```
import pandas as pd
file_path = 'diabetes.csv'
read_file = pd.read_csv(file_path)
read_file.columns
read_file
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  
\
0              6      148             72             35        0  33.6

1              1       85             66             29        0  26.6

2              8      183             64              0        0  23.3

3              1       89             66             23       94  28.1

4              0      137             40             35      168  43.1

..           ...      ...            ...            ...      ...   ...

763           10      101             76             48      180  32.9

764            2      122             70             27        0  36.8

765            5      121             72             23      112  26.2

766            1      126             60              0        0  30.1

767            1       93             70             31        0  30.4


     DiabetesPedigreeFunction  Age  Outcome  
0                       0.627   50        1  
1                       0.351   31        0  
2                       0.672   32        1  
3                       0.167   21        0  
4                       2.288   33        1  
..                        ...  ...      ...  
763                     0.171   63        0  
764                     0.340   27        0  
765                     0.245   30        0  
766                     0.349   47        1  
767                     0.315   23        0  

[768 rows x 9 columns]
```

```
read_file.describe()
```

```
       Pregnancies     Glucose  BloodPressure  SkinThickness  
Insulin  \
count   768.000000  768.000000     768.000000     768.000000
768.000000
mean      3.845052  120.894531      69.105469      20.536458
79.799479
std       3.369578   31.972618      19.355807      15.952218
115.244002
min       0.000000    0.000000       0.000000       0.000000
0.000000
25%       1.000000   99.000000      62.000000       0.000000
0.000000
50%       3.000000  117.000000      72.000000      23.000000
30.500000
75%       6.000000  140.250000      80.000000      32.000000
127.250000
max      17.000000  199.000000     122.000000      99.000000
846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```

```
y=read_file.Outcome
y  #"""1-  true
      #0- false"""
```

```
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

```
features = ['Glucose','BloodPressure','Insulin','BMI','Age']
x =read_file[features]
x.head()
```

```
   Glucose  BloodPressure  Insulin   BMI  Age
0      148             72        0  33.6   50
```

```
1       85             66        0  26.6   31
2      183             64        0  23.3   32
3       89             66       94  28.1   21
4      137             40      168  43.1   33
```

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(random_state=1)#define model with random state
model.fit(x, y)#fit
```

```
DecisionTreeRegressor(random_state=1)
```

```
print('Make predcitions for diabetes:- ')
print(x.head())
print('The predictions are:- ')
print(model.predict(x.head()))
```

```
Make predcitions for diabetes:- 
   Glucose  BloodPressure  Insulin   BMI  Age
0      148             72        0  33.6   50
1       85             66        0  26.6   31
2      183             64        0  23.3   32
3       89             66       94  28.1   21
4      137             40      168  43.1   33
The predictions are:- 
[1. 0. 1. 0. 1.]
```

```
from sklearn.metrics import mean_absolute_error

predicted_diabetes = model.predict(x)
mean_absolute_error(y, predicted_diabetes)
```

```
0.0
```

```
from sklearn.model_selection import train_test_split

# split data into training and validation data, for both features and target
# The split is based on a random number generator. Supplying a numeric value to
# the random_state argument guarantees we get the same split every time we
# run this script.
train_x, val_x, train_y, val_y = train_test_split(x, y, random_state = 0)
model = DecisionTreeRegressor()
# Fit model
model.fit(train_x, train_y)

# get predicted prices on validation data
```

```
val_predictions = model.predict(val_x)
print(mean_absolute_error(val_y, val_predictions))
```

```
0.21875
```

```
from sklearn.metrics import mean_absolute_error
from sklearn.tree import DecisionTreeRegressor

def get_mae(max_leaf_nodes, train_x, val_x, train_y, val_y):
    model = DecisionTreeRegressor(max_leaf_nodes=max_leaf_nodes,
random_state=0)
    model.fit(train_x, train_y)
    preds_val = model.predict(val_x)
    mae = mean_absolute_error(val_y, preds_val)
    return(mae)

# compare MAE with differing values of max_leaf_nodes
for max_leaf_nodes in [100, 200, 300, 400, 500]:
    my_mae = get_mae(max_leaf_nodes, train_x, val_x, train_y, val_y)
    print("Max leaf nodes: %d  \t\t Mean Absolute Error:  %d" %
(max_leaf_nodes, my_mae))
```

```
Max leaf nodes: 100         Mean Absolute Error:  0
Max leaf nodes: 200         Mean Absolute Error:  0
Max leaf nodes: 300         Mean Absolute Error:  0
Max leaf nodes: 400         Mean Absolute Error:  0
Max leaf nodes: 500         Mean Absolute Error:  0
```