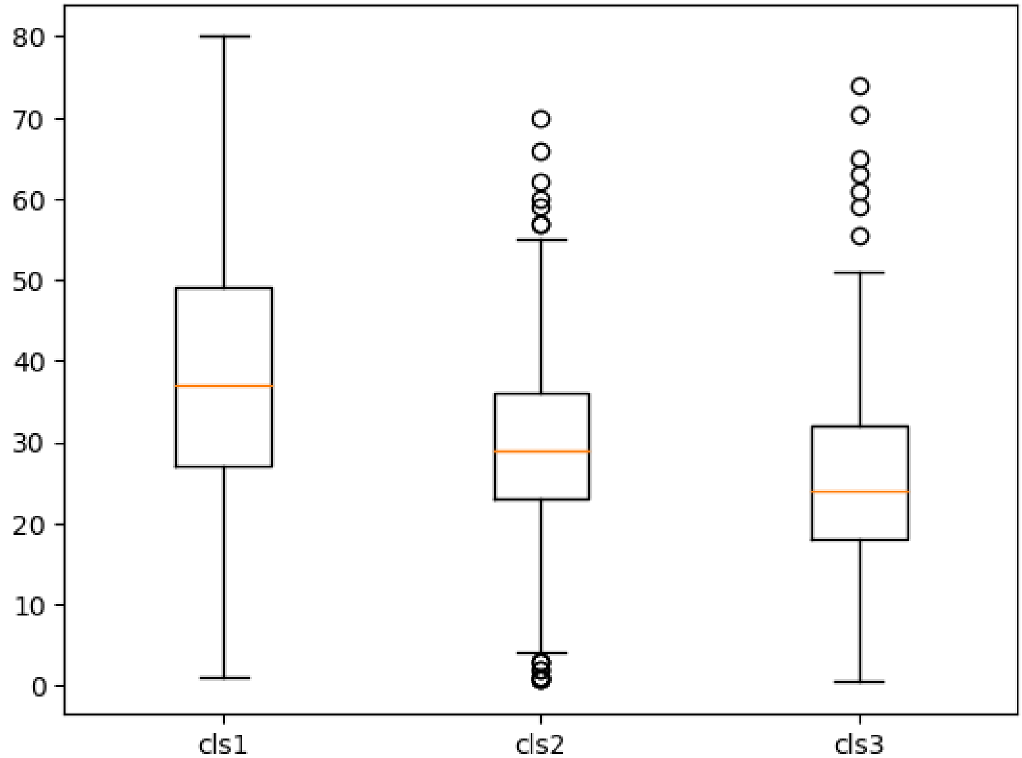
day-5-630

February 16, 2024

[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
df_titanic = pd.read_csv('titanic_train.csv')
cls1=df_titanic[df_titanic['Pclass']==1]['Age'].dropna()
cls2=df_titanic[df_titanic['Pclass']==2]['Age'].dropna()
cls3=df_titanic[df_titanic['Pclass']==3]['Age'].dropna()
```

[2]:
```python
l1=[cls1,cls2,cls3]
plt.boxplot(l1 ,labels=["cls1","cls2","cls3"])
```

[2]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1c6164f3d50>,
  <matplotlib.lines.Line2D at 0x1c616504b90>,
  <matplotlib.lines.Line2D at 0x1c616511290>,
  <matplotlib.lines.Line2D at 0x1c616511e10>,
  <matplotlib.lines.Line2D at 0x1c61651e210>,
  <matplotlib.lines.Line2D at 0x1c61651ed50>],
 'caps': [<matplotlib.lines.Line2D at 0x1c616505710>,
  <matplotlib.lines.Line2D at 0x1c616506410>,
  <matplotlib.lines.Line2D at 0x1c6165129d0>,
  <matplotlib.lines.Line2D at 0x1c616513590>,
  <matplotlib.lines.Line2D at 0x1c61651f910>,
  <matplotlib.lines.Line2D at 0x1c616528510>],
 'boxes': [<matplotlib.lines.Line2D at 0x1c6164c6c90>,
  <matplotlib.lines.Line2D at 0x1c6165106d0>,
  <matplotlib.lines.Line2D at 0x1c61651d750>],
 'medians': [<matplotlib.lines.Line2D at 0x1c616506fd0>,
  <matplotlib.lines.Line2D at 0x1c61651c110>,
  <matplotlib.lines.Line2D at 0x1c616529050>],
 'fliers': [<matplotlib.lines.Line2D at 0x1c6165079d0>,
  <matplotlib.lines.Line2D at 0x1c61651cb90>,
  <matplotlib.lines.Line2D at 0x1c616529ad0>],
 'means': []}
```



[3]:
```
df_titanic.rename(columns={'Sex':'Gender'})
```

[3]:
```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           …         …     …
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name  Gender   Age  SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                               Heikkinen, Miss. Laina  female  26.0      0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                             Allen, Mr. William Henry    male  35.0      0
..                                                 …     …   …    …
```

```
886                            Montvila, Rev. Juozas    male  27.0      0
887                     Graham, Miss. Margaret Edith  female  19.0      0
888         Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                            Behr, Mr. Karl Howell    male  26.0      0
890                              Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     …                 …       …     …       …
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

[4]:

```python
df_titanic.rename(columns={'Sex':'Gender'},inplace=True)
df_titanic
```

[4]:

```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           …       …     …
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name  Gender   Age  SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                               Heikkinen, Miss. Laina  female  26.0      0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                             Allen, Mr. William Henry    male  35.0      0
..                                                 …     …   …    …
886                            Montvila, Rev. Juozas    male  27.0      0
887                     Graham, Miss. Margaret Edith  female  19.0      0
888         Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
```

```
889                            Behr, Mr. Karl Howell    male  26.0      0
890                              Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     …                 …      …   …      …
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

[5]:
```
df_titanic['Gender']=df_titanic['Gender'].map({'male':0,'female':1})
df_titanic
```

[5]:
```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           …       …     …
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name  Gender   Age  SibSp  \
0                              Braund, Mr. Owen Harris       0  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…       1  38.0      1
2                               Heikkinen, Miss. Laina       1  26.0      0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)       1  35.0      1
4                             Allen, Mr. William Henry       0  35.0      0
..                                                 …     …   …    …
886                              Montvila, Rev. Juozas       0  27.0      0
887                       Graham, Miss. Margaret Edith       1  19.0      0
888           Johnston, Miss. Catherine Helen "Carrie"       1   NaN      1
889                              Behr, Mr. Karl Howell       0  26.0      0
890                                Dooley, Mr. Patrick       0  32.0      0
```

```
     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     …               …       …   …     …
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

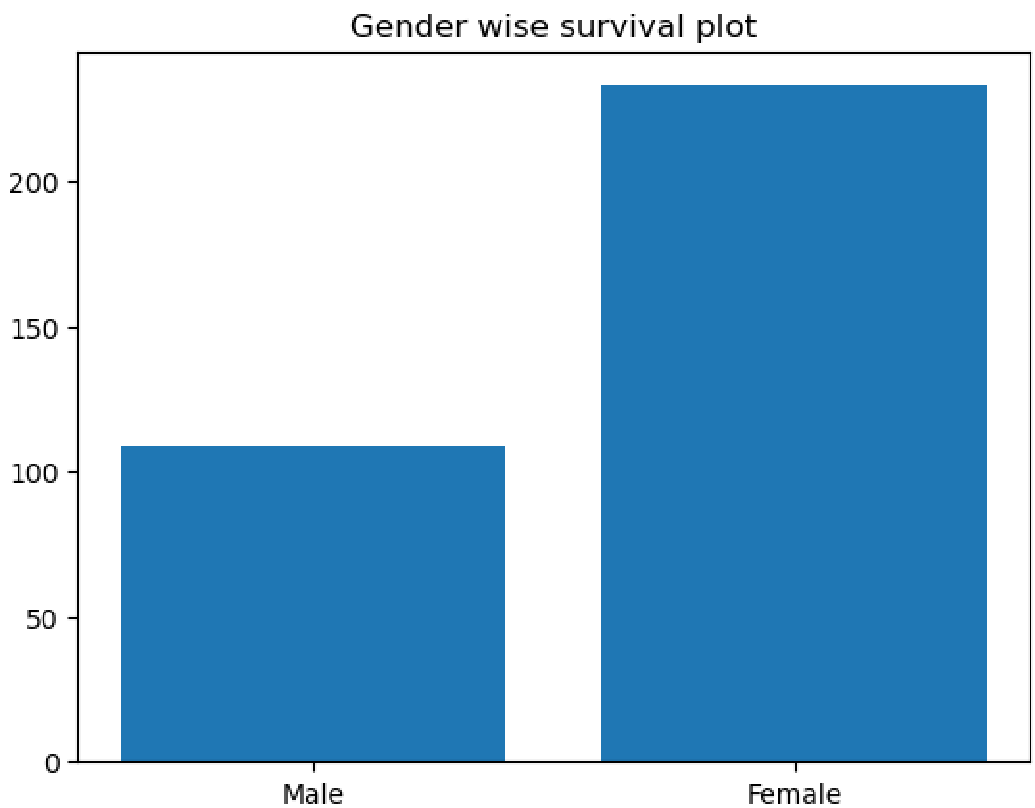
[6]:
```
df_titanic[(df_titanic['Gender']==1) & (df_titanic['Age'] < 25)]
```

[6]:
```
     PassengerId  Survived  Pclass                                     Name  \
9             10         1       2      Nasser, Mrs. Nicholas (Adele Achem)
10            11         1       3          Sandstrom, Miss. Marguerite Rut
14            15         0       3     Vestrom, Miss. Hulda Amanda Adolfina
22            23         1       3              McGowan, Miss. Anna "Annie"
24            25         0       3            Palsson, Miss. Torborg Danira
..           …       …     …                                      …
855          856         1       3                Aks, Mrs. Sam (Leah Rosen)
858          859         1       3   Baclini, Mrs. Solomon (Latifa Qurban)
875          876         1       3         Najib, Miss. Adele Kiamie "Jane"
882          883         0       3             Dahlberg, Miss. Gerda Ulrika
887          888         1       1             Graham, Miss. Margaret Edith

     Gender   Age  SibSp  Parch   Ticket     Fare Cabin Embarked
9         1  14.0      1      0   237736  30.0708   NaN        C
10        1   4.0      1      1  PP 9549  16.7000    G6        S
14        1  14.0      0      0   350406   7.8542   NaN        S
22        1  15.0      0      0   330923   8.0292   NaN        Q
24        1   8.0      3      1   349909  21.0750   NaN        S
..      …   …    …    …      …    …   …
855       1  18.0      0      1   392091   9.3500   NaN        S
858       1  24.0      0      3     2666  19.2583   NaN        C
875       1  15.0      0      0     2667   7.2250   NaN        C
882       1  22.0      0      0     7552  10.5167   NaN        S
887       1  19.0      0      0   112053  30.0000   B42        S

[117 rows x 12 columns]
```

[7]:
```
gender=['Male','Female']
male = ((df_titanic['Gender']==0) &( df_titanic['Survived'])).sum()
```

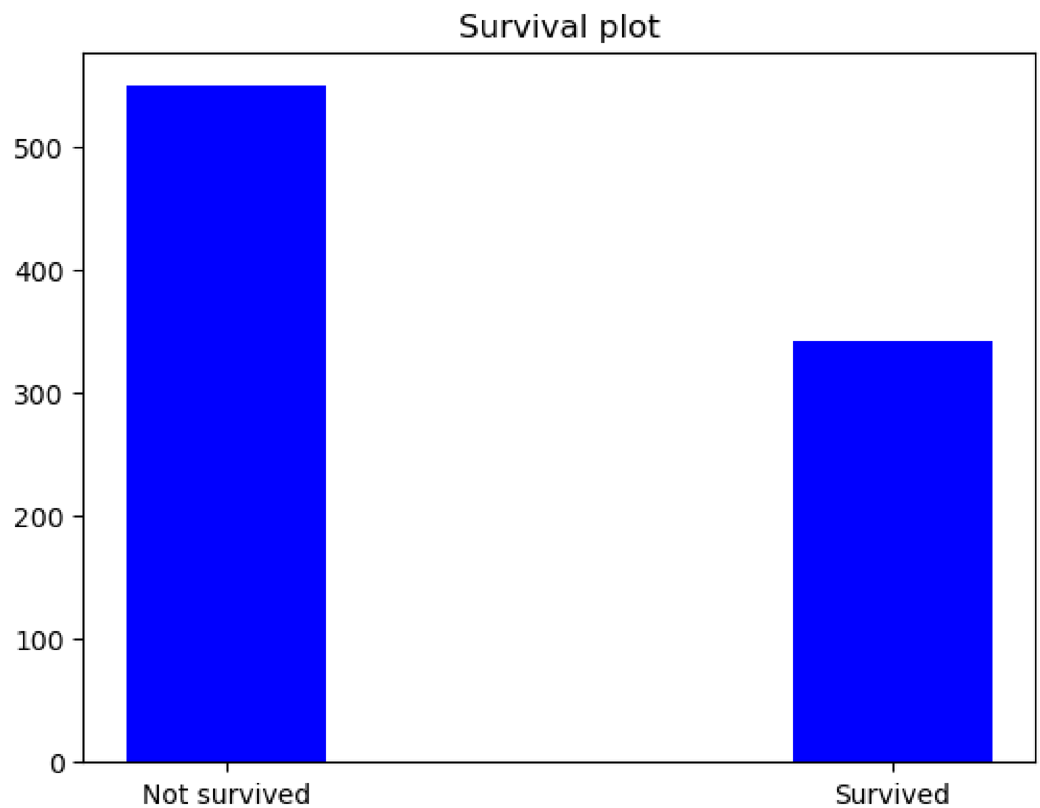
```
female = ((df_titanic['Gender']==1) & (df_titanic['Survived'])).sum()
count=[male,female]
plt.bar(gender,count)
plt.title("Gender wise survival plot")
```

[7]: Text(0.5, 1.0, 'Gender wise survival plot')




```
[8]: gender=['Not survived','Survived']
not_survived = ((df_titanic['Survived']==0)).sum()
Survive = ((df_titanic['Survived']==1)).sum()
count=[not_survived,Survive]
plt.bar(gender,count,color='blue',width=0.3)
plt.title('Survival plot')
```

[8]: Text(0.5, 1.0, 'Survival plot')



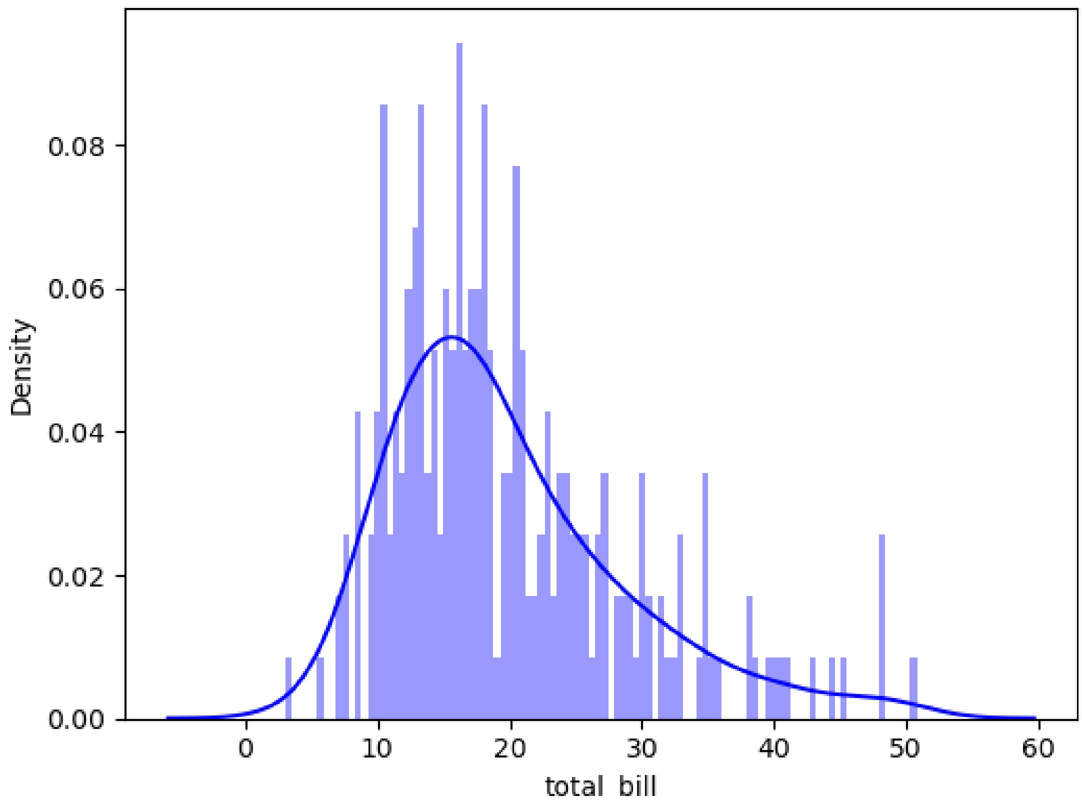

```
[9]: import seaborn as sns
     tips = sns.load_dataset('tips')
     tips
```

```
[9]:      total_bill   tip     sex smoker   day    time  size
     0         16.99  1.01  Female     No   Sun  Dinner     2
     1         10.34  1.66    Male     No   Sun  Dinner     3
     2         21.01  3.50    Male     No   Sun  Dinner     3
     3         23.68  3.31    Male     No   Sun  Dinner     2
     4         24.59  3.61  Female     No   Sun  Dinner     4
     ..          …     …      …      …    …      …    …
     239       29.03  5.92    Male     No   Sat  Dinner     3
     240       27.18  2.00  Female    Yes   Sat  Dinner     2
     241       22.67  2.00    Male    Yes   Sat  Dinner     2
     242       17.82  1.75    Male     No   Sat  Dinner     2
     243       18.78  3.00  Female     No  Thur  Dinner     2

     [244 rows x 7 columns]
```

```
[10]: sns.distplot(tips['total_bill'],bins=100,kde=True,hist=True,color='blue')
```

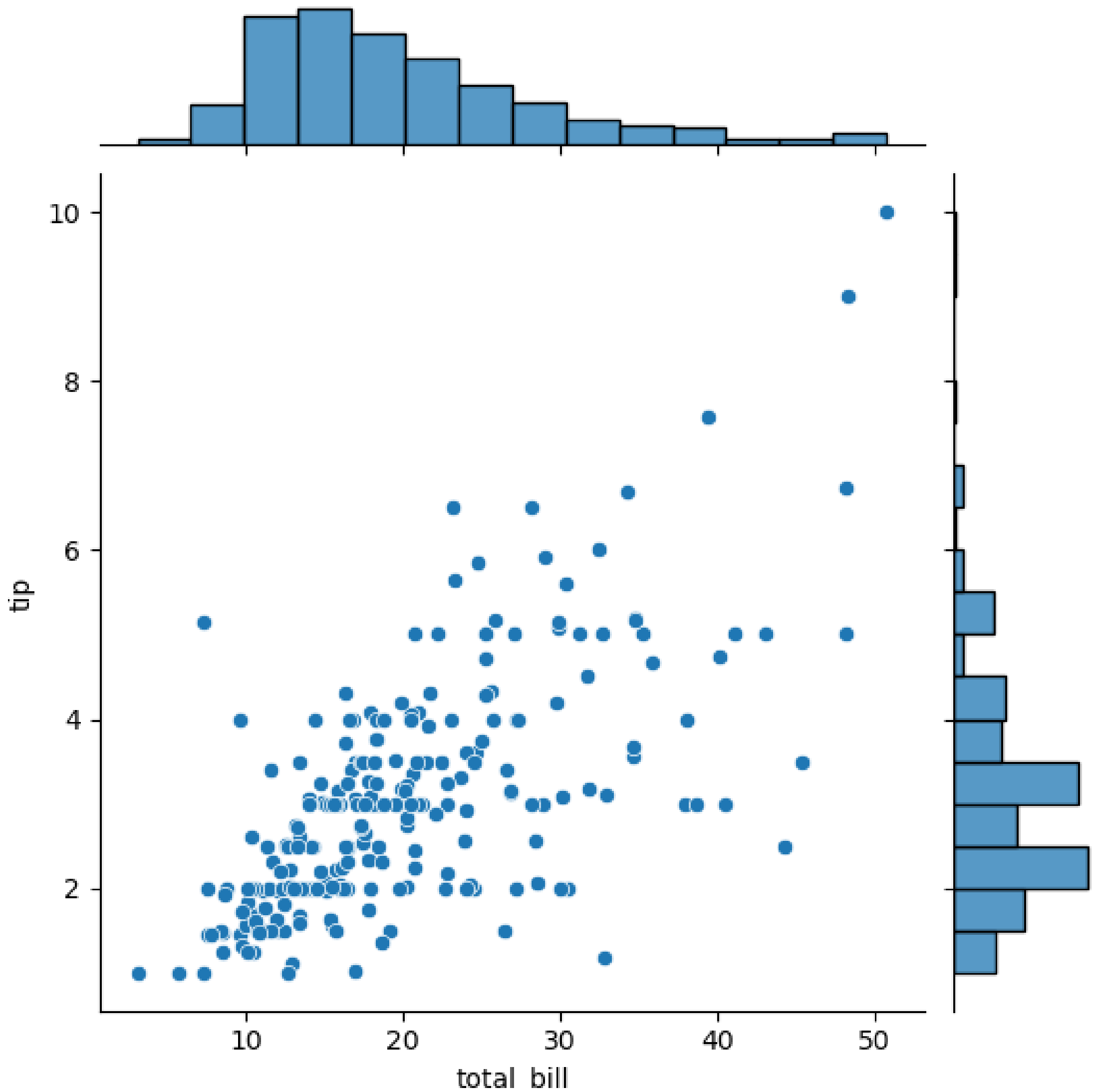
```
C:\Users\DELL\AppData\Local\Temp\ipykernel_7808\83553870.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(tips['total_bill'],bins=100,kde=True,hist=True,color='blue')
```

[10]: <Axes: xlabel='total_bill', ylabel='Density'>



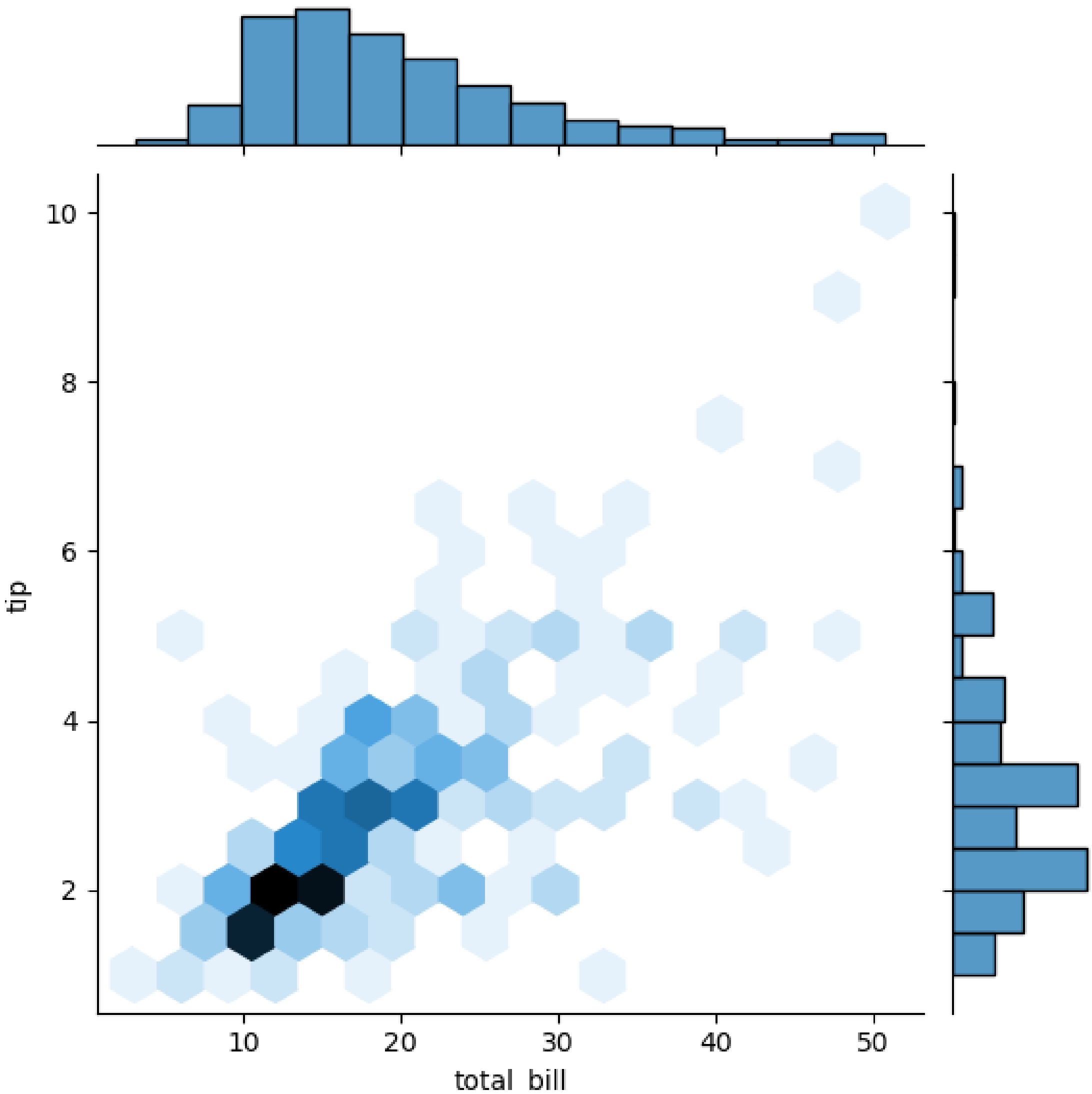
[11]:
```
sns.jointplot(x='total_bill',y='tip',data=tips)
```

[11]: <seaborn.axisgrid.JointGrid at 0x1c619664350>



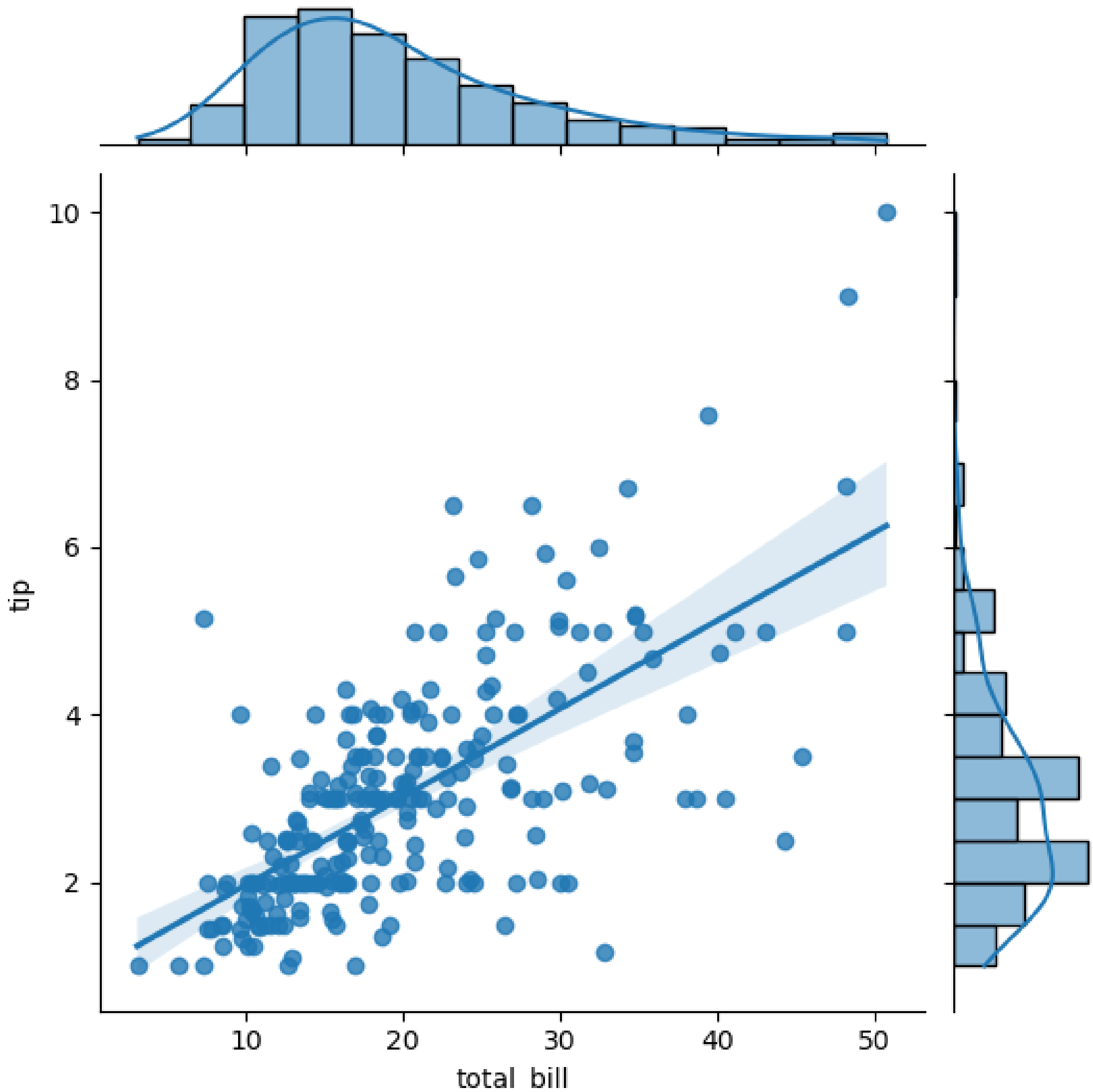
[12]:
```
sns.jointplot(x='total_bill',y='tip',data=tips,kind='hex')
```

[12]: <seaborn.axisgrid.JointGrid at 0x1c61a002890>



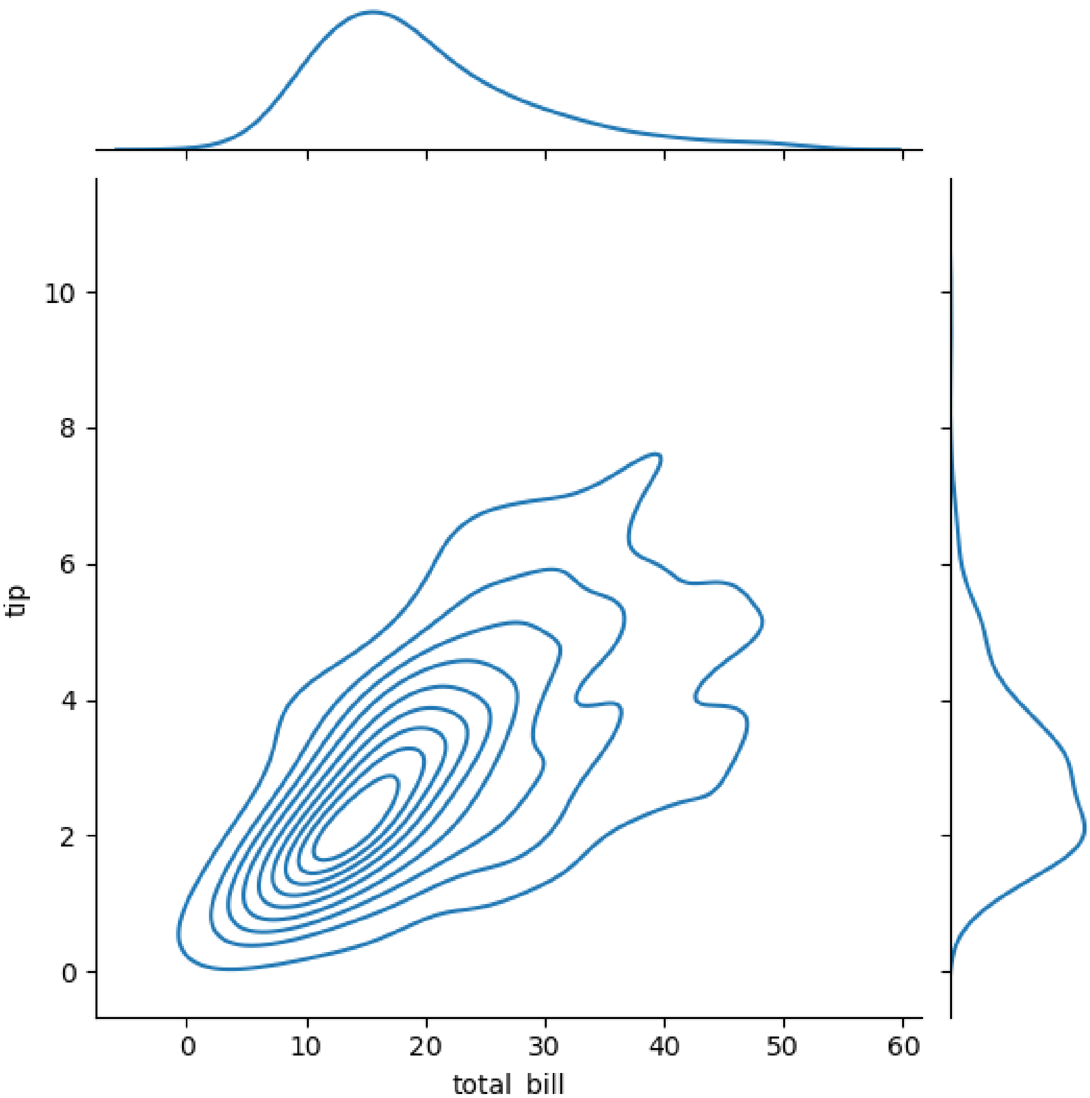
[13]:
```
sns.jointplot(x='total_bill',y='tip',data=tips,kind='reg')
```

[13]: <seaborn.axisgrid.JointGrid at 0x1c61a5431d0>




```
[14]: sns.jointplot(x='total_bill',y='tip',data=tips,kind='kde')
```

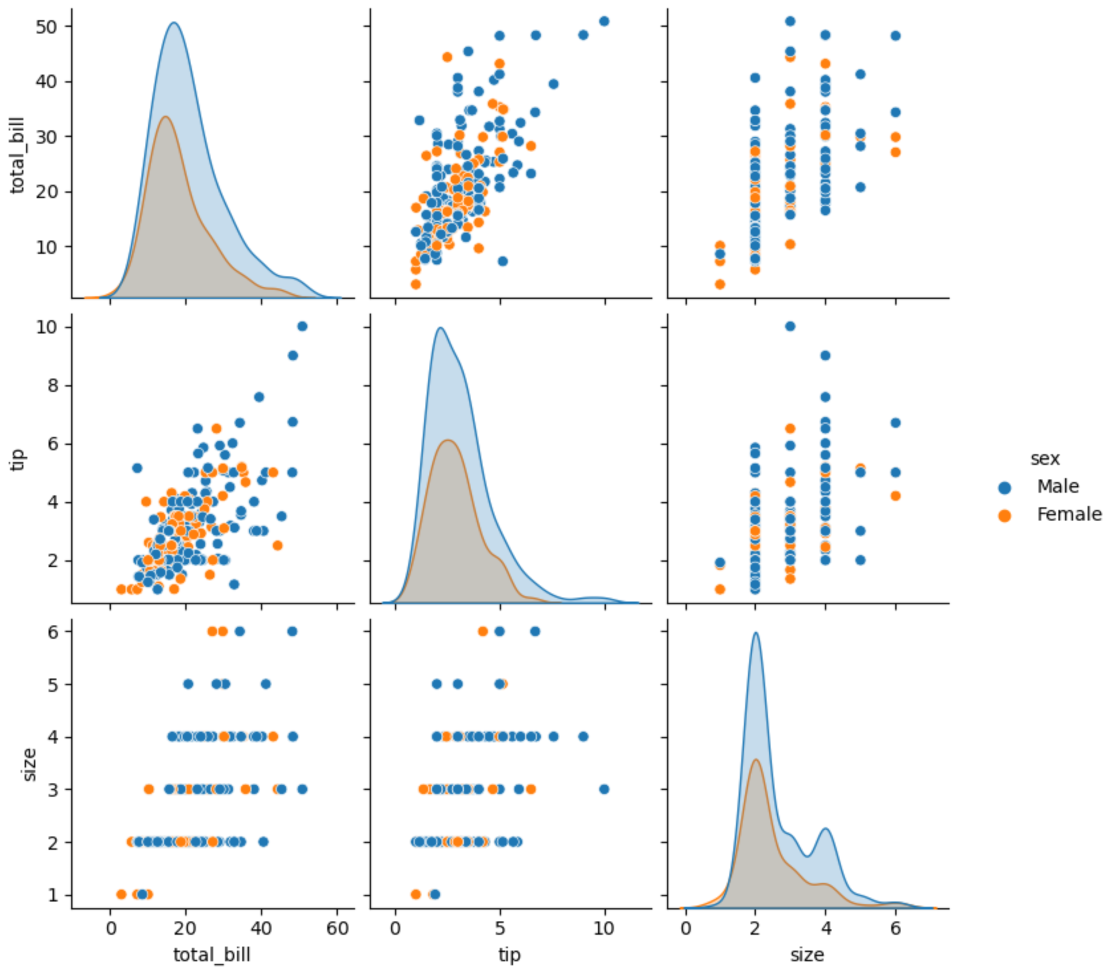
```
[14]: <seaborn.axisgrid.JointGrid at 0x1c61a965710>
```



[15]:
```
sns.pairplot(tips,hue='sex')
```

```
D:\anaconda\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure
layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

[15]:
```
<seaborn.axisgrid.PairGrid at 0x1c61c732910>
```



[16]:
```
#machine learning algorithms
#logistic regressiion is a statistical method used to model the relationship between a binary dependent variable and one or more independent variables in logistic regression
#the dependent variable is binary meaning it can only take two values 0 or 1
#The independent variable is can be either continuous or catagorical
```

[17]:
```
import numpy as np
from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score,confusion_matrix
```

[18]:
```
y_pred=np.array([0.3,0.6,0.8,0.2,0.4,0.9,0.1,0.7,0.5,0.6])
y_true=np.array([0,1,1,0,0,1,0,1,0,1])
```

[19]:
```
#accuracy
#accuracy meaures the percentage of correctly classified instancesout of all instances
```

```
accuracy=accuracy_score(y_true,np.round(y_pred))
accuracy
```

[19]: 1.0

[20]:
```
# precision
# measures the proportion of true +ve prediction out of all +ve predictions
# precision = true +ve / all +ve
precision = precision_score(y_true,np.round(y_pred))
precision
```

[20]: 1.0

[21]:
```
# recall
# measures the proportion of true +ve prediction out of all actual true +ve␣
↪cases
# recall = true +ve / actual +ve
recall = recall_score(y_true,np.round(y_pred))
recall
```

[21]: 1.0

[22]:
```
# f1_score
# It is the mean of precision and recall
f1_score = f1_score(y_true,np.round(y_pred))
f1_score
```

[22]: 1.0

[23]:
```
# confusion matrix
# It is a table that gives the performance of a classification model
# It shows true +ve , true -ve , false +ve , false -ve
matrix = confusion_matrix(y_true,np.round(y_pred))
matrix
```

[23]:
```
array([[5, 0],
       [0, 5]], dtype=int64)
```

[24]:
```
# Example - 1 for logistic regression
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
```

[25]:
```
iris = load_iris()
iris
```

```
[25]: {'data': array([[5.1, 3.5, 1.4, 0.2],
        [4.9, 3. , 1.4, 0.2],
        [4.7, 3.2, 1.3, 0.2],
        [4.6, 3.1, 1.5, 0.2],
        [5. , 3.6, 1.4, 0.2],
        [5.4, 3.9, 1.7, 0.4],
        [4.6, 3.4, 1.4, 0.3],
        [5. , 3.4, 1.5, 0.2],
        [4.4, 2.9, 1.4, 0.2],
        [4.9, 3.1, 1.5, 0.1],
        [5.4, 3.7, 1.5, 0.2],
        [4.8, 3.4, 1.6, 0.2],
        [4.8, 3. , 1.4, 0.1],
        [4.3, 3. , 1.1, 0.1],
        [5.8, 4. , 1.2, 0.2],
        [5.7, 4.4, 1.5, 0.4],
        [5.4, 3.9, 1.3, 0.4],
        [5.1, 3.5, 1.4, 0.3],
        [5.7, 3.8, 1.7, 0.3],
        [5.1, 3.8, 1.5, 0.3],
        [5.4, 3.4, 1.7, 0.2],
        [5.1, 3.7, 1.5, 0.4],
        [4.6, 3.6, 1. , 0.2],
        [5.1, 3.3, 1.7, 0.5],
        [4.8, 3.4, 1.9, 0.2],
        [5. , 3. , 1.6, 0.2],
        [5. , 3.4, 1.6, 0.4],
        [5.2, 3.5, 1.5, 0.2],
        [5.2, 3.4, 1.4, 0.2],
        [4.7, 3.2, 1.6, 0.2],
        [4.8, 3.1, 1.6, 0.2],
        [5.4, 3.4, 1.5, 0.4],
        [5.2, 4.1, 1.5, 0.1],
        [5.5, 4.2, 1.4, 0.2],
        [4.9, 3.1, 1.5, 0.2],
        [5. , 3.2, 1.2, 0.2],
        [5.5, 3.5, 1.3, 0.2],
        [4.9, 3.6, 1.4, 0.1],
        [4.4, 3. , 1.3, 0.2],
        [5.1, 3.4, 1.5, 0.2],
        [5. , 3.5, 1.3, 0.3],
        [4.5, 2.3, 1.3, 0.3],
        [4.4, 3.2, 1.3, 0.2],
        [5. , 3.5, 1.6, 0.6],
        [5.1, 3.8, 1.9, 0.4],
        [4.8, 3. , 1.4, 0.3],
        [5.1, 3.8, 1.6, 0.2],
```

```
[4.6, 3.2, 1.4, 0.2],
[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2],
[7. , 3.2, 4.7, 1.4],
[6.4, 3.2, 4.5, 1.5],
[6.9, 3.1, 4.9, 1.5],
[5.5, 2.3, 4. , 1.3],
[6.5, 2.8, 4.6, 1.5],
[5.7, 2.8, 4.5, 1.3],
[6.3, 3.3, 4.7, 1.6],
[4.9, 2.4, 3.3, 1. ],
[6.6, 2.9, 4.6, 1.3],
[5.2, 2.7, 3.9, 1.4],
[5. , 2. , 3.5, 1. ],
[5.9, 3. , 4.2, 1.5],
[6. , 2.2, 4. , 1. ],
[6.1, 2.9, 4.7, 1.4],
[5.6, 2.9, 3.6, 1.3],
[6.7, 3.1, 4.4, 1.4],
[5.6, 3. , 4.5, 1.5],
[5.8, 2.7, 4.1, 1. ],
[6.2, 2.2, 4.5, 1.5],
[5.6, 2.5, 3.9, 1.1],
[5.9, 3.2, 4.8, 1.8],
[6.1, 2.8, 4. , 1.3],
[6.3, 2.5, 4.9, 1.5],
[6.1, 2.8, 4.7, 1.2],
[6.4, 2.9, 4.3, 1.3],
[6.6, 3. , 4.4, 1.4],
[6.8, 2.8, 4.8, 1.4],
[6.7, 3. , 5. , 1.7],
[6. , 2.9, 4.5, 1.5],
[5.7, 2.6, 3.5, 1. ],
[5.5, 2.4, 3.8, 1.1],
[5.5, 2.4, 3.7, 1. ],
[5.8, 2.7, 3.9, 1.2],
[6. , 2.7, 5.1, 1.6],
[5.4, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[6.7, 3.1, 4.7, 1.5],
[6.3, 2.3, 4.4, 1.3],
[5.6, 3. , 4.1, 1.3],
[5.5, 2.5, 4. , 1.3],
[5.5, 2.6, 4.4, 1.2],
[6.1, 3. , 4.6, 1.4],
[5.8, 2.6, 4. , 1.2],
[5. , 2.3, 3.3, 1. ],
```

```
[5.6, 2.7, 4.2, 1.3],
[5.7, 3. , 4.2, 1.2],
[5.7, 2.9, 4.2, 1.3],
[6.2, 2.9, 4.3, 1.3],
[5.1, 2.5, 3. , 1.1],
[5.7, 2.8, 4.1, 1.3],
[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
[6.9, 3.2, 5.7, 2.3],
[5.6, 2.8, 4.9, 2. ],
[7.7, 2.8, 6.7, 2. ],
[6.3, 2.7, 4.9, 1.8],
[6.7, 3.3, 5.7, 2.1],
[7.2, 3.2, 6. , 1.8],
[6.2, 2.8, 4.8, 1.8],
[6.1, 3. , 4.9, 1.8],
[6.4, 2.8, 5.6, 2.1],
[7.2, 3. , 5.8, 1.6],
[7.4, 2.8, 6.1, 1.9],
[7.9, 3.8, 6.4, 2. ],
[6.4, 2.8, 5.6, 2.2],
[6.3, 2.8, 5.1, 1.5],
[6.1, 2.6, 5.6, 1.4],
[7.7, 3. , 6.1, 2.3],
[6.3, 3.4, 5.6, 2.4],
[6.4, 3.1, 5.5, 1.8],
[6. , 3. , 4.8, 1.8],
[6.9, 3.1, 5.4, 2.1],
[6.7, 3.1, 5.6, 2.4],
```

```
        [6.9, 3.1, 5.1, 2.3],
        [5.8, 2.7, 5.1, 1.9],
        [6.8, 3.2, 5.9, 2.3],
        [6.7, 3.3, 5.7, 2.5],
        [6.7, 3. , 5.2, 2.3],
        [6.3, 2.5, 5. , 1.9],
        [6.5, 3. , 5.2, 2. ],
        [6.2, 3.4, 5.4, 2.3],
        [5.9, 3. , 5.1, 1.8]]),
 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]),
 'frame': None,
 'target_names': array(['setosa', 'versicolor', 'virginica'], dtype='<U10'),
 'DESCR': '.. _iris_dataset:\n\nIris plants
dataset\n--------------------\n\n**Data Set Characteristics:**\n\n    :Number of
Instances: 150 (50 in each of three classes)\n    :Number of Attributes: 4
numeric, predictive attributes and the class\n    :Attribute Information:\n
- sepal length in cm\n        - sepal width in cm\n        - petal length in
cm\n        - petal width in cm\n        - class:\n                - Iris-
Setosa\n                - Iris-Versicolour\n                - Iris-Virginica\n
\n    :Summary Statistics:\n\n    ============== ==== ==== ======= =====
====================\n                    Min  Max   Mean    SD   Class
Correlation\n    ============== ==== ==== ======= ===== ====================\n
sepal length:   4.3  7.9   5.84   0.83    0.7826\n    sepal width:    2.0  4.4
3.05   0.43   -0.4194\n    petal length:   1.0  6.9   3.76   1.76    0.9490
(high!)\n    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)\n
============== ==== ==== ======= ===== ====================\n\n    :Missing
Attribute Values: None\n    :Class Distribution: 33.3% for each of 3 classes.\n
:Creator: R.A. Fisher\n    :Donor: Michael Marshall
(MARSHALL%PLU@io.arc.nasa.gov)\n    :Date: July, 1988\n\nThe famous Iris
database, first used by Sir R.A. Fisher. The dataset is taken\nfrom Fisher\'s
paper. Note that it\'s the same as in R, but not as in the UCI\nMachine Learning
Repository, which has two wrong data points.\n\nThis is perhaps the best known
database to be found in the\npattern recognition literature.  Fisher\'s paper is
a classic in the field and\nis referenced frequently to this day.  (See Duda &
Hart, for example.)  The\ndata set contains 3 classes of 50 instances each,
where each class refers to a\ntype of iris plant.  One class is linearly
separable from the other 2; the\nlatter are NOT linearly separable from each
other.\n\n.. topic:: References\n\n   - Fisher, R.A. "The use of multiple
measurements in taxonomic problems"\n     Annual Eugenics, 7, Part II, 179-188
(1936); also in "Contributions to\n     Mathematical Statistics" (John Wiley,
```

```
NY, 1950).\n   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and
Scene Analysis.\n     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See
page 218.\n   - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New
System\n     Structure and Classification Rule for Recognition in Partially
Exposed\n     Environments".  IEEE Transactions on Pattern Analysis and
Machine\n     Intelligence, Vol. PAMI-2, No. 1, 67-71.\n   - Gates, G.W. (1972)
"The Reduced Nearest Neighbor Rule".  IEEE Transactions\n     on Information
Theory, May 1972, 431-433.\n   - See also: 1988 MLC Proceedings, 54-64.
Cheeseman et al"s AUTOCLASS II\n     conceptual clustering system finds 3
classes in the data.\n   - Many, many more …',
 'feature_names': ['sepal length (cm)',
  'sepal width (cm)',
  'petal length (cm)',
  'petal width (cm)'],
 'filename': 'iris.csv',
 'data_module': 'sklearn.datasets.data'}
```

[26]:
```python
iris = load_iris()
iris_df = pd.DataFrame(data = iris.data, columns = iris.feature_names)
iris_df
```

[26]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 …                 …                  …                 …
145                6.7               3.0                5.2               2.3
146                6.3               2.5                5.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
149                5.9               3.0                5.1               1.8

[150 rows x 4 columns]
```

[27]:
```python
iris = load_iris()
iris_df = pd.DataFrame(data = iris.data, columns = iris.feature_names)
iris_df['target'] = iris.target
iris_df['target_names'] = iris.target_names[iris.target]
iris_df
```

[27]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
```

```
4             5.0           3.6           1.4           0.2
..            …             …             …             …
145           6.7           3.0           5.2           2.3
146           6.3           2.5           5.0           1.9
147           6.5           3.0           5.2           2.0
148           6.2           3.4           5.4           2.3
149           5.9           3.0           5.1           1.8

     target target_names
0         0        setosa
1         0        setosa
2         0        setosa
3         0        setosa
4         0        setosa
..      …           …
145       2     virginica
146       2     virginica
147       2     virginica
148       2     virginica
149       2     virginica

[150 rows x 6 columns]
```

[28]:
```
x = iris.data
x
```

[28]:
```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
       [5.4, 3.4, 1.7, 0.2],
```

```
[5.1, 3.7, 1.5, 0.4],
[4.6, 3.6, 1. , 0.2],
[5.1, 3.3, 1.7, 0.5],
[4.8, 3.4, 1.9, 0.2],
[5. , 3. , 1.6, 0.2],
[5. , 3.4, 1.6, 0.4],
[5.2, 3.5, 1.5, 0.2],
[5.2, 3.4, 1.4, 0.2],
[4.7, 3.2, 1.6, 0.2],
[4.8, 3.1, 1.6, 0.2],
[5.4, 3.4, 1.5, 0.4],
[5.2, 4.1, 1.5, 0.1],
[5.5, 4.2, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.2],
[5. , 3.2, 1.2, 0.2],
[5.5, 3.5, 1.3, 0.2],
[4.9, 3.6, 1.4, 0.1],
[4.4, 3. , 1.3, 0.2],
[5.1, 3.4, 1.5, 0.2],
[5. , 3.5, 1.3, 0.3],
[4.5, 2.3, 1.3, 0.3],
[4.4, 3.2, 1.3, 0.2],
[5. , 3.5, 1.6, 0.6],
[5.1, 3.8, 1.9, 0.4],
[4.8, 3. , 1.4, 0.3],
[5.1, 3.8, 1.6, 0.2],
[4.6, 3.2, 1.4, 0.2],
[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2],
[7. , 3.2, 4.7, 1.4],
[6.4, 3.2, 4.5, 1.5],
[6.9, 3.1, 4.9, 1.5],
[5.5, 2.3, 4. , 1.3],
[6.5, 2.8, 4.6, 1.5],
[5.7, 2.8, 4.5, 1.3],
[6.3, 3.3, 4.7, 1.6],
[4.9, 2.4, 3.3, 1. ],
[6.6, 2.9, 4.6, 1.3],
[5.2, 2.7, 3.9, 1.4],
[5. , 2. , 3.5, 1. ],
[5.9, 3. , 4.2, 1.5],
[6. , 2.2, 4. , 1. ],
[6.1, 2.9, 4.7, 1.4],
[5.6, 2.9, 3.6, 1.3],
[6.7, 3.1, 4.4, 1.4],
[5.6, 3. , 4.5, 1.5],
[5.8, 2.7, 4.1, 1. ],
```

```
[6.2, 2.2, 4.5, 1.5],
[5.6, 2.5, 3.9, 1.1],
[5.9, 3.2, 4.8, 1.8],
[6.1, 2.8, 4. , 1.3],
[6.3, 2.5, 4.9, 1.5],
[6.1, 2.8, 4.7, 1.2],
[6.4, 2.9, 4.3, 1.3],
[6.6, 3. , 4.4, 1.4],
[6.8, 2.8, 4.8, 1.4],
[6.7, 3. , 5. , 1.7],
[6. , 2.9, 4.5, 1.5],
[5.7, 2.6, 3.5, 1. ],
[5.5, 2.4, 3.8, 1.1],
[5.5, 2.4, 3.7, 1. ],
[5.8, 2.7, 3.9, 1.2],
[6. , 2.7, 5.1, 1.6],
[5.4, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[6.7, 3.1, 4.7, 1.5],
[6.3, 2.3, 4.4, 1.3],
[5.6, 3. , 4.1, 1.3],
[5.5, 2.5, 4. , 1.3],
[5.5, 2.6, 4.4, 1.2],
[6.1, 3. , 4.6, 1.4],
[5.8, 2.6, 4. , 1.2],
[5. , 2.3, 3.3, 1. ],
[5.6, 2.7, 4.2, 1.3],
[5.7, 3. , 4.2, 1.2],
[5.7, 2.9, 4.2, 1.3],
[6.2, 2.9, 4.3, 1.3],
[5.1, 2.5, 3. , 1.1],
[5.7, 2.8, 4.1, 1.3],
[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
```

```
       [6.4, 3.2, 5.3, 2.3],
       [6.5, 3. , 5.5, 1.8],
       [7.7, 3.8, 6.7, 2.2],
       [7.7, 2.6, 6.9, 2.3],
       [6. , 2.2, 5. , 1.5],
       [6.9, 3.2, 5.7, 2.3],
       [5.6, 2.8, 4.9, 2. ],
       [7.7, 2.8, 6.7, 2. ],
       [6.3, 2.7, 4.9, 1.8],
       [6.7, 3.3, 5.7, 2.1],
       [7.2, 3.2, 6. , 1.8],
       [6.2, 2.8, 4.8, 1.8],
       [6.1, 3. , 4.9, 1.8],
       [6.4, 2.8, 5.6, 2.1],
       [7.2, 3. , 5.8, 1.6],
       [7.4, 2.8, 6.1, 1.9],
       [7.9, 3.8, 6.4, 2. ],
       [6.4, 2.8, 5.6, 2.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.1, 2.6, 5.6, 1.4],
       [7.7, 3. , 6.1, 2.3],
       [6.3, 3.4, 5.6, 2.4],
       [6.4, 3.1, 5.5, 1.8],
       [6. , 3. , 4.8, 1.8],
       [6.9, 3.1, 5.4, 2.1],
       [6.7, 3.1, 5.6, 2.4],
       [6.9, 3.1, 5.1, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [6.8, 3.2, 5.9, 2.3],
       [6.7, 3.3, 5.7, 2.5],
       [6.7, 3. , 5.2, 2.3],
       [6.3, 2.5, 5. , 1.9],
       [6.5, 3. , 5.2, 2. ],
       [6.2, 3.4, 5.4, 2.3],
       [5.9, 3. , 5.1, 1.8]])
```

[29]:

```
y = iris.target
y
```

[29]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

[30]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.
 ↪2,random_state=101)
x_train.shape
```

[30]:
```
(120, 4)
```

[31]:
```
y_test.shape
```

[31]:
```
(30,)
```

[32]:
```
# To train the algorithm
clf = LogisticRegression()
```

[33]:
```
clf.fit(x_train,y_train)
```

[33]:
```
LogisticRegression()
```

[34]:
```
y_pred = clf.predict(x_test)
y_pred
```

[34]:
```
array([0, 0, 0, 2, 1, 2, 1, 1, 2, 0, 2, 0, 0, 2, 2, 1, 1, 1, 0, 2, 1, 0,
       1, 1, 1, 1, 1, 2, 0, 0])
```

[35]:
```
accuracy = accuracy_score(y_test,y_pred)
accuracy
```

[35]:
```
1.0
```

[36]:
```
data = pd.read_csv('bmi.csv')
data
```

[36]:
```
     Gender  Height  Weight  Index
0      Male     174      96      4
1      Male     189      87      2
2    Female     185     110      4
3    Female     195     104      3
4      Male     149      61      3
..      …       …       …      …
495  Female     150     153      5
496  Female     184     121      4
497  Female     141     136      5
498    Male     150      95      5
499    Male     173     131      5

[500 rows x 4 columns]
```

[37]:
```
# project - 1
print('Male',(data['Gender'] == 'Male').sum())
print('Female',(data['Gender'] == 'Female').sum())
```

```
Male 245
Female 255
```

[38]:
```
data['Gender'].replace({'Male':0,'Female':1},inplace=True)
data
```

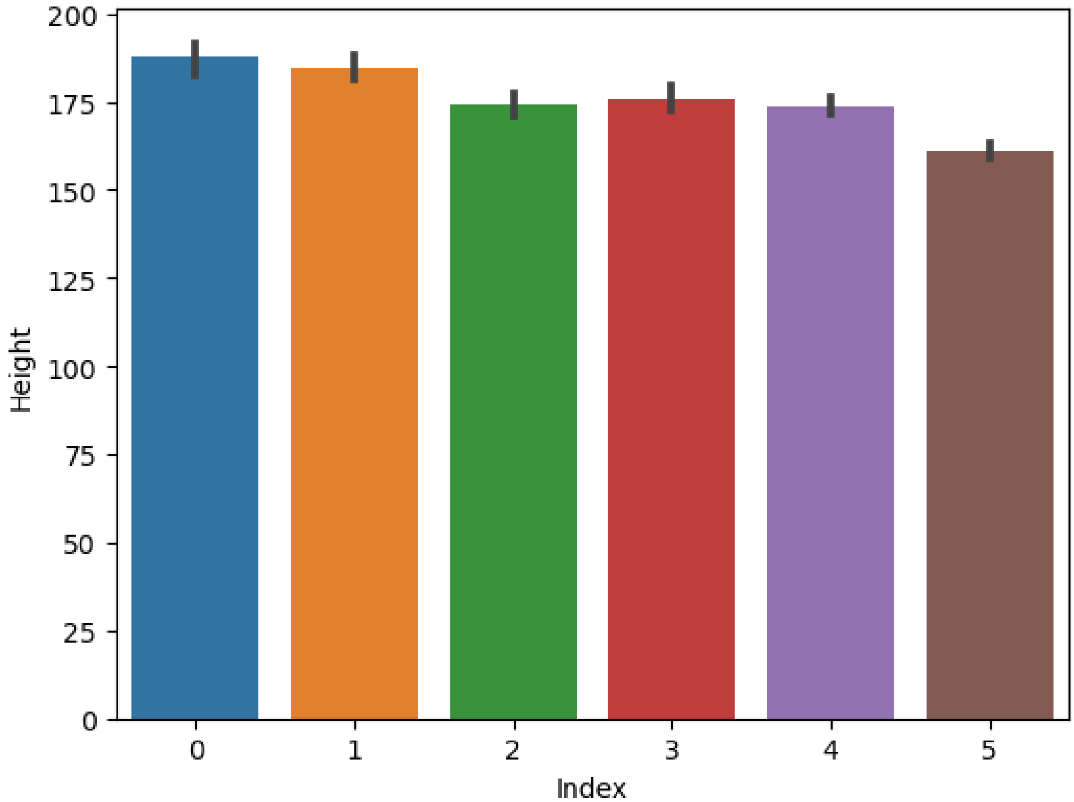
[38]:

	Gender	Height	Weight	Index
0	0	174	96	4
1	0	189	87	2
2	1	185	110	4
3	1	195	104	3
4	0	149	61	3
..	…	…	…	…
495	1	150	153	5
496	1	184	121	4
497	1	141	136	5
498	0	150	95	5
499	0	173	131	5

[500 rows x 4 columns]

[39]:
```
sns.barplot(x='Index',y='Height',data=data)
```

[39]: <Axes: xlabel='Index', ylabel='Height'>



[40]:
```
y=data['Index']
y
```

[40]:
```
0      4
1      2
2      4
3      3
4      3
      ..
495    5
496    4
497    5
498    5
499    5
Name: Index, Length: 500, dtype: int64
```

[41]:
```
x=data[['Gender','Height','Weight']]
y=data.Index
```

[42]:
```
from sklearn.model_selection import train_test_split
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.3,␣
↪random_state=101)
x_train.shape
```

[42]: (350, 3)

[43]:
```
# To train the dataset
from sklearn.linear_model import LogisticRegression
log_model = LogisticRegression()
log_model.fit(x_train,y_train)
```

```
D:\anaconda\Lib\site-packages\sklearn\linear_model\_logistic.py:460:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[43]: LogisticRegression()

[44]:
```
pred=log_model.predict(x_test)
pred
```

[44]:
```
array([5, 4, 5, 2, 3, 3, 1, 4, 5, 4, 5, 3, 5, 3, 5, 2, 5, 5, 4, 5, 4, 5,
       4, 5, 2, 4, 3, 4, 5, 2, 5, 4, 4, 5, 4, 5, 0, 2, 5, 4, 3, 5, 4, 5,
       5, 5, 4, 2, 1, 3, 5, 5, 5, 4, 2, 2, 2, 5, 4, 5, 3, 3, 5, 5, 3, 5,
       4, 4, 4, 5, 5, 4, 5, 5, 1, 4, 3, 3, 5, 2, 2, 2, 5, 3, 5, 5, 5, 5,
       5, 2, 3, 5, 2, 4, 4, 0, 4, 5, 5, 5, 2, 4, 4, 5, 2, 3, 5, 5, 1, 1,
       5, 4, 5, 3, 5, 5, 5, 2, 4, 4, 5, 4, 4, 5, 4, 5, 5, 1, 4, 2, 5, 1,
       5, 4, 3, 5, 5, 5, 3, 5, 5, 4, 3, 5, 1, 5, 2, 4, 5, 5], dtype=int64)
```

[45]:
```
(data['Gender']==0).sum()
```

[45]: 245

[46]:
```
(data['Gender']==0).sum()
```

[46]: 245

[47]:
```
data = pd.get_dummies(data,columns=['Gender'],dtype=int,drop_first=True)
data
```

[47]:

```
      Height  Weight  Index  Gender_1
0        174      96      4         0
1        189      87      2         0
2        185     110      4         1
3        195     104      3         1
4        149      61      3         0
..       …       …     …       …
495      150     153      5         1
496      184     121      4         1
497      141     136      5         1
498      150      95      5         0
499      173     131      5         0

[500 rows x 4 columns]
```

[48]:

```python
from sklearn.metrics import accuracy_score
accuracy_score(y_test,pred)
```

[48]:

```
0.7666666666666667
```

[49]:

```python
# LInear Tegression
# data
# x(week) y(sales in thousand)
# -------------------------------
# 1                   1.2
# 2                   1.8
# 3                   2.5
# 4                   3.2
# 5                   3.4
# Linear regression formula ---> y = a0+a1*x
# a1 ---> (mean(x*y)) - (mean(x)*(mean(y))) / (mean(x^2) - mean(x)^2)
# a0 = mean(y) - a1*mean(x)
#              x            y             x^2           x^y
#              1            1.2           1             1.2
#              2            1.8           4             3.6
#              3            2.5           9             7.5
#              4            3.2           16            12.8
#              5            3.4           25            17
# ----------------------------------------------------------------------
# sum          15           12.5          55             44.1
# average      3            2.5           11             8.82
# a1 = 0.66
# a0 = 0.52
# value for 1st week
# y = 1.18
# value for 2nd week
# y = 2.36
```

```
# value for 3rd week
# y = 3.54
# value for 4th week
# y = 4.72
# value for 5th week
# y = 5.9
```

[50]:
```
df = pd.read_csv('Linear_regr_Salary_dataset.csv')
df.head()
```

[50]:

	Unnamed: 0	YearsExperience	Salary
0	0	1.2	39344.0
1	1	1.4	46206.0
2	2	1.6	37732.0
3	3	2.1	43526.0
4	4	2.3	39892.0

[51]:
```
df.shape
```

[51]:
```
(30, 3)
```

[52]:
```
df.isnull().sum()
```

[52]:
```
Unnamed: 0         0
YearsExperience    0
Salary             0
dtype: int64
```

[53]:
```
x=df[['YearsExperience']]
x
y=df['Salary']
y
```

[53]:
```
0      39344.0
1      46206.0
2      37732.0
3      43526.0
4      39892.0
5      56643.0
6      60151.0
7      54446.0
8      64446.0
9      57190.0
10     63219.0
11     55795.0
12     56958.0
13     57082.0
```

```
14     61112.0
15     67939.0
16     66030.0
17     83089.0
18     81364.0
19     93941.0
20     91739.0
21     98274.0
22    101303.0
23    113813.0
24    109432.0
25    105583.0
26    116970.0
27    112636.0
28    122392.0
29    121873.0
Name: Salary, dtype: float64
```

[54]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.
 ↪4,random_state=101)
```

[55]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

[56]:
```
model
```

[56]:
```
LinearRegression()
```

[57]:
```
model.fit(x_train,y_train)
```

[57]:
```
LinearRegression()
```

[58]:
```
y_pred=model.predict(x_test)
y_pred
```

[58]:
```
array([ 91487.32335452, 109831.9872175 ,  56729.0128773 ,  82797.74573522,
        40315.36626306, 118521.56483681, 117556.05621244,  75073.67674028,
       112728.5130906 , 125280.12520738,  63487.57324787,  45142.90938489])
```

[59]:
```
import numpy as np
inputdata = [[4.5]]
prediction = model.predict(inputdata)
prediction
```

```
D:\anaconda\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have
valid feature names, but LinearRegression was fitted with feature names
```

```
warnings.warn(
```

[59]: array([68315.11636971])

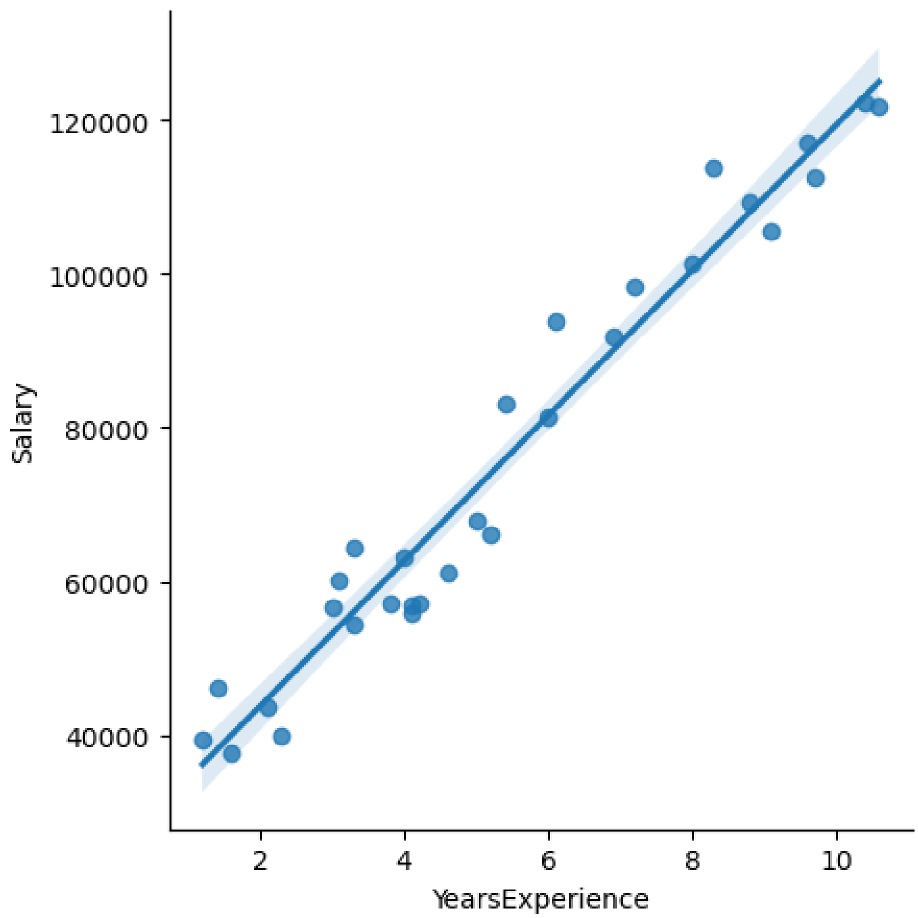
[60]:
```
from sklearn.metrics import mean_squared_error
mse=mean_squared_error(y_test,y_pred)
mse
```

[60]: 16085205.26610922

[61]:
```
import seaborn as sns
sns.lmplot(x='YearsExperience',y='Salary',data = df)
```

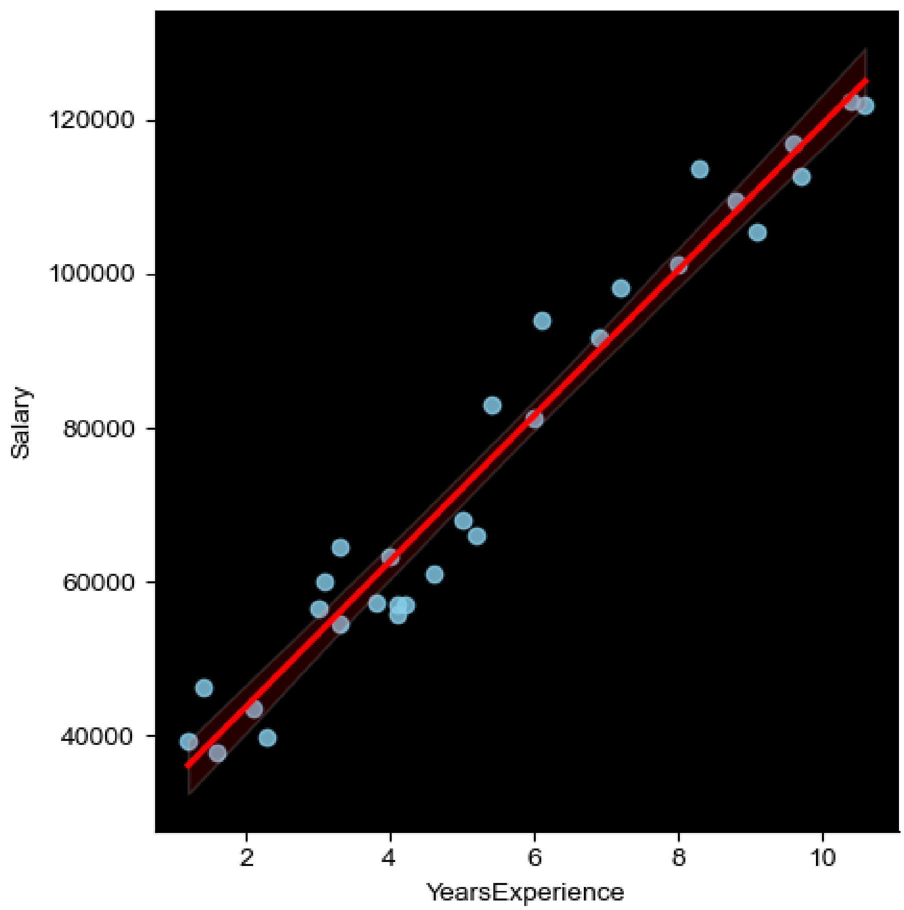
```
D:\anaconda\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure
layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

[61]: <seaborn.axisgrid.FacetGrid at 0x1c61e48bfd0>



[62]:
```
import matplotlib.pyplot as plt
sns.lmplot(x='YearsExperience',y='Salary',data = df,scatter_kws = {'color':
  ↪'skyblue'},line_kws = {'color':'red'})
sns.set_style('darkgrid')
ax = plt.gca()
plt.gca().set_facecolor('black')
```

```
D:\anaconda\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure
layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



[63]:
```
# Project - 3
df1 = pd.read_csv('LR_Student_Performance.csv')
df1
```

[63]:
```
   Hours Studied  Previous Scores Extracurricular Activities  Sleep Hours  \
0              7               99                        Yes            9
1              4               82                         No            4
```

```
2                  8            51                           Yes            7
3                  5            52                           Yes            5
4                  7            75                            No            8
…             …             …                            …          …
9995               1            49                           Yes            4
9996               7            64                           Yes            8
9997               6            83                           Yes            8
9998               9            97                           Yes            7
9999               7            74                            No            8

      Sample Question Papers Practiced  Performance Index
0                                    1               91.0
1                                    2               65.0
2                                    2               45.0
3                                    2               36.0
4                                    5               66.0
…                                …                 …
9995                                 2               23.0
9996                                 5               58.0
9997                                 5               74.0
9998                                 0               95.0
9999                                 1               64.0

[10000 rows x 6 columns]
```

[64]:
```python
df1.shape
```

[64]:
```
(10000, 6)
```

[65]:
```python
df.describe()
```

[65]:
```
       Unnamed: 0  YearsExperience         Salary
count   30.000000        30.000000      30.000000
mean    14.500000         5.413333   76004.000000
std      8.803408         2.837888   27414.429785
min      0.000000         1.200000   37732.000000
25%      7.250000         3.300000   56721.750000
50%     14.500000         4.800000   65238.000000
75%     21.750000         7.800000  100545.750000
max     29.000000        10.600000  122392.000000
```

[66]:
```python
df1.isnull().sum()
```

[66]:
```
Hours Studied                       0
Previous Scores                     0
Extracurricular Activities          0
Sleep Hours                         0
```

```
Sample Question Papers Practiced    0
Performance Index                   0
dtype: int64
```

[67]:
```
df.isna().sum()
```

[67]:
```
Unnamed: 0         0
YearsExperience    0
Salary             0
dtype: int64
```

[68]:
```
dup = df1.duplicated()
dup.sum()
```

[68]: 127

[69]:
```
df1.drop_duplicates(inplace=True)
df1
```

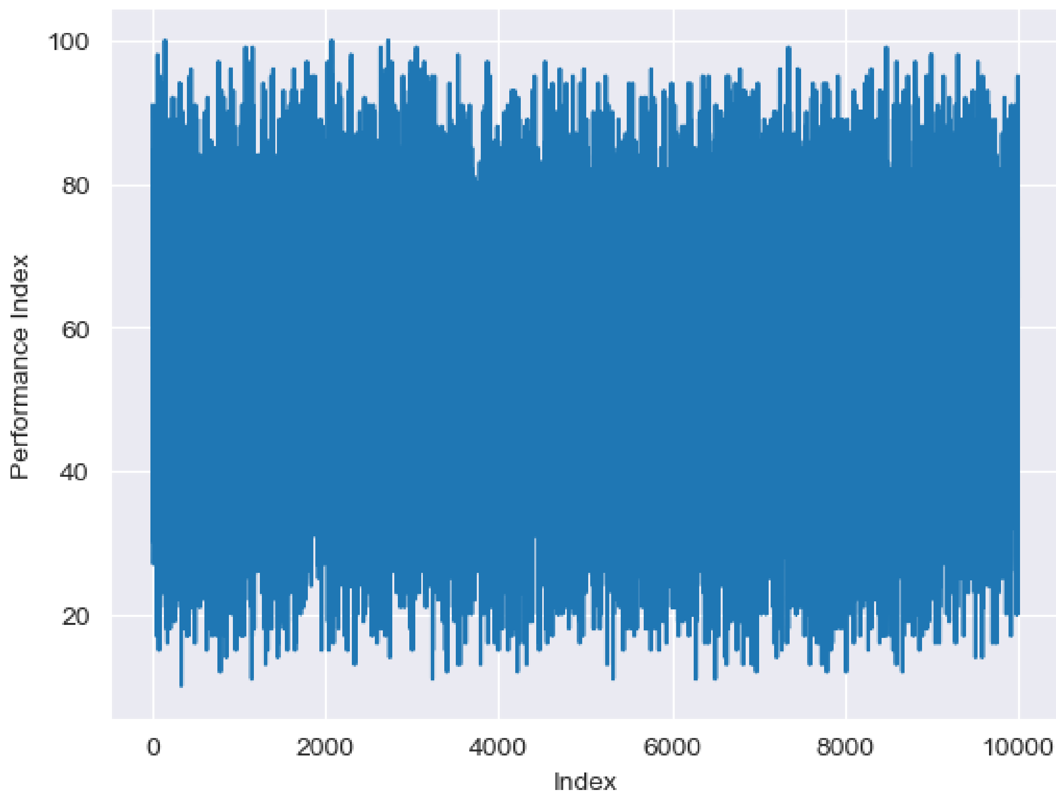
[69]:
```
      Hours Studied  Previous Scores Extracurricular Activities  Sleep Hours  \
0                 7               99                        Yes            9
1                 4               82                         No            4
2                 8               51                        Yes            7
3                 5               52                        Yes            5
4                 7               75                         No            8
…               …               …                          …           …
9995              1               49                        Yes            4
9996              7               64                        Yes            8
9997              6               83                        Yes            8
9998              9               97                        Yes            7
9999              7               74                         No            8

      Sample Question Papers Practiced  Performance Index
0                                    1               91.0
1                                    2               65.0
2                                    2               45.0
3                                    2               36.0
4                                    5               66.0
…                                  …                …
9995                                 2               23.0
9996                                 5               58.0
9997                                 5               74.0
9998                                 0               95.0
9999                                 1               64.0

[9873 rows x 6 columns]
```

[70]:
```
df1.shape
```

[70]: (9873, 6)

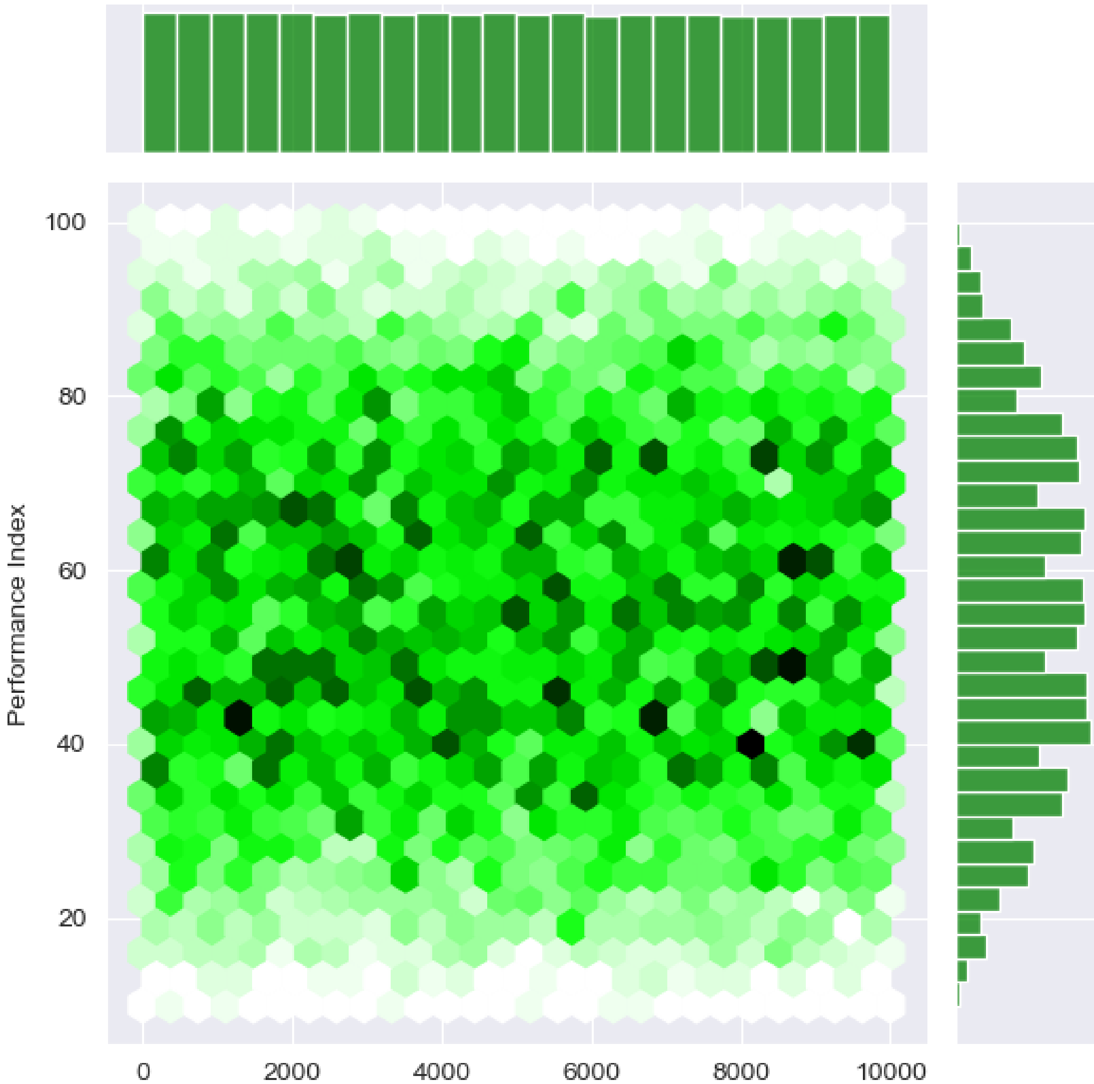
[71]:
```
response = df1['Performance Index']
response.dtype
plt.plot(response.index,response)
plt.xlabel('Index')
plt.ylabel('Performance Index')
```

[71]: Text(0, 0.5, 'Performance Index')



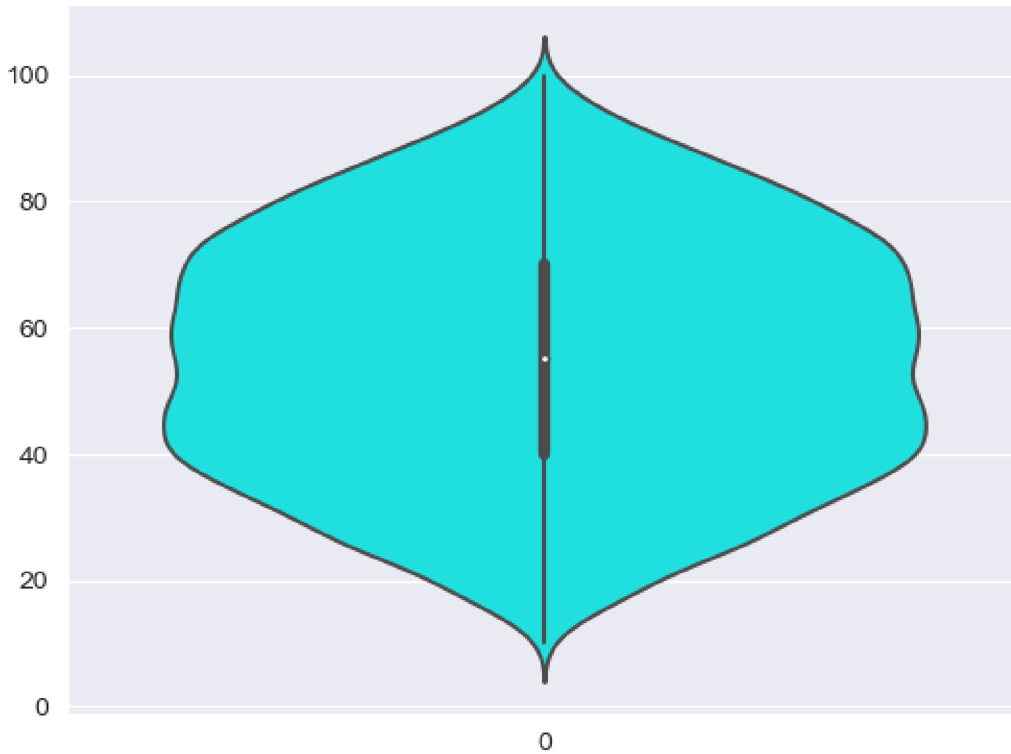
[72]:
```
sns.jointplot(x=response.index,y='Performance Index',data = df1,kind =␣
↪'hex',color='green')
```

[72]: <seaborn.axisgrid.JointGrid at 0x1c61e7a1f50>



[138]:
```
sns.violinplot(response,color='#00ffff')
```

[138]: <Axes: >



[74]:
```
ma = df1['Performance Index'].max()
ma
```

[74]: 100.0

[75]:
```
mi = df1['Performance Index'].min()
mi
```

[75]: 10.0

[76]:
```
(df1['Performance Index']==mi).sum()
```

[76]: 1

[77]:
```
(df1['Performance Index']==ma).sum()
```

[77]: 3

[78]:
```
df1['Hours Studied'].sum()
```

[78]: 49287

[79]:
```
df1['Hours Studied'].max()
```

[79]: 9

[80]:
```
df1['Hours Studied'].min()
```

[80]: 1

[81]:
```
val = pd.DataFrame(df1['Hours Studied']).value_counts()
val
```

[81]:
```
Hours Studied
1                1133
6                1122
7                1118
3                1110
9                1099
2                1077
8                1074
4                1071
5                1069
Name: count, dtype: int64
```

[82]:
```
df1['Hours Studied'].unique()
```

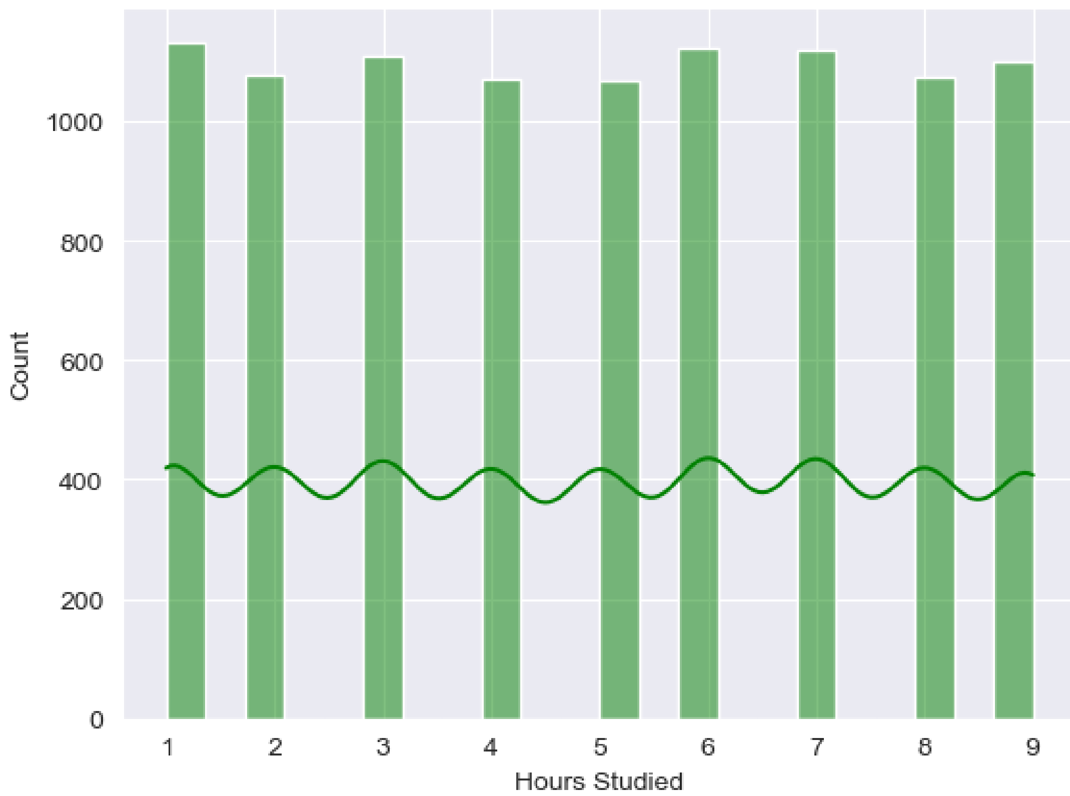
[82]: array([7, 4, 8, 5, 3, 6, 2, 1, 9], dtype=int64)

[83]:
```
from collections import Counter
dict(Counter(df1['Hours Studied']))
# method - 2
```

[83]:
```
{7: 1118,
 4: 1071,
 8: 1074,
 5: 1069,
 3: 1110,
 6: 1122,
 2: 1077,
 1: 1133,
 9: 1099}
```

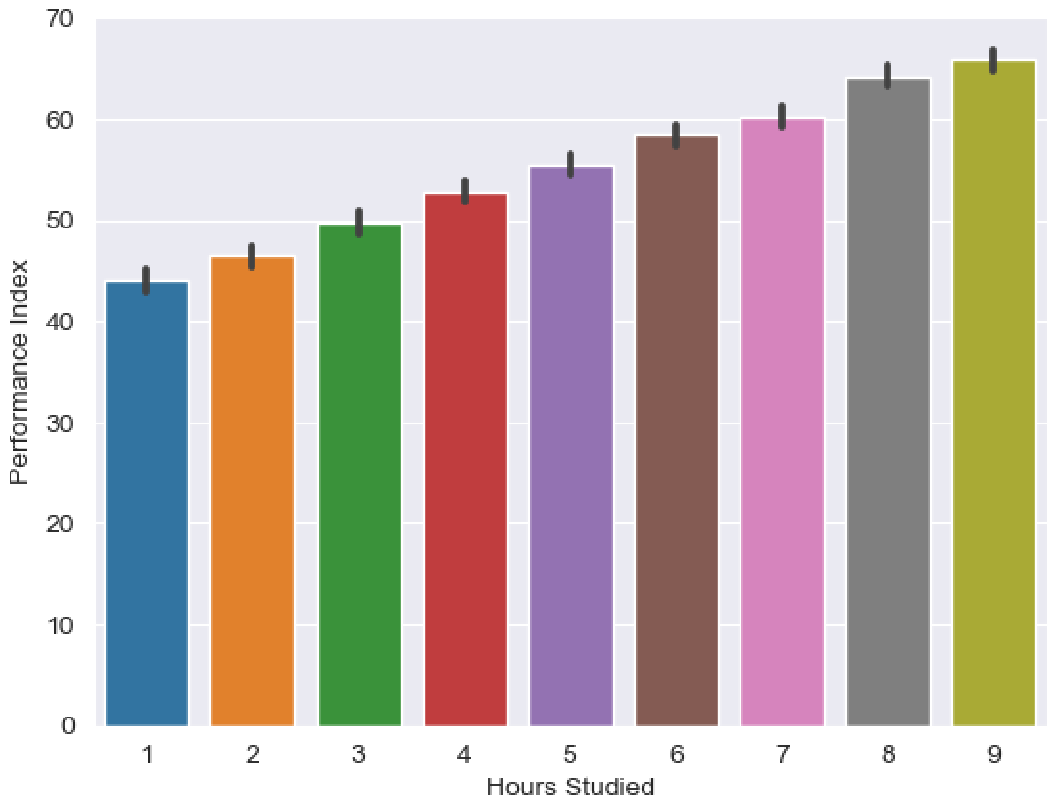
[84]:
```
sns.histplot(df1['Hours Studied'],color='green',kde=True)
```

[84]: <Axes: xlabel='Hours Studied', ylabel='Count'>



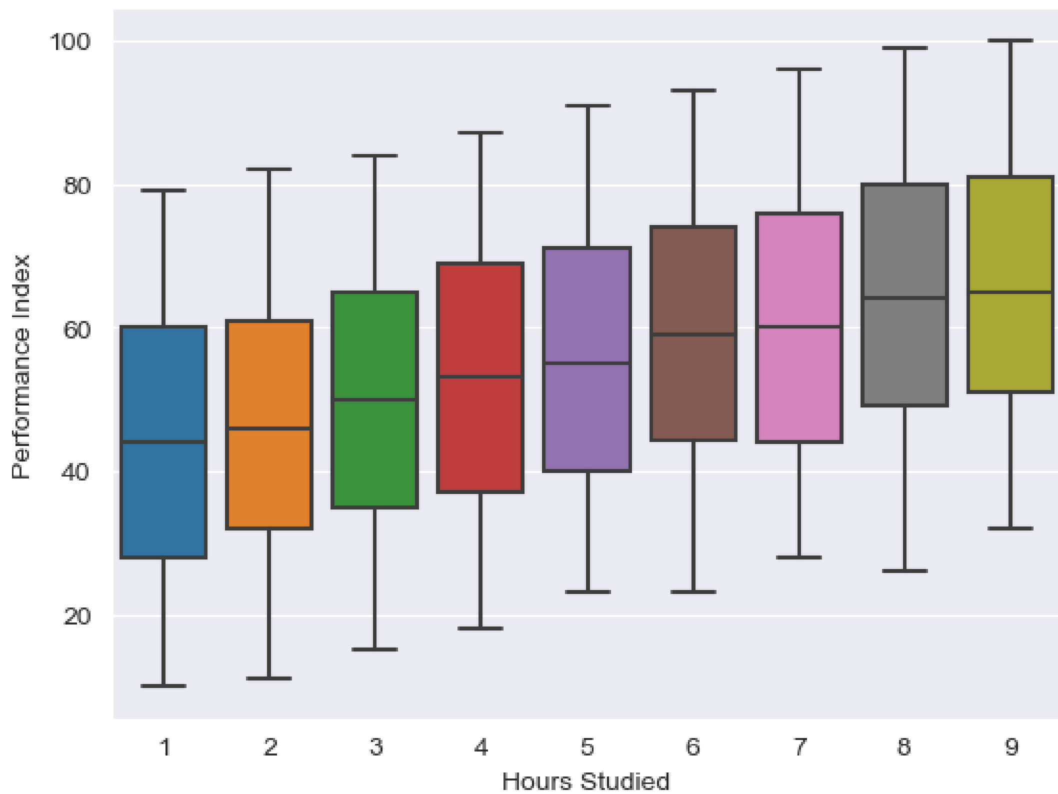
[85]:
```
sns.barplot(x = df1['Hours Studied'],y = df1['Performance Index'])
```

[85]: <Axes: xlabel='Hours Studied', ylabel='Performance Index'>



[86]:
```
sns.boxplot(x = df1['Hours Studied'],y = df1['Performance Index'])
```

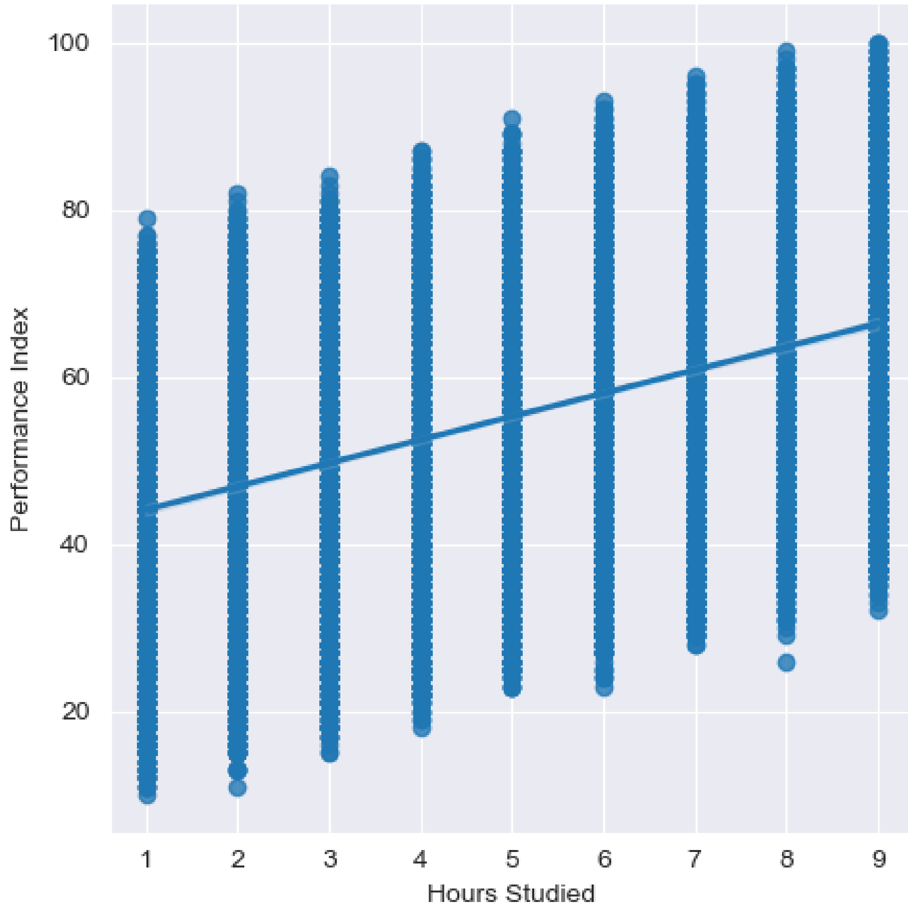
[86]: <Axes: xlabel='Hours Studied', ylabel='Performance Index'>



[87]:
```
sns.lmplot(x = 'Hours Studied',y = 'Performance Index',data= df1)
```

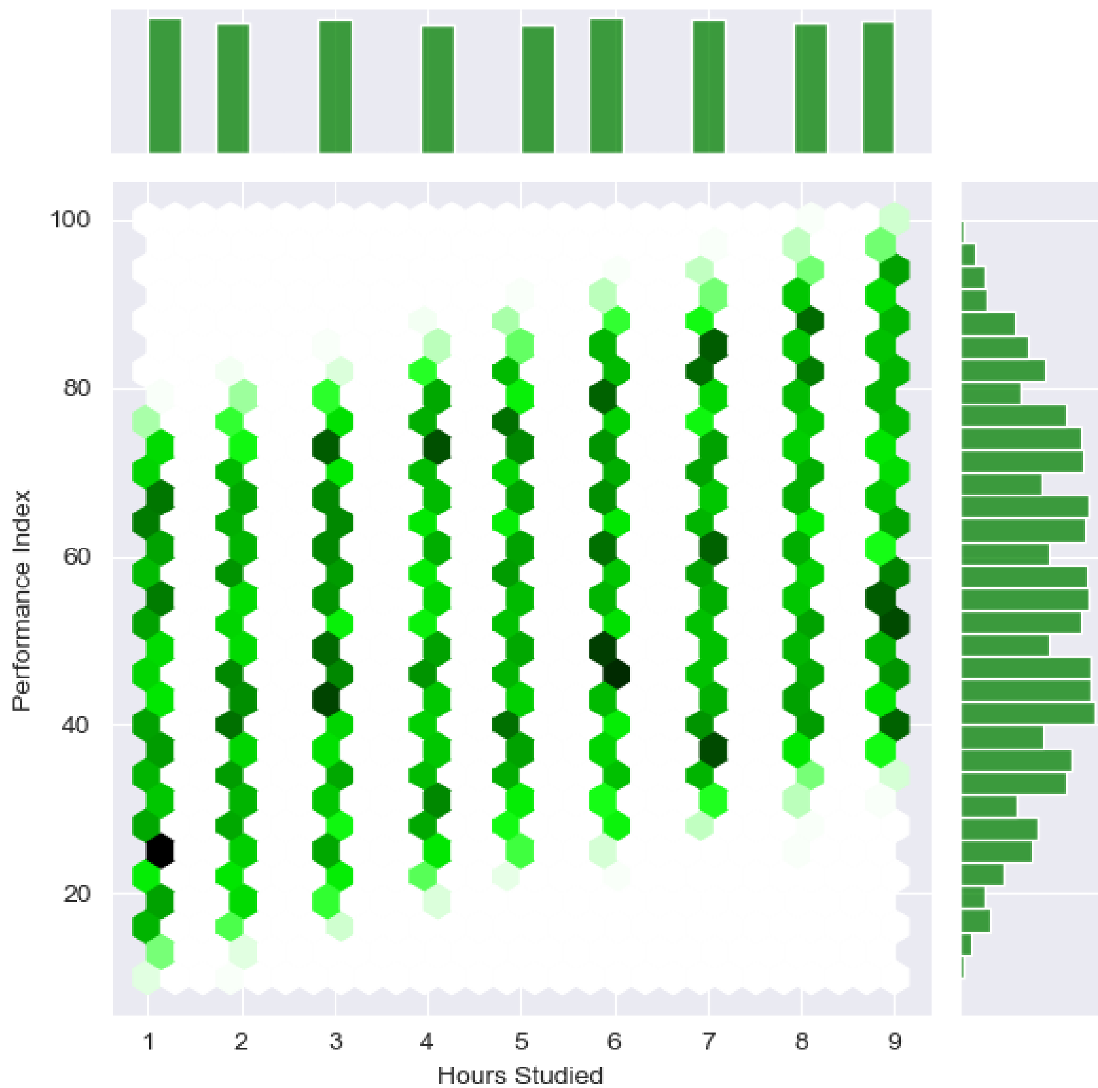
```
D:\anaconda\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure
layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

[87]:
```
<seaborn.axisgrid.FacetGrid at 0x1c620cc5050>
```



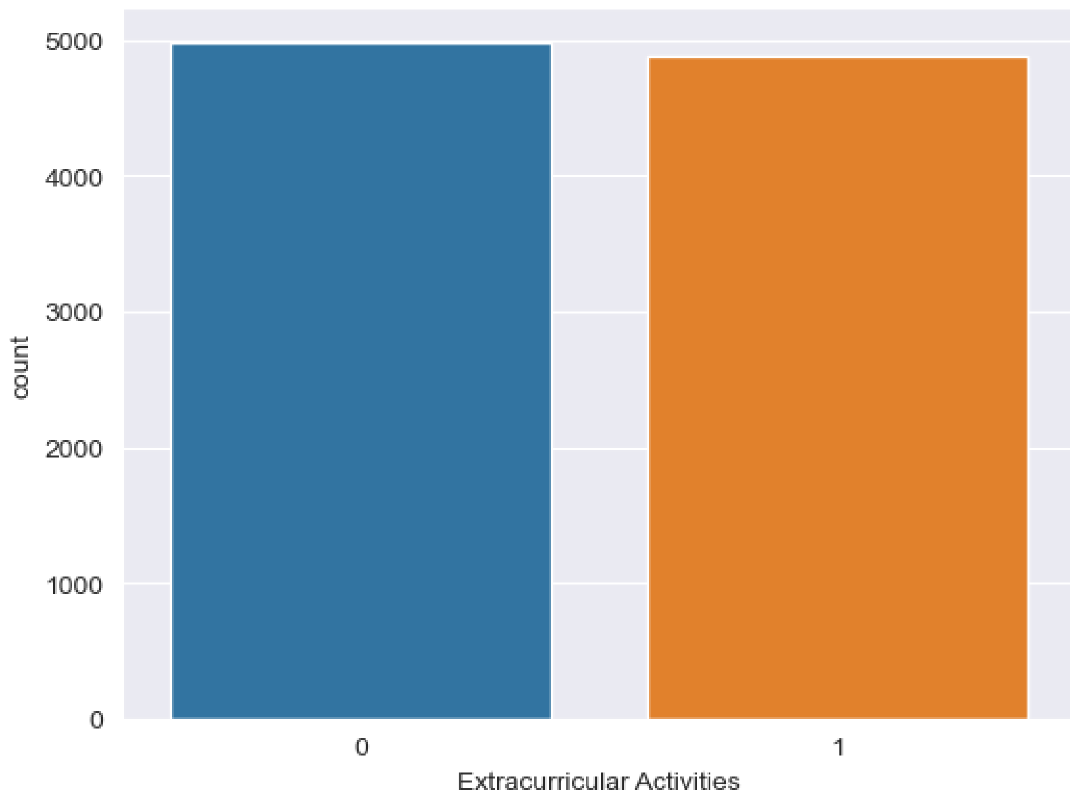
[88]:
```
sns.jointplot(x='Hours Studied',y='Performance Index',data = df1,kind =␣
↪'hex',color='green')
```

[88]: <seaborn.axisgrid.JointGrid at 0x1c620ee7110>



[125]:
```
sns.countplot(x='Extracurricular Activities',data = df1)
```

[125]: <Axes: xlabel='Extracurricular Activities', ylabel='count'>



[126]:
```
df1['Extracurricular Activities'].replace({'Yes':1,'No':0},inplace=True)
df1
```

[126]:
```
      Hours Studied  Previous Scores  Extracurricular Activities  Sleep Hours  \
0                 7               99                           1            9
1                 4               82                           0            4
2                 8               51                           1            7
3                 5               52                           1            5
4                 7               75                           0            8
…               …                …                           …          …
9995              1               49                           1            4
9996              7               64                           1            8
9997              6               83                           1            8
9998              9               97                           1            7
9999              7               74                           0            8

      Sample Question Papers Practiced  Performance Index
0                                    1               91.0
1                                    2               65.0
2                                    2               45.0
3                                    2               36.0
```

```
4                                      5               66.0
…                                  …               …
9995                                   2               23.0
9996                                   5               58.0
9997                                   5               74.0
9998                                   0               95.0
9999                                   1               64.0

[9873 rows x 6 columns]
```

[113]:
```
x = df1[['Hours Studied', 'Previous Scores', 'Extracurricular Activities',
       'Sleep Hours', 'Sample Question Papers Practiced']]
y = df1[['Performance Index']]
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.
 ↪4,random_state=101)
model = LinearRegression()
model
```

[113]:
```
LinearRegression()
```

[127]:
```
model = LinearRegression()
model.fit(x_train,y_train)
y_pred = model.predict(x_test)
y_pred
```

[127]:
```
array([[31.77074959],
       [64.03350895],
       [59.60018183],
       …,
       [38.92149791],
       [71.4083267 ],
       [40.63996372]])
```

[128]:
```
accuracy = accuracy_score(y_test,np.round(y_pred))
accuracy
```

[128]:
```
0.20278481012658228
```

[129]:
```
inputdata = [[5,87,0,7,5]]
prediction = model.predict(inputdata)
prediction
```

```
D:\anaconda\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have
valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

[129]:
```
array([[73.13673869]])
```

[139]:
```
from sklearn.metrics import mean_squared_error
mse=mean_squared_error(y_test,y_pred)
mse
```

[139]:
```
4.20547881932315
```

[145]:
```
# Instead of linear regression use ridge
from sklearn.linear_model import Ridge
```

[151]:
```
clf = Ridge()
clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)
y_pred
```

[151]:
```
array([[31.77111194],
       [64.03318608],
       [59.60058437],
       …,
       [38.92110706],
       [71.40812022],
       [40.63972288]])
```

[155]:
```
df = pd.read_csv("framingham.csv")
df
```

[155]:
```
      male  age  education  currentSmoker  cigsPerDay  BPMeds  \
0        1   39        4.0              0         0.0     0.0
1        0   46        2.0              0         0.0     0.0
2        1   48        1.0              1        20.0     0.0
3        0   61        3.0              1        30.0     0.0
4        0   46        3.0              1        23.0     0.0
…      …   …          …              …         …       …
4233     1   50        1.0              1         1.0     0.0
4234     1   51        3.0              1        43.0     0.0
4235     0   48        2.0              1        20.0     NaN
4236     0   44        1.0              1        15.0     0.0
4237     0   52        2.0              0         0.0     0.0

      prevalentStroke  prevalentHyp  diabetes  totChol  sysBP  diaBP    BMI  \
0                   0             0         0    195.0  106.0   70.0  26.97
1                   0             0         0    250.0  121.0   81.0  28.73
2                   0             0         0    245.0  127.5   80.0  25.34
3                   0             1         0    225.0  150.0   95.0  28.58
4                   0             0         0    285.0  130.0   84.0  23.10
…                 …             …         …      …      …      …      …
```

```
4233               0           1       0    313.0  179.0   92.0  25.97
4234               0           0       0    207.0  126.5   80.0  19.71
4235               0           0       0    248.0  131.0   72.0  22.00
4236               0           0       0    210.0  126.5   87.0  19.16
4237               0           0       0    269.0  133.5   83.0  21.47

      heartRate  glucose  TenYearCHD
0          80.0     77.0           0
1          95.0     76.0           0
2          75.0     70.0           0
3          65.0    103.0           1
4          85.0     85.0           0
…           …        …          …
4233       66.0     86.0           1
4234       65.0     68.0           0
4235       84.0     86.0           0
4236       86.0      NaN           0
4237       80.0    107.0           0

[4238 rows x 16 columns]
```

[161]:
```
m = df.isnull().sum()
m
```

[161]:
```
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
dtype: int64
```

[168]:
```
df = df.fillna(m)
df
```

[168]:
```
      male  age  education  currentSmoker  cigsPerDay  BPMeds  \
0        1   39        4.0              0         0.0     0.0
```

```
1          0   46         2.0              0          0.0      0.0
2          1   48         1.0              1         20.0      0.0
3          0   61         3.0              1         30.0      0.0
4          0   46         3.0              1         23.0      0.0
…     …   …         …            …          …       …
4233       1   50         1.0              1          1.0      0.0
4234       1   51         3.0              1         43.0      0.0
4235       0   48         2.0              1         20.0     53.0
4236       0   44         1.0              1         15.0      0.0
4237       0   52         2.0              0          0.0      0.0

      prevalentStroke  prevalentHyp  diabetes  totChol  sysBP  diaBP    BMI  \
0                   0             0         0    195.0  106.0   70.0  26.97
1                   0             0         0    250.0  121.0   81.0  28.73
2                   0             0         0    245.0  127.5   80.0  25.34
3                   0             1         0    225.0  150.0   95.0  28.58
4                   0             0         0    285.0  130.0   84.0  23.10
…                …            …         …       …     …     …     …
4233                0             1         0    313.0  179.0   92.0  25.97
4234                0             0         0    207.0  126.5   80.0  19.71
4235                0             0         0    248.0  131.0   72.0  22.00
4236                0             0         0    210.0  126.5   87.0  19.16
4237                0             0         0    269.0  133.5   83.0  21.47

      heartRate  glucose  TenYearCHD
0          80.0     77.0           0
1          95.0     76.0           0
2          75.0     70.0           0
3          65.0    103.0           1
4          85.0     85.0           0
…          …       …          …
4233       66.0     86.0           1
4234       65.0     68.0           0
4235       84.0     86.0           0
4236       86.0    388.0           0
4237       80.0    107.0           0

[4238 rows x 16 columns]
```

[171]:
```
df.isnull().sum()
```

[171]:
```
male               0
age                0
education          0
currentSmoker      0
cigsPerDay         0
BPMeds             0
```

```
prevalentStroke    0
prevalentHyp       0
diabetes           0
totChol            0
sysBP              0
diaBP              0
BMI                0
heartRate          0
glucose            0
TenYearCHD         0
dtype: int64
```

[176]:
```
df.columns
```

[176]:
```
Index(['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose', 'TenYearCHD'],
      dtype='object')
```

[181]:
```
from sklearn.model_selection import train_test_split
x = df[['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose']]
y = df[['TenYearCHD']]
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.
 ↪4,random_state=101)
model = LinearRegression()
model
```

[181]:
```
LinearRegression()
```

[183]:
```
model = LinearRegression()
model.fit(x_train,y_train)
y_pred = model.predict(x_test)
y_pred
```

[183]:
```
array([[ 0.175186  ],
       [ 0.39124632],
       [ 0.26406313],
       …,
       [ 0.27954268],
       [-0.01360633],
       [ 0.03153421]])
```

[185]:
```
accuracy = accuracy_score(y_test,np.round(y_pred))
accuracy
```

[185]: 0.8608490566037735

[186]:
```
inputdata = [[0,45,2.0,1,20,25,0,1,1,230,120,80,15,85,100]]
prediction = model.predict(inputdata)
prediction
```

```
D:\anaconda\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have
valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

[186]: array([[0.35811095]])