In [9]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [10]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\C9_Data.csv")
df
```

Out[10]:

	row_id	user_id	timestamp	gate_id
0	0	18	2022-07-29 09:08:54	7
1	1	18	2022-07-29 09:09:54	9
2	2	18	2022-07-29 09:09:54	9
3	3	18	2022-07-29 09:10:06	5
4	4	18	2022-07-29 09:10:08	5
...	...	...	...	...
37513	37513	6	2022-12-31 20:38:56	11
37514	37514	6	2022-12-31 20:39:22	6
37515	37515	6	2022-12-31 20:39:23	6
37516	37516	6	2022-12-31 20:39:31	9
37517	37517	6	2022-12-31 20:39:31	9

37518 rows × 4 columns

In [11]:
```
ge=df[['row_id','user_id','gate_id']]
```

In [12]:
```
d=ge.fillna(20)
```

In [13]:
```
ge['gate_id'].value_counts()
```

Out[13]:
```
 4     8170
 3     5351
 10    4767
 5     4619
 11    4090
 9     3390
 7     3026
 6     1800
 13    1201
 12     698
 15     298
-1       48
 8       48
 1        5
 16       4
 0        2
 14       1
Name: gate_id, dtype: int64
```

In [14]:
```
x=ge.drop('gate_id',axis=1)
y=ge['gate_id']
```

In [15]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [16]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[16]:
```
RandomForestClassifier()
```

In [17]:
```
paramets = {'max_depth':[1,2,3,4,5],
              'min_samples_leaf':[5,10,15,20,25],
              'n_estimators':[10,20,30,40,50]}
```

In [18]:
```
from sklearn.model_selection import GridSearchCV
grid_search= GridSearchCV(estimator = rfc,param_grid=paramets,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarnin
g: The least populated class in y has only 1 members, which is less than n_splits=2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[18]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [19]:
```
grid_search.best_score_
```

Out[19]:
```
0.22237453354656916
```

In [20]:
```
rfc_best=grid_search.best_estimator_
```

In [21]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-21-d9a5e60a8034> in <module>
      1 from sklearn.tree import plot_tree
      2 plt.figure(figsize=(80,40))
----> 3 plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],fi
lled=True)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **kw
args)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in plot_tree(decision_tre
e, max_depth, feature_names, class_names, label, filled, impurity, node_ids, proportion, rota
te, rounded, precision, ax, fontsize)
    192         proportion=proportion, rotate=rotate, rounded=rounded,
    193         precision=precision, fontsize=fontsize)
--> 194     return exporter.export(decision_tree, ax=ax)
    195
    196

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in export(self, decision_t
ree, ax)
    582         ax.clear()
    583         ax.set_axis_off()
--> 584         my_tree = self._make_tree(0, decision_tree.tree_,
    585                                   decision_tree.criterion)
    586         draw_tree = buchheim(my_tree)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in _make_tree(self, node_i
d, et, criterion, depth)
    563         # traverses _tree.Tree recursively, builds intermediate
    564         # "_reingold_tilford.Tree" object
--> 565         name = self.node_to_str(et, node_id, criterion=criterion)
    566         if (et.children_left[node_id] != _tree.TREE_LEAF
    567                 and (self.max_depth is None or depth <= self.max_depth)):

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in node_to_str(self, tree,
node_id, criterion)
    353                 node_string += 'class = '
    354             if self.class_names is not True:
--> 355                 class_name = self.class_names[np.argmax(value)]
    356             else:
    357                 class_name = "y%s%s%s" % (characters[1],

IndexError: list index out of range
```

In [ ]: