Logistics Regression

What is our GOAL for this MODULE?

The goal of this module is to learn about logistics regression and write a prediction algorithm.

What did we ACHIEVE in the class TODAY?

We learned about the logistics regression and wrote our own prediction algorithm.

Which CONCEPTS/CODING BLOCKS did we cover today?

- Sigmoid function
- Prediction algorithm

How did we DO the activities?

1. We uploaded the height and weight data and plotted the height and weight data on the scatter plot.

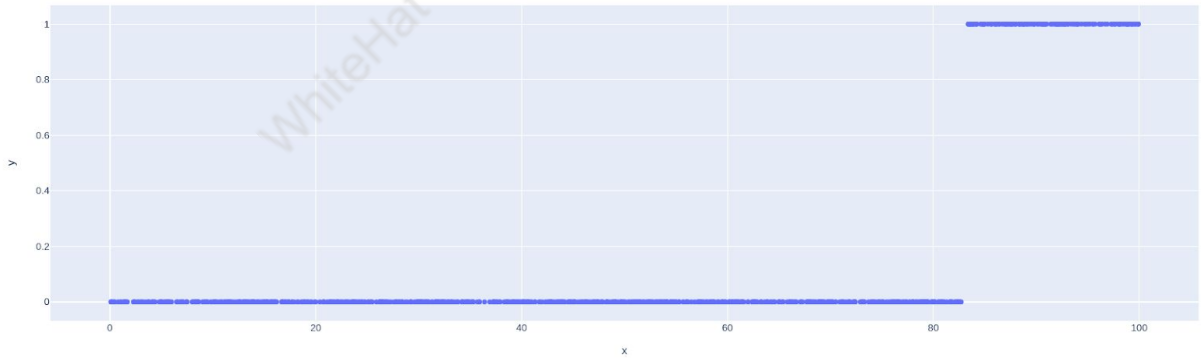
```
[3] #Uploading the csv
    from google.colab import files
    data_to_load = files.upload()
```

Choose Files data.csv

- **data.csv**(text/csv) - 7936 bytes, last modified: 04/08/2020 - 100% done

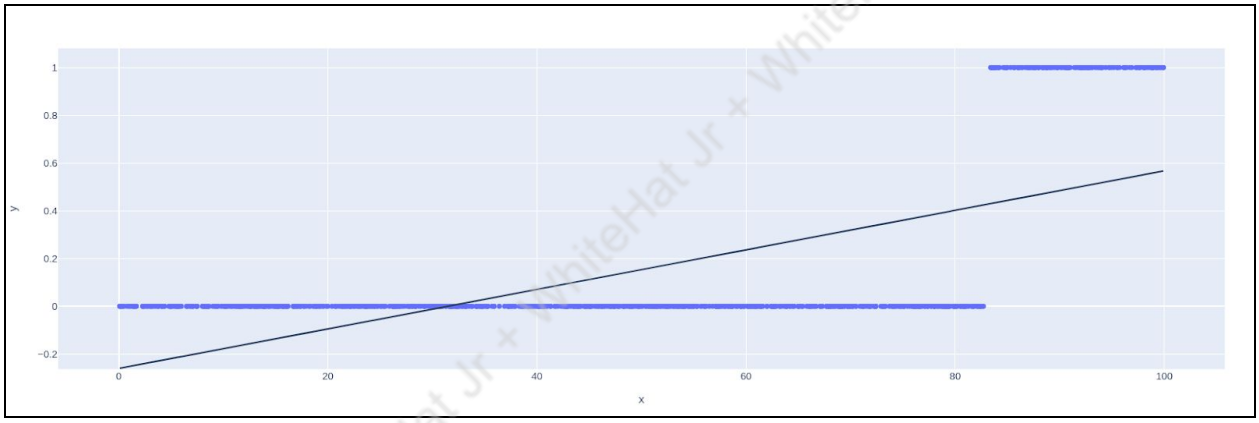
```
Saving data.csv to data.csv
```

```
] import pandas as pd
  import plotly.express as px

  df = pd.read_csv("data.csv")

  score_list = df["Score"].tolist()
  accepted_list = df["Accepted"].tolist()

  fig = px.scatter(x=score_list, y=accepted_list)
  fig.show()
```



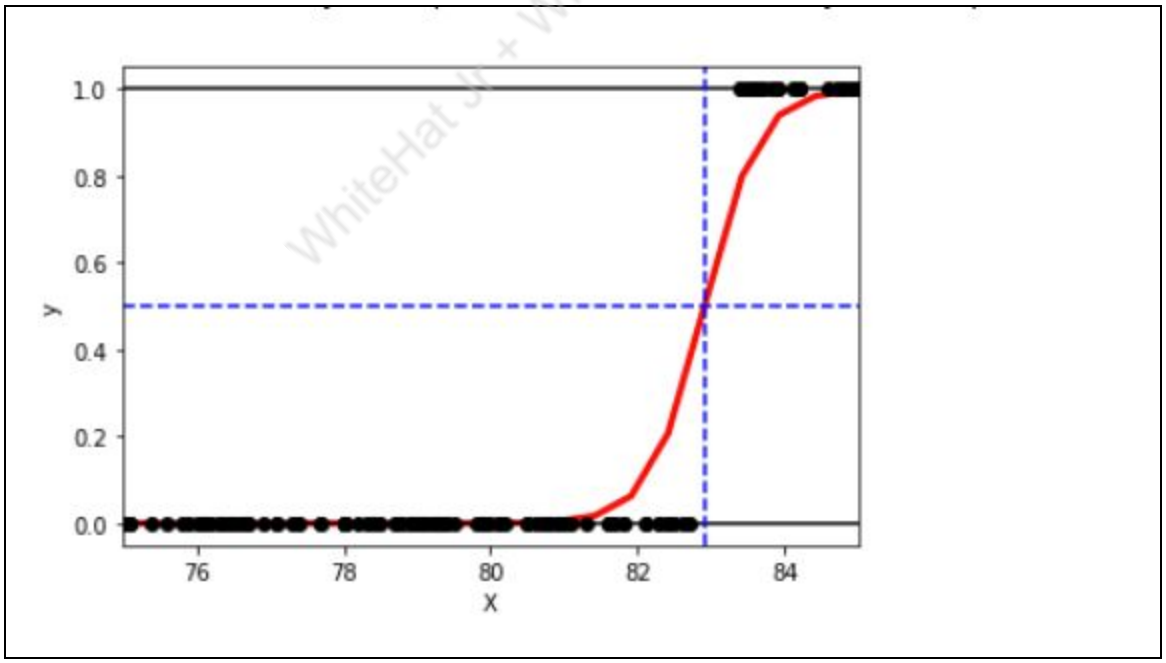
2. We saw the line equation and tried to plot a line on the points in our graph.

```
import numpy as np
score_array = np.array(score_list)
accepted_array = np.array(accepted_list)

#Slope and intercept using pre-built function of Numpy
m, c = np.polyfit(score_array, accepted_array, 1)

y = []
for x in score_array:
  y_value = m*x + c
  y.append(y_value)

#plotting the graph
fig = px.scatter(x=score_array, y=accepted_array)
fig.update_layout(shapes=[
    dict(
      type= 'line',
      y0= min(y), y1= max(y),
      x0= min(score_array), x1= max(score_array)
    )
])
fig.show()
```



3. Using the hit and trial method in the sigmoid function we found the value which intersected the line of regression.

```
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression

X = np.reshape(score_list, (len(score_list), 1))
Y = np.reshape(accepted_list, (len(accepted_list), 1))

lr = LogisticRegression()
lr.fit(X, Y)

plt.figure()
plt.scatter(X.ravel(), Y, color='black', zorder=20)

def model(x):
  return 1 / (1 + np.exp(-x))

#Using the line formula
X_test = np.linspace(0, 100, 200)
chances = model(X_test * lr.coef_ + lr.intercept_).ravel()

plt.plot(X_test, chances, color='red', linewidth=3)
plt.axhline(y=0, color='k', linestyle='-')
plt.axhline(y=1, color='k', linestyle='-')
plt.axhline(y=0.5, color='b', linestyle='--')

# do hit and trial by changing the value of X_test
plt.axvline(x=X_test[165], color='b', linestyle='--')

plt.ylabel('y')
plt.xlabel('X')
plt.xlim(75, 85)
plt.show()
```



4. Then using the line equation we wrote the prediction algorithm.

```
user_score = float(input("Enter your marks here:- "))
chances = model(user_score * lr.coef_ + lr.intercept_).ravel()[0]
if chances <= 0.01:
  print("The student will not get accepted")
elif chances >= 1:
  print("The student will get accepted!")
elif chances < 0.5:
  print("The student might not get accepted")
else:
  print("The student may get accepted")
```

```
Enter your marks here:- 99
The student will get accepted!
```

We tried the same method with another data of tungsten melting point.

1. Uploaded the data.

```
#Uploading the csv
from google.colab import files
data_to_load = files.upload()
```

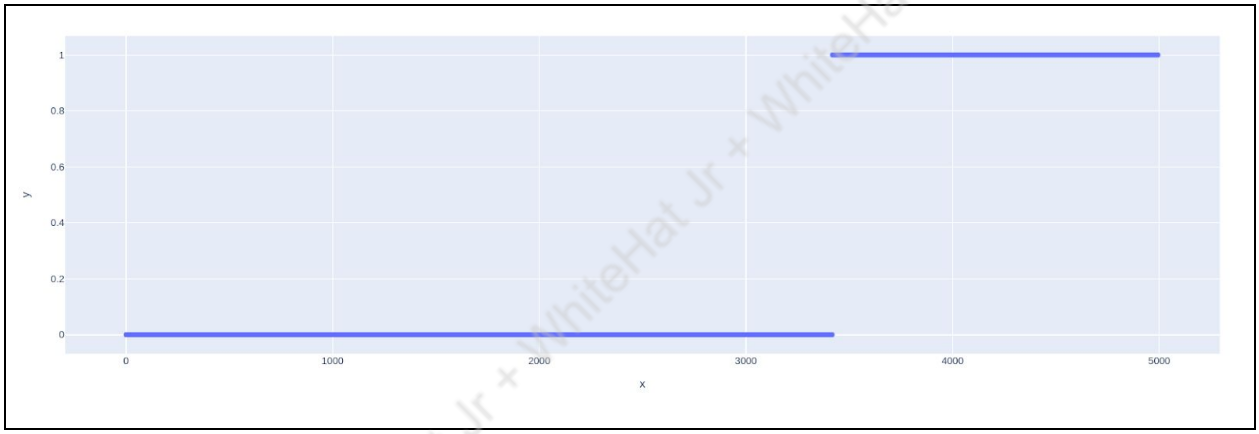
Choose Files data2.csv

- **data2.csv**(text/csv) - 97824 bytes, last modified: 04/08/2020 - 100% done

```
Saving data2.csv to data2.csv
```

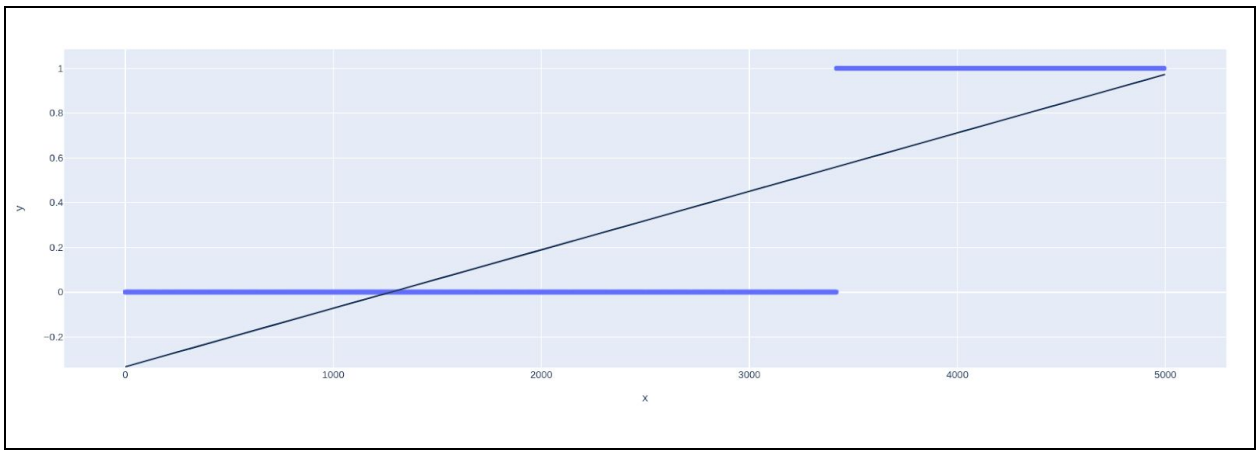
2. Plotted the data on scatter plot.

```
import pandas as pd
import plotly.express as px

df = pd.read_csv("data2.csv")

temperature_list = df["Temperature"].tolist()
melted_list = df["Melted"].tolist()

fig = px.scatter(x=temperature_list, y=melted_list)
fig.show()
```



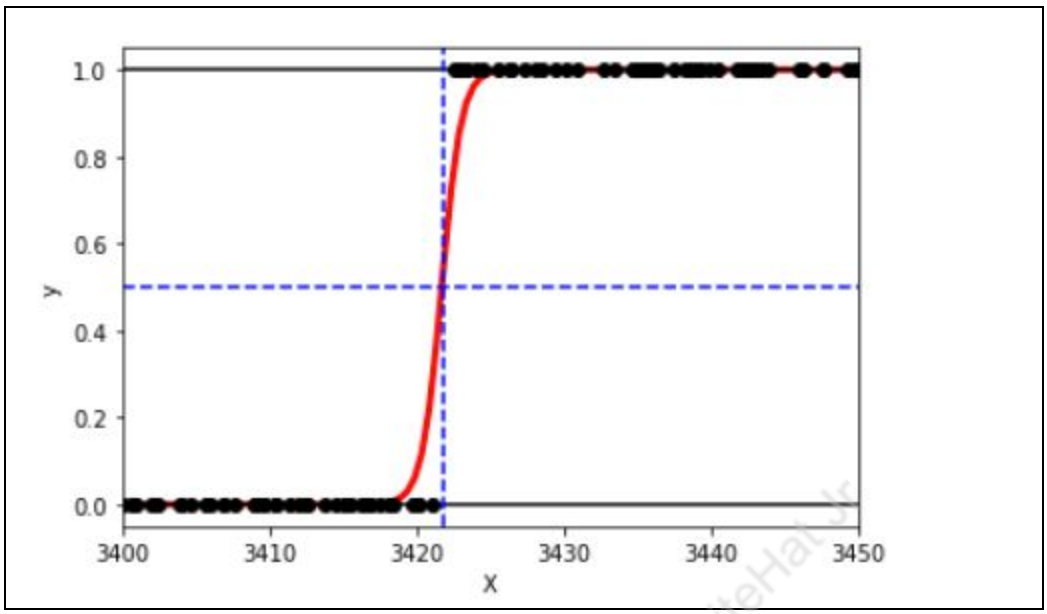
3. We plotted the regression line on the graph.

```
import numpy as np
temperature_array = np.array(temperature_list)
melted_array = np.array(melted_list)

#Slope and intercept using pre-built function of Numpy
m, c = np.polyfit(temperature_array, melted_array, 1)

y = []
for x in temperature_array:
  y_value = m*x + c
  y.append(y_value)

#plotting the graph
fig = px.scatter(x=temperature_array, y=melted_array)
fig.update_layout(shapes=[
    dict(
      type= 'line',
      y0= min(y), y1= max(y),
      x0= min(temperature_array), x1= max(temperature_array)
    )
])
fig.show()
```



4. Using the sigmoid function and hit and trial method we found the value which intersects the line of regression and plotted it.

```
]  import matplotlib.pyplot as plt
   from sklearn.linear_model import LogisticRegression

   X = np.reshape(temperature_list, (len(temperature_list), 1))
   Y = np.reshape(melted_list, (len(melted_list), 1))

   lr = LogisticRegression()
   lr.fit(X, Y)

   plt.figure()
   plt.scatter(X.ravel(), Y, color='black', zorder=20)

   def model(x):
     return 1 / (1 + np.exp(-x))

   #Using the line formula
   X_test = np.linspace(0, 5000, 10000)
   melting_chances = model(X_test * lr.coef_ + lr.intercept_).ravel()

   plt.plot(X_test, melting_chances, color='red', linewidth=3)
   plt.axhline(y=0, color='k', linestyle='-')
   plt.axhline(y=1, color='k', linestyle='-')
   plt.axhline(y=0.5, color='b', linestyle='--')

   #do hit and trial by changing the vlaue of X_test here.
   plt.axvline(x=X_test[6843], color='b', linestyle='--')

   plt.ylabel('y')
   plt.xlabel('X')
   plt.xlim(3400, 3450)
   plt.show()
```



5. Using the line equation we wrote the prediction algorithm.

```
temp = float(input("Enter the temperature here:- "))
chances = model(temp * lr.coef_ + lr.intercept_).ravel()[0]
if chances <= 0.01:
  print("Tungsten will not be melted")
elif chances >= 1:
  print("Tungsten will be melted")
elif chances < 0.5:
  print("Tungsten might not get melted")
else:
  print("Tungsten might get melted")
```

```
Enter the temperature here:- 60
Tungsten will not be melted
```

We concluded that using the previous data and it's analysis we can make the prediction and arrive at conclusions.

What's NEXT?

In the next class, we will learn about multilinear regression.

EXTEND YOUR KNOWLEDGE:

Read the following blog to understand more about the logistic regression:
https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148