PREDICTING BREAST CANCER VIA MULTIPLE SUPERVISED MACHINE LEARNING METHODS

Al Rahrooh

Medical Informatics
University of California, Los Angeles
Los Angeles, CA
Email Address: arahrooh@g.ucla.edu

ABSTRACT

Objective: To demonstrate the effectiveness of multiple supervised machine learning algorithms (Support Vector Machine, Logistic Regression, Kernel-SVM, Random Forest, K-nearest neighbors, Naïve Bayes, Decision Trees) in classifying the Wisconsin Breast Cancer dataset obtained from UCI [1]. This analysis aims to observe which machine learning technique is most successful in predicting malignant or benign cancer and to see general trends that may aid us in model selection and hyper parameter selection. The goal is to classify whether the breast cancer is benign or malignant based on the tumor's physical features.
Methods: Scikit-learn machine learning library for Python was used in Visual Studio Code to preprocess the data, then create the machine learning models, followed by evaluation with performance metrics, and analysis.
Results: The presented machine learning algorithms performed well, all exceeded 90% test accuracy, on the classification task. Random Forest allowed for the highest performance evaluation among the seven machine learning methods tested with an accuracy score of 98.92%, sensitivity of 99.74% and precision of 98.53%.
Conclusion: Multiple supervised machine learning classification algorithms are able to accurately predict benign or malignant classification based on the tumor's characteristics.

1. INTRODUCTION

A significant issue in the field of medical informatics and precision medicine is the accurate analysis of certain important diagnosable features. The diagnosis of a disease is a very difficult task in which the burden is placed on imaging and physician decision making. Because of this, there are circumstances of errors, unwanted biases, and also requires a long time for exact diagnosis of disease. However, there is a large amount of medical diagnosis data available in many diagnostic centers, hospitals, and research centers along with numerous open-source databases that can be utilized with the proper analytical approaches to aid in the medical decision-making process.

Breast cancer is the second leading cause of cancer deaths among U.S. women and represents 15% of all new cancer cases [2]. Screening mammography has been found to reduce mortality but is associated with a high risk of false positives as well as false negatives [3]. This procedure is also time-consuming, and in some worse cases, detects the disease with the wrong outcome. With the rapid population growth, the risk of death incurred by breast cancer is rising exponentially. The utilization of data science and machine learning approaches in medical fields proves to be prolific as such approaches may be considered of great assistance in the decision-making process of medical practitioners.

With an unfortunate increasing trend of breast cancer cases [3], along with the growth of computation in medicine we have seen an influx of data which is of significant use in furthering clinical and medical research. Even though it is evident that the use of machine learning methods can improve our understanding of cancer progression, an appropriate level of validation is needed in order for these methods to be considered in the everyday clinical practice. Such advances allow for the future goal of precision medicine and an automatic disease detection system which would aid medical staffs in disease diagnosis and offers reliable, effective, and rapid response as well as decreases the risk of death.

The involvement of data mining techniques in predicting breast cancer based on the patterns and relationships found among the breast cancer risk factors reduce diagnosis time by physicians and financial costs [2]. Thus, the survival rate for breast cancer can be increased immensely with diagnosis and treatment at an early stage [4]. As the chances of survival differ largely by breast cancer stages, the earliest diagnosis will improve the rate of survival greatly. Women who were diagnosed at the early, noninvasive stage will have better chances of survival than those diagnosed at the later invasive stages. It is crucial for clinicians to diagnose women who

have breast cancer accurately and prevent false positive results [4]. Therefore, the purpose of this study is to determine which predictive model of breast cancer occurrence for women by applying supervised machine learning on physical breast cancer risk factor and attribute data to provide a fair decision support system for medical practitioners to diagnose the incidence of breast cancer accurately and enhance the survivability rate of patients.

2. METHODS

The methodology deployed involves key processes such as the selection of target data, pre-processing the chosen data, transforming the data into a structured and comprehensible format, implementing supervised machine learning techniques and evaluating the machine learning performance using evaluation metrics. These steps ultimately lead to knowledge extraction from the target dataset where new insights and ideas can be developed to assist in enhancing business operations or in this case, aid in early diagnosis and prediction of diseases such as breast cancer.

2.1 Machine Learning Library

Scikit-learn machine learning library for Python was used in Visual Studio Code to implement the packages for each machine learning algorithm [5].

2.2 The Dataset

Data was obtained from the UCI Breast Cancer Wisconsin Diagnostic Data Set. The multivariate dataset consists of features which were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The features describe the characteristics of the cell nuclei found in the image. There are 569 data points with a class distribution: of 357 benign and 212 malignant. There are 10 tumor features: Radius, Texture, Perimeter, Area, Smoothness, Compactness, Concavity, Concave Points, Symmetry, Fractal Dimension. The mean, standard error and worst/largest (mean of the three largest values) of these features were computed for each image, resulting in 30 total potential features used for the modeling. The two target classes correspond to negative outcomes (Benign) and positive outcomes (Malignant). All feature values are used without manipulation and there are no missing attribute values. [1]

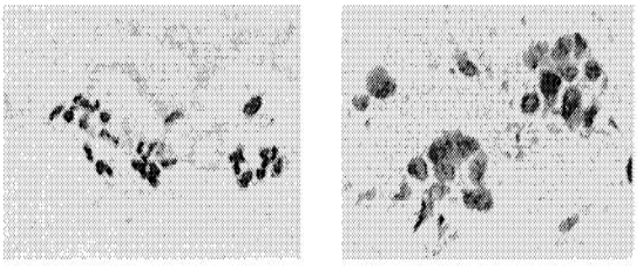

Figure 1. Computed from a digitized image of a fine needle aspirate of a tissue mass. Image on right is classified as benign. Image on left is classified as malignant.

2.2 Dataset Preprocessing

In the stage of pre-processing, it is essential to eliminate any missing values, noise and other anomalies in the selected data. Any inconsistency in the chosen data, especially disease related data may lead to unreliable results or misdiagnosis of test data, which could be fatal if the model is implemented in real-life situations [6]. However, the Wisconsin Breast Cancer Diagnostic Dataset contained no missing or null values.

The dataset was then split into features (X) and diagnosis (Y). The features dataset contained the 10 tumor features that influence the diagnosis of malignancy versus benign. X and Y is then split 80/20 into a training and testing phase. The training phase extracts the features from the dataset and the testing phase is used to determine how the appropriate model behaves for prediction.

To avoid inappropriate assignment of relevance, the dataset was standardized using the following equation.

$$z = \frac{X - \mu}{\sigma}$$

Where X is the feature to be standardized, μ is the mean value of the feature, and σ is the standard deviation of the feature.

2.3 Feature Correlation

The Pearson correlation coefficient was used to calculate the strength of the correlation between the feature variables that influence the cancer classification. The values were visualized within a matrix to explore the high dimensionality of the Wisconsin Breast Cancer dataset.

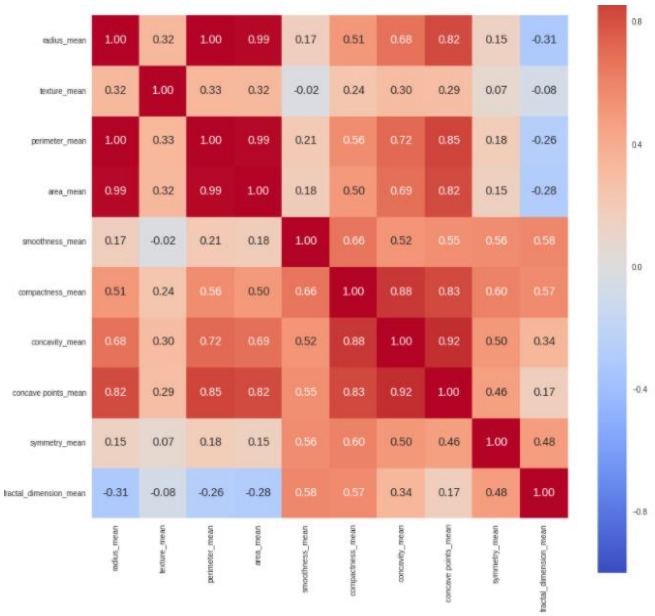

Figure 2. Pearson Correlation Heat map to determine feature reduction

2.4 Feature Reduction
There is a high correlation between features which are directly related to each other. The mean radius, mean perimeter, and mean area are highly correlated as expected in which mean radius will be used for modeling while mean perimeter and mean area will be removed from the features dataset. Compactness mean, concavity mean, and concave point mean are highly correlated so compactness mean will be used for modeling while the other two will be disregarded.

2.5 Modeling
Multiple machine learning models were used on the Wisconsin Breast Cancer dataset in order to compare the evaluation metrics among different algorithms.

Logistic Regression
Logistic Regression is an analytical modeling technique where the likelihood of an event is associated with a set of explicit variables. It is used for analyzing a dataset in which there are one or more independent variables that decide a result. The result is measured with a binary variable. It is applied to predict a binary result (Malignant/Benign) given a set of independent variables

Support Vector Machines
Support Vector Machines is a discriminative machine learning classification algorithm. It classifies data points into two classes at a time, using a decision boundary known as a hyperplane. The primary objective of the SVM classifier is finding the optimal separating hyperplane.

K-Nearest Neighbors
K-nearest neighbor algorithm is utilized for grouping and used in pattern recognition. It is widely used in predictive analysis. On the arrival of new data, the KNN algorithm identifies existing data points that are nearest to it. Any attributes that can differ on a large scale may have sufficient influence on the interval between data points. The feature vectors, as well as class labels, are stored in the training phase. KNN makes predictions using the training dataset directly. Predictions are made for a new instance (x) by searching through the entire training set for the K most similar instances (the neighbors) and summarizing the output variable for those K instances.

Kernel-SVM
Kernel-SVM builds upon SVM; however, when there are more and more dimensions, computations within that space become more and more difficult. The kernel trick is implemented to allows us to operate in the original feature space without computing the coordinates of the data in a higher dimensional space.

Naïve Bayes
Naive Bayes algorithm is based on Bayes theorem with an assumption of independence among features. Naïve bayes classifiers use the probabilities of certain events being true, given other events are true, in order to make predictions about new data points.

Decision Tree
A decision tree applies the reasoning approach to obtain solutions for a given problem. This data mining technique is very flexible and simple which makes it an attractive choice for applications in diverse fields. The tree-shaped structures in decision tree represent decision sets which are easy to interpret and understand for decision-makers to assess and choose the best course of action based on the risk and benefits for each possible outcome for distinct options

Random Forest
The Random Forest classification is an ensemble method that is composed of different decision trees, each with the same nodes, but using different data that leads to different leaves. It merges the decisions of multiple decision trees in order to find an answer, which represents the average of all these decision trees. Random Forest generates a forest of classification trees from a given dataset, rather than a single classification tree. Each of these trees produces a classification for a given set of attributes.

2.4 Evaluation
Each machine learning algorithm is evaluated on certain performance metrics to determine the success of each algorithm and if the hyperparameters need to be adjusted.

Confusion Matrix
The confusion matrix describes the performance of the multiple classification models on the set of test data for which the true values are known.

Accuracy
Classification accuracy method was used find the accuracy of the multiple models. It is the ratio of number of correct predictions to the total number of input samples.

Sensitivity
Sensitivity is also regarded as recall in which it is the rate of the perceived positive case with the total positive.

Precision
Precision is the fraction of relevant instances among all retrieved instances.

2.5 Analysis
Cross Validation is used to evaluate the multiple machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data. Model validation helps in ensuring that the model performs well on new data, and helps in selecting the best model, the parameters, and the accuracy metrics

3. RESULTS

All presented machine learning algorithms exhibited high performance on the binary classification of breast cancer, in which there was successful determination of whether there was a benign tumor or a malignant tumor.

The validation dataset which was used to validate the classifiers generated on the training set resulted in the construction of classifiers with similar values of evaluation metrics and there is no enormous difference between the classifiers from the training and validation set. This shows that all the classifiers have performed well upon the evaluation using the test set.

The confusion matrix of the used machine learning strategies provides the prediction outcome for LR, KNN, SVM, K-SVM, NB, DT, and RF. The confusion matrix alone isn't enough to properly compare the algorithms in their success for breast cancer classification. Therefore, accuracy, sensitivity, and precision is used for better comparison. The accuracy score shown in Figure 3 reveals RF as the most successful model with the highest accuracy and sensitivity scores of 98.92% and 99.74% respectively, along with the second highest precision score of 98.53%. NB performed the weakest with accuracy, sensitivity, and precision scores of 91.68%, 93.53%, and 92.44% respectively. KNN and SVM showed comparable sensitivity scores of 99.34% and 99.23%. The highest precision score of 98.73% was seen with KNN. Overall, each technique ranged from 91% - 99% for each performance evaluation metric.

The mean accuracy for the models after cross validation which demonstrated splitting the data, fitting a model, and computing the score five consecutive times, varied among the models. The accuracy of LR, KNN, SVM, K-SVM, and NB increased after cross validation. While the accuracy of DT and RF decreased after cross validation. The standard deviation for LR, KNN, SVM, K-SVM, and NB were analogous in which it ranged from 0.011 to 0.018. The standard deviation of DT registered the largest at 0.032 while RF had the lowest with a standard deviation of 0.009.

	Performance Evaluation				
Classifier	Accuracy	Sensitivity	Precision	Cross Validation	Standard Deviation
LR	95.86%	97.13%	96.65%	98.12%	0.016
KNN	95.12%	99.34%	98.73%	96.24%	0.011
SVM	97.23%	99.23%	96.65%	98.35%	0.015
K-SVM	93.42%	95.67%	94.98%	97.41%	0.017
NB	91.68%	93.53%	92.44%	94.62%	0.018
DT	94.72%	96.83%	92.76%	92.03%	0.032
RF	98.92%	99.74%	98.53%	95.77%	0.009

Figure 3. Performance evaluation metrics for the seven machine learning algorithms used in this study

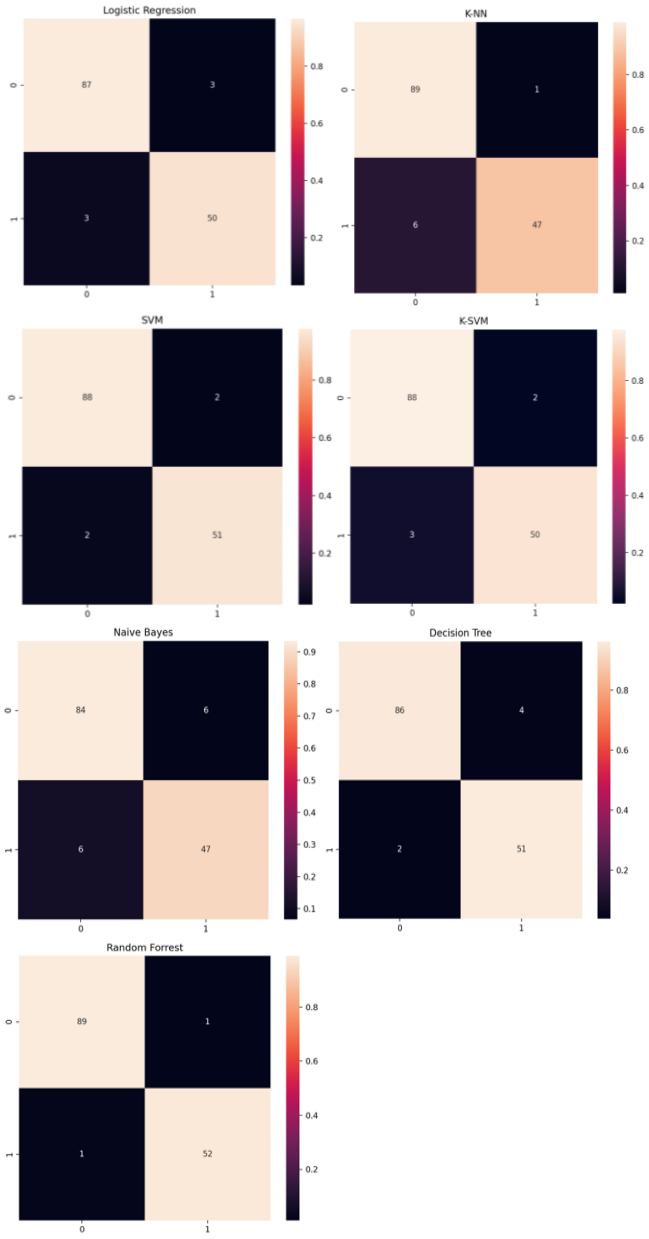

Figure 4 – 10. Confusion matrix for LR, KNN, SVM, K-SVM, NB, DT, and RF Machine Learning models

4. DISCUSSION

This study was conducted using the Wisconsin Breast Cancer dataset from the UCI machine learning database which consisted of 569 data points with a class distribution of 357 benign and 212 malignant.

4.1 Predictive Value of Breast Cancer Attributes

For an ideal breast cancer prediction model, a greater True Positive rate indicates that cancer patients are predicted correctly to have cancer. A higher True Negative rate is also

preferred but this measure does not carry as much importance as True Positive rate. A prediction model needs to detect the presence of a disease correctly and prevent any misdiagnosis. The misleading results due to False Negative rate and False Positive rate can be fatal to patients as cancer is a lethal disease and the earlier the diagnosis, the better are the chances of survival. The lesser the False Positive rate and False Negative rate, and the higher the True Positive rate and True Negative rate, the better is the performance of the classification model. This is some general criteria for a disease prediction model, but this may vary depending on the dataset and the type of classifiers. The seven techniques used in this study were able to model this concept by allowing for successful classification with minimal error.

By completing feature reduction through comparing the Pearson correlation coefficients we were able to reduce unnecessary features that would have decreased training speed, decreased model interpretability, and, most importantly, decreased generalization performance on the test set. The Wisconsin Breast Cancer dataset from UCI had many features that were directly correlated and unneeded for further modeling. These features were those that were closely related within the field of physical attributes that depended on each other for the given values.

In general, each model was successful in classifying benign and malignant tumors. There weren't any discrepancies among the evaluation metrics that would flag the particular machine learning method as invalid compared to the others in the pipeline.

Precision medicine and precision imaging necessitate the development of interactive clinical decision support systems that assist physicians with decision making at the point of care. Increasing utilization of these systems will be catalyzed by the implementation, as well as the limitations, of electronic health records. Given the volume and complexity of data that are now available and the expectation of exponential increases in available data in the future, decision support tools designed to intelligently filter patient data are already essential for optimizing clinical workflow.

5. CONCLUSIONS

This study presented a comparative study of seven machine learning methods to predict breast cancer, namely Logistic Regression, K-Nearest Neighbors, Support Vector Machines, Kernel Support Vector Machines, Naïve Bayes, Decisions Trees, and Random Forest. The basic features and working principles of each of the machine learning techniques were illustrated alongside their performance evaluation. For a predictive model to be utilized in the clinical setting, the performance must be exceptional. Wisconsin Breast Cancer dataset provides the optimal situation in which machine learning can be successful. The data was clean and the classification was supervised. Real time clinical settings are never as straight forward; however, with further exploration of these machine learning focused methods, possibly the combination of methods, show the possibility of precision medicine.

Regarding the diagnosing of an individual patient case imaging alongside machine learning classification allows for precision medicine to reach its full potential. [8] The future of medicine is in early diagnosis and individually tailored treatments, a concept that has been designated as precision medicine, which relates to delivering the right treatment to the right patient at the right time.

Further works can involve feature selection on the dataset and segmentation of the variables with similar characteristics. As it is not guaranteed that conclusions from this study could be generalized to other mammography datasets with different properties, it would also be interesting to apply this methodology on other data with features that aren't directly related to a tumor.

6. REFERENCES

[1] http://archive.ics.uci.edu/ml/datasets/Breast+Cance

[2] Konstantina Kourou, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, Dimitrios I. Fotiadis, Machine learning applications in cancer prognosis and prediction, Computational and Structural Biotechnology Journal, Volume 13, 2015, Pages 8-17, ISSN 2001-0370, https://doi.org/10.1016/j.csbj.2014.11.005.

[3] Islam, M.M., Haque, M.R., Iqbal, H. et al. Breast Cancer Prediction: A Comparative Study Using Machine Learning Techniques. SN COMPUT. SCI. 1, 290 (2020). https://doi.org/10.1007/s42979-020-00305-w

[4] DeSantis, C.E., Ma, J., Gaudet, M.M., Newman, L.A., Miller, K.D., Goding Sauer, A., Jemal, A. and Siegel, R.L. (2019), Breast cancer statistics, 2019. CA A Cancer J Clin, 69: 438-451. https://doi.org/10.3322/caac.21583

[5] https://scikit-learn.org/stable/

[7] A. Osareh and B. Shadgar, "Machine learning techniques to diagnose breast cancer," 2010 5th International Symposium on Health Informatics and Bioinformatics, Ankara, Turkey, 2010, pp. 114-120, doi: 10.1109/HIBIT.2010.5478895.

[8] M. Amrane, S. Oukid, I. Gagaoua and T. Ensarİ, "Breast cancer classification using machine learning," 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT), Istanbul, Turkey, 2018, pp. 1-4, doi: 10.1109/EBBT.2018.8391453.

Supplementary Material

Python Code

```
#Dependencies
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn import metrics
import seaborn as sns
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report


###PREPROCESSING###


#importing the dataset
dataset = pd.read_csv("data.csv", header = 0)
dataset.drop("Unnamed: 32", axis = 1, inplace = True)
dataset.drop("id",axis=1,inplace=True)
dataset['diagnosis']=dataset['diagnosis'].map({'M':1,'B':0})
Y = dataset['diagnosis'].values
X = dataset.drop('diagnosis', axis=1).values

#Analysis
corr = dataset[X].corr()
plt.figure(figsize=(14,14))
sns.heatmap(dataset[X].corr() , cbar = True,  square = True, annot=True, fmt= '.2f',annot_kws={'size': 15}, xticklabels= X,
yticklabels= X, cmap= 'coolwarm')
plt.show()


# Splitting the dataset into the Training set and Test set
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20, random_state = 0)

#Visualization of data
dataset.groupby('diagnosis').hist(figsize=(12, 12))
dataset.isnull().sum()
dataset.isna().sum()
dataframe = pd.DataFrame(Y)
```

```
#Feature Reduction
s_corr_target = XY_abs_corr['diagnosis']
s_corr_target_sort = s_corr_target.sort_values(ascending=False)

s_low_correlation_ftrs = s_corr_target_sort[s_corr_target_sort <= CORRELATION_MIN]

prediction_var = ['texture_mean','perimeter_mean','smoothness_mean','compactness_mean','symmetry_mean']

#Feature Scaling
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)



####MODELING###


#Logistic Regression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, Y_train)

Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('Logistic Regression')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

log_reg_accuracy = metrics.accuracy_score(predictions, Y_test)
log_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)

print(classification_report(prediction, Y_test)

#Fitting K-NN Algorithm
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
```

```
    plt.title('K-NN')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

KNN_accuracy_score = metrics.accuracy_score(predictions, Y_test)
KNN_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)


#Fitting SVM
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('SVM')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

SVM_accuracy_score = metrics.accuracy_score(predictions, Y_test)
SVM_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)


#Fitting K-SVM
classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('K-SVM')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')
```

```
plot_confusion_matrix(cm)
#plt.show()

K_SVM_accuracy_score = metrics.accuracy_score(predictions, Y_test)
K_SVM_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)


#Fitting Naive_Bayes
classifier = GaussianNB()
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('Naive Bayes')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

Naive_Bayes_accuracy_score = metrics.accuracy_score(predictions, Y_test)
NB_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)


#Fitting Decision Tree Algorithm
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('Decision Tree')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

Decision_tree_accuracy_score = metrics.accuracy_score(predictions, Y_test)
DT_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)
```

```
#Fitting Random Forest Classification Algorithm
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
predictions = [round(value) for value in Y_pred]
Y_test = np.squeeze(Y_test)

cm = confusion_matrix(Y_test, predictions)

def plot_confusion_matrix(cm, normalized=True, cmap='bone'):
    plt.figure(figsize=[7, 6])
    plt.title('Random Forrest')
    norm_cm = cm
    if normalized:
        norm_cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        sns.heatmap(norm_cm, annot=cm, fmt='g')

plot_confusion_matrix(cm)
plt.show()

Random_Forrest_accuracy_score = metrics.accuracy_score(predictions, Y_test)
RF_reg_scores = cross_val_score(classifier, X_train, Y_train, cv = 5)
print(classification_report(prediction, Y_test)


####EVALUATION####


#Accuracy Scores
print('Logistic Regression Accuracy Score = ' , log_reg_accuracy)
print('SVM Accuracy Score = ', SVM_accuracy_score)
print('K-NN Accuracy Score = ', KNN_accuracy_score)
print('K-SVM Accuracy Score = ', K_SVM_accuracy_score)
print('Naive Bayes Accuracy Score = ', Naive_Bayes_accuracy_score)
print('Decision Tree Accuracy Score = ', Decision_tree_accuracy_score)
print('Random Forrest Accuracy Score = ', Random_Forrest_accuracy_score)


#Cross Validation
print(log_reg_scores.mean(), "accuracy with a standard devciation of ", log_reg_scores.std())
print(KNN_reg_scores.mean(), "accuracy with a standard devciation of ", KNN_reg_scores.std())
print(SVM_reg_scores.mean(),"accuracy with a standard devciation of ", SVM_reg_scores.std())
print(K_SVM_reg_scores.mean(), "accuracy with a standard devciation of ", K_SVM_reg_scores.std())
print(NB_reg_scores.mean(), "accuracy with a standard devciation of ", NB_reg_scores.std())
print(DT_reg_scores.mean(), "accuracy with a standard devciation of ", DT_reg_scores.std())
print(RF_reg_scores.mean(), "accuracy with a standard devciation of ", RF_reg_scores.std())
```