MANB 2153 ASSIGNMENT 1 (INDIVIDUAL)

Performance Comparison of Machine Learning Classifiers on Vertebral Column Diagnosis

---

MANB2153

MACHINE LEARNING FOR BUSINESS PROBLEM

Prepared By:

LIM YEE FANG (MAN161019)

Prepared For:

DR. NILAM NUR BINTI AMIR SJARIF

Table of Contents

1.0 Introduction

This report compares a total of three classifiers, namely K-Nearest Neighbors (KNN), Decision Tree and Support Vector Machines (SVM) with machine learning approaches on vertebral column dataset. Vertebral column dataset was retrieved from UCI Machine Learning Repository at: http://archive.ics.uci.edu/ml/datasets/Vertebral+Column . The dataset has 310 instances and 6 attributes. The 7 attributes include 1 class attribute and another 6 biomechanical attributes derived from the shape and orientation of the pelvis and lumbar spine, namely pelvic incidence, pelvic tilt, lumbar lordosis angle, sacral slope, pelvic radius and degree spondylolisthesis. The class attribute classifies orthopedic patients into 3 classes which are normal, hernia or spondylolisthesis. The dataset consist medical records of 310 which categorized into groups of disk hernia (60 patients), spondylolisthesis (150 patients) and normal (100 patients). The best performed classifier which produces optimal result of vertebral column disorders classification is identify. The classification performance for each classifier is evaluate based on performance measurement such as accuracy, precision, recall and confusion matrix.

2.0 Objective

The objectives of this comparative study are as follow:

i. To study vertebral column disorders classification.
ii. To analyze vertebral column disorders classification based on selected classifiers.
iii. To evaluate the performance of classifier by using evaluation metrics such as accuracy, precision, recall and confusion matrix.

3.0 Scope

The scope of data only used column_3C_weka.cvs of vertebral column dataset. Tools used to perform classification in this comparative study is python and jupyter notebook. The comparative study tests only selective classifiers which are KNN, Decision Tree and SVM. The performance of classifiers are evaluate by using evaluation metrics such as accuracy, precision, recall and confusion matrix.

4.0 Import Libraries and Dataset

```
import pandas as pd
import numpy as np
from matplotlib import style
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display, HTML
```

```
data = pd.read_csv('column_3C_weka.csv')
data.head(3)
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis	class
0	63.027818	22.552586	39.609117	40.475232	98.672917	-0.254400	Hernia
1	39.056951	10.060991	25.015378	28.995960	114.405425	4.564259	Hernia
2	68.832021	22.218482	50.092194	46.613539	105.985135	-3.530317	Hernia

Figure 1: A quick glance at dataset

First and foremost, import the necessary libraries and read the column_3C_weka.cvs file. The first three rows are displayed for quick glance of the dataset.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 310 entries, 0 to 309
Data columns (total 7 columns):
pelvic_incidence            310 non-null float64
pelvic_tilt                 310 non-null float64
lumbar_lordosis_angle       310 non-null float64
sacral_slope                310 non-null float64
pelvic_radius               310 non-null float64
degree_spondylolisthesis    310 non-null float64
class                       310 non-null object
dtypes: float64(6), object(1)
memory usage: 17.0+ KB
```

Figure 2: Data information

The basic information of dataset is showed in Figure 2. The dataset has a total of 310 entries with 7 attributes.

5.0 Data Pre-processing

```
#Check missing value/null
tab_info=pd.DataFrame(data.dtypes).T.rename(index={0:'column type'})
tab_info=tab_info.append(pd.DataFrame(data.isnull().sum()).T.rename(index={0:'null values (nb)'}))
tab_info=tab_info.append(pd.DataFrame(data.isnull().sum()/data.shape[0]*100).T.rename(index={0:'null values (%)'}))
display(tab_info)
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis	class
column type	float64	float64	float64	float64	float64	float64	object
null values (nb)	0	0	0	0	0	0	0
null values (%)	0	0	0	0	0	0	0

```
#Check duplicate instances
data.duplicated().sum()
```

```
0
```

Figure 3: Checking missing value or null and duplicate instance

The dataset is checked for missing values or null and duplicate instances before proceed to classification. There is no need for data cleaning since the dataset is free from missing values and duplicate instances.

6.0 Data Exploration

	count	mean	std	min	25%	50%	75%	max
pelvic_incidence	310.0	60.496653	17.236520	26.147921	46.430294	58.691038	72.877696	129.834041
pelvic_tilt	310.0	17.542822	10.008330	-6.554948	10.667069	16.357689	22.120395	49.431864
lumbar_lordosis_angle	310.0	51.930930	18.554064	14.000000	37.000000	49.562398	63.000000	125.742385
sacral_slope	310.0	42.953831	13.423102	13.366931	33.347122	42.404912	52.695888	121.429566
pelvic_radius	310.0	117.920655	13.317377	70.082575	110.709196	118.268178	125.467674	163.071041
degree_spondylolisthesis	310.0	26.296694	37.559027	-11.058179	1.603727	11.767934	41.287352	418.543082

Figure 4: Summary of statistic of the dataset

Figure 4 reviews the summary of the vertebral column dataset statistics.

```
data['class'].unique()
```

```
array(['Hernia', 'Spondylolisthesis', 'Normal'], dtype=object)
```

```
pd.value_counts(data["class"])
```

```
Spondylolisthesis    150
Normal               100
Hernia                60
Name: class, dtype: int64
```

Figure 5: Class labels and count

Check the class label and count. The class attribute categories orthopedic patients into 3 classes which are hernia, spondylolisthesis and normal. The dataset consists 60 records of hernia, 150 records of spondylolisthesis and 100 records of normal.

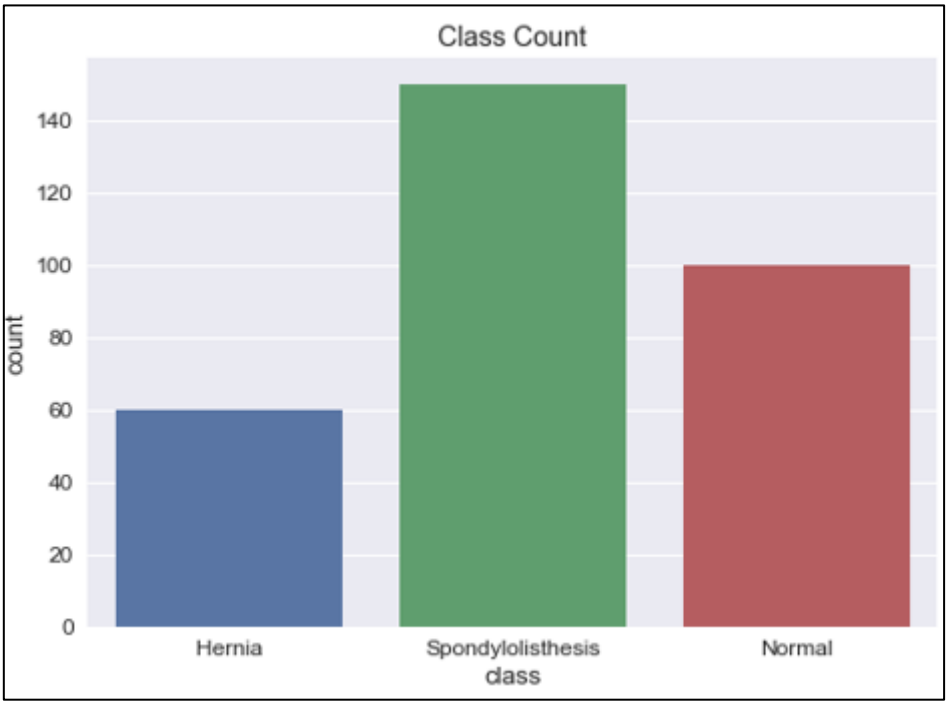

Figure 6: Countplot of each class label

The class count is plot according to the class label: Hernia, Spondylolisthesis and Normal.

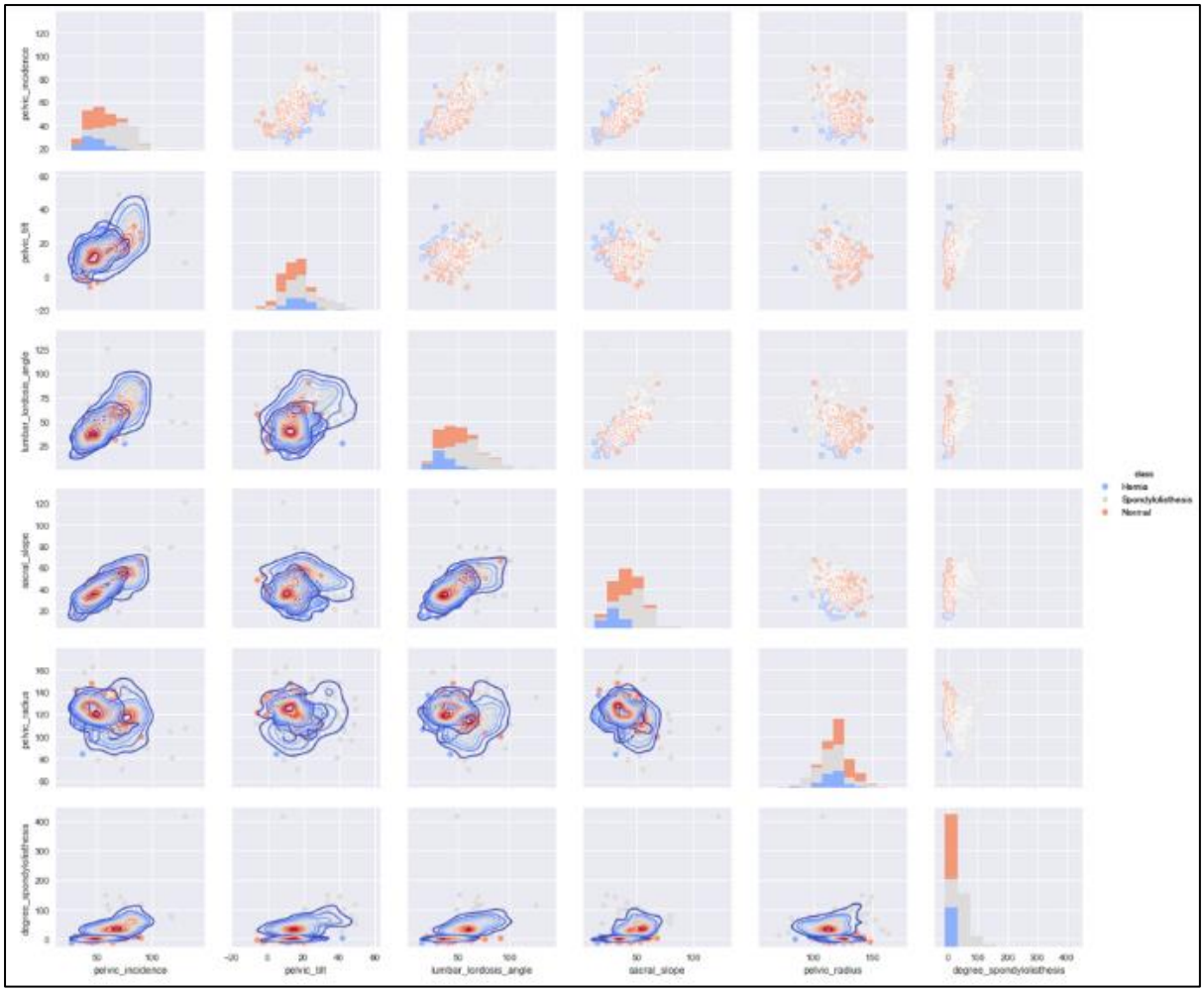

Figure 7: Data distribution with scatter plot, histogram and kernel density estimation

The data distributions are observed by plotting the scatter plot, histogram and kernel density estimation.

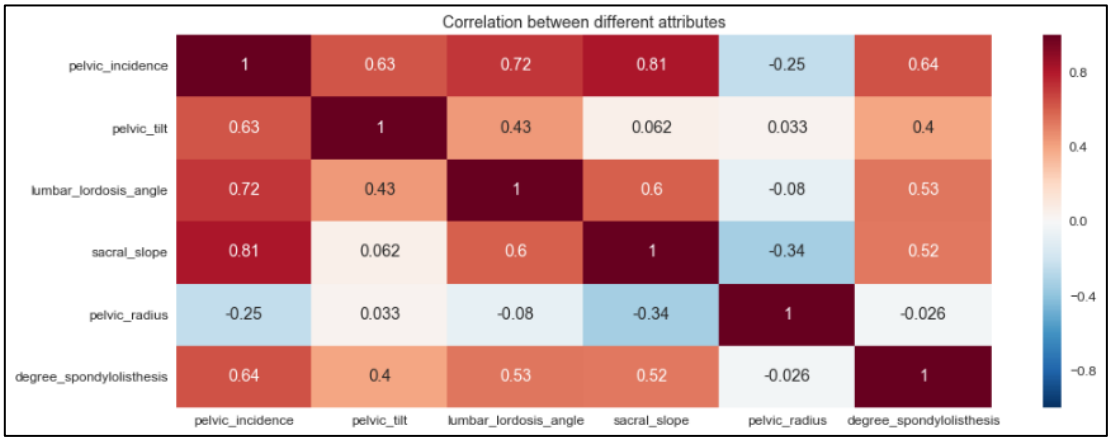

Figure 8: Correlation between different attributes

The correlation of attributes are analysed. All attributes are included in modelling as the vertebral column diagnostics required to access every changes and symptoms carefully since many types of vertebral column disorders shared similar symptoms.

7.0 Modelling

```
# train test split
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = 1)
```

Figure 9: Training and Testing set split

The experiment adopted common proportion of split which is 70% for training set and 30% for testing set. The split ratio of 30% testing set was indicated in parameter : test_size = 0.3. Training set is used to fit the classifier, training set is used to make prediction.

7.1 K-Nearest Neighbors (KNN)

```
# KNN
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3)
x,y = data.loc[:,data.columns != 'class'], data.loc[:,'class']
knn.fit(x,y)
knnprediction = knn.predict(x)
print('Accuracy of KNN with (K=3) is: ',knn.score(x,y))
```

```
('Accuracy of KNN with (K=3) is: ', 0.91935483870967738)
```

```
knn = KNeighborsClassifier(n_neighbors = 3)
x,y = data.loc[:,data.columns != 'class'], data.loc[:,'class']
knn.fit(x_train,y_train)
knnprediction = knn.predict(x_test)
print('Accuracy of KNN with (K=3) is: ',knn.score(x_train,y_train))
print('Accuracy of KNN with (K=3) is: ',knn.score(x_test,y_test))
```

```
('Accuracy of KNN with (K=3) is: ', 0.89861751152073732)
('Accuracy of KNN with (K=3) is: ', 0.83870967741935487)
```

Figure 10: Fit data into KNN classifier

Accuracy of KNN classifier on training set is 89.86%. Accuracy of KNN classifier on testing set is 83.87%.

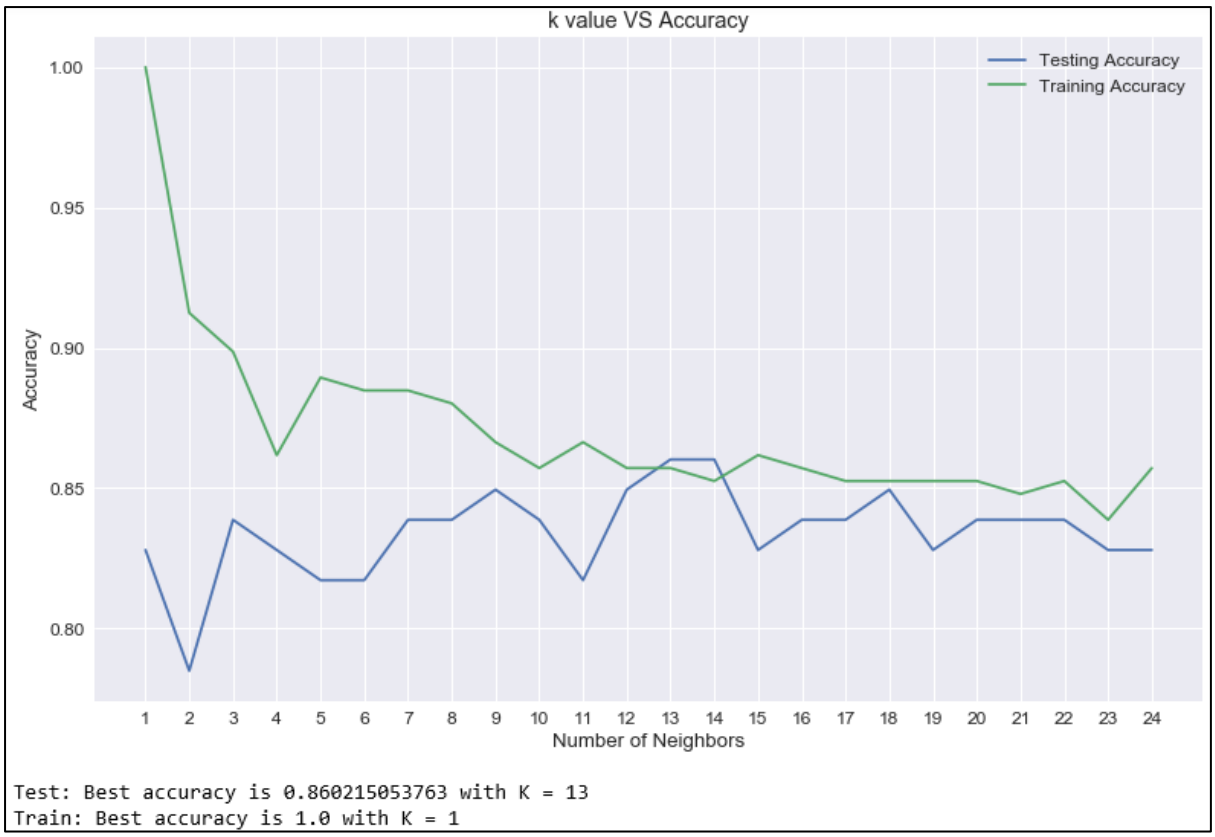

Figure 11: Model Complexity

Model complexity is run to select k (hyperparameter) that presents optimal performance. Lower k can lead to overfitting. From Figure 11, the KNN classifiers performs high accuracy classification with k=13.

7.2 Decision Tree

```
#Decision Tree
from sklearn import tree
tree = tree.DecisionTreeClassifier()
tree = tree.fit(x, y)
treeprediction = tree.predict(x)
print('Accuracy of Decision Tree is: ',tree.score(x,y))
```

```
('Accuracy of Decision Tree is: ', 1.0)
```

```
tree.fit(x_train,y_train)
treeprediction = tree.predict(x_test)
print('Accuracy of Decision Tree is: ',knn.score(x_train,y_train))
print('Accuracy of Decision Tree is: ',knn.score(x_test,y_test))
```

```
('Accuracy of Decision Tree is: ', 0.8571428571428571)
('Accuracy of Decision Tree is: ', 0.82795698924731187)
```

Figure 12: Fit data into Decision Tree classifier

Accuracy of Decision Tree classifier on training set is 85.71%. Accuracy of Decision Tree classifier on testing set is 82.80%.

7.3 Support Vector Machines (SVM)

```
# SVM, pre-process and pipeline
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
steps = [('scalar', StandardScaler()),
         ('SVM', SVC())]
pipeline = Pipeline(steps)
parameters = {'SVM__C':[1, 10, 100],
              'SVM__gamma':[0.1, 0.01]}
cv = GridSearchCV(pipeline,param_grid=parameters,cv=3)

cv.fit(x, y)
svmprediction = cv.predict(x)

cv.fit(x_train,y_train)
svmprediction = cv.predict(x_test)

print("Accuracy: {}".format(cv.score(x_train,y_train)))
print("Accuracy: {}".format(cv.score(x_test, y_test)))
print("Tuned Model Parameters: {}".format(cv.best_params_))
```

```
Accuracy: 0.884792626728
Accuracy: 0.849462365591
Tuned Model Parameters: {'SVM__C': 100, 'SVM__gamma': 0.01}
```

Figure 13: Fit data into SVM classifier

Accuracy of SVM classifier on training set is 88.48%. Accuracy of SVM classifier on testing set is 84.95%.

8.0 Ten Fold Cross Validation

```
KNN:
[ 0.77419355  0.67741935  0.90322581  0.70967742  0.93548387  0.93548387
  0.93548387  0.87096774  0.90322581  0.87096774]
('Accuracy of KNN is: ', 85.161290322580641)
Decision Tree:
[ 0.70967742  0.80645161  0.83870968  0.74193548  0.83870968  0.87096774
  0.83870968  0.80645161  0.74193548  0.74193548]
('Accuracy of Decision Tree is: ', 78.709677419354847)
SVM:
[ 0.77419355  0.67741935  0.90322581  0.74193548  0.93548387  0.93548387
  0.87096774  0.87096774  0.90322581  0.87096774]
('Accuracy of SVM is: ', 84.838709677419345)
```

Figure 14: 10 fold cross validation of all classifiers

The 10 fold cross validation of KNN has accuracy 85.16%. The 10 fold cross validation of Decision Tree has accuracy 78.71%. The 10 fold cross validation of SVN has accuracy 84.84%.

9.0 Confusion Matrix

Confusion matrices are plotted for all classifiers on their performance of classification. The correctly classified instances and incorrectly classified instances are derived from the confusion matrix results in which the correctly classified instances are sum of instances that fall in true positive and true negative matrices whereas the incorrectly classified instances are sum of instances that fall in false positive and false negative matrices.

9.1 K-Nearest Neighbors (KNN)

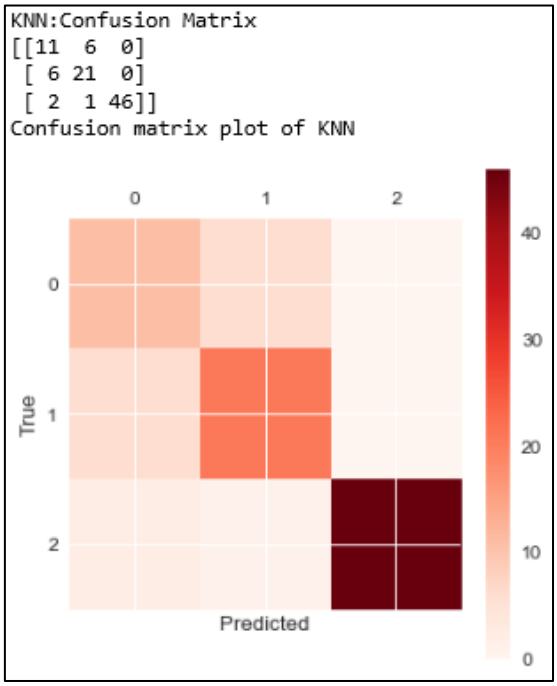

Figure 15: Confusion matrix of KNN

KNN:-

Correctly classified instances: 11+21+46=78

Incorrectly classified instances: 6+0+6+0+2+1=15

9.2 Decision Tree

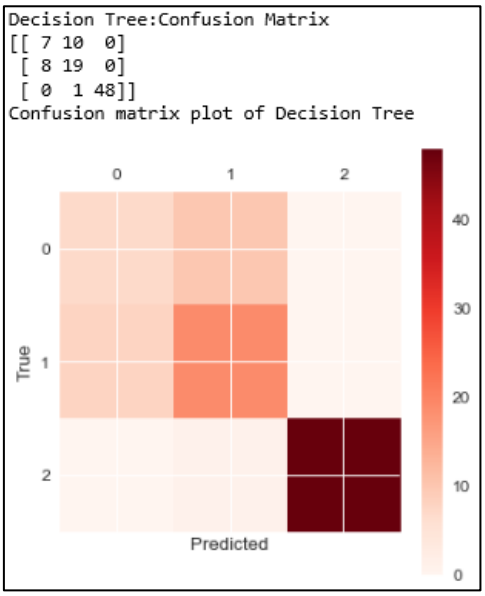

Figure 16: Confusion matrix of Decision Tree

Decision Tree:-

Correctly classified instances: 7+19+48=74

Incorrectly classified instances: 10+0+8+0+0+1=19

9.3 Support Vector Machines (SVM)

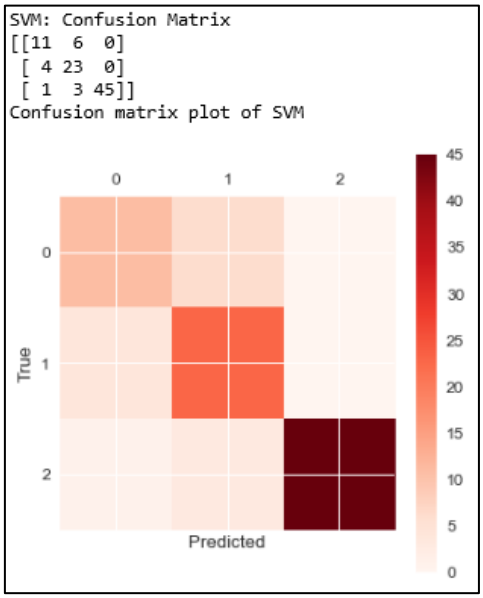

Figure 17: Confusion matrix of SVM

SVM:-

Correctly classified instances: 11+23+45=79

Incorrectly classified instances: 6+0+4+0+1+3=14

10.0 Classification Report

Classification report shows scores of precision, recall, f1-score and support of all classes.

10.1 K-Nearest Neighbors (KNN)

Classification Report

	precision	recall	f1-score	support
Hernia	0.58	0.65	0.61	17
Normal	0.75	0.78	0.76	27
Spondylolisthesis	1.00	0.94	0.97	49
avg / total	0.85	0.84	0.84	93

Figure 18: KNN classification report

The test prediction of KNN classifier has average total of 0.85 precision, 0.84 recall, 0.84 f1-score and 93 support.

10.2 Decision Tree

Classification Report

	precision	recall	f1-score	support
Hernia	0.47	0.41	0.44	17
Normal	0.63	0.70	0.67	27
Spondylolisthesis	1.00	0.98	0.99	49
avg / total	0.80	0.80	0.79	93

Figure 19: Decision Tree classification report

The test prediction of Decision Tree classifier has average total of 0.80 precision, 0.80 recall, 0.79 f1-score and 93 support.

10.3 Support Vector Machines (SVM)

Classification Report

	precision	recall	f1-score	support
Hernia	0.69	0.65	0.67	17
Normal	0.72	0.85	0.78	27
Spondylolisthesis	1.00	0.92	0.96	49
avg / total	0.86	0.85	0.85	93

Figure 20: SVM classification report

The test prediction of SVM classifier has average total of 0.86 precision, 0.85 recall, 0.85 f1-score and 93 support.

11.0 Results and Discussions

Tables and figures below show the classification performance of KNN, Decision Tree and SVM classifiers on vertebral column dataset. The performance evaluation metrics include accuracy, precision, recall, f1-score, support and confusion matrix.

Table 1: Classification Performance of KNN, Decision Tree and SVM Classifiers on Vertebral Column Dataset

Classifiers	Accuracy (%)	Precision	Recall	F1-score	Support
KNN	83.87	0.85	0.84	0.84	93
Decision Tree	82.80	0.80	0.80	0.79	93
SVM	84.95	0.86	0.85	0.85	93

Table 2: Results of 10 Fold Cross Validation

Number of Folds	Classifier Results (Accuracy %)		
	KNN	Decision Tree	SVM
1	77.42	70.97	77.42
2	67.74	80.64	67.74
3	90.32	83.87	90.32
4	70.97	74.19	74.19
5	93.55	83.87	93.55
6	93.55	87.10	93.55

7	93.55	83.87	87.10
8	87.10	80.65	87.10
9	90.32	74.19	90.32
10	87.10	74.19	87.10
Total Average	85.16	78.71	84.84

Table 3: Results of Confusion Matrix on Test Dataset

Classifiers	Correctly classified instances	Incorrectly classified instances
KNN	78	15
Decision Tree	74	19
SVM	79	8

The support is the number of samples of the true response that lie in that class. From the results, there are 93 samples of true response that lie in the class. The precision is the ability of the classifier to misclassify a positive sample that is negative. The recall is the ability of the classifier to find all the positive samples. Higher precision and recall scores indicate the less the classifier to misclassify a sample. In medical diagnostic, high classification precision and recall is necessary to avoid mistreatment of patient. The f1-score provides the harmonic mean of precision and recall. From Table 1, SVM displays high percentage of classification accuracy which is 84.95% follows by KNN with 83.87% accuracy and Decision Tree with 82.80% accuracy. Meanwhile, the SVM scores of precision, recall and f1-score are also high ranked among the classifiers. This implies that the SVM has better classification performance in vertebral column disorders classification compare to KNN and Decision Tree classifiers.

The results of 10 fold cross validation in Table 2 reveals the influence of smoothing effect on KNN classifiers. KNN classifier exhibits better accuracy in cross validation. The improvement of accuracy in KNN might induce by the smoothing nature of the classifier along with the increase of k value. On contrary, Decision Tree show significant drop in accuracy. The contradiction of accuracy displayed in Table 1 and Table 2 indicates the performance of the KNN and Decision Tree classifiers are prone to overfitting problem. Apparently, SVM that shows consistency in classification accuracy is able to produce more generalized results.

Furthermore, it is preferable if the number of correctly classified instance is higher in classification performance assessment. From Table 3, comparison of the number of correctly classified instances between classifiers shows that SVM has highest number of correctly classified instances which is 79 instances, followed by KNN with 78 instances and Decision Tree with 74 instances. Decision Tree has more incorrectly classified instances than KNN and SVM.

The overall performance evaluation denotes SVM has more precise and better classification capability than KNN and Decision Tree in vertebral column disorders diagnosis. SVM classifier able to generate a more generalized results and exhibits high the accuracy rate of classification with most of the instances correctly classified.

12.0 Conclusions

Integration of machine learning in medical diagnostics contributes to the diagnostics errors reduction and better health care performance. This report demonstrates performance comparison of KNN, Decision Tree and SVM classifiers in machine learning context on vertebral column dataset. The results of the experiment demonstrate SVM classification capability that yield high accuracy of 84.95%. The classification results are able to produce the proper and correct classification of different types of vertebral column disorder which shared almost similar symptoms. High quality classification performance improve diagnostic quality and reduce the medical complications and consequences resulted from misdiagnosis. Further studies are recommended to use different classifiers and pre-processing techniques to classify medical dataset.