Machine learning assignment 1 report

M22CS007

Task 1: Simple linear regression

In the given dataset, the columns are as follows:

Year   Rainfall (mm)   Yield (hg/ha)   Remarks

The output of the dataset is "and Yield (hg/ha)", which tells about the yield of the crop that year. The dependent variable is Rainfall (mm).

To know about the dependencies, I have used correlation and covariance parameters. These can be clearly shown using the heatmap using Seaborn library.But since we are not allowed to use the libraries, I used only the corr(), cov() functions on the data.

Corr() is used to know the relationship between two variables. However, in our dataset, 'Unnamed' column is just for the numbers(index) , and as we already know about the 'Year' column, We are only left with rainfall and yield.

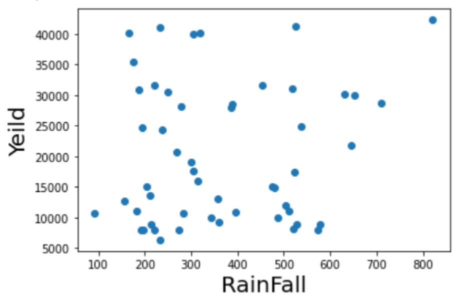
The value of the correlation coefficient can take any values from -1 to 1.

- **If the value is 1**, it is said to be a **positive correlation** between two variables. This means that when one variable increases, the other variable also increases.
- **If the value is -1**, it is said to be a **negative correlation** between the two variables. This means that when one variable increases, the other variable decreases.
- **If the value is 0**, there is no correlation between the two variables. This means that the variables changes in a random manner with respect to each other.

Now that, we understood about the dependent and independent variables, lets move further.

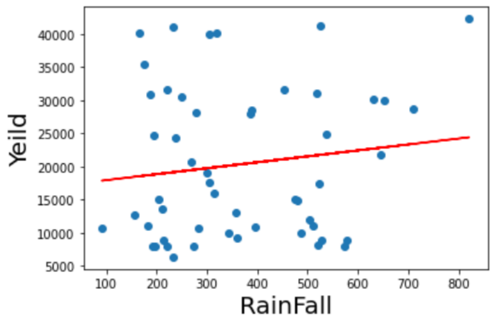
I have plotted a scatter plot between Rainfall and yield quantities. For which, I had to first rename the columns, as they were having spaces, pandas considers t as different names and displays error. So I changed it to Rainfall and yield.

```
<matplotlib.collections.PathCollection at 0x7f886dc802d0>
```



To get the regression line, we use y=m*x+c equation (generally termed as regression equation), where m is the slope and c is the coefficient (bias/intercept).

```
[<matplotlib.lines.Line2D at 0x7f12e8a061d0>]
```



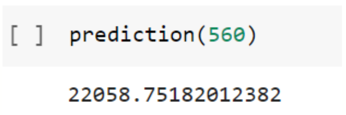
To get m, we use the formula (x-xmean)(y-ymean)/(x-xmean)**2;

And bias or intercept is just the difference between predicted output and actual output, (y_predicted-y).

slope m=8.972274380148036 and Intercept = 17034.278167240922

So, the predicted value becomes, m*value+bias

In the question, the asked us to predict the yield of the crop for the average rainfall of 560mm this year. According to my prediction, the yield would be 22058.7 hg/ha.



```
[ ] prediction(560)

    22058.75182012382
```

prediction(560)
22058.75182012382

To report errors, we use MAE and MSE generally, which stand for mean absolute error and mean square error. MAE means the average of all absolute errors. The absolute average distance between the real data and the predicted data.

Find all of your absolute errors, xi – x. Add them all up.Divide by the number of errors.

MSE , measures the average of the squares of the errors— that is, the average squared difference between the estimated values and the actual value

Mean Square Error(MSE)= 9764.615081494694And

Mean Absolute Error(MAE)= 122363336.42449357

Google colab link: https://colab.research.google.com/drive/1_nRfegNFZLR-H6Nda6r295YUA7Zi3SuT?usp=sharing

---

Task 2: Multiple linear regression

The dataset given here is about the cars and prices.

We are interested in knowing the relationship between the attributes. How the dependent and independent attributes are connected to each other.In order to process the data, first we have to clean the data, find if there are any missing values and change the data accordingly. So this is just pre-processing the data. In our data, we found some missing values, so we handled the data accordingly.

And since the data didn’t have the labels, we inserted the column names, taking the reference from the link given in the question.

We have considered price as an independent variable. And we split the data into training and testing in 70:30 scale. Now, performing multiple linear regression using gradient descent algorithm. Gradient Descent is an iterative algorithm used in loss function calculation to find the global minima. The loss can be any differential loss function. The different types of loss functions are linear loss, logistic loss, hinge loss, etc. For our dataset, we will be using linear loss because the target is a continuous variable.

When we say we are building a Linear Regression model, It is nothing but we are trying to find a straight line (one feature) or a hyperplane (multiple features) that best fits the data. And the equation of a line is represented by “y=mx+c” where “m” is the slope and “b” is bias. We can also refer to “m” as weight and “b” as intercept.

```
loss in training = 8560580.68938437
loss in testing = 8326999.935208425
```

Both square mean loss is approximate same so not the model is not over fitting.

R-Squared is a statistical measure of fit that indicates how much variation of a dependent variable is explained by the independent variable(s) in a regression model.

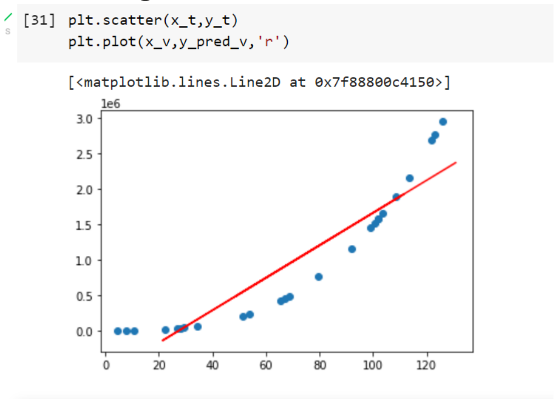
R square value is 0.41438 for my model. It means there is approximately 41.43% dependency between the dependent and independent variables.

Google colab link:
https://colab.research.google.com/drive/1vUdLvyl fzBe0RUPzokM27GKwRgVIad?usp=sharing

Task 3: **Polynomial Regression**

Ramya has to decide which model is best, for which she used three models.

1. Linear regression

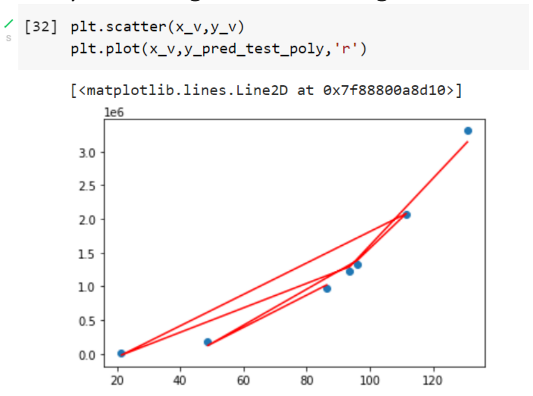


```
[31]  plt.scatter(x_t,y_t)
      plt.plot(x_v,y_pred_v,'r')
```

```
[<matplotlib.lines.Line2D at 0x7f88800c4150>]
```

r2 socre is 0.8277400752428455
mean_sqrd_error is== 188213463154.6782
root_mean_squared error of is== 433835.75596610084

2. Polynomial regression with degree 2

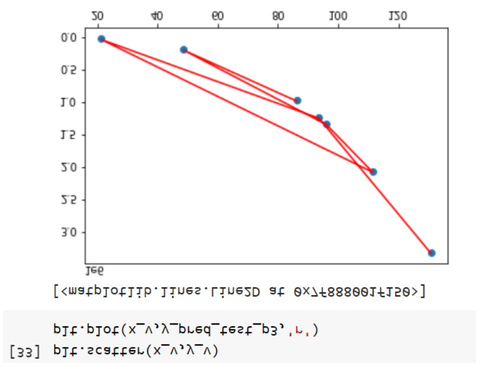


```
[32]  plt.scatter(x_v,y_v)
      plt.plot(x_v,y_pred_test_poly,'r')
```

```
[<matplotlib.lines.Line2D at 0x7f88800a8d10>]
```

r2 socre is 0.9941932947337196
mean_sqrd_error is== 6344482671.903297
root_mean_squared error of is== 79652.26093403311

3. polynomial regression with degree 3



```
[<matplotlib.lines.Line2D at 0x7f888001f120>]
```

```
      plt.plot(x_v,y_pred_test_p3,'r')
[33]  plt.scatter(x_v,y_v)
```

r2 socre is 0.9999999999995652
mean_sqrd_error is== 0.4750870597036694

root_mean_squared error of is== 0.6892655944580938

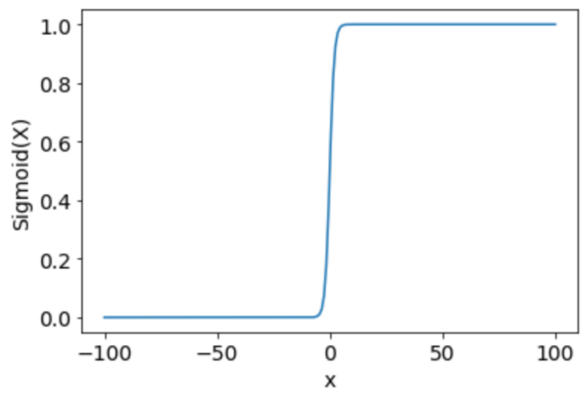
To find a better or goodmodel, the model should have **A lower value of RMSE and a higher value of R^2** for the prediction.

On seeing three model outputs, it is clear that model 3 that is, polynomial regression with degree 3 is good model.
Google colab link: https://colab.research.google.com/drive/15-KvgXs844FytRGvmMp0JsCYi9TkrQn9?usp=sharing

---

Task 5: Logistic regression

5.1 Sigmoid function:



5.2 Linear classifier

Given dataset is Algerian_forest_fires_dataset. The columns are: 'month', 'year', 'Temperature', ' RH', ' Ws', 'Rain ', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'output'.

We split the data according to input output and divide the data on 60:40 scale.

The accuracy we got is 95.91836734693877.

5.3 Building logistic regression model using Gradient descent method

For this I have used the glass dataset. Performed logistic regression using gradient descent algorithm.

```
[-4.14156901e-03  3.23963021e-04  2.68182302e-03  1.23406301e-03
  1.09234657e-04  1.53738935e-02  5.32353717e-05  1.93120574e-03
 -1.35270290e-04  3.23466423e-05  2.13138077e-04]
[1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1]
  logrithimerrormean= 0.11627906976744184
```

google colab link:

https://colab.research.google.com/drive/1J3oE7gy70RrT9NzqFx7lu1xRaXxXHHNi?usp=sharing

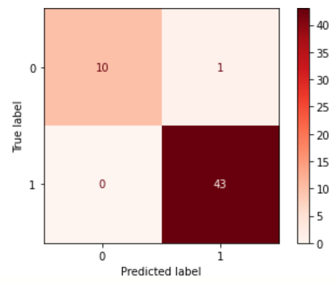
5.4 AUC-ROC

This is the confusion matrix related to **Algerian_forest_fires_dataset**

Array[[16 24]

[32 26]]

confusion matrix plot:- this is related to the glass data



array([[10, 1], [0, 43]])

google colab link: Glassdata : https://colab.research.google.com/drive/16g-HIPjfssawFJplCt7XBmZyLvbAPzVk?usp=sharing

https://colab.research.google.com/drive/1lVhbPHhCZL2zqWibYNQp4eWKUalJ3qEV?usp=sharing