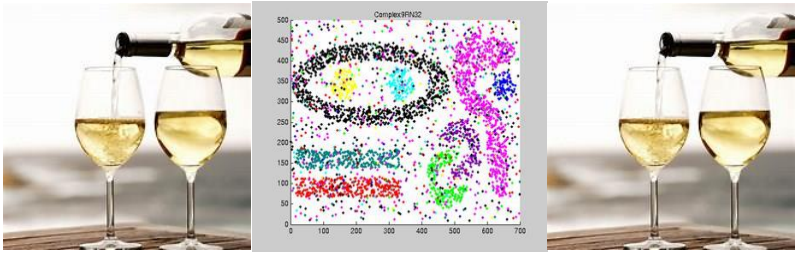
Assignment2 COSC 4335 Fall 2018
Clustering and Using Clustering for Prediction
Individual Project[1]



Learning Objectives:

1. Learn to use popular clustering algorithms, namely K-means, K-medoids/PAM and DBSCAN
2. Learn how to summarize and interpret clustering results
3. Learn to write Python functions which operate on the top of clustering algorithms and clustering results
4. Learning how to make sense of unsupervised data mining results
5. Learn how clustering can be used to create useful background knowledge for prediction and classification problems.
6. Learn how to create distance function and distance matrices in Python
7. Learn how to create and evaluate prediction models

Deadlines: Friday, Oct 19th Midnight

Datasets: In this project we will use the *Complex9_RN32 dataset* and the White Wine Quality (WWQ) Dataset *dataset* which is a modification of the white wine subset of the *Wine Quality dataset*. The *Complex9_RN32* dataset is a 2D dataset with 9 classes[2] and Wine Quality Dataset is an 12D dataset and one numerical output attribute—however, we use a transformed version of this dataset called *WWQ* which has 13 attributes, including one numerical output attribute and one ordinal class attribute (5 classes are in the dataset: A, B, C, D, and E, as explained later); the last attribute of each dataset serves as the class attibute which should be ignored when clustering the data sets; the 12th attibute of the WWQ dataset should be ignored as well—however, the class attribute as well the numerical 12th attribute of the WWQ dataset will be used in the post analysis of the clusters that have been generated by K-means, k-medoids and DBSCAN.

[1] No collaboration with your class mates is allowed!

[2] It has been obtained by modifying the original Complex9 dataset, by exposing 32% of its examples to random noise and then adding the modified examples to the original dataset.

Assignment2 Tasks:

0. The orginal Wine Quality datasets[3] have the following attibutes:

Input variables (based on physicochemical tests):
1 - fixed acidity
2 - volatile acidity
3 - citric acid
4 - residual sugar
5 - chlorides
6 - free sulfur dioxide
7 - total sulfur dioxide
8 - density
9 - pH
10 - sulphates
11 - alcohol
Output variable (based on sensory data):
12 - quality (score between 0 and 10)

Transform the White Wine Quality dataset into a new 13D dataset called WWQ as follows:

a. Normalize the first through 11^{th} attribute into z-scores
b. Keep the 12^{th} attribute 'quality' as it is
c. Introduce a new ordinal attribute called *class* (attribute 13) based on the value of the 12th attribute 'quality' as follows: 10-8→A, 7→B, 6→C, 5→D, 4-0→E

Remark: When clustering the dataset only the first 11 attributes will be used; attributes 12 and 13 will be used to evaluate the quality of a clustering result. *

1. Write an Python-function entropy(a,b) that computes the entropy and the pecentage of outliers of a clustering result based on an apriori given set of class lables, where *a* gives the assignment of objects in O to clusters, and *b* contains the class labels of the examples in O. The entropy function *H* is defined as follows:
Assume we have m classes in our clustering problem; for each cluster C_i we have proportions $p_i=(p_{i1},\ldots,p_{im})$ of examples belonging to the m different classes (for cluster numbers i=1,..,k); the entropy of a cluster C_i is computed as follows:
$H(p_i)= \Sigma_{j=1} (p_{ij}*\log_2(1/p_{ij}))$ (H is called the **entropy function**)
Moreover, if $p_{ij}=0$, $p_{ij}*\log_2(1/p_{ij})$ is defined to be 0.
The entropy of a clustering X is the size-weighted sum of the entropies on the individual clusters:
$H(X)= \Sigma_{r=1} (|C_r|/|\Sigma_p|C_p|)*H(p_r)$
In the above formulas "|…|" represents the set cadinality function[4].
Moreover, we assume that $X=\{C_1,\ldots,C_k\}$ is a clustering with k clusters $C_1,\ldots,C_k$,

You can assume that cluster 0 contains all the outliers, and clusters 1,2,…,k represent "true" clusters; therefore, you should ignore cluster 0 and its instances when computing H(X). The entropy function returns a vector: (<entropy>,<percentage_of_outliers); e.g. if the function

[3] There is a white wine and a red wine dataset.
[4] E.g. |{2,3,5}|=3 and |∅|=0!

returns (0.11, 0.2) this would indicate that the entropy is 0.11, but 20% of the objects in dataset O have been classified as outliers.

2. Write an Python-function ordinal-variation (a,b) that computes the original agreement of a bag b of ordinal classes associated with the instances of clusters given by a—the orignial classes are named A, B, C, D, and E in the WWQ dataset. It is defined as follows:

Let ϕ be definied as follows: $\phi(A)=4$, $\phi(B)=3$, $\phi(C)=2$, $\phi(D)=1$, $\phi(E)=0$,

If o is an object in the WWQ dataset, o.class denotes the value of the 13th attribute of o (which takes values A, B, C, D or E)

Let C be a cluster of WWQ objects, then the ordinal agreement in C[5] is defined as follows:

Ordinal-variation(C) = $(\Sigma_{c,c' \in C \text{ and } c \neq c'} |\phi(c.class)-\phi(c'.class)|)/(|C|**2-|C|))$

If |C|=1 then Ordinal_variation(C)=0

In the above formulas "|…|" represents the set cadinality function.

Moreover, assuming $X=\{C_1,\ldots,C_k\}$ is a clustering consisting of k clusters $C_1,\ldots,C_k$,

Ordinal-variation(X) is the number of instances weighted sum of Ordinal-variation(C_1),…,Ordinal-variation(C_k); that is:

Ordinal-variation(X)= $\Sigma_{r=1} (|C_r|/|\Sigma_p|C_p|)$*Ordinal-variation($C_r$)

However, we give X in the form of (a,b) where *a* gives the assignment of objects in O to clusters, and *b* is class variable assoicated with each object in O. Again, ignore all instances of cluster 0 from ordinal agreement computations, as those example represent outliers.

3. Write an Python-function variance(a,b) which computes the variance of the clustering result X based on an apriori given set of numerical observations—one numerical observation is associated with with each object, where *a* gives the assignment of objects in O to clusters, and *b* is the numerical observation associated with each object in O. The *variance of a clustering* is the weighted sum of the variance[6] observed in each cluster with respect to the numerical variable. The observed cluster variance is weighted by number_of_example_in_the cluster/total number of examples in all clusters; the same way how variance is assessed by regression tree learning algorithms.

In general, the function variance returns a vector: (<variance>,<percentage_of_outliers). If the used clustering algorithm supports outliers, outliers should be ignored in variance computations; you can assume that cluster 0 contains all the outliers, and clusters 1,2,…,k represent "true" clusters. For example if the function variance returns (2.8, 0.3) this would indicate that the variance of the evaluated clustering is 2.8 and that 30% of the objects in the clustered dataset are outliers. If cluster 0 does not exist, assume that there are no outliers!

(suggest to use numpy.var)

4. Write an Python-function mdist(d) that takes a dataframe d containing only continous attributes as its input, transforms the attribute values in d into z-scores, and then returns a distance matrix[7] of the Manhattan distances of the objects in the z-scored dataframe as its result.

---

[5] Assume C is given as a vector of class memberships, and the ordinal agreement adds up the values of the distances between these memberships and then divides it by the number of memberships compared.

[6] If a cluster contains only 1 object, its variance is defined to be 0.

[7] The obtained distance matrix can then be used to cluster a dataset using hierarchical clustering or with K-medoids.

input:
df=[
[1 5]
[2 6]
[3 7]
[4 8]]

return:

	1	2	3	4
1	0.000000000			
2	1.549193338	0.000000000		
3	3.098386677	1.549193338	0.000000000	
4	4.647580015	3.098386677	1.549193338	0.000000000

5. Run K-means for k=9 and k=18 twice for the Complex9-RN32 dataset. Visualize and interpret the obtained four clusterings! Also compute the entropy of the clustering results using the function you developed earlier. *(USE sklearn.cluster.KMeans (use default parameter settings))*

6. Run K-means for k=5 and k=10 for the WWQ dataset (set seed *random_state=4335*, before running k-means and use n_init=15). Next, apply mdist---the function you wrote for task 4---to a dataframe consisting of the first 11 attributes of the original White Wine Quality dataset, obtaining a distance matrix D. Report SSE, entropy, ordinal ageement and variance (using the 12^{th} and 13^{th} attribute) of the 2 clustering results obtained. *(USE sklearn.cluster.KMeans)*

7. Run DBSCAN for the Complex9-RN32 data set trying to find a clustering with the lowest entropy (try to find good parameters by manual trial and error) with 20% or less outliers. Do the same for the WWQ dataset minimizing the ordinal variation of the obtained clustering result; again you can only have 20% or less outliers! Report the obtained 2 clustering results including the entropy of the first result and ordinal-variation and entropy of the second result! Also briefly describe how you found the two clusterings! *(USE sklearn.cluster.DBSCAN)*

8. Write a search procedure in Python that looks for the "best" K-means clustering for the WWQ dataset—trying to minimize the variance of the 12^{th} attribute, assuming k=8, by exploring different distance metrics for the WWQ dataset. Distance metrics are modified by multiplying the first 11 attributes of the WWQ dataset with weight vectors $(a_1,\ldots,a_{11})$ with each weight being a number in $[0,\infty)$, set *random_state*(123), and then running K-means[8] for the transformed dataset. The search procedure you are supposed to develop returns the "best" K-Means clustering found—the one for which the variance is the lowest[9]—, the weight vector used to obtain this result and the accomplished variance as well each cluster's size and variance; please limit the number of tested weight vectors to 5000 in your implementation! Report the best clustering you found using this procedure. Also report the entropy and ordinal variation of the best clustering(s) you found! What does this result/these results tell you about the importance of the 11 attributes for predicting white wine quality? Explain how the search procedure you deleloped works! *(USE sklearn.cluster.KMeans)*

9. Learn a linear model that predicts the 12^{th} attribute using the first 11 attributes for the WWQ dataset. Interpret the obtained coefficents of the obtained linear model and access its quality of the obtained regression function and the importance of the 8 attributes. Compare this task's finding with the findings of the previous task! Next, learn a different prediction model of your own liking[10] for the same task. Report the mean squared error and the R^2 for the two models

[8] Run k-means as follows: kmeans(<number of clusters).fit(dataset); do not use other parameters!
[9] The variance of the 9^{th} attribute is low in a clustering this would indicate that the clusters contain examples of white wines with similar quality.
[10] We recommend to use SVM regression or regression trees from sklearn to obtain the second prediction model!

you obtained! Evaluate and compare the two results you obtained. *(Suggest: use linear_model from sklearn)*

10. Summarize to which extend the K-Means and DBSCAN where able to rediscover the classes in the COMPLEX9-RN32 and WWQ dataset!
(USE sklearn.cluster.KMeans AND sklearn.cluster.DBSCAN)

Training Cases for Python-Functions that need to be developed for tasks 1-3:
1st case
Let a, b, c be the following vectors:
a=(0,1,1,1,1,2,2,3)
b=(A,A,A,E,E,D,D,C)
c=(8,8,8,4,4,5,5, 6)

entropy(a,b)=4/7*H(0.5,0,0,0,0.5)+ 2/7*0 + 1/7*0=4/7
ordinal-variatio+n(a,b)=4/7*(16/6)+0+0=32/21
variance(a,c)=4/7*(16/3)+0+0=64/21 note: 4/7*variance(8,8,4,4)+0+0

2nd case
a=(1,1,1,0,0,2,2,2)
b=(A,A,A,E,E,D,D,C))
c=(8,8,8,4,4,5,5,6))

entropy(a,b)= ½*0+1/2*H(0,0,1/3,2/3,0)=1.37/3=0.456
ordinal-variation(a,b)= ½*0 +1/2*(2/3)=1/3
variance(a,c)= ½*0+1/2*((2*(1/3)**2)+(2/3)**2)/2)=1/6 note: 0+1/2*variance(5,5,6)

Deliverables for Assignment2:

A. A Report[11] which contains all deliverables for all tasks of Project2. (For python function questions, just attach your implemented function, and results for 2 test cases above)
B. An Appendix which describes how to run the procedure that you developed for Task 8.
C. An Jupyter nodebook which contains all the Python-functions you wrote for tasks 0-9 should be included.
D. Delivery of Project2 Reports: use blackboard to submit your assignment and call the attached files <*last name*>_*StudentID*_P2.docx (or <*last name*>_ *StudentID*_P2_.pdf) and <*lastname*>_ *StudentID*_P2.ipynb

[11] Single-spaced; please use an 11-point or 12-point font!