Contents

Introduction

Machine Learning has been on a steady rise over the last few decades. Thanks to the availability of big data and computing becoming faster and cheaper, we are able to garner a large amount of information and form useful insights about them using Machine Learning algorithms. Machine Learning has been adopted to automate complex and cumbersome tasks in many industries. One of the main industries where ML has proved to be increasingly useful is the healthcare industry. With the digitization of patient records all over the world, we now have access to larger datasets than ever before. Several healthcare companies spend a lot of capital on big data analytics to develop better products/services and thus serve the market better.

Breast Cancer happens to be the most-common invasive cancer in women worldwide. BC affects about 12% of women worldwide. In most cases, the patient is diagnosed too late and either has to undergo complex surgeries or do not survive. With the advent of Machine Learning, we aim to build a model that will be able to classify a BC case as benign or malignant with at least >95% accuracy using certain parameters.

Classification is a kind of complex optimization problem. Many ML techniques have been applied by researchers in solving this classification problem. Scientists strive to find the best algorithm to achieve the most accurate classification result, however, data of variable quality will also influence the classification result. Further, the rarity of data will influence the number of algorithm applications as well. Overall, most ML techniques are first tested in open source databases. Over time, a benchmark dataset has arisen in the literature: Wisconsin breast cancer diagnosis (WBCD). There are also many other BC benchmark data sets, for instance Wisconsin Prognostic Breast Cancer Chemotherapy (WPBCC), Wisconsin Diagnostic Breast Cancer (WDBC) and so on. ML techniques that have been used on the WBCD database in BC diagnosis and prognosis show different levels of accuracy that ranged between 94.36% and 99.90%. Similarly, there are results with differently modified algorithms relating to BC databases. By using ML techniques to analyse the WBCD database, BC can be diagnosed accurately based on 9 attributes as can be seen from Table 1.

Table 1: The Wisconsin Breast Cancer Dataset (WBCD)

0	Sample code number	id number
1	Clump Thickness	1–10
2	Uniformity of Cell Size	1–10
3	Uniformity of Cell Shape	1–10
4	Marginal Adhesion	1–10
5	Single Epithelial Cell Size	1–10
6	Bare Nuclei	1–10
7	Bland Chromatin	1–10
8	Normal Nucleoli	1–10
9	Mitoses	1–10
10	Class	2- benign4-malignant

Methodology

We have implemented the project using a Support Vector Machine (SVM) and a K-Nearest Neighbors (KNN) classifier and compared the accuracy of the two.

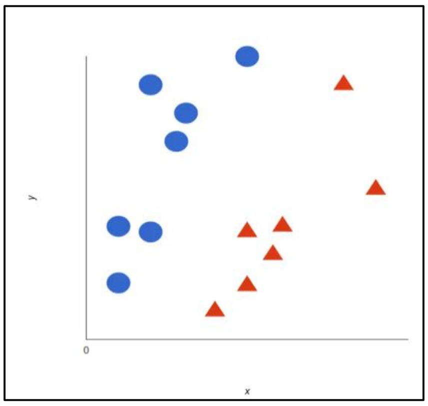
Support Vector Machines (SVMs)

SVMs are a supervised classification technique used extensively in Machine Learning. SVMs attempt to classify objects by drawing a hyperplane separating two classes. (A hyperplane is a figure having (n-1) dimensions in an n-dimensional ambient space. In the example provided in Fig 1, the input space is 2-dimensional while the hyperplane is a1-dimensional line. The line is plotted in a way that there is maximum separation between the extreme data points of both classes (also called support vectors).

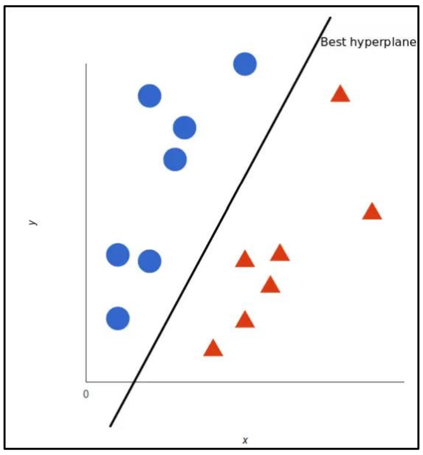
Definition: -

A support-vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outlier's detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class, since in general the larger the margin, the lower the generalization error of the classifier.

Working on a Linear data



Let's imagine we have two tags: *red* and *blue*, and our data has two features: *x* and *y*. We want a classifier that, given a pair of *(x, y)* coordinates, outputs if it's either *red* or *blue*. We plot our already labelled training data on a plane as shown in the graph.



SVM takes these data points and outputs the hyperplane (which in two dimensions it’s simply a line) that best separates the tags. This line is the decision boundary: anything that falls to one side of it we will classify as *blue*, and anything that falls to the other as *red*.

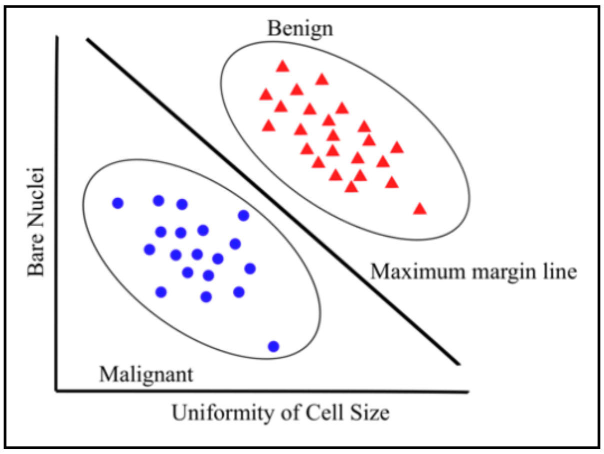


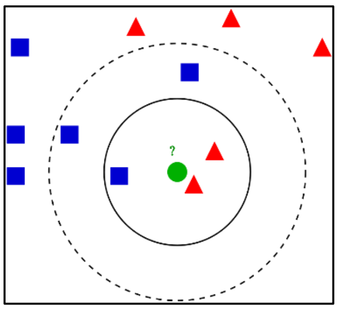
Fig 1. A simple example of how an SVM might work in distinguishing between benign and malignant tumour.

k-nearest Neighbours algorithm (KNN)

K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection. K nearest neighbours is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions).

Working

It works like this: we have an existing set of example data, our training set. We have labels for all of this data—we know what class each piece of the data should fall into. When we're given a new piece of data without a label, we compare that new piece of data to the existing data, every piece of existing data. We then take the most similar pieces of data (the nearest neighbours) and look at their labels. We look at the top k most similar pieces of data from our known dataset; this is where the *k* comes from. (*k* is an integer and it's usually less than 20.) Lastly, we take a majority vote from the k most similar pieces of data, and the majority is the new class we assign to the data we were asked to classify.



The test sample (green dot) should be classified either to blue squares or to red triangles. If *k* = *3* (solid line circle) it is assigned to the red triangles because there are 2 triangles and only 1 square inside the inner circle. If *k* = *5* (dotted line circle) it is assigned to the blue squares.

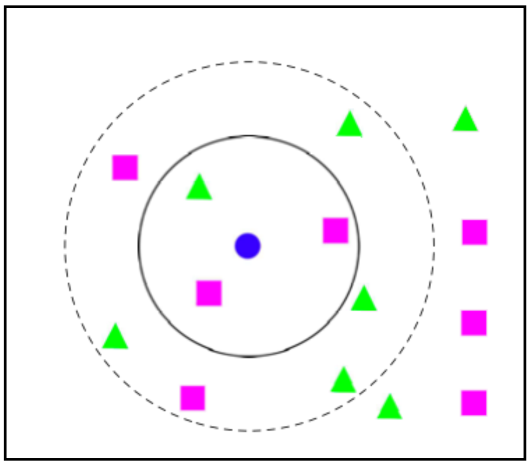


Fig 2. *k*-Nearest neighbour for breast cancer diagnosis. Blue circle means the test sample, green triangle means the malignant BC and pink square means the benign BC.

Applications

SVM

- Face Detection: - It classifies the parts of the image as face and non-face. It contains training data of n x n pixels with a two-class face (+1) and non-face (-1). Then it extracts features from each pixel as face or non-face. Creates a square boundary around faces on the basis of pixel brightness and classifies each image by using the same process.
- Classification of Images: - SVMs can classify images with higher search accuracy. Its accuracy is higher than traditional query-based refinement schemes.
- Handwriting Recognition: - We can also use SVMs to recognize hand-written characters that use for data entry and validating signatures on documents.

KNN

- Text Mining: - The KNN algorithm is one of the most popular algorithms for text categorization or text mining. Some of the most recent works on this topic are for instance. Different numbers of nearest neighbours are used for different classes in this approach, rather than a fixed number across all classes. In this way, the only parameter that needs to be chosen by the user when using KNN, the K value, becomes less sensible and hence it does not need to be carefully chosen as in the standard algorithm.
- Finance: - Data mining as a process of discovering useful patterns and correlations has its own niche in financial modelling. Similar to other computational methods almost every data mining method and technique has been used in financial modelling.

Medicine: - Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack. The prediction is to be based on demographic, diet and clinical measurements for that patient.

Implementation of the Project:

We have sourced the WBCD database directly from the UCI repository for datasets using the pandas module in Python. Further, we began preprocessing the data before fitting it to our models.

The following steps have been taken to preprocess the raw WBCD data:

1. Records with missing values have been omitted

2. The first column [‘id’] has been deleted as it provides us with no unique information.

Further, we analysed the preprocessed dataset to understand the distribution and correlations between the given parameters. This was achieved using the following steps:

1. The count, mean ,standard deviation, minimum ,maximum, 25,50 & 75 percentiles of each parameter was checked.

2. Histograms of each parameter were plotted

3. A scatter matrix of the entire dataset was plotted

Then, we began the creation of our testing and training datasets from the original dataset. We chose a train-test split ratio of 4:1 for this purpose.

Then, we defined our KNN and SVM models with the following attributes:

1. We set the value of n_neighbors = 5 for our K-Neighbors Classifier

2. We set the ‘gamma’ attribute of our Support Vector Classifier to ‘auto’

Then, we trained both our models in succession and cross-validated our results for greater accuracy. We chose 10 random splits of the training dataset and averaged out the predictions of all the splits.With both our models trained, we compared our models’ predictions on the testing dataset with the original values using scatter plots. Finally, we demonstrate both our models predicting outputs based on random dummy data.

Code for Mini-Project:

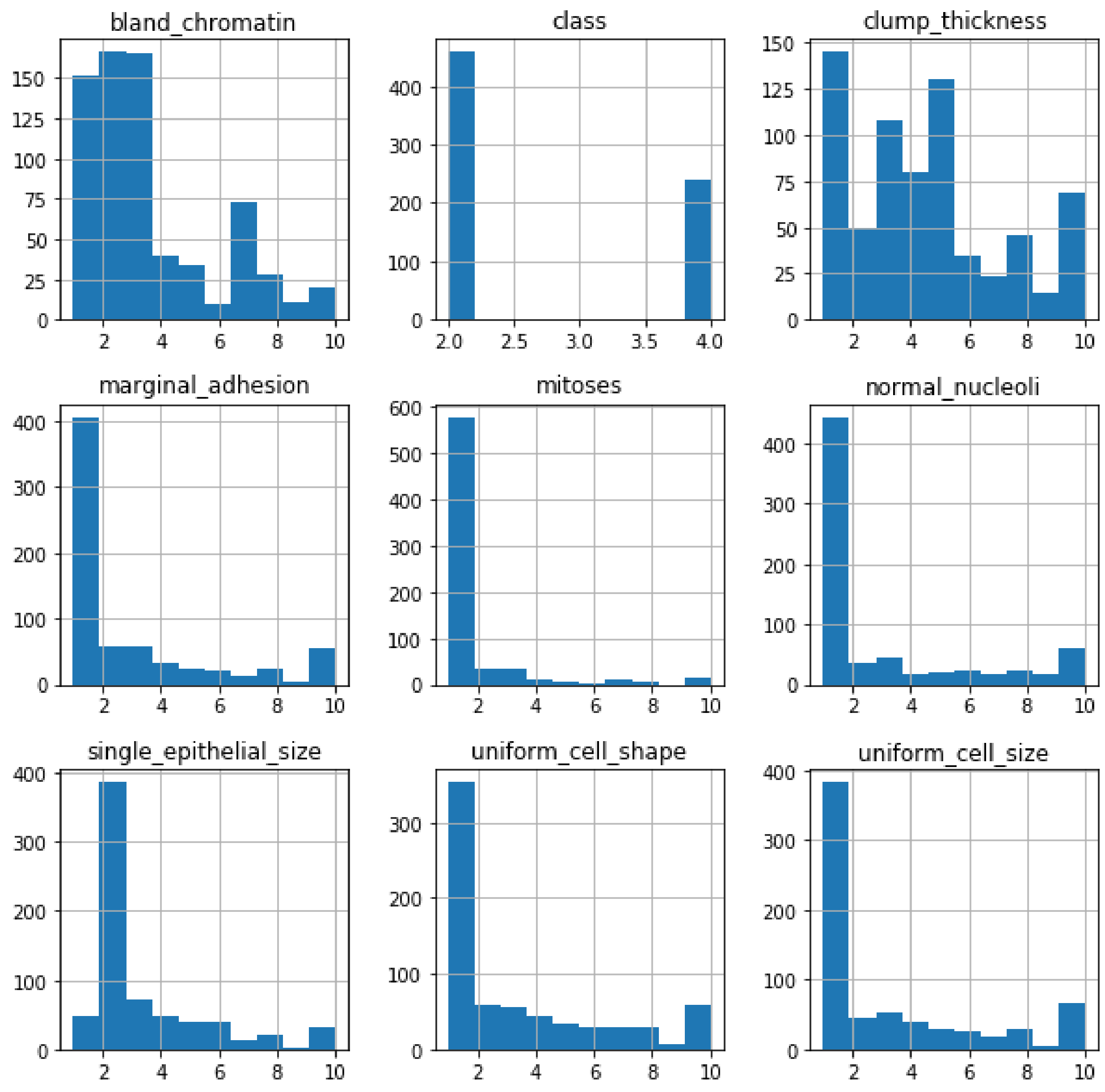
```
import sys
import numpy
import matplotlib
import pandas
import sklearn

print('Python: {}'.format(sys.version))
print('Numpy: {}'.format(numpy.__version__))
print('matplotlib: {}'.format(matplotlib.__version__))
print('pandas: {}'.format(pandas.__version__))
print('sklearn: {}'.format(sklearn.__version__))


import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn import model_selection
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,accuracy_score
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
import pandas as pd





#loading the dataset
url="https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data"
names=['id','clump_thickness','uniform_cell_size','uniform_cell_shape','marginal_adhesion','single_epithelial_size','bare_nuclei','bland_chromatin','normal_nucleoli','mitoses','class']
df=pd.read_csv(url,names=names)


# Preprocessing the data
df.replace('?',-99999,inplace=True) #replacing missing data
print(df.axes)

df.drop(['id'],1,inplace=True)
#print the shape of the dataset
print(df.shape)


#Do dataset visualizations
print(df.loc[0])  #first patient data
print(df.describe())  #pandas method to show numeric properties of dataframe
```

```
#Plot histogram for each variable
df.hist(figsize=(10,10))
plt.savefig('histogram')
plt.show()


#Create scatter plot matrix
scatter_matrix(df,figsize=(15,20))
plt.savefig('scatter.png')
plt.show()


#Create X and Y datasets for training
X= np.array(df.drop(['class'],1))
y=np.array(df['class'])

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2)
#same random-state gives same split
#specifying no random-state generates random value internally
#thus each time the code is run, the splits will be different


#specify testing options
seed = 0
scoring= "accuracy"


#defining the models to train
models=[]
models.append(('KNN',KNeighborsClassifier(n_neighbors = 5)))
models.append(('SVM',SVC(gamma='auto')))

#evaluate each model in turn

results=[]
names=[]

for name,model in models:
   kfold = model_selection.KFold(n_splits=10,random_state=seed)
   cv_results=model_selection.cross_val_score(model,X_train,y_train,cv=kfold,scoring=scoring)
   results.append(cv_results)
   names.append(name)
   msg="%s: min: %f max: %f avg: %f std: %f" % (name,cv_results.min(),cv_re-
sults.mean(),cv_results.std())
   print(msg) #prints min,ma & avg accuracy and standard deviation of each model


#make predictions on validation dataset
for name,model in models:
   model.fit(X_train,y_train)
   predictions=model.predict(X_test)
   print(name)
   print(accuracy_score(y_test,predictions))
   print(classification_report(y_test,predictions))
```

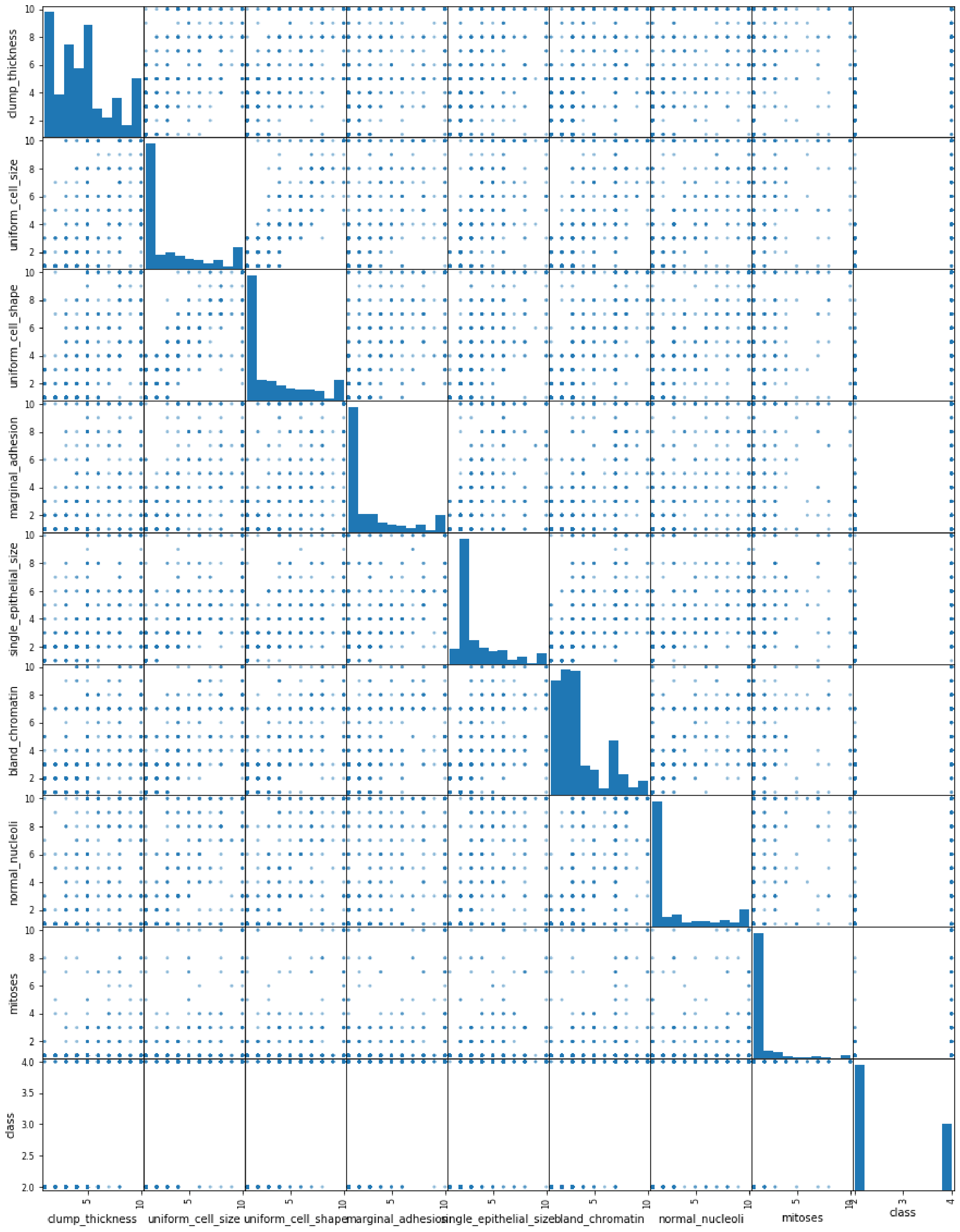
```
    plt.scatter(np.arange(len(y_test)),y_test,label='actual value')
    plt.scatter(np.arange(len(y_test)),predictions,label=name)

    plt.xlabel('Patient ID')
    plt.ylabel('Class')
    plt.legend()
    plt.figure(figsize=(20,20))
    plt.savefig('output')
    plt.show()


#Using SVC to predict dummy case
clf = SVC(gamma='auto')

clf.fit(X_train,y_train)
accuracy=clf.score(X_test,y_test)
print(accuracy)

example=np.array([[4,2,1,8,1,2,3,2,2]])
example=example.reshape(len(example),-1)
prediction = clf.predict(example)
print(prediction)


#Using KNN to predict dummy case
clf = KNeighborsClassifier(n_neighbors=5)
clf.fit(X_train,y_train)
accuracy=clf.score(X_test,y_test)
print(accuracy)
example=np.array([[4,2,1,6,1,2,3,2,5]])
example=example.reshape(len(example),-1)
prediction = clf.predict(example)
print(prediction)
```

Histogram :

Inference: All parameters have a lower value bias except clump_thickness which ap-



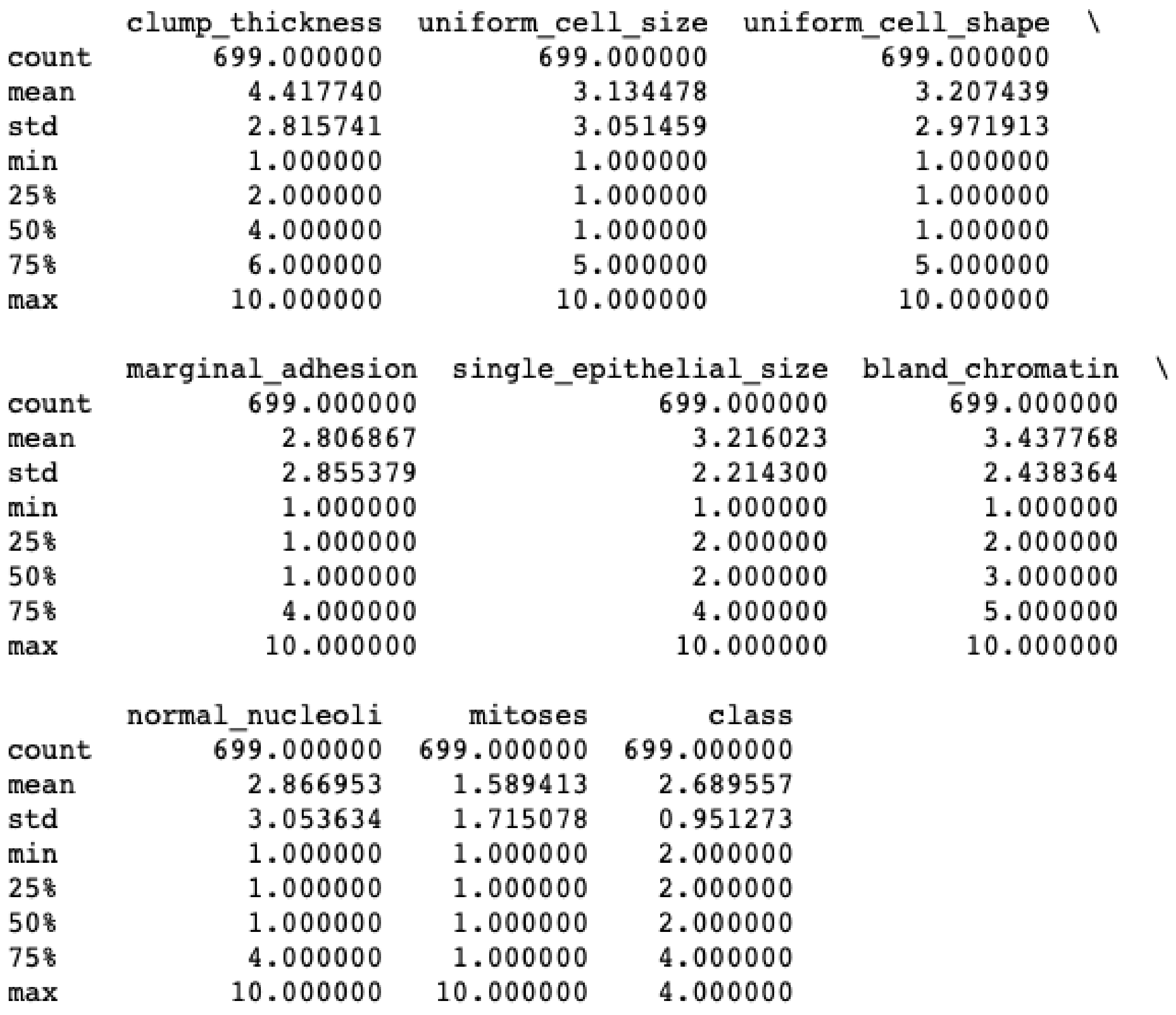
pears to be randomly distributed.



Scatter Matrix: Only cell size & cell shape have strong linear correlation.

Dataset statistical description:

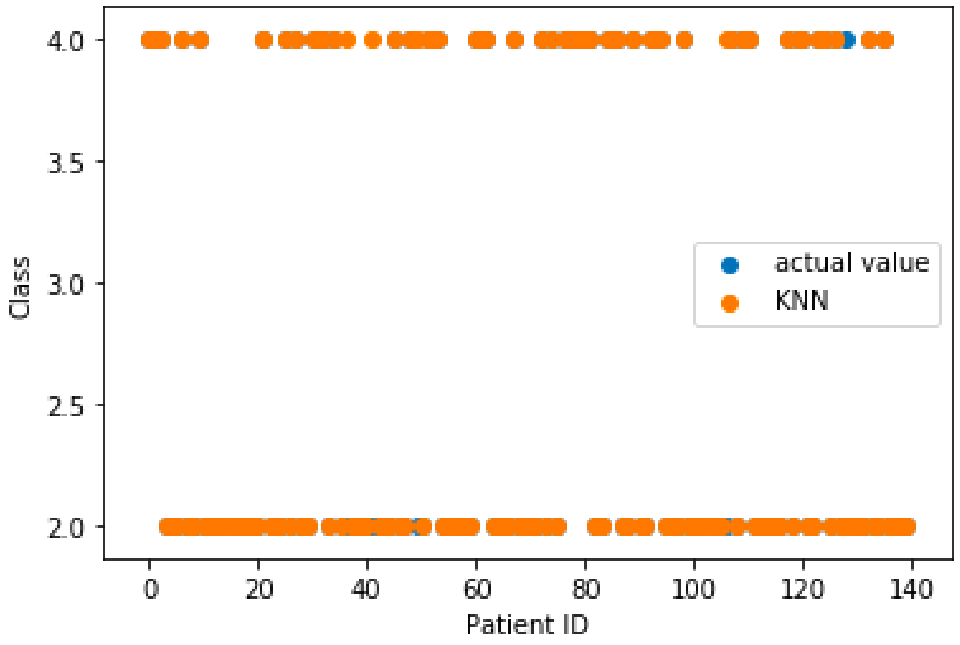
Total 699 records. All parameters have values ranging from 1-10.



```
       clump_thickness  uniform_cell_size  uniform_cell_shape  \
count       699.000000         699.000000          699.000000
mean          4.417740           3.134478            3.207439
std           2.815741           3.051459            2.971913
min           1.000000           1.000000            1.000000
25%           2.000000           1.000000            1.000000
50%           4.000000           1.000000            1.000000
75%           6.000000           5.000000            5.000000
max          10.000000          10.000000           10.000000

       marginal_adhesion  single_epithelial_size  bland_chromatin  \
count         699.000000              699.000000       699.000000
mean            2.806867                3.216023         3.437768
std             2.855379                2.214300         2.438364
min             1.000000                1.000000         1.000000
25%             1.000000                2.000000         2.000000
50%             1.000000                2.000000         3.000000
75%             4.000000                4.000000         5.000000
max            10.000000               10.000000        10.000000

       normal_nucleoli     mitoses       class
count       699.000000  699.000000  699.000000
mean          2.866953    1.589413    2.689557
std           3.053634    1.715078    0.951273
min           1.000000    1.000000    2.000000
25%           1.000000    1.000000    2.000000
50%           1.000000    1.000000    2.000000
75%           4.000000    1.000000    4.000000
max          10.000000   10.000000    4.000000
```

KNN Classifier Testing Result:

```
KNN
0.9642857142857143
              precision    recall  f1-score   support

           2       0.99      0.96      0.97        91
           4       0.92      0.98      0.95        49

    accuracy                           0.96       140
   macro avg       0.96      0.97      0.96       140
weighted avg       0.97      0.96      0.96       140
```



Our KNN model delivered predictions with approximately **96.43**% accuracy.

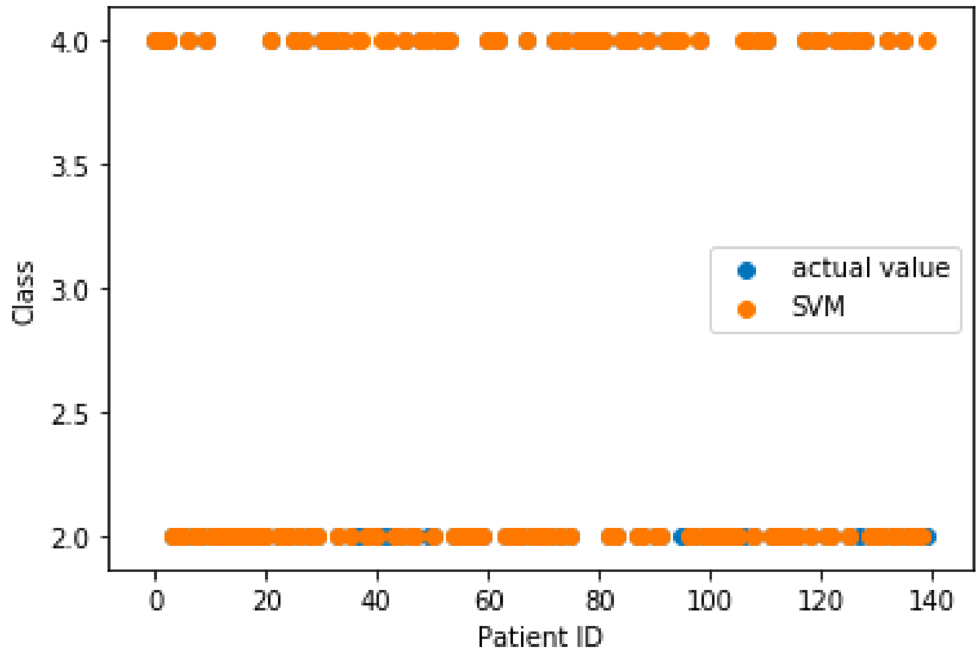
The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

F1-score gives us the harmonic mean of precision and recall value of each class.

'Support' gives us the count of each class. It is found that there are 91 benign cases and 49 malignant cases.

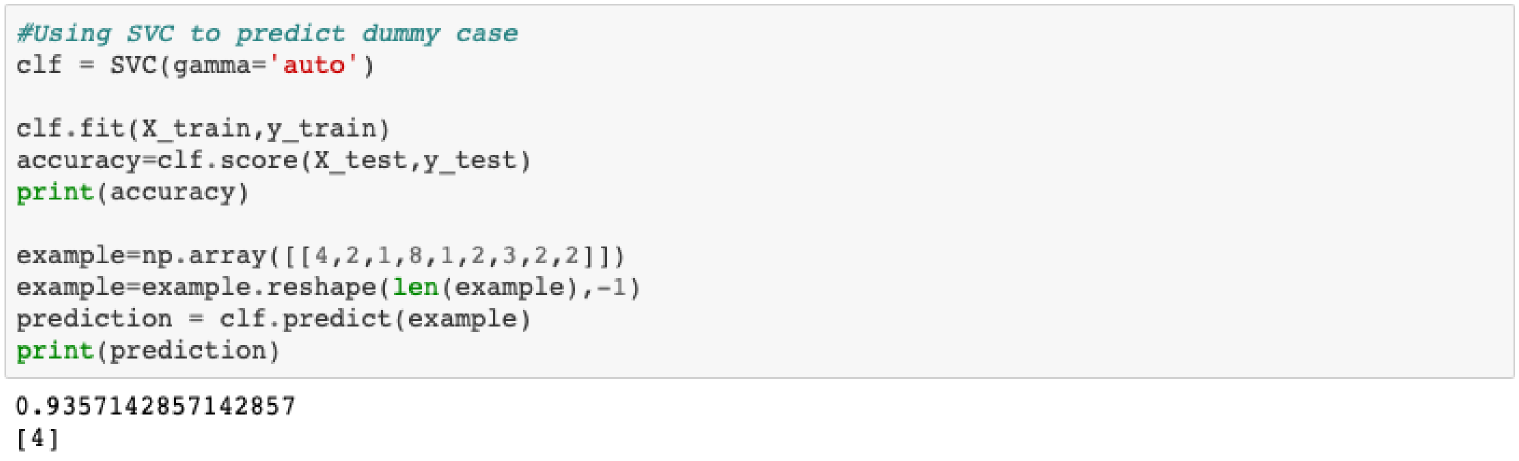
```
SVM
0.9357142857142857
              precision    recall  f1-score   support

           2       1.00      0.90      0.95        91
           4       0.84      1.00      0.92        49

    accuracy                           0.94       140
   macro avg       0.92      0.95      0.93       140
weighted avg       0.95      0.94      0.94       140
```



SVM Classifier Testing Result:

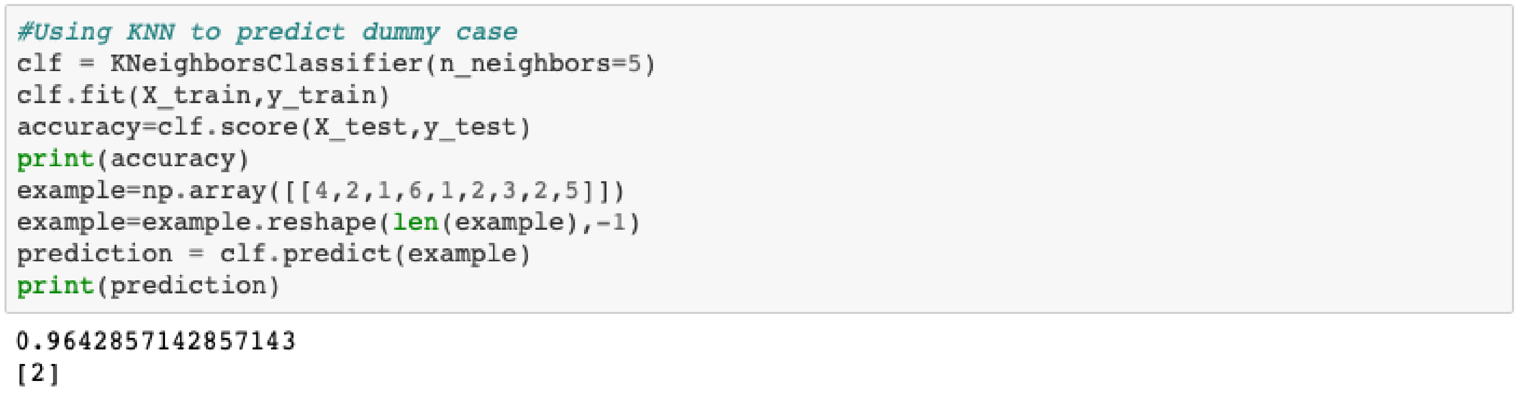
Our SVM model predicted results with approximately **93.57**% accuracy.

Testing models with dummy inputs:



```
#Using SVC to predict dummy case
clf = SVC(gamma='auto')

clf.fit(X_train,y_train)
accuracy=clf.score(X_test,y_test)
print(accuracy)

example=np.array([[4,2,1,8,1,2,3,2,2]])
example=example.reshape(len(example),-1)
prediction = clf.predict(example)
print(prediction)
```

```
0.9357142857142857
[4]
```



```
#Using KNN to predict dummy case
clf = KNeighborsClassifier(n_neighbors=5)
clf.fit(X_train,y_train)
accuracy=clf.score(X_test,y_test)
print(accuracy)
example=np.array([[4,2,1,6,1,2,3,2,5]])
example=example.reshape(len(example),-1)
prediction = clf.predict(example)
print(prediction)
```

```
0.9642857142857143
[2]
```

Conclusions & future enhancements

In this report, we have explored some ML techniques that can be used to detect breast cancer among women worldwide. **We found that the KNN and SVM models classified the WBCD with almost similar accuracy (with KNN being more accurate in this case).** ML techniques have shown their remarkable ability to improve classification and prediction accuracy. Although lots of algorithms have achieved very high accuracy in WBCD, the development of improved algorithms is still necessary. Classification accuracy is a very important assessment criteria but it is not the only one. Different algorithms consider different aspects, and have different mechanisms. Although for several decades ANNs have dominated BC diagnosis and prognosis, it is clear that more recently alternative ML methods have been applied to intelligent healthcare systems to provide a variety of options to physicians.

Moving ahead, we would like to explore other machine learning algorithms such as Decision Trees(DTs) , Artificial Neural Networks (ANNs) and also improve upon our basic SVM and KNN models to study other datasets in the healthcare industry- an industry where we believe lies a great potential of a quantum leap using the advancements of machine learning.

References

1. Andrew Ng's Coursera Course - Machine Learning
   https://www.coursera.org/learn/machine-learning?

2. Eduonix's course - Learn Machine Learning By Building Projects.
   https://www.eduonix.com/learn-machine-learning-by-building-projects

3. Sentdex's videos on Machine Learning on YouTube
   https://www.youtube.com/watch?v=OGxgnH8y2NM&list=PLQVvvaa0QuD-fKTOs3Keq_kaG2P55YRn5v

4. UCI Machine Learning Repository
   https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original)

5. Breast Cancer, Wikipedia
   https://en.wikipedia.org/wiki/Breast_cancer

6. Scikit Learn Online Documentation
   https://scikit-learn.org/stable/