Logistic Regression

October 29, 2020

1 Logistic Regression Notebook

This notebook uses the datasets in this link https://goo.gl/zjS4C6. Please refer to chaper 6.

Imagine a situation where we have a dataset from a supermarket store about the gender of the customer and whether that person bought a particular product or not. We are interested in finding the chances of a customer buying that particular product, given their gender. What comes to mind when someone poses this question to you? Probability anyone? Odds of success?

What is the probability of a customer buying a product, given he is a male? What is the probability of a customer buying that product, given she is a female? If we know the answers to these questions, we can make a leap towards predicting the chances of a customer buying a product, given their gender.

1.0.1 Import libraries and load dataset

[1]:
```
import pandas as pd
import numpy as np

url = ('https://raw.githubusercontent.com/JasonMDev/'
       'learning-python-predictive-analytics/master/'
       'datasets/Gender%20Purchase.csv')
```

[3]:
```
# Read dataset
df = pd.read_csv(url)
#df = pd.read_csv('Supporting Material/Gender Purchase.csv')
df.head()
```

[3]:
```
   Gender Purchase
0  Female      Yes
1  Female      Yes
2  Female       No
3    Male       No
4    Male      Yes
```

[4]:
```
# Show information related to dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 511 entries, 0 to 510
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Gender    511 non-null    object
 1   Purchase  511 non-null    object
dtypes: object(2)
memory usage: 8.1+ KB
```

[5]:
```
df.describe()
```

[5]:
```
       Gender Purchase
count     511      511
unique      2        2
top    Female      Yes
freq      265      280
```

[6]:
```
# Show the amount yes or nos by gender
contingency_table=pd.crosstab(df['Gender'],df['Purchase'])
contingency_table
```

[6]:
```
Purchase   No  Yes
Gender            
Female    106  159
Male      125  121
```

1.0.2 Defining Conditional Probability

$P(A|B) = \frac{P(A \cap B)}{P(B)}$

Let: * M -> The person is male * F -> The person is female * B -> The person bought a product * D -> The peron did not buy a product

Then the conditional probabilities are:

- Probability of buying males:

$$P(B|M) = \frac{P(B \cap M)}{P(M)} = \frac{121}{121 + 125} = 0.49$$

- Probability of not buying males:

$$P(B|M) = \frac{P(D \cap M)}{P(M)} = \frac{125}{121 + 125} = 0.51$$

- Probability of buying females:

$$P(B|M) = \frac{P(B \cap F)}{P(F)} = \frac{159}{159 + 106} = 0.60$$

- Probability of not buying females:

$$P(B|M) = \frac{P(D \cap F)}{P(F)} = \frac{106}{159+106} = 0.40$$

```
[7]: # Show frequencies
     probabilities = contingency_table.apply(lambda x: x/sum(x), axis=1)
     probabilities
```

```
[7]: Purchase        No      Yes
     Gender
     Female     0.40000  0.60000
     Male       0.50813  0.49187
```

1.0.3 Odds

Define a success rate for a desired event

- Odds of purchase by males =

$$\frac{P(B|M)}{P(D|M)} = \frac{P(B|M)}{1-P(B|M)} = \frac{0.49}{0.51} = 0.96$$

- Odds of purchase by females =

$$\frac{P(B|F)}{P(D|F)} = \frac{P(B|F)}{1-P(B|F)} == \frac{0.60}{0.40} = 1.5$$

```
[8]: # Show frequencies
     odds_purchase_male = probabilities.loc['Male']['Yes'] / probabilities.
      ↪loc['Male']['No']
     odds_purchase_female = probabilities.loc['Female']['Yes'] / probabilities.
      ↪loc['Female']['No']
```

```
[9]: print('Odds Male = {} Odds Female = {}'.format(odds_purchase_male,␣
      ↪odds_purchase_female))
```

```
Odds Male = 0.968 Odds Female = 1.4999999999999998
```

1.0.4 Odds between groups

One better way to determine which group has better odds of success is by calculating odds ratios for each group. The odds ratio is defined as follows:

$$OddsRatio = \frac{OddsGroup1}{OddsGroup2}$$

Then:

$$OddsRatio(males) = \frac{OddsMales}{OddsFemales} = \frac{0.96}{1.5} = 0.64$$

$$OddsRatio(females) = \frac{OddsFeales}{OddsMales} = \frac{1.5}{0.96} = 1.54$$

```
[10]: odds_ratio_males = odds_purchase_male / odds_purchase_female
      odds_ratio_females = odds_purchase_female / odds_purchase_male
```

```
[11]: print('Odds Ratio Male = {} Odds ratio Female = {}'.format(odds_ratio_males,␣
       ↪odds_ratio_females))
```

```
Odds Ratio Male = 0.6453333333333334 Odds ratio Female = 1.549586776859504
```

1.0.5 Considerations About Odds

There are a couple of important things to note about the odds ratio: * The more the odds ratio, the more the chances of success from that group. In our case, the female group has an odds ratio of 1.54, which means that it is more probable to get success (purchase) from a female customer than a male customer. * If the odds ratio=1, then there is no association between the two variables. If odds ratio>1, then the event is more likely to happen in Group 1. If the odds ratio<1, then the event is more likely to happen in Group 2. * Also, the odds ratio for one group can be obtained by taking the reciprocal of the odds ratio of the other group.

2 Linear Regression analogy

Remember linear regression equation:

$$Y = \beta_0 + \beta_1 * X + \epsilon$$

- X can assume any value in range $-\infty, +\infty$. Therefore, it is hard to properly match these values in a $ [0, 1]$ range
- What if we try to predict the probabilities associated with the two events rather than the binary outcomes? Predicting the probabilities will be feasible as their range spans from 0 to 1.

$$P(Y) = a + b * X$$

- The range problem persists, P $[0,1]$ while X $[-\infty, +\infty]$

What if we use the odds instead of P, the range would be $[0, \infty]$

$$P/(1-P) = a + b * X$$

What if we use the log of odds?

$$log(P/(1-P)) = a + b * X$$

[12]:
```python
# Lets test the range of log function
import math

print(math.log(10**-20))
print(math.log(10**20))
```

```
-46.051701859880914
46.051701859880914
```

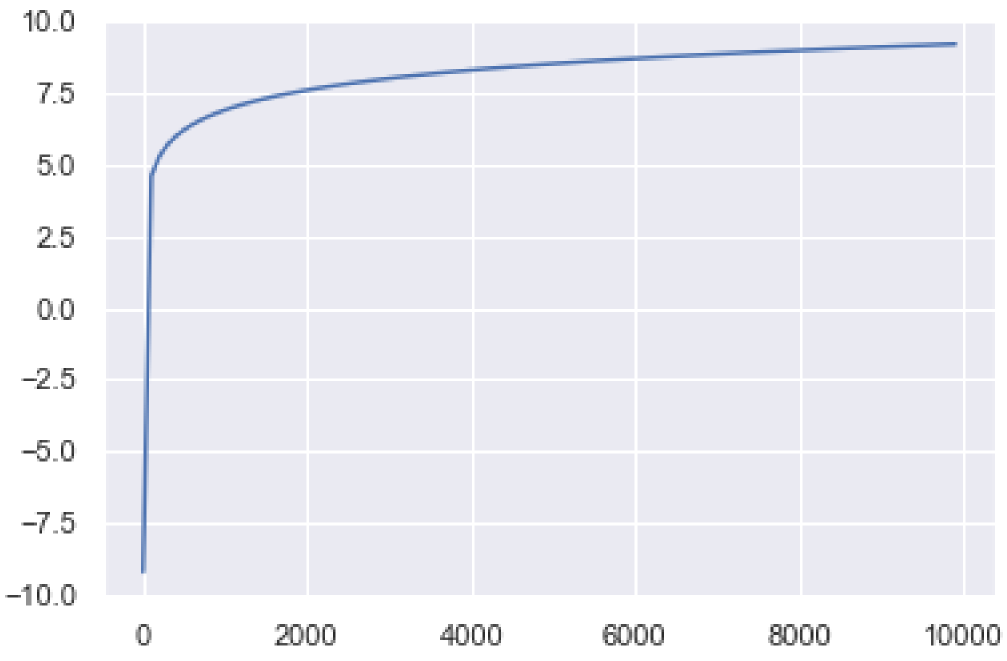
[13]:
```python
# Observe logarithm can assume any value in -infinite to infinite
from matplotlib import pyplot as plt
import seaborn as sns

sns.set()


X = np.arange(10**-4, 10**4, 100)
Y = np.log(X)


plt.plot(X, Y)
```

[13]:
```
[<matplotlib.lines.Line2D at 0x131c90898>]
```



2.0.1 Final equation to logistic regression

$$log(P/(1-P)) = a + b*X$$

$$\frac{P}{1-P} = e^{a+b*X}$$

$$P = (1-P) * e^{a+b*X}$$

$$P = e^{a+b*X} - P * e^{a+b*X}$$

$$P + P * e^{a+b*X} = e^{a+b*X}$$

$$P(1 + e^{a+b*X}) = e^{a+b*X}$$

$$P = \frac{e^{a+b*X}}{1 + e^{a+b*X}}$$

$$P = \frac{1}{1 + e^{-(a+b*X)}}$$

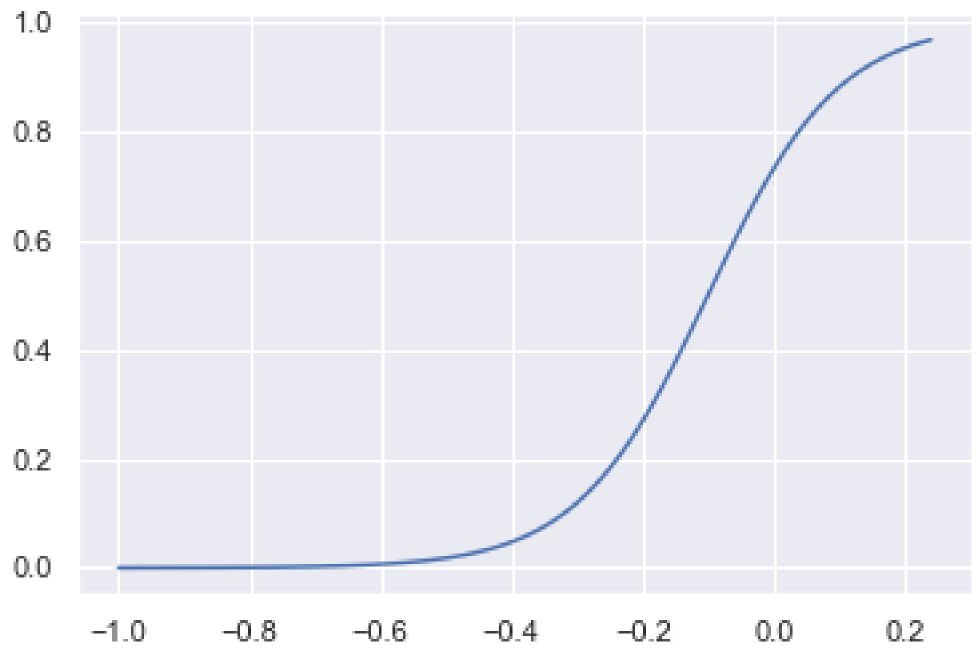
[14]:
```
# Lets see it in a graph

a, b = 1,10
logistics = lambda x: 1 / (1 + np.e**-(a + b*x))
vlogistcs = np.vectorize(logistics)

X = np.arange(-1, 0.25, 0.01)
Y = vlogistcs(X)

plt.plot(X, Y)
```

[14]:
```
[<matplotlib.lines.Line2D at 0x131e1cfd0>]
```



[ ]:

[15]:
```
np.log(0)
```

```
/Users/brunosilva/Dropbox/Mackenzie/Aulas/venv_mackenzie/lib/python3.7/site-
packages/ipykernel_launcher.py:1: RuntimeWarning: divide by zero encountered in
log
  """Entry point for launching an IPython kernel.
```

[15]:
```
-inf
```

[ ]: