```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,
classification_report,confusion_matrix

url
="https://raw.githubusercontent.com/plotly/datasets/master/diabetes.cs
v"
df = pd.read_csv(url)
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 768,\n  \"fields\": [\
n    {\n      \"column\": \"Pregnancies\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 3,\n        \"min\": 0,\n
\"max\": 17,\n        \"num_unique_values\": 17,\n        \"samples\":
[\n          6,\n          1,\n          3\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Glucose\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 31,\n
\"min\": 0,\n        \"max\": 199,\n        \"num_unique_values\":
136,\n        \"samples\": [\n          151,\n          101,\n
112\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BloodPressure\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 19,\n        \"min\": 0,\n
\"max\": 122,\n        \"num_unique_values\": 47,\n
\"samples\": [\n          86,\n          46,\n          85\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"SkinThickness\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 15,\n        \"min\": 0,\n
\"max\": 99,\n        \"num_unique_values\": 51,\n        \"samples\":
[\n          7,\n          12,\n          48\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Insulin\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 115,\n
\"min\": 0,\n        \"max\": 846,\n        \"num_unique_values\":
186,\n        \"samples\": [\n          52,\n          41,\n
183\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BMI\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 7.8841603203754405,\n        \"min\": 0.0,\n        \"max\":
67.1,\n        \"num_unique_values\": 248,\n        \"samples\": [\n
19.9,\n          31.0,\n          38.1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
```

```
n    },\n    {\n      \"column\": \"DiabetesPedigreeFunction\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.33132859501277484,\n        \"min\": 0.078,\n        \"max\": 2.42,\
n        \"num_unique_values\": 517,\n        \"samples\": [\n
1.731,\n          0.426,\n          0.138\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 11,\n        \"min\": 21,\n
\"max\": 81,\n        \"num_unique_values\": 52,\n        \"samples\":
[\n          60,\n          47,\n          72\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Outcome\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 0,\n        \"max\": 1,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          0,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


```
df.info()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
X = df.drop("Outcome", axis=1)
y = df["Outcome"]
X_train, X_test, y_train, y_test = train_test_split(
 X, y, test_size=0.2, random_state=42
)
rf_model = RandomForestClassifier(
    n_estimators=100,
    criterion='entropy',  # <--- Uses Information Gain/Entropy
    random_state=42
)
rf_model.fit(X_train, y_train)
RandomForestClassifier(random_state=42)
y_pred = rf_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n",
classification_report(y_test,y_pred))
```

```
Accuracy: 0.7207792207792207

Classification Report:
               precision    recall  f1-score   support

           0       0.79      0.78      0.78        99
           1       0.61      0.62      0.61        55

    accuracy                           0.72       154
   macro avg       0.70      0.70      0.70       154
weighted avg       0.72      0.72      0.72       154
```

```
last_tree = rf_model.estimators_[-1]
print(last_tree)
```

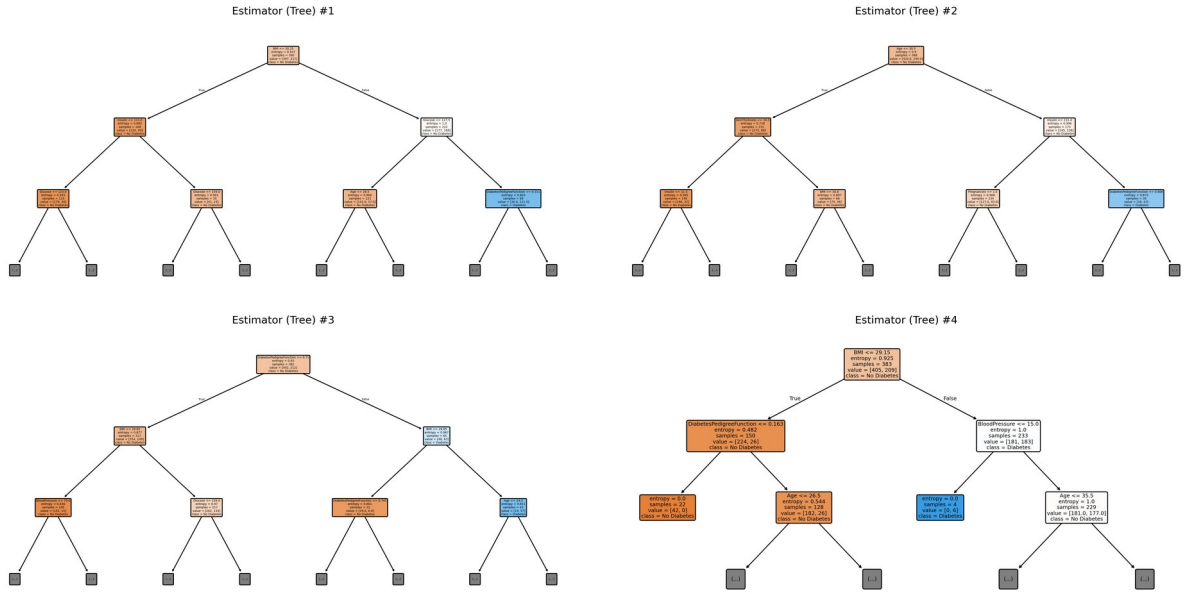
```
DecisionTreeClassifier(criterion='entropy', max_features='sqrt',
                       random_state=134489564)
```

```
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(20, 10), dpi=300)

for index in range(0, 4):
    row = index // 2
    col = index % 2
    plot_tree(
        rf_model.estimators_[index],
        feature_names=X.columns,
        class_names=["No Diabetes", "Diabetes"],
        filled=True,
        rounded=True,
        max_depth=2, # Limits depth to keep the diagram readable
        ax=axes[row, col]
    )
    axes[row, col].set_title(f"Estimator (Tree) #{index+1}",
```

```
fontsize=12)

plt.tight_layout()
plt.show()
```




```
import pickle
# Save trained model
with open("rf.pkl", "wb") as file:
 pickle.dump(rf_model, file)
from google.colab import files
files.download('rf.pkl')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.Javascript object>
```