Université Paris Descartes
Faculté d'informatique

# Rapport de projet

Module : Modèles de mélange finis

Master 2 : MLDS

**Modèles de mélange–Clustering–Co-clustering–Consensus Clustering**

• Réalisé par :

**Nazim Messous**

**Wissam Benhaddad**

**22-11-2019**

Part I

# Select a model for a known number of clusters

## 1 Dataset presentation

In this section we will briefly introduce each of the datasets we have studied. We will mainly talk about the size, the nature of the variables and eventual caracteristic specific to each one of them.

### 1.1 MFEA dataset

Is a dataset describing features of handwritten digits from zero to nine extracted from a collection of Dutch utility maps. The maps were scanned in 8 bits grey values at density of 400dpi, scanned, sharpened, and thresholded. Corresponding patterns in different datasets correspond to the same original character. 200 instances per class (for a total of 2,000 instances and 241 features) have been digitized in binary images [JDM00].

### 1.2 OPTIDIGITS

Is a dataset of normalized bitmaps of handwritten digits extracted from a preprinted form. Its developement required the participation of 43 people of which 13 contributed to the test set. The 32x32 bitmaps are divided into nonoverlapping blocks of 4x4 and the number of *On* pixels are counted in each block. This generates an input matrix of 8x8 where each element is an integer in the range 0..16. Thus, the dimension of the dataset after flattening the bitmap images is of 5620 individuals described by 64 features [DG17]

## 2 Gaussian model

In this section, we will emit the hypothesis that the data is generated by a mixture of Gaussian models with a set of parameters $\theta$ to be estimated. We will train the estimator on the two previously cited datasets and compare the results in terms of Accuracy, NMI, ARI and BIC (see the figure 2)

### MFEA dataset

For this dataset, we tried the 15 types of Gaussian models, but only 4 of them gave a valid partitioning. i.e, at least one non-null partition. The results were as follow

| Model | BIC | Accuracy | NMI | ARI |
|---|---|---|---|---|
| EEI | -20101 | 0.3485 | 0.7374 | 0.6534 |
| EII | -19821 | 0.3465 | 0.7308 | 0.6511 |
| EEE | -1604 | 0.104 | 0.0090 | 1.566403e-05 |
| EEV | -3281049 | 0.252 | 0.4787 | 0.1933 |

Table 1: Results of the experiments for the MFEA dataset

**Conclusion:** We can see that the spherical and diagonal distribution (EEI and EII) outperform the two models (EEE and EEV) surely due to their non ellipsoidal distribution. But since EEE gave the best bic criterion, we will compare EEE's partition with the partiton obtained with Rmixmod.

### OPTIDIGITS dataset

for this dataset we tried the 15 Gaussian models, but only two of them (the 2 models have the same characterising except for the volume) worked with similar measures :

| Model | BIC | Accuracy | NMI | ARI |
|---|---|---|---|---|
| EII | -1891797 | 0.0266 | 0.7012 | 0.5850 |
| VII | -1883543 | 0.0101 | 0.68 | 0.5534 |

Table 2: Results of the experiments for the OPTIDIGITS dataset

**Conclusion:** The two models gave slightly similar results with a benefit to the **EII** model. This can be explained by the nature of the data which corresponds to the hypothesis of this model.

## Multinomial model: Rmixmod

**Method:** we combined the 4 clusters obtained with Mclust into a single matrix $n \times t$ ($t$ is number of cluster and $n$ is the number of samples) and we apply a multinomial model using **Rmixmod**

| Dataset | Accuracy | NMI | ARI |
|---|---|---|---|
| MFEA | 0.3854 | 0.73681 | 0.6572 |
| OPTIDIGITS | 0.1818 | 0.6765 | 00.5623 |

Table 3: Results of the experiments for the Multinomial model

**MFEA analysis :** According to last experimentation with **Mclust**, EEE is the model with the best BIC despite having a poor NMI, ARI and Accuracy score (NMI and ARI that we can't compute in more realistic cases due to the lack of the label information). However, the EEI and EII models gave us interesting partitions in terms of ARI, NMI and accuracy. These results showed the importance of the consensus clustering, the aspect of combining many propositions by taking advantage of each one of them has lead to a more robust decision making process. In a sense, the EEE model was clearly the optimal choice considering only the BIC criterion. Thus, leading to a relatively bad partitioning. The consensus gave as good, and even better results than the two previous models.

**OPTIDIGITS analysis:** **Rmixmod** gave us a partition pretty similar to the previous two generated by **Mclust** with a mixture of Gaussian distribution.Therefore, it can be considered as a good consensus method.

## Results amelioration strategy

**Method:** The first step is to take the best partitioning given by the two previous models, in our case it's EII and EEI for the MFEA dataset and EII VII for the other dataset. Then, we will construct the Co-Association matrix $M^P$ which general term $\alpha^P_{i,j}$ is the following :

$$CA_{i,j} = \frac{1}{P} \sum_{t=1}^{P} \delta(P_t(x_i), P_t(x_j)) \quad (1)$$

with :

- $P$ : the number of partitioning
- $P_t(x_i)$ : is the function that associates a label to a given entry point $x_i$
- $\delta$ : is the comparative function which takes the value 1 if the two classes are the same and 0 otherwise.

which is one of the simplest consensus function because it is used to avoid the label correspondence problem which can be very hard to solve it can take high computational cost with a complexity in $O(k^3)$ [VR11].

The problem with Co Association matrix is the space the matrix can occupy ($n \times n$ dimension ), but the advantage is that it's done only once.

Third step is to choose what strategy to use in order to regroup the n samples into clusters

- **First solution :** according to [Fre01] the solution is to join in the same cluster, objects with a co-association value greater than 0,5, the results were not a very good, we can say that's it is a bad consensus in this case:

| Dataset | NMI | ARI |
|---|---|---|
| MFEA | 0.50732 | 0.26897 |
| OPTIDIGITS | 0.6543 | 0.5211 |

Table 4: Results of the experiments for first option

- **Second Solution:** The second idea is to apply a simple K-means on our matrix and to take the clusters to generate as our consensus the results were much better with a K-means than wiht the last strategy, still Rmixmod gave us a better consensus :

| Dataset | NMI | ARI |
|---|---|---|
| MFEA | 0.7125605 | 0.6070182 |
| OPTIDIGITS | 0.682112 | 0.5924156 |

Table 5: Results of the experiments for second option

**Conclusion:** First, we can observe via the results that the k-means based regrouping method is more efficient than the thresholding. The second Point is that for the second dataset **Optidigits**, the partition obtained with the K-means strategy gave the best results in term of NMI and ARI amongs partitions. So we can agree that creataing a CO Association matrix followed by a clustering with K-means is a good option as a clustering consensus.

**Note about the JAFEE dataset:** We selected 4 models since $n << p$, EII VII EEI VEI and it systematically gave very interesting results in terms of NMI (about 0.95 0.96), ARI (too 0.92 0.93) and accuracy 0.94 0.96. In addition, the partitions given by EII and VII were identical (parition $x$), as well as the ones given by EEI and VEI (partion $y$). Finally there was only a small difference between partition x and partition y. This was probably due to small number of samples (223). So we did not continue the work with JAFEE, there was no need to doing a consensus on partitions which were already almost identical.

# Part II
# Mutual Information

**Note:** We will use only data set which were normalized with L2 norm

## 3 CoClustInfo :

**CoCLustInfo** was executed about 20 times, each time changing the parameters, CoCLustInfo is about the maximization of the mutual information

- max-iter : the maximum number of inner iteration of the algorithm
- n-init : Number of time the algorithm will be run with different initialization
- random-state : the generator used to initialize the centers

After the execution, the top 5 partitions, in term of the maximised criterion which is the mutual information), were selected. The results were as follow :

| **Dataset** | **NMI** | **ARI** | **Mutual information** |
|---|---|---|---|
| Classic4 | 0.68-0.72 | 0.63-0.70 | 0.41-0.42 |
| CSTR | 0.71-0.78 | 0.74-0.81 | 0.50-0.51 |
| ng5 | 0.69-0.74 | 0.66-0.77 | 0.36-0.37 |
| rcv4 | 0.39-0.55 | 0.33-0.53 | 0.47-0.49 |

Table 6: Results of the experiments for CoClustInfo

**Classic 4 data set** This data set consists of 4 different document collections: CACM, CISI, CRAN, and MED. $n = 7094$ is number of samples and $p = 5896$ is the number of features. The experiments show that the best partitions are number 16, 19, 14, 4, 12 in that order.

**CSTR data set** This CSTR VCTK Corpus includes speech data uttered by 109 native speakers of English with various accents. $n = 475$ samples described by $p = 1000$ features. This time the experiments show that the best partitions are 16, 18, 15, 13, 8 also in that order.

**NG data set** Multilabel data set from the text domain, it is about $n = 4905$ samples described by $p = 10167$ features. This time the experiments show that the best partitions are 18, 15, 16, 17, 13 also in that order.

**RCV4** The worst results (in terms of ARI and NMI) were the last dataset's (rcv4) despite having a better Mutual Information value than the third one (ng5), it shows that the criterion can mislead, in real use cases, the evaluation of the model.

## 4 Concensus clutering using hypergraph

The Cluster-Ensembles library use 3 different algorithm to try to find a suitable and optimal solution for choosing the right partitions from a set of proposed paritions $P$

- **CSPA:** Cluster-based Similarity Partitioning Algorithm

- **HGPA:** HyperGraph Partitioning Algorithm
- **MCLA:** Meta-CLustering Algorithm

MCLA usually give the best results among the 3 algorithm when there is no outliers nor noise "This is because MCLA assumes that there are meaningful cluster correspondences which is more likely to be true when there is little noise and less diversity", according to literature MCLA give best results following by CSPA and finally HGPA has the worst results. Indeed during all the experimentations MCLA was the best criterion.

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.7324 | 0.7155 |
| CRST | 0.7395 | 0.724 |
| ng5 | 0.7013 | 0.661 |
| rcv4 | 0.5416 | 0.4828 |

Table 7: Results of the experiments for Cluster Ensembles

**CLassic4** After the end of the experiments, a small improvement in the NMI and the ARI was noticed especially after we applying the MCLA algorithm.

**CRST and NG5** It is noticeable that the NMI and ARI we have obtained are in the vicinity to the one we get with CoClustInfo, the conclusion is that Hypergraph consensus function gave one good results in term of NMI and ARI.

**rcv4** In these last dataset, it is clearly noticable that the MCLA consensu method provided a great boost in the quality of the results in terms of NMI and ARI. RCV4 data set's clustering has proven to be a hard task for the previous consensus methods (based on the top-K picking of partitions based on a metric). Despite its important computational needs, these methods can overcome some difficulties.

## 4.1 Consensus clustering using Co association matrix

**Method :** We will compute again a Co-association matrix from the dataset and apply a CoClustring algorithm on our matrix and not a simple clustering like before.

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.52760 | 0.4640 |
| CRST | 0.6027 | 0.589 |
| ng5 | 0.5033 | 0.477 |
| rcv4 | 0.49 | 0.46 |
| here big problem | | |

Table 8: Results of the experiments for CoClustInfo on the Co-ass matrix

**Classic4** This the consensus algorithm didn't work well on this data set, even the first partitions (chosen with the mutual information criterion) outperformed it. We can easily conclude by saying that hypergraph consensus function is a better strategy than Co Association matrix. This constatation is made even worse when we consider the amount of time spent on constructing the matrix.

**ng5, rcv4 and CSTR** The same constatation can be made about these three datasets, the baseline model of CoClustering using the mutual information outperformed this attempt of consensus clustering.

## 4.2 Adding Spherical k-means

It is a clustering algorithm based on cosine distance we will add the partition obtained with Spherical k-means in our matrix, in order to try to increase our results doing a consensus on the partitions

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.7268 | 0.6776 |
| CSTR | 0.7318 | 0.716 |
| NG5 | 0.7286 | 0.6688 |
| RCV4 | 0.511 | 0.434 |

Table 9: Results of the experiments for CoClustInfo on the paritítions augmented with the spherical K-means

**Classic4** Unlike what we expected adding the spherical K-means partition decrease the nmi and ari this means that the s k-means partition didn't influance a lot the consensus which is intuitively logical since it's only one partition amongst 6 others, Co Association matrix gave even worse results applied on this six partition than hypergraph function. We can conclude that adding a Spherical Kmeans partition is not always a good idea.

**CSTR and rcv4** This time adding the S k-means partition didn't change anything to the results it even decrease the nmi and ari a little for the rcvs data set.

**ng5** It can be observed a improvement of the nmi value and a little increasing of the ari value, and we can conclude that adding the S k-means partition was worth it even if it was just a little improvement which can be important in the research field.

## 4.3 Multiply number of classes in columns

In the previous experiments, an important assumption was emitted, the number of classes on the columns and rows must be equal. In this section, this hypothesis will be tested by scaling the number of classes of the columns and the aim is to improve the results. The study will be completely redone for this use case but the addition of the S k-means partition method will be ignored.

| **Dataset** | **NMI** | **ARI** | **Mutual Information** |
|---|---|---|---|
| Classic4(*2) | 0.7263-0.7509 | 0.677-0.724 | 0.521-0.523 |
| Classic4(*3) | 0.731-0.766 | 0.6487-0.7333 | 0.548-0.55 |
| CSTR(*2) | 0.746-0.763 | 0.794-0.825 | 0.611-0.614 |
| CSTR(*3) | 0.737-0.772 | 0.778-0.812 | 0.637-0.644 |
| ng5(*2) | .739-0.826 | 0.7411-0.8552 | 0.444-0.469 |
| ng5(*3) | 0.7817-0.8272 | 0.8083-0.8500 | 0.4871-0.4973 |
| rcv4(*2) | 0.41-0.47 | 0.20-0.26 | 0.152-0.156 |
| rcv4(*3) | 0.40-0.43 | 0.15-0.21 | 0.12-0.14 |

Table 10: Results of the experiments for CoClustInfo on the datasets with a scaled column classes number

### 4.3.1 Hyper-graph consensus strategy

The first method is the consensus based on the Hyper-Graphs.

| **Dataset** | **NMI** | **ARI** |
|---|---|---|
| Classic4(*2) | 0.7516 | 0.7172 |
| Classic4(*3) | 0.7517 | 0.6730 |
| CSTR(*2) | 0.7577 | 0.804 |
| CSTR(*3) | 0.7941 | 0.8405 |
| ng5(*2) | 0.698 | 0.640 |
| ng5(*3) | 0.750 | 0.687 |
| rcv4(*2) | 0.4583 | 0.2263 |
| rcv4(*3) | 0.4309 | 0.1669 |

Table 11: Results of the experiments for Cluster Ensembles on the datasets with a scaled column classes number

### 4.3.2 Co-Association matrix consensus strategy

The Co-Association matrix method will once again be explored in this section for a better error-proof experiments. Unfortunately, the results remained of bad quality, always in term of the two used metrics but also in term of execution time.

| **Dataset** | **NMI** | **ARI** |
|---|---|---|
| Classic4 (*2) | 0.6804 | 0.6402 |
| Classic4(*3) | 0.596 | 0.5283 |
| CSTR(*2) | 0.6427 | 0.663 |
| CSTR(*3) | 0.4946 | 0.49589 |
| ng5(*2) | 0.554 | 0.478 |
| ng5(*3) | 0.633 | 0.572 |
| rcv4(*2) | 0.5154 | 0.3706 |
| rcv4(*3) | 0.4774 | 0.2701 |

Table 12: Results of the experiments for CoClustInfo on the datasets with a scaled column classes number

**Classic4**

- Hypergraph consensus function still outperforms Co association matrix consensus function.
- Increasing the number of classes in columns gave us much better results than having number of rows = number of columns
- Taking the double of the number of classes in columns was a better option than the triplet for both consensus function.

**CSTR**

- The same points as for Classic4
- Taking the double of the number of classes in columns was a better option than the triplet for Co association matrix consensus function, but this time taking the triplet was better than taking only double for the hypergraph consensus function.

**rcv4**

- Interestingly enough, the Co-ass matrix gave better results than the Hyper Graph method, finally the double of the number of classes in columns was a better option than the triplet.

**NG5**

- Taking the triple of the number of classes in columns was a better option than the double.

# Part III
# CoclustMod and CoClustSpecMod

## 5 CoClustMod :

We've launched CoCLusMod for about 20 times, each time changing the parameters varied in the previous part.After obtaining the 20 partitions, we measured the optimized criterion in the case of CoClustMod. In our case, the criterion is the mutual modularity. The results were as follow:

| **Dataset** | **NMI** | **ARI** | **Modularity** |
|---|---|---|---|
| Classic4 | 0.68-0.69 | 0.64-0.68 | 0.4185-0.4195 |
| CSTR | 0.69-0.74 | 0.74-0.79 | 0.45-0.46 |
| ng5 | 0.409-0.542 | 0.4-0.555 | 0.307-0.332 |
| rcv4 | 0.41-0.44 | 0.38-0.40 | 0.418-0.44 |

Table 13: Results of the experiments for CoClustMod

### 5.1 Concensus clutering using hypergraph

| **Dataset** | **NMI** | **ARI** |
|---|---|---|
| Classic4 | 0.7264 | 0.7054 |
| CSTR | 0.6982 | 0.74 |
| ng5 | 0.55 | 0.535 |
| rcv4 | 0.492 | 0.434 |

Table 14: Results of the experiments for the Hyper-Graph based consensus after using CoClustMod

### 5.2 Consensus clustering using Co association matrix

| **Dataset** | **NMI** | **ARI** |
|---|---|---|
| Classic4 | 0.610 | 0.563 |
| CSTR | 0.639 | 0.674 |
| ng5 | 0.23 | 0.218 |
| rcv4 | 0.496 | 0.45 |

Table 15: Results of the experiments for the Co-Ass matrix based consensus after using CoClustMod

## 5.3 Adding Spherical k-means

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.7265 | 0.7065 |
| CSTR | 0.7077 | 0.75 |
| ng5 | 0.678 | 0.629 |
| rcv4 | 0.496 | 0.45 |

Table 16: Results of the experiments for CoClustMod after adding the partition of S-kmeans

## 5.4 Data set summarize:

**Classic4 :** Globally using CoclustMod instead of CoCLust Info wasn't a big success the results drop in term of nmi,ari when we applied the coclustring algorithm, the consensus function hypergraph worked less well too, but nmi and ari improved very well with the co Association matrix (10 point each of them) compared to when we tried CoCLustInfo on the co ass matrix,and even adding the S k-means partition was little better under CoclustMod, but since hypergraph function is still the best we can say than CoClust info is a better choice for this data set.

**CSTR and ng5** This time we see even better that CoClustInfo works better than CoclustMod nmi and ari drop with absolutely all the methods and strategies.

**rcv4** As it is observed, the CoClustMod algorithm gave worse results when directly compared to CoClustInfo. The same goes for the Hyper-graph based consensus.As for the co-association based consensus, the results remained the relatively the same.

## 5.5 CoClustSpecMod :

we launch CoCLusMod about 20 times, each time changing the parameters: -max_iter : ( Maximum number of iterations) -n_init : Number of time the algorithm will be run with different initializations -random_state : the generator used to initialize the centers

Since CoClustSpecMod doesn't return any criterion on which we can work on, we will take the 5 partitions randomly, the results were :

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.0090-0.009 | -0.001–0.001 |
| CSTR | 0.759-0.76 | 0.795-0.796 |
| ng5 | 0.7349-0.7469 | 0.6529-0.6641 |
| rcv4 | 0.468 | 0.428 |

Table 17: Results of the experiments for CoClustSpecMod

### 5.5.1 Concensus clutering using hypergraph

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.0011 | -0.00085 |
| CSTR | 0.759 | 0.7954 |
| ng5 | 0.7334 | 0.6517 |
| rcv4 | 0.377 | 0.459 |

Table 18: Results of the experiments for the Hyper-Graph based consensus after using CoClustSpecMod

### 5.5.2 Consensus clustering using Co association matrix

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.00844 | -0.001 |
| CSTR | 0.600 | 0.4745 |
| ng5 | 0.021 | -2.235e-5 |
| rcv4 | 0.468 | 0.428 |

Table 19: Results of the experiments for the Co-Ass matrix based consensus after using CoClustSpecMod

#### 5.5.3 Adding Spherical k-means

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.7474 | 0.6776 |
| CSTR | 0.7422 | 0.7933 |
| ng5 | 0.733 | 0.651 |
| rcv4 | 0.496 | 0.45 |

Table 20: Results of the experiments for CoClustMod after adding the partition of S-kmeans

#### 5.5.4 Data set summarize:

**Classic4 :** The results with CoClustSpecMod on this Data set were not concluent with a NMI near 0 and a negative ARI, and this regardless of the consensus and the method (Hypergraph, Co association) But in the other hand when added the S k-means partition to our 5 best partitions and we applied the hypergraph function on the these 6 partitions the nmi and ari increased very well (0.74 and 0.67 for each of them), we can say that using a good clustering method like S k-means which give generally good results combined to CoCLustInfo and CoClustMod can imporve CoCLustSpecMod results. We can add that taking 10 partitions with CoClustSpecMod instead of only 5 drop the nmi from 0.74 to 0.34 and the ari from 0.79 to 0.32. We can probably explain this low nmi and ari by the nature of the data.

**CSTR and rcv4:** For CSTR we can conclude that CoClustInfo is still the best method specially when me mutiply the number of classes in columns by 2 and 3, still CoClustSpecMod gave better results for CSTR than CoclustMod showing that we can not generalize the results we get previously with classic4

**NG5:** For CSTR we can conclude that CoClustInfo is still the best method specially when me mutiply the number of classes in columns by 2 and 3, still CoClustSpecMod gave better results for ng5 than CoclustMod (much better) when it is about hypergraph consensus function and decrease drastically for the Co ass consensus function

## 6 Combined all methods:

Since from the beginning of the project hypergraph always outperform the Co Association matrix method, we will be satisfied by applying this method only.

| Dataset | NMI | ARI |
|---|---|---|
| Classic4 | 0.6787 | 0.67863 |
| CSTR | 0.746 | 0.7867 |
| ng5 | 0.8015 | 0.7144 |
| rcv4 | 0.496 | 0.45 |

Table 21: Results of the experiments for CoClustMod after adding the partition of S-kmeans

### 6.1 Data set summarize:

**Classic4 :** The results are logical since there is intuition that would suggest that combining many Co clustering and clustering( S k-means) algorithm would improve the results, indeed the partition we get is better that the one we get from SpecMod but worse than the one we get from multiplying the number of classes in column, it's a better to work directly with the best algorithm (Coclust info in our case)

**CSTR and ng5:** The results are more interesting than the one we get from Classic 4 since it outperform the majorities of methods we tried except multiplying the number of classes in column with ColustInfo.

**RCV4:** Approximately the same results we obtained with others method, less well than CoClustInfo however.

Part IV

# Conclusion

We have seen through our project, that criterions to choose the best partitions in a non supervised context could be misleading for example the MFEA data set in the second part or with the data set RCV4 in the third part when we applied CoClustInfo and here we see the importance of a consensus to be sure of the results, in addition combining many methods and applying a consensus can be an additional security in the case where one of them didn't work well (like SpecMod with classic4) . Finally we want to conclude that hypergraph consensus function was better than a simple Co ass matrix, for the Co-clustering algorithm CoClustInfo when we increase the number of classes in columns outperformed the others algorithms, but we need to keep in mind that the results when adding a S k-means partition, double or triple the number of classes in columns is variable and it all depend on the data set so the best solution is to experiment on all the possibilities in an exhaustive manner and keep the best.

## References

[JDM00] Anil Jain, Robert Duin, and Jianchang Mao. "Statistical Pattern Recognition: A Review". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22 (Jan. 2000), pp. 4–37. DOI: 10.1109/34.824819.

[Fre01] Ana Fred. "Finding Consistent Clusters in Data Partitions". In: *Multiple Classifier Systems*. Ed. by Josef Kittler and Fabio Roli. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 309–318.

[VR11] Sandro Vega-Pons and José Ruiz-Shulcloper. "A Survey of Clustering Ensemble Algorithms." In: *IJPRAI* 25 (May 2011), pp. 337–372. DOI: 10.1142/S0218001411008683.

[DG17] Dheeru Dua and Casey Graff. "UCI Machine Learning Repository". In: (2017). URL: http://archive.ics.uci.edu/ml.