```
import pandas as pd
data=pd.read_csv('diabetes.csv')
data.head()
```

```
   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


   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
```

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
class SLP:
    def __init__(self, learning_rate=0.01, epochs=1000):
        self.learning_rate=learning_rate
        self.epochs=epochs
        self.weights=None

    def sigmoid(self, x):
        return 1 / (1 + np.exp(-x))

    def fit(self, x, y):
        x=np.c_[np.ones(x.shape[0]), x]  #bias
        self.weights=np.zeros(x.shape[1])
        for epoch in range(self.epochs):
            output=np.dot(x, self.weights)
            y_pred=self.sigmoid(output)
            error=y-y_pred
            update=self.learning_rate*error*y_pred*(1 - y_pred)
            self.weights+=np.dot(x.T, update)

    def predict(self, x):
        x=np.c_[np.ones(x.shape[0]), x]  #bias
        output=np.dot(x, self.weights)
        return np.round(self.sigmoid(output))
```

```
data = pd.read_csv('diabetes.csv')
data=data.iloc[:50]
X=np.array(data.iloc[:, :-1].values)
y=np.array(data.iloc[:, -1].values)
scaler=StandardScaler()
X=scaler.fit_transform(X)
perceptron=SLP()
perceptron.fit(X, y)
predictions = perceptron.predict(X)
print("Predicted:", predictions)
print("Actual:", y)
print("\n")
print("mean square error (loss):",(np.square(y-predictions)).mean())
print("cost (overall):", ((y-predictions)**2).sum())
print("accuracy:",accuracy_score(y,predictions))
```

```
Predicted: [1. 0. 1. 0. 1. 0. 0. 0. 1. 0. 0. 1. 0. 1. 1. 0. 1. 0. 1.
1. 1. 0. 1. 1.
 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 1. 1. 1. 1. 0. 0. 1. 0. 1. 0.
0.
 1. 0.]
Actual: [1 0 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 0 0 0
1 0 0 0 0 0
 1 1 1 0 0 0 1 0 1 0 0 1 0]


mean square error (loss): 0.18
cost (overall): 9.0
accuracy: 0.82
```