

# Implement K-Nearest Neighbors algorithm on diabete

In [1]:

```
import pandas as pd
import numpy as np
from sklearn import metrics
```

In [2]:

```
df = pd.read_csv("diabetes.csv")
```

In [3]:

```
df
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	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
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

In [4]:

```
df.shape
```

Out[4]:

```
(768, 9)
```

In [5]:

```
# checking for null values
df.isnull().any().value_counts()
```

Out[5]:

```
False    9
dtype: int64
```

In [6]:

```
df.columns
```

Out[6]:

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'Pedigree', 'Age', 'Outcome'],
      dtype='object')
```

In [7]:

```
df_x = df.drop(columns='Outcome', axis=1)
df_y = df['Outcome']
```

In [8]:

```
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
scaledX = scale.fit_transform(df_x)
```

In [9]:

```
# split into train and test
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(scaledX, df_y, test_size=0.2, random_state=42)
```

In [10]:

```
# KNN
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(x_train, y_train)
y_pred = knn.predict(x_test)
```

In [11]:

```
# Confusion matrix
cs = metrics.confusion_matrix(y_test, y_pred)
print("Confusion matrix: \n", cs)
```

```
Confusion matrix:
[[78 21]
 [28 27]]
```

In [12]:

```
# Accuracy score
ac = metrics.accuracy_score(y_test, y_pred)
print("Accuracy score: ", ac)
```

```
Accuracy score: 0.6818181818181818
```

In [13]:

```
# Error rate (error_rate = 1- accuracy)
er = 1-ac
print("Error rate: ", er)
```

```
Error rate: 0.31818181818181823
```

In [14]:

```
# Precision
p = metrics.precision_score(y_test, y_pred)
print("Precision: ", p)
```

```
Precision: 0.5625
```

In [15]:

```
# Recall
r = metrics.recall_score(y_test, y_pred)
print("Recall: ", r)
```

```
Recall: 0.4909090909090909
```

In [16]:

```
# Classification report
cr = metrics.classification_report(y_test,y_pred)
print("Classification report: \n\n", cr)Implement K-Nearest Neighbors algorithm on diabete
```

Classification report:

	precision	recall	f1-score	support
0	0.74	0.79	0.76	99
1	0.56	0.49	0.52	55
accuracy			0.68	154
macro avg	0.65	0.64	0.64	154
weighted avg	0.67	0.68	0.68	154