In [2]:

```
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import numpy as np
```

In [4]:

```
#1. Loading and Preprocessing
#Loading dataset
data = load_breast_cancer()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target)
```

In [17]:

```
#Preprocessing steps
# Check for missing values
print(X.isnull().sum())
```

```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
dtype: int64
```

In [8]:

```
# Feature Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size
```

In [9]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Logistic Regression
log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)
y_pred_log_reg = log_reg.predict(X_test)
accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)
print(f'Logistic Regression Accuracy: {accuracy_log_reg:.2f}')
```

```
Logistic Regression Accuracy: 0.97
```

In [11]:

```
from sklearn.tree import DecisionTreeClassifier

# Decision Tree Classifier
dt_classifier = DecisionTreeClassifier()
dt_classifier.fit(X_train, y_train)
y_pred_dt = dt_classifier.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print(f'Decision Tree Accuracy: {accuracy_dt:.2f}')
```

```
Decision Tree Accuracy: 0.93
```

In [12]:

```
from sklearn.ensemble import RandomForestClassifier

# Random Forest Classifier
rf_classifier = RandomForestClassifier()
rf_classifier.fit(X_train, y_train)
y_pred_rf = rf_classifier.predict(X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f'Random Forest Accuracy: {accuracy_rf:.2f}')
```

```
Random Forest Accuracy: 0.96
```

In [13]:

```
from sklearn.svm import SVC

# Support Vector Machine Classifier
svm_classifier = SVC()
svm_classifier.fit(X_train, y_train)
y_pred_svm = svm_classifier.predict(X_test)
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print(f'SVM Accuracy: {accuracy_svm:.2f}')
```

```
SVM Accuracy: 0.97
```

In [14]:

```
from sklearn.neighbors import KNeighborsClassifier

# k-Nearest Neighbors Classifier
knn_classifier = KNeighborsClassifier()
knn_classifier.fit(X_train, y_train)
y_pred_knn = knn_classifier.predict(X_test)
accuracy_knn = accuracy_score(y_test, y_pred_knn)
print(f'k-NN Accuracy: {accuracy_knn:.2f}')
```

```
k-NN Accuracy: 0.95
```

In [15]:

```python
# Compare accuracies
accuracies = {
    'Logistic Regression': accuracy_log_reg,
    'Decision Tree': accuracy_dt,
    'Random Forest': accuracy_rf,
    'SVM': accuracy_svm,
    'k-NN': accuracy_knn
}

# Identify the best and worst performing models
best_model = max(accuracies, key=accuracies.get)
worst_model = min(accuracies, key=accuracies.get)

print("\nModel Accuracies:")
for model, accuracy in accuracies.items():
    print(f"{model}: {accuracy:.2f}")

print(f"\nBest Model: {best_model} with accuracy {accuracies[best_model]:.
print(f"Worst Model: {worst_model} with accuracy {accuracies[worst_model]:
```

```

Model Accuracies:
Logistic Regression: 0.97
Decision Tree: 0.93
Random Forest: 0.96
SVM: 0.97
k-NN: 0.95

Best Model: Logistic Regression with accuracy 0.97
Worst Model: Decision Tree with accuracy 0.93
```

In [ ]: