```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
%matplotlib inline
```

```
train = pd.read_csv('/content/titanic.csv')
```

```
train.head()
```

	PassengerId	Survived	Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female
4	5	0	3	Allen, Mr. William Henry	male

```
sns.heatmap(train.isnull(),yticklabels=False)
```

gender_submission.csv tita

1 to 10 of 891 entries Filter

ss	Name	Sex	Age	SibSp	Pa
	Braund, Mr. Owen Harris	male	22	1	0
	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0
	Heikkinen, Miss. Laina	female	26	0	0
	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0
	Allen, Mr. William Henry	male	35	0	0
	Moran, Mr. James	male		0	0
	McCarthy, Mr. Timothy J	male	54	0	0
	Palsson, Master. Gosta Leonard	male	2	3	1
	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2
	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0

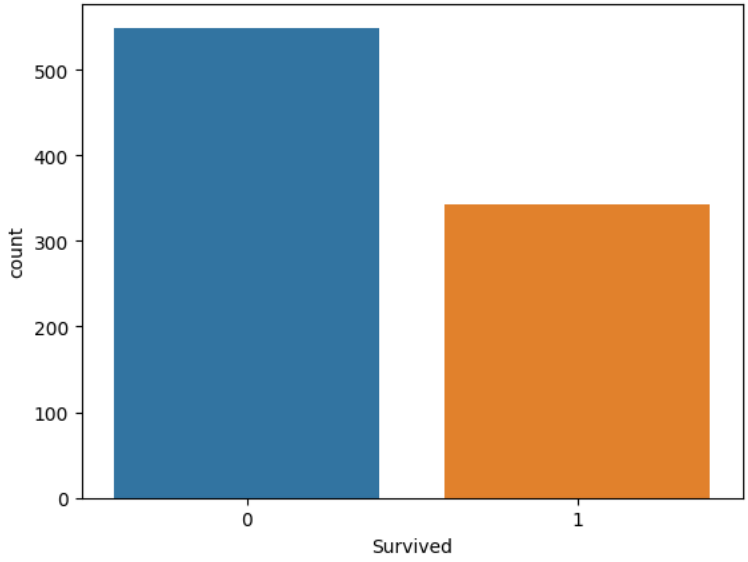
Show 10 per page

1 2 10 80 90

```
<Axes: >
```

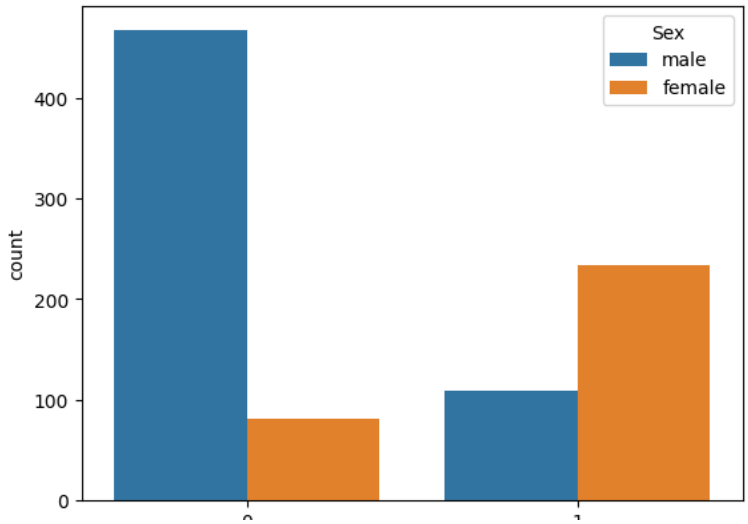

```
sns.countplot(x='Survived', data=train)
```

```
<Axes: xlabel='Survived', ylabel='count'>
```



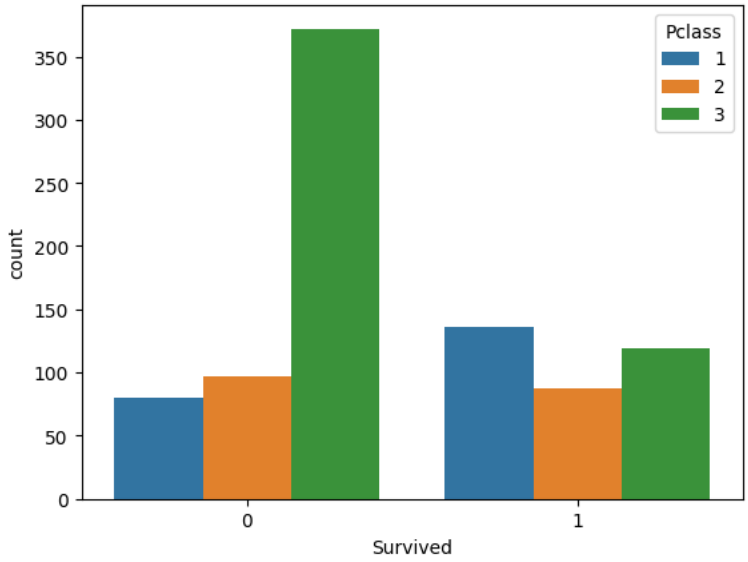

```
sns.countplot(x='Survived', data=train, hue='Sex')
```

```
<Axes: xlabel='Survived', ylabel='count'>
```



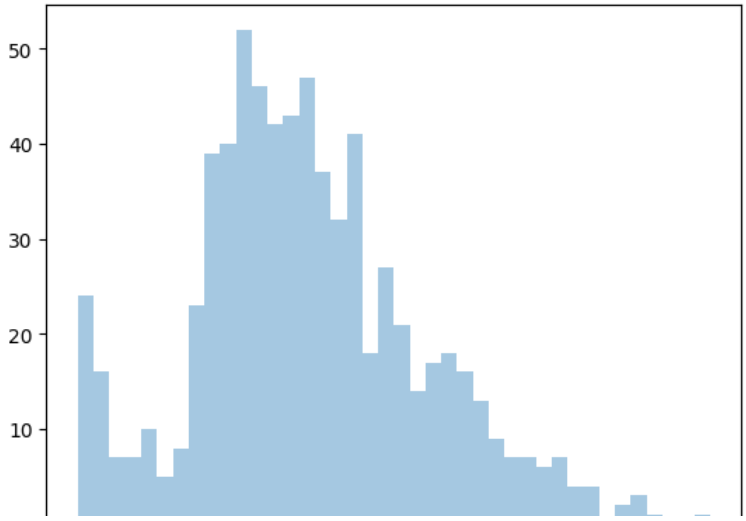

```
sns.countplot(x='Survived', data=train, hue='Pclass')
```

```
<Axes: xlabel='Survived', ylabel='count'>
```




```
sns.distplot(train.Age.dropna(), kde=False, bins=40)
```

```
<ipython-input-42-4da8785f8ce8>:1: UserWarning:

`distplot` is a deprecated function and will be re

Please adapt your code to use either `displot` (a
similar flexibility) or `histplot` (an axes-level

For a guide to updating your code to use the new f
https://gist.github.com/mwaskom/de44147ed2974457ad

  sns.distplot(train.Age.dropna(), kde=False, bins
<Axes: xlabel='Age'>
```

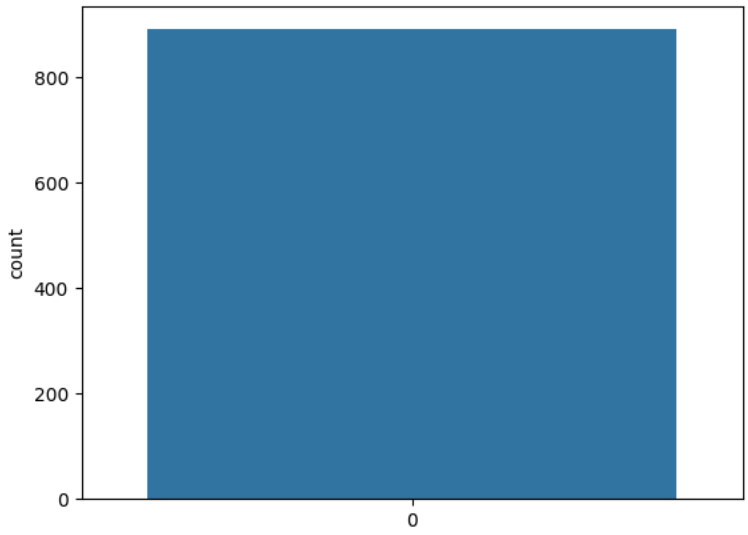



```
sns.distplot(train.Fare.dropna(), kde=False, bins=45)
```

```
<ipython-input-44-9333a98ee152>:1: UserWarning:

`distplot` is a deprecated function and will be re

Please adapt your code to use either `displot` (a
similar flexibility) or `histplot` (an axes-level

For a guide to updating your code to use the new f
```

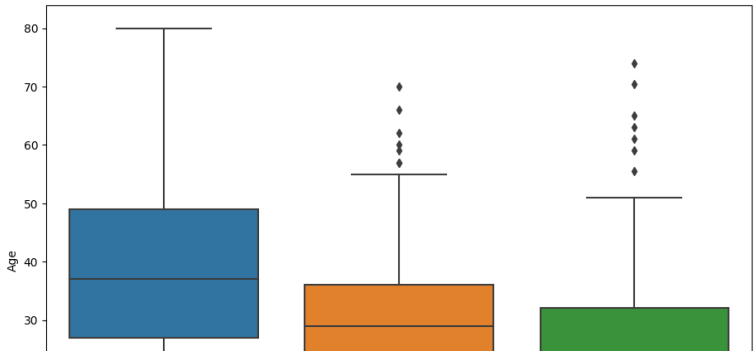
```
sns.countplot(train.SibSp)
```

```
<Axes: ylabel='count'>
```




```
plt.figure(figsize=(11,8))
sns.boxplot(x='Pclass', y='Age', data=train)
```

```
<Axes: xlabel='Pclass', ylabel='Age'>
```



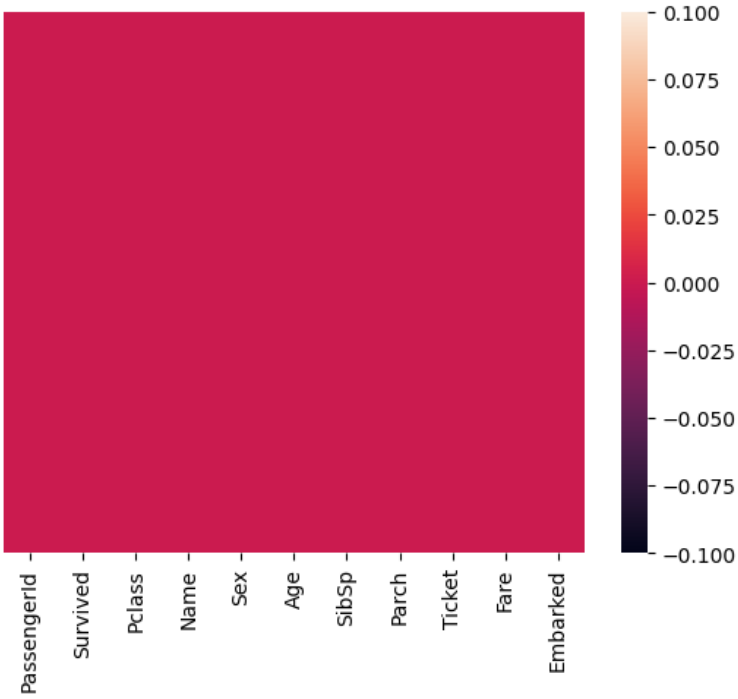

```
def calc_age(col):
    Age = col[0]
    Pass_class = col[1]

    if pd.isnull(Age):

        if Pass_class == 1:
            return 37
        elif Pass_class == 2:
            return 29
        else:
            return 24
    else:
        return Age
train['Age'] = train[['Age','Pclass']].apply(calc_age,


# Missing age values have been filled
sns.heatmap(train.isnull(),yticklabels=False)
```

```
<Axes: >
```


```
train.drop('Cabin', axis=1, inplace=True)
```

```
train.dropna(inplace=True)
sns.heatmap(train.isnull(),yticklabels=False)
```

```
<Axes: >
```



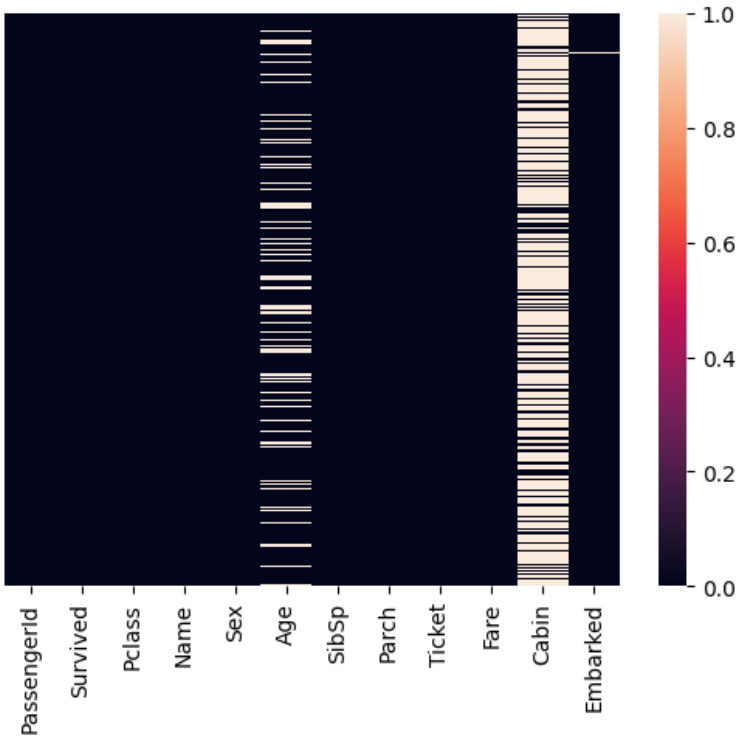

```
# Handling catagorical features
binarysex = pd.get_dummies(train['Sex'],drop_first=Tru
embarked = pd.get_dummies(train['Embarked'], drop_firs
Passengerclass = pd.get_dummies(train['Pclass'],drop_f
train.drop(['Ticket','Embarked','Name','Sex','Passenge
# drop one column to avoid multicollinearity
train = pd.concat([binarysex,embarked,Passengerclass,t


train.head()
```

	male	Q	S	2	3	Survived	Age	SibSp	Parch
0	1	0	1	0	1	0	22.0	1	0

```
# Extracting the training set and test set from the da
X = train.drop('Survived', axis = 1)
y = train['Survived']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,
```

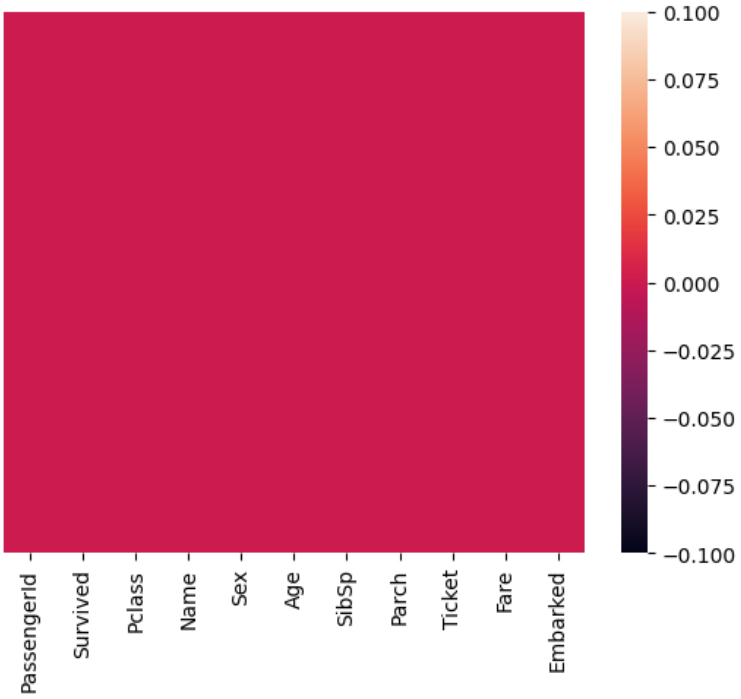
```
# Cleaning and preparing the new Xdata
NewX = pd.read_csv('/content/titanic.csv')


sns.heatmap(NewX.isnull(),yticklabels=False)
```

```
<Axes: >
```




```
NewX['Age'] = NewX[['Age','Pclass']].apply(calc_age, a
NewX.drop('Cabin', axis=1, inplace=True)
NewX.dropna(inplace=True)


sns.heatmap(NewX.isnull(),yticklabels=False)
```

```
<Axes: >
```




```
# Convert feature names to strings for both training a
X_train.columns = X_train.columns.astype(str)
X_test.columns = X_test.columns.astype(str)

# Fitting and applying the logistic regression model w
logmodel = LogisticRegression(max_iter=1000)
logmodel.fit(X_train, y_train)
y_pred = logmodel.predict(X_test)
accuracy = logmodel.score(X_test, y_test)

print("Accuracy:", accuracy)
```

```
Accuracy: 0.7940074906367042
```

```
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

```
                  precision    recall  f1-score   supp

               0       0.81      0.86      0.84
               1       0.77      0.69      0.72

        accuracy                           0.79
       macro avg       0.79      0.77      0.78
    weighted avg       0.79      0.79      0.79
```

```
array([[140,  22],
       [ 33,  72]])
```

```
# Assuming 'Survived' is the target variable to predic
# If it's not, adjust accordingly

# Convert feature names to strings for both training a
X.columns = X.columns.astype(str)
NewX.columns = NewX.columns.astype(str)

# Fitting the logistic regression model on the trainir
logmodel.fit(X, y)

# Remove the 'Survived' column from NewX if it's prese
if 'Survived' in NewX.columns:
    NewX.drop('Survived', axis=1, inplace=True)

# Predicting the new data
y_pred2 = logmodel.predict(NewX)
print(y_pred2)
```

```
[0 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 0 0 0 1 0 0 1 0 1
 0 1 1 0 1 0 1 1 0 0 1 0 1 0 0 1 1 0 0 1 0 1 0 0 0
 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0
 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1
 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0
 1 0 0 0 1 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 1 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 1 1 0 0 0 0 1 1
 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0
 1 0 1 1 0 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 0 0
 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1
 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 0 1 0 1 0 0 0 1 1 0
 0 0 0 0 1 0 0 1 1 1 0 1 0 0 0 0 0 0 1 1 0 0 0 1 1
 1 1 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1
 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 1 1 1 1
 0 1 0 0 1 0 0 1 1 0 0 1 0 0 1 1 1 0 1 0 1 1 0 0 0
 1 1 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 1 0 0 1 1 1 0 1
 1 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 0 0 1 0 1
 0 0 1 0 1 1 0 0 0 0 0 1 1 0 1 0 0 0 0 1 0 1 0 1 1
 0 0 1 1 0 0 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 1 0 1
 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0
 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1
 0 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1
 0 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0
 1 1 1 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 1 1 0 0
 0]
```

```
# Tuning the C value
# Fitting and applying the logistic regression model
import numpy as np
max_score=[]
best_c = []
inter = [0.1,0.3,0.5,0.7,0.9,1.1,1.3,1.5,1.7,1.9,2.1,2
for num in inter:
    logmodel = LogisticRegression(C = num)
    logmodel.fit(X_train,y_train)
    score = logmodel.score(X_test,y_test)
    max_score.append(score)
    best_c.append(num)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
    https://scikit-learn.org/stable/modules/pre
Please also refer to the documentation for alte
    https://scikit-learn.org/stable/modules/lin
  options={"iprint": iprint, "gtol": tol, "maxi
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
    https://scikit-learn.org/stable/modules/pre
Please also refer to the documentation for alte
    https://scikit-learn.org/stable/modules/lin
  options={"iprint": iprint, "gtol": tol, "maxi
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
    https://scikit-learn.org/stable/modules/pre
Please also refer to the documentation for alte
    https://scikit-learn.org/stable/modules/lin
  options={"iprint": iprint, "gtol": tol, "maxi
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
    https://scikit-learn.org/stable/modules/pre
Please also refer to the documentation for alte
    https://scikit-learn.org/stable/modules/lin
  options={"iprint": iprint, "gtol": tol, "maxi
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
    https://scikit-learn.org/stable/modules/pre
Please also refer to the documentation for alte
    https://scikit-learn.org/stable/modules/lin
  options={"iprint": iprint, "gtol": tol, "maxi
/usr/local/lib/python3.10/dist-packages/sklearn
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or
```

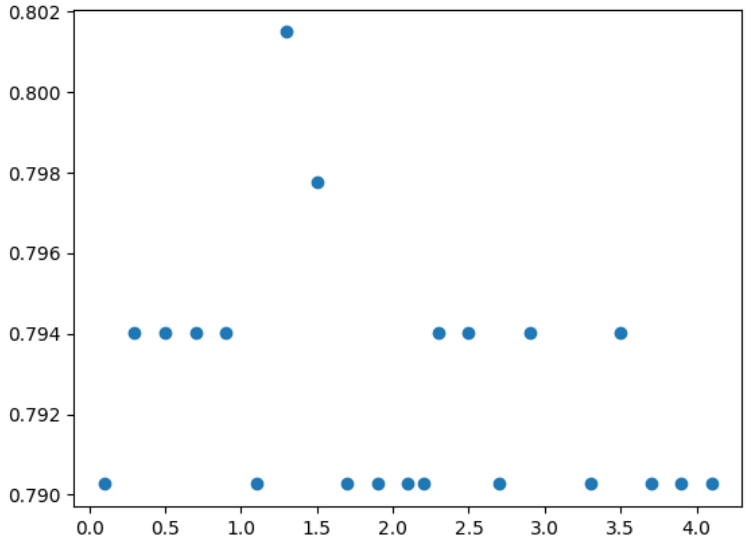
```
        https://scikit-learn.org/stable/modules/pre
    Please also refer to the documentation for alte
        https://scikit-learn.org/stable/modules/lin
      options={"iprint": iprint, "gtol": tol, "maxi
    /usr/local/lib/python3.10/dist-packages/sklearn
    STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

    Increase the number of iterations (max_iter) or
        https://scikit-learn.org/stable/modules/pre
    Please also refer to the documentation for alte
        https://scikit-learn.org/stable/modules/lin
      options={"iprint": iprint, "gtol": tol, "maxi
    /usr/local/lib/python3.10/dist-packages/sklearn
```

```
print(max_score)
print(best_c)
```

```
    [0.7902621722846442, 0.7940074906367042, 0.794007
    [0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9,
```

```
plt.scatter(best_c,max_score)
```

```
    <matplotlib.collections.PathCollection at
    0x7800c52663e0>
```




```
# using the best C we found in our findings
logmodel = LogisticRegression(C = 0.1)
logmodel.fit(X,y)
y_pred2 = logmodel.predict(NewX)
print(y_pred2)
```

```
[0 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 0 0 0 1 0 0 1 0 0
 0 0 1 0 1 0 1 1 0 0 1 0 1 0 0 1 1 0 0 1 0 1 0 0 0
 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0
 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1
 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0
 1 0 0 0 1 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 1 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 1 1 0 0 0 0 1 1
 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0
 1 0 1 1 0 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 0 0
 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1
 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 0 1 0 1 0 0 0 1 1 0
 0 0 0 0 1 0 0 1 1 1 0 1 0 0 0 0 0 0 1 1 0 0 0 1 1
 1 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1
 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 0 1 1 1
 0 1 0 0 1 0 0 1 1 0 0 1 0 0 1 1 1 0 1 0 1 1 0 0 0
 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1
 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0
 0 0 0 0 0 1 0 0 0 0 0 1 1 0 1 0 0 0 0 1 0 1 0 1 1
 0 0 1 1 0 0 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 1 0 1
 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0
 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1
 0 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1
 0 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0
 1 1 1 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 1 1 0 0
 0]
/usr/local/lib/python3.10/dist-packages/sklearn/li
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or sc
    https://scikit-learn.org/stable/modules/preprc
Please also refer to the documentation for alterna
    https://scikit-learn.org/stable/modules/linear
  options={"iprint": iprint, "gtol": tol, "maxiter
```

```
# Assuming 'Survived' is the target variable to predic
# If it's not, adjust accordingly

# Convert feature names to strings for both training a
X.columns = X.columns.astype(str)
NewX.columns = NewX.columns.astype(str)

# Fitting the logistic regression model on the trainir
logmodel = LogisticRegression(C=0.1)
logmodel.fit(X, y)

# Remove the 'Survived' column from NewX if it's prese
if 'Survived' in NewX.columns:
    NewX.drop('Survived', axis=1, inplace=True)

# Predicting the new data with probabilities
y_probabilities = logmodel.predict_proba(NewX)

# Setting a threshold of 0.5 to convert probabilities
threshold = 0.5



# Counting the number of 0s and 1s
num_survived = sum(y_pred_binary)
num_not_survived = len(y_pred_binary) - num_survived

print("Number of survived:", num_survived)
print("Number of not survived:", num_not_survived)
```

```
Number of survived: 288
Number of not survived: 601
/usr/local/lib/python3.10/dist-packages/sklearn/li
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or sc
    https://scikit-learn.org/stable/modules/preprc
Please also refer to the documentation for alterna
```