j6l9sulgi

January 23, 2025

1 Machine Learning

Deep Neural Network, Convolution Neural Network

Types of Machine Learning

1. Supervised Learning #Data-> (Features, Target)- Linear Regression
2. Unsupervised Learning #Data-> (Features)- Clustering
3. Reinforcement Learning #Data-> (Penalty/Reward)-
4. Hybrid (Based on requirement)

Machine Learning Process

1. Data Collection- If domain is predefined/ Fixed
2. Data Analysis
3. Data Preprocessing- Cleaning
4. Model Building/ Training 1. Model Testing (Optional)
5. Model Validation
6. Deployment/ Production

```
[ ]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
```

```
[ ]: from sklearn import datasets, linear_model
     from sklearn.metrics import mean_squared_error
```

```
[ ]: #Diabetes Dataset
     diabetes_X,diabetes_y =datasets.load_diabetes(return_X_y=True)
```

```
[ ]: #using one feature only
     diabetes_X=diabetes_X[:,np.newaxis,2] #slicing to get only one feature
```

```
[ ]: diabetes_X_train=diabetes_X[:-20] #slicing from zero to before last 20
     diabetes_X_test=diabetes_X[-20:] #slicing from last 20 to -1
```

```
[ ]: diabetes_y_train=diabetes_y[:-20]
     diabetes_y_test=diabetes_y[-20:]
```

[ ]:

```
#Linear Regression model
regression=linear_model.LinearRegression()
regression.fit(diabetes_X_train,diabetes_y_train)
```

[ ]:

```
LinearRegression()
```

[ ]:

```
diabetes_y_pred=regression.predict(diabetes_X_test)
```

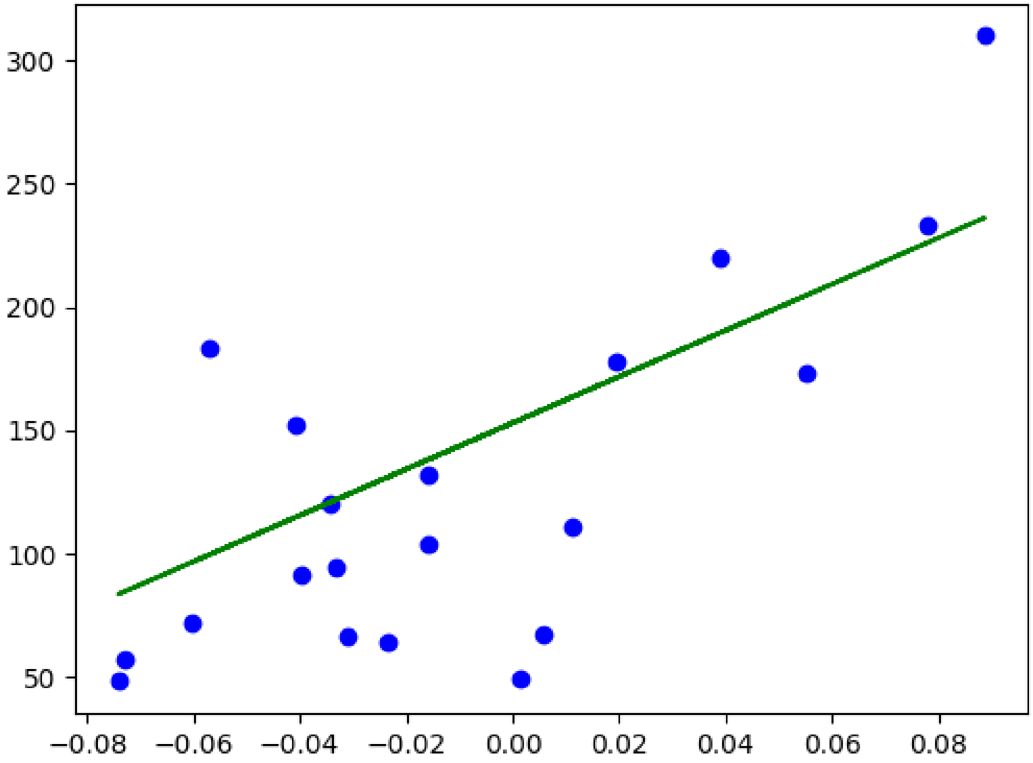
[ ]:

```
print("Weights: ",regression.coef_)
```

```
Weights:  [938.23786125]
```

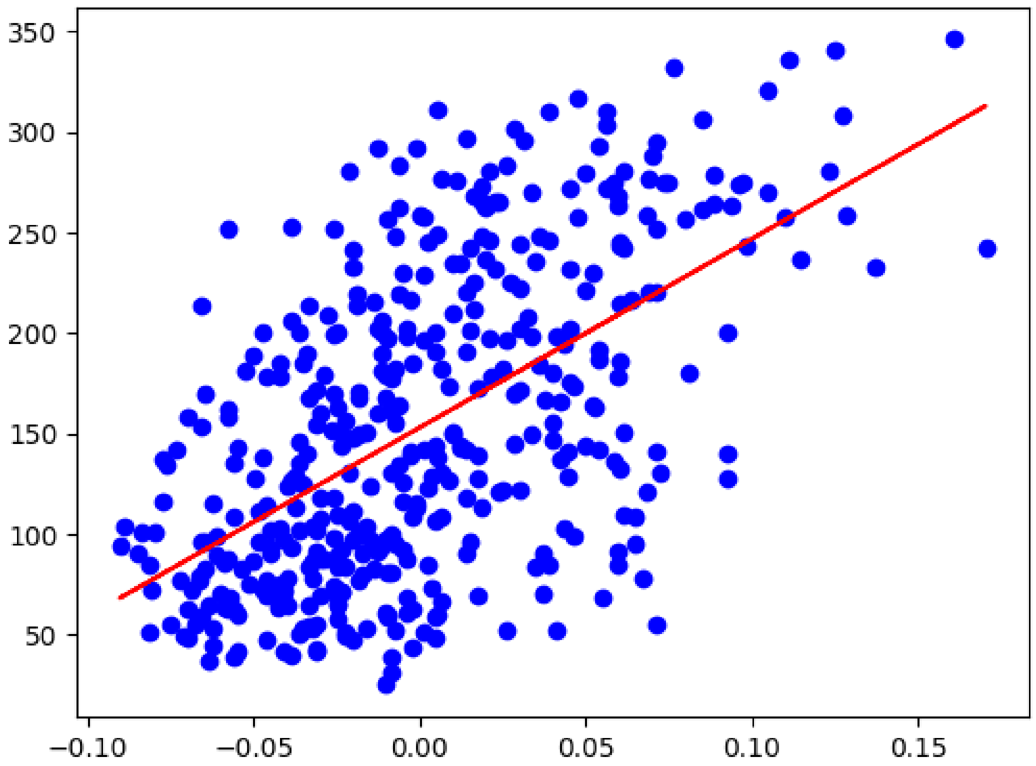
[ ]:

```
#checcking on test data
plt.scatter(diabetes_X_test,diabetes_y_test,color="blue")
plt.plot(diabetes_X_test,diabetes_y_pred,color="green")
plt.show()
```



[ ]:

```
#checking on train data
diabetes_y_pred=regression.predict(diabetes_X_train)
plt.scatter(diabetes_X_train,diabetes_y_train,color="blue")
plt.plot(diabetes_X_train,diabetes_y_pred,color="red")
plt.show()
```



2 Using another index feature for linear Regression

[ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error
```

[ ]:

```
#Diabetes Dataset
diabetes_X,diabetes_y =datasets.load_diabetes(return_X_y=True)
```

[ ]:

```
#using one feature only
diabetes_X=diabetes_X[:,np.newaxis,7] #slicing to get only one feature
```

[ ]:

```
diabetes_X_train=diabetes_X[:-20] #slicing from zero to before last 20
diabetes_X_test=diabetes_X[-20:] #slicing from last 20 to -1
```

[ ]:

```
#Linear Regression model
regression=linear_model.LinearRegression()
regression.fit(diabetes_X_train,diabetes_y_train)
```

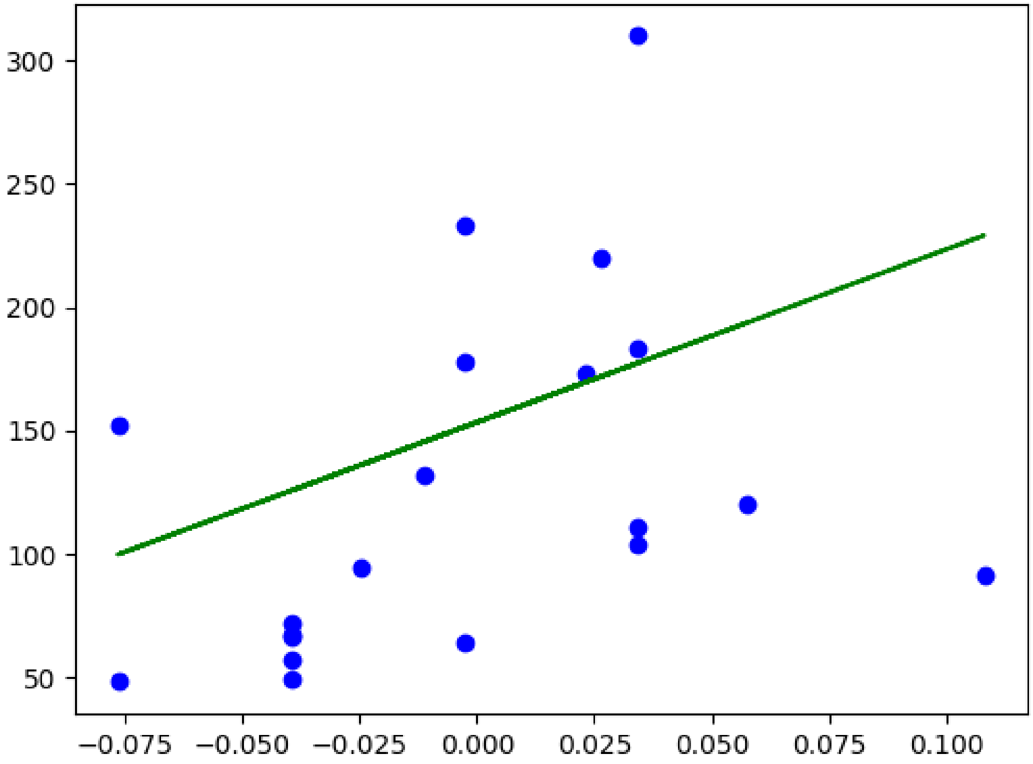
```
[ ]: LinearRegression()
```

```
[ ]: diabetes_y_pred=regression.predict(diabetes_X_test)
```

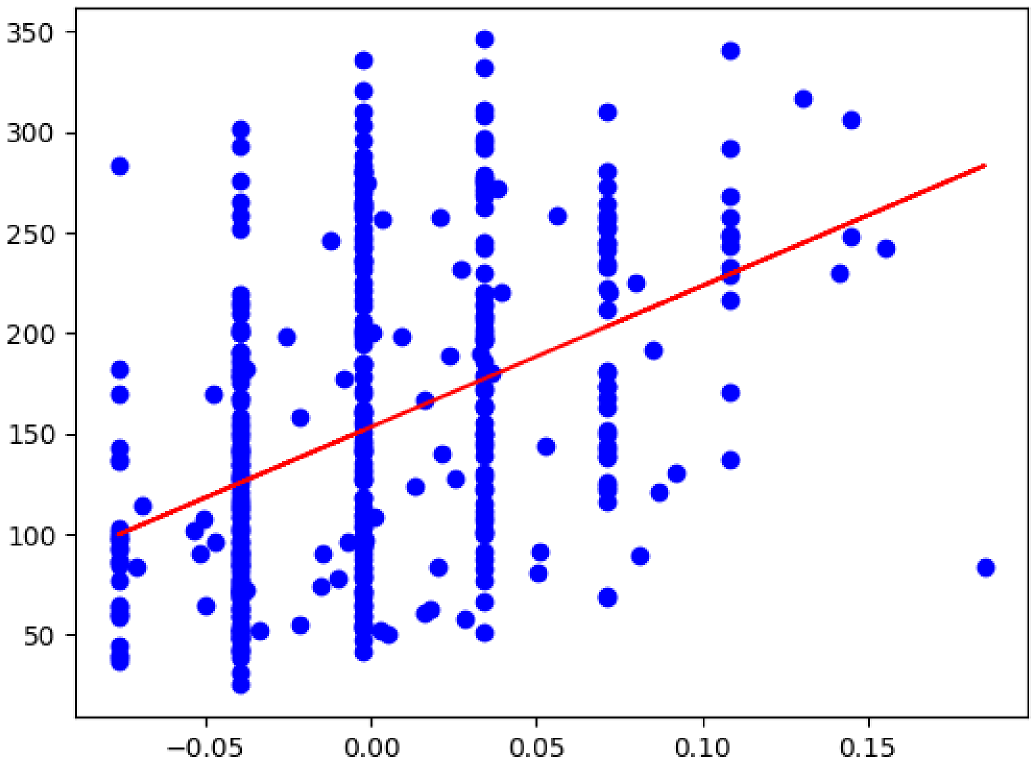
```
[ ]: print("Weights: ",regression.coef_)
```

```
Weights:  [701.12961061]
```

```
[ ]: #checcking on test data
     plt.scatter(diabetes_X_test,diabetes_y_test,color="blue")
     plt.plot(diabetes_X_test,diabetes_y_pred,color="green")
     plt.show()
```



```
[ ]: #checking on train data
     diabetes_y_pred=regression.predict(diabetes_X_train)
     plt.scatter(diabetes_X_train,diabetes_y_train,color="blue")
     plt.plot(diabetes_X_train,diabetes_y_pred,color="red")
     plt.show()
```


350
300
250
200
150
100
50
−0.05
0.00
0.05
0.10
0.15

```
[ ]:
```