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

In [2]:
```
df=pd.read_csv(r"C:\Users\venka\OneDrive\Documents\ionosphere_data karthik.cs
df
```

Out[2]:

| | column_a | column_b | column_c | column_d | column_e | column_f | column_g | column_h | co |
|---|---|---|---|---|---|---|---|---|---|
| **0** | True | False | 0.99539 | -0.05889 | 0.85243 | 0.02306 | 0.83398 | -0.37708 | 1 |
| **1** | True | False | 1.00000 | -0.18829 | 0.93035 | -0.36156 | -0.10868 | -0.93597 | 1 |
| **2** | True | False | 1.00000 | -0.03365 | 1.00000 | 0.00485 | 1.00000 | -0.12062 | ( |
| **3** | True | False | 1.00000 | -0.45161 | 1.00000 | 1.00000 | 0.71216 | -1.00000 | ( |
| **4** | True | False | 1.00000 | -0.02401 | 0.94140 | 0.06531 | 0.92106 | -0.23255 | ( |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **346** | True | False | 0.83508 | 0.08298 | 0.73739 | -0.14706 | 0.84349 | -0.05567 | ( |
| **347** | True | False | 0.95113 | 0.00419 | 0.95183 | -0.02723 | 0.93438 | -0.01920 | ( |
| **348** | True | False | 0.94701 | -0.00034 | 0.93207 | -0.03227 | 0.95177 | -0.03431 | ( |
| **349** | True | False | 0.90608 | -0.01657 | 0.98122 | -0.01989 | 0.95691 | -0.03646 | ( |
| **350** | True | False | 0.84710 | 0.13533 | 0.73638 | -0.06151 | 0.87873 | 0.08260 | ( |

351 rows × 35 columns

In [3]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
print('This dataframe has %d rows and %d columns'%(df.shape))
```

```
This dataframe has 351 rows and 35 columns
```

In [4]:
```
df.head(10)
```

Out[4]:

| | column_a | column_b | column_c | column_d | column_e | column_f | column_g | column_h | colu |
|---|---|---|---|---|---|---|---|---|---|
| **0** | True | False | 0.99539 | -0.05889 | 0.85243 | 0.02306 | 0.83398 | -0.37708 | 1.0 |
| **1** | True | False | 1.00000 | -0.18829 | 0.93035 | -0.36156 | -0.10868 | -0.93597 | 1.0 |
| **2** | True | False | 1.00000 | -0.03365 | 1.00000 | 0.00485 | 1.00000 | -0.12062 | 0.8 |
| **3** | True | False | 1.00000 | -0.45161 | 1.00000 | 1.00000 | 0.71216 | -1.00000 | 0.0 |
| **4** | True | False | 1.00000 | -0.02401 | 0.94140 | 0.06531 | 0.92106 | -0.23255 | 0.7 |
| **5** | True | False | 0.02337 | -0.00592 | -0.09924 | -0.11949 | -0.00763 | -0.11824 | 0.1 |
| **6** | True | False | 0.97588 | -0.10602 | 0.94601 | -0.20800 | 0.92806 | -0.28350 | 0.8 |
| **7** | False | False | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 1.00000 | -1.00000 | 0.0 |
| **8** | True | False | 0.96355 | -0.07198 | 1.00000 | -0.14333 | 1.00000 | -0.21313 | 1.0 |
| **9** | True | False | -0.01864 | -0.08459 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.1 |

In [5]:
```
features_matrix=df.iloc[:,0:34]
```

In [6]:
```
target_vector=df.iloc[:,-1]
```

In [7]:
```
print('The features matrix has %d rows and %d columns'%(features_matrix.shape
```

```
The features matrix has 351 rows and 34 columns
```

In [8]:
```
print('The target matrix has %d rows and %d columns'%(np.array(target_vector)
```

```
The target matrix has 351 rows and 1 columns
```

In [9]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [10]:
```
algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_inter
```

In [21]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_v
```

In [22]:
```
observation=[[1,0,0.99539,-0.05889,0.8524299999999999,0.02306,0.8339799999999
```

In [23]:
```
from sklearn.linear_model import LogisticRegression
predictions=Logistic_Regression_Model.predict(observation)
```

In [24]:
```
print('The model predicted the observation to belong to class %s'%(prediction
```

```
The model predicted the observation to belong to class ['g']
```

In [25]:
```
print('The algorithm was trained to predict one of the two classes :%s'%(algo
```

```
The algorithm was trained to predict one of the two classes :['b' 'g']
```

In [26]:
```
print("""The model says the probability of the observation we passed belonging
```

```
The model says the probability of the observation we passed belonging to cla
ss['b'] is 0.00777393160013784
```

In [27]:
```
print()
```

In [28]:
```
print("""The model says the probability of the observation we passed belonging
```

```
The model says the probability of the observation we passed belonging to cla
ss['g'] is 0.9922260683998622
```

In [29]:
```
#2 DIGITS
import re
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
%matplotlib inline
digits=load_digits()
```

In [21]:

```
print("Image Data Shape",digits.data.shape)
print("Label Data Shape",digits.target.shape)
```

```
Image Data Shape (1797, 64)
Label Data Shape (1797,)
```

In [22]:

```
plt.figure(figsize=(20,4))
```

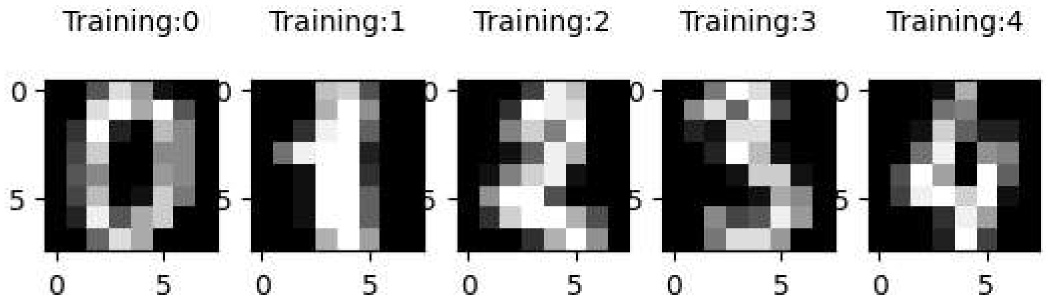
Out[22]:

```
<Figure size 2000x400 with 0 Axes>

<Figure size 2000x400 with 0 Axes>
```

In [23]:

```
for index,(image,label)in enumerate(zip(digits.data[0:5],digits.target[0:5]))
 plt.subplot(1,5,index+1)
 plt.imshow(np.reshape(image,(8,8)),cmap=plt.cm.gray)
 plt.title('Training:%i\n'%label,fontsize=10)
```



In [24]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_
```

In [25]:

```
print(x_train.shape)
```

```
(1257, 64)
```

In [26]:

```
print(y_train.shape)
```

```
(1257,)
```

In [27]:

```
print(x_test.shape)
```

```
(540, 64)
```

In [28]:
```
print(y_test.shape)
```

```
(540,)
```

In [29]:
```
from sklearn.linear_model import LogisticRegression
logisticRegr=LogisticRegression(max_iter=10000)
```

In [30]:
```
logisticRegr.fit(x_train,y_train)
```

Out[30]:
```
▾ LogisticRegression
LogisticRegression(max_iter=10000)
```

In [31]:
```
print(logisticRegr.predict(x_test))
```

```
[4 0 9 1 8 7 1 5 1 6 6 7 6 1 5 5 8 6 2 7 4 6 4 1 5 2 9 5 4 6 5 6 3 4 0 9 9
 8 4 6 8 8 5 7 9 8 9 6 1 7 0 1 9 7 3 3 1 8 8 8 9 8 5 8 4 9 3 5 8 4 3 1 3 8
 7 3 3 0 8 7 2 8 5 3 8 7 6 4 6 2 2 0 1 1 5 3 5 7 1 8 2 2 6 4 6 7 3 7 3 9 4
 7 0 3 5 1 5 0 3 9 2 7 3 2 0 8 1 9 2 1 5 1 0 3 4 3 0 8 3 2 2 7 3 1 6 7 2 8
 3 1 1 6 4 8 2 1 8 4 1 3 1 1 9 5 4 8 7 4 8 9 5 7 6 9 4 0 4 0 0 9 0 6 5 8 8
 3 7 9 2 0 8 2 7 3 0 2 1 9 2 7 0 6 9 3 1 1 3 5 2 5 5 2 1 2 9 4 6 5 5 5 9 7
 1 5 9 6 3 7 1 7 5 1 7 2 7 5 5 4 8 6 6 2 8 7 3 7 8 0 9 5 7 4 3 4 1 0 3 3 5
 4 1 3 1 2 5 1 4 0 3 1 5 5 7 4 0 1 0 9 5 5 5 4 0 1 8 6 2 1 1 1 7 9 6 7 9 7
 0 4 9 6 9 2 7 2 1 0 8 2 8 6 5 7 8 4 5 7 8 6 4 2 6 9 3 0 0 8 0 6 6 7 1 4 5
 6 9 7 2 8 5 1 2 4 1 8 8 7 6 0 8 0 6 1 5 7 8 0 4 1 4 5 9 2 2 3 9 1 3 9 3 2
 8 0 6 5 6 2 5 2 3 2 6 1 0 7 6 0 6 2 7 0 3 2 4 2 3 6 9 7 7 0 3 5 4 1 2 2 1
 2 7 7 0 4 9 8 5 6 1 6 5 2 0 8 2 4 3 3 2 9 3 8 9 9 5 9 0 3 4 7 9 8 5 7 5 0
 5 3 5 0 2 7 3 0 4 3 6 6 1 9 6 3 4 6 4 6 7 2 7 6 3 0 3 0 1 3 6 1 0 4 3 8 4
 3 3 4 8 6 9 6 3 3 0 5 7 8 9 1 5 3 2 5 1 7 6 0 6 9 5 2 4 4 7 2 0 5 6 2 0 8
 4 4 4 7 1 0 4 1 9 2 1 3 0 5 3 9 8 2 6 0 0 4]
```

In [32]:
```
score=logisticRegr.score(x_test,y_test)
print(score)
```

```
0.9537037037037037
```

In [ ]:

In [1]:

```
#3 GENDER SUBMISSION
import re
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import metrics
%matplotlib inline
```

In [6]:

```
df=pd.read_csv(r"C:\Users\venka\OneDrive\Documents\gender_submission.csv")
print(df)
```

```
     PassengerId  Survived
0            892         0
1            893         1
2            894         0
3            895         0
4            896         1
..           ...       ...
413         1305         0
414         1306         1
415         1307         0
416         1308         0
417         1309         0

[418 rows x 2 columns]
```

In [7]:

```
plt.figure(figsize=(20,2))
```

Out[7]:

```
<Figure size 2000x200 with 0 Axes>

<Figure size 2000x200 with 0 Axes>
```

In [8]:

```
df.describe()
```

Out[8]:

| | PassengerId | Survived |
|---|---|---|
| **count** | 418.000000 | 418.000000 |
| **mean** | 1100.500000 | 0.363636 |
| **std** | 120.810458 | 0.481622 |
| **min** | 892.000000 | 0.000000 |
| **25%** | 996.250000 | 0.000000 |
| **50%** | 1100.500000 | 0.000000 |
| **75%** | 1204.750000 | 1.000000 |
| **max** | 1309.000000 | 1.000000 |

In [9]:
```
df.isnull().any()
```

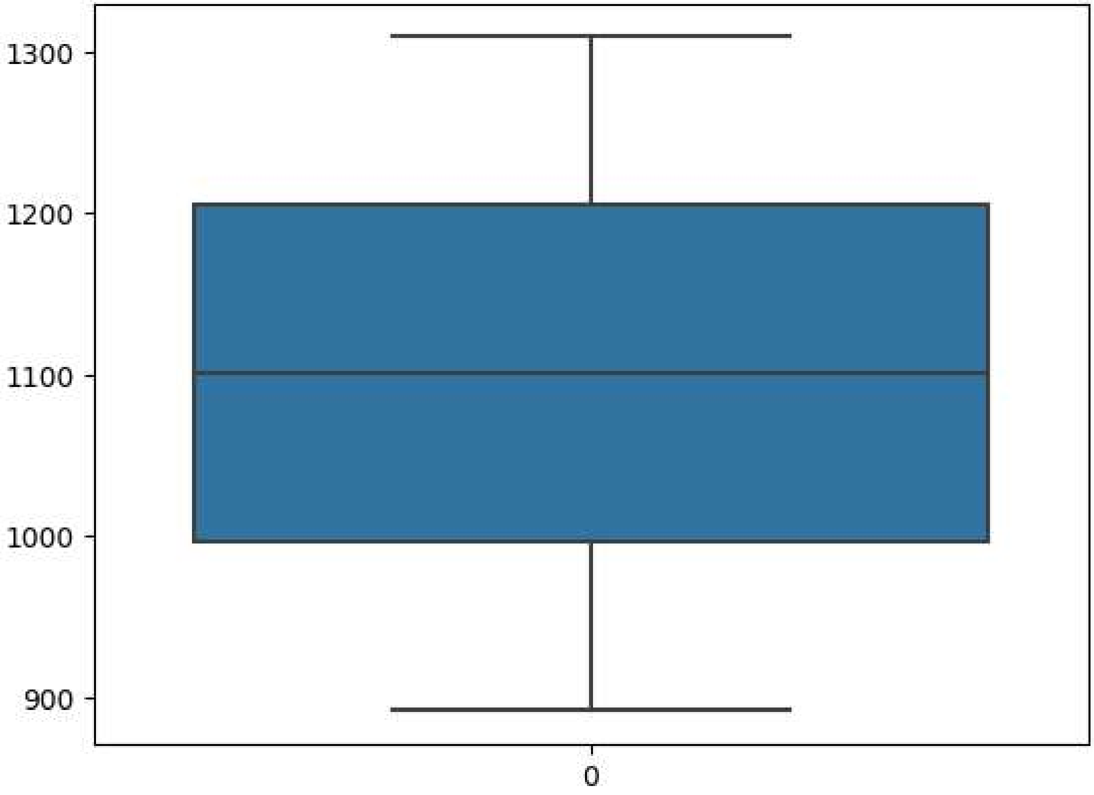
Out[9]:
```
PassengerId    False
Survived       False
dtype: bool
```

In [10]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
print('This dataframe has %d rows and %d columns'%(df.shape))
```

```
This dataframe has 418 rows and 2 columns
```

In [11]:
```
sns.boxplot(df['PassengerId'])
```

Out[11]:
```
<Axes: >
```



In [12]:
```
df.head(10)
```

Out[12]:

| | PassengerId | Survived |
|---|---|---|
| **0** | 892 | 0 |
| **1** | 893 | 1 |
| **2** | 894 | 0 |
| **3** | 895 | 0 |
| **4** | 896 | 1 |
| **5** | 897 | 0 |
| **6** | 898 | 1 |
| **7** | 899 | 0 |
| **8** | 900 | 1 |
| **9** | 901 | 0 |

In [13]:
```
features_matrix=df.iloc[:,0:1]
```

In [14]:
```
target_vector=df.iloc[:,-1]
```

In [15]:
```
print('The features matrix has %d rows and %d columns'%(features_matrix.shape
```

```
The features matrix has 418 rows and 1 columns
```

In [16]:
```
print('The target matrix has %d rows and %d columns'%(np.array(target_vector)
```

```
The target matrix has 418 rows and 1 columns
```

In [17]:
```
from sklearn.preprocessing import StandardScaler
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [18]:
```
algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_inter
```

In [21]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_v
```

In [22]:
```
observation=[[1]]
```

In [27]:
```
predictions=Logistic_Regression_Model.predict(observation)
```

In [33]:
```
print('The model predicted the observation to belong to class %s'%(prediction
```

```
The model predicted the observation to belong to class ['g']
```

In [34]:
```
print('The algorithm was trained to predict one of the two classes :%s' %(alg
```

```
The algorithm was trained to predict one of the two classes :['b' 'g']
```

In [30]:
```
print("""The model says the probability of the observation we passed belongin
```

```
The model says the probability of the observation we passed belonging to cla
ss['0'] is 0.6474324251144166
```

In [31]:
```
print("""The model says the probability of the observation we passed belongin
```

```
The model says the probability of the observation we passed belonging to cla
ss['1'] is 0.35256757488558343
```

In [ ]: