Classification-Shizhi Chen-10307389

1. Description of the classification method

K-Means Clustering (Unsupervised)

Unsupervised classification refers to the classification of data without any labels, the algorithm can cluster unlabeled data directly by the data's features and their correlation. The K-means algorithm is a popular unsupervised clustering method, which is an appropriate tool under the assumption that the clusters are hyper spheroidal since Euclidean distance measures are employed (Clausi, 2012).

The K-Mean algorithm takes k as a parameter and divides n objects x_1, …, x_n into k (≤ n) clusters C_1, …, C_k, which makes the clusters have higher inner similarity and the similarity between clusters is lower. The way to judge the performance of K-Mean clustering is when the total within-cluster sum of squares is minimized:

$$\arg\min_C \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \boldsymbol{\mu}_i||_2$$

where $\boldsymbol{\mu_i}$ = mean ({x $\in$ C_i}).

K-Nearest Neighbor (Supervised)

The rule of the KNN algorithm is to assume that the samples in a sample space are divided into several types and then given a data to be classified, by calculating the K samples closest to the data, it is determined which classification the data to be classified belongs to.

The steps of the KNN algorithm can be described as:

1. Calculate the distance (Usually use the Euclidean distance) between the sample data and the data to be classified;

Euclidean Distance: $d_{12} = \sqrt{\sum_{k=1}^{n} (x_{1k} - x_{2k})^2}$

2. Select K samples with the smallest distance for the data to be classified;

3. Count the classifications of most of the K samples.

According to this rule, an unclassified sample is assigned to the class represented by a majority of its k nearest neighbours in the training set (Denoeux,1995).

2. Exploratory analysis of the data and Pre-processing

In this classification task, Vertebral Column dataset (Dua, Karra, 2017) were used which including 310 patients and each patient is represented in the data set by six biomechanical attributes derived from the shape and orientation of the pelvis and lumbar spine. The following convention is used for the class labels: Normal (NO) and Abnormal (AB).

First, import some useful packages and load the dataset.

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import StandardScaler
```

```
df = pd.read_table('vertebral_column_data.txt',header=None,delim_whitespace=True)
df.head()
```

	0	1	2	3	4	5	6
0	63.03	22.55	39.61	40.48	98.67	-0.25	AB
1	39.06	10.06	25.02	29.00	114.41	4.56	AB
2	68.83	22.22	50.09	46.61	105.99	-3.53	AB
3	69.30	24.65	44.31	44.64	101.87	11.21	AB
4	49.71	9.65	28.32	40.06	108.17	7.92	AB

Rename the columns name for a better explanation before EDA.

```
df.columns = ['pelvic_incidence','pelvic_tilt','lumbar_lordosis_angle',
              'sacral_slope','pelvic_radius','spondylolisthesis_grade','label']
df.head()
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	spondylolisthesis_grade	label
0	63.03	22.55	39.61	40.48	98.67	-0.25	AB
1	39.06	10.06	25.02	29.00	114.41	4.56	AB
2	68.83	22.22	50.09	46.61	105.99	-3.53	AB

Then, boxplot can be used to detect the relationship between label and each biomechanical attribute

```
plt.style.use('ggplot')
fig = plt.figure(figsize=(15,10))
for i in range(1,7):
    axi = fig.add_subplot(2,3,i)
    axi = sns.boxplot(x="label", y=df.columns[i-1], data = df)
```

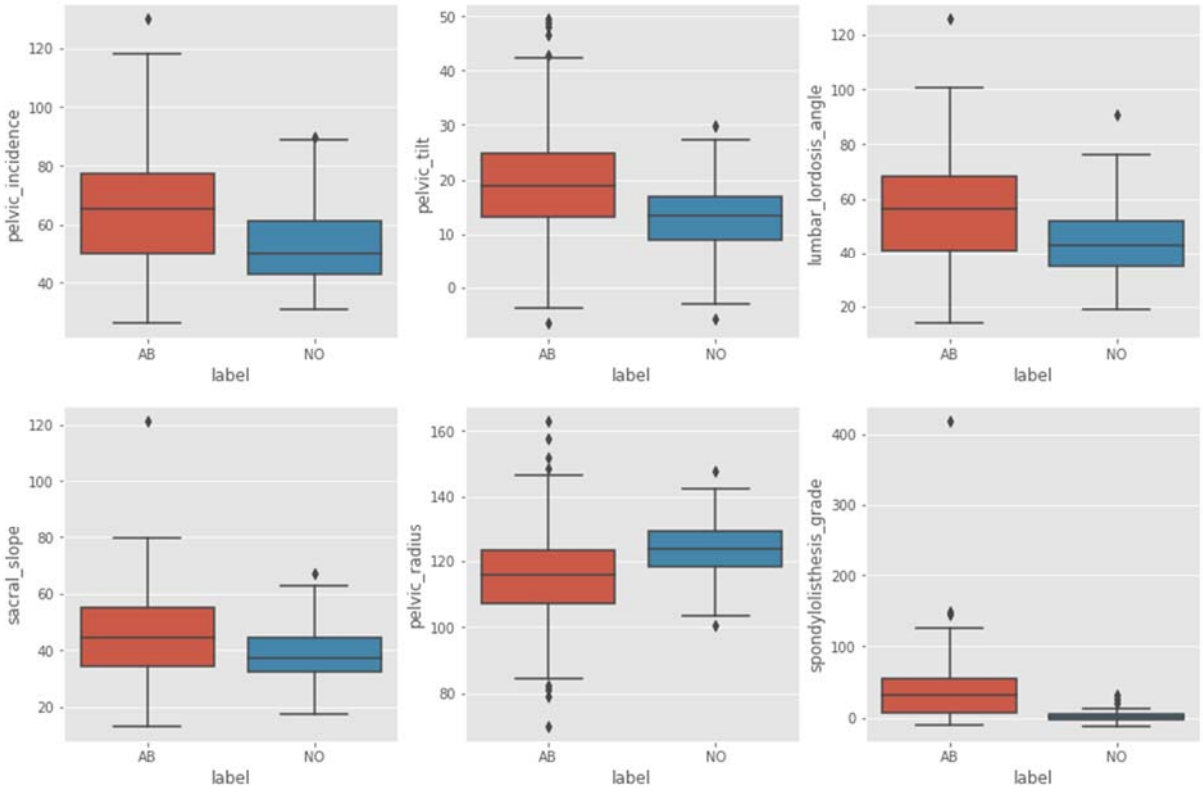

Figure 1: Boxplot between the label and biomechanical attribute

As can be seen from the figure, each attribute has a significant difference between the abnormal class and the normal class. Higher pelvic incidence, pelvic tilt, lumbar lordosis angle, sacral slope and grade of spondylolisthesis tend to higher probability in abnormal class while higher pelvic radius tends to have a higher probability in the normal class.

These six biomechanical attributes are in different scale range as such the standardization is needed before the classification algorithm applied.

```
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df.iloc[:,0:6])
```

After standardization, the average value of each column dimension is 0, and the variance is 1.

Also, encoding the label to 0 and1 instead of ‘AB’ and ’NO’.

```
from sklearn.preprocessing import LabelEncoder
labelencoder_X = LabelEncoder()
df['label'] = labelencoder_X.fit_transform(df['label'])
```

The data set after processing and transformations as shown:

```
df.iloc[:,0:6] = scaled_data
df.head()
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	spondylolisthesis_grade	label
0	0.147227	0.501111	-0.665128	-0.184602	-1.447831	-0.707946	0
1	-1.245707	-0.748891	-1.452763	-1.041250	-0.264028	-0.579673	0
2	0.484273	0.468085	-0.099370	0.272823	-0.897295	-0.795417	0
3	0.511586	0.711280	-0.411401	0.125820	-1.207159	-0.402332	0
4	-0.626819	-0.789923	-1.274614	-0.215943	-0.733337	-0.490069	0

```
df.tail()
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	spondylolisthesis_grade	label
305	-0.732001	-0.392605	-0.860012	-0.646505	-0.035390	-0.814618	1
306	-0.381007	0.317965	-1.226028	-0.726350	-0.267036	-0.712480	1
307	0.055410	0.515123	-0.310989	-0.313696	0.582835	-0.773549	1
308	-0.885997	-0.886000	-0.558778	-0.477116	0.047341	-0.695679	1
309	-1.549049	-1.248291	-0.825462	-1.058412	0.453474	-0.706613	1

3. Results of the analyses

K-Means Clustering (Unsupervised)

In the unsupervised method, the labels are unknown so only the six attributes data were used in the algorithm. The data set is expected to be classified into normal and abnormal categories so the K-Mean clusters are set to 2.

```
from sklearn.cluster import KMeans
km = KMeans(n_clusters = 2)
# Fitting Model
km.fit(X)
# Prediction on the entire data
km_predictions = km.predict(X)
print(km_predictions)
```

```
[1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 1 0 0 0 1
 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0
 0 1 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1
 0 0 0 1 1 1 1 0 0 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 1 1 1 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1
 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
 1 1 1 0 1 0 1 1 1 1 1 1 1 1]
```

Then, visualize the clustering results in two dimensions between each biomechanical feature by pair plot and colouring the figures based on a categorical prediction.

```
new_df = df
new_df['label'] = predictions
```

```
sns.pairplot(new_df,vars=new_df.columns[0:6], hue="label")
plt.show()
```

In the figure below, red colour dots are abnormal patients and the blue dots are normal patients. The figure shows a good clustering result except for some outliers.

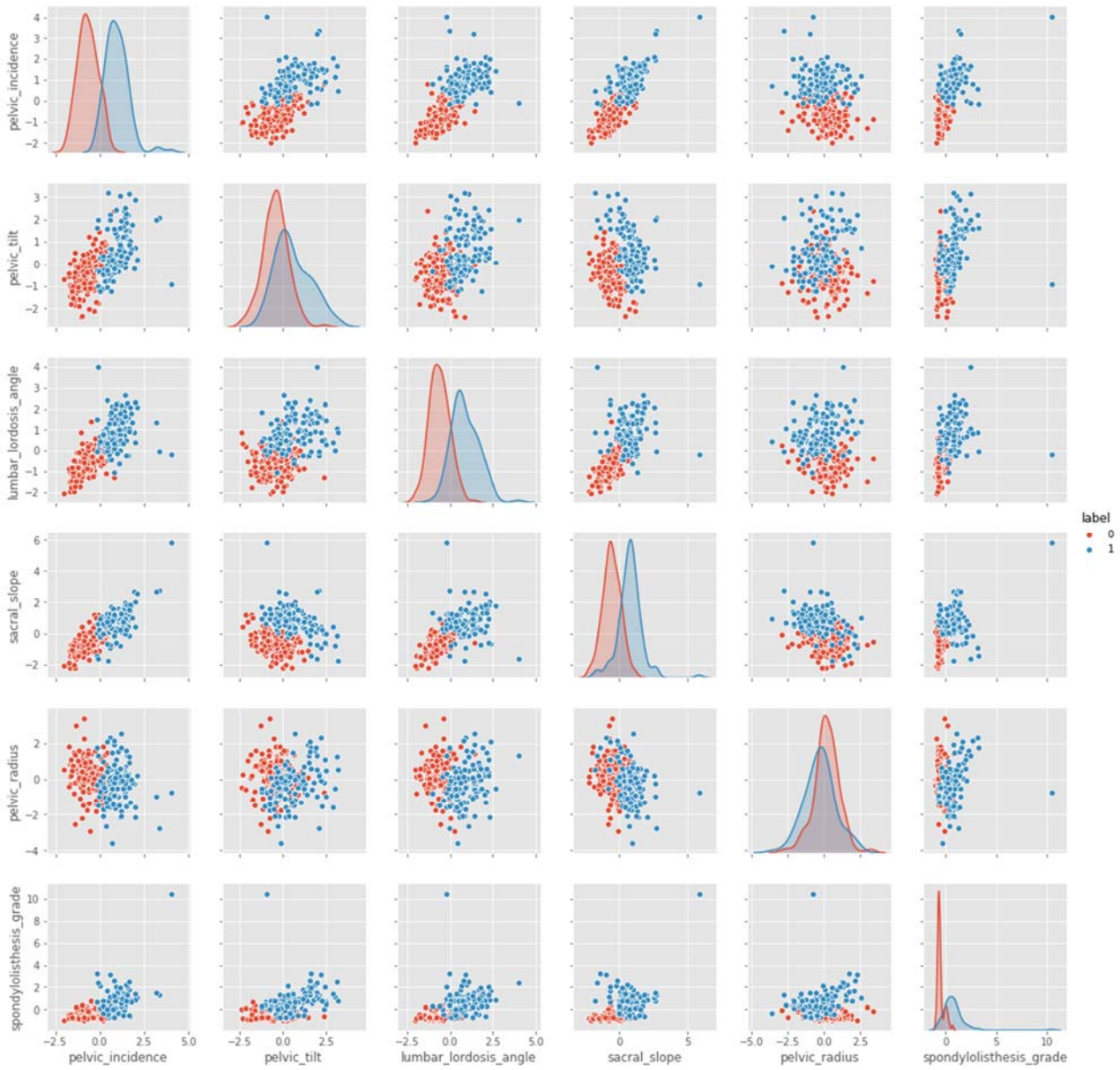

Figure 2: pair plot of biomechanical attributes

K-Nearest Neighbor (Supervised)

In supervised classification, the class labels of the data set were known and can be used to train the model. The model is able to learn the feature of the data in the labelled training data set and then achieve clustering in the unlabeled test data set.

In the KNN algorithm, if the parameter K is too small, it means that the model will become complicated and it is easy to over-fitting. Otherwise, the model becomes simple and easy to underfitting.

Therefore, we use the 10-fold cross-validation to the data and visualize the cross-validated accuracy of the different value of K for the KNN algorithm.

```
#find the best k
#set a k range between 1 to 20
k_range = range(1, 21 ,2) #The parameter k is usually odd
k_scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors = k)
    scores = cross_val_score(knn, X, y, cv=10, scoring='accuracy')
    k_scores.append(scores.mean())
plt.plot(k_range, k_scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Cross-Validated Accuracy')
plt.show()
```

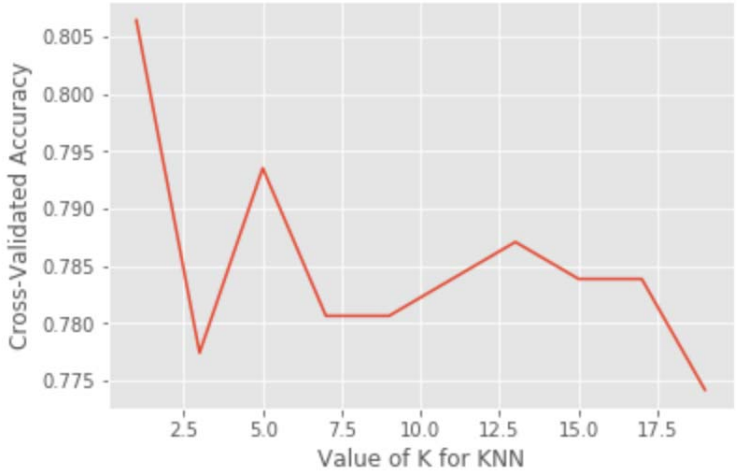

Figure 3: Accuracy & Value of K

The result shows that when K = 5 (exclude K = 1) the algorithm has the highest mean cross-validated accuracy. Here, we setting the parameter K = 5 to test the result.

```
knn = KNeighborsClassifier(n_neighbors = 5)
knn.fit(X, y)
knn_predictions = knn.predict(X)
print('Predicted Labels:\n',y_predict)
```

```
Predicted Labels:
 [0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1
 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 1
 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 0 0 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 1 0 1 0 0 1 1 1 0 1 1 1]
```

Next, evaluate the classification results.

```
from sklearn.metrics import classification_report
label = ['Abnormal', 'Normal']
print('KNN Classfication Results:')
print(classification_report(y, knn_predictions, target_names=label))
```

```
KNN Classfication Results:
             precision    recall  f1-score   support

   Abnormal       0.91      0.89      0.90       210
     Normal       0.78      0.81      0.79       100

avg / total       0.87      0.86      0.87       310
```

The evaluation of the classification results predicted by the algorithm shows that the precision, recall rate and f1-score are higher than 86%, which indicates that the supervised learning algorithm KNN does have a good performance on this data set classification task.

Discussion of how the supervised and unsupervised analyses inform each other

The biggest difference between supervised learning and unsupervised learning is whether the given data sets have labels.

Although unsupervised learning usually cannot know the clustering accuracy, here data labels are known so the accuracy can be seen as well.

```
print('K-Mean accuracy:',accuracy_score(km_predictions,y))
```

```
K-Mean accuracy: 0.6645161290322581
```

```
sum(km_predictions == knn_predictions)/len(y)
```

```
0.7032258064516129
```

In this report, the same data were used in both supervised(KNN) and unsupervised(K-Means) analyses but the predicted labels between two analyses method show clear differences. There are 70.3% predicted labels are the same between two methods and the accuracy on the K-Mean clustering is lower than KNN.

The K-Mean only considers unlabeled data. As the data may be more than two classes actually, artificially setting cluster = 2 may result in inaccurate clustering. However, the KNN algorithm can find more closely related data features through supervised learning and obtain more accurate classification results.

Reference

Clausi, D.A. (2002). K-means Iterative Fisher (KIF) unsupervised clustering algorithm applied to image texture segmentation. *Pattern Recognition, 35*, 1959-1972.

Denoeux, T. (1995). A k-nearest neighbour classification rule based on Dempster-Shafer theory. *IEEE Trans. Systems, Man, and Cybernetics, 25*, 804-813.

Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.