```
import numpy as np
import pandas as pd

df = pd.read_csv('/content/diabetes.csv')


null_columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'Age', 'Outcome']
df[null_columns] = df[null_columns].replace({0:np.nan})

df.to_csv('/content/diabetes.csv', index=False)

df.head(20)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeF
0	6.0	148.0	72.0	35.0	NaN	33.6	
1	1.0	85.0	66.0	29.0	NaN	26.6	
2	8.0	183.0	64.0	NaN	NaN	23.3	
3	1.0	89.0	66.0	23.0	94.0	28.1	
4	NaN	137.0	40.0	35.0	168.0	43.1	
5	5.0	116.0	74.0	NaN	NaN	25.6	
6	3.0	78.0	50.0	32.0	88.0	31.0	
7	10.0	115.0	NaN	NaN	NaN	35.3	
8	2.0	197.0	70.0	45.0	543.0	30.5	
9	8.0	125.0	96.0	NaN	NaN	NaN	
10	4.0	110.0	92.0	NaN	NaN	37.6	
11	10.0	168.0	74.0	NaN	NaN	38.0	
12	10.0	139.0	80.0	NaN	NaN	27.1	
13	1.0	189.0	60.0	23.0	846.0	30.1	
14	5.0	166.0	72.0	19.0	175.0	25.8	
15	7.0	100.0	NaN	NaN	NaN	30.0	
16	NaN	118.0	84.0	47.0	230.0	45.8	
17	7.0	107.0	74.0	NaN	NaN	29.6	
18	1.0	103.0	30.0	38.0	83.0	43.3	
19	1.0	115.0	70.0	30.0	96.0	34.6	

```
import numpy as np
import pandas as pd

df = pd.read_csv('/content/diabetes.csv')
null_columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'Age', 'Outcome']
df[null_columns] = df[null_columns].replace({0:np.nan})

df.fillna(df.mean(), inplace=True)
df = df.round(2)

df.to_csv('/content/diabetes.csv', index=False)

df.head(20)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigree
0	6.00	148.0	72.00	35.00	155.55	33.60	
1	1.00	85.0	66.00	29.00	155.55	26.60	
2	8.00	183.0	64.00	29.15	155.55	23.30	
3	1.00	89.0	66.00	23.00	94.00	28.10	
4	4.49	137.0	40.00	35.00	168.00	43.10	
5	5.00	116.0	74.00	29.15	155.55	25.60	
6	3.00	78.0	50.00	32.00	88.00	31.00	
7	10.00	115.0	72.41	29.15	155.55	35.30	
8	2.00	197.0	70.00	45.00	543.00	30.50	
9	8.00	125.0	96.00	29.15	155.55	32.46	
10	4.00	110.0	92.00	29.15	155.55	37.60	
11	10.00	168.0	74.00	29.15	155.55	38.00	
12	10.00	139.0	80.00	29.15	155.55	27.10	
13	1.00	189.0	60.00	23.00	846.00	30.10	
14	5.00	166.0	72.00	19.00	175.00	25.80	
15	7.00	100.0	72.41	29.15	155.55	30.00	
16	4.49	118.0	84.00	47.00	230.00	45.80	
17	7.00	107.0	74.00	29.15	155.55	29.60	
18	1.00	103.0	30.00	38.00	83.00	43.30	
19	1.00	115.0	70.00	30.00	96.00	34.60	

```
from sklearn.datasets import load_diabetes
from sklearn.model_selection import KFold
from sklearn.preprocessing import StandardScaler
import numpy as np
import pandas as pd

diabetes = load_diabetes()

X_diabetes = diabetes.data
y_diabetes = diabetes.target

scaler = StandardScaler()
X_scaler = scaler.fit_transform(X_diabetes)

cnt = 0
n_splits = 10
kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)
for train_index, test_index in kf.split(X_scaler, y_diabetes):
    print(f'Fold:{cnt}, Train set: {len(train_index)}, \
    Test set:{len(test_index)}')
```

```
Fold:0, Train set: 397,     Test set:45
Fold:0, Train set: 397,     Test set:45
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
Fold:0, Train set: 398,     Test set:44
```

```
from sklearn.datasets import load_diabetes
from sklearn.model_selection import KFold, cross_val_score
from sklearn.tree import DecisionTreeRegressor
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
import math
```

```
diabetes = load_diabetes()

X = diabetes.data
y = diabetes.target

scaler = StandardScaler()
X_scaler = scaler.fit_transform(X)

cnt = 0
n_splits = 10
kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)
for train_index, test_index in kf.split(X_scaler, y):
    print(f'Fold:{cnt}, Train set: {len(train_index)}, \
    Test set:{len(test_index)}')
    cnt += 1

def rmse(mse):
    return math.sqrt(abs(mse))

score = cross_val_score(DecisionTreeRegressor(random_state= 42), X, y, cv=kf, scoring="neg_mean_squared_error")
print(f'Scores for each fold: {score}')
rmse(score.mean())
```

```
Fold:0, Train set: 397,     Test set:45
Fold:1, Train set: 397,     Test set:45
Fold:2, Train set: 398,     Test set:44
Fold:3, Train set: 398,     Test set:44
Fold:4, Train set: 398,     Test set:44
Fold:5, Train set: 398,     Test set:44
Fold:6, Train set: 398,     Test set:44
Fold:7, Train set: 398,     Test set:44
Fold:8, Train set: 398,     Test set:44
Fold:9, Train set: 398,     Test set:44
Scores for each fold: [-5343.88888889 -7760.44444444 -6265.11363636 -9223.81818182
 -8044.81818182 -7328.65909091 -6032.70454545 -7414.09090909
 -6354.90909091 -6602.40909091]
83.88733877088131
```