Assignment 10

You will be using the Breast Cancer Data Set (Links to an external site.)Links to an external site. used in Lesson 8.

Data Set Information: There are 10 predictors, all quantitative, and a binary dependent variable, indicating the presence or absence of breast cancer. The predictors are anthropometric data and parameters which can be gathered in routine blood analysis. Prediction models based on these predictors, if accurate, can potentially be used as a biomarker of breast cancer.

Attribute Information: Quantitative Attributes: Age (years) BMI (kg/m2) Glucose (mg/dL) Insulin (µU/mL) HOMA Leptin (ng/mL) Adiponectin (µg/mL) Resistin (ng/mL) MCP-1(pg/dL) (ng/mL) Labels:

1 = Healthy Controls

2 = Patients

In [1]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.model_selection import cross_validate
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
```

In [2]:
```
os.chdir(r"E:\GoogleDriveNew\PSU\DAAN862\Course contents\Lesson 10")
breastcancer = pd.read_csv("dataR2.csv")
```

1. Explore the Brest Cancer Data (10 points)

In [3]:
```
breastcancer.head()
```

Out[3]:

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP.1	C
0	48	23.500000	70	2.707	0.467409	8.8071	9.702400	7.99585	417.114	1
1	83	20.690495	92	3.115	0.706897	8.8438	5.429285	4.06405	468.786	1
2	82	23.124670	91	4.498	1.009651	17.9393	22.432040	9.27715	554.697	1
3	68	21.367521	77	3.226	0.612725	9.8827	7.169560	12.76600	928.220	1
4	86	21.111111	92	3.549	0.805386	6.6994	4.819240	10.57635	773.920	1

In [4]:
```
breastcancer.describe()
```

Out[4]:

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponec
count	116.000000	116.000000	116.000000	116.000000	116.000000	116.000000	116.0000
mean	57.301724	27.582111	97.793103	10.012086	2.694988	26.615080	10.18087
std	16.112766	5.020136	22.525162	10.067768	3.642043	19.183294	6.843341
min	24.000000	18.370000	60.000000	2.432000	0.467409	4.311000	1.656020
25%	45.000000	22.973205	85.750000	4.359250	0.917966	12.313675	5.474282
50%	56.000000	27.662416	92.000000	5.924500	1.380939	20.271000	8.352692
75%	71.000000	31.241442	102.000000	11.189250	2.857787	37.378300	11.81597(
max	89.000000	38.578759	201.000000	58.460000	25.050342	90.280000	38.04000

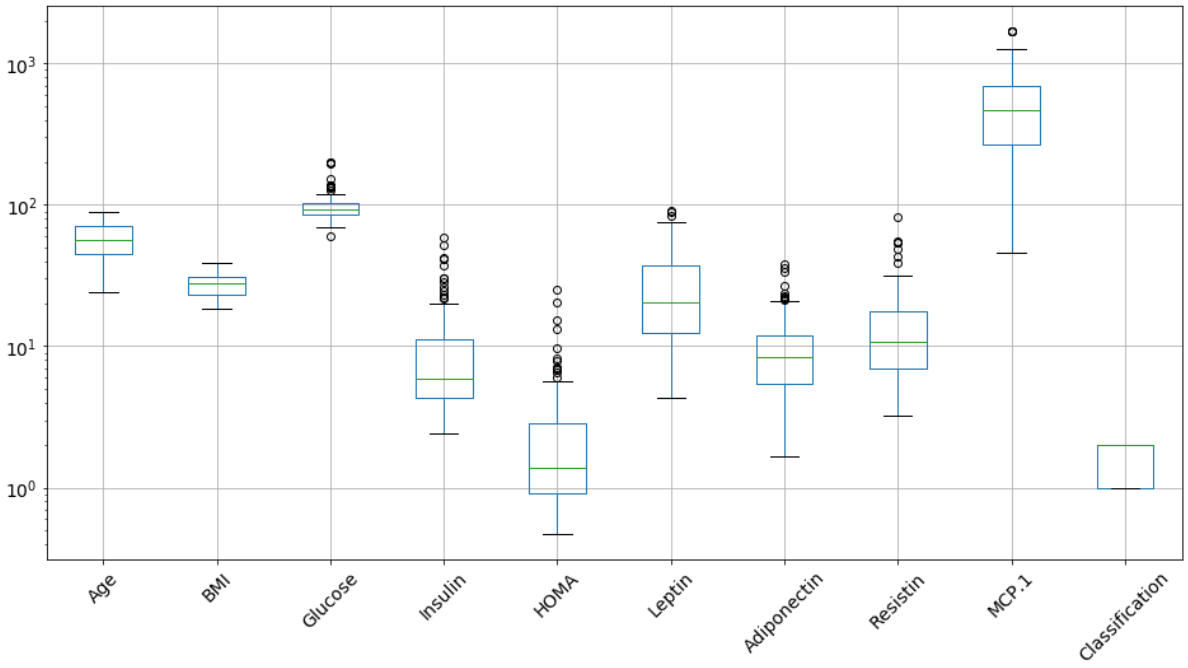
In [5]:
```
breastcancer.Classification.value_counts()
```

Out[5]:
```
2    64
1    52
Name: Classification, dtype: int64
```

In [6]:
```
breastcancer.isnull().sum()
```

Out[6]:
```
Age               0
BMI               0
Glucose           0
Insulin           0
HOMA              0
Leptin            0
Adiponectin       0
Resistin          0
MCP.1             0
Classification    0
dtype: int64
```

In [7]:
```
plt.figure(figsize = (16, 8))
breastcancer.boxplot(rot = 45, fontsize=14)
plt.yscale('log')
```



2. Build and evaluate SCV models with different kernel function (40 points)

In [8]:
```
X = breastcancer.iloc[:, 0:9]
y = breastcancer.Classification
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, ran
dom_state = 12)
```

In [9]:
```
result = pd.DataFrame()
i = 0
for kerneltype in ['linear', 'rbf', 'poly']:
    svm_model = svm.SVC(kernel = kerneltype)
    svm_model.fit(X_train, y_train)
    ytrain_pred = svm_model.predict(X_train)
    train_accuracy = metrics.accuracy_score(y_train, ytrain_pred)
    ytest_pred = svm_model.predict(X_test)
    test_accuracy = metrics.accuracy_score(y_test, ytest_pred)
    result[i] = [kerneltype, train_accuracy, test_accuracy]
    i += 1
```

In [10]:
```
result.index =  ['kernel', 'train_accuracy', 'test_accuracy']
result
```

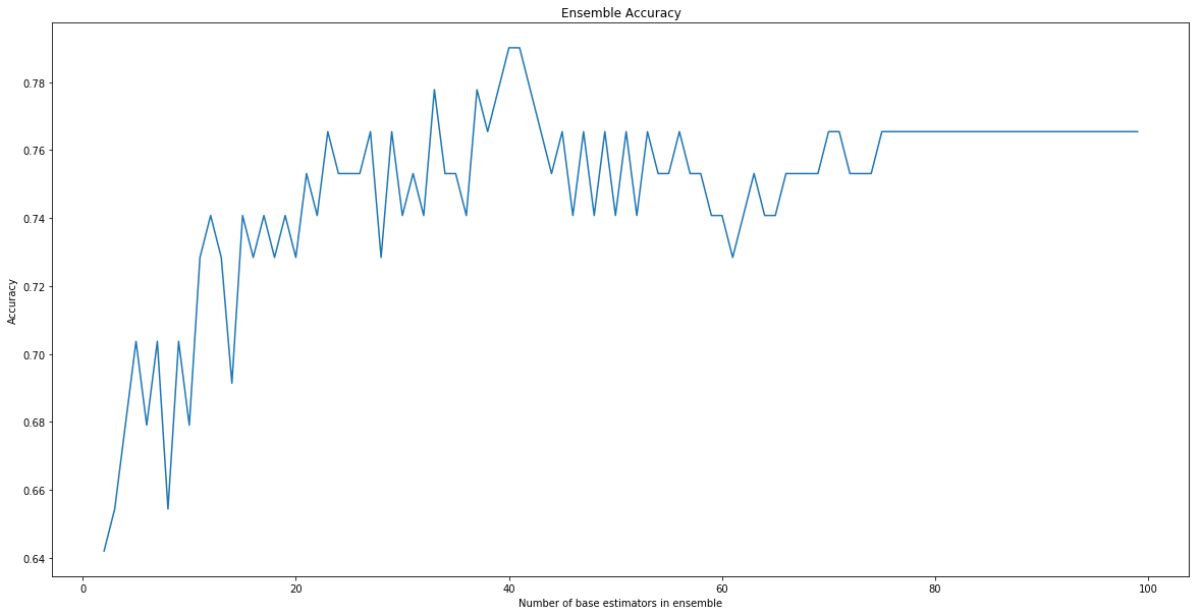
Out[10]:

	0	1	2
kernel	linear	rbf	poly
train_accuracy	0.753086	1	0.987654
test_accuracy	0.628571	0.457143	0.714286

3. Find the best n_estimator for Random Forests model (25 points)

In [11]:
```
# Accuracy vs n_estimator
n_estimator=range(2, 100, 1)
RF_results = []
for i in n_estimator:
    rf=RandomForestClassifier(n_estimators=i, random_state=10)
    scores=cross_val_score(rf, X_train, y_train)
    RF_results.append(scores.mean())
plt.figure(figsize=(20,10))
plt.plot(n_estimator, RF_results)
plt.title('Ensemble Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Number of base estimators in ensemble')
```

Out[11]: Text(0.5,0,'Number of base estimators in ensemble')



In [12]:
```
# find the n_estimator values at the maximum of accuracy.
maxindex = RF_results.index(max(RF_results))
n_estimator[maxindex]
```

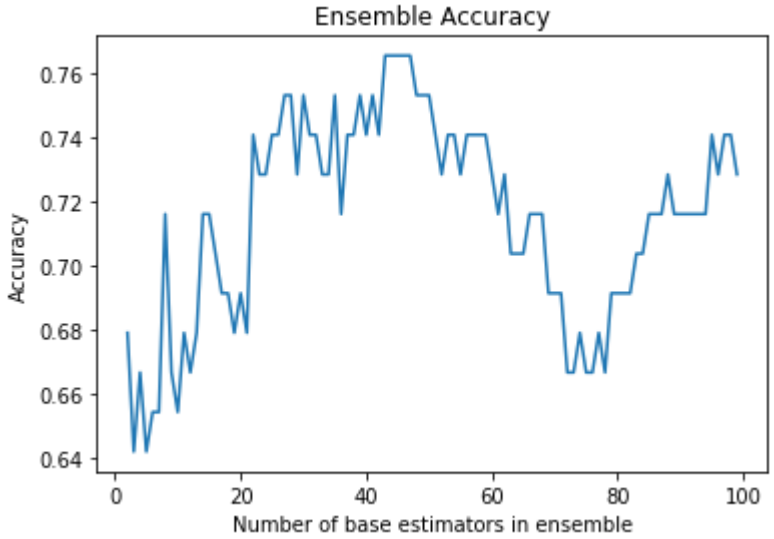
Out[12]: 40

4. Find the best n_estimatro for Adaboost model (25 points)

In [13]:
```
AB_results = []
n_estimators = range (2, 100)
for i in n_estimators:
    AB_model = AdaBoostClassifier(n_estimators = i, learning_rate = 0.1)
    scores = cross_val_score(AB_model, X_train, y_train)
    AB_results.append(scores.mean())
```

In [14]:
```
plt.figure()
plt.plot(n_estimators, AB_results)
plt.title('Ensemble Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Number of base estimators in ensemble')
```

Out[14]: Text(0.5,0,'Number of base estimators in ensemble')



In [15]:
```
maxindex = AB_results.index(max(AB_results))
n_estimator[maxindex]
```

Out[15]: 43