```
from google.colab import files
upload=files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving diabetes.csv to diabetes.csv

```
import pandas as pd
df=pd.read_csv("diabetes.csv")
df
```

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFunction | Age | Outcome |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 50 | 1 |
| **1** | 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 31 | 0 |
| **2** | 8 | 183 | 64 | 0 | 0 | 23.3 | 0.672 | 32 | 1 |
| **3** | 1 | 89 | 66 | 23 | 94 | 28.1 | 0.167 | 21 | 0 |
| **4** | 0 | 137 | 40 | 35 | 168 | 43.1 | 2.288 | 33 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **763** | 10 | 101 | 76 | 48 | 180 | 32.9 | 0.171 | 63 | 0 |
| **764** | 2 | 122 | 70 | 27 | 0 | 36.8 | 0.340 | 27 | 0 |
| **765** | 5 | 121 | 72 | 23 | 112 | 26.2 | 0.245 | 30 | 0 |
| **766** | 1 | 126 | 60 | 0 | 0 | 30.1 | 0.349 | 47 | 1 |
| **767** | 1 | 93 | 70 | 31 | 0 | 30.4 | 0.315 | 23 | 0 |

768 rows × 9 columns

```
df.shape
```

```
(768, 9)
```

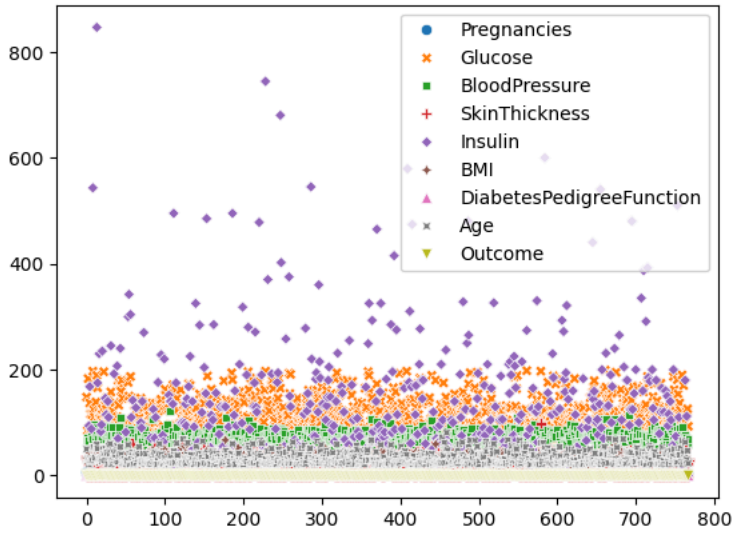
```
df.isnull().sum()
```

| | 0 |
|---|---|
| **Pregnancies** | 0 |
| **Glucose** | 0 |
| **BloodPressure** | 0 |
| **SkinThickness** | 0 |
| **Insulin** | 0 |
| **BMI** | 0 |
| **DiabetesPedigreeFunction** | 0 |
| **Age** | 0 |
| **Outcome** | 0 |

**dtype:** int64

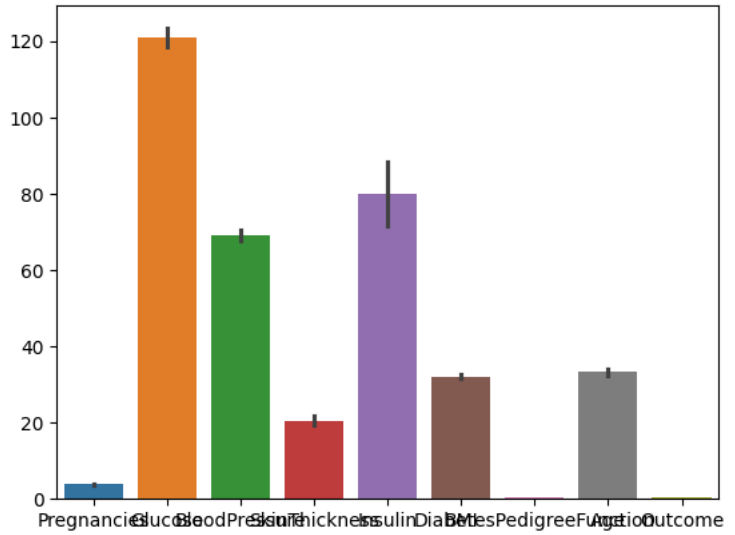
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(df)
```

<Axes: >



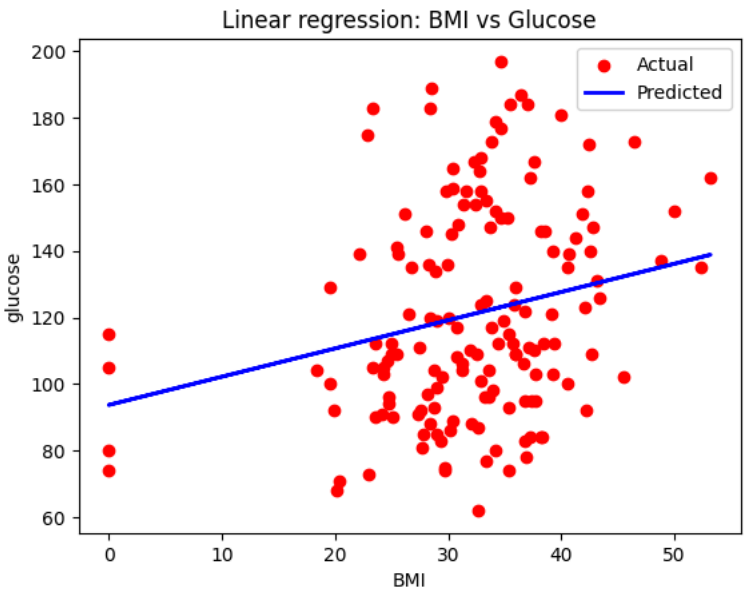

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.barplot(df)
```

<Axes: >




```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Load the diabetes dataset (replace 'diabetes.csv' with the actual path if needed)
df = pd.read_csv('diabetes.csv')

# Assuming 'BMI' is a column in your dataset and you want to predict 'Glucose'
X = df[['BMI']]  # Features (BMI in this case)
y = df['Glucose']  # Target variable (Glucose)

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  # Adjust test_size and random_state as needed

# Create and train a Linear Regression model
model = LinearRegression()
model.fit(x_train, y_train)

# Make predictions on the test set
y_pred = model.predict(x_test)

# Now you can plot the results
plt.scatter(x_test, y_test, color='red', label='Actual')
plt.plot(x_test, y_pred, color='blue', linewidth=2, label='Predicted')
plt.xlabel('BMI')
plt.ylabel('glucose')
plt.title('Linear regression: BMI vs Glucose')
plt.legend()
```

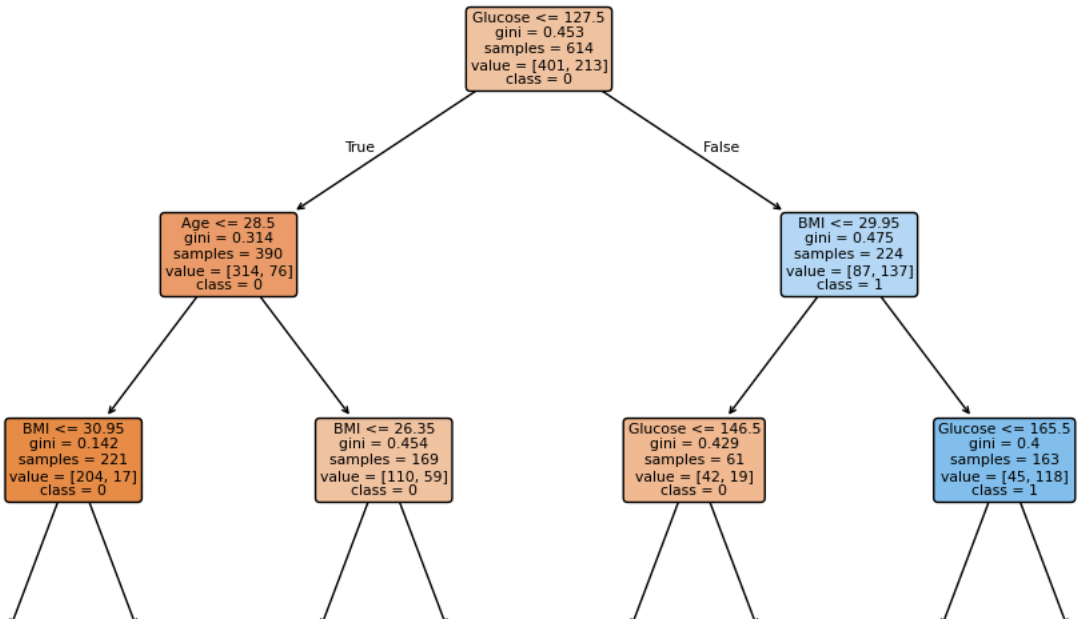
```
plt.show()
```




```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# 1. Load your dataset:
# Assuming your dataset is in a CSV file named 'data.csv'
data = pd.read_csv('diabetes.csv')

# 2. Prepare the data:
# Separate features (X) and target variable (y)
X = data.drop('Outcome', axis=1)  # Features (all columns except 'Outcome')
y = data['Outcome']  # Target variable ('Outcome' column)

# 3. Split data into training and testing sets:
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 4. Create and train the Decision Tree Classifier:
clf = DecisionTreeClassifier(max_depth=3)
clf.fit(X_train, y_train)

# 5. Make predictions on the test set:
y_pred = clf.predict(X_test)

# 6. Evaluate the model:
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.7597402597402597
```

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt

# ... (rest of your code as before) ...

# After training the model (clf.fit)
plt.figure(figsize=(12, 8))  # Adjust figure size as needed
plot_tree(clf, feature_names=X.columns, class_names=['0', '1'], filled=True, rounded=True)
plt.show()
```


Glucose <= 127.5
gini = 0.453
samples = 614
value = [401, 213]
class = 0
True
False
Age <= 28.5
gini = 0.314
samples = 390
value = [314, 76]
class = 0
BMI <= 29.95
gini = 0.475
samples = 224
value = [87, 137]
class = 1
BMI <= 30.95
gini = 0.142
samples = 221
value = [204, 17]
class = 0
BMI <= 26.35
gini = 0.454
samples = 169
value = [110, 59]
class = 0
Glucose <= 146.5
gini = 0.429
samples = 61
value = [42, 19]
class = 0
Glucose <= 165.5
gini = 0.4
samples = 163
value = [45, 118]
class = 1