Data Science HW2 Report

Clustering Problem

B10915008 汪萬丁

A lot of approaches has been tried for this homework, and the source code might be a bit messy. But for proofing that such research has been taken, I purposely left the part undeleted.

My approach uses the jupyter notebook file, and the rest of the code will be explained in the cell manner for clarity (that is, I will explain the usage of cell1, cell2, etc). There are a total of 34 cells, but for creating the final model used for submission, only the last cell correspond to it, while the rest is for loading the dataset, transforming categorical variables to number (although later on it will be one hot encoded), Exploratory Data Analysis, generating combinations of feature on dataset column ,and so on.

To run my code, one can simply open the file in jupyter notebook or Visual Studio code, and execute all parts of the program sequentially to get the desired output. The cell that is responsible to make the final output is the last cell, and there is a comment about the setting about the final model produced (it will also explained further down in the report regarding the setting of the model).

The following is the explanation of each cell (markdown cells are not counted here):

- Cell 1
  First of all, this part of the program tries to import all the necessary libraries and classes needed for the first part of the program (visualization and also data loading).

  The convertPitchInt(strInt) function tries to convert the input of feature 13, which is the pitch of the song, into enumeration of integer. I use my own function for interpretability.

  After loading the dataset (train.csv) I tried to convert the column names of respective csv file into these names:
  - song_id → song_id
  - Feature 1 → Authenticity
  - Feature 2 → Intensity
  - Feature 3 → SuitDance
  - Feature 4 → PlayTime
  - Feature 5 → BPM
  - Feature 6 → Live
  - Feature 7→ Positivity
  - Feature 8 → Duration
  - Feature 9 → Loudness
  - Feature 10 → Vocal
  - Feature 11 → Key

- Feature 12 → InstProportion
- Feature 13 → Pitch

Finally, just like the last homework, I group some columns for useful fast experiments, for example I grouped all the variables with the name XAllDefault, the numeric variables into XNumericDefault, categorical variables into XCategoryDefault. The rest of the variable groups are related to the experimental part of the future model making input, which may be related with further analysis that has been done on the further below cells. Note that the best variable which I will use for the model is the 3VarBest.

- Cell 2 - 3
  I check for the cardinality of pitch feature, which turns out all pitches exist from C, C#, and so on all the way to B. Then, I also check for null values which turns out to be nonexistent.
- Cell 4 - 5
  I try to map the correlation table into the screen. On cell 4, I try to map the correlation without normalization, and just for reference I tried to map on cell 5 about the correlation with normalization. Some notable discovery from cell 4 is that Authenticity, Intensity and Loudness are highly correlated (their absolute value of correlation is more than 0.7). Using these variables altogether might not, but not guaranteed, to be not wise because of these 3 features might giving similar information to the clustering model and making feature duplicity.
- Cell 6 - 8
  I try to map the distribution of each variable. Where on cell 6 I try to plot the whole variable without transforming it, while on cell 7 I try to apply PowerTransform to analyze how much the graph has been normalized. From the analysis on cell 6, we can see that the some of the graph are badly skewed (most of them are skewed to the left side), while there are also variable that bimodal. The only variables that follow (similar to) normal distribution are only SuitDance, PlayTime, BPM, Positivity.

  On Cell 7, we can see that we can fix the skewness of the graph closer to center, and also the bimodal are starting to converge together, but overall they are still similar shape to that in cell 6.

  The program in cell 6 and 7 is not working with variables with non numeric input, so for the pitch variable which contain strings, I use countplot of seaborn to visualize to the screen on cell 8.
- Cell 9 – 21
  On these few cells I try to plot each variable and paired with another to the screen, in hope of appearance of distinct feature shape between them, this way clustering algorithm dbscan might work effectively. But with no avail, the plot between two variables don’t show distinct result difference. Using dbscan might be hard because there is no distinct shape in the plotting and predicting the accurate parameters for the algorithm might be hard.

Furthermore, I set the color of the point to be purple if it doesn't appear on the test set, and yellow if the point appear on the test set. We can note that it may have been some of the test set points are actually outlier, as they are isolated from the majority distribution of other points.

- Cell 22 – 23
  The variables are plotted into the boxplot for the distribution to be visualized. The cell 22 visualized the not normalized variable, which gives so little information because duration variable has greater range compared to the other variables. So, I will try to plot the boxplot once more on cell 23 with the min max scaler applied. We can see that some of the variables have point way off the interquartile range of the variables, which indicates that they have a lot of outlier.
- Cell 24 – Cell 27
  This section will try to detect the outlier using two methods. Each of the two methods will point that a point is outlier if:
  - It lies under Quartile 1 – 1.5*Interquartile Range, or abover Quartile 3 + 1.5 * Interquartile Range
  - It lies under mean – 3*standard deviation, or mean + 3*standard deviation.

  The cell 24 will print true or false whether each of the variables have outlier points in any of it contents using the first method. The feature that doesn't follow normal distribution are most likely must have outlier. The cell 25 will particularly print the points and count the amount of points that has outlier feature (based on the selection SuitDance, PlayTime, BPM, Positivity, InstPropotion. This particular variables are inspected because at one point of my analysis they yield the best result on Kaggle), but appear on the test set.

  The cell 26 and 27 conducts similar experiment with that in 24 and 25, but now using the second method for inspection.

- Cell 28
  In this cell, just then the content of the pitch variables which is previously a string, is converted to numeric (or one might say a label encoding). It is just converted at this point because I want the graph analysis to be using the original string for easier analysis (such as when the seaborn countplot, the label on the x axis is the original content).
- Cell 29 – 31
  These few cells are related to me generating variable groups of columns for model input. First of all, in the beginning I have tried as a comparison of answer sheet and I found the answer that yield high answer is by using PlayTime variable with KMeans and predefined cluster 3, MinMax scaled. The other method that I have tried has no avail and doesn't pass the baseline. In general, I have tried various models, which are: KMeans, Agglomerative, Mini KMeans, Affinity Propagation, DBScan, Spectral, OPTICS, Bisecting KMeans, HDBScan, Birch, KPrototypes. As far as I have cross examined, the best model that can generate high score in Kaggle (and also by comparing manually with the test

answer that yield high score in Kaggle) is Kmeans. Furthermore, the cluster number that yield the highest number of accuracy is also 7. The categorical variables (if used) are applied with one hot encoding. Applying outlier capping (either by IQR method or the standard deviation method), using power transform to fix skew, using PCA actually making the score smaller so they are not used. For the rest of the cells 29-31, I try to use setting those setting that get the best setting to fit with each one of the combination of variable grouping with 2 variable, 3 variable, and 4 variable in one grouping (applied with Python itertools combination). Then, the result will be analyzed with its silhouette score, the number of 0 and 1 it produced, the number of 0 and 1 it has same answer with the answer set that I submit previously with high answer. Note that in this particular homework, I have tried various combination also and it turns out that having high silhouette score won't necessarily output high score also on the final result of Kaggle, so silhouette score is only set as a reference.

It will generate the output file: evaldf_minmax_2.csv, evaldf_minmax_3.csv, evaldf_minmax_4.csv, each with the result of each trial. The number on the filename indicate how many variable exist inside the variable grouping.

- Cell 32
  This cell is the one that makes the model. A lot experiments has been tried such as inputting all of the variables, only picking the numerical, only picking the one that has normal distribution alike, only picking the one that influence genre in general by intuition, removing possible duplicate column by previous correlation EDA conclusion, etc. But they in turn making the final result on Kaggle worse. I have also tried using the Elbow method, where the amount of cluster making elbow will be regarded as the answer, the cluster that is causing elbow is 5. Furthermore, actually following this number of cluster also doesn't create a high score. Various variable groups have also been tried, ones picking by intuition or the one picked by highest/decent silhouette score, but it also yields bad score. I have also tried the following models, and write simple review regarding each model:
  - KMeans: yields the most decent result
  - Agglomerative: yields decent result
  - Mini KMeans: Similar result to the one of KMeans
  - Affinity Propagation: Memory Overflow, since we have too many data points. I have also tried using sampling (remove random points, or remove the one with outlier detected), but it creates worse result than that of KMeans
  - DBScan: Estimating the parameter is hard since there is no specific distinct pattern in the data visualization.
  - Spectral: Similar to the problem with affinity propagation
  - OPTICS: Create a really bad result, with most of them are detected to be one cluster
  - Bisecting KMeans: Yields decent result of that of KMeans
  - BIRCH: Similar problem with that of OPTICS
  - KPrototypes: Yield decent result, converge a long time, but it still yield a result worse than that of KMeans with categorical variable one hot encoded.

The pipeline of the processing will be as follows:

1. Pick the variable that we want to pick for model training.
2. If we desired to do outlier capping, then do capping first (either using the outlier defined by IQR or Standard Deviation).
3. If we desired to sample the dataset, either by random or by dropping the outlier (it is deprecated, and I don't further develop it, since methods correlated to outlier experimented bad result score), then drop the data points here (points appear on the test data must not be dropped).
4. Separate the data that belongs to numerical and categorical types respectively.
5. If we desire to fix the skew of the numerical data, use the PowerTransform of sklearn.
6. If we desire to scale the dataset, either by StandardScaler, MinMaxScaler, or RobustScaler, do so at this point (do so for categorical data. In the end if we do one hot encoding, it can still work since same label integer will have same scaled number).
7. For the numerical dataset, if we want to do PCA, then do so at this point.
8. For the categorical dataset, if we want to do one hot encoding, do so at this point.
9. Gather the transformed numeric and categorical dataset.
10. Pick the desired model and feed the hyperparameters.
11. From now on, it will branch to three different pipeline regarding the choice wanted for the kernel:
    - The first method called the "GreedyMode", incrementally selects feature greedily that causes the highest silhouette score. It first picks only single feature that creates high silhouette score, then examine if there exist using column group of that feature + 1 new other feature will create better silhouette score, if yes then add that variable and reexamined by adding another feature, if no then stops the algorithm immediately. This feature is deprecated because high silhouette score doesn't necessarily create high score in Kaggle.
    - The second method is called the "ElbowMode", where the dataset will be fit with kmeans of cluster number 1 to 13. Then the inertia and silhouette will be plotted to the screen. This feature is also deprecated because it emerges the best cluster as either 3 or 5, which doesn't follow the criteria of the problem which said to be 7-13.
    - The final method is essentially the regular fit and output csv file for submission.

Two different dictionary exist in the cell that plays the crucial role of model customization, which are called datasetConfig and also kernelConfig. The former is related to how the dataset will be preprocessed without feeding into the algorithm, the latter is related to how the pipeline setting for the data to be fed (the three mode selection on pipeline 11) and also regarding the hyperparameter for a few model. The following two tables will explain the setting meaning of the two dictionaries.

key name of datasetConfig	meaning and format
OutlierChoices	whether outlier wanted to be capped. Two modes exist, CAP_STD for the threshold of the outlier using standard deviation, or using CAP_IQR for the threshold of the outlier using interquartile range. Set to None for no outlier capping.
SamplingChoices	whether we want to sample the dataset (out of the dataset that doesn't appear on the test set). Set to None if we don't want to sample, or set to integer to define how many sampling amount desired.
columnGroups	the column group of variables desired, which defined in the first cell
skewFix?	Boolean, True or False whether PowerTransform wants to be applied to the numeric data.
OneHotEncoding	Boolean, True or False whether one hot encoding wants to be applied to be applied to the categorical data.
Scaler	"Standard", "MinMax", or "Robust" for the scaler used for the dataset. Set to None for no scaling.
PCA?	None or integer consisting number of components wanted as the result of PCA.

key name of kernelConfig	meaning and format
OptunaMode	deprecated, not used, set always to False
GreedyMode	Boolean, whether GreedyMode in pipeline 11 which is explained before is desired.
ElbowMode	Boolean, whether ElbowMode in pipeline 11 which is explained before is desired.
FinalPredict	Boolean, if GreedyMode and ElbowMode are both False, which means that the pipeline 11 will enter the model construction. Set this to True if after model construction, the submission csv for Kaggle wants to be created, otherwise set to False.
Model	Desired model for model construction, available choices are: - KMEANS - AGG (Agglomerative)

	- MINIKMEANS - AFFINITY - DBSCAN - SPECTRAL - OPTICS - BISKMEANS - HDBSCAN - BIRCH - KPROTOTYPES - GMM (Gaussian Mixture Model) - ENSEMBLE2 (Creates two models, which are Kmeans and Agglomerative, final output will define two points are one cluster if in either model they are detected as one cluster). - ENSEMBLE3 (Creates three models, which are Kmeans, Agglomerative, BisectingKMeans. final output will define two points are one cluster if in any model they are detected as one cluster). - ENSEMBLE3VOTE (Creates three models, which are Kmeans, Agglomerative, BisectingKMeans. final output will define two points are one cluster if in two out of three model they are detected as one cluster). - ENSEMBLE4 (Creates four models, which are Kmeans, Agglomerative, BisectingKMeans, and Gaussian Mixture Model. Final output will define two points are one cluster if in any model they are detected as one cluster).
PredefinedNeighbor	The number of neighbor desired, integer. That is, if the used model require neighbor number as hyper parameters.
minimalDistanceAGG	If the model selected is AGG, and also the PredefinedNeighbor is set to None, the AGG will not define the number of predefined neighbor as input, instead it

	will put this integer as the distance_threshold hyper parameter.
mixtureComponentGMM	Integer that is fed for Gaussian Mixture, for the n_components hyper parameter.

Now, I will copy the two model setting used for final submission of the Kaggle, along copy paste the comments inside the cell of what the setting of the model.

- First Model

```
datasetConfig = {
            "OutlierChoices":None, #None, CAP_STD,CAP_IQR,
DROP_IQR,DROP_STD,DROP_CAP_STD,DROP_CAP_IQR. CAP IQR
IS NOT A GOOD CHOICE!
            "SamplingChoices":None, #None, or int, 0 for target only
            "columnGroups": '3VarBest',#"columnGroups":
'XAllDefaultNoLoudnessAuthenticityPitchNumericOnly',
           "skewFix?":False,
            "OneHotEncoding?":True,
           "Scaler": "MinMax", #"Standard",
            "PCA?":None, #None or int for the number of components
           }

kernelConfig = {"OptunaMode":False,
                "GreedyMode":False,
                "ElbowMode":False,
               "FinalPredict":True,
               "Model":"ENSEMBLE4", #KMEANS, AGG,
MINIKMEANS,AFFINITY,DBSCAN, SPECTRAL, OPTICS, or
ENSEMBLE2(KMEANS, AGG),
                #ENSEMBLE3(KMEANS,AGG,BISKMEANS(from external file
since on kaggle not available)),
                #ENSEMBLE3VOTE(same with ensemble 3 but is using
majority vote to get answer, not using or of all answer)
                #ENSEMBLE4(KMEANS,AGG,BISKMEANS(from external file
since on kaggle not available), GMM)
                #BISKMEANS(version of sklearn not
compatible),HDBSCAN,BIRCH,KPROTOTYPES
                "PredefinedNeighbor":7,
                "minimalDistanceAGG":21,
                "mixtureComponentGMM":7,
              }
```

This model uses ensemble of four models: KMeans, Agglomerative, Bisecting Kmeans, and Gaussian Mixture Model, with predefined neighbor and mixture component all set to 7. No sampling and outlier capping used, as it makes the result worse. The columnGroups used is 3VarBest, which is the best combination of variables experimented on the few cells above it where they tried to generate various variable combination (three variables consist of PlayTime, Vocal, and InstsProportion). No SkewFix and PCA since by experiment it makes the score lower.

- Second Model

```
datasetConfig = {
            "OutlierChoices":None, #None, CAP_STD,CAP_IQR,
DROP_IQR,DROP_STD,DROP_CAP_STD,DROP_CAP_IQR. CAP IQR
IS NOT A GOOD CHOICE!
            "SamplingChoices":None, #None, or int, 0 for target only
            "columnGroups": 'PopularityOnly',#"columnGroups":
'XAllDefaultNoLoudnessAuthenticityPitchNumericOnly',
           "skewFix?":False,
            "OneHotEncoding?":True,
           "Scaler": "MinMax", #"Standard",
            "PCA?":None, #None or int for the number of components
           }

kernelConfig = {"OptunaMode":False,
                "GreedyMode":False,
                "ElbowMode":False,
               "FinalPredict":True,
               "Model":"KMEANS", #KMEANS, AGG,
MINIKMEANS,AFFINITY,DBSCAN, SPECTRAL, OPTICS, or
ENSEMBLE2(KMEANS, AGG),
                #ENSEMBLE3(KMEANS,AGG,BISKMEANS(from external file
since on kaggle not available)),
                #ENSEMBLE3VOTE(same with ensemble 3 but is using
majority vote to get answer, not using or of all answer)
                #ENSEMBLE4(KMEANS,AGG,BISKMEANS(from external file
since on kaggle not available), GMM)
                #BISKMEANS(version of sklearn not
compatible),HDBSCAN,BIRCH,KPROTOTYPES
                "PredefinedNeighbor":3,
                "minimalDistanceAGG":21,
                "mixtureComponentGMM":7,
              }
```

This model uses KMeans, with number of neighbors 3. Although it is defined that the number of neighbors are around 7-13, but this model yield a good score. No sampling and outlier capping used, as it makes the result worse. The columnGroups used is PopularityOnly (only using PlayTime feature). No SkewFix since the graph is already similar to normal distribution and also no PCA since it is only 1 feature.