In [1]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

In [3]:
```
df=pd.read_csv("C3_bot_detection_data.csv")
df
```

Out[3]:

	User ID	Username	Tweet	Retweet Count	Mention Count	Follower Count	Verified	Bot Label	Location	Creat
0	132131	flong	Station activity person against natural majori...	85	1	2353	False	1	Adkinston	202 05-15:29:
1	289683	hinesstephanie	Authority research natural life material staff...	55	5	9617	True	0	Sanderston	202 11-05:18:
2	779715	roberttran	Manage whose quickly especially foot none to g...	6	2	4363	True	0	Harrisonfurt	202 08-03:16:
3	696168	pmason	Just cover eight opportunity strong policy which.	54	5	2242	True	1	Martinezberg	202 08-22:27:
4	704441	noah87	Animal sign six data good or.	26	3	8438	False	1	Camachoville	202 04-21:24:
...	...	...	...	...	...	...	...	...	...	
49995	491196	uberg	Want but put card direction know miss former h...	64	0	9911	True	1	Lake Kimberlyburgh	202 04-11:06:
49996	739297	jessicamunoz	Provide whole maybe agree church respond most ...	18	5	9900	False	1	Greenbury	202 10-03:57:
49997	674475	lynncunningham	Bring different everyone international capital...	43	3	6313	True	1	Deborahfort	202 07-03:54:
49998	167081	richardthompson	Than about single generation itself seek sell ...	45	1	6343	False	0	Stephenside	202 03-12:13:
49999	311204	daniel29	Here morning class various room human true bec...	91	4	4006	False	0	Novakberg	202 12-06:11:

50000 rows × 11 columns

In [5]:
```
df1=df.iloc[:,3:8]
df1
```

Out[5]:

	Retweet Count	Mention Count	Follower Count	Verified	Bot Label
0	85	1	2353	False	1
1	55	5	9617	True	0
2	6	2	4363	True	0
3	54	5	2242	True	1
4	26	3	8438	False	1
...	...	...	...	...	...
49995	64	0	9911	True	1
49996	18	5	9900	False	1
49997	43	3	6313	True	1
49998	45	1	6343	False	0
49999	91	4	4006	False	0

50000 rows × 5 columns

In [6]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Retweet Count   50000 non-null  int64
 1   Mention Count   50000 non-null  int64
 2   Follower Count  50000 non-null  int64
 3   Verified        50000 non-null  bool
 4   Bot Label       50000 non-null  int64
dtypes: bool(1), int64(4)
memory usage: 1.6 MB
```

In [8]:
```
y=df1["Verified"]
x=df1.drop(["Verified"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [9]:
```
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

Out[9]:
```
LogisticRegression()
```

In [13]:
```
lr.predict(x_test)
```

Out[13]:
```
array([ True,  True,  True, ..., False,  True,  True])
```

In [14]:
```
lr.score(x_test,y_test)
```

Out[14]: 0.494

In [15]:
```
df2=pd.read_csv("C4_framingham.csv")
df2
```

Out[15]:

	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBl
	39	4.0	0	0.0	0.0	0	0	0	195.0	106.
	46	2.0	0	0.0	0.0	0	0	0	250.0	121.
	48	1.0	1	20.0	0.0	0	0	0	245.0	127.
	61	3.0	1	30.0	0.0	0	1	0	225.0	150.
	46	3.0	1	23.0	0.0	0	0	0	285.0	130.
	...	...	...	...	...	...	...	...	...	.
	50	1.0	1	1.0	0.0	0	1	0	313.0	179.
	51	3.0	1	43.0	0.0	0	0	0	207.0	126.
	48	2.0	1	20.0	NaN	0	0	0	248.0	131.
	44	1.0	1	15.0	0.0	0	0	0	210.0	126.
	52	2.0	0	0.0	0.0	0	0	0	269.0	133.

× 16 columns

In [16]:
```
df2=df2.dropna()
```

In [17]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3656 entries, 0 to 4237
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             3656 non-null   int64  
 1   age              3656 non-null   int64  
 2   education        3656 non-null   float64
 3   currentSmoker    3656 non-null   int64  
 4   cigsPerDay       3656 non-null   float64
 5   BPMeds           3656 non-null   float64
 6   prevalentStroke  3656 non-null   int64  
 7   prevalentHyp     3656 non-null   int64  
 8   diabetes         3656 non-null   int64  
 9   totChol          3656 non-null   float64
 10  sysBP            3656 non-null   float64
 11  diaBP            3656 non-null   float64
 12  BMI              3656 non-null   float64
 13  heartRate        3656 non-null   float64
 14  glucose          3656 non-null   float64
 15  TenYearCHD       3656 non-null   int64  
dtypes: float64(9), int64(7)
memory usage: 485.6 KB
```

In [26]:
```
y=df2["diabetes"]
x=df2.drop(["diabetes"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [27]:
```
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: Conve
rgenceWarning: 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 (https://scikit-learn.or
g/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 (http
s://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[27]:
```
LogisticRegression()
```

In [29]:
```
val=[[1,34,5,1,4,1,0,1,123,108,89,29,84,70,1]]
lr.predict(val)
```

Out[29]:
```
array([0], dtype=int64)
```

In [31]:
```
lr.score(x_test,y_test)
```

Out[31]:
```
0.9817684594348223
```

In [30]:
```
df3=pd.read_csv("C5_health care diabetes.csv")
df3
```

Out[30]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Out
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	
...	...	...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.171	63	
764	2	122	70	27	0	36.8	0.340	27	
765	5	121	72	23	112	26.2	0.245	30	
766	1	126	60	0	0	30.1	0.349	47	
767	1	93	70	31	0	30.4	0.315	23	

768 rows × 9 columns

In [32]:
```
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [34]:
```
y=df3["Outcome"]
x=df3.drop(["Outcome"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [35]:
```
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: Conve
rgenceWarning: 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 (https://scikit-learn.or
g/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 (http
s://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[35]:
```
LogisticRegression()
```

In [36]:
```
lr.predict(x_test)
```

Out[36]:
```
array([0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0,
       1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0], dtype=int64)
```

In [37]:
```
val1=[[1,34,5,1,4,1,123,10]]
lr.predict(val1)
```

Out[37]:
```
array([1], dtype=int64)
```

In [ ]:
```

```