# 50. Regression algorithms o Decision Tree Regressor o Random Forest Regressor o Support Vector Regression

In [11]:
```
import pandas as p
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split

# Breast cancer dataset
data = p.read_csv("breast-cancer.csv")
data.info()
data.drop(["id"],axis=1,inplace=True)
M=data[data.diagnosis=="M"]
B=data[data.diagnosis=="B"]

data.diagnosis=[1 if i == "M" else 0 for i in data.diagnosis]
x=data.drop(["diagnosis"],axis=1)
y=data.diagnosis.values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=42)

from sklearn.tree import DecisionTreeRegressor
model=DecisionTreeRegressor()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print("\nAccuracy of the model using Decision tree regression algorithm is ",r2_score(y_test,y_pred))

from sklearn.ensemble import RandomForestRegressor
model1 = RandomForestRegressor()
model1.fit(x_train,y_train)
y_pred1 = model1.predict(x_test)
print("\nAccuracy of the model using Random forest regression algorithm is ",r2_score(y_test,y_pred1))

from sklearn.svm import SVR
model2 = SVR(kernel='rbf')
model2.fit(x_train,y_train)
y_pred2 = model2.predict(x_test)
print("\nAccuracy of the model using Support vector regression algorithm is ",r2_score(y_test,y_pred2))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 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
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB

Accuracy of the model using Decision tree regression algorithm is  0.6984126984126984

Accuracy of the model using Random forest regression algorithm is  0.852717328042328

Accuracy of the model using Support vector regression algorithm is  0.8114302960086689
```

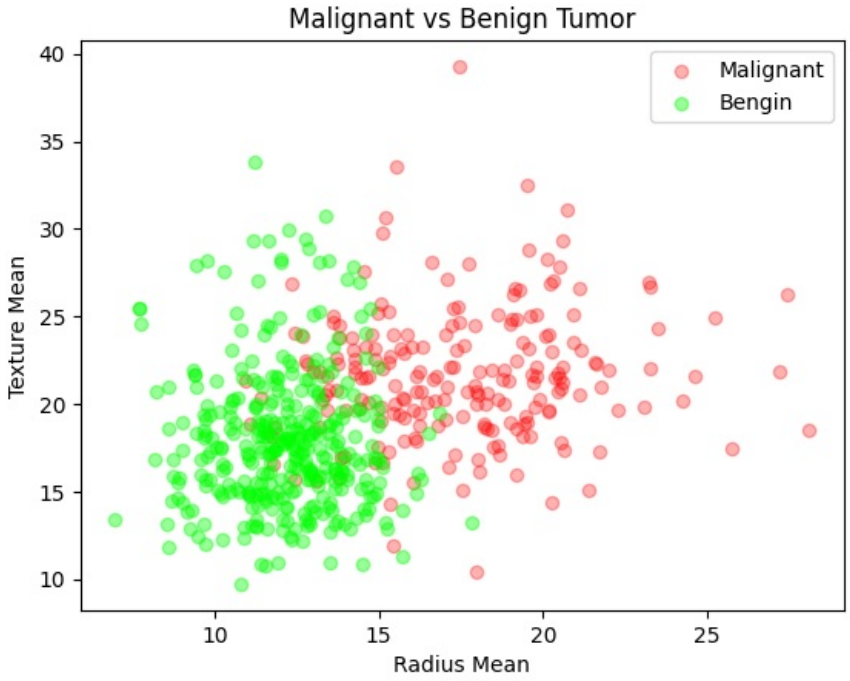
# 51. Build decision tree-based model for Breast Cancer Wisconsin (diagnostic) dataset.[Classifier]

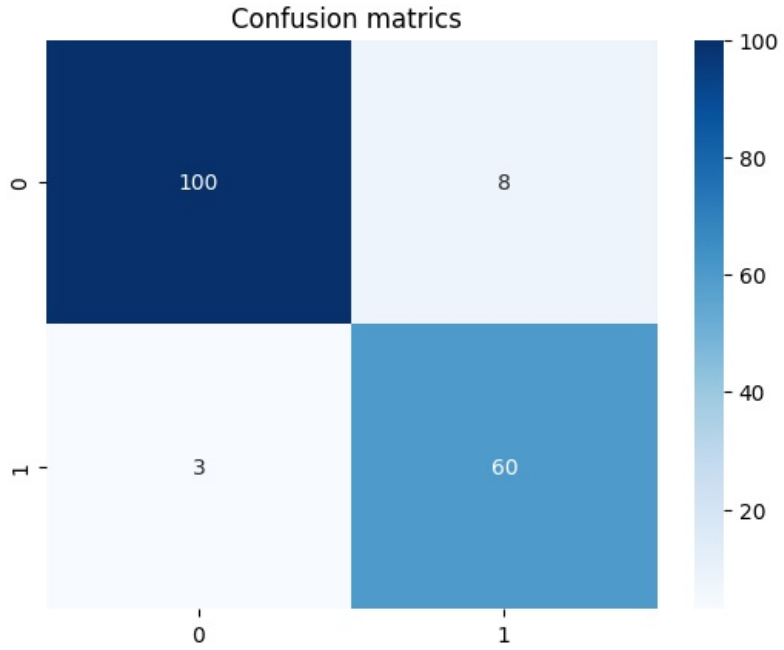
In [16]:

```
import numpy as n
import pandas as p
import seaborn as s
import matplotlib.pyplot as m
from sklearn.metrics import r2_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix

data = p.read_csv("breast-cancer.csv")
data.info()
data.drop(["id"],axis=1,inplace=True)
M=data[data.diagnosis=="M"]
B=data[data.diagnosis=="B"]

m.title("Malignant vs Benign Tumor")
m.xlabel("Radius Mean"); m.ylabel("Texture Mean")
m.scatter(M.radius_mean,M.texture_mean,color='red',label='Malignant',alpha=0.3)
m.scatter(B.radius_mean,B.texture_mean,color='lime',label='Bengin',alpha=0.4)
m.legend(); m.show()

data.diagnosis=[1 if i == "M" else 0 for i in data.diagnosis]
x=data.drop(["diagnosis"],axis=1)
y=data.diagnosis.values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=42)

dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_pred=dt.predict(x_test)

#Confusion Matrix
```

```
cm = confusion_matrix(y_test,y_pred)
s.heatmap(cm,annot=True,fmt='d',cmap="Blues")
m.title("Confusion matrics ")
m.show()
print("Accuracy of the classifier model is",accuracy_score(y_test,y_pred))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 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
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB
```






```
Accuracy of the classifier model is 0.935672514619883
```

In [ ]: