DIABETES PREDICTION

Importing Libraries and Loading Dataset

In [39]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/diabetes_prediction_dataset.csv')
df
```

Out[39]:

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
0	Female	80.0	0	1	never	25.19	6.6	140	0
1	Female	54.0	0	0	No Info	27.32	6.6	80	0
2	Male	28.0	0	0	never	27.32	5.7	158	0
3	Female	36.0	0	0	current	23.45	5.0	155	0
4	Male	76.0	1	1	current	20.14	4.8	155	0
...	...	...	...	...	...	...	...	...	...
99995	Female	80.0	0	0	No Info	27.32	6.2	90	0
99996	Female	2.0	0	0	No Info	17.37	6.5	100	0
99997	Male	66.0	0	0	former	27.83	5.7	155	0
99998	Female	24.0	0	0	never	35.42	4.0	100	0
99999	Female	57.0	0	0	current	22.43	6.6	90	0

100000 rows × 9 columns

In [40]:
```
df.head()
```

Out[40]:

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
0	Female	80.0	0	1	never	25.19	6.6	140	0
1	Female	54.0	0	0	No Info	27.32	6.6	80	0
2	Male	28.0	0	0	never	27.32	5.7	158	0
3	Female	36.0	0	0	current	23.45	5.0	155	0
4	Male	76.0	1	1	current	20.14	4.8	155	0

In [41]:
```
df.tail()
```

Out[41]:

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
99995	Female	80.0	0	0	No Info	27.32	6.2	90	0
99996	Female	2.0	0	0	No Info	17.37	6.5	100	0
99997	Male	66.0	0	0	former	27.83	5.7	155	0
99998	Female	24.0	0	0	never	35.42	4.0	100	0
99999	Female	57.0	0	0	current	22.43	6.6	90	0

In [42]:
```
df.isna().sum()
```

Out[42]:

	0
gender	0
age	0
hypertension	0
heart_disease	0
smoking_history	0
bmi	0
HbA1c_level	0
blood_glucose_level	0
diabetes	0

dtype: int64

In [43]:
```
df.dtypes
```

Out[43]:

	0
gender	object
age	float64
hypertension	int64
heart_disease	int64
smoking_history	object
bmi	float64
HbA1c_level	float64
blood_glucose_level	int64
diabetes	int64

dtype: object

In [44]:
```
df['diabetes'].value_counts()
```

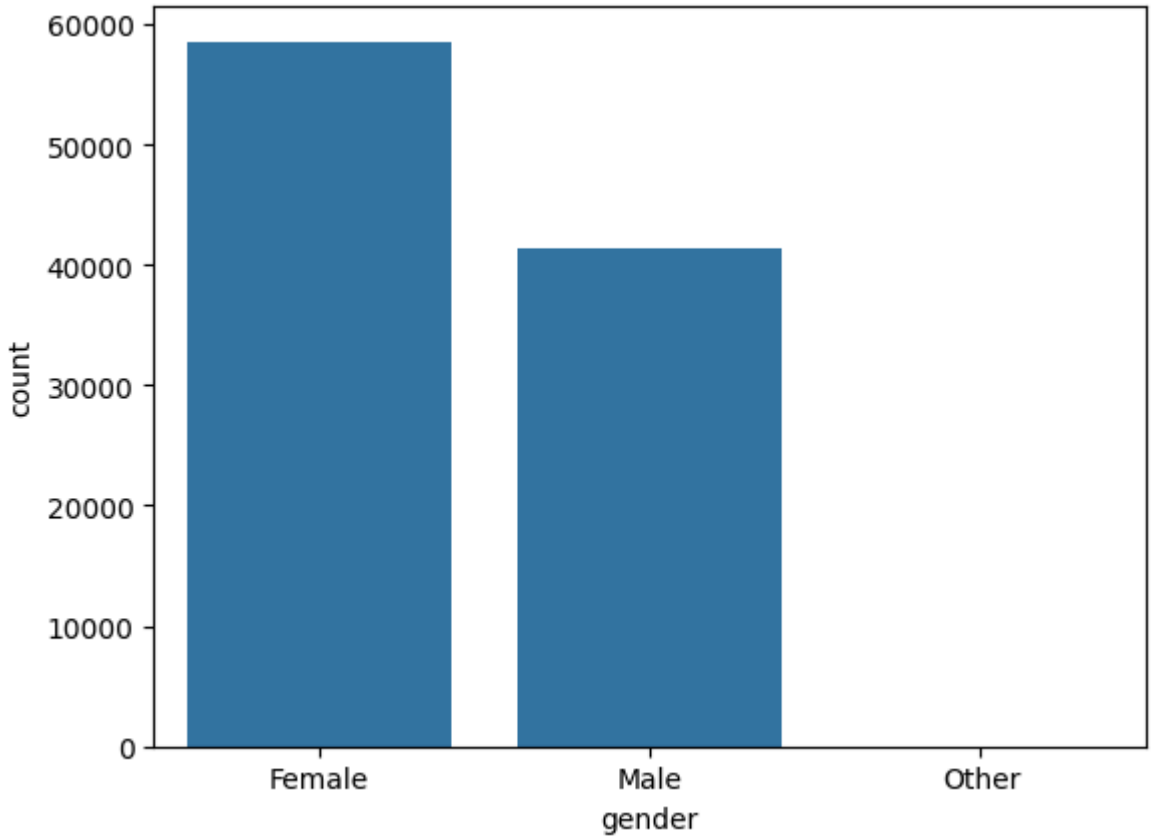
Out[44]:

diabetes	count
0	91500
1	8500

dtype: int64

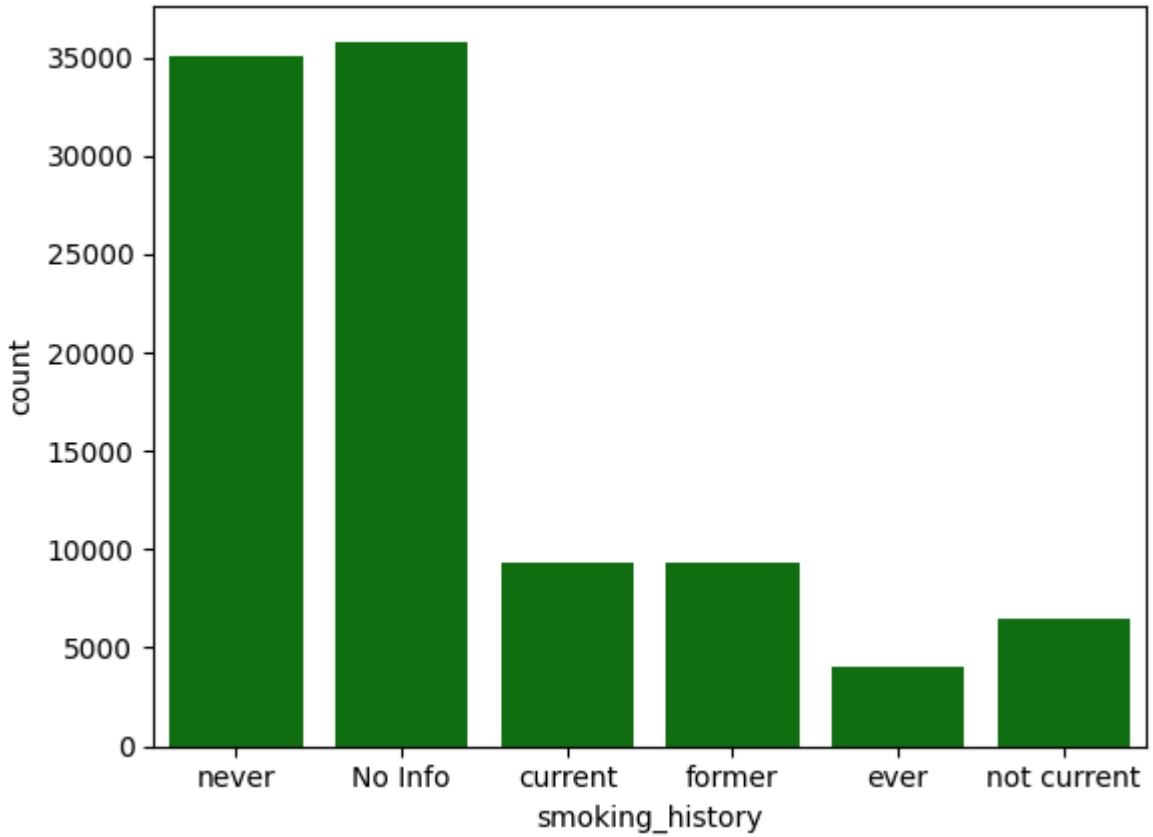
In [45]:
```
sns.countplot(x='gender',data=df)
```

Out[45]:
```
<Axes: xlabel='gender', ylabel='count'>
```



In [46]:
```
sns.countplot(x='smoking_history',data=df,color='g')
```

Out[46]:
```
<Axes: xlabel='smoking_history', ylabel='count'>
```



Data Preprocessing

In [47]:
```
from sklearn.preprocessing import LabelEncoder
lab=LabelEncoder()
for column in df.select_dtypes(include='object'):
    df[column]=lab.fit_transform(df[column])
```

In [48]:
```
df.dtypes
```

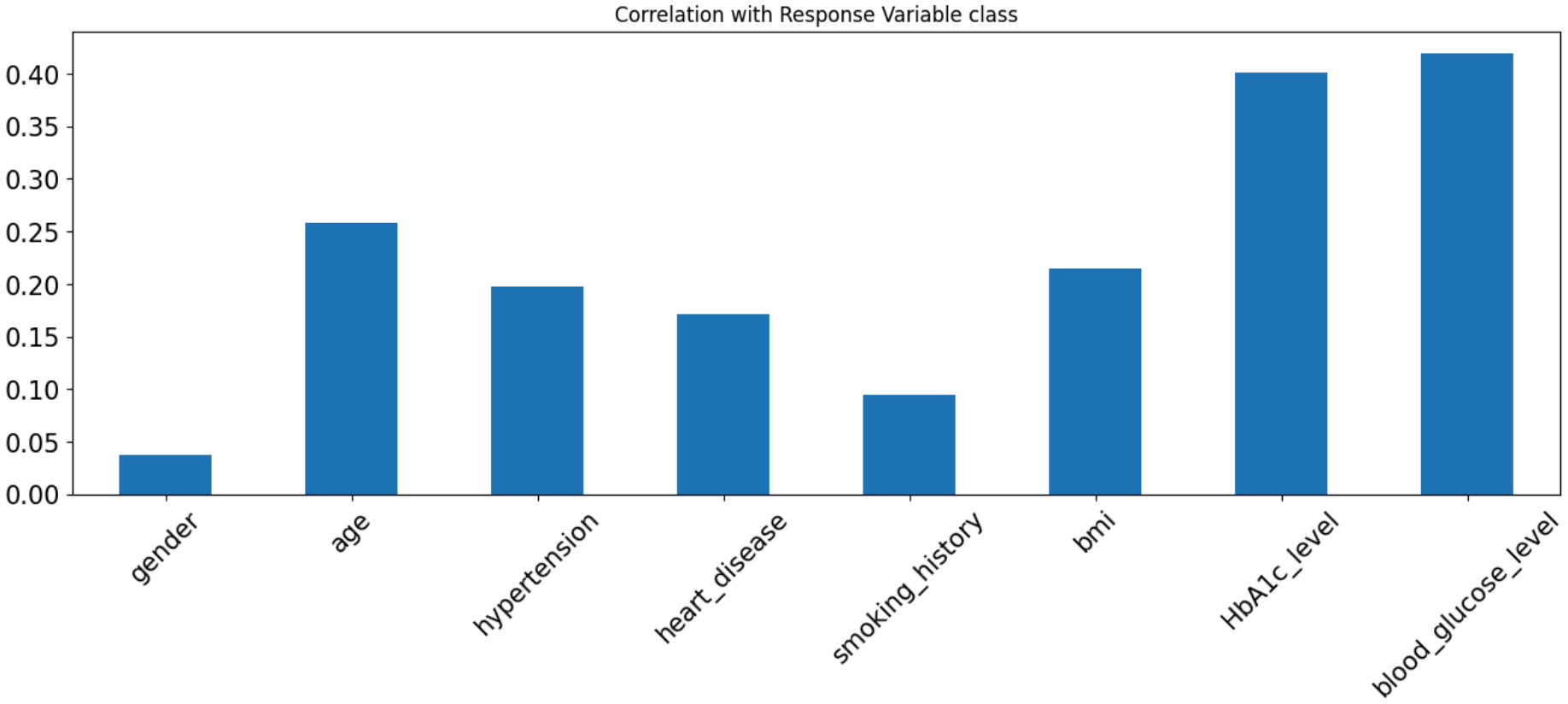
Out[48]:

	0
gender	int64
age	float64
hypertension	int64
heart_disease	int64
smoking_history	int64
bmi	float64
HbA1c_level	float64
blood_glucose_level	int64
diabetes	int64

dtype: object

In [49]:

```
df1 = df.copy()
 #Correlation with Response Variable class
X = df1.drop(['diabetes'],axis=1)
y = df1['diabetes']
X.corrwith(y).plot.bar(
figsize = (16, 5), title = "Correlation with Response Variable class", fontsize = 15,
rot = 45, grid = False)
plt.show()
```


Correlation with Response Variable class
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
gender
age
hypertension
heart_disease
smoking_history
bmi
HbA1c_level
blood_glucose_level

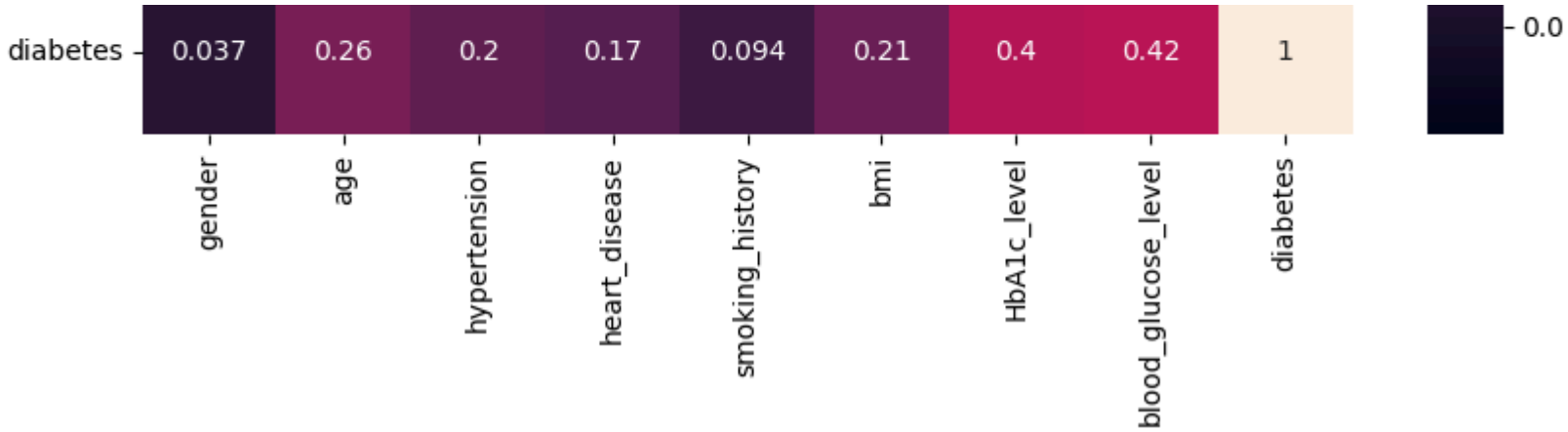
In [50]:
```
df.corr()
```

Out[50]:

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
gender	1.000000	-0.030656	0.014203	0.077696	-0.077919	-0.022994	0.019957	0.017199	0.037411
age	-0.030656	1.000000	0.251171	0.233354	0.228608	0.337396	0.101354	0.110672	0.258008
hypertension	0.014203	0.251171	1.000000	0.121262	0.093177	0.147666	0.080939	0.084429	0.197823
heart_disease	0.077696	0.233354	0.121262	1.000000	0.027598	0.061198	0.067589	0.070066	0.171727
smoking_history	-0.077919	0.228608	0.093177	0.027598	1.000000	0.179361	0.037369	0.040219	0.094290
bmi	-0.022994	0.337396	0.147666	0.061198	0.179361	1.000000	0.082997	0.091261	0.214357
HbA1c_level	0.019957	0.101354	0.080939	0.067589	0.037369	0.082997	1.000000	0.166733	0.400660
blood_glucose_level	0.017199	0.110672	0.084429	0.070066	0.040219	0.091261	0.166733	1.000000	0.419558
diabetes	0.037411	0.258008	0.197823	0.171727	0.094290	0.214357	0.400660	0.419558	1.000000

In [51]:
```
plt.figure(figsize=(10,10))
sns.heatmap(df1.corr(),annot=True)
plt.show()
```


gender
age
hypertension
heart_disease
smoking_history
bmi
HbA1c_level
blood_glucose_level
1 -0.031 0.014 0.078 -0.078 -0.023 0.02 0.017 0.037
-0.031 1 0.25 0.23 0.23 0.34 0.1 0.11 0.26
0.014 0.25 1 0.12 0.093 0.15 0.081 0.084 0.2
0.078 0.23 0.12 1 0.028 0.061 0.068 0.07 0.17
-0.078 0.23 0.093 0.028 1 0.18 0.037 0.04 0.094
-0.023 0.34 0.15 0.061 0.18 1 0.083 0.091 0.21
0.02 0.1 0.081 0.068 0.037 0.083 1 0.17 0.4
0.017 0.11 0.084 0.07 0.04 0.091 0.17 1 0.42
1.0
0.8
0.6
0.4
0.2



In [52]:
```
x=df.iloc[:,:-1].values
x
```

Out[52]:
```
array([[  0.  ,  80.  ,   0.  , ...,  25.19,   6.6 , 140.  ],
       [  0.  ,  54.  ,   0.  , ...,  27.32,   6.6 ,  80.  ],
       [  1.  ,  28.  ,   0.  , ...,  27.32,   5.7 , 158.  ],
       ...,
       [  1.  ,  66.  ,   0.  , ...,  27.83,   5.7 , 155.  ],
       [  0.  ,  24.  ,   0.  , ...,  35.42,   4.  , 100.  ],
       [  0.  ,  57.  ,   0.  , ...,  22.43,   6.6 ,  90.  ]])
```

In [53]:
```
y=df.iloc[:,-1].values
y
```

Out[53]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [54]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=0)
x_train
```

Out[54]:
```
array([[  0.  ,  80.  ,   0.  , ...,  27.32,   4.8 , 159.  ],
       [  0.  ,   0.72,   0.  , ...,  16.02,   5.8 ,  90.  ],
       [  0.  ,  32.  ,   0.  , ...,  27.28,   6.6 , 159.  ],
       ...,
       [  1.  ,  66.  ,   0.  , ...,  41.23,   9.  , 145.  ],
       [  0.  ,  37.  ,   0.  , ...,  30.18,   5.8 ,  90.  ],
       [  0.  ,  52.  ,   0.  , ...,  27.32,   4.5 , 158.  ]])
```

In [55]:
```
x_test
```

Out[55]:
```
array([[  0.  ,  52.  ,   0.  , ...,  27.32,   4.8 , 140.  ],
       [  1.  ,  56.  ,   0.  , ...,  27.32,   4.8 , 100.  ],
       [  0.  ,  22.  ,   0.  , ...,  37.16,   6.6 ,  85.  ],
       ...,
       [  1.  ,  26.  ,   0.  , ...,  27.32,   6.2 , 145.  ],
       [  0.  ,  46.  ,   0.  , ...,  25.58,   5.7 , 200.  ],
       [  0.  ,  20.  ,   0.  , ...,  21.68,   5.7 , 155.  ]])
```

In [56]:
```
y_train
```

Out[56]:
```
array([0, 0, 0, ..., 1, 0, 0])
```

In [57]:
```
y_test
```

Out[57]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

Model Creation and Performance Evaluation

In [58]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(x_train)
x_train=scaler.transform(x_train)
x_test=scaler.transform(x_test)
```

In [59]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,classification_report
knn=KNeighborsClassifier(n_neighbors=7)
dectree=DecisionTreeClassifier(random_state=42)
ranfor=RandomForestClassifier(n_estimators=100,random_state=42)
lst=[knn,dectree,ranfor]
```

In [60]:
```
for i in lst:
 i.fit(x_train,y_train)
 y_pred=i.predict(x_test)
 print("R2_score of",i,"model is",accuracy_score(y_test,y_pred))
 print(classification_report(y_test,y_pred))
```

```
R2_score of KNeighborsClassifier(n_neighbors=7) model is 0.9612666666666667
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     27461
           1       0.90      0.61      0.73      2539

    accuracy                           0.96     30000
   macro avg       0.93      0.80      0.85     30000
weighted avg       0.96      0.96      0.96     30000

R2_score of DecisionTreeClassifier(random_state=42) model is 0.9491
              precision    recall  f1-score   support

           0       0.98      0.97      0.97     27461
           1       0.69      0.73      0.71      2539

    accuracy                           0.95     30000
   macro avg       0.83      0.85      0.84     30000
weighted avg       0.95      0.95      0.95     30000

R2_score of RandomForestClassifier(random_state=42) model is 0.9699333333333333
              precision    recall  f1-score   support

           0       0.97      1.00      0.98     27461
           1       0.95      0.68      0.79      2539

    accuracy                           0.97     30000
   macro avg       0.96      0.84      0.89     30000
weighted avg       0.97      0.97      0.97     30000
```