The code given below is used to load the model onto the Jupyter Notebook.

```
import torch
import torch.nn as nn
import numpy as np
import pandas as pd

def haversine_distance(df, lat1, long1, lat2, long2):
    r = 6371
    phi1 = np.radians(df[lat1])
    phi2 = np.radians(df[lat2])
    delta_phi = np.radians(df[lat2]-df[lat1])
    delta_lambda = np.radians(df[long2]-df[long1])
    a = np.sin(delta_phi/2)**2 + np.cos(phi1) *
np.cos(phi2) * np.sin(delta_lambda/2)**2
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1-a))
    return r * c

class TabularModel(nn.Module):
  def __init__(self, n_cont, out_sz, layers, p):

    super().__init__()
    self.emb_drop = nn.Dropout(p)
    self.cont_norm = nn.BatchNorm1d(n_cont)
    layerlist = []

    for i in layers:
      layerlist.append(nn.Linear(n_cont,i))
      layerlist.append(nn.ReLU(inplace=True))
      layerlist.append(nn.BatchNorm1d(i))
      layerlist.append(nn.Dropout(p))
      n_cont = i
    layerlist.append(nn.Linear(layers[-1],out_sz))
    self.layers = nn.Sequential(*layerlist)

  def forward(self, x_cont):
    x_cont = self.emb_drop(x_cont)
```

```
        x_cont = self.cont_norm(x_cont)
        x_cont = self.layers(x_cont)
        return x_cont

new_model = TabularModel(6,2,[200,100,100,64], p = 0.4)

new_model.load_state_dict(torch.load('Bikeshare.pt'))
new_model.eval()
```

To predict new data, use the code given below:

```
def test_data(mdl): # pass in the name of the new model
    # INPUT NEW DATA
    trip_duration = float(input('What is the trip
duration '))
    plat = float(input('What is the pickup latitude?
'))
    plong = float(input('What is the pickup longitude?
'))
    dlat = float(input('What is the dropoff latitude?
'))
    dlong = float(input('What is the dropoff longitude?
'))

    # PREPROCESS THE DATA
    dfx_dict =
{'trip_duration':trip_duration,'pickup_latitude':plat,'
pickup_longitude':plong,'dropoff_latitude':dlat,
          'dropoff_longitude':dlong}
    dfx = pd.DataFrame(dfx_dict, index=[0])
    dfx['trip_distance'] =
haversine_distance(dfx,'pickup_latitude',
'pickup_longitude',

'dropoff_latitude', 'dropoff_longitude')

    cont_cols = ['trip_duration','pickup_latitude',
'pickup_longitude', 'dropoff_latitude',
```

```
                'dropoff_longitude', 'trip_distance']
    xconts = np.stack([dfx[col].values for col in
cont_cols], 1)
    xconts = torch.tensor(xconts, dtype=torch.float)

    # PASS NEW DATA THROUGH THE MODEL WITHOUT PERFORMING
A BACKPROP
    with torch.no_grad():
        z = mdl(xconts).argmax().item()
    print(f'\nThe Predicted User Type is {z}')
```