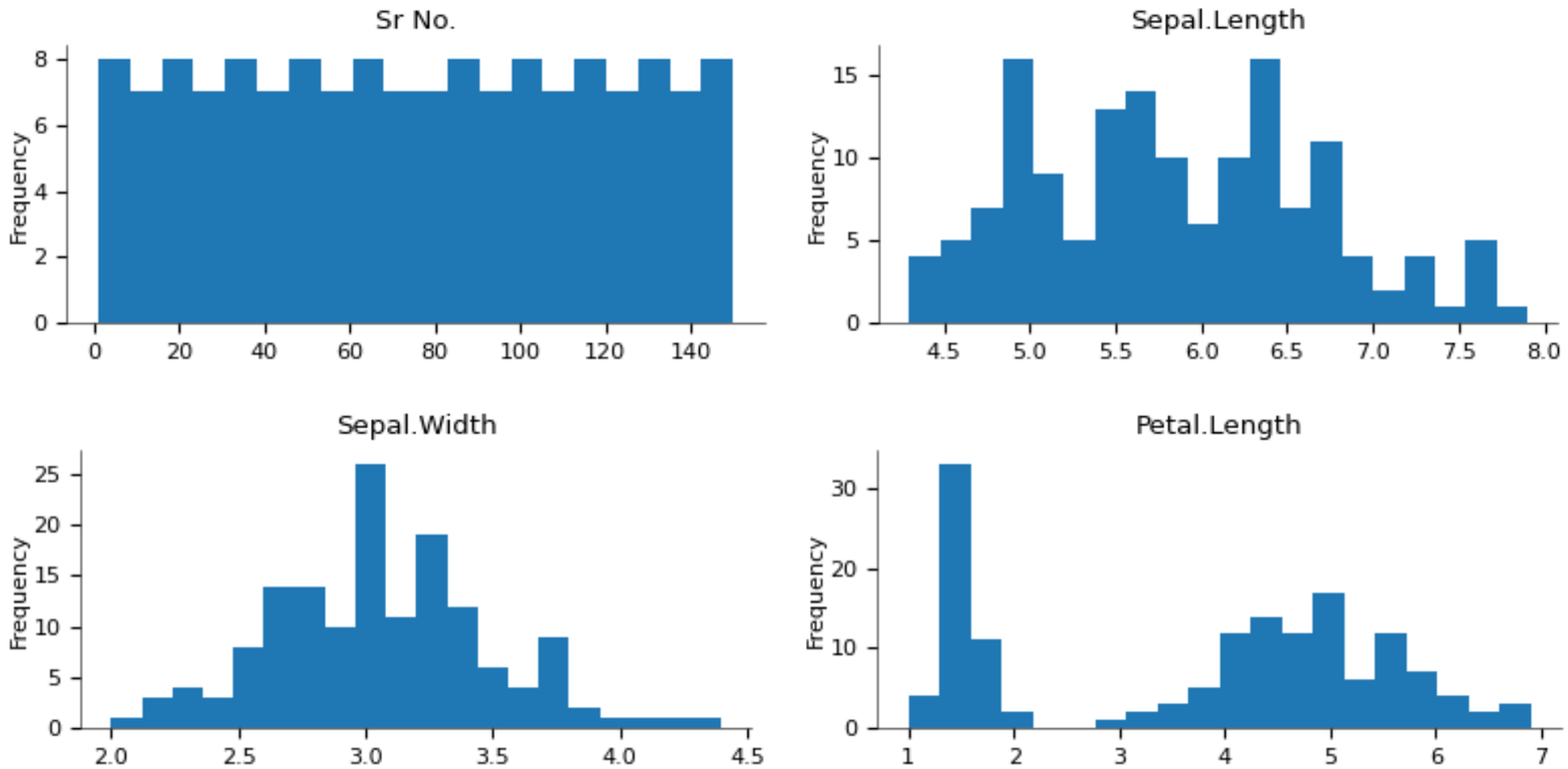
	Sr No.	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	Unnamed: 6
0	1	5.1	3.5	1.4	0.2	setosa	NaN
1	2	4.9	3.0	1.4	0.2	setosa	NaN
2	3	4.7	3.2	1.3	0.2	setosa	NaN
3	4	4.6	3.1	1.5	0.2	setosa	NaN
4	5	5.0	3.6	1.4	0.2	setosa	NaN
...	...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	setosa	NaN
146	147	6.3	2.5	5.0	1.9	setosa	NaN
147	148	6.5	3.0	5.2	2.0	setosa	NaN
148	149	6.2	3.4	5.4	2.3	setosa	NaN
149	150	5.9	3.0	5.1	1.8	setosa	NaN

150 rows × 7 columns

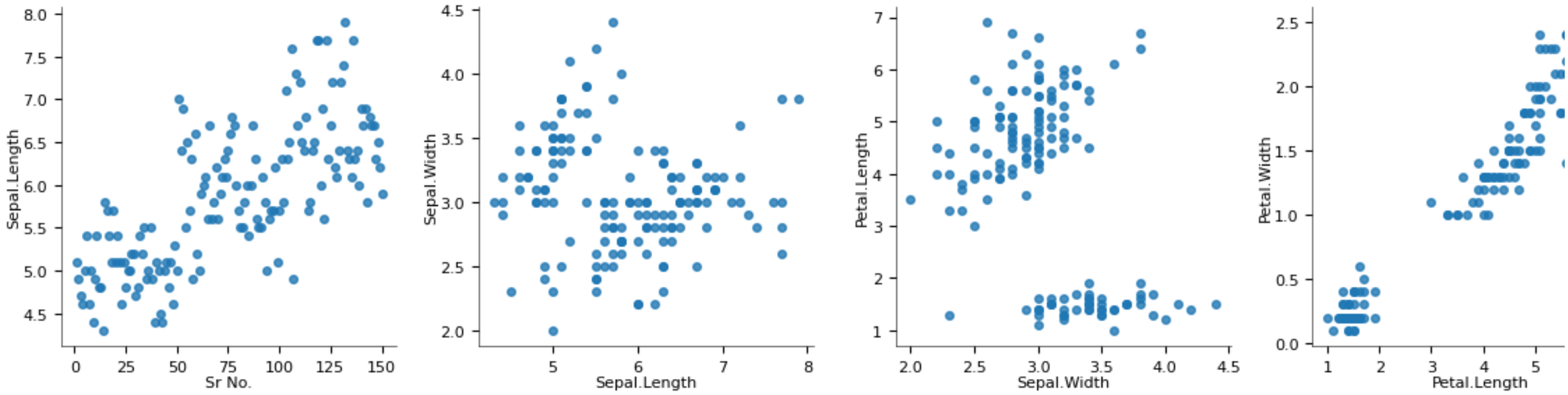
Values



Distributions



2-d distributions



Time series

```
data.isnull().sum()
```

```
Sr No.           0
Sepal.Length     0
Sepal.Width      0
Petal.Length     0
Petal.Width      0
Species          0
Unnamed: 6     150
dtype: int64
```

```
data.columns
```

```
Index(['Sr No.', 'Sepal.Length', 'Sepal.Width', 'Petal.Length', 'Petal.Width',
       'Species', 'Unnamed: 6'],
      dtype='object')
```

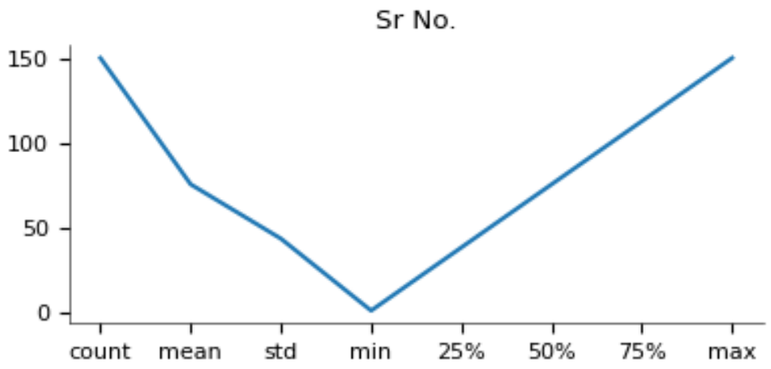
```
top_data=data.head(6)
print(top_data)
```

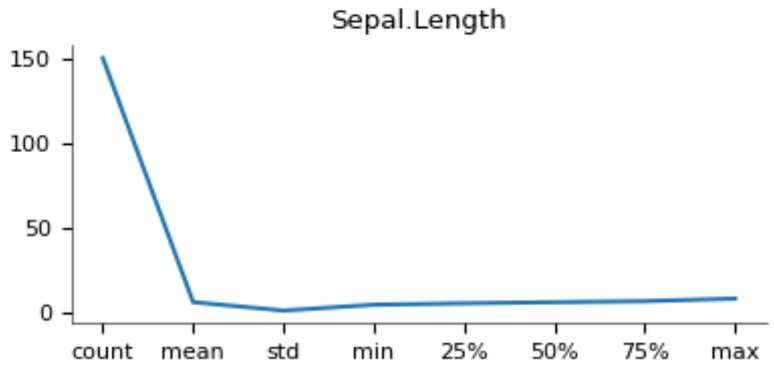
```
   Sepal.Length  Sepal.Width  Petal.Length  Petal.Width Species  Unnamed: 6
0           5.1          3.5           1.4          0.2  setosa         NaN
1           4.9          3.0           1.4          0.2  setosa         NaN
2           4.7          3.2           1.3          0.2  setosa         NaN
3           4.6          3.1           1.5          0.2  setosa         NaN
4           5.0          3.6           1.4          0.2  setosa         NaN
5           5.4          3.9           1.7          0.4  setosa         NaN
```

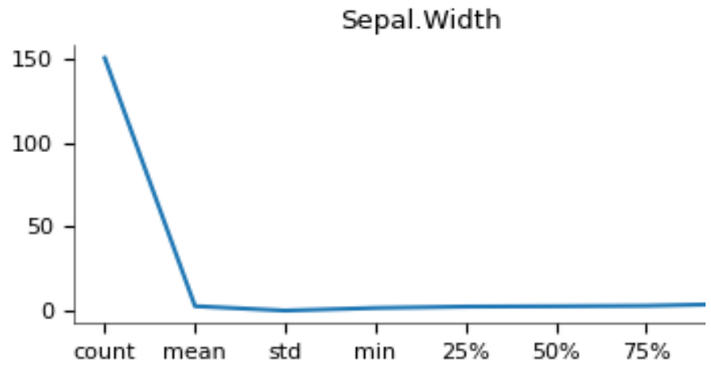
```
data.describe()
```

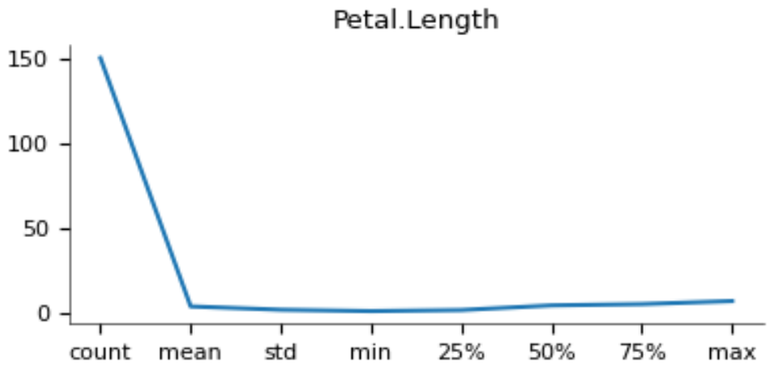
	Sr No.	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Unnamed: 6
count	150.000000	150.000000	150.000000	150.000000	150.000000	0.0
mean	75.500000	5.843333	3.057333	3.758000	1.199333	NaN
std	43.445368	0.828066	0.435866	1.765298	0.762238	NaN
min	1.000000	4.300000	2.000000	1.000000	0.100000	NaN
25%	38.250000	5.100000	2.800000	1.600000	0.300000	NaN
50%	75.500000	5.800000	3.000000	4.350000	1.300000	NaN
75%	112.750000	6.400000	3.300000	5.100000	1.800000	NaN
max	150.000000	7.900000	4.400000	6.900000	2.500000	NaN

Values

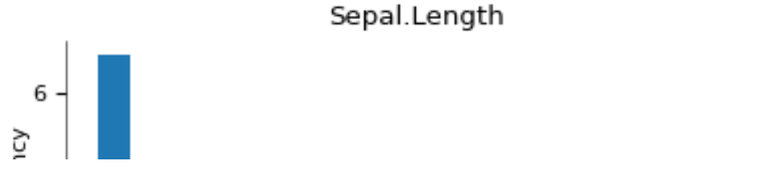


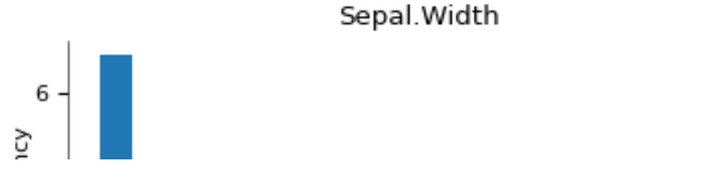


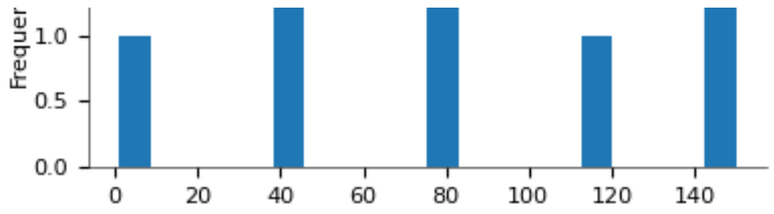


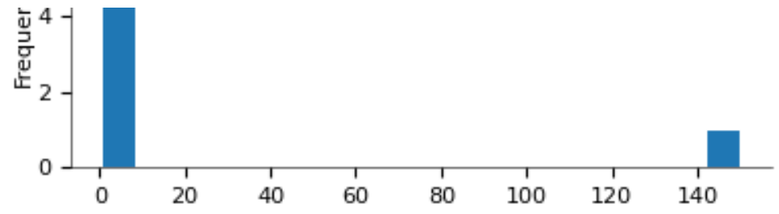


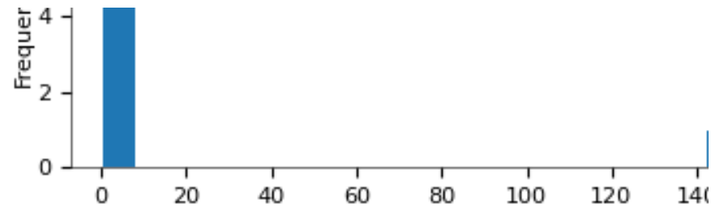
Distributions

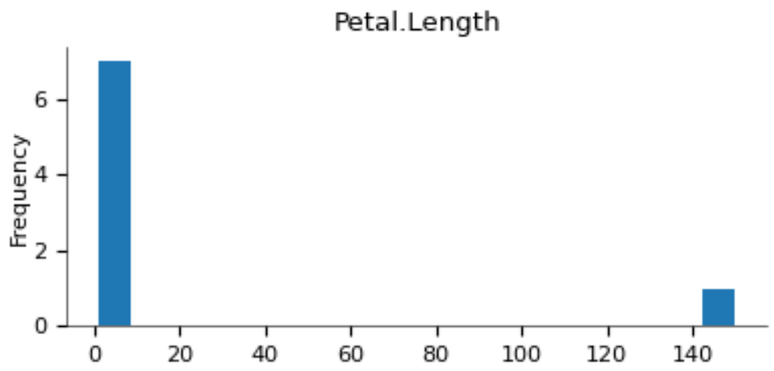




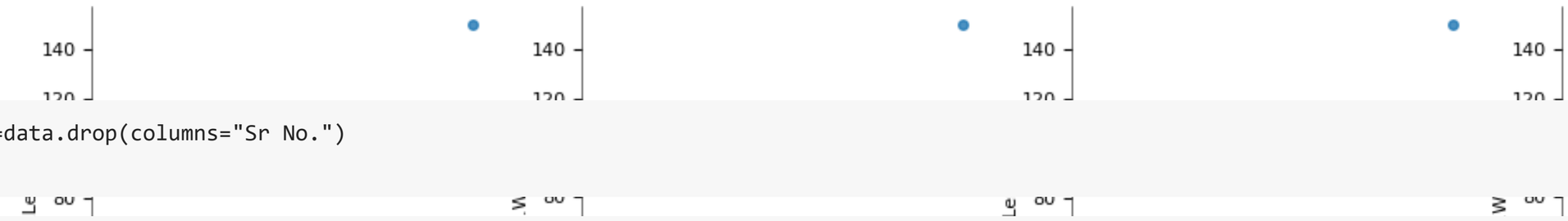








2-d distributions




```
data=data.drop(columns="Sr No.")
```

```
x=data.iloc[:,:4]
y=data.iloc[:,4]
```

```
print(x)
print(y)
```

```
     Sr No.  Sepal.Length  Sepal.Width  Petal.Length
0         1           5.1          3.5           1.4
1         2           4.9          3.0           1.4
2         3           4.7          3.2           1.3
3         4           4.6          3.1           1.5
4         5           5.0          3.6           1.4
..      ...           ...          ...           ...
145     146           6.7          3.0           5.2
146     147           6.3          2.5           5.0
147     148           6.5          3.0           5.2
148     149           6.2          3.4           5.4
149     150           5.9          3.0           5.1
```

```
[150 rows x 4 columns]
0      0.2
1      0.2
2      0.2
3      0.2
4      0.2
      ...
145    2.3
146    1.9
147    2.0
148    2.3
149    1.8
Name: Petal.Width, Length: 150, dtype: float64
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,random_state=0)
```

```
from sklearn.linear_model import LogisticRegression
prateek_iris_act2=LogisticRegression()
```

```
# Create an instance of the DecisionTreeClassifier
prateek_iris_act2 = DecisionTreeClassifier()

# Fit the model to the training data
prateek_iris_act2.fit(x_train, y_train)
```

▾ DecisionTreeClassifier

DecisionTreeClassifier()

```
y_prediction=prateek_iris_act2.predict(x_test)
```

```
y_prediction
```

```
array(['setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
```

```
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa'], dtype=object)
```

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
confusion_matrix(y_test,y_prediction)
```

```
array([[38]])
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
# Import your specific model

# Create and train the model
model = DecisionTreeClassifier()
model.fit(x_train, y_train)

# Make predictions
y_pred = model.predict(x_test)

accuracy = accuracy_score(y_test, y_pred) * 100
print(f"Accuracy: {accuracy}")

classification_rep = classification_report(y_test, y_pred)
print(f"Classification Report:\n{classification_rep}")

conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d')
plt.xlabel('--- Predicted ---')
plt.ylabel('--- True ---')
plt.show()
```

```
Accuracy: 100.0
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        38

    accuracy                           1.00        38
   macro avg       1.00      1.00      1.00        38
weighted avg       1.00      1.00      1.00        38
```