Importing Libraries

```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
import warnings
warnings.filterwarnings('ignore')
```

Data Reading & Manipulating

```
df = pd.read_csv('/content/Task.csv')
df
```

	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

```
df.drop(['Id'], axis=1, inplace=True)
```

```
df.head()
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

```
df.describe()
```

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

```
df['Species'].value_counts()
```

```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

```
df.isna().sum()
```

```
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

```
df.duplicated().sum()
```

```
3
```

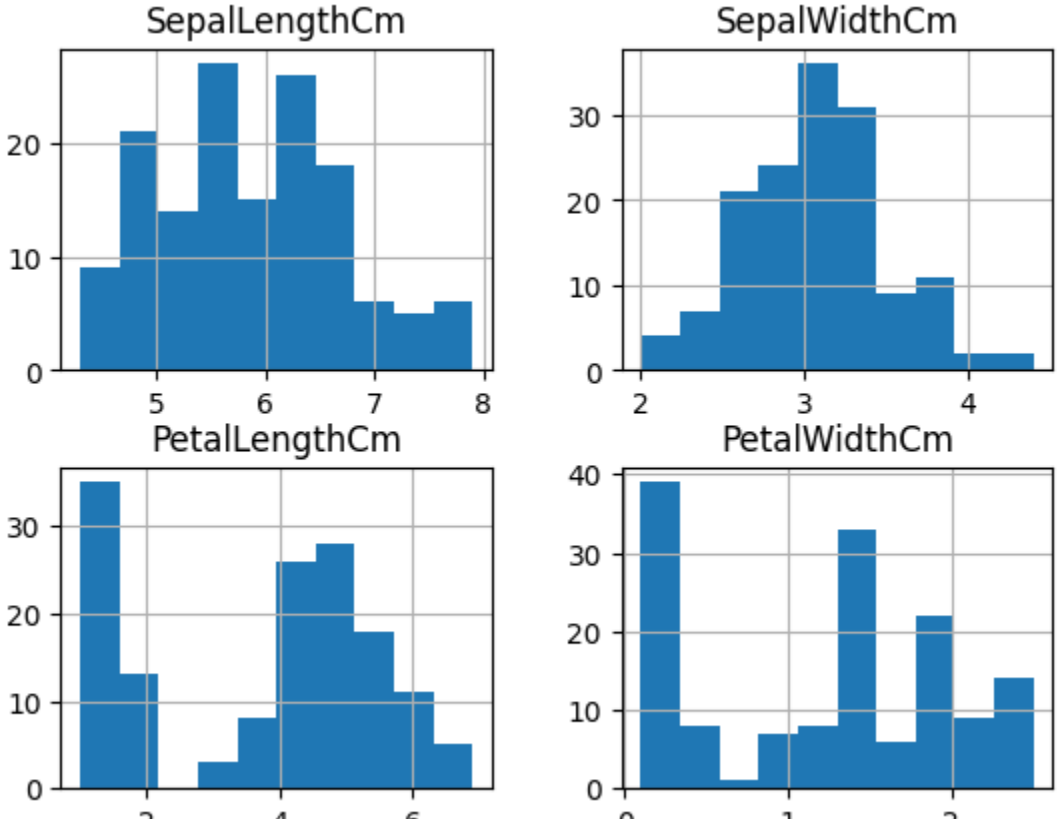
```
df = df.drop_duplicates()
```

```
df.duplicated().sum()
```

```
0
```

Data Visaulization

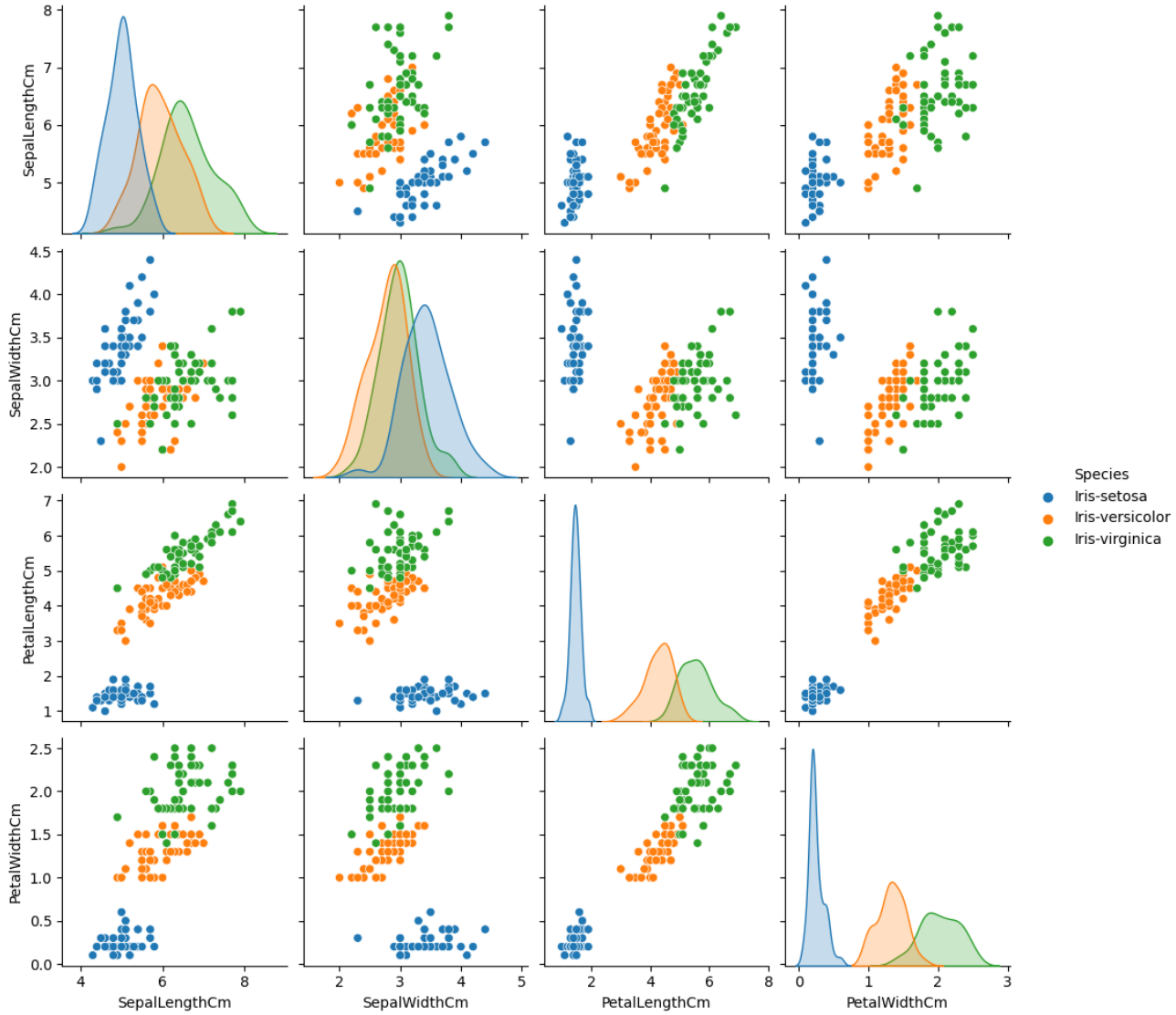
```
df.hist();
```



2 4 6 0 1 2

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

```
<seaborn.axisgrid.PairGrid at 0x783c6397f8e0>
```

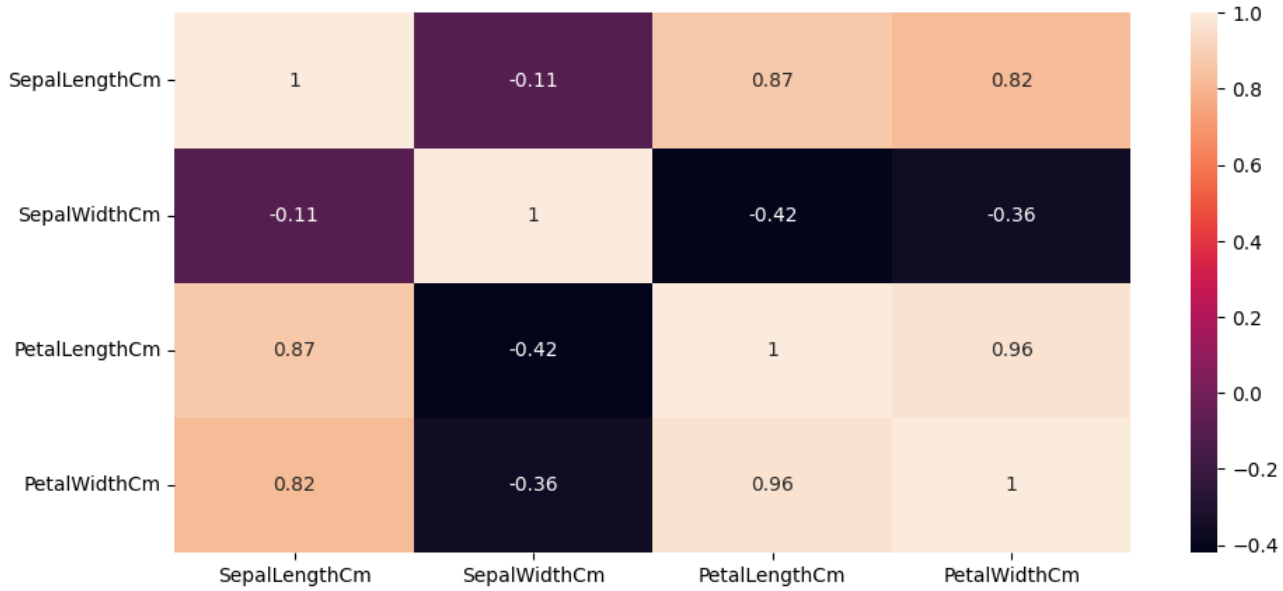


```
corr = df.corr()
corr
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
SepalLengthCm	1.000000	-0.109321	0.871305	0.817058
SepalWidthCm	-0.109321	1.000000	-0.421057	-0.356376
PetalLengthCm	0.871305	-0.421057	1.000000	0.961883
PetalWidthCm	0.817058	-0.356376	0.961883	1.000000

```
fig, ax = plt.subplots(figsize = (11,5))
sns.heatmap(corr, annot=True, ax=ax)
```

```
<Axes: >
```



Machine Learning Algorithmn

```
from sklearn.model_selection import train_test_split

X = df.drop(['Species'], axis=1)
y = df['Species']

x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.35, random_state=3
```

1) K - Means Model

```
#KNN Model training
from sklearn.neighbors import KNeighborsClassifier
model_1 = KNeighborsClassifier()
```

```
#model Training
model_1.fit(x_train, y_train)
```

▾ KNeighborsClassifier
KNeighborsClassifier()

```
#print matrix to get performance
print('Accuracy: ',model_1.score(x_test, y_test) * 100)

prediction_1 = model_1.predict(x_test)
print(prediction_1)

Actual_1 = y_test
print(Actual_1)
```

```
Accuracy:  98.07692307692307
['Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica'
```

```
 Iris-virginica   Iris-setosa   Iris-virginica   Iris-setosa
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica']
5          Iris-setosa
46         Iris-setosa
47         Iris-setosa
104     Iris-virginica
87     Iris-versicolor
148     Iris-virginica
39         Iris-setosa
69     Iris-versicolor
130     Iris-virginica
103     Iris-virginica
134     Iris-virginica
28         Iris-setosa
4          Iris-setosa
143     Iris-virginica
22         Iris-setosa
126     Iris-virginica
52     Iris-versicolor
89     Iris-versicolor
53     Iris-versicolor
135     Iris-virginica
35         Iris-setosa
8          Iris-setosa
91     Iris-versicolor
63     Iris-versicolor
140     Iris-virginica
73     Iris-versicolor
16         Iris-setosa
43         Iris-setosa
145     Iris-virginica
114     Iris-virginica
41         Iris-setosa
110     Iris-virginica
74     Iris-versicolor
138     Iris-virginica
78     Iris-versicolor
27         Iris-setosa
108     Iris-virginica
70     Iris-versicolor
120     Iris-virginica
127     Iris-virginica
132     Iris-virginica
1          Iris-setosa
72     Iris-versicolor
6          Iris-setosa
```

```
# Adetailed classification Report Linear RegressionM
from sklearn.metrics import classification_report
print(classification_report(Actual_1, prediction_1))
```

```
                    precision    recall  f1-score   support

        Iris-setosa       1.00      1.00      1.00        18
```

```
     Iris-versicolor       1.00      0.92      0.96        13
      Iris-virginica       0.95      1.00      0.98        21

            accuracy                           0.98        52
           macro avg       0.98      0.97      0.98        52
        weighted avg       0.98      0.98      0.98        52
```

2) Decision Tree Model

```
#Decision Tree Model
from sklearn.tree import DecisionTreeClassifier
model_2 = DecisionTreeClassifier()


#Model Training
model_2.fit(x_train, y_train)
```

▾ DecisionTreeClassifier
DecisionTreeClassifier()

```
#print matrix to get performance
print('Accuracy: ',model_2.score(x_test, y_test) * 100)

prediction_2 = model_2.predict(x_test)
print(prediction_2)

Actual_2 = y_test
print(Actual_2)
```

```
Accuracy:  94.23076923076923
['Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-setosa' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica']
5          Iris-setosa
46         Iris-setosa
47         Iris-setosa
104     Iris-virginica
87     Iris-versicolor
```

```
148     Iris-virginica
39         Iris-setosa
69     Iris-versicolor
130     Iris-virginica
103     Iris-virginica
134     Iris-virginica
28         Iris-setosa
4          Iris-setosa
143     Iris-virginica
22         Iris-setosa
126     Iris-virginica
52     Iris-versicolor
89     Iris-versicolor
53     Iris-versicolor
135     Iris-virginica
35         Iris-setosa
8          Iris-setosa
91     Iris-versicolor
63     Iris-versicolor
140     Iris-virginica
73     Iris-versicolor
16         Iris-setosa
43         Iris-setosa
145     Iris-virginica
114     Iris-virginica
41         Iris-setosa
110     Iris-virginica
74     Iris-versicolor
138     Iris-virginica
78     Iris-versicolor
27         Iris-setosa
108     Iris-virginica
70     Iris-versicolor
120     Iris-virginica
127     Iris-virginica
132     Iris-virginica
1          Iris-setosa
72     Iris-versicolor
6          Iris-setosa
```

```
# Adetailed classification Report Linear RegressionM
from sklearn.metrics import classification_report
print(classification_report(Actual_2, prediction_2))
```

```
                  precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        18
 Iris-versicolor       1.00      0.77      0.87        13
  Iris-virginica       0.88      1.00      0.93        21

        accuracy                           0.94        52
       macro avg       0.96      0.92      0.93        52
    weighted avg       0.95      0.94      0.94        52
```

3) Support Vector Machine Model

```
#Support Vector Machine Model
from sklearn.svm import SVC
model_3 = SVC()


#Model Training
model_3.fit(x_train, y_train)
```

▾ SVC
SVC()

```
#print matrix to get performance
print('Accuracy: ',model_3.score(x_test, y_test) * 100)

prediction_3 = model_3.predict(x_test)
print(prediction_3)

Actual_3 = y_test
print(Actual_3)
```

```
Accuracy:  94.23076923076923
['Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-setosa' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica']
5          Iris-setosa
46         Iris-setosa
47         Iris-setosa
104     Iris-virginica
87     Iris-versicolor
148     Iris-virginica
39         Iris-setosa
69     Iris-versicolor
130     Iris-virginica
103     Iris-virginica
134     Iris-virginica
28         Iris-setosa
```

```
4          Iris-setosa
143     Iris-virginica
22         Iris-setosa
126     Iris-virginica
52     Iris-versicolor
89     Iris-versicolor
53     Iris-versicolor
135     Iris-virginica
35         Iris-setosa
8          Iris-setosa
91     Iris-versicolor
63     Iris-versicolor
140     Iris-virginica
73     Iris-versicolor
16         Iris-setosa
43         Iris-setosa
145     Iris-virginica
114     Iris-virginica
41         Iris-setosa
110     Iris-virginica
74     Iris-versicolor
138     Iris-virginica
78     Iris-versicolor
27         Iris-setosa
108     Iris-virginica
70     Iris-versicolor
120     Iris-virginica
127     Iris-virginica
132     Iris-virginica
1          Iris-setosa
72     Iris-versicolor
6          Iris-setosa
```

```
# Adetailed classification Report VSM Model
from sklearn.metrics import classification_report
print(classification_report(Actual_3, prediction_3))
```

```
                    precision    recall  f1-score   support

         Iris-setosa       1.00      1.00      1.00        18
     Iris-versicolor       1.00      0.77      0.87        13
      Iris-virginica       0.88      1.00      0.93        21

            accuracy                           0.94        52
           macro avg       0.96      0.92      0.93        52
        weighted avg       0.95      0.94      0.94        52
```

4) Random Forest Model

```
#Random Forest Machine Model
```

```
from sklearn.ensemble import RandomForestClassifier
model_4 = RandomForestClassifier()


#Model Training
model_4.fit(x_train, y_train)
```

▾ RandomForestClassifier

RandomForestClassifier()

```
#print matrix to get performance
print('Accuracy: ',model_4.score(x_test, y_test) * 100)

prediction_4 = model_4.predict(x_test)
print(prediction_4)

Actual_4 = y_test
print(Actual_4)
```

```
Accuracy:  96.15384615384616
['Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-setosa' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica']
5          Iris-setosa
46         Iris-setosa
47         Iris-setosa
104     Iris-virginica
87     Iris-versicolor
148     Iris-virginica
39         Iris-setosa
69     Iris-versicolor
130     Iris-virginica
103     Iris-virginica
134     Iris-virginica
28         Iris-setosa
4          Iris-setosa
143     Iris-virginica
22         Iris-setosa
126     Iris-virginica
52     Iris-versicolor
89     Iris-versicolor
53     Iris-versicolor
```

```
135     Iris-virginica
35         Iris-setosa
8          Iris-setosa
91     Iris-versicolor
63     Iris-versicolor
140     Iris-virginica
73     Iris-versicolor
16         Iris-setosa
43         Iris-setosa
145     Iris-virginica
114     Iris-virginica
41         Iris-setosa
110     Iris-virginica
74     Iris-versicolor
138     Iris-virginica
78     Iris-versicolor
27         Iris-setosa
108     Iris-virginica
70     Iris-versicolor
120     Iris-virginica
127     Iris-virginica
132     Iris-virginica
1          Iris-setosa
72     Iris-versicolor
6          Iris-setosa
```

```
# Adetailed classification Report VSM Model
from sklearn.metrics import classification_report
print(classification_report(Actual_4, prediction_4))
```

```
                    precision    recall  f1-score   support

         Iris-setosa       1.00      1.00      1.00        18
     Iris-versicolor       1.00      0.85      0.92        13
      Iris-virginica       0.91      1.00      0.95        21

            accuracy                           0.96        52
           macro avg       0.97      0.95      0.96        52
        weighted avg       0.96      0.96      0.96        52
```

Results

```
import plotly.graph_objects as pgo


Result = pd.DataFrame({'Models':['KNN', 'DT', 'SVM', 'RF'],
             'Accuracy':[model_1.score(x_test, y_test) * 100, model_2.score(x_test, y_test
                         model_3.score(x_test, y_test) * 100, model_4.score(x_test, y_test)

Result
```

	Models	Accuracy
0	KNN	98.076923
1	DT	94.230769
2	SVM	94.230769
3	RF	96.153846

```
Model = Result['Models']

Accuracy = Result['Accuracy']

fig = pgo.Figure()
fig.add_trace(pgo.Bar(x = Model, y = Accuracy, name = 'Accuracy', width = 0.3))

fig.update_layout(title = 'Accuracy score of performed Models in %.')

fig.show()
```

Model Testing

```
X_new = np.array([[9, 8, 3, 4], [15, 56, 16, 23], [784, 235, 498, 123]])
#Prediction of the Species from the input vector
prediction = model_1.predict(X_new)
print("Prediction of species: {}".format(prediction))
```

```
Prediction of species: ['Iris-virginica' 'Iris-virginica' 'Iris-virginica']
```

```
X_new = np.array([[0.9, 0.8, 0.3, 0.4], [0.49, 0.56, 0.16, 0.23], [0.357, 0.236, 0.498, 0.
#Prediction of the Species from the input vector
prediction = model_1.predict(X_new)
print("Prediction of species: {}".format(prediction))
```

```
Prediction of species: ['Iris-setosa' 'Iris-setosa' 'Iris-setosa']
```