**LAB - 3**

**MACHINE LEARNING**

**PRIYANSHU SHARMA**

**15BCE1282**

**LOGISTIC REGRESSION**

**CODE**

#logistic regression

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

data = pd.read\_csv('C:/Users/PRIYANSHU SHARMA/Desktop/PRIYANSHU/6 STUDY/6 SEMSTER/MACHINE LEARNING/LAB/breast.csv')

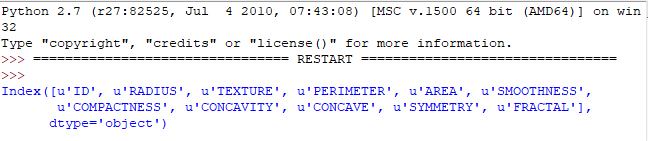
data.head()

colnames=['ID', 'RADIUS', 'TEXTURE', 'PERIMETER', 'AREA', 'SMOOTHNESS', 'COMPACTNESS', 'CONCAVITY', 'CONCAVE', 'SYMMETRY', 'FRACTAL']

data = pd.read\_csv('C:/Users/PRIYANSHU SHARMA/Desktop/PRIYANSHU/6 STUDY/6 SEMSTER/MACHINE LEARNING/LAB/breast.csv', names=colnames, header=None)

data.head()

print(data.columns)

****

data.describe()

def sigmoid(z):

return 1.0 / (1.0 + np.exp(-z))

z = np.arange(-7, 7, 0.1)

phi\_z = sigmoid(z)

plt.plot(z, phi\_z)

plt.axvline(0.0, color='k')

plt.axhspan(0.0, 1.0, facecolor='1.0', alpha=1.0, ls='dotted')

plt.axhline(y=0.5, ls='dotted', color='k')

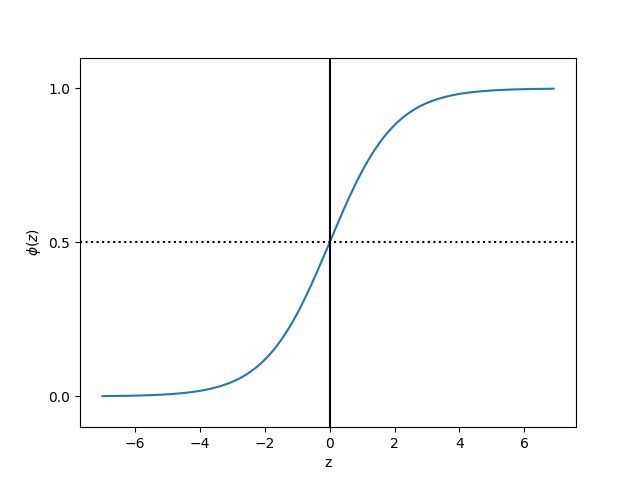
plt.yticks([0.0, 0.5, 1.0])

plt.ylim(-0.1, 1.1)

plt.xlabel('z')

plt.ylabel('$\phi (z)$')

plt.show()

****

from sklearn.cross\_validation import train\_test\_split

X = data.iloc[0:, [1,5]].values

X

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,data['FRACTAL'], test\_size=0.3, random\_state=0)

col=['COMPACTNESS','SMOOTHNESS']

#For visualization

from matplotlib.colors import ListedColormap

def plot\_decision\_regions(X, y, classifier,test\_idx=None, resolution=0.02):

# setup marker generator and color map

markers = ('s', 'x', 'o', '^', 'v')

colors = ('green', 'red', 'lightgreen', 'gray', 'cyan')

cmap = ListedColormap(colors[:len(np.unique(y))])

# plot the decision surface

x1\_min, x1\_max = X[:, 0].min() - 1, X[:, 0].max() + 1

x2\_min, x2\_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx1, xx2 = np.meshgrid(np.arange(x1\_min, x1\_max, resolution),

np.arange(x2\_min, x2\_max, resolution))

Z = classifier.predict(np.array([xx1.ravel(), xx2.ravel()]).T)

Z = Z.reshape(xx1.shape)

plt.contourf(xx1, xx2, Z, alpha=0.4, cmap=cmap)

plt.xlim(xx1.min(), xx1.max())

plt.ylim(xx2.min(), xx2.max())

# plot class samples

for idx, cl in enumerate(np.unique(y)):

plt.scatter(x=X[y == cl, 0], y=X[y == cl, 1],

alpha=0.8, c=cmap(idx),

marker=markers[idx], label='Benign' if cl == 2 else 'Malignant')

from sklearn.linear\_model import LogisticRegression

from sklearn.preprocessing import StandardScaler

import numpy as np

sc = StandardScaler()

sc.fit(X\_train)

X\_train\_std = sc.transform(X\_train)

X\_test\_std = sc.transform(X\_test)

lr = LogisticRegression(C=1000.0, random\_state=0)

lr.fit(X\_train\_std, Y\_train)

print("Accuracy on the training set: %.3f" % lr.score(X\_train,Y\_train))

print("Accuracy on the testing set: %.3f" % lr.score(X\_test,Y\_test))

****

X\_combined\_std = np.vstack((X\_train\_std, X\_test\_std))

y\_combined = np.hstack((Y\_train, Y\_test))

plot\_decision\_regions(X\_combined\_std,y\_combined, classifier=lr)

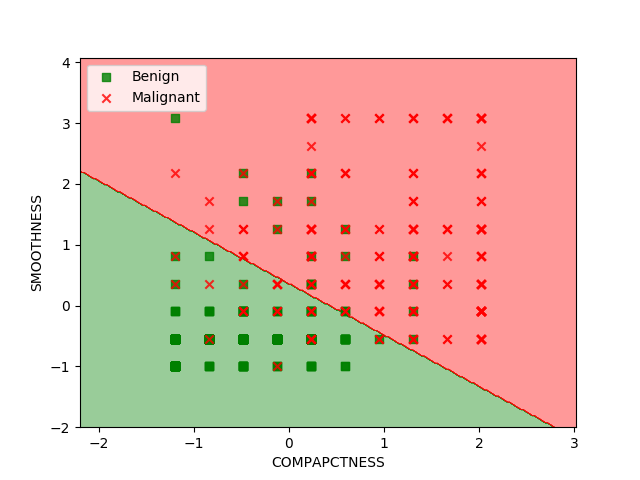
plt.xlabel('COMPAPCTNESS')

plt.ylabel('SMOOTHNESS')

plt.legend(loc='upper left')

plt.show()

**OUTPUT**

****