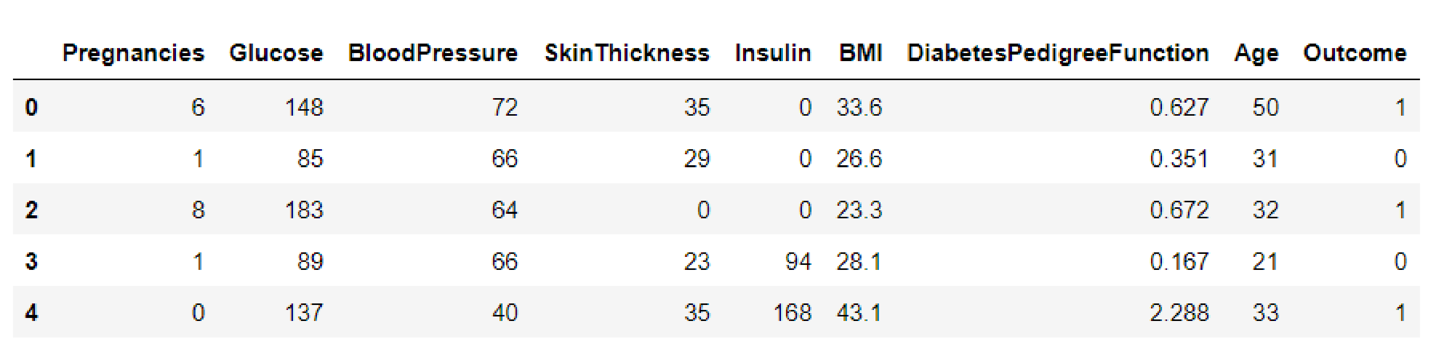
Predict Diabetes

PRESENTING TEACHER NOPADON

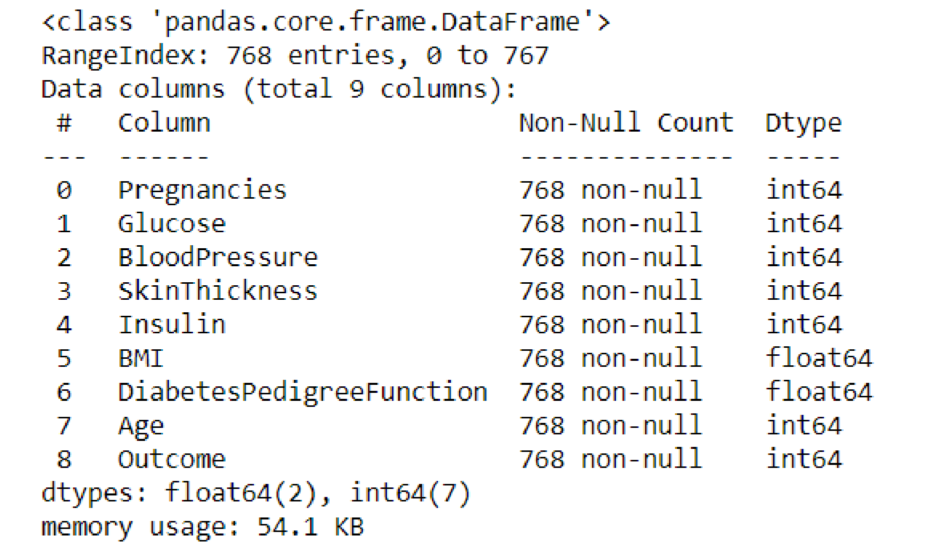
Public Data

Predict Diabetes Dataset

Ex. Dataset

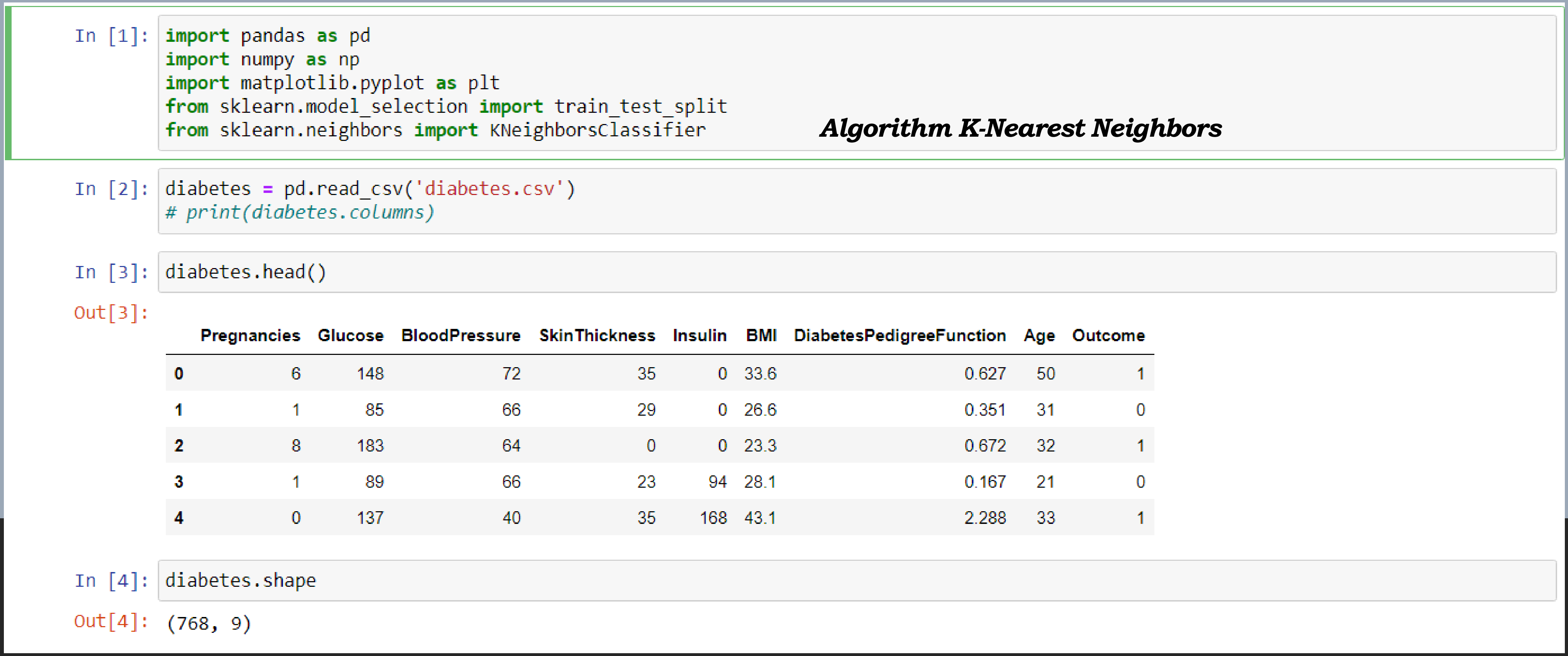


	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Coding



Algorithm K-Nearest Neighbors

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn.model_selection import train_test_split
        from sklearn.neighbors import KNeighborsClassifier
```

```
In [2]: diabetes = pd.read_csv('diabetes.csv')
        # print(diabetes.columns)
```

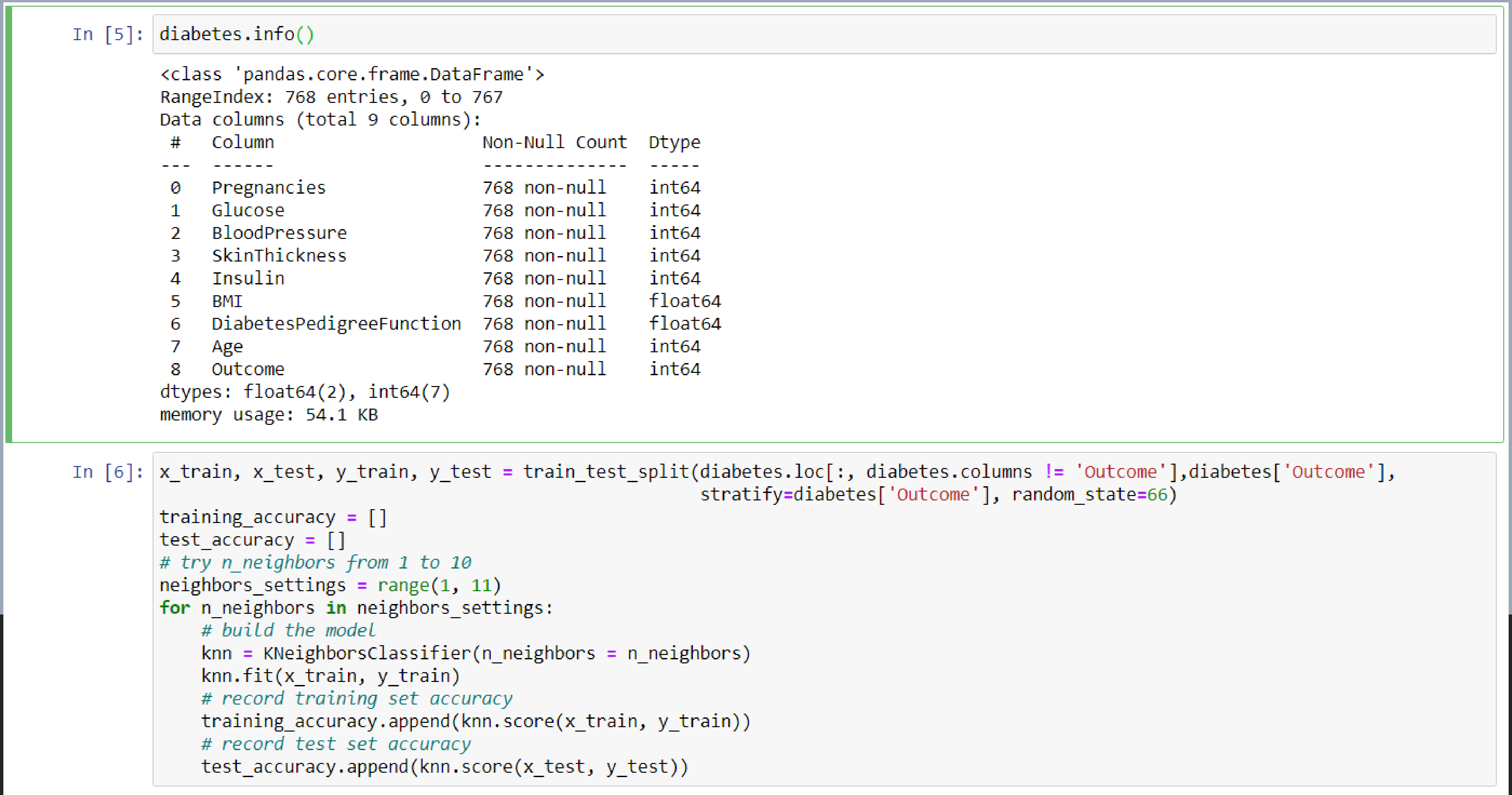
```
In [3]: diabetes.head()
```

Out[3]:

	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

```
In [4]: diabetes.shape

Out[4]: (768, 9)
```

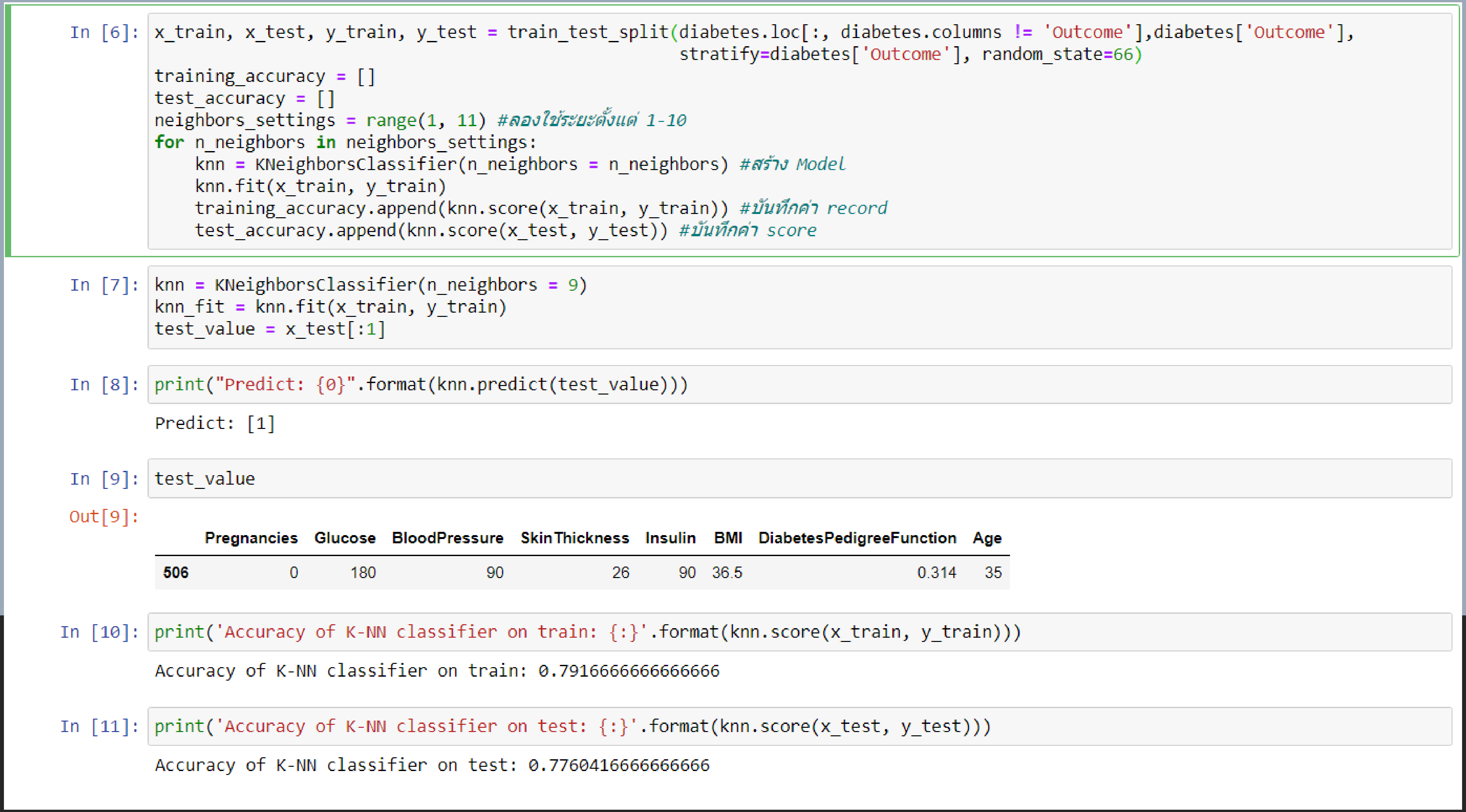


In [5]:
```
diabetes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [6]:
```
x_train, x_test, y_train, y_test = train_test_split(diabetes.loc[:, diabetes.columns != 'Outcome'],diabetes['Outcome'],
                                                    stratify=diabetes['Outcome'], random_state=66)

training_accuracy = []
test_accuracy = []
# try n_neighbors from 1 to 10
neighbors_settings = range(1, 11)
for n_neighbors in neighbors_settings:
    # build the model
    knn = KNeighborsClassifier(n_neighbors = n_neighbors)
    knn.fit(x_train, y_train)
    # record training set accuracy
    training_accuracy.append(knn.score(x_train, y_train))
    # record test set accuracy
    test_accuracy.append(knn.score(x_test, y_test))
```



In [6]:
```python
x_train, x_test, y_train, y_test = train_test_split(diabetes.loc[:, diabetes.columns != 'Outcome'],diabetes['Outcome'],
                                                    stratify=diabetes['Outcome'], random_state=66)
training_accuracy = []
test_accuracy = []
neighbors_settings = range(1, 11) #ลองใช้ระยะตั้งแต่ 1-10
for n_neighbors in neighbors_settings:
    knn = KNeighborsClassifier(n_neighbors = n_neighbors) #สร้าง Model
    knn.fit(x_train, y_train)
    training_accuracy.append(knn.score(x_train, y_train)) #บันทึกค่า record
    test_accuracy.append(knn.score(x_test, y_test)) #บันทึกค่า score
```

In [7]:
```python
knn = KNeighborsClassifier(n_neighbors = 9)
knn_fit = knn.fit(x_train, y_train)
test_value = x_test[:1]
```

In [8]:
```python
print("Predict: {0}".format(knn.predict(test_value)))
```

```
Predict: [1]
```

In [9]:
```python
test_value
```

Out[9]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
506	0	180	90	26	90	36.5	0.314	35

In [10]:
```python
print('Accuracy of K-NN classifier on train: {:}'.format(knn.score(x_train, y_train)))
```

```
Accuracy of K-NN classifier on train: 0.7916666666666666
```

In [11]:
```python
print('Accuracy of K-NN classifier on test: {:}'.format(knn.score(x_test, y_test)))
```

```
Accuracy of K-NN classifier on test: 0.7760416666666666
```