In [186]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
```

Load the data

In [187]:

```
data=pd.read_csv('breast_cancer.csv')
data
```

Out[187]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	c
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	
...	...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	

569 rows × 33 columns

In [188]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       569 non-null    int64
 1   diagnosis                569 non-null    object
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
```

```
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

In [189]:

```
data.describe()
```

Out[189]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	conc
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	

8 rows × 32 columns

In [190]:

```
data.head()
```

Out[190]:

	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 [191]:

```
data.tail()
```

Out[191]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	con
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	

5 rows × 33 columns

In [192]:

```
data.shape
```

Out[192]:

```
(569, 33)
```

In [193]:

```
data.describe().T
```

Out[193]:

	count	mean	std	min	25%	50%	75%	
id	569.0	3.037183e+07	1.250206e+08	8670.000000	869218.000000	906024.000000	8.813129e+06	9.113
radius_mean	569.0	1.412729e+01	3.524049e+00	6.981000	11.700000	13.370000	1.578000e+01	2.811
texture_mean	569.0	1.928965e+01	4.301036e+00	9.710000	16.170000	18.840000	2.180000e+01	3.928
perimeter_mean	569.0	9.196903e+01	2.429898e+01	43.790000	75.170000	86.240000	1.041000e+02	1.885
area_mean	569.0	6.548891e+02	3.519141e+02	143.500000	420.300000	551.100000	7.827000e+02	2.501
smoothness_mean	569.0	9.636028e-02	1.406413e-02	0.052630	0.086370	0.095870	1.053000e-01	1.634
compactness_mean	569.0	1.043410e-01	5.281276e-02	0.019380	0.064920	0.092630	1.304000e-01	3.454
concavity_mean	569.0	8.879932e-02	7.971981e-02	0.000000	0.029560	0.061540	1.307000e-01	4.268
concave points_mean	569.0	4.891915e-02	3.880284e-02	0.000000	0.020310	0.033500	7.400000e-02	2.012
symmetry_mean	569.0	1.811619e-01	2.741428e-02	0.106000	0.161900	0.179200	1.957000e-01	3.040
fractal_dimension_mean	569.0	6.279761e-02	7.060363e-03	0.049960	0.057700	0.061540	6.612000e-02	9.744
radius_se	569.0	4.051721e-01	2.773127e-01	0.111500	0.232400	0.324200	4.789000e-01	2.873
texture_se	569.0	1.216853e+00	5.516484e-01	0.360200	0.833900	1.108000	1.474000e+00	4.885
perimeter_se	569.0	2.866059e+00	2.021855e+00	0.757000	1.606000	2.287000	3.357000e+00	2.198
area_se	569.0	4.033708e+01	4.549101e+01	6.802000	17.850000	24.530000	4.519000e+01	5.422
smoothness_se	569.0	7.040979e-03	3.002518e-03	0.001713	0.005169	0.006380	8.146000e-03	3.113
compactness_se	569.0	2.547814e-02	1.790818e-02	0.002252	0.013080	0.020450	3.245000e-02	1.354
concavity_se	569.0	3.189372e-02	3.018606e-02	0.000000	0.015090	0.025890	4.205000e-02	3.960
concave points_se	569.0	1.179614e-02	6.170285e-03	0.000000	0.007638	0.010930	1.471000e-02	5.279
symmetry_se	569.0	2.054230e-02	8.266372e-03	0.007882	0.015160	0.018730	2.348000e-02	7.895
fractal_dimension_se	569.0	3.794904e-03	2.646071e-03	0.000895	0.002248	0.003187	4.558000e-03	2.984
radius_worst	569.0	1.626919e+01	4.833242e+00	7.930000	13.010000	14.970000	1.879000e+01	3.604
texture_worst	569.0	2.567722e+01	6.146258e+00	12.020000	21.080000	25.410000	2.972000e+01	4.954
perimeter_worst	569.0	1.072612e+02	3.360254e+01	50.410000	84.110000	97.660000	1.254000e+02	2.512
area_worst	569.0	8.805831e+02	5.693570e+02	185.200000	515.300000	686.500000	1.084000e+03	4.254
smoothness_worst	569.0	1.323686e-01	2.283243e-02	0.071170	0.116600	0.131300	1.460000e-01	2.226

	count	mean	std	min	25%	50%	75%	
compactness_worst	569.0	2.542650e-01	1.573365e-01	0.027290	0.147200	0.211900	3.391000e-01	1.058
concavity_worst	569.0	2.721885e-01	2.086243e-01	0.000000	0.114500	0.226700	3.829000e-01	1.252
concave points_worst	569.0	1.146062e-01	6.573234e-02	0.000000	0.064930	0.099930	1.614000e-01	2.910
symmetry_worst	569.0	2.900756e-01	6.186747e-02	0.156500	0.250400	0.282200	3.179000e-01	6.638
fractal_dimension_worst	569.0	8.394582e-02	1.806127e-02	0.055040	0.071460	0.080040	9.208000e-02	2.075
Unnamed: 32	0.0	NaN	NaN	NaN	NaN	NaN	NaN	

Clean and Prepare the data

In [194]:

```
data.diagnosis.unique()
```

Out[194]:

```
array(['M', 'B'], dtype=object)
```

In [195]:

```
data['diagnosis'].value_counts()
```

Out[195]:

```
B    357
M    212
Name: diagnosis, dtype: int64
```

In [196]:

```
data.drop('id',axis=1,inplace=True)
```

In [197]:

```
data.drop('Unnamed: 32',axis=1,inplace=True)
```

In [198]:

```
data.head()
```

Out[198]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mea
0	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.300
1	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.086
2	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.197
3	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.241
4	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.198

5 rows × 31 columns

In [199]:

```
data['diagnosis']=data['diagnosis'].map({'M':1,'B':0})
```

In [200]:

```
data.head()
```

Out[200]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mea
0	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.300
1	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.086
2	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.197
3	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.241
4	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.198

5 rows × 31 columns

In [201]:

```
data.isnull().sum()
```

Out[201]:

```
diagnosis                0
radius_mean              0
texture_mean             0
perimeter_mean           0
area_mean                0
smoothness_mean          0
compactness_mean         0
concavity_mean           0
concave points_mean      0
symmetry_mean            0
fractal_dimension_mean   0
radius_se                0
texture_se               0
perimeter_se             0
area_se                  0
smoothness_se            0
compactness_se           0
concavity_se             0
concave points_se        0
symmetry_se              0
fractal_dimension_se     0
radius_worst             0
texture_worst            0
perimeter_worst          0
area_worst               0
smoothness_worst         0
compactness_worst        0
concavity_worst          0
concave points_worst     0
symmetry_worst           0
fractal_dimension_worst  0
dtype: int64
```

In [202]:

```
data.corr()
```

Out[202]:

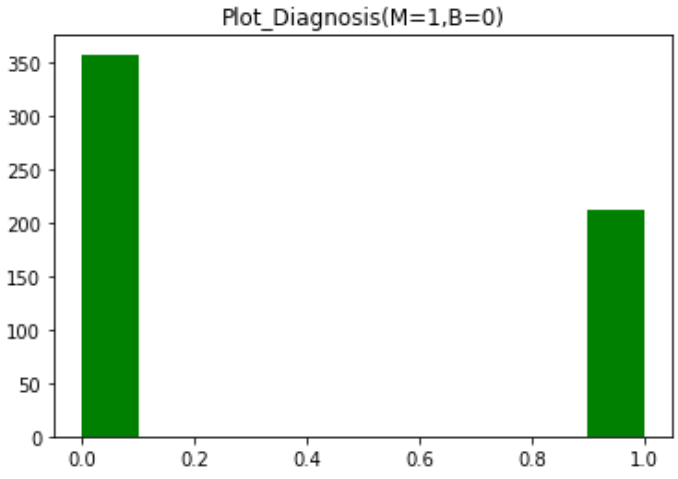
	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness
diagnosis	1.000000	0.730029	0.415185	0.742636	0.708984	0.358560	0.
radius_mean	0.730029	1.000000	0.323782	0.997855	0.987357	0.170581	0.
texture_mean	0.415185	0.323782	1.000000	0.329533	0.321086	-0.023389	0.
perimeter_mean	0.742636	0.997855	0.329533	1.000000	0.986507	0.207278	0.
area_mean	0.708984	0.987357	0.321086	0.986507	1.000000	0.177028	0.
smoothness_mean	0.358560	0.170581	-0.023389	0.207278	0.177028	1.000000	0

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness
compactness_mean	0.596534	0.506124	0.236702	0.556986	0.498502	0.659123	1.
concavity_mean	0.696360	0.676764	0.302418	0.716136	0.685983	0.521984	0.
concave points_mean	0.776614	0.822529	0.293464	0.850977	0.823269	0.553695	0.
symmetry_mean	0.330499	0.147741	0.071401	0.183027	0.151293	0.557775	0.
fractal_dimension_mean	-0.012838	-0.311631	-0.076437	-0.261477	-0.283110	0.584792	0.
radius_se	0.567134	0.679090	0.275869	0.691765	0.732562	0.301467	0.
texture_se	-0.008303	-0.097317	0.386358	-0.086761	-0.066280	0.068406	0.
perimeter_se	0.556141	0.674172	0.281673	0.693135	0.726628	0.296092	0.
area_se	0.548236	0.735864	0.259845	0.744983	0.800086	0.246552	0.
smoothness_se	-0.067016	-0.222600	0.006614	-0.202694	-0.166777	0.332375	0.
compactness_se	0.292999	0.206000	0.191975	0.250744	0.212583	0.318943	0.
concavity_se	0.253730	0.194204	0.143293	0.228082	0.207660	0.248396	0.
concave points_se	0.408042	0.376169	0.163851	0.407217	0.372320	0.380676	0.
symmetry_se	-0.006522	-0.104321	0.009127	-0.081629	-0.072497	0.200774	0.
fractal_dimension_se	0.077972	-0.042641	0.054458	-0.005523	-0.019887	0.283607	0.
radius_worst	0.776454	0.969539	0.352573	0.969476	0.962746	0.213120	0.
texture_worst	0.456903	0.297008	0.912045	0.303038	0.287489	0.036072	0.
perimeter_worst	0.782914	0.965137	0.358040	0.970387	0.959120	0.238853	0.
area_worst	0.733825	0.941082	0.343546	0.941550	0.959213	0.206718	0.
smoothness_worst	0.421465	0.119616	0.077503	0.150549	0.123523	0.805324	0.
compactness_worst	0.590998	0.413463	0.277830	0.455774	0.390410	0.472468	0.
concavity_worst	0.659610	0.526911	0.301025	0.563879	0.512606	0.434926	0.
concave points_worst	0.793566	0.744214	0.295316	0.771241	0.722017	0.503053	0.
symmetry_worst	0.416294	0.163953	0.105008	0.189115	0.143570	0.394309	0.
fractal_dimension_worst	0.323872	0.007066	0.119205	0.051019	0.003738	0.499316	0.

31 rows × 31 columns

In [204]:

```
plt.hist(data['diagnosis'],color='g')
plt.title('Plot_Diagnosis(M=1,B=0)')
plt.show()
```



Building Model

In [205]:

In [205]:

```
X=data.drop(['diagnosis'],axis=1)
```

In [206]:

```
Y=data['diagnosis']
```

In [207]:

```
from sklearn.model_selection import train_test_split
```

In [208]:

```
X_train, X_test, y_train, y_test = train_test_split(
  X, Y, test_size=0.33, random_state=42)
```

Feature scaling

In [214]:

```
from sklearn.preprocessing import StandardScaler
```

In [216]:

```
ss = StandardScaler()
x_train = ss.fit_transform(X_train)
x_test = ss.transform(X_test)
```

Models and find the Best one

Logistic Regression

In [217]:

```
from sklearn.linear_model import LogisticRegression
```

In [220]:

```
lr=LogisticRegression()
```

In [221]:

```
model=lr.fit(X_train,y_train)
```

```
/home/anjali/.local/lib/python3.6/site-packages/sklearn/linear_model/_logistic.py:765: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

In [222]:

```
prediction=model.predict(X_test)
```

In [223]:

```
prediction
```

Out[223]:

```
array([0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0,
```

```
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0])
```

In [227]:

```
from sklearn.metrics import confusion_matrix
```

In [229]:

```
cm=confusion_matrix(y_test,prediction)
cm
```

Out[229]:

```
array([[118,   3],
       [  3,  64]])
```

In [230]:

```
from sklearn.metrics import accuracy_score
```

In [231]:

```
accuracy_score(y_test,prediction)
```

Out[231]:

```
0.9680851063829787
```

Decision Tree

In [232]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [234]:

```
dec=DecisionTreeClassifier()
model1=dec.fit(X_train,y_train)
```

In [235]:

```
model1
```

Out[235]:

```
DecisionTreeClassifier()
```

In [236]:

```
prediction=model1.predict(X_test)
```

In [237]:

```
prediction
```

Out[237]:

```
array([0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0
```

```
       1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0])
```

In [238]:

```
cnn=confusion_matrix(y_test,prediction)
```

In [239]:

```
cnn
```

Out[239]:

```
array([[104,  17],
       [  4,  63]])
```

In [240]:

```
accuracy_score(y_test,prediction)
```

Out[240]:

```
0.8882978723404256
```

In [241]:

```
from sklearn.metrics import classification_report
```

In [242]:

```
print(classification_report(y_test,prediction))
```

```
              precision    recall  f1-score   support

           0       0.96      0.86      0.91       121
           1       0.79      0.94      0.86        67

    accuracy                           0.89       188
   macro avg       0.88      0.90      0.88       188
weighted avg       0.90      0.89      0.89       188
```

In [ ]:

```

```