In [1]:
```
# Load libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split funct
from sklearn import metrics #Import scikit-learn metrics module for accuracy calcula
```

In [2]:
```
import matplotlib.pyplot as plt
```

```
Bad key "text.kerning_factor" on line 4 in
C:\Users\91920\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_tes
t_patch.mplstyle.
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.1.3/matplotlibrc.template
or from the matplotlib source distribution
```

In [3]:
```
# Load dataset
pima = pd.read_csv("C:/Users/91920/Python/diabetes.csv",)
```

In [4]:
```
pima.head()
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148	72	35	0	33.6	0.627	50
1	1	85	66	29	0	26.6	0.351	31
2	8	183	64	0	0	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33

In [5]:
```
print("dimension of diabetes data: {}".format(pima.shape))
```

```
dimension of diabetes data: (768, 9)
```

In [6]:
```
print(pima.groupby('Outcome').size())
```

```
Outcome
0    500
1    268
dtype: int64
```

In [7]:
```
X_train, X_test, y_train, y_test = train_test_split(pima.loc[:, pima.columns != 'Out
```

In [8]:
```
print(y_train.value_counts())
print(y_test.value_counts())
```

```
0    375
1    201
Name: Outcome, dtype: int64
0    125
1     67
Name: Outcome, dtype: int64
```

In [9]:
```
feature_name=list(X_train.columns)
class_name = list(y_train.unique())
feature_name
```

Out[9]:
```
['Pregnancies',
 'Glucose',
 'BloodPressure',
 'SkinThickness',
 'Insulin',
 'BMI',
 'DiabetesPedigreeFunction',
 'Age']
```

In [10]:
```
class_name
```

Out[10]:
```
[0, 1]
```

In [11]:
```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

In [12]:
```
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7135416666666666
```

In [17]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=20)
model.fit(X_train, y_train)
```

Out[17]:
```
RandomForestClassifier(n_estimators=20)
```

In [19]:
```
y_predicted = model.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_predicted))
```

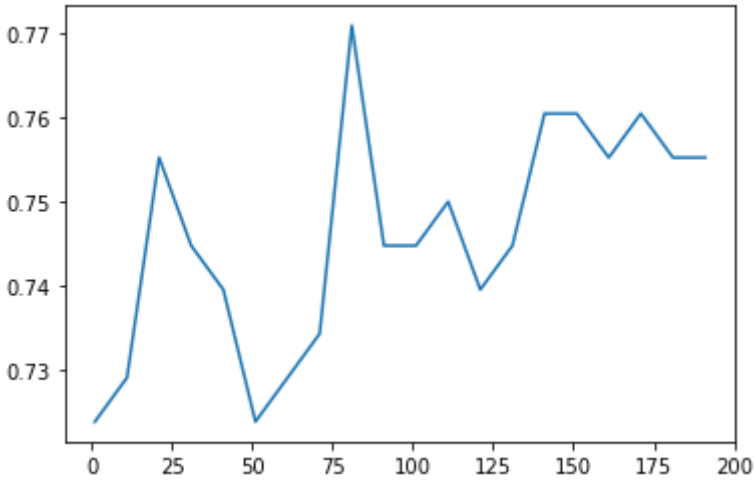
```
Accuracy: 0.7447916666666666
```

In [22]:
```
acc=[]
for i in range(1,200,10):
    model = RandomForestClassifier(n_estimators=20)
    model.fit(X_train, y_train)
    y_predicted = model.predict(X_test)
    a=metrics.accuracy_score(y_test, y_predicted)
    acc.append(a)
print(acc)
```

```
[0.7239583333333334, 0.7291666666666666, 0.7552083333333334, 0.7447916666666666, 0.7
395833333333334, 0.7239583333333334, 0.7291666666666666, 0.734375, 0.770833333333333
4, 0.7447916666666666, 0.7447916666666666, 0.75, 0.7395833333333334, 0.7447916666666
666, 0.7604166666666666, 0.7604166666666666, 0.7552083333333334, 0.7604166666666666,
0.7552083333333334, 0.7552083333333334]
```

In [23]:
```
plt.plot(range(1,200,10), acc)
```

```
plt.show()
```



In [26]:
```
pwd
```

Out[26]:
```
'C:\\Users\\91920\\Machine Learning'
```

In [ ]:
```

```