In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.tree import plot_tree
from sklearn.tree import plot_tree
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_1612\4235877999.py:2: DeprecationWarning:
Pyarrow will become a required dependency of pandas in the next major release of pandas (pandas 3.0),
(to allow more performant data types, such as the Arrow string type, and better interoperability with other libraries)
but was not found to be installed on your system.
If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/54466

  import pandas as pd
```

In [2]:

```
 import pandas as pd
```

In [5]:

```
df = pd.read_csv('Diabetes - Diabetes.csv')
```

In [6]:

```
X = df.drop('Outcome', axis=1)
y = df['Outcome']
```

In [7]:

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3,random_state=0)
print("No error")
```

```
No error
```

In [8]:

```
tree_model = DecisionTreeClassifier()
tree_model = tree_model.fit(X_train,y_train)
tree_model
```

Out[8]:

```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

In [9]:

```
y_pred = tree_model.predict(X_test)
y_pred
```

Out[9]:

```
array([1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0], dtype=int64)
```

In [10]:

```
print('Accuracy:', metrics.accuracy_score(y_test,y_pred))
```

```
Accuracy: 0.7489177489177489
```

In [11]:

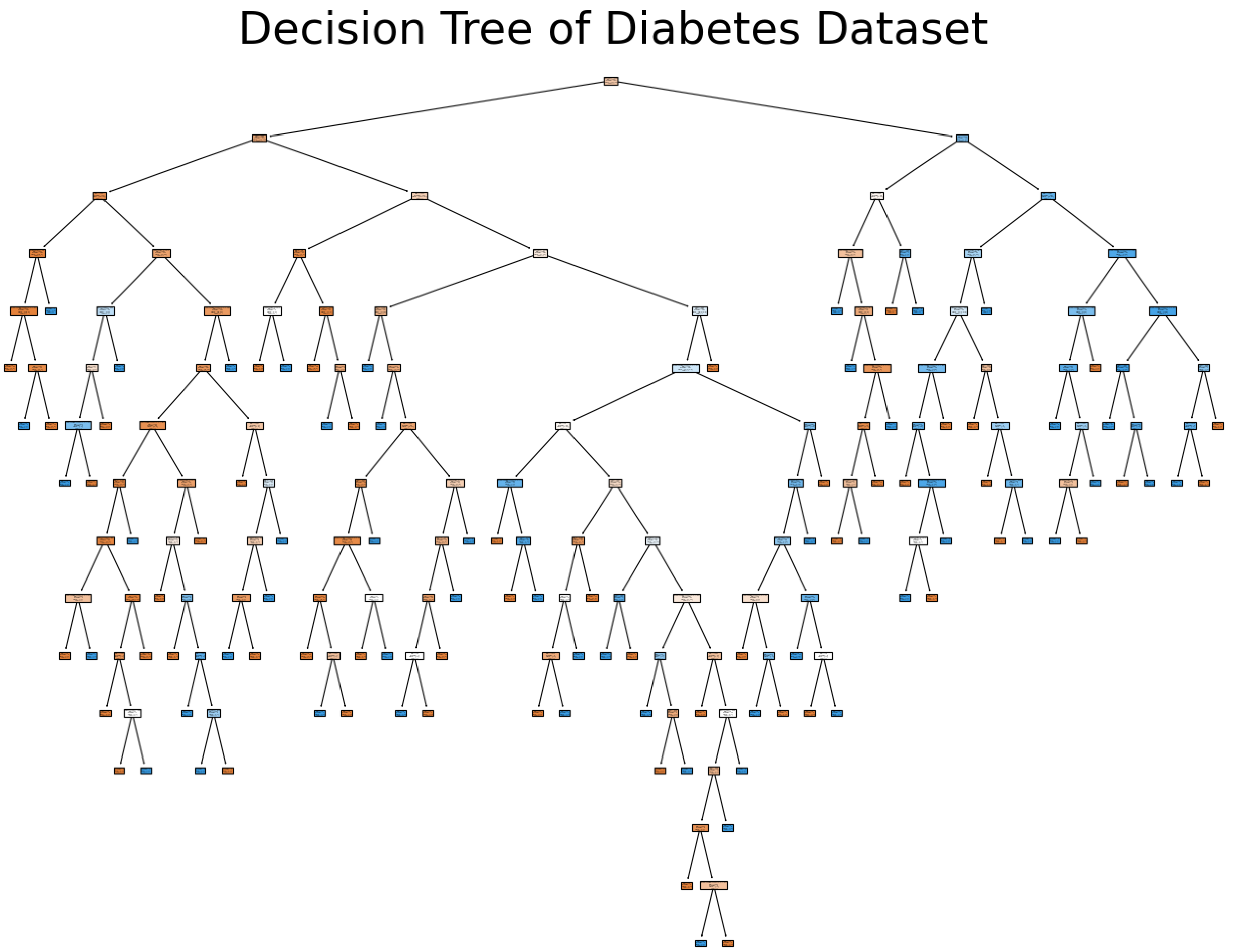
```
features=X.columns
features
```

Out[11]:

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age'],
      dtype='object')
```

In [12]:

```
plt.figure(figsize=(20,15),dpi= 80)
class_labels = ['Negative', 'Positive']
plot_tree(tree_model, filled=True, feature_names=list(features), class_names=['0', '1'])
plt.title("Decision Tree of Diabetes Dataset",fontsize=40)
plt.show()
```



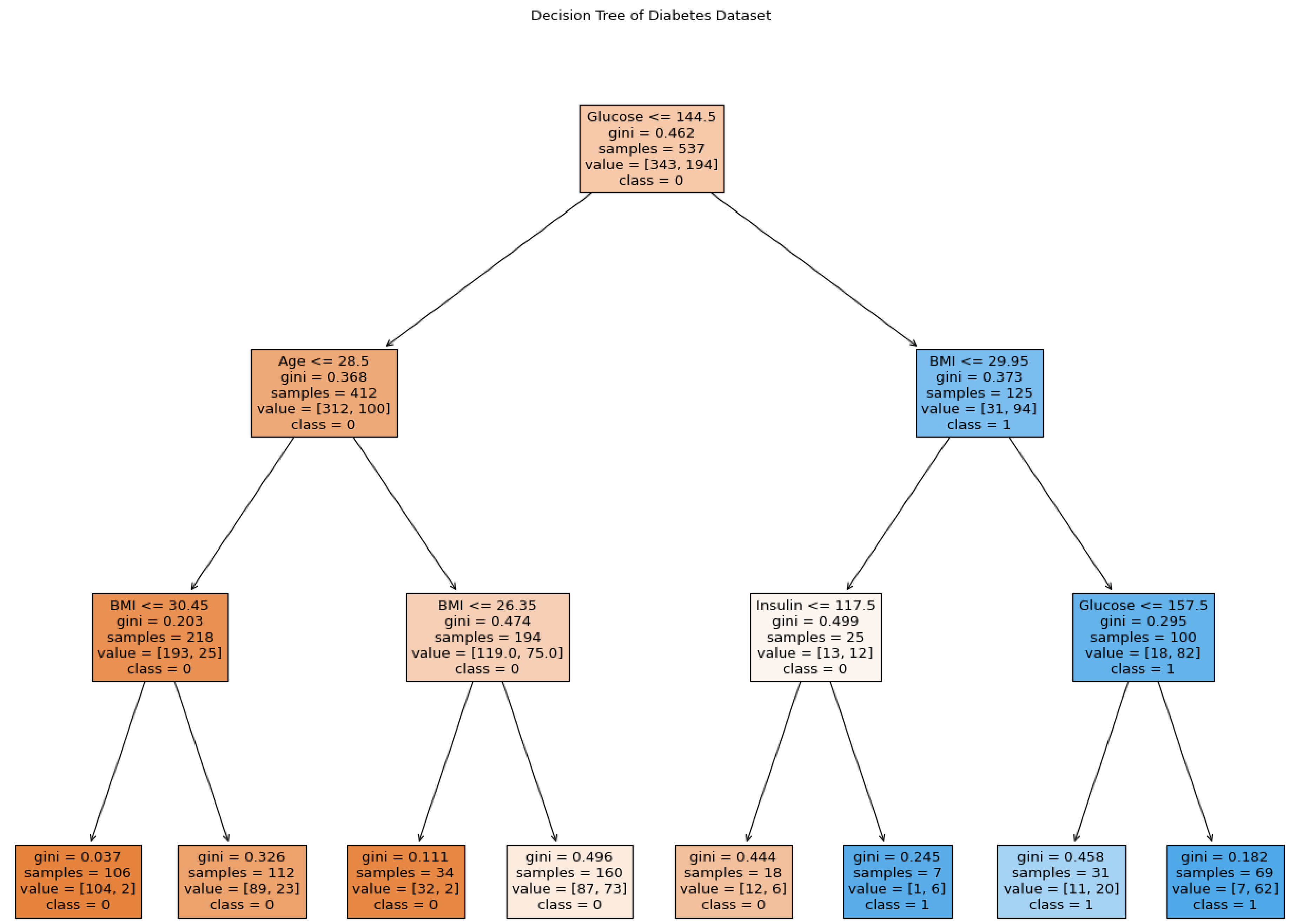
```
In [13]: tree_model1 = DecisionTreeClassifier(max_depth=3)
         tree_model1 = tree_model1.fit(X_train,y_train)
         tree_model1
```

```
Out[13]: ▾ DecisionTreeClassifier
         DecisionTreeClassifier(max_depth=3)
```

```
In [14]: y_pred = tree_model.predict(X_test)
         y_pred
```

```
Out[14]: array([1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
                0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1,
                1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1,
                0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
                0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
                0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0,
                0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
                0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0], dtype=int64)
```

```
In [15]: plt.figure(figsize=(20,15),dpi= 80)
         class_labels = ['Negative', 'Positive']
         plot_tree(tree_model1, filled=True, feature_names=list(features), class_names=['0', '1'])
         plt.title("Decision Tree of Diabetes Dataset")
         plt.show()
```



Decision Tree of Diabetes Dataset

In [16]:

```
cm = confusion_matrix(y_test,y_pred)
print(cm)
```

```
[[124  33]
 [ 25  49]]
```

In [17]:

```
print(classification_report(y_test , y_pred))
```

```
              precision    recall  f1-score   support

           0       0.83      0.79      0.81       157
           1       0.60      0.66      0.63        74

    accuracy                           0.75       231
   macro avg       0.71      0.73      0.72       231
weighted avg       0.76      0.75      0.75       231
```

In [18]:

```
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print("Accuracy: {}\nPrecision: {}\nRecall: {}\nF1-Score: {}".format(acc,prec,rec,f1))
```

```
Accuracy: 0.7489177489177489
Precision: 0.5975609756097561
Recall: 0.6621621621621622
F1-Score: 0.6282051282051282
```

In [19]:

```
from sklearn.model_selection import GridSearchCV
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
}
grid = GridSearchCV(tree_model, param_grid, cv=10)
grid.fit(X, y)
grid
```

Out[19]:

```
▸          GridSearchCV
▸ estimator: DecisionTreeClassifier
    ▸ DecisionTreeClassifier
```

In [20]:

```
print(grid.best_params_)
print(grid.best_estimator_)
```

```
{'criterion': 'gini', 'splitter': 'best'}
DecisionTreeClassifier()
```