ML Experiment 1

ML Experiment 1 DATE:

Kreena Shah
60004210243
C'32

Aim : To perform data cleaning & preprocessing on the dataset & implement linear regression.

Theory :

Data Cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate or incomplete data within a dataset.

While integrating multiple data sources, discrepancy is possible which makes outcomes & algorithms unreliable

Steps to clean data :

(1) Remove irrelevant data
(2) Remove duplicate & incomplete cases
(3) Fix structural errors
(4) Deal with missing data
(5) Identify & review outliers
(6) Filter out outliers
(7) Encoding categorical data
(8) Splitting the dataset
(9) Feature Scaling

Linear Regression

It is a special type of machine learning algorithm more specifically a supervised algorithm

It learns from the labelled dataset & maps the data points to the most optimized linear functions,

DATE:

It computes the linear relationship between a dependent variable & one or more independent features

Simple Linear Regression

$y_i = \beta_0 + \beta_1 X$

Y : dependent variable
X : independent variable
β_0 : intercept
β_1 : slope

The goal of the algorithm is to find the best fit line equation that can predict the values based on independent variables

$$\text{Cost function } (J) = \frac{1}{n} \sum_{n}^{i} (\hat{y}_i - y_i)^2$$

n : no. of data points
y_i : actual value
$\hat{y}_i$: predicted value

Conclusion : Thus, we understood steps of data preprocessing & cleaning & thereafter implemented linear regression

```
import numpy as np
import pandas as pd
import seaborn as sns
```

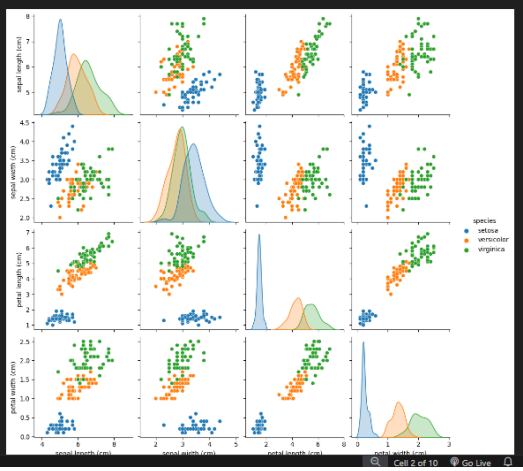
```
# Import Dataset from sklearn
from sklearn.datasets import load_iris
# Load Iris Data
iris = load_iris()
# Creating pd DataFrames
iris_df = pd.DataFrame(data= iris.data, columns= iris.feature_names)
target_df = pd.DataFrame(data= iris.target, columns= ['species'])
def converter(specie):
    if specie == 0:
        return 'setosa'
    elif specie == 1:
        return 'versicolor'
    else:
        return 'virginica'
target_df['species'] = target_df['species'].apply(converter)
# Concatenate the DataFrames
iris_df = pd.concat([iris_df, target_df], axis= 1)
```

```
iris_df.describe()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238

```
sns.pairplot(iris_df, hue= 'species')
```

```
<seaborn.axisgrid.PairGrid at 0x23fd2d7e800>
```




```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

```
# Converting Objects to Numerical dtype
iris_df.drop('species', axis= 1, inplace= True)
target_df = pd.DataFrame(columns= ['species'], data= iris.target)
iris_df = pd.concat([iris_df, target_df], axis= 1)
# Variables
X= iris_df.drop(labels= 'sepal length (cm)', axis= 1)
y= iris_df['sepal length (cm)']

# Splitting the Dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 0.33, random_state= 10

# Instantiating LinearRegression() Model
lr = LinearRegression()

# Training/Fitting the Model
lr.fit(X_train, y_train)

# Making Predictions
lr.predict(X_test)
pred = lr.predict(X_test)

# Evaluating Model's Performance
print('Mean Absolute Error:', mean_absolute_error(y_test, pred))
print('Mean Squared Error:', mean_squared_error(y_test, pred))
print('Mean Root Squared Error:', np.sqrt(mean_squared_error(y_test, pred)))
```

```
Mean Absolute Error: 0.2595570975563035
```

```
iris_df.loc[6]
```

```
sepal length (cm)    4.6
sepal width (cm)     3.4
petal length (cm)    1.4
petal width (cm)     0.3
species              0.0
Name: 6, dtype: float64
```

```
d = {'sepal length (cm)' : [4.6],
     'sepal width (cm)' : [3.4],
     'petal length (cm)' : [1.4],
     'petal width (cm)' : [0.3],
     'species' : 0}
test_df = pd.DataFrame(data= d)
test_df
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	4.6	3.4	1.4	0.3	0

```
pred = lr.predict(X_test)
print('Predicted Sepal Length (cm):', pred[0])
print('Actual Sepal Length (cm):', 4.6)
```

```
Predicted Sepal Length (cm): 5.461145872156033
Actual Sepal Length (cm): 4.6
```