In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
tou= 0.5 #tou is the bandwidth parameter and controls the rate at which w^{(i)} fall
```

In [3]:
```
#TRAINSET DATA
X_train = np.array(list(range(3, 33)) )
print(X_train)
X_train=X_train[:,np.newaxis]
print(X_train)
y_train = np.array([1,2,1,2,1,1,3,4,5,4,5,6,5,6,7,8,9,10,11,11,12,11,11,10,12,11,11,10,
print(y_train)
#the newaxis is used to increase the dimension of the existing array by one more dimens
#1D array will become 2D array
#2D array will become 3D array
```

```
[ 3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
 27 28 29 30 31 32]
[[ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]
 [24]
 [25]
 [26]
 [27]
 [28]
 [29]
 [30]
 [31]
 [32]]
[ 1  2  1  2  1  1  3  4  5  4  5  6  5  6  7  8  9 10 11 11 12 11 11 10
 12 11 11 10  9  8]
```

In [4]:
```
X_train.shape
```

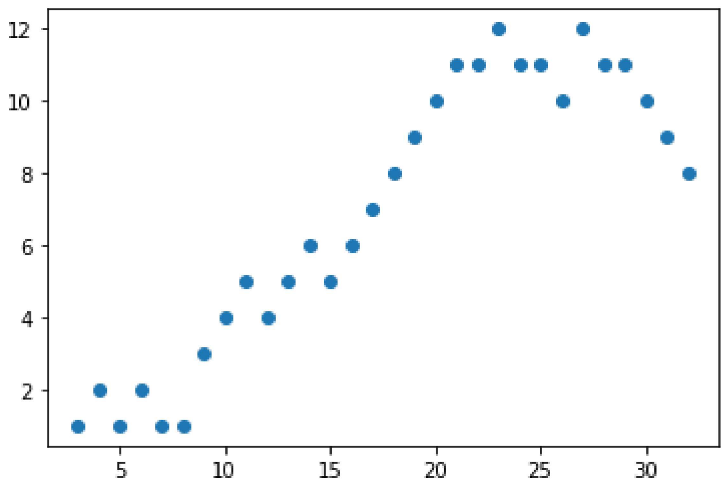
Out[4]: (30, 1)

In [5]:
```
y_train.shape
```

Out[5]: (30,)

In [12]:
```
plt.scatter(X_train,y_train)
```

Out[12]: <matplotlib.collections.PathCollection at 0x1fed90809a0>



In [7]:
```
X_test = np.array([i/10. for i in range(400)])
print(X_test)
X_test=X_test[:,np.newaxis]
print(X_test)
y_test=[]
print(y_test)
```

```
[ 0.   0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.   1.1  1.2  1.3
  1.4  1.5  1.6  1.7  1.8  1.9  2.   2.1  2.2  2.3  2.4  2.5  2.6  2.7
  2.8  2.9  3.   3.1  3.2  3.3  3.4  3.5  3.6  3.7  3.8  3.9  4.   4.1
  4.2  4.3  4.4  4.5  4.6  4.7  4.8  4.9  5.   5.1  5.2  5.3  5.4  5.5
  5.6  5.7  5.8  5.9  6.   6.1  6.2  6.3  6.4  6.5  6.6  6.7  6.8  6.9
  7.   7.1  7.2  7.3  7.4  7.5  7.6  7.7  7.8  7.9  8.   8.1  8.2  8.3
  8.4  8.5  8.6  8.7  8.8  8.9  9.   9.1  9.2  9.3  9.4  9.5  9.6  9.7
  9.8  9.9 10.  10.1 10.2 10.3 10.4 10.5 10.6 10.7 10.8 10.9 11.  11.1
 11.2 11.3 11.4 11.5 11.6 11.7 11.8 11.9 12.  12.1 12.2 12.3 12.4 12.5
 12.6 12.7 12.8 12.9 13.  13.1 13.2 13.3 13.4 13.5 13.6 13.7 13.8 13.9
 14.  14.1 14.2 14.3 14.4 14.5 14.6 14.7 14.8 14.9 15.  15.1 15.2 15.3
 15.4 15.5 15.6 15.7 15.8 15.9 16.  16.1 16.2 16.3 16.4 16.5 16.6 16.7
 16.8 16.9 17.  17.1 17.2 17.3 17.4 17.5 17.6 17.7 17.8 17.9 18.  18.1
 18.2 18.3 18.4 18.5 18.6 18.7 18.8 18.9 19.  19.1 19.2 19.3 19.4 19.5
 19.6 19.7 19.8 19.9 20.  20.1 20.2 20.3 20.4 20.5 20.6 20.7 20.8 20.9
 21.  21.1 21.2 21.3 21.4 21.5 21.6 21.7 21.8 21.9 22.  22.1 22.2 22.3
 22.4 22.5 22.6 22.7 22.8 22.9 23.  23.1 23.2 23.3 23.4 23.5 23.6 23.7
 23.8 23.9 24.  24.1 24.2 24.3 24.4 24.5 24.6 24.7 24.8 24.9 25.  25.1
 25.2 25.3 25.4 25.5 25.6 25.7 25.8 25.9 26.  26.1 26.2 26.3 26.4 26.5
 26.6 26.7 26.8 26.9 27.  27.1 27.2 27.3 27.4 27.5 27.6 27.7 27.8 27.9
 28.  28.1 28.2 28.3 28.4 28.5 28.6 28.7 28.8 28.9 29.  29.1 29.2 29.3
 29.4 29.5 29.6 29.7 29.8 29.9 30.  30.1 30.2 30.3 30.4 30.5 30.6 30.7
 30.8 30.9 31.  31.1 31.2 31.3 31.4 31.5 31.6 31.7 31.8 31.9 32.  32.1
```

```
 32.2 32.3 32.4 32.5 32.6 32.7 32.8 32.9 33.  33.1 33.2 33.3 33.4 33.5
 33.6 33.7 33.8 33.9 34.  34.1 34.2 34.3 34.4 34.5 34.6 34.7 34.8 34.9
 35.  35.1 35.2 35.3 35.4 35.5 35.6 35.7 35.8 35.9 36.  36.1 36.2 36.3
 36.4 36.5 36.6 36.7 36.8 36.9 37.  37.1 37.2 37.3 37.4 37.5 37.6 37.7
 37.8 37.9 38.  38.1 38.2 38.3 38.4 38.5 38.6 38.7 38.8 38.9 39.  39.1
 39.2 39.3 39.4 39.5 39.6 39.7 39.8 39.9]
[[ 0. ]
 [ 0.1]
 [ 0.2]
 [ 0.3]
 [ 0.4]
 [ 0.5]
 [ 0.6]
 [ 0.7]
 [ 0.8]
 [ 0.9]
 [ 1. ]
 [ 1.1]
 [ 1.2]
 [ 1.3]
 [ 1.4]
 [ 1.5]
 [ 1.6]
 [ 1.7]
 [ 1.8]
 [ 1.9]
 [ 2. ]
 [ 2.1]
 [ 2.2]
 [ 2.3]
 [ 2.4]
 [ 2.5]
 [ 2.6]
 [ 2.7]
 [ 2.8]
 [ 2.9]
 [ 3. ]
 [ 3.1]
 [ 3.2]
 [ 3.3]
 [ 3.4]
 [ 3.5]
 [ 3.6]
 [ 3.7]
 [ 3.8]
 [ 3.9]
 [ 4. ]
 [ 4.1]
 [ 4.2]
 [ 4.3]
 [ 4.4]
 [ 4.5]
 [ 4.6]
 [ 4.7]
 [ 4.8]
 [ 4.9]
 [ 5. ]
 [ 5.1]
 [ 5.2]
 [ 5.3]
```

```
[ 5.4]
[ 5.5]
[ 5.6]
[ 5.7]
[ 5.8]
[ 5.9]
[ 6. ]
[ 6.1]
[ 6.2]
[ 6.3]
[ 6.4]
[ 6.5]
[ 6.6]
[ 6.7]
[ 6.8]
[ 6.9]
[ 7. ]
[ 7.1]
[ 7.2]
[ 7.3]
[ 7.4]
[ 7.5]
[ 7.6]
[ 7.7]
[ 7.8]
[ 7.9]
[ 8. ]
[ 8.1]
[ 8.2]
[ 8.3]
[ 8.4]
[ 8.5]
[ 8.6]
[ 8.7]
[ 8.8]
[ 8.9]
[ 9. ]
[ 9.1]
[ 9.2]
[ 9.3]
[ 9.4]
[ 9.5]
[ 9.6]
[ 9.7]
[ 9.8]
[ 9.9]
[10. ]
[10.1]
[10.2]
[10.3]
[10.4]
[10.5]
[10.6]
[10.7]
[10.8]
[10.9]
[11. ]
[11.1]
[11.2]
[11.3]
```

```
[11.4]
[11.5]
[11.6]
[11.7]
[11.8]
[11.9]
[12. ]
[12.1]
[12.2]
[12.3]
[12.4]
[12.5]
[12.6]
[12.7]
[12.8]
[12.9]
[13. ]
[13.1]
[13.2]
[13.3]
[13.4]
[13.5]
[13.6]
[13.7]
[13.8]
[13.9]
[14. ]
[14.1]
[14.2]
[14.3]
[14.4]
[14.5]
[14.6]
[14.7]
[14.8]
[14.9]
[15. ]
[15.1]
[15.2]
[15.3]
[15.4]
[15.5]
[15.6]
[15.7]
[15.8]
[15.9]
[16. ]
[16.1]
[16.2]
[16.3]
[16.4]
[16.5]
[16.6]
[16.7]
[16.8]
[16.9]
[17. ]
[17.1]
[17.2]
[17.3]
```

```
 [17.4]
 [17.5]
 [17.6]
 [17.7]
 [17.8]
 [17.9]
 [18. ]
 [18.1]
 [18.2]
 [18.3]
 [18.4]
 [18.5]
 [18.6]
 [18.7]
 [18.8]
 [18.9]
 [19. ]
 [19.1]
 [19.2]
 [19.3]
 [19.4]
 [19.5]
 [19.6]
 [19.7]
 [19.8]
 [19.9]
 [20. ]
 [20.1]
 [20.2]
 [20.3]
 [20.4]
 [20.5]
 [20.6]
 [20.7]
 [20.8]
 [20.9]
 [21. ]
 [21.1]
 [21.2]
 [21.3]
 [21.4]
 [21.5]
 [21.6]
 [21.7]
 [21.8]
 [21.9]
 [22. ]
 [22.1]
 [22.2]
 [22.3]
 [22.4]
 [22.5]
 [22.6]
 [22.7]
 [22.8]
 [22.9]
 [23. ]
 [23.1]
 [23.2]
 [23.3]
```

```
 [23.4]
 [23.5]
 [23.6]
 [23.7]
 [23.8]
 [23.9]
 [24. ]
 [24.1]
 [24.2]
 [24.3]
 [24.4]
 [24.5]
 [24.6]
 [24.7]
 [24.8]
 [24.9]
 [25. ]
 [25.1]
 [25.2]
 [25.3]
 [25.4]
 [25.5]
 [25.6]
 [25.7]
 [25.8]
 [25.9]
 [26. ]
 [26.1]
 [26.2]
 [26.3]
 [26.4]
 [26.5]
 [26.6]
 [26.7]
 [26.8]
 [26.9]
 [27. ]
 [27.1]
 [27.2]
 [27.3]
 [27.4]
 [27.5]
 [27.6]
 [27.7]
 [27.8]
 [27.9]
 [28. ]
 [28.1]
 [28.2]
 [28.3]
 [28.4]
 [28.5]
 [28.6]
 [28.7]
 [28.8]
 [28.9]
 [29. ]
 [29.1]
 [29.2]
 [29.3]
```

```
 [29.4]
 [29.5]
 [29.6]
 [29.7]
 [29.8]
 [29.9]
 [30. ]
 [30.1]
 [30.2]
 [30.3]
 [30.4]
 [30.5]
 [30.6]
 [30.7]
 [30.8]
 [30.9]
 [31. ]
 [31.1]
 [31.2]
 [31.3]
 [31.4]
 [31.5]
 [31.6]
 [31.7]
 [31.8]
 [31.9]
 [32. ]
 [32.1]
 [32.2]
 [32.3]
 [32.4]
 [32.5]
 [32.6]
 [32.7]
 [32.8]
 [32.9]
 [33. ]
 [33.1]
 [33.2]
 [33.3]
 [33.4]
 [33.5]
 [33.6]
 [33.7]
 [33.8]
 [33.9]
 [34. ]
 [34.1]
 [34.2]
 [34.3]
 [34.4]
 [34.5]
 [34.6]
 [34.7]
 [34.8]
 [34.9]
 [35. ]
 [35.1]
 [35.2]
 [35.3]
```

```
 [35.4]
 [35.5]
 [35.6]
 [35.7]
 [35.8]
 [35.9]
 [36. ]
 [36.1]
 [36.2]
 [36.3]
 [36.4]
 [36.5]
 [36.6]
 [36.7]
 [36.8]
 [36.9]
 [37. ]
 [37.1]
 [37.2]
 [37.3]
 [37.4]
 [37.5]
 [37.6]
 [37.7]
 [37.8]
 [37.9]
 [38. ]
 [38.1]
 [38.2]
 [38.3]
 [38.4]
 [38.5]
 [38.6]
 [38.7]
 [38.8]
 [38.9]
 [39. ]
 [39.1]
 [39.2]
 [39.3]
 [39.4]
 [39.5]
 [39.6]
 [39.7]
 [39.8]
 [39.9]]
[]
```

In [8]:

```
X_test.shape
```

Out[8]:

```
(400, 1)
```

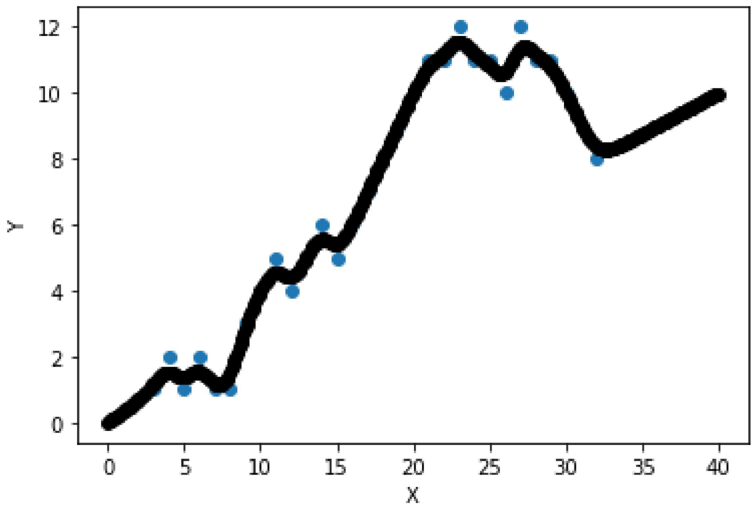
In [9]:

```
for r in range(len(X_test)):
    wts=np.exp(-np.sum((X_train-X_test[r])**2,axis=1)/(2*tou)**2)
    W=np.diag(wts)
    #constant value
    factor1 = np.linalg.inv(X_train.T.dot(W).dot(X_train)) # find inverse of (X.T*W*X)
    parameters=factor1.dot(X_train.T).dot(W).dot(y_train) # final values of beta
```

```
    prediction=X_test[r].dot(parameters)
    y_test.append(prediction)
```

In [13]:

```
y_test = np.array(y_test)
plt.scatter(X_train, y_train)
plt.scatter(X_test, y_test, color='black')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



In [ ]:

In [ ]: