Priyasri Sankaran 5/24/2025

Dataset Source:Kaggle.com Dataset Iris dataset is a very know dataset used is a good example of classification problem. There are 50 sample on each species. 150 samples. 3 different Species(Iris-setosa, Iris-versicolor, Iris-virginica). Features are SepalLength(Cm), SepalWidth(Cm),PetalLength(Cm), Species name. We will not be using the Id, we will drop them.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_s
import xgboost as xgb
import warnings
warnings.filterwarnings("ignore")
```

```
print(pd.__version__, np.__version__)
```

```
2.2.3 1.26.4
```

Loading the Dataset

```
df= pd.read_csv("iris.csv")
```

```
#Display the first few rows
df.head()
```

| | 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 |

```
print(df.columns)
```

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

```
#Checking dataset shape(rows and columns)
df.shape
```

```
(150, 6)
```

```
df.describe(include="all")
```

| | Id | SepalLengthCm | SepalWidthCm | PetalLengthCm | PetalWidthCm | Species |
|---|---|---|---|---|---|---|
| count | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150 |
| unique | NaN | NaN | NaN | NaN | NaN | 3 |
| top | NaN | NaN | NaN | NaN | NaN | Iris-setosa |
| freq | NaN | NaN | NaN | NaN | NaN | 50 |
| mean | 75.500000 | 5.843333 | 3.054000 | 3.758667 | 1.198667 | NaN |
| std | 43.445368 | 0.828066 | 0.433594 | 1.764420 | 0.763161 | 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 |

```
df.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
```

```
df.isnull().sum()
```

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

No null values in the data.

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

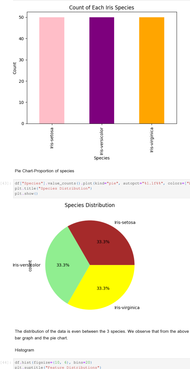
Dropped the Id column.

```
#checking after dropping the id column
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
```

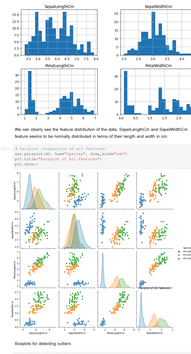
Bar Graph-count of each species

```
df["Species"].value_counts().plot(kind="bar", color=["pink","purple","orange"])
plt.title("Count of Each Iris Species")
plt.xlabel("Species")
plt.ylabel("Count")
plt.show()
```




Pie Chart:Proportion of species

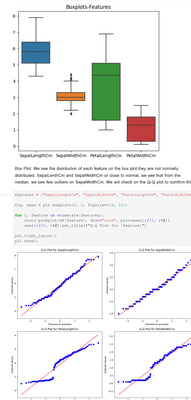
```
df["Species"].value_counts().plot(kind="pie", autopct="%1.1f%%", colors=["green
plt.title("Species Distribution")
plt.ylabel("")
plt.show()
```




The distribution of the data is even between the 3 species. We observe that from the above bar graph and the pie chart.

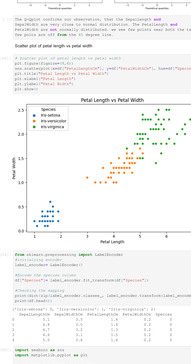
Histogram

```
df.hist(figsize=(15, 6), bins=30)
plt.suptitle("Feature Distributions")
plt.show()
```



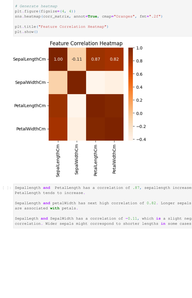

We can clearly see the feature distribution of the data. SepalLengthCm and SepalWidthCm feature seems to be normally distributed in terms of their length and width in cm.

```
# Pairplot (Comparison of all features)
sns.pairplot(df, hue="Species", palette="Set1")
plt.suptitle("Pairplot of All Features")
plt.show()
```




Boxplots for detecting outliers

```
plt.figure(figsize=(8, 6))
sns.boxplot(data=df)
plt.title("Boxplots-Features")
plt.show()
```




Box Plot: We see the distribution of each feature on the box plot.they are not normally distributed. SepalLengthCm and SepalWidthCm is close to normal. we see that from the median. we see few outliers on SepalWidthCm. We will check on the Q-Q plot to confirm this.

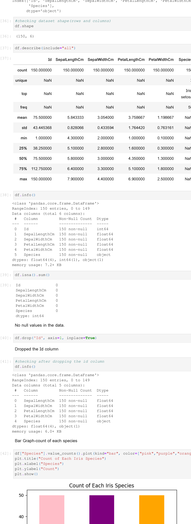
```
features = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']

fig, axes = plt.subplots(2, 2, figsize=(12, 10))

for i, feature in enumerate(features):
    stats.probplot(df[feature], dist="norm", plot=axes[i//2, i%2])
    axes[i//2, i%2].set_title(f'Q-Q Plot for {feature}')

plt.tight_layout()
plt.show()
```




The Q-Q plot confirms our observation, that the SepalLength and SepalWidth are normally distributed. Slight variations in the PetalLength and PetalWidth are not normally distributed. we see few points near both the tail s few points are off from the 45 degree line.

Scatter plot of sepal length vs petal width

```
# Scatter plot of petal length vs petal width
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x="SepalLengthCm", y="PetalWidthCm", hue="Species")
plt.title("Petal Length vs Petal Width")
plt.xlabel("Petal Length")
plt.ylabel("Petal Width")
plt.show()
```




```
from sklearn.preprocessing import LabelEncoder
#Initializing encoder
label_encoder= LabelEncoder()

#Encode the species column
df["Species"]= label_encoder.fit_transform(df["Species"])

#Checking the mapping
print(dict(zip(label_encoder.classes_, label_encoder.transform(label_encoder.cl
print(df.head())
```

```
{'Iris-setosa': 0, 'Iris-versicolor': 1, 'Iris-virginica': 2}
   SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  Species
0            5.1           3.5            1.4           0.2        0
1            4.9           3.0            1.4           0.2        0
2            4.7           3.2            1.3           0.2        0
3            4.6           3.1            1.5           0.2        0
4            5.0           3.6            1.4           0.2        0
```

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

# Ensure only numeric data is used
numeric_df = df.select_dtypes(include=['float64', 'int64'])

# Compute correlation matrix
corr_matrix = numeric_df.corr()

# Create heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(corr_matrix, annot=True, cmap="Oranges", fmt=".2f")

plt.title("Feature Correlation Heatmap")
plt.show()
```




SepalLength and PetalLength has a correlation of .87. sepallength increases Petallength tends to increase.

SepalLength and petalWidth has next high correlation of 0.82. longer sepals are associated with wide petals.

SepalLength and SepalWidth has a correlation of -0.11, which is a slight negative correlation. Wider sepals might correspond to shorter lengths in some cases.