INSTRUCTION BEFORE RUNNING:

I use google collaboratory to build model as well as write insights for report here. In .zip file submitted, a PDF version of this notebook is included as report for lab project

Confusion matrix, Decision tree image and Classification report are included in submit folder. Please visit it to see all of them because decision tree images are big

Please upload file connect-4.data to directory of google colab. Link to .data file is somehow look like: /content/connect-4.data

- **Run pip command line below to install/ re-install lastest scikit-learn package because there are some function I used from the lastest version of scikit-learn**

In [ ]:

```
!pip install -U scikit-learn
```

```
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-packages (1.0.1)
Collecting scikit-learn
  Downloading scikit_learn-1.0.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (24.8 MB)
     |████████████████████████████████| 24.8 MB 1.4 MB/s
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn) (1.1.0)
Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn) (1.4.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn) (3.0.0)
Requirement already satisfied: numpy>=1.14.6 in /usr/local/lib/python3.7/dist-packages (from scikit-learn) (1.19.5)
Installing collected packages: scikit-learn
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.0.1
    Uninstalling scikit-learn-1.0.1:
      Successfully uninstalled scikit-learn-1.0.1
Successfully installed scikit-learn-1.0.2
```

In [ ]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import classification_report, confusion_matrix, precision_recall_curve, auc, roc_curve
from sklearn.tree import DecisionTreeClassifier, export_graphviz
import graphviz
import os
import warnings
warnings.filterwarnings('always')
```

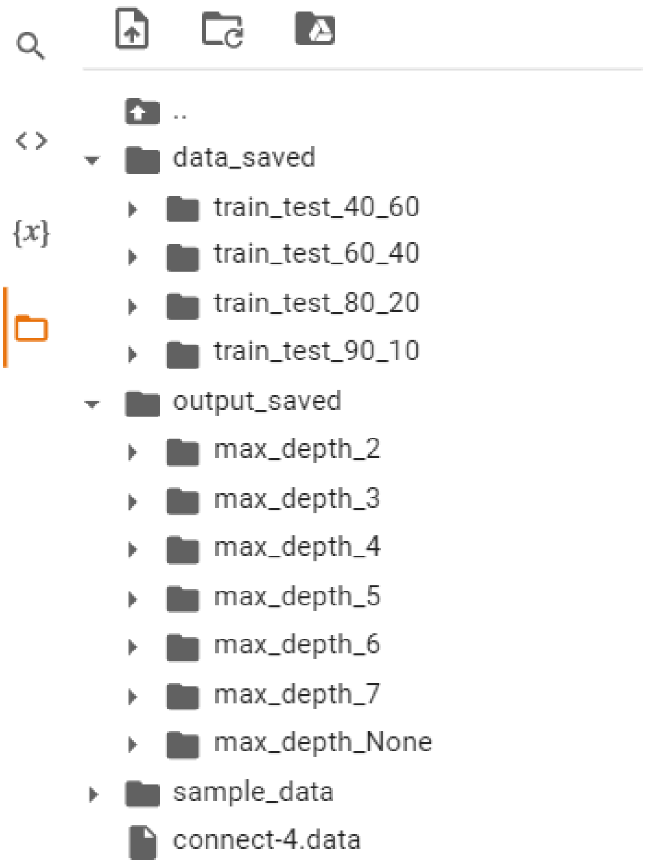
In [ ]:

```
import shutil
import shutil
def remove_folder(path):
    # check if folder exists
    if os.path.exists(path):
        # remove if exists
        shutil.rmtree(path)
    else:
        # throw your exception to handle this special scenario
        raise Exception("your exception")
```

In [ ]:

```
# create folders, utilities, ...
FILE_DATA = "/content/connect-4.data"
FILE_NAME = "/content/connect-4.names"
DATA_SAVE = "/content/data_saved"
OUTPUT_SAVE = "/content/output_saved"

percentage_tests = [60, 40, 20, 10]
MAX_DEPTH = [None, 2,3,4,5,6,7]

if os.path.isdir(DATA_SAVE):
  remove_folder(DATA_SAVE)
  os.mkdir(DATA_SAVE)
else:
  os.mkdir(DATA_SAVE)

for percent in percentage_tests:
  os.mkdir(DATA_SAVE + "/train_test_{}_{}".format(100-percent, percent))

if os.path.isdir(OUTPUT_SAVE):
  remove_folder(OUTPUT_SAVE)
  os.mkdir(OUTPUT_SAVE)
else:
  os.mkdir(OUTPUT_SAVE)

for depth in MAX_DEPTH:
  os.mkdir(OUTPUT_SAVE + "/max_depth_{}".format(depth))
```

After creating vital folders, the directory of project somehow look like:



```
..
data_saved
    train_test_40_60
    train_test_60_40
    train_test_80_20
    train_test_90_10
output_saved
    max_depth_2
    max_depth_3
    max_depth_4
    max_depth_5
    max_depth_6
    max_depth_7
    max_depth_None
sample_data
connect-4.data
```

In [ ]:

```
def read_data(file_data = FILE_DATA, file_name = FILE_NAME, percentage_test = 20):
  raw_df = pd.read_csv(file_data, sep=",")
  labelencoder = LabelEncoder()
  # transform values of attributes to trainable form (numbers)
  for column in raw_df.columns:
      raw_df[column] = labelencoder.fit_transform(raw_df[column])

  X, y = raw_df.drop('win', axis=1) , raw_df["win"]
  X_train, X_test, y_train, y_test = train_test_split_data(X, y, percentage_test)
  return X_train, X_test, y_train, y_test
```

In [182]:

```
def train_test_split_data(X, y, percentage_test, isDepth = False):
  X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=percentage_test/1
00, random_state=42, stratify=y, shuffle=True)

  train_set = pd.concat([X_train, y_train], axis=1)
  test_set = pd.concat([X_test, y_test], axis=1)
  # just save subset data for part 1
  if isDepth == False:
    X_train.to_csv("/{}/train_test_{}_{}/feature_train_{}_{}.dat".format(DATA_SAVE,100-p
ercentage_test, percentage_test,100-percentage_test, percentage_test), sep = "|")
    y_train.to_csv("/{}/train_test_{}_{}/label_train_{}_{}.dat".format(DATA_SAVE,100-perc
entage_test, percentage_test,100-percentage_test, percentage_test), sep = "|")
    X_test.to_csv("/{}/train_test_{}_{}/feature_test_{}_{}.dat".format(DATA_SAVE,100-perc
entage_test, percentage_test,100- percentage_test, percentage_test), sep = "|")
    y_test.to_csv("/{}/train_test_{}_{}/label_test_{}_{}.dat".format(DATA_SAVE,100-percen
tage_test, percentage_test,100-percentage_test, percentage_test), sep = "|")

  return X_train, X_test, y_train, y_test
```

In [ ]:

```
# pick randomly 22 attributes from original set
attributes = ['b.9', 'b.13', 'b.24', 'b.2', 'b.30',
       'b.1', 'b.29', 'b.17', 'b.23', 'b.8', 'b.28', 'b', 'x.2', 'b.12',
       'b.7', 'o.1', 'b.6', 'x', 'o', 'b.22', 'b.16', 'x.1']
```

In [ ]:

```
def classification_report_csv(report, percentage_test, depth, isDepth = False):
    dataframe = pd.DataFrame(report).transpose()
    if isDepth == False:
      dataframe.to_csv(DATA_SAVE + '/train_test_{}_{}/classification_report_{}_{}.csv'.f
ormat(100-percentage_test, percentage_test,100-percentage_test, percentage_test), index =
False)
    else :
      dataframe.to_csv(OUTPUT_SAVE + '/max_depth_{}/classification_report_{}_{}.csv'.form
at(depth ,100-percentage_test, percentage_test), index = False)
```

In [ ]:

```
def export_graphviz_to_png(graph, percent, depth, isDepth = False):
  png_bytes = graph.pipe(format='png')
  if isDepth == False:
    with open(DATA_SAVE + '/train_test_{}_{}/decision_tree_{}_{}.png'.format(100-percent,
percent, 100-percent, percent),'wb') as f:
      f.write(png_bytes)
  else:
    with open(OUTPUT_SAVE + '/max_depth_{}/decision_tree_{}_{}.png'.format(depth, 100-per
cent, percent),'wb') as f:
      f.write(png_bytes)
```

In [ ]:

```
def train(depth = None, percentage_test = 60 ,isDepth = False):
  """
  depth - max_depth of decision tree. default: None
  percentage_test - percent (on 100%) of test set. default: 60
  isDepth - used for part 2 (training model with different max_depth). If True, files will be saved to OUTPUT_SAVE directory, else saved to DATA_SAVE directory
  """
  clf = DecisionTreeClassifier(max_depth=depth)
  X_train, X_test, y_train, y_test = read_data(percentage_test = percentage_test)
  X_train = X_train[attributes]
  X_test = X_test[attributes]

  clf = clf.fit(X_train, y_train)
  dot_data = export_graphviz(clf, out_file=None,
                        feature_names=X_train.columns,
                        filled=True, rounded=True,
                        special_characters=True)
  graph = graphviz.Source(dot_data)
  y_pred = clf.predict(X_test)
  report = classification_report(y_test, y_pred, output_dict = True)
  classification_report_csv(report, percentage_test, isDepth )

  print("Decision Tree Classifier report \n", classification_report(y_test, y_pred))
  cfm=confusion_matrix(y_test, y_pred )

  sns.heatmap(cfm, annot = True, fmt='g', linewidths=.5, cbar =None)
  plt.title('Decision Tree Classifier confusion matrix')
  plt.ylabel('True label')
  plt.xlabel('Predicted label');

  if isDepth == False:
    plt.savefig(DATA_SAVE + '/train_test_{}_{}/confusion_matrix_{}_{}.png'.format(100-percentage_test, percentage_test,100-percentage_test, percentage_test), dpi=400)
    export_graphviz_to_png(graph, percentage_test,depth, isDepth)
  else:
    plt.savefig(OUTPUT_SAVE + '/max_depth_{}/confusion_matrix_{}_{}.png'.format(depth ,100-percentage_test, percentage_test), dpi=400)
    export_graphviz_to_png(graph, percentage_test,depth, isDepth)
```

PART 1 : Define Model Decision Tree and Train with 4 kinds of spliting data

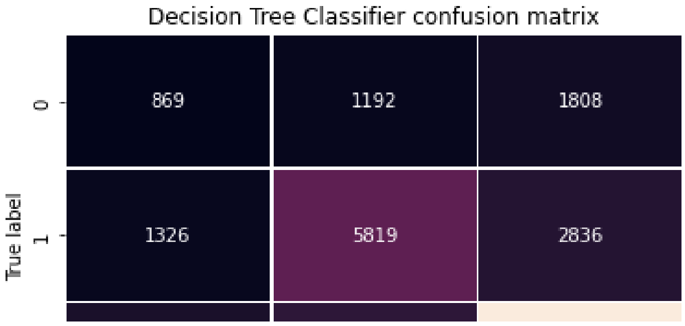
- 40 - 60
- 60 - 40
- 80 - 20
- 90 - 10

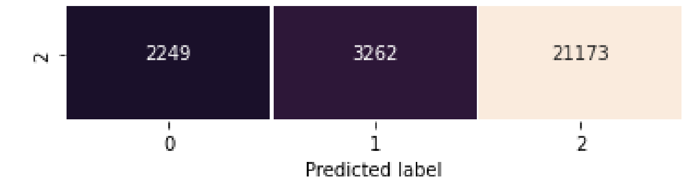
In [ ]:

```
train(depth = None, percentage_test = 60)
```

```
Decision Tree Classifier report 
               precision    recall  f1-score   support

           0       0.20      0.22      0.21      3869
           1       0.57      0.58      0.57      9981
           2       0.82      0.79      0.81     26684

    accuracy                           0.69     40534
   macro avg       0.53      0.53      0.53     40534
weighted avg       0.70      0.69      0.69     40534
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.0658634 to fit
```



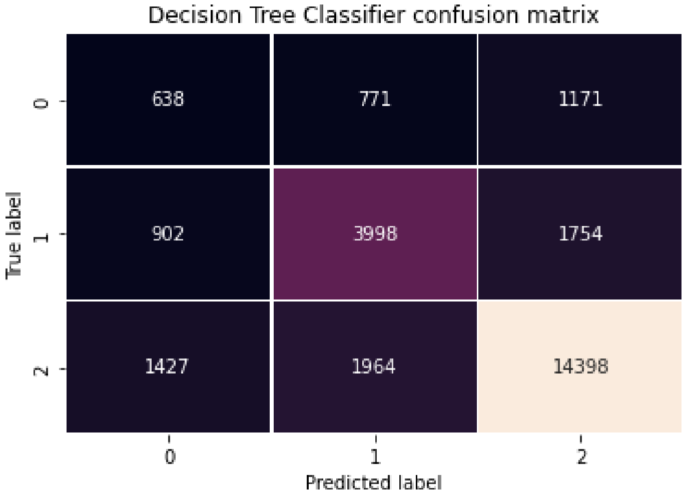


In [ ]:

```
train(depth = None, percentage_test = 40)
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.22      0.25      0.23      2580
           1       0.59      0.60      0.60      6654
           2       0.83      0.81      0.82     17789

    accuracy                           0.70     27023
   macro avg       0.55      0.55      0.55     27023
weighted avg       0.71      0.70      0.71     27023
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.0451335 to fit
```

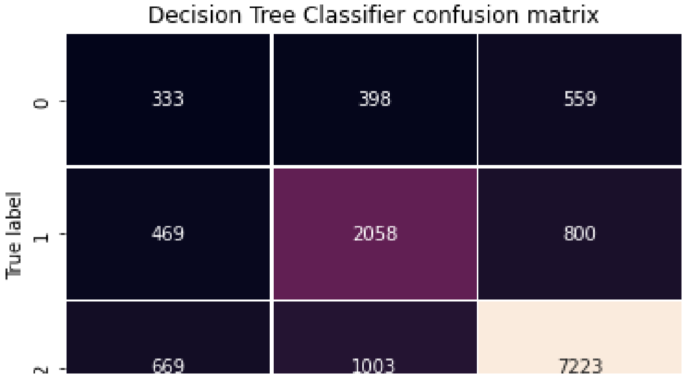


In [ ]:

```
train(depth = None, percentage_test = 20)
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.23      0.26      0.24      1290
           1       0.59      0.62      0.61      3327
           2       0.84      0.81      0.83      8895

    accuracy                           0.71     13512
   macro avg       0.55      0.56      0.56     13512
weighted avg       0.72      0.71      0.72     13512
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.0346389 to fit
```

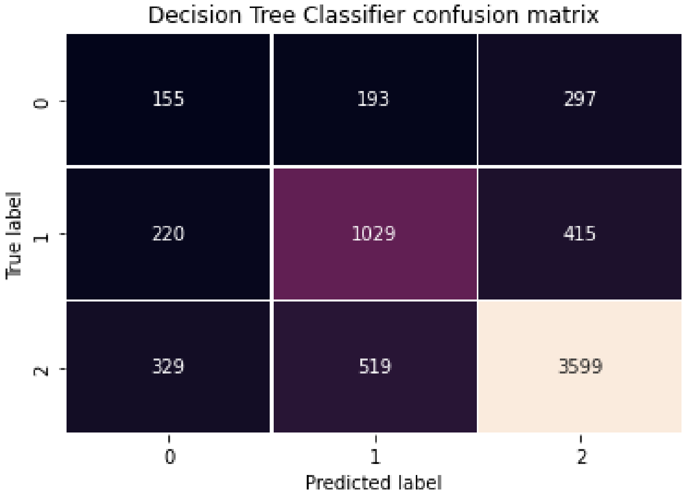


In [ ]:

```
train(depth = None, percentage_test = 10)
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.22      0.24      0.23       645
           1       0.59      0.62      0.60      1664
           2       0.83      0.81      0.82      4447

    accuracy                           0.71      6756
   macro avg       0.55      0.56      0.55      6756
weighted avg       0.72      0.71      0.71      6756
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.0311851 to fit
```



In [ ]:

```
# You do not need to run this command since I use it to extract files to zip and download
it to local
# !zip -r /content/train_test_90_10.zip /content/data_saved/train_test_90_10
# from google.colab import files
# files.download("/content/train_test_90_10.zip")
```

Interpretation from confusion matrix and classification report + comments of train_test proportion = 40 : 60

From confusion matrix

I think that before diving deep into interpretation of classification report, it's vital to have a look at confusion matrix first. Confusion matrix is a NxN matrix with N is the number of labels. 'True label' is ground-truth label that we need to classify data into 'Predicted label' is prediction from model Hence, each cell (row A, column B) of matrix represents the number of samples predicted wrongly on label B but should be classified correctly on label A For example, cell (row 1, column 2) (2876 samples) is the number of samples predicted on label 2 (wrong) and it should be label 1 (true) instead From above inspections, we have that the cross line (cell (0,0), (1,1), (2,2)) are samples which are classified on correct labels Label 0: only 868 samples are correctly classified Label 1: 5766 samples are correctly classified Label 2: 21169 samples are correctly classified Model seems to work well on samples with label 2 and bad on labels 0 Besides, number of wrong predictions varies from 1181 (cell (0, 1)) to ~3287 (cell (2, 1))

With only confusion matrix, it's vague to conclude our model works or not since we need to consider not only corrected samples but how many percentage they make up in total

From classification report Precision: percent of predictions are correct For example: with label 0, Precision ~ 868 / (868 + 1339 + 2228) ~ 0.20 This means, in occurences predicted as label 0, only 0.20 percent of them is classified true Precision is important, especially in fraud detection. For example, the email user might lose important emails if the precision is not high for the spam detection model.

Recall: percent of the positive cases For example: with label 0, Recall ~ 868 / (868 + 1181 + 1820) ~ 0.22 This means, in occurences labeled as 0, only 0.22 percent of them is classified true Recall is important as well. In spam detection, if recall is low, many spam emails are not predicted.

F-1 score: percent of positive predictions were correct For example: with label 0, F-1 = 2 *Precision* Recall / (Precision + Recall) = 0.21 F-1 score is needed because we want to find a metric that balances Precision and Recall. F-score aligns between Recall and Precision

Support is total occurrences of the class (sum of values in row)

Label 2 with high F2-score while Label 0 is low and Label 1 is medium

We can apply the same interpretation for other cases

There is not too much difference between 4 kinds of spliting data but if you compare the accuracy of them, case 80-20 and 90-10 have better chance of correct prediction. By theory, spliting dataset into subsets with ratio 70-30, or 80-20 or even 90-10 is a good choice at the starting point

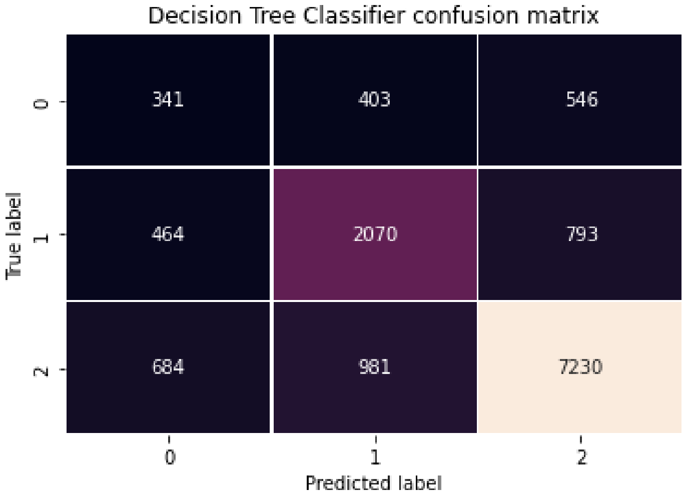
PART 2 : max_depth investigation with 80-20 spliting

In [ ]:

```
train(None, percentage_test=20, isDepth = True)
```

```
Decision Tree Classifier report
               precision    recall  f1-score   support

           0       0.23      0.26      0.25      1290
           1       0.60      0.62      0.61      3327
           2       0.84      0.81      0.83      8895

    accuracy                           0.71     13512
   macro avg       0.56      0.57      0.56     13512
weighted avg       0.72      0.71      0.72     13512
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.034617 to fit
```



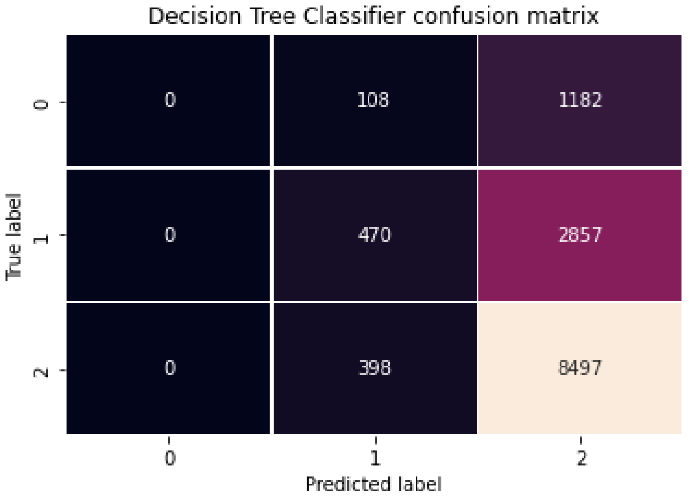
In [ ]:

```
train(2, percentage_test = 20, isDepth = True)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.48      0.14      0.22      3327
           2       0.68      0.96      0.79      8895

    accuracy                           0.66     13512
   macro avg       0.39      0.37      0.34     13512
weighted avg       0.56      0.66      0.58     13512
```


Decision Tree Classifier confusion matrix

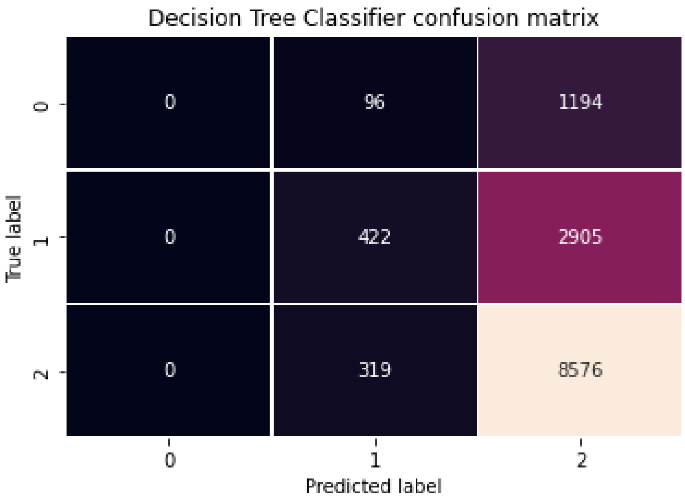
In [ ]:

```
train(3, percentage_test = 20, isDepth = True)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
```

```
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.50      0.13      0.20      3327
           2       0.68      0.96      0.80      8895

    accuracy                           0.67     13512
   macro avg       0.39      0.36      0.33     13512
weighted avg       0.57      0.67      0.57     13512
```



In [ ]:

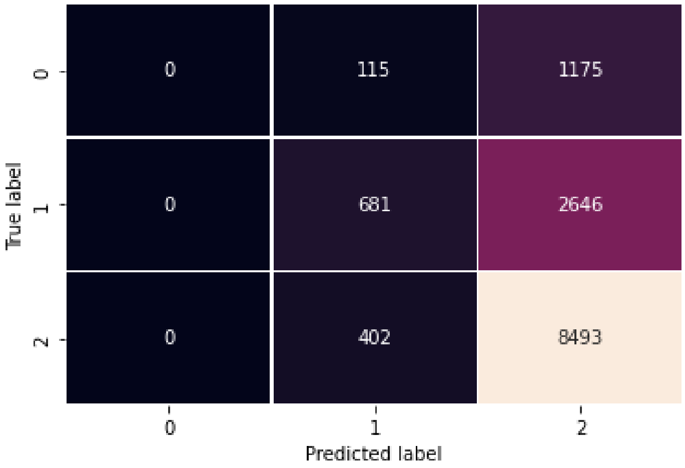
```
train(4, percentage_test = 20, isDepth = True)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.57      0.20      0.30      3327
           2       0.69      0.95      0.80      8895
```

```
    accuracy                           0.68     13512
   macro avg       0.42      0.39      0.37     13512
weighted avg       0.59      0.68      0.60     13512
```

Decision Tree Classifier confusion matrix



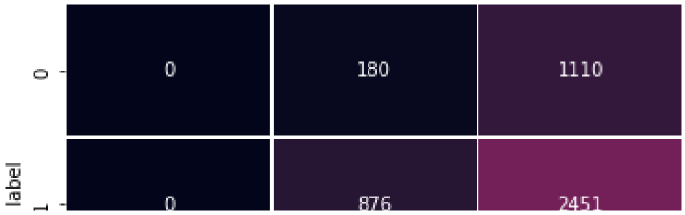
In [ ]:

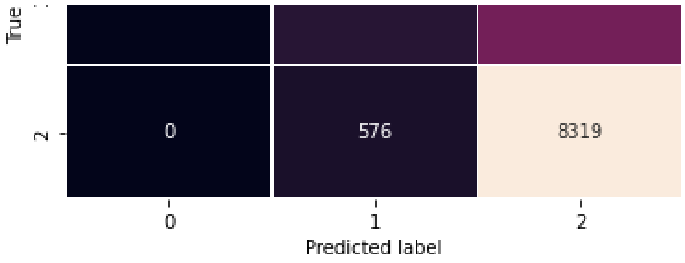
```
train(5, percentage_test = 20, isDepth = True)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.54      0.26      0.35      3327
           2       0.70      0.94      0.80      8895

    accuracy                           0.68     13512
   macro avg       0.41      0.40      0.38     13512
weighted avg       0.59      0.68      0.61     13512
```

Decision Tree Classifier confusion matrix





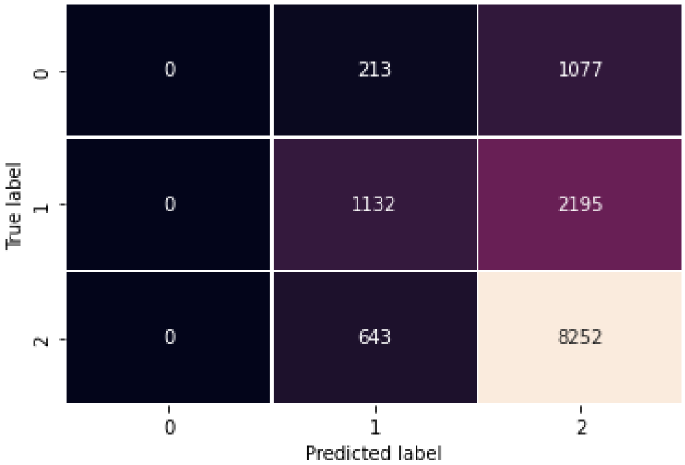
In [ ]:

```
train(6, percentage_test = 20, isDepth = True)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
               precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.57      0.34      0.43      3327
           2       0.72      0.93      0.81      8895

    accuracy                           0.69     13512
   macro avg       0.43      0.42      0.41     13512
weighted avg       0.61      0.69      0.64     13512
```

Decision Tree Classifier confusion matrix

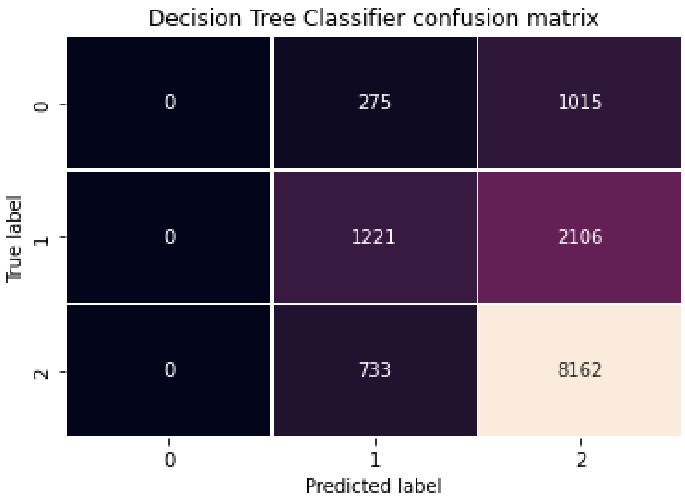


In [ ]:

```
train(7, percentage_test = 20, isDepth = True)
```

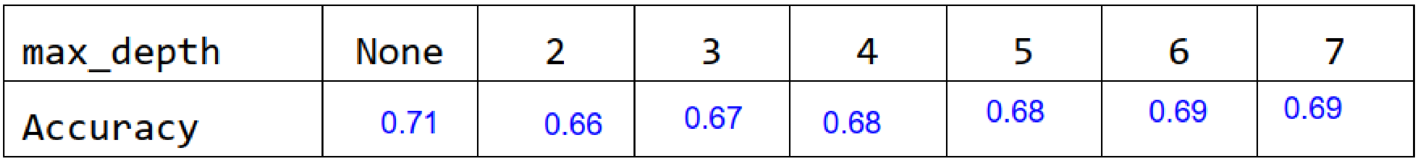
```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1308: Undefined
MetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
```

```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1290
           1       0.55      0.37      0.44      3327
           2       0.72      0.92      0.81      8895

    accuracy                           0.69     13512
   macro avg       0.42      0.43      0.42     13512
weighted avg       0.61      0.69      0.64     13512
```



In [ ]:

```
# You do not need to run this command since I use it to extract files to zip and download
it to local
# !zip -r /content/max_depth_7.zip /content/output_saved/max_depth_7
# from google.colab import files
# files.download("/content/max_depth_7.zip")
```



max_depth	None	2	3	4	5	6	7
Accuracy	0.71	0.66	0.67	0.68	0.68	0.69	0.69

By specifying depth_max = None, our decision tree will explore until all leaves are pure. In other words, all attributes of dataset will be included as node in decision tree. This becomes a complete tree (often lead to

overfitting)

So, if we set depth_max = N (N is not None), that means the decision tree can end up its finding at a depth <= N. In this case, decision tree might not be able to go through every attributes of dataset. Theoritically, it's not good because we want decision tree can interprete more patterns of dataset (this tree is called shallow tree) and underfitting happens. But in practical, expanding the tree to its marginality is not always right thing to do, because many attributes are helpless and are not likely to contribute to result. A complex tree might lead to overfitting as it results in poor generalization performance.

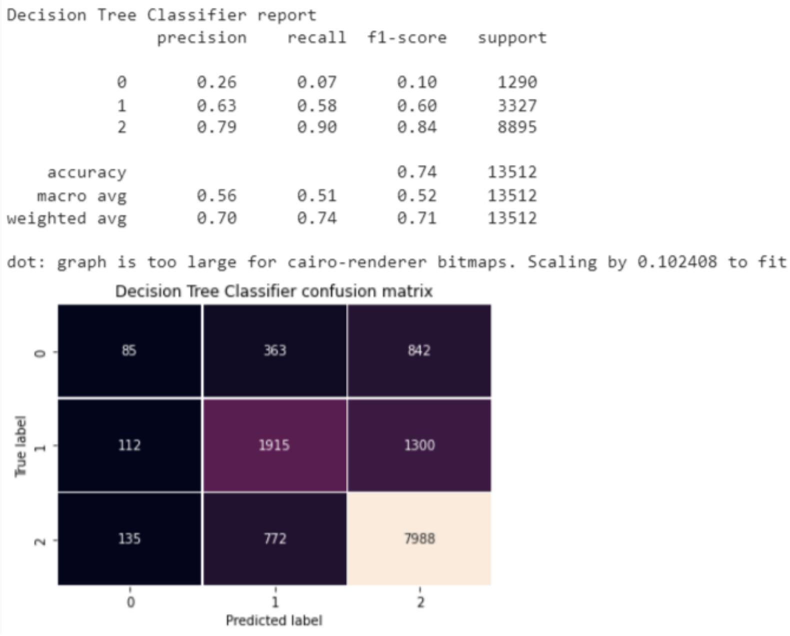
So selecting a appropriate value for N is critical.

In this experiment, since I pick 22 attributes randomly, so I have no idea if decision tree performs well as expected or not.

From above N = 2,3,4,5,6,7 obvisously makes model underfiting. No predictions are 0. The one of reasons that leads to this problem is the imbalance in training set

To obtain the best max depth is iterating through incremental number, starting from 2, until prediction's accuracy has tendency to overfit I have tried this method and get the max depth that yields the highest accuracy is max_depth = 12. When max_depth > 12, model's accuracy decreases a bit and then is stable at 0.71 (can try with max_depth = 100, 150, etc.)

max_depth = 12, train_test = 80 : 20:



```
Decision Tree Classifier report
              precision    recall  f1-score   support

           0       0.26      0.07      0.10      1290
           1       0.63      0.58      0.60      3327
           2       0.79      0.90      0.84      8895

    accuracy                           0.74     13512
   macro avg       0.56      0.51      0.52     13512
weighted avg       0.70      0.74      0.71     13512

dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.102408 to fit
```