Big Data Analysis

Perform data Wrangling on IRIS dataset in python

- Teaching Assistant
Yash More

INDEX :

- Importing libraries,
- Loading Dataset,
- Check for missing values,
- Check for distribution of the target values,
- Splitting dataset into features and target variable,
- Last step description,
- Graph Plotting,
- Graph descriptions.

Importing libraries :

Firstly, let's import the necessary libraries and load the dataset:

```
import pandas as pd
import seaborn as sns

iris = sns.load_dataset('iris')
```

Loading Dataset :

The load_dataset function from the seaborn library is used to load the IRIS dataset. Now, let's take a look at the dataset:

```
print(iris.head())
```

Loading Dataset :

This will print the first few rows of the dataset, which looks something like this:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Loading Dataset :

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Here, we can see that the dataset contains 5 columns - sepal length, sepal width, petal length, petal width, and species. The species column is the target variable we will be predicting.

Check for missing values :

Now, let's check if there are any missing values in the dataset:

```
print(iris.isnull().sum())
```

Check for missing values :

```
print(iris.isnull().sum())
```

This will output the number of missing values in each column:

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

Since there are no missing values in any column, we can move on to the next step.

Check for distribution of the target values :

Let's now check the distribution of the target variable - species:

```
print(iris['species'].value_counts())
```

Check for distribution of the target values :

Let's now check the distribution of the target variable - species:

```
print(iris['species'].value_counts())
```

This will output the number of instances of each species in the dataset:

```
versicolor    50
virginica     50
setosa        50
Name: species, dtype: int64
```

Splitting dataset into features and target variable :

Finally, let's split the dataset into features and target variable:

```
X = iris.drop('species', axis=1)
y = iris['species']
```

Splitting dataset into features and target variable :

Finally, let's split the dataset into features and target variable:

```
X = iris.drop('species', axis=1)
y = iris['species']
```

- Here, we are creating two separate dataframes - X contains all the columns except the target variable, and y contains only the target variable column.
- This completes the process of data wrangling on the IRIS dataset in Python.

Last step description :

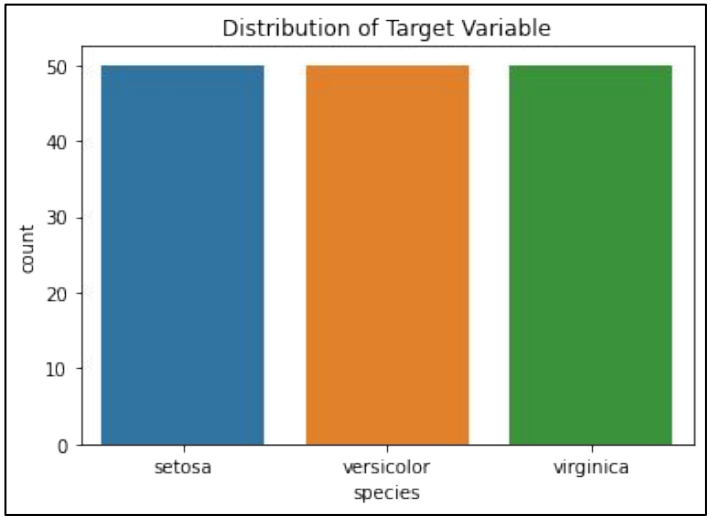
- In the last step of data wrangling on the IRIS dataset, we split the dataset into features and target variable. We created two separate dataframes - X contains all the columns except the target variable, and y contains only the target variable column.
- The reason we excluded the target variable from the X dataframe is because we want to use the features to predict the target variable. In other words, we want to train a machine learning model using the features (sepal length, sepal width, petal length, and petal width) to predict the species of the iris flower.
- The target variable y is kept separate because it is what we are trying to predict. We will use y to evaluate the performance of the machine learning model by comparing the predicted values with the actual values.
- Therefore, we separate the target variable from the features in order to perform supervised learning on the dataset.

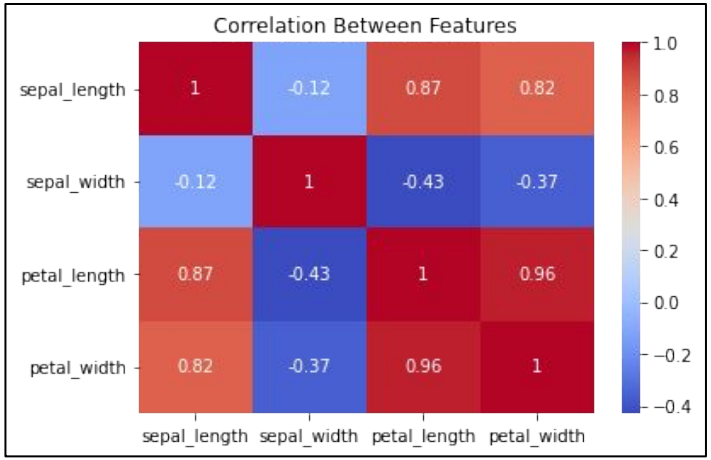
Graph Plotting :

```
import matplotlib.pyplot as plt

# Plot the distribution of target variable
plt.figure(figsize=(6, 4))
sns.countplot(x='species', data=iris)
plt.title('Distribution of Target Variable')
plt.show()

# Plot the correlation between features
plt.figure(figsize=(6, 4))
sns.heatmap(X.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Between Features')
plt.show()
```

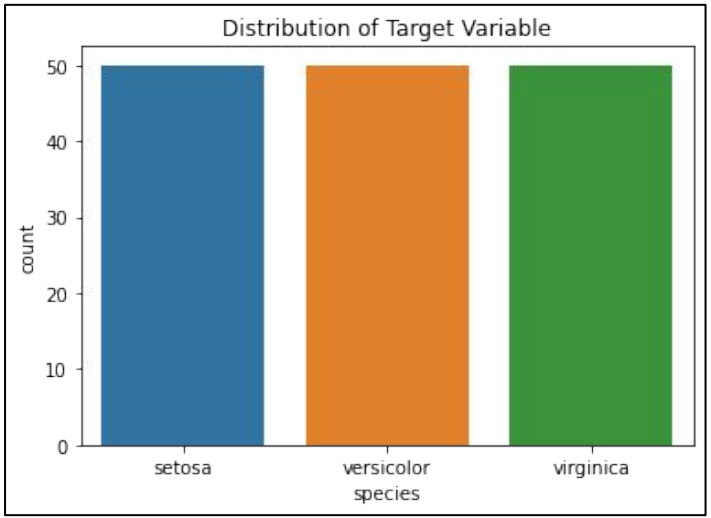




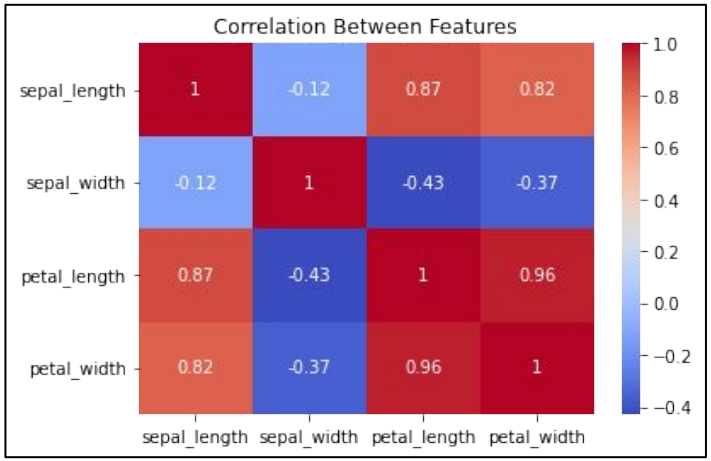
- The first plot shows the distribution of the target variable species, using a countplot. This graph will show the number of instances of each species in the dataset.

- The second plot shows the correlation between the features using a heatmap. This graph will show how strongly each feature is correlated with the other features. The annot=True parameter is used to show the correlation values in each cell of the heatmap.

- Note that we are using the sns object from the Seaborn library to create the plots, which provides a higher-level interface to create beautiful and informative visualizations in Python. The plt object from the Matplotlib library is used to customize the plots further.



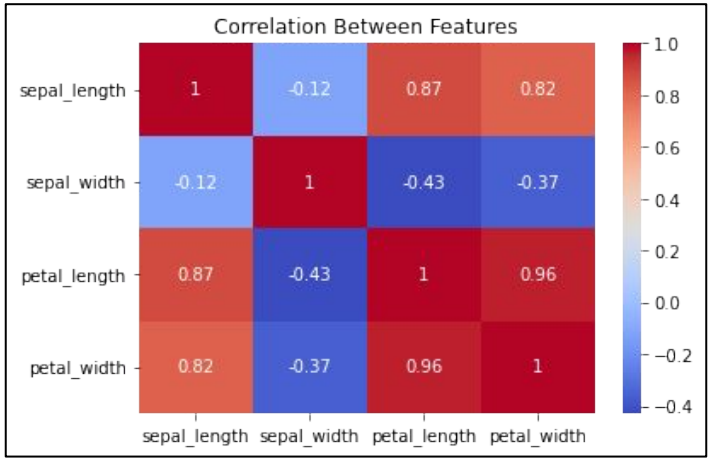
- The target variable in the IRIS dataset is the species of the iris flower, which can take on one of three values - setosa, versicolor, or virginica.
- The distribution of the target variable refers to how the instances of each species are distributed in the dataset. In other words, it refers to how many instances of each species are present in the dataset.
- In the IRIS dataset, we can see that the distribution of the target variable is balanced. There are exactly 50 instances of each species in the dataset, making it a perfectly balanced dataset. This means that each species is equally represented in the dataset, and we do not have to worry about any bias towards one species.
- It is important to understand the distribution of the target variable because it can affect the performance of the machine learning model. In an imbalanced dataset where one species is overrepresented, the machine learning model may be biased towards that species and may not perform well on the other species. Therefore, it is important to ensure that the dataset is balanced or to take steps to balance the dataset if it is imbalanced.



- The Correlation Between Features graph in the IRIS dataset shows the correlation between the four features - sepal length, sepal width, petal length, and petal width.

- The correlation between two features is a measure of the strength of the linear relationship between them. It ranges from -1 to 1, where -1 indicates a perfect negative correlation (as one feature increases, the other feature decreases), 0 indicates no correlation, and 1 indicates a perfect positive correlation (as one feature increases, the other feature also increases).

- In the Correlation Between Features graph, we use a heatmap to visualize the correlation matrix between the four features. Each cell in the heatmap represents the correlation between two features. The color of each cell indicates the strength of the correlation - warmer colors (e.g., red) indicate a positive correlation, while cooler colors (e.g., blue) indicate a negative correlation.



- In the IRIS dataset, we can see that the petal length and petal width features are highly correlated (correlation coefficient = 0.96), while the sepal length and sepal width features are weakly correlated (correlation coefficient = -0.12).
- This information is useful when selecting features for the machine learning model. Highly correlated features can be problematic for some machine learning algorithms, as they can lead to overfitting and reduced generalization performance. Therefore, we may consider removing one of the highly correlated features (e.g., petal width) from the dataset to improve the performance of the model. On the other hand, weakly correlated features can provide complementary information to the model and may be useful for improving its performance.

Thank You!!