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(r"C9_Data.csv")
df
```

Out[2]:

	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 [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37518 entries, 0 to 37517
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   row_id     37518 non-null  int64 
 1   user_id    37518 non-null  int64 
 2   timestamp  37518 non-null  object
 3   gate_id    37518 non-null  int64 
dtypes: int64(3), object(1)
memory usage: 1.1+ MB
```

In [4]:
```
df=df.dropna()
```

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
row_id       0
user_id      0
timestamp    0
gate_id      0
dtype: int64
```

In [6]:
```
df.describe()
```

Out[6]:

	row_id	user_id	gate_id
count	37518.000000	37518.000000	37518.000000
mean	18758.500000	28.219015	6.819607
std	10830.658036	17.854464	3.197746
min	0.000000	0.000000	-1.000000
25%	9379.250000	12.000000	4.000000
50%	18758.500000	29.000000	6.000000
75%	28137.750000	47.000000	10.000000
max	37517.000000	57.000000	16.000000

In [7]:
```
df.columns
```

Out[7]:
```
Index(['row_id', 'user_id', 'timestamp', 'gate_id'], dtype='object')
```

In [8]:
```
df['user_id'].value_counts()
```

Out[8]:

37	2262
55	2238
6	2013
12	1953
19	1793
15	1756
18	1578
47	1341
53	1311
1	1299
33	1285
11	1281
49	1275
0	1250
39	1144
32	1076
54	1070
9	1034
50	994
29	990
3	989
48	743
14	696
17	677
27	603
35	601
46	502
57	497
24	416
42	359
26	316
34	284
23	261
25	247
40	242
31	191
56	137
43	124
41	124
20	115
22	96
28	64
45	57
7	49
36	48
2	39
8	29
10	17
38	13
30	10
5	10
21	5
52	5
44	4
51	3

```
4        2
Name: user_id, dtype: int64
```

In [9]:
```
g1={"gate_id":{'6':1,'5':4}}
df=df.replace(g1)
print(df)
```

```
       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 x 4 columns]
```

In [10]:
```
x=df.drop(["row_id","timestamp"],axis=1)
y=df["row_id"]
```

In [11]:
```
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 [12]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)
<ipython-input-12-3872a98b0b86> in <module>
      1 from sklearn.ensemble import RandomForestClassifier
      2 rfc=RandomForestClassifier()
----> 3 rfc.fit(x_train,y_train)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\_forest.py in fit
(self, X, y, sample_weight)
    385             # parallel_backend contexts set at a higher level,
    386             # since correctness does not rely on using threads.
--> 387             trees = Parallel(n_jobs=self.n_jobs, verbose=self.verbos
e,
    388                              **_joblib_parallel_args(prefer='thread
s'))(
    389                 delayed(_parallel_build_trees)(

C:\ProgramData\Anaconda3\lib\site-packages\joblib\parallel.py in __call__(sel
f, iterable)
   1042                 self._iterating = self._original_iterator is not None
   1043
-> 1044             while self.dispatch_one_batch(iterator):
   1045                 pass
   1046

C:\ProgramData\Anaconda3\lib\site-packages\joblib\parallel.py in dispatch_one
_batch(self, iterator)
    857                 return False
    858             else:
--> 859                 self._dispatch(tasks)
    860                 return True
    861

C:\ProgramData\Anaconda3\lib\site-packages\joblib\parallel.py in _dispatch(se
lf, batch)
    775         with self._lock:
    776             job_idx = len(self._jobs)
--> 777             job = self._backend.apply_async(batch, callback=cb)
    778             # A job can complete so quickly than its callback is
    779             # called before we get here, causing self._jobs to

C:\ProgramData\Anaconda3\lib\site-packages\joblib\_parallel_backends.py in ap
ply_async(self, func, callback)
    206     def apply_async(self, func, callback=None):
    207         """Schedule a func to be run"""
--> 208         result = ImmediateResult(func)
    209         if callback:
    210             callback(result)

C:\ProgramData\Anaconda3\lib\site-packages\joblib\_parallel_backends.py in __
init__(self, batch)
    570         # Don't delay the application, to avoid keeping the input
    571         # arguments in memory
--> 572         self.results = batch()
    573
    574     def get(self):
```

```
C:\ProgramData\Anaconda3\lib\site-packages\joblib\parallel.py in __call__(sel
f)
    260         # change the default number of processes to -1
    261         with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 262             return [func(*args, **kwargs)
    263                     for func, args, kwargs in self.items]
    264

C:\ProgramData\Anaconda3\lib\site-packages\joblib\parallel.py in <listcomp>(.
0)
    260         # change the default number of processes to -1
    261         with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 262             return [func(*args, **kwargs)
    263                     for func, args, kwargs in self.items]
    264

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\fixes.py in __call__
(self, *args, **kwargs)
    220     def __call__(self, *args, **kwargs):
    221         with config_context(**self.config):
--> 222             return self.function(*args, **kwargs)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\_forest.py in _pa
rallel_build_trees(tree, forest, X, y, sample_weight, tree_idx, n_trees, verb
ose, class_weight, n_samples_bootstrap)
    167                                                     indices=indic
es)
    168
--> 169         tree.fit(X, y, sample_weight=curr_sample_weight, check_input=
False)
    170     else:
    171         tree.fit(X, y, sample_weight=sample_weight, check_input=Fals
e)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_classes.py in fit(se
lf, X, y, sample_weight, check_input, X_idx_sorted)
    896         """
    897
--> 898         super().fit(
    899             X, y,
    900             sample_weight=sample_weight,

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_classes.py in fit(se
lf, X, y, sample_weight, check_input, X_idx_sorted)
    387                                            min_impurity_split)
    388
--> 389         builder.build(self.tree_, X, y, sample_weight)
    390
    391         if self.n_outputs_ == 1 and is_classifier(self):

sklearn\tree\_tree.pyx in sklearn.tree._tree.DepthFirstTreeBuilder.build()

sklearn\tree\_tree.pyx in sklearn.tree._tree.DepthFirstTreeBuilder.build()

sklearn\tree\_tree.pyx in sklearn.tree._tree.Tree._add_node()

sklearn\tree\_tree.pyx in sklearn.tree._tree.Tree._resize_c()
```

```
sklearn\tree\_utils.pyx in sklearn.tree._utils.safe_realloc()

MemoryError: could not allocate 215138304 bytes
```

In [13]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]}
```

In [14]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accu
grid_search.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-14-fe48119762b8> in <module>
      1 from sklearn.model_selection import GridSearchCV
      2 grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,sco
ring="accuracy")
----> 3 grid_search.fit(x_train,y_train)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inn
er_f(*args, **kwargs)
     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\model_selection\_search.py
in fit(self, X, y, groups, **fit_params)
    839                 return results
    840
--> 841             self._run_search(evaluate_candidates)
    842
    843             # multimetric is determined here because in the case of a
callable

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_search.py
in _run_search(self, evaluate_candidates)
   1286     def _run_search(self, evaluate_candidates):
   1287         """Search all candidates in param_grid"""
-> 1288         evaluate_candidates(ParameterGrid(self.param_grid))
   1289
   1290

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_search.py
in evaluate_candidates(candidate_params, cv, more_results)
    805                                                 **fit_and_scor
e_kwargs)
    806                                 for (cand_idx, parameters),
--> 807                                     (split_idx, (train, test)) in prod
uct(
    808                                     enumerate(candidate_params),
    809                                     enumerate(cv.split(X, y, group
s))))

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py
in split(self, X, y, groups)
    330                 .format(self.n_splits, n_samples))
    331
--> 332         for train, test in super().split(X, y, groups):
    333             yield train, test
    334

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py
in split(self, X, y, groups)
     78         X, y, groups = indexable(X, y, groups)
     79         indices = np.arange(_num_samples(X))
---> 80         for test_index in self._iter_test_masks(X, y, groups):
```

```
     81             train_index = indices[np.logical_not(test_index)]
     82             test_index = indices[test_index]

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py
in _iter_test_masks(self, X, y, groups)
    691
    692     def _iter_test_masks(self, X, y=None, groups=None):
--> 693         test_folds = self._make_test_folds(X, y)
    694         for i in range(self.n_splits):
    695             yield test_folds == i

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py
in _make_test_folds(self, X, y)
    660         min_groups = np.min(y_counts)
    661         if np.all(self.n_splits > y_counts):
--> 662             raise ValueError("n_splits=%d cannot be greater than the"
    663                              " number of members in each class."
    664                              % (self.n_splits))

ValueError: n_splits=2 cannot be greater than the number of members in each c
lass.
```

In [15]:
```
grid_search.best_score_
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-15-99e6964859a0> in <module>
----> 1 grid_search.best_score_

AttributeError: 'GridSearchCV' object has no attribute 'best_score_'
```

In [16]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]}
```

In [17]:
```
rfc_best=grid_search.best_estimator_
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-17-bc0e385e72ac> in <module>
----> 1 rfc_best=grid_search.best_estimator_

AttributeError: 'GridSearchCV' object has no attribute 'best_estimator_'
```

In [18]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','N
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-18-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'],filled=True)

NameError: name 'rfc_best' is not defined


<Figure size 5760x2880 with 0 Axes>
```

In [ ]: