Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

Tutorial No.3

Implement following algorithms using Python on suitable data sets.i. Gradient Descent ii. Linear Regression iii. Polynomial Regression iv. Logistic Regression.

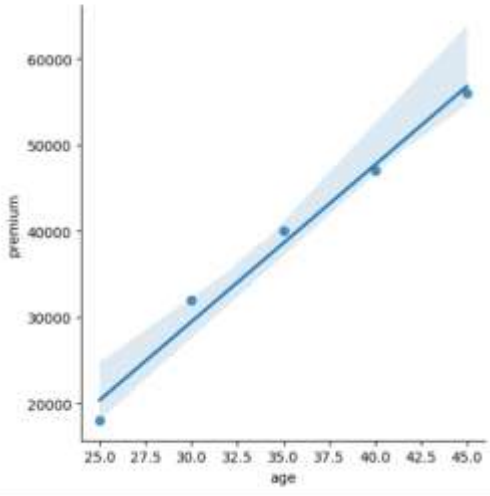
1)Gradient Descent and

2)Linear Regression

```
[3]: import pandas as pd
     import seaborn as sns
     from sklearn import linear_model

[4]: df=pd.read_csv("htrain.csv")
     #m then  import
     #m then read the
     #m implot
     #m then linearregression
     #then predic
     # then
     #m
     #m
     #m

[5]: df

[5]:
```

	age	premium
0	25	18000
1	30	32000
2	35	40000
3	40	47000
4	45	56000

```
[6]: df.columns = df.columns.str.strip()
     #m single variable regression

[6]: sns.lmplot(x="age",y="premium",data=df)
     # for the better visualize scatter plo use

[6]: <seaborn.axisgrid.FacetGrid at 0x1827ac32870>
```



Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

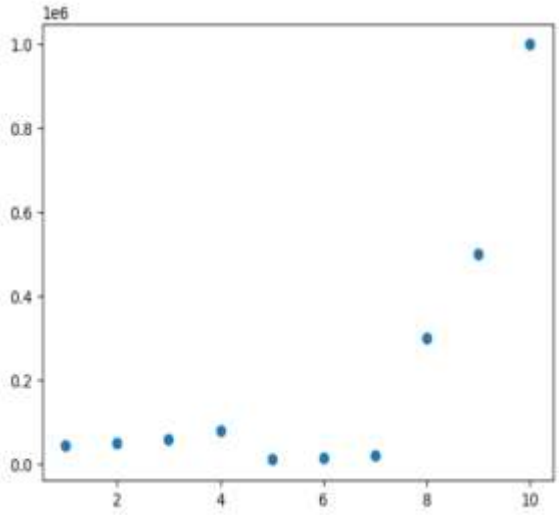
```
[34]: reglinear_model.LinearRegression();

[35]: reg.fit(df[["age"]],df["premium"])
      #reshape then multidimensional use

[35]: LinearRegression
      LinearRegression()

[36]: reg.predict([[33]])##he age input give

      C:\Users\COMP\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
        warnings.warn(

[36]: array([14960.])

[37]: reg.coef_
      #the remove coefficient slope

[37]: array([1820.])

[38]: reg.intercept_
      #remove intercept c

[38]: -25100.0

[ ]:

[ ]:

[ ]:
```

3)Polynomial Regression

```
[31]: import pandas as pd
      import matplotlib.pyplot as plt
      import seaborn as sns

[32]: dfnpd.read_csv("mpoly.csv")

[33]: df.columns

[33]: Index(['Position ', 'Level ', 'Salary'], dtype='object')

[34]: df.columns = df.columns.str.strip()#to remove whitspace

[35]: xndf.iloc[:,1:2].values
      #xndf["Level"].values

[36]: x

[36]: array([[ 1],
             [ 2],
             [ 3],
             [ 4],
             [ 5],
             [ 6],
             [ 7],
             [ 8],
             [ 9],
             [10]], dtype=int64)
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

```
[37]: ydf.iloc[:,2].values

[38]: y

[38]: array([  45000,   50000,   60000,   80000,   11000,   15000,   20000,
         300000,  500000, 1000000], dtype=int64)

[39]: plt.scatter(x,y)

[39]: <matplotlib.collections.PathCollection at 0x13f3e059ee0>
```




```
[40]: #df.columns
      #sns.lmplot(x='Level',y='Salary',data=df)
      from sklearn import linear_model

      reg=linear_model.LinearRegression()

[41]: reg=linear_model.LinearRegression()

[42]: reg.fit(x,y) #e the predict

[42]: LinearRegression()

[43]: reg.predict([[6.6]])#e the gave not correcct

[43]: array([293226.66666667])

[44]: from sklearn.preprocessing import PolynomialFeatures

[45]: poly=PolynomialFeatures(degree=1)#e the fit or poly

[46]: x_polypoly.fit_transform(x)#e the fit on the poly in e

[47]: reg2=linear_model.LinearRegression()#e thhe fit on the poly areg2.pred

[48]: reg2.fit(x_poly,y)

[48]: LinearRegression()
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

```
LinearRegression
LinearRegression()
```

```
reg.predict([[6.8]])#w the guve not corrosom
```

```
array([293226.66666667])
```

```
from sklearn.preprocessing import PolynomialFeatures
```

```
poly=PolynomialFeatures(degree=1)#w the fit on poly
```

```
x_poly=poly.fit_transform(x)#w the fit on the poly in m
```

```
reg2=linear_model.LinearRegression()#w thhe fit on the poly mreg2.pred
```

```
reg2.fit(x_poly,y)
```

```
LinearRegression
LinearRegression()
```

```
reg2.predict(poly.fit_transform([[6.5]]))
```

```
array([285487.87878788])
```

iv. Logistic Regression.

Binary Clssification

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

```
C:\Users\COMP\AppData\Local\Temp\ipykernel_16776\1492400551.py:1: DeprecationWarning:
Pyarrow will become a required dependency of pandas in the next major release of pandas (pandas 3.0),
(to allow more performant data types, such as the Arrow string type, and better interoperability with other libraries)
but was not found to be installed on your system.
If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/54466

  import pandas as pd
```

```
dm=pd.read_csv("Mlogisticregr.csv")
```

```
dm.dtypes
```

```
Mage                int64
Boughtminsurance    object
dtype: object
```

```
dm
```

	Mage	Boughtminsurance
0	21	no
1	48	yes
2	32	yes
3	41	yes
4	20	no
5	35	yes
6	20	no

5	35	yes
6	20	no
7	23	no

```
[4]: dm["Boughtminsurance"].replace({'no':'0','yes':1},inplace=True)## the convert to the numeric
```

```
C:\Users\COMP\AppData\Local\Temp\ipykernel_36776\2924367234.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through c
hained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves
as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) inste
ad, to perform the operation inplace on the original object.


  dm["Boughtminsurance"].replace({'no':'0','yes':1},inplace=True)
```

```
[32]: dm["Boughtminsurance"]=pd.to_numeric(dm["Boughtminsurance"],errors='coerce')
```

```
[33]: dm.dtypes## check the datatype
```

```
[33]: Mage                int64
      Boughtminsurance    int64
      dtype: object
```

```
[6]: #dm.columns = dm.columns.str.strip()
```

```
[34]: #plt.scatter(x='Mage',y='Boughtminsurance',data=dm)
      from sklearn.model_selection import train_test_split
```

```
[35]: xcm[["Mage"]]
```

```
[34]: #plt.scatter(x='Mage',y='Boughtminsurance',data=dm)
      from sklearn.model_selection import train_test_split
```

```
[35]: xcm[["Mage"]]
```

```
[36]: ycm[["Boughtminsurance"]]
```

```
[37]: x_train,x_test,y_train,y_test=train_test_split(x,dm['Boughtminsurance'],test_size=0.2)
```

```
[38]: len(x_test)
```

```
[38]: 2
```

```
[39]: len(x_train)
```

```
[39]: 6
```

```
[40]: len(y_train)
```

```
[40]: 6
```

```
[41]: from sklearn.linear_model import LogisticRegression## implement in prob of the result give
```

```
[42]: lr=LogisticRegression() ## the 30 below not buy
```

```
[43]: lr.fit(x_train,y_train)## the 1 mean buy
```

```
[43]: LogisticRegression()
```

```
[44]: lr.predict(x_test)## 1 mean bought insurance and 0 mean not below 30 not boughtthe abouu bought m
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

```
lr.fit(x_train,y_train)#m the I mean buy
```

```
LogisticRegression()
```

```
lr.predict(x_test)#m I mean bought insurance and 0 mean not below 30 not boughtthe above bought m
```

```
array([1, 1], dtype=int64)
```

```
x_test
```

	Mage
1	48
3	41

```
lr.predict([[9]])
```

```
C:\Users\COMP\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
array([0], dtype=int64)
```

Multiclass Logistic Regression

```
import pandas as pd
```

```
datasetpd.read_csv("Irislogimultmmm.csv")
```

```
dataset
```

	msepalLengthCm	SepeWidthCm	PetalLengthCm	petalwidthCmmm	species
0	6.8	3.2	5.9	2.3	Iris-virginic
1	6.9	3.1	5.1	2.3	Iris-virginic
2	4.9	3.0	1.4	0.2	Iris-seastoas
3	5.6	3.0	4.5	1.5	Iris-versicolor
4	4.8	3.1	1.6	0.2	Iris-seastoas
5	5.8	2.8	5.1	2.4	Iris-virginic
6	7.2	3.6	6.1	2.5	Iris-virginic
7	5.1	3.5	1.4	0.3	Iris-seastoas
8	4.7	3.2	1.6	0.2	Iris-seastoas
9	6.6	3.0	4.4	1.4	Iris-versicolor

```
dataset["species"].unique
```

```
<bound method Series.unique of 0      Iris-virginic
1      Iris-virginic
2      Iris-seastoas
3     Iris-versicolor
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

```
[31]: <bound method Series.unique of 0      Iris-virginic
      1       Iris-virginic
      2       Iris-seastoas
      3    Iris-versicolor
      4       Iris-seastoas
      5       Iris-virginic
      6       Iris-virginic
      7       Iris-seastoas
      8       Iris-seastoas
      9    Iris-versicolor
      Name: species, dtype: object>
```

```
[34]: dataset["species"].replace({'Iris-virginic':'1','Iris-versicolor':'2','Iris-seastoas':'3'},inplace=True)
```

```
[35]: dataset
```

[35]:

	msepalLengthCm	SepeWidthCm	PetalLengthCm	petalwidthCmmm	species
0	6.8	3.2	5.9	2.3	1
1	6.9	3.1	5.1	2.3	1
2	4.9	3.0	1.4	0.2	3
3	5.6	3.0	4.5	1.5	2
4	4.8	3.1	1.6	0.2	3
5	5.8	2.8	5.1	2.4	1
6	7.2	3.6	6.1	2.5	1
7	5.1	3.5	1.4	0.3	3
8	4.7	3.2	1.6	0.2	3
9	6.6	3.0	4.4	1.4	2

```
[36]: from sklearn.model_selection import train_test_split
```

```
[37]: dataset.columns
```

```
[37]: Index(['msepalLengthCm', 'SepeWidthCm', 'PetalLengthCm', 'petalwidthCmmm',
             'species'],
            dtype='object')
```

```
[38]: X_train, X_test, y_train, y_test =train_test_split(dataset[['msepalLengthCm','SepeWidthCm','PetalLengthCm','petalwidthCmmm']],dataset['species'],test_si
```

```
[39]: len(X_train)
```

```
[39]: 8
```

```
[40]: len(X_test)
```

```
[40]: 2
```

```
[41]: from sklearn.linear_model import LogisticRegression
```

```
[42]: lr=LogisticRegression()
```

```
[43]: lr.fit(X_test,y_test)
```

```
[43]: LogisticRegression()
```

```
[44]: lr.fit(X_test,y_test)
```

```
[44]: LogisticRegression()
```

```
[44]: lr.predict(X_test)
```

```
[44]: array(['3', '1'], dtype=object)
```

```
[45]: X_test de e m({'Iris-virginic':'1','Iris-versicolor':'2','Iris-seastoas':'3'
```

[46]:

	msepalLengthCm	SepeWidthCm	PetalLengthCm	petalwidthCmmm
2	4.9	3.0	1.4	0.2
1	6.9	3.1	5.1	2.3

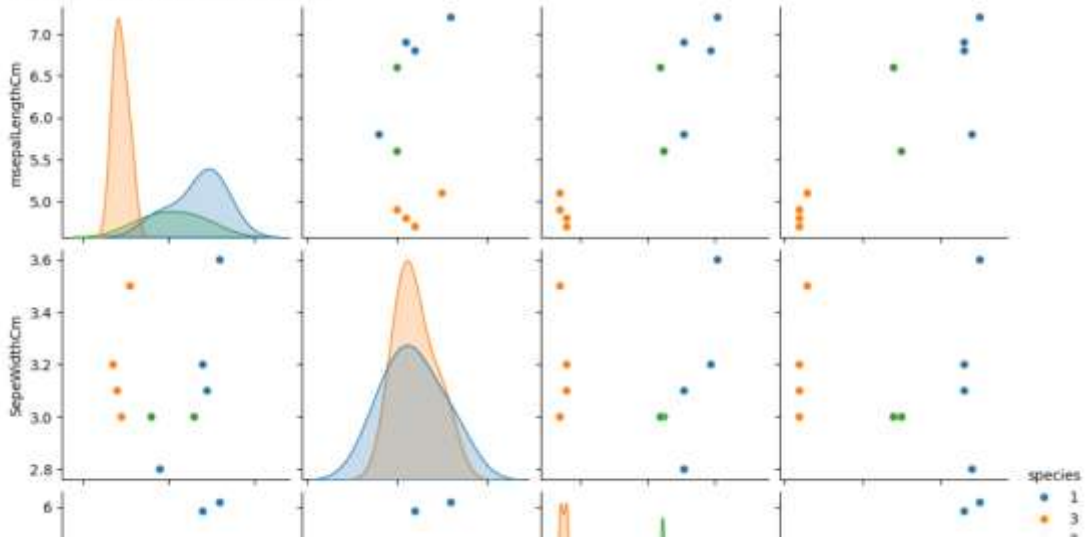
```
[46]: lr.score(X_test,y_test)
```

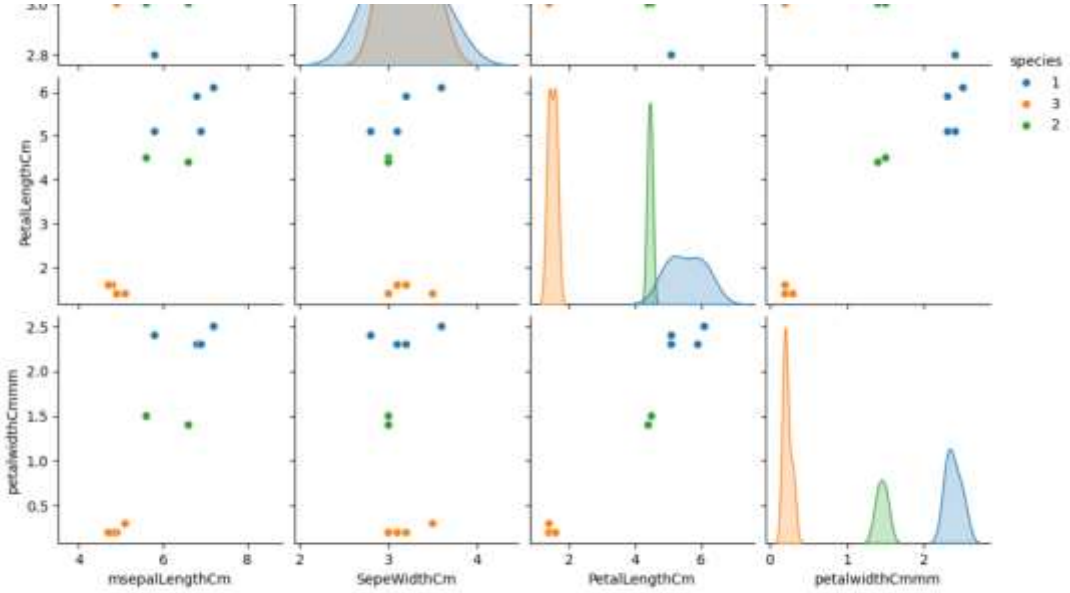
```
[46]: 1.0
```

```
[56]: import seaborn as sns
```

Name: Abhishek Mali
Roll No.: 213070
PRN No.: 22320024

```
[36]: import seaborn as sns

[37]: sns.pairplot(dataset[['msepalLengthCm','SepeWidthCm','PetalLengthCm','petalwidthCmmm','species']],hue='species')

[37]: <seaborn.axisgrid.PairGrid at 0x27e19873470>
```






```
[ ]:
```