Prediction of Iris Data by using K-mean,

SVM and Hierarchical Clustering Algorithm

*Faculty of Engineering Environment and Computing Coventry University*

Coventry, England [bertulia@uni.coventry.ac.uk](mailto:bertulia@uni.coventry.ac.uk)

***Abstract*—This paper looks at The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). These measures were used to create a linear discriminant model to classify the species. this could be predicted by analyzing their behavior and applying K-mean, SVM and Hierarchical Clustering Algorithm learning algorithms. Multiple data preparation techniques were implemented covering class imbalance issues, encoding categorical data and feature extraction. Techniques and algorithms were implemented using Python programing language utilizing machine learning repositories, prediction results and algorithm performance measures were obtained, and visualized for comparison and discussion.**

***Keywords—machine learning; python; class imbalance; feature extraction; categorical data; K-mean,***

***SVM and Hierarchical Clustering Algorithm***

1. INTRODUCTION

The Iris flower data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems. It is sometimes called Anderson’s Iris data set because Edgar Anderson collected the data to quantify the morphologic variation of Iris flowers of three related species. The data set consists of 50 samples from each of three species of Iris (Iris Setosa, Iris virginica, and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters.

This dataset became a typical test case for many statistical classification techniques in machine learning such as support vector machines.

1. THE DATA SET

Dataset was obtained from the UCI Machine Learning repository. The dataset contains a set of 150 records under 5 attributes — Petal Length, Petal Width, Sepal Length, Sepal Width, and Class(Species).to the site the length and the width of the [sepals](https://en.wikipedia.org/wiki/Sepal) and [petals](https://en.wikipedia.org/wiki/Petal), in centimeters. Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other. Fisher's paper was published in the [Annals of Eugenics](https://en.wikipedia.org/wiki/Annals_of_Eugenics) and includes discussion of the contained techniques' applications to the field of [phrenology](https://en.wikipedia.org/wiki/Phrenology).

TABLE 1. DATASET FEATURES

This section covers data analysis on class imbalance and overcoming imbalance issues, dealing with categorical data and categorical data encoding, analysis of numerical values and data scaling, feature analysis, and feature extraction.

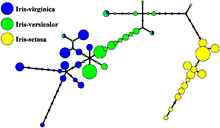


**Use of the data set**

Originally used as an example data set on which Fisher's [linear discriminant analysis](https://en.wikipedia.org/wiki/Linear_discriminant_analysis) was applied, it became a typical test case for many [statistical classification](https://en.wikipedia.org/wiki/Statistical_classification) techniques in [machine learning](https://en.wikipedia.org/wiki/Machine_learning) such as [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machines).[[5]](https://en.wikipedia.org/wiki/Iris_flower_data_set#cite_note-5)

The use of this data set in [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis) however is not common, since the data set only contains two clusters with rather obvious separation. One of the clusters contains *Iris setosa*, while the other cluster contains both *Iris virginica* and *Iris versicolor* and is not separable without the species information Fisher used. This makes the data set a good example to explain the difference between supervised and unsupervised techniques in [data mining](https://en.wikipedia.org/wiki/Data_mining): Fisher's linear discriminant model can only be obtained when the object species are known: class labels and clusters are not necessarily the same.[[6]](https://en.wikipedia.org/wiki/Iris_flower_data_set#cite_note-6)

Nevertheless, all three species of *Iris* are separable in the projection on the nonlinear and branching principal component.[[7]](https://en.wikipedia.org/wiki/Iris_flower_data_set#cite_note-7) The data set is approximated by the closest tree with some penalty for the excessive number of nodes, bending and stretching. Then the so-called "metro map" is constructed.[[4]](https://en.wikipedia.org/wiki/Iris_flower_data_set#cite_note-GorbanZinovyev2010-4) The data points are projected into the closest node. For each node the [pie diagram](https://en.wikipedia.org/wiki/Pie_chart) of the projected points is prepared. The area of the pie is proportional to the number of the projected points. It is clear from the diagram (left) that the absolute majority of the samples of the different *Iris* species belong to the different nodes. Only a small fraction of *Iris-virginica* is mixed with *Iris-versicolor* (the mixed blue-green nodes in the diagram). Therefore, the three species of Iris (*Iris setosa*, *Iris virginica* and *Iris versicolor*) are separable by the unsupervised procedures of nonlinear [principal component analysis](https://en.wikipedia.org/wiki/Principal_component_analysis). To discriminate them, it is sufficient just to select the corresponding nodes on the principal tree.



1. *Categorical Feature Encoding*

Many machine learning algorithms only work with numeric values, but categorical data can be represented as text. Even if categorical data is represented as a number (i.e. months 1-12) machine learning algorithm could give higher weight to December as it carries number 12 than January 1 and not classify them in the same way. To avoid this categorical data must be encoded into binary values and to simplify further dummy variable can be removed in order to reduce dimensionality of the dataset (Brownlee 2017, Alpaydın 2014). In this paper, eight categorical features have been encoded using python library scikit-learn utilizing LabelEncoder to encode labels into numbers (e.g. month from text February to number 2 and Tree to 1) and OneHotEncoder to convert values to binary as per example displayed in [*TABLE*](#_bookmark1)

1. *Dimensionality Reduction and Feature Extraction*

Dimensionality reduction is performed to improve the speed of data processing by reducing the amount of data to be processed while minimizing information loss, it also enables users to reduce data to two dimensions or three dimensions for visualization purposes. Dimensionality reduction can be done using Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA) or Kernel PCA (Gnanadesikan 1988, Hackeling 2014). For source code of techniques see appendix [IX.A.7)](#_bookmark29)

* 1. *LDA*

LDA is a supervised learning technique that analyses and identifies the features that have the highest class separation. In python, LDA was implemented using scikit-learn repository. LDA analysis was run however it has identified that features are collinear meaning features are closely correlated and it is unable to separate them, therefore, LDA is not an option for extracting features from the online shopper’s data set. A paper by Naes and Mevik (2001) states that this issue can be overcome by applying PCA.

* 1. *PCA*

PCA analyses features identifies their variance and sorts by highest variance. In python, PCA was implemented using scikit-learn repository and results displayed in [*Fig. 3*](#_bookmark5) and [*Fig. 4*](#_bookmark6)obtained.

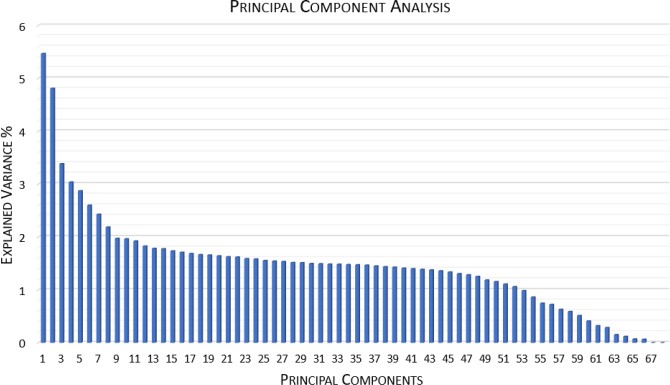


Fig. 3 PCA Individual Component Variance

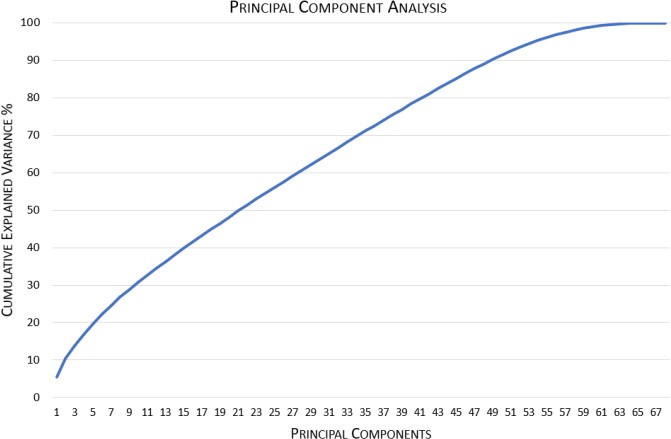


Fig. 4 PCA Cumulative Variance

Results show that most of the variables carry similar variance, meaning variables are almost equally important and only minimal dimensionality reduction can be performed.

* 1. *Kernel PCA*

Kernel PCA extends PCA by incorporating a kernel allowing to separate features that are non-linearly separable.

In python Kernel PCA was implemented using scikit-learn repository and results displayed in [*Fig. 5*](#_bookmark7)and [*Fig. 6*](#_bookmark8)(Hill and Lewicki 2006).

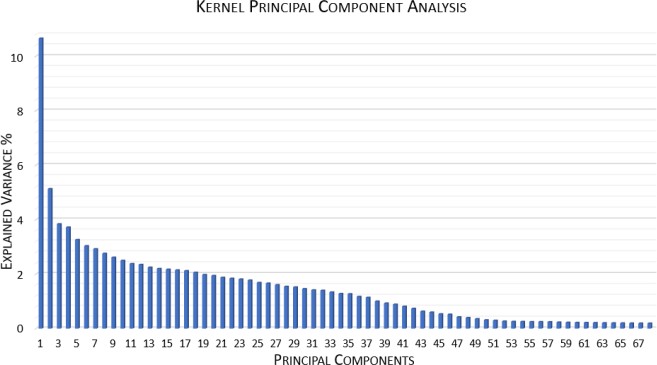


Fig. 5 Kernel PCA Individual Component Variance

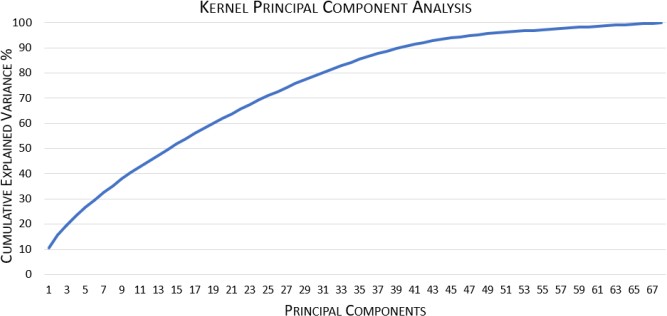


Fig. 6 Kernel PCA Cumulative Variance

Analyzing results and comparing to PCA differences can be identified. Kernel PCA displays 38 components with a variance above 1% where PCA had 52 and Kernel PCA show that 40 components add up 90% of cumulative variance wherein PCA this number is 49. The comparison suggests that Kernel PCA performs better and it is the technique to be used for online shopper’s dataset, therefore LDA and PCA will not be used in this paper. Kernel PCA with 38 features will be used as it reduces dataset by 30 features while still having 88.77% cumulative variance.

1. MACHINE LEARNING CLASSIFICATION TECHNIQUES
2. ***k*-means clustering**

***this*** is a method of [vector quantization](https://en.wikipedia.org/wiki/Vector_quantization), originally from [signal processing](https://en.wikipedia.org/wiki/Signal_processing), that aims to [partition](https://en.wikipedia.org/wiki/Partition_of_a_set) *n* observations into *k* clusters in which each observation belongs to the [cluster](https://en.wikipedia.org/wiki/Cluster_(statistics)) with the nearest [mean](https://en.wikipedia.org/wiki/Mean) (cluster centers or cluster [centroid](https://en.wikipedia.org/wiki/Centroid)), serving as a prototype of the cluster. This results in a partitioning of the data space into [Voronoi cells](https://en.wikipedia.org/wiki/Voronoi_cell). *k*-means clustering minimizes within-cluster variances ([squared Euclidean distances](https://en.wikipedia.org/wiki/Squared_Euclidean_distance)), but not regular Euclidean distances, which would be the more difficult [Weber problem](https://en.wikipedia.org/wiki/Weber_problem): the mean optimizes squared errors, whereas only the [geometric median](https://en.wikipedia.org/wiki/Geometric_median) minimizes Euclidean distances. For instance, better Euclidean solutions can be found using [*k*-medians](https://en.wikipedia.org/wiki/K-medians_clustering) and [*k*-medoids](https://en.wikipedia.org/wiki/K-medoids).

The problem is computationally difficult ([NP-hard](https://en.wikipedia.org/wiki/NP-hardness)); however, efficient [heuristic algorithms](https://en.wikipedia.org/wiki/Heuristic_algorithm) converge quickly to a [local optimum](https://en.wikipedia.org/wiki/Local_optimum). These are usually similar to the [expectation-maximization algorithm](https://en.wikipedia.org/wiki/Expectation-maximization_algorithm) for [mixtures](https://en.wikipedia.org/wiki/Mixture_model) of [Gaussian distributions](https://en.wikipedia.org/wiki/Gaussian_distribution) via an iterative refinement approach employed by both *k-means* and *Gaussian mixture modeling*. They both use cluster centers to model the data; however, *k*-means clustering tends to find clusters of comparable spatial extent, while the Gaussian mixture model allows clusters to have different shapes.

The unsupervised *k*-means algorithm has a loose relationship to the [*k*-nearest neighbor classifier](https://en.wikipedia.org/wiki/K-nearest_neighbor)popular means due to the name. Applying the 1-nearest neighbor classifier.

1. *Support Vector Machines (SVM)*

SVM is used to classify instances by linearly separating them with the highest margin possible between the class instances. When data is non-linear separable SVM uses a kernel and introduces another dimension in order to make the data separable with a hyperplane this type of action is referred to as kernel trick. Examples of linear separation and non- linear separation utilizing kernel displayed in [Fig 7](#_bookmark9) [*Fig. 8*.](#_bookmark10) Figures are for example purposes and do not represent online shoppers dataset.

*Fig 7 SVM Linear Separation* Chart, scatter chart

Description automatically generated

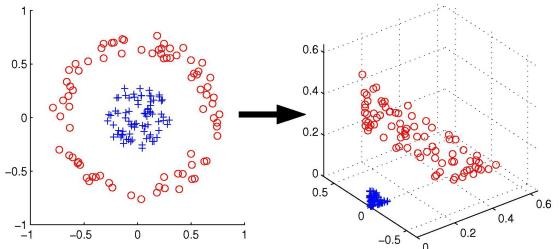


Fig. 8 SVM Non-linear Separation Using Kernel

Hierarc

**C. Hierarchical clustering**

**hierarchical clustering** (also called **hierarchical cluster analysis** or **HCA**) is a method of [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis) that seeks to build a [hierarchy](https://en.wikipedia.org/wiki/Hierarchy) of clusters. Strategies for hierarchical clustering generally fall into two categories:

**Agglomerative**: This is a "[bottom-up](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design)" approach: Each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

**Divisive**: This is a "[top-down](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design)" approach: All observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

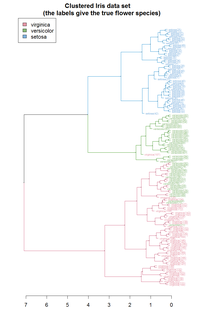
In general, the merges and splits are determined in a [greedy](https://en.wikipedia.org/wiki/Greedy_algorithm) manner. The results of hierarchical clustering[[1]](https://en.wikipedia.org/wiki/Hierarchical_clustering#cite_note-1) are usually presented in a [dendrogram](https://en.wikipedia.org/wiki/Dendrogram).

The standard algorithm for **hierarchical agglomerative clustering** (HAC) has a [time complexity](https://en.wikipedia.org/wiki/Time_complexity) of �(�3)and requires Ω(�2)memory, which makes it too slow for even medium data sets. However, for some special cases, optimal efficient agglomerative methods (of complexity �(�2)) are known: **SLINK**[[2]](https://en.wikipedia.org/wiki/Hierarchical_clustering#cite_note-SLINK-2) for [single-linkage](https://en.wikipedia.org/wiki/Single-linkage_clustering) and CLINK[[3]](https://en.wikipedia.org/wiki/Hierarchical_clustering#cite_note-CLINK-3) for [complete-linkage clustering](https://en.wikipedia.org/wiki/Complete-linkage_clustering). With a [heap](https://en.wikipedia.org/wiki/Heap_(data_structure)), the runtime of the general case can be reduced to �(�2log⁡�), an improvement on the aforementioned bound of �(�3), at the cost of further increasing the memory requirements. In many cases, the memory overheads of this approach are too large to make it practically usable.

Except for the special case of single-linkage, none of the algorithms (except exhaustive search in �(2�)) can be guaranteed to find the optimum solution.

Divisive clustering with an exhaustive search is �(2�), but it is common to use faster heuristics to choose splits, such as [*k*-means](https://en.wikipedia.org/wiki/K-means_clustering).

Hierarchical clustering has the distinct advantage that any valid measure of distance can be used. In fact, the observations themselves are not required: all that is used is a [matrix of distances](https://en.wikipedia.org/wiki/Distance_matrix).



**K- Mean clustering**

| **x** | **y** | **cluster** | **distance\_from\_0** | **distance\_from\_1** | **distance\_from\_2** | **closest** | **color** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 5.1 | 3.5 | 0 | 0.118406 | 1.109863 | 1.578233 | 0 | r |
| **1** | 4.9 | 3.0 | 0 | 0.440931 | 1.061224 | 1.688200 | 0 | r |
| **2** | 4.7 | 3.2 | 0 | 0.381602 | 1.308662 | 1.901478 | 0 | r |
| **3** | 4.6 | 3.1 | 0 | 0.521939 | 1.376153 | 1.991989 | 0 | r |
| **4** | 5.0 | 3.6 | 0 | 0.172105 | 1.250998 | 1.706933 | 0 | r |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **145** | 6.7 | 3.0 | 2 | 1.747232 | 0.797870 | 0.114978 | 2 | b |
| **146** | 6.3 | 2.5 | 2 | 1.592363 | 0.453206 | 0.554635 | 1 | g |
| **147** | 6.5 | 3.0 | 2 | 1.554098 | 0.609094 | 0.091761 | 2 | b |
| **148** | 6.2 | 3.4 | 2 | 1.194328 | 0.683078 | 0.576212 | 2 | b |
| **149** | 5.9 | 3.0 | 2 | 0.991171 | 0.232800 | 0.688491 | 1 | g |

**let’s use the concept of Inertia which is the sum of the squared distances of samples to their closest cluster center.If the value of K is huge, then the no. of points within a cluster will be less and hence the inertia will be less** **Now we will implement ‘The elbow method’ on the Iris dataset. The elbow method allows us to pick the optimum no. of clusters for classification.** **Although we already know the answer is 3 as there are 3 unique class in Iris flowersWe can get an absolute**

Chart, scatter chart

Description automatically generated

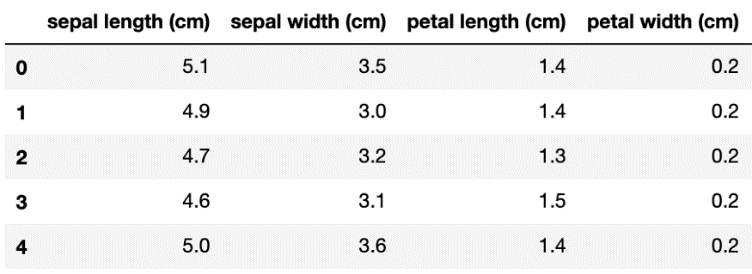
# **Output Result of K-Means clustering with K = 3 on Iris dataset.**

# **segmentation when we put higher K values but if the points with in each cluster are very less then the variation on the real data will be high leading it into over simplifying the dataSo, with K=3 we have obtained an optimal distortion/inertia with which we can segment the data into 3 different clusters with minimal error in segmentation.**

We notice that we get good results on both training and testing sets. The training set gives us a score of 99.10, whereas the testing set gives us a score of 94.73.

***B – PCA***

If you’re unfamiliar with PCA, or need a refresher, no worries. It’s one of those auxiliary DS techniques which may be necessary, or entirely useless, depending on the size and complexity of your data, as well as the intent of your project.PCA stands for ‘Principal Component Analysis’, and refers to recombinations of your original data (the principal components), which is quite similar to the process of extracting, then sorting, the color from Skittles. The metaphor in this case is useful because the process of PCA isn’t just about sorting Skittles, but extracting then sorting **color**from Skittles. The color in the metaphor is **statistical variance**in the original data.



Let’s look at the Iris dataset again visually, this time with boththe petal and sepal features, for a total of four features. The following displays the first five rows of the 150-item Iris dataset, followed by scatterplots corresponding to the petal and sepal features.

Chart, scatter chart

Description automatically generated

The standard scaler basically scales each data point based on a Gaussian, or normal distribution (i.e. it subtracts the mean of each feature from each row’s corresponding column value, then divides it by the standard deviation). We can see the first five Irises again, below, after applying the standard scaler. The fact that most of the values are negative makes sense, since the blue (Iris setosa) values in the scatter plot are mostly less than the mean of the rest of the data, regardless of feature (except for sepal width, hence some positive values).

Graphical user interface

Description automatically generated with medium confidence

Finally we are ready to apply PCA.

from sklearn. decomposition import PCApca = PCA(n\_components=2)  
principalComponents = pca.fit\_transform(scaled\_features)

And that’s it! The parameter **n\_components** allows us to select how many principle components we want to reduce our features to, in this case 2. Let’s look at the Iris dataset again, now with *only two features. (*In other words, we’ve reduced the size of our data by 50%).

Chart, scatter chart

Description automatically generated

Even though we’ve reduced our data by half, we still have two principle components which contain the majority of the statistical variance in the original data, and clustering or classification would still be an easy task. Even if we reduce all four components to just one, **throwing out a massive 75% of our original data**:

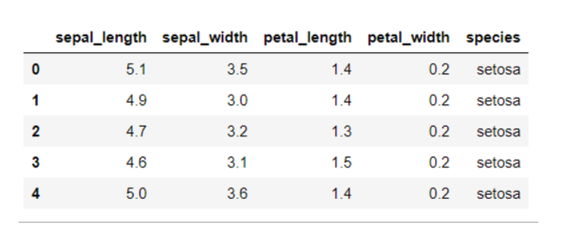
Text

Description automatically generated with medium confidence

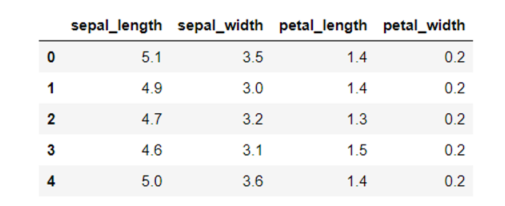
we lost about 8% accuracy going from four features to one, but that means that we **lost only 8% of our original accuracy despite throwing out 75% of our data!** That’s impressive. You can imagine how important, even necessary, PCA might be for large data sets or complex data sets with many feature.

**C- Hierarchical Clustering**

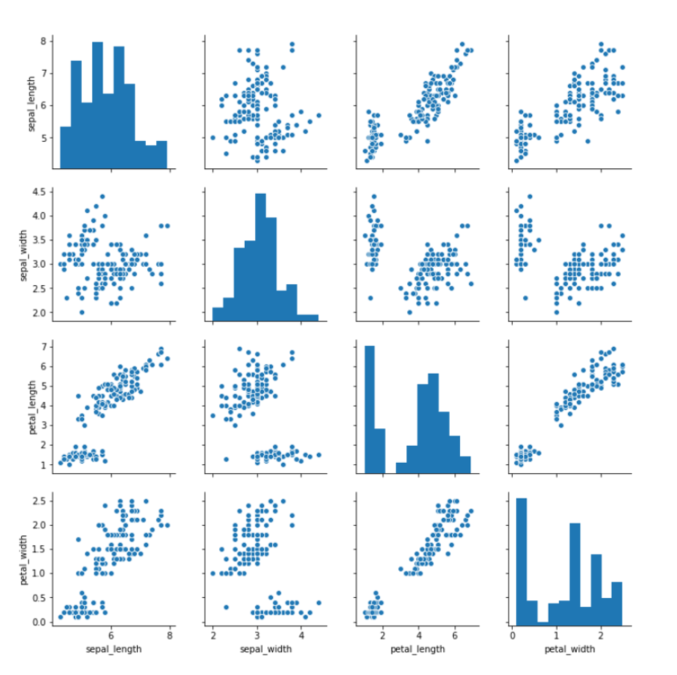
Iris dataset contains plants of three different types: setosa, virginica and versicolor. The dataset contains labeled data where sepal-length, sepal-width and petal-length, petal-width of each plant is available. We will use the four attributes of the plants to cluster them into three different groups.



Clustering is an unsupervised technique, therefore we do not require labels in our dataset. The following script removes the “species” column that contains labels, from the dataset.

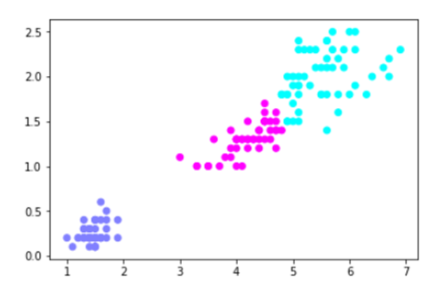


We will plot a pair plot to see if we can find any relation between the attributes. It can help us reduce the number of dimensions or attributes in our dataset.



From the output, you can clearly see that there is positive correlation between the petal-length and petal-width column which is a good indicator for clustering.

The output of the script above looks like this:



You can see the Iris data divided into three clusters.

###### Conclusion

Hierarchical clustering is one of the most popular unsupervised learning algorithms. In this article, we explained the theory behind hierarchical clustering along. Furthermore, we implemented hierarchical clustering with the help of Python’s Scikit learn library to cluster Iris data.

**FUTURE RESEARCH**

The trained model already performed well, however deep learning methods or neural networks could be applied to it in attempt to get better performance as well as more in-depth grid search technique combined with piping which allows to search for an optimum number of components for an algorithm for each of the specific parameters.

**REFERENCES**

[Kriegel, Hans-Peter](https://en.wikipedia.org/wiki/Hans-Peter_Kriegel); Schubert, Erich; [Zimek, Arthur](https://en.wikipedia.org/wiki/Arthur_Zimek" \o "Arthur Zimek) (2016). "The (black) art of runtime evaluation: Are we comparing algorithms or implementations?". Knowledge and Information Systems. **52** (2): 341–378. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1007/s10115-016-1004-2](https://doi.org/10.1007%2Fs10115-016-1004-2). [ISSN](https://en.wikipedia.org/wiki/ISSN_(identifier)) [0219-1377](https://www.worldcat.org/issn/0219-1377). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [40772241](https://api.semanticscholar.org/CorpusID:40772241).

[**^**](https://en.wikipedia.org/wiki/K-means_clustering#cite_ref-macqueen19672_2-0) MacQueen, J. B. (1967). [Some Methods for classification and Analysis of Multivariate Observations](http://projecteuclid.org/euclid.bsmsp/1200512992). Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. Vol. 1. University of California Press. pp. 281–297. [MR](https://en.wikipedia.org/wiki/MR_(identifier)) [0214227](https://mathscinet.ams.org/mathscinet-getitem?mr=0214227). [Zbl](https://en.wikipedia.org/wiki/Zbl_(identifier)" \o "Zbl (identifier)) [0214.46201](https://zbmath.org/?format=complete&q=an:0214.46201). Retrieved 2009-04-07.

[**^**](https://en.wikipedia.org/wiki/K-means_clustering#cite_ref-3) [Steinhaus, Hugo](https://en.wikipedia.org/wiki/Hugo_Steinhaus) (1957). "Sur la division des corps matériels en parties". Bull. Acad. Polon. Sci. (in French). **4** (12): 801–804. [MR](https://en.wikipedia.org/wiki/MR_(identifier)) [0090073](https://mathscinet.ams.org/mathscinet-getitem?mr=0090073). [Zbl](https://en.wikipedia.org/wiki/Zbl_(identifier)" \o "Zbl (identifier)) [0079.16403](https://zbmath.org/?format=complete&q=an:0079.16403).

[**^**](https://en.wikipedia.org/wiki/K-means_clustering#cite_ref-lloyd19572_4-0) Lloyd, Stuart P. (1957). "Least square quantization in PCM". Bell Telephone Laboratories Paper. Published in journal much later: Lloyd, Stuart P. (1982). ["Least squares quantization in PCM"](http://www.cs.toronto.edu/~roweis/csc2515-2006/readings/lloyd57.pdf) (PDF). [IEEE Transactions on Information Theory](https://en.wikipedia.org/wiki/IEEE_Transactions_on_Information_Theory). **28** (2): 129–137. [CiteSeerX](https://en.wikipedia.org/wiki/CiteSeerX_(identifier)) [10.1.1.131.1338](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.131.1338). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1109/TIT.1982.1056489](https://doi.org/10.1109%2FTIT.1982.1056489). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [10833328](https://api.semanticscholar.org/CorpusID:10833328). Retrieved 2009-04-15.

[**^**](https://en.wikipedia.org/wiki/K-means_clustering#cite_ref-forgy652_5-0) Forgy, Edward W. (1965). "Cluster analysis of multivariate data: efficiency versus interpretability of classifications". Biometrics. **21** (3): 768–769. [JSTOR](https://en.wikipedia.org/wiki/JSTOR_(identifier)) [2528559](https://www.jstor.org/stable/2528559).

Markopoulos, Panos P.; Karystinos, George N.; Pados, Dimitris A. (October 2014). "Optimal Algorithms for L1-subspace Signal Processing". IEEE Transactions on Signal Processing. **62** (19): 5046–5058. [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1405.6785](https://arxiv.org/abs/1405.6785). [Bibcode](https://en.wikipedia.org/wiki/Bibcode_(identifier)):[2014ITSP...62.5046M](https://ui.adsabs.harvard.edu/abs/2014ITSP...62.5046M). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1109/TSP.2014.2338077](https://doi.org/10.1109%2FTSP.2014.2338077). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [1494171](https://api.semanticscholar.org/CorpusID:1494171).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-7) Zhan, J.; Vaswani, N. (2015). ["Robust PCA With Partial Subspace Knowledge"](https://doi.org/10.1109/tsp.2015.2421485). IEEE Transactions on Signal Processing. **63** (13): 3332–3347. [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1403.1591](https://arxiv.org/abs/1403.1591). [Bibcode](https://en.wikipedia.org/wiki/Bibcode_(identifier)):[2015ITSP...63.3332Z](https://ui.adsabs.harvard.edu/abs/2015ITSP...63.3332Z). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1109/tsp.2015.2421485](https://doi.org/10.1109%2Ftsp.2015.2421485). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [1516440](https://api.semanticscholar.org/CorpusID:1516440).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-8) Kanade, T.; Ke, Qifa (June 2005). Robust L1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE. p. 739. [CiteSeerX](https://en.wikipedia.org/wiki/CiteSeerX_(identifier)) [10.1.1.63.4605](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.63.4605). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1109/CVPR.2005.309](https://doi.org/10.1109%2FCVPR.2005.309). [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-0-7695-2372-9](https://en.wikipedia.org/wiki/Special:BookSources/978-0-7695-2372-9). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [17144854](https://api.semanticscholar.org/CorpusID:17144854).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-9) [Pearson, K.](https://en.wikipedia.org/wiki/Karl_Pearson) (1901). ["On Lines and Planes of Closest Fit to Systems of Points in Space"](https://zenodo.org/record/1430636). Philosophical Magazine. **2** (11): 559–572. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1080/14786440109462720](https://doi.org/10.1080%2F14786440109462720).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-10) Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. [Journal of Educational Psychology](https://en.wikipedia.org/wiki/Journal_of_Educational_Psychology), **24**, 417–441, and 498–520.  
Hotelling, H (1936). "Relations between two sets of variates". [Biometrika](https://en.wikipedia.org/wiki/Biometrika" \o "Biometrika). **28** (3/4): 321–377. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.2307/2333955](https://doi.org/10.2307%2F2333955). [JSTOR](https://en.wikipedia.org/wiki/JSTOR_(identifier)) [2333955](https://www.jstor.org/stable/2333955).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-11) Stewart, G. W. (1993). ["On the early history of the singular value decomposition"](http://purl.umn.edu/1868). [SIAM Review](https://en.wikipedia.org/wiki/SIAM_Review). **35** (4): 551–566. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1137/1035134](https://doi.org/10.1137%2F1035134).

^ [Jump up to:**a**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-Jolliffe2002_12-0) [**b**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-Jolliffe2002_12-1) [**c**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-Jolliffe2002_12-2) [**d**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-Jolliffe2002_12-3) [**e**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-Jolliffe2002_12-4) Jolliffe, I. T. (2002). [Principal Component Analysis](http://link.springer.com/10.1007/b98835). Springer Series in Statistics. New York: Springer-Verlag. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1007/b98835](https://doi.org/10.1007%2Fb98835). [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-0-387-95442-4](https://en.wikipedia.org/wiki/Special:BookSources/978-0-387-95442-4).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-13) Bengio, Y.; et al. (2013). "Representation Learning: A Review and New Perspectives". IEEE Transactions on Pattern Analysis and Machine Intelligence. **35** (8): 1798–1828. [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1206.5538](https://arxiv.org/abs/1206.5538). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1109/TPAMI.2013.50](https://doi.org/10.1109%2FTPAMI.2013.50). [PMID](https://en.wikipedia.org/wiki/PMID_(identifier)) [23787338](https://pubmed.ncbi.nlm.nih.gov/23787338). [S2CID](https://en.wikipedia.org/wiki/S2CID_(identifier)) [393948](https://api.semanticscholar.org/CorpusID:393948).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-14) Forkman J., Josse, J., Piepho, H. P. (2019). ["Hypothesis tests for principal component analysis when variables are standardized"](https://doi.org/10.1007%2Fs13253-019-00355-5). Journal of Agricultural, Biological, and Environmental Statistics. **24** (2): 289–308. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1007/s13253-019-00355-5](https://doi.org/10.1007%2Fs13253-019-00355-5).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-15) A. A. Miranda, Y. A. Le Borgne, and G. Bontempi. [New Routes from Minimal Approximation Error to Principal Components](http://www.ulb.ac.be/di/map/yleborgn/pub/NPL_PCA_07.pdf), Volume 27, Number 3 / June, 2008, Neural Processing Letters, Springer

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-16) [Fukunaga, Keinosuke](https://en.wikipedia.org/wiki/Keinosuke_Fukunaga) (1990). [Introduction to Statistical Pattern Recognition](https://dl.acm.org/doi/book/10.5555/92131). Elsevier. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-0-12-269851-4](https://en.wikipedia.org/wiki/Special:BookSources/978-0-12-269851-4).

[**^**](https://en.wikipedia.org/wiki/Principal_component_analysis#cite_ref-17) Alizadeh, Elaheh; Lyons, Samanthe M; Castle, Jordan M; Prasad, Ashok (2016). ["Measuring systematic changes in invasive cancer cell shape using Zernike moments"](https://pubs.rsc.org/en/Content/ArticleLanding/2016/IB/C6IB00100A). Integrative Biology. **8** (11): 1183–1193. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1039/C6IB00100A](https://doi.org/10.1039%2FC6IB00100A). [PMID](https://en.wikipedia.org/wiki/PMID_(identifier)) [27735002](https://pubmed.ncbi.nlm.nih.gov/27735002).

**APPENDIX**

*# Importing the libraries*

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

from scipy.cluster.hierarchy import linkage, dendrogram, cut\_

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

*# Load the iris dataset*

df = datasets.load\_iris()

df = pd.DataFrame(df.data, columns = df.feature\_names)

df.head() *# See the first 5 rows*

**# Blocplot for sepal width (cm) after outlier treatment**

**sns.boxplot(y=df['sepal width (cm)'])**

**plt.show()**

*#To find the optimal no. of cluster*

cluster\_range = range(1,20)

cluster\_errors = []

for num\_cluster in cluster\_range:

    clusters = KMeans(num\_cluster, n\_init = 10)

    clusters.fit(df\_norm)

    labels = clusters.labels\_

    centroids = clusters.cluster\_centers\_

    cluster\_errors.append(clusters.inertia\_)

clusters\_df = pd.DataFrame({'num\_cluster': cluster\_range, 'cluster\_errors': cluster\_errors})

clusters\_df[0:20]

*#Ploting elbow curve or sree to find the no. of cluster*

plt.figure(figsize=(12,6))

plt.plot(clusters\_df.num\_cluster, clusters\_df.cluster\_errors, marker = 'o')

plt.xlabel('Values of K')

plt.ylabel('Error')

plt.title('The Elbow Method using Distortion')

plt.show()

*# Creating object of the model and fitting it*

model = KMeans(n\_clusters=3, max\_iter=50)

model.fit(df)

*#analysis of cluster found*

df.index = pd.RangeIndex(len(df.index))

df\_km = pd.concat([df, pd.Series(model.labels\_)], axis=1)

df\_km.columns = ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)', 'ClusterID']

km\_clusters\_Slength = pd.DataFrame(df\_km.groupby(['ClusterID']).agg({'sepal length (cm)':'mean'}))

km\_clusters\_Swidth = pd.DataFrame(df\_km.groupby(['ClusterID']).agg({'sepal width (cm)':'mean'}))

km\_clusters\_Plength = pd.DataFrame(df\_km.groupby(['ClusterID']).agg({'petal length (cm)':'mean'}))

km\_clusters\_Pwidth = pd.DataFrame(df\_km.groupby(['ClusterID']).agg({'petal width (cm)':'mean'}))

df2 = pd.concat([pd.Series([0,1,2]), km\_clusters\_Slength, km\_clusters\_Swidth, km\_clusters\_Plength, km\_clusters\_Pwidth

                ], axis=1)

df2.columns = ['ClusterID','sepal length (cm)\_mean','sepal width (cm)\_mean','petal length (cm)\_mean',

               'petal width (cm)\_mean']

df2.head()

sns.countplot(x=df\_km.ClusterID)

plt.title('Count plot for ClusterID column')

plt.show()

*#heirarchical clustering with full dendrogram*

plt.figure(figsize=(15,7))

mergings = linkage(df\_km, method = 'ward', metric='euclidean')

*# set cut-off to 50*

max\_d = 7.08                *# max\_d as in max\_distance*

dendrogram(mergings,

           truncate\_mode='lastp',  *# show only the last p merged clusters*

           p=150,                  *# Try changing values of p*

           leaf\_rotation=90.,      *# rotates the x axis labels*

           leaf\_font\_size=8.,      *# font size for the x axis labels*

          )

plt.axhline(y=max\_d, c='k')

plt.show()

*#heirarchical clustering with full dendrogram for 50*

plt.figure(figsize=(15,7))

mergings = linkage(df\_km, method = 'ward', metric='euclidean')

*# set cut-off to 50*

max\_d = 7.08                *# max\_d as in max\_distance*

dendrogram(mergings,

           truncate\_mode='lastp',  *# show only the last p merged clusters*

           p=50,                  *# Try changing values of p*

           leaf\_rotation=90.,      *# rotates the x axis labels*

           leaf\_font\_size=8.,      *# font size for the x axis labels*

          )

plt.axhline(y=max\_d, c='k')

plt.show()

*# Scatter plot to visualize the clusters*

plt.figure(figsize=(10,7))

sns.scatterplot(x='sepal length (cm)',y='sepal width (cm)', data=df\_km, hue='ClusterID', palette=['green','blue','red'])

*# Plotting the centroids of the clusters*

plt.scatter(model.cluster\_centers\_[:, 0], model.cluster\_centers\_[:,1],

            s = 100, c = 'yellow', label = 'Centroids')

plt.show()