In [2]:

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

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	cor
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	

5 rows × 33 columns

In [3]:

```
# removing id column
data.drop("id", axis = 'columns', inplace = True)

# removing Unnamed: 32 column
data.drop("Unnamed: 32", axis = 'columns', inplace = True)
```

In [4]:

```
# converting categorical target variable to numerical i.e. diagnosis
data.replace({"diagnosis" : {"M":1,"B":0}}, inplace = True)
data.diagnosis
```

Out[4]:

```
0      1
1      1
2      1
3      1
4      1
      ..
564    1
565    1
566    1
567    1
568    0
Name: diagnosis, Length: 569, dtype: int64
```

In [5]:

```
# checking for relevant independent variables
data.groupby("diagnosis").mean()

# based on the mean we remove:
# 1.) symmetry_mean
# 2.) fractal_dimension_mean
# 3.) smoothness_worst
# 4.) fractal_dimension_worst
# from our analysis
```

Out[5]:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean

diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
0	12.146524	17.914762	78.075406	462.790196	0.092478	0.080085	0.046058
diagnosis	17.462830	21.604906	115.365377	978.376415	0.102898	0.145188	0.160775

2 rows × 30 columns

In [6]:

```
# dropping these variables from our data
data.drop(['symmetry_mean', 'fractal_dimension_mean', 'smoothness_worst', 'fractal_dimens
ion_worst'], axis = 'columns',
        inplace = True)
```

In [7]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
```

In [9]:

```
x = data.drop('diagnosis', axis = 'columns')
```

In [10]:

```
y = data.diagnosis
```

In [33]:

```
from sklearn.model_selection import cross_val_score
lr_score = cross_val_score(LogisticRegression(max_iter = 3500), x,y, cv = 5)
lr_score
```

Out[33]:

```
array([0.93859649, 0.94736842, 0.98245614, 0.92982456, 0.96460177])
```

In [34]:

```
rf_score = cross_val_score(RandomForestClassifier(n_estimators = 100), x,y, cv = 5)
rf_score
```

Out[34]:

```
array([0.92105263, 0.93859649, 0.98245614, 0.97368421, 0.99115044])
```

In [35]:

```
dt_score = cross_val_score(DecisionTreeClassifier(), x,y, cv = 5)
dt_score
```

Out[35]:

```
array([0.90350877, 0.9122807 , 0.92105263, 0.95614035, 0.92035398])
```

In [36]:

```
svm_score = cross_val_score(SVC(), x,y, cv = 5)
svm_score
```

Out[36]:

```
array([0.85087719, 0.89473684, 0.92982456, 0.93859649, 0.9380531 ])
```

In [38]:

```
# taking mean of all scores and selecting the best algorithm
import numpy as np
print("Mean logistic regression score is: ",np.mean(lr_score))
```

```python
print("Mean random forest score: ",np.mean(rf_score))
print("Mean decision tree score: ",np.mean(dt_score))
print("Mean support vector machine score: ",np.mean(svm_score))
```

```
Mean logistic regression score is:  0.9525694767893184
Mean random forest score:  0.9613879832324173
Mean decision tree score:  0.9226672876882471
Mean support vector machine score:  0.9104176370128861
```

So the best algorithm for our dataset is Random Forest