In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df = pd.read_csv('scaler.csv')
```

In [3]:

```
df.shape
```

Out[3]:

```
(400, 5)
```

In [4]:

```
df.head()
```

Out[4]:

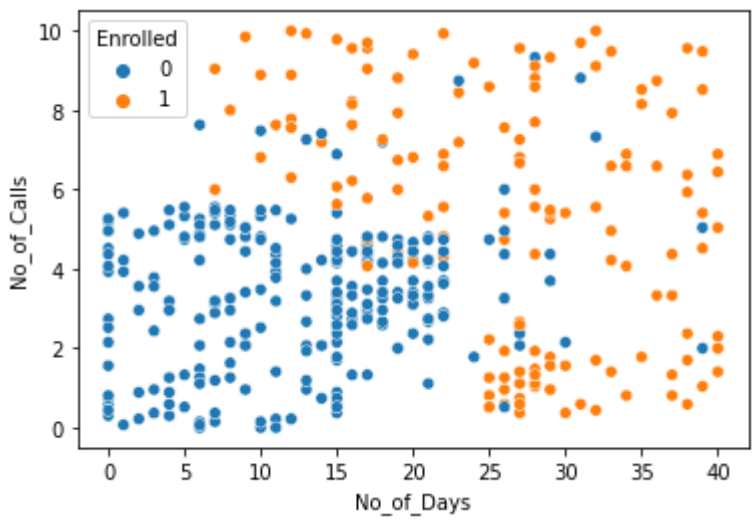
	User_ID	Gender	No_of_Days	No_of_Calls	Enrolled
0	15624510	Male	0	0.30	0
1	15810944	Male	15	0.37	0
2	15668575	Female	6	2.07	0
3	15603246	Female	7	3.11	0
4	15804002	Male	0	4.52	0

In [5]:

```
sns.scatterplot(data=df, x='No_of_Days',y='No_of_Calls', hue = 'Enrolled' )
```

Out[5]:

```
<AxesSubplot:xlabel='No_of_Days', ylabel='No_of_Calls'>
```



In [12]:

```
```

Out[12]:

```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0,
1,
       0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1,
0,
       1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1,
0,
       1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
1,
       0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0,
1,
       1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1,
1,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1,
0,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0,
1,
       0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0,
1,
       1, 1, 0, 1])
```

In [6]:

```
from sklearn.model_selection import train_test_split
```

In [13]:

```
X = df[['No_of_Days', 'No_of_Calls']].values
y = df['Enrolled'].values
```

In [65]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_stat
```

In [66]:

```
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(320, 2) (320,)
(80, 2) (80,)
```

Standardisation

In [67]:

```
mu = X_train.mean(axis=0)
sig = X_train.std(axis=0)
```

In [68]:

```
X_train = (X_train-mu)/sig
X_test = (X_test-mu)/sig
```

k-NN Implementation from Scratch

In [69]:

```
from scipy.stats import mode
```

In [70]:

```
def distance(v1, v2):
    return np.sqrt(np.sum((v1-v2)**2))
```

In [71]:

```
def kNN(X, Y, query_point, k = 5):
    '''Predict the class label for the query point'''

    m = X.shape[0]
    distances = []

    for i in range(m):
        d = distance(X[i], query_point)
        distances.append((d, Y[i]))

    distances = sorted(distances)

    distances = distances[:k]
    distances = np.array(distances)

    labels = distances[: ,1]

    pred = mode(labels)[0][0]

    return int(pred)
```

In [73]:

```
kNN(X_train, y_train, X_test[70])
```

Out[73]:

```
1
```

Actual Class

In [74]:

```
y_test[70]
```

Out[74]:

```
1
```

Accuracy on Test

In [108]:

```
y_pred = np.zeros(X_test.shape[0])

for i in range(X_test.shape[0]):
    y_pred[i] =  kNN(X_train, y_train, X_test[i], k=320)
```

In [109]:

```
from sklearn.metrics import accuracy_score
```

In [110]:

```
round(accuracy_score(y_test, y_pred)*100, 1)
```

Out[110]:

```
65.0
```

Accuracy on Train

In [111]:

```
pred_tr = np.zeros(X_train.shape[0])

for i in range(X_train.shape[0]):
    pred_tr[i] =  kNN(X_train, y_train, X_train[i], k=320)
```

In [112]:

```
round(accuracy_score(y_train, pred_tr)*100, 1)
```

Out[112]:

```
64.1
```

In [ ]:

```
```

Sklearn

In [113]:

```
from sklearn.neighbors import KNeighborsClassifier
```

In [119]:

```
knn = KNeighborsClassifier(n_neighbors=7)
```

In [120]:

```
knn= knn.fit(X_train, y_train)
```

In [121]:

```
y_pred = knn.predict(X_test)
```

In [122]:

```
accuracy_score(y_test, y_pred)
```

Out[122]:

```
0.9375
```

In [ ]:

```
```