In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sys
import os
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

#Feature Scaling (Scaling using minmaxscaler)
from sklearn.preprocessing import MinMaxScaler

#Regression libraries used in this kernel
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import GradientBoostingRegressor

#Metrics for regression model performance
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
```

In [2]:

```
df = pd.read_csv('D:/Users/Janhavi/TE/TE1/ML/Admission_Predict_Ver1.1.csv')

print(df.columns)   # print column name and it's type
print(df.shape)   # print the number of rows and columns as a tuple
#df['Chance of Admit ']
```

```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SO
P',
       'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
      dtype='object')
(500, 9)
```

In [3]:

```
sl = df[df['Chance of Admit ']>0.80]
sl.count()
```

Out[3]:

```
Serial No.           142
GRE Score            142
TOEFL Score          142
University Rating    142
SOP                  142
LOR                  142
CGPA                 142
Research             142
Chance of Admit      142
dtype: int64
```

In [4]:

```
df.loc[df['Chance of Admit ']>=0.80, 'Chance of Admit ']=1
df.loc[df['Chance of Admit ']<0.80, 'Chance of Admit ']=0
```

In [5]:

```
df['Chance of Admit ']
```

Out[5]:

```
0      1.0
1      0.0
2      0.0
3      1.0
4      0.0
      ...
495    1.0
496    1.0
497    1.0
498    0.0
499    1.0
Name: Chance of Admit , Length: 500, dtype: float64
```

In [6]:

```
X= df.drop(['Chance of Admit ', 'Serial No.'],axis=1)
y= df['Chance of Admit ']
```

In [7]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=123)
```

In [8]:

```
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

In [9]:

```
matrix = confusion_matrix(y_test,y_pred, labels=[0.0,1.0])
print('Confusion matrix ',matrix)

accuracy1=accuracy_score(y_test,y_pred)
print('Accuracy11111111111111' , accuracy1)
```

```
Confusion matrix  [[62  3]
 [ 3 32]]
Accuracy11111111111111 0.94
```

In [10]:

```
from sklearn.metrics import classification_report
cr = classification_report(y_test,y_pred)

print(cr)
```

```
              precision    recall  f1-score   support

         0.0       0.95      0.95      0.95        65
         1.0       0.91      0.91      0.91        35

    accuracy                           0.94       100
   macro avg       0.93      0.93      0.93       100
weighted avg       0.94      0.94      0.94       100
```

In [11]:

```
feature_names = df.columns[:10]
print(feature_names)
target_names =  df['Chance of Admit '].unique().tolist()
```

```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SO
P',
       'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
      dtype='object')
```

In [12]:

```
class_names=[str(x) for x in model.classes_]
print(model.classes_)
print(str(class_names))
```

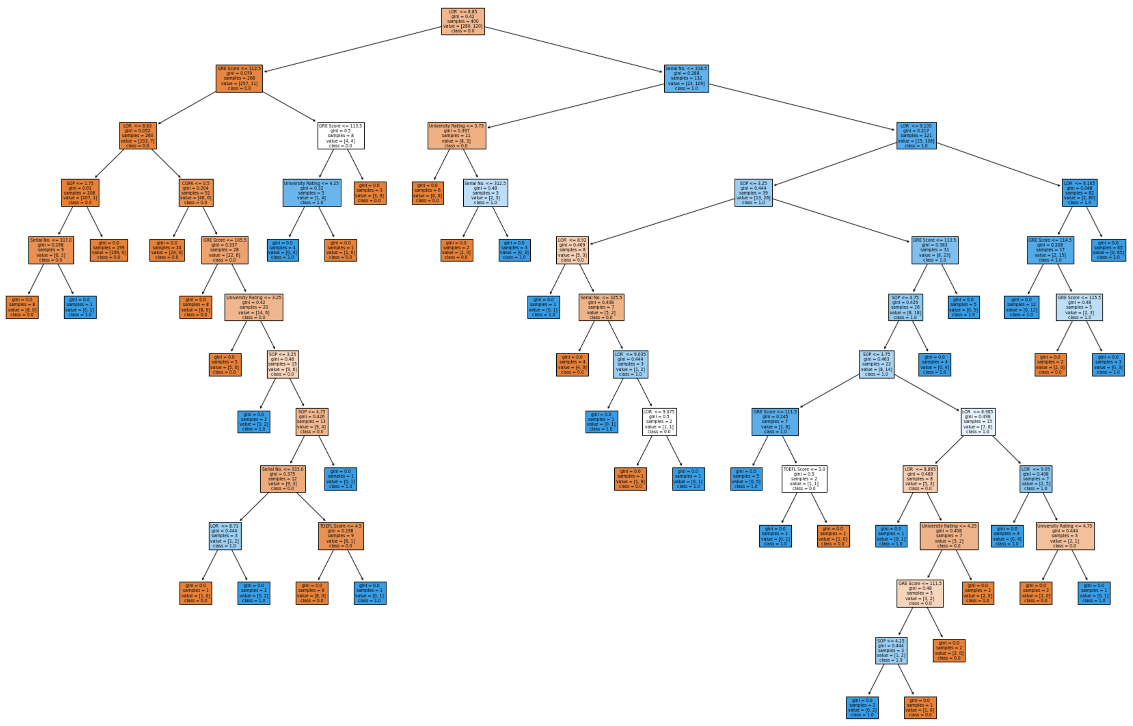
```
[0. 1.]
['0.0', '1.0']
```

In [13]:

```
from sklearn.tree import plot_tree
fig = plt.figure(figsize=(30,20))

plot_tree(model,feature_names=feature_names,class_names=class_names,filled=True)  #,rounded=True)

plt.savefig('tree_visualization.png')
```



In [14]:

```
clf = DecisionTreeClassifier(criterion="entropy",max_depth=3)

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",accuracy_score(y_test,y_pred))
```

```
Accuracy: 0.95
```