Summative2

January 6, 2023

1 Q1

In [1]:
```
## Load dataset

from sklearn import datasets
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
breast_cancer_data = datasets.load_breast_cancer()
df = breast_cancer_data.data

labels = breast_cancer_data.target
```

In [2]:
```
# Reshaping labels to append to dataframe

labels = np.reshape(labels,(569,1))
```

In [3]:
```
breast_cancer_df = np.concatenate([df, labels], axis=1)
```

In [4]:
```
# converting to dataframe

breast_cancer_df = pd.DataFrame(breast_cancer_df)
```

In [5]:
```
features = breast_cancer_data.feature_names
```

In [6]:
```
# Adding label column name

features_labels = np.append(features, "label")

#Adding the labels to the dataframe columns
breast_cancer_df.columns = features_labels
```

In [7]:
```
#Separate features and target variables
X = breast_cancer_df.loc[:, features].values
y = breast_cancer_df.loc[:, 'label'].values

#Normalising data using StandardScaler
from sklearn.preprocessing import StandardScaler
#x = breast_cancer_df.loc[:, features].values
#x = StandardScaler().fit_transform(x)
```

In [8]:
```python
#3 Splitting the X and Y into the
# Training set and Testing set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 0)
```

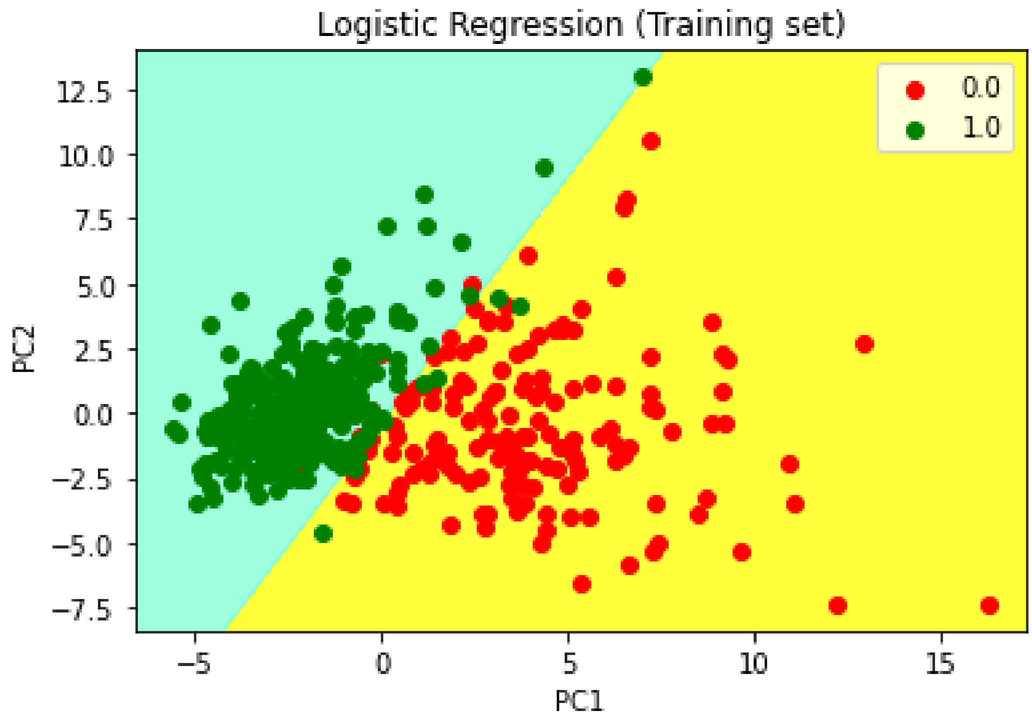
In [9]:
```python
#4 performing preprocessing part
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

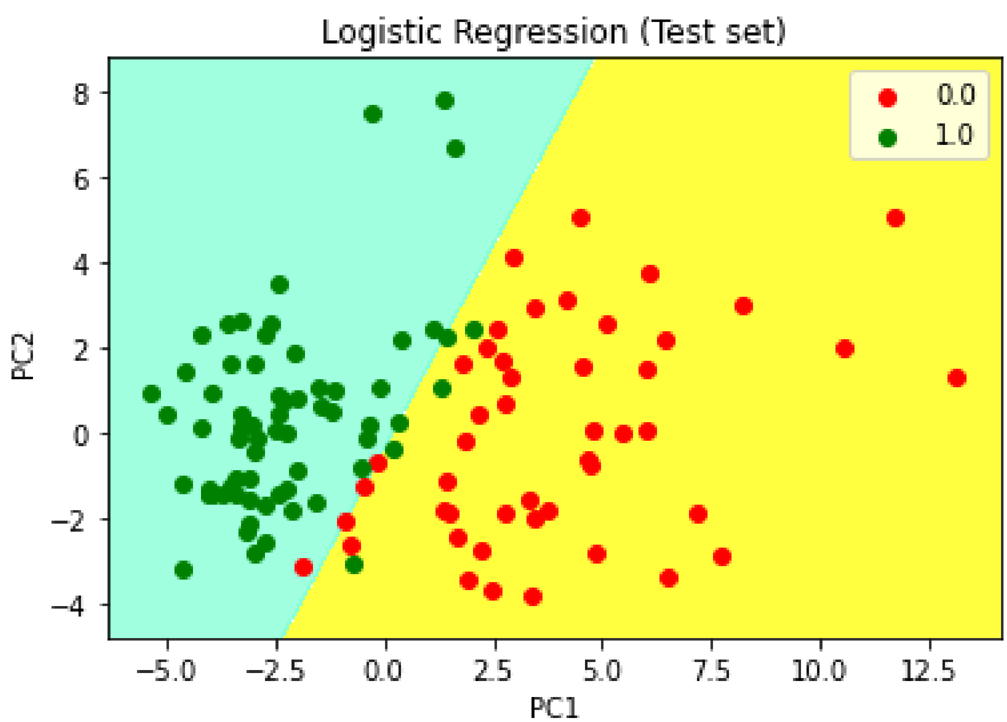
In [10]:
```python
#5 Applying PCA function on training
# and testing set of X component
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
```

In [11]:
```python
#6 Fitting Logistic Regression To the training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)

#7 Predicting the test set result using
# predict function under LogisticRegression
y_pred = classifier.predict(X_test)

#8 making confusion matrix between
# test set of Y and predicted value.
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

#9 Predicting the training set
# result through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,
                               stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1,
                               stop = X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),
                                                  X2.ravel()]).T).reshape(X1.shape), a
             cmap = ListedColormap(('yellow', 'white', 'aquamarine')))

plt.xlim(X1.min(), X1.max())
```

```python
plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green', 'blue'))(i), label = j)

plt.title('Logistic Regression (Training set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend() # to show legend
# show scatter plot
plt.show()


#10 Visualising the Test set results through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,stop = X_set[:, 0].max()
                     np.arange(start = X_set[:, 1].min() - 1,
                               stop = X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),
                                                   X2.ravel()]).T).reshape(X1.shape), a
             cmap = ListedColormap(('yellow', 'white', 'aquamarine')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                color = ListedColormap(('red', 'green', 'blue'))(i), label = j)

# title for scatter plot
plt.title('Logistic Regression (Test set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend()

# show scatter plot
plt.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as valu
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as valu
```


Logistic Regression (Training set)
0.0
1.0
PC1
PC2


Logistic Regression (Test set)
0.0
1.0
PC1
PC2

2 Q2

2.1 Neural network calculations

2.2 Steps to train a simple neural network

- Collect and preprocess the data: this includes data cleansing, formatting the data to have suitable data type and splitting these data into training and test sets.

- Define the model: Next, we must define the neural network architecture. This will include choosing the number of layers, the number of nodes in each layer, and the activation functions to use.

- Compile the model: After defining the model, you will need to compile it with a loss function, an optimizer, and any metrics that need to be tracked

- Train the model: train the model on the training data. This will involve providing the model with the training data and allowing it to learn the relationships between the input features and the target variables.

- Evaluate the model: After training, we need to evaluate the model's performance on the test data. This will give us a sense of how well the model is able to generalise to unseen data. For example we can switch between LogSoftMax, NLLLoss and Cross Entropy to check the difference in performance.

- Fine-tune the model: Depending on the results of the evaluation, we may want to adjust the model's architecture or hyperparameters to improve its performance. This process is known as fine-tuning.

- Make predictions: Finally, we can use the trained model to make predictions on new data.

Forward Pass

$$\text{Sum } h_1 = i_1 \times w_1 + i_2 \times w_3 + b_1$$
$$= (0.2 \times 0.2) + (0.4 \times 0.1) + 0.35$$
$$= 0.04 + 0.04 + 0.35$$
$$= 0.43$$

Pass the weighted sum through logistic function

$$\text{Output } h_1 = \frac{1}{1+e^{-\text{Sum} h_1}} = \frac{1}{1+e^{-0.43}} = 0.60587$$

H_2

$$\text{Sum } h_2 = i_1 \times w_2 + i_2 \times w_4 + b1$$
$$= (0.2 \times 0.4) + (0.4 \times 0.3) + 0.35$$
$$= 0.08 + 0.12 + 0.35$$
$$= 0.55$$

$$\text{Output } h_2 = \frac{1}{1+e^{-\text{Sum} h_2}} = \frac{1}{1+e^{-0.55}} = 0.63414$$

Using the outputs for the next layer

$$\text{Sum } O_1 = \text{Output} h_1 \times w_5 + \text{output} h_2 \times w_6 + b2$$
$$= (0.60587 \times 0.8) + (0.63414 \times 0.6) + 0.45$$
$$= 0.48469 + 0.380484 + 0.45$$
$$= 1.31518$$

$$\text{Output } o_1 = \frac{1}{1+e^{-\text{sum} o_1}} = \frac{1}{1+e^{-1.31518}} = 0.788379$$

$$Sum_{o_2} = output_{h_1} \times w_7 + output_{h_2} \times w_8 + b_2$$

$$= (\cancel{0.62481}\ 0.60587 \times 0.5) + (0.63414 \times 0.7) + 0.45$$

$$= \cancel{0.312405}\ 0.302935 + 0.443898 + 0.45$$

$$= \cancel{1.206303}\ 1.196833$$

$$Output_{o_2} = \frac{1}{1+e^{-Sum_{o_2}}} = \frac{1}{1+e^{-\cancel{1.206303}\ 1.19683}} = \cancel{0.769644} \approx 0.76796$$

Computing the total error

The expected outputs are $o_1 = 0.02$

$o_2 = 0.85$

$$\text{Error Formula} = \sum \frac{1}{2}(\text{target} - \text{output})^2$$

$$E_1 = \frac{1}{2}(\text{target}_1 - \text{output}_1)^2$$

$$= \frac{1}{2}(0.02 - 0.788379)^2$$

$$= 0.2952$$

$$E_2 = \frac{1}{2}(\text{target}_2 - \text{output}_2)^2$$

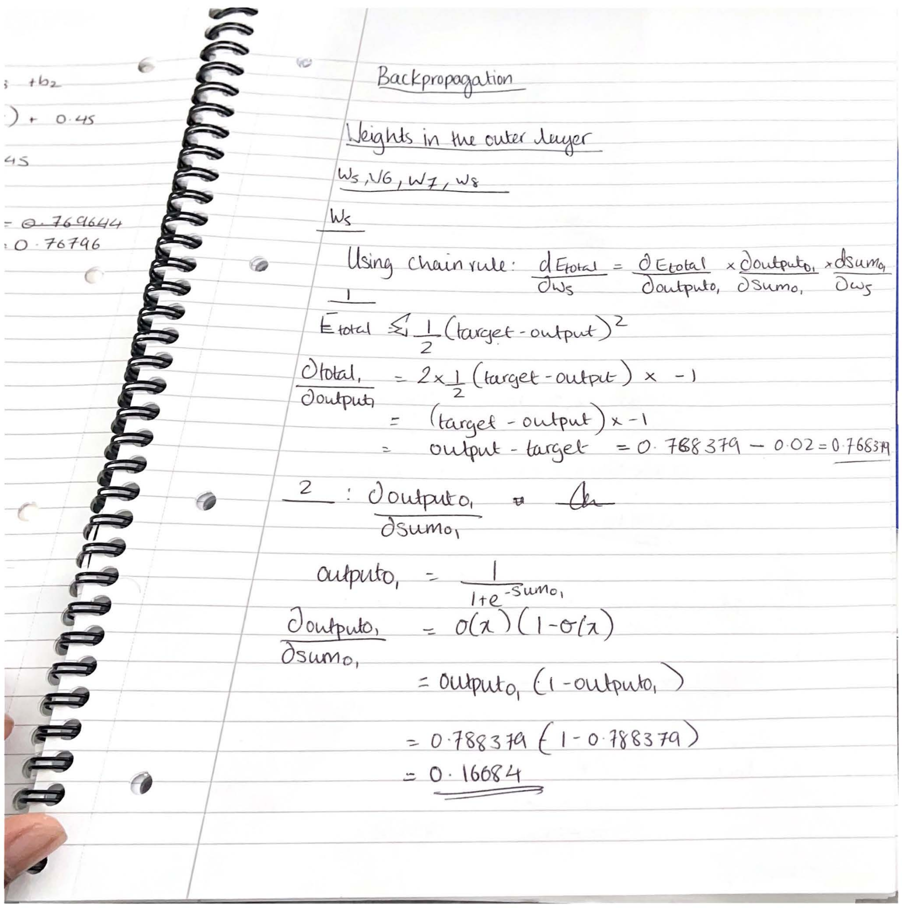
$$= \frac{1}{2}(0.85 - \cancel{0.769644}\ 0.76796)^2$$

$$= \cancel{0.003229}\ 0.003365$$

$$E_{total} = 0.2952 + \cancel{0.003229}\ 0.003365$$

$$= \cancel{0.2984}$$

$$0.2986$$



Backpropagation

Weights in the outer layer

w_5, w_6, w_7, w_8

w_5

Using chain rule: $\frac{d E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial output_{o_1}} \times \frac{\partial output_{o_1}}{\partial sum_{o_1}} \times \frac{\partial sum_{o_1}}{\partial w_5}$

1

$E_{total} = \sum \frac{1}{2}(target - output)^2$

$\frac{\partial total_1}{\partial output_1} = 2 \times \frac{1}{2}(target - output) \times -1$

$= (target - output) \times -1$

$= output - target = 0.788379 - 0.02 = 0.768379$

2 : $\frac{\partial output_{o_1}}{\partial sum_{o_1}}$

$output_{o_1} = \frac{1}{1+e^{-sum_{o_1}}}$

$\frac{\partial output_{o_1}}{\partial sum_{o_1}} = \sigma(x)(1-\sigma(x))$

$= output_{o_1}(1 - output_{o_1})$

$= 0.788379(1 - 0.788379)$

$= 0.16684$

3

$Sum_{o_1} = output_{h_1} \times w_5 + output_{h_2} \times w_6 + b_2$

$\frac{\partial sum_{o_1}}{\partial w_5} = output_{h_1} = 0.60587$

$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial output_{o_1}} \times \frac{\partial output_{o_1}}{\partial sum_{o_1}} \times \frac{\partial sum_{o_1}}{\partial w_5}$

$= 0.768379 \times 0.16684 \times 0.60587$

$= 0.07767$

W_6

$\frac{\partial E_{total}}{\partial w_6} = \frac{\partial E_{total}}{\partial output_{o_1}} \times \frac{\partial output_{o_1}}{\partial sum_{o_1}} \times \frac{\partial sum_{o_1}}{\partial w_6}$

$= 0.768379 \times 0.16684 \times 0.63414$

$= 0.08129$

W_7

$\frac{\partial E_{total}}{\partial w_7} = \frac{\partial E_{total}}{\partial output_{o_2}} \times \frac{\partial output_{o_2}}{\partial sum_{o_2}} \times \frac{\partial sum_{o_2}}{\partial w_7}$

$= \cancel{0.768379 \times 0.16684 \times 0.634}$

$\frac{\partial E_{total}}{\partial output_{o_2}} = output_{o_2} - target_2$

$= 0.76796 - 0.85$

$= \cancel{0.76796}\ -0.08204$

$$\frac{\partial output_{o2}}{\partial sum_{o2}} = output_{o2}(1 - output_{o2})$$

$$= 0.76796(1 - 0.76796)$$

$$= 0.17819$$

$$\frac{\partial E_{total}}{\partial w_7} = \overset{-0.08204}{\cancel{0.26796}} \times 0.17819 \times 0.60587$$

$$= \cancel{0.0289} \quad \cancel{-0.000} \quad -0.00886$$

W8

$$= -0.08204 \times 0.17819 \times 0.63414$$

$$= -0.00927$$

New weights

$$\eta = 0.6$$

W5

$$new\_w_5 = w_5 - \eta \times \frac{\partial E_{total}}{\partial w_5}$$

$$= 0.8 - 0.6 \times 0.07767$$

$$= 0.753398$$

W6

$$new\_w_6 = w_6 - \eta \times \frac{\partial E_{total}}{\partial w_6}$$

$= 0.6 - 0.6 \times 0.08129$

$= \underline{0.551226}$

$\text{new } w_7 = w_7 - n \times \frac{\partial E_{total}}{\partial w_7}$

$= 0.5 - 0.6 \times -0.08204$

$= \underline{\underline{0.5492}}$

$\text{new\_}w_8 = w_8 - n \times \frac{\partial E_{total}}{\partial w_8}$

$= 0.7 - 0.6(\times -0.00927)$

$= \underline{\underline{0.7056}}$

<u>Weights in hidden layer</u> (w_1, w_2, w_3, w_4)

$$\frac{\partial E_1}{\partial w_1} = \frac{\partial E_1}{\partial output_1} \times \frac{\partial output_{o1}}{\partial sum_{o1}} \times \frac{\partial sum_{o1}}{\partial output_{h1}} \times \frac{\partial output_{h1}}{\partial sum_{h1}} \times \frac{\partial sum_{h1}}{\partial w_1}$$

$= 0.768379 \times 0.16684 \times 0.8 \times 0.23879 \times 0.2$

$= \underline{\underline{0.0048979}}$

$\frac{\partial E_2}{\partial w_1} = -0.08204 \times 0.178197 \times 0.5 \times 0.23879 \times 0.2$

$= \underline{\underline{-0.0003496}}$

6

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_1}{\partial w_1} + \frac{\partial E_2}{\partial w_1}$$

$$= 0.0048979 + (-0.0003\,4908)$$

$$= \underline{0.0045489}$$

$\boxed{n = 0.6}$

$$\text{new } w_1 = w_1 - n \times \frac{\partial E_{total}}{\partial w_1}$$

$$= 0.2 - 0.6 \times 0.0045489$$

$$= \underline{0.19727066}$$

$\underline{W_2}$

$$\frac{\partial E_1}{\partial w_2} \; 0.76838 \times 0.16684 \times 0.6 \times 0.232006 \times 0.4$$
$$= 0.007138166$$

$$\frac{\partial E_1}{\partial w_2} = -0.08204 \times 0.178197 \times 0.7 \times 0.232006 \times 0.4$$

$$= -0.000949693$$

$$\text{new } w_2 = w_2 - n \times \frac{\partial E_{total}}{\partial w_2} = 0.4 - 0.5 \times 0.006188473$$

$$= \underline{0.39691}$$

W_3 - Using same steps

$$\frac{\partial E_1}{\partial w_3} = 0.76838 \times 0.16684 \times 0.8 \times 0.23879 \times 0.1 =$$
$$= 0.002448963$$

$$\frac{\partial E_2}{\partial w_3} = -0.08204 \times 0.178197 \times 0.5 \times 0.23879 \times 0.1$$
$$= \cancel{0.0024}\ -0.00017456$$

$$\frac{\partial E_{total}}{\partial w_3} = \frac{\partial E_1}{\partial w_3} + \frac{\partial E_2}{\partial w_3} = 0.002448963 - 0.00017456$$
$$= 0.002274416$$

$$\cancel{ws}\ new\_w_3 = w_3 - n \times \frac{\partial E_{tot}}{\partial w_3}$$

$$= 0.1 - 0.5 \times 0.002274416$$
$$= 0.0098863$$

W_4

$$\frac{\partial E_1}{\partial w_4} = 0.76838 \times 0.16684 \times 0.6 \times 0.232006 \times 0.3$$
$$= 0.005353625$$

$$\frac{\partial E_2}{\partial w_4} = -0.08204 \times 0.178197 \times 0.7 \times 0.232006 \times 0.3$$
$$= -0.000712269$$

8

$$\frac{\partial E_{total}}{\partial w_4} = \frac{\partial E_1}{\partial w_4} + \frac{\partial E_2}{\partial w_4}$$

$$= 0.005353625 + -0.000712269$$

$$= \underline{0.00464135}$$

$$mew\_w4 = w_4 - n \times \frac{\partial E_{total}}{d w_4}$$

$$= 0.4 - 0.5 \times 0.004641355$$

$$= \underline{\underline{0.3976793224}}$$