UNIVERSITÀ DI PISA
DEPARTMENT OF COMPUTER SCIENCE
Data Science and Business Informatics

Data Mining I Project

The Glasgow Norms

Simone Di Luna 544322

29/01/2022

Contents

1. Data Understanding and Preparation

The linguistic norm can be defined as a set of rules which cover all levels of the language—phonology, morphology, syntax, vocabulary, textuality—accepted by the vast majority of speakers and writers in a given period and cultural context (Giovanardi 2010).

The Glasgow Norms are a set of normative ratings for 5553 English words on nine psycholinguistic dimensions: arousal, valence, dominance, concreteness, imageability, familiarity, age of acquisition, semantic size, and gender association. Furthermore, the dataset under investigation presents three more attributes: length, polysemy and web corpus frequency.

Data from the original study (Scott et al. 2019) was collected by a group of researchers at the University of Glasgow through an experiment involving 829 native English-speaking students from the same university. Participants voluntarily decided to take part in the experiment, attracted by academic or monetary rewards.

The sample has some several problems: it is non-probabilistic and presents self-selection bias, hence it is a non-representative sample. Moreover, the interviewees are all young students, predominantly female, and, presumably, of upper middle class. Therefore, regardless of the analysis that is intended to be carried out on the sample, it is not possible to make inference on any population: neither that of college students, nor that of Glasgow, let alone that of native English speakers.

All attributes in the dataset are quantitative. Each participant was randomly assigned a list of words to be evaluated: some contained 101 words, others 150.

The nine psycholinguistics variables (henceforth PSY attributes) are continuous since each word in the dataset is given by the average, approximately, of the ratings of 33 different participants. The *polysemy* variable is binary, while the remaining two, *length* and *web corpus frequency*, not being the result of individual evaluations, are discrete. Therefore, the dataset can be considered as a *data matrix*.

1.1 Data Semantics

The first three dimensions were assessed by individual participants on a scale of integers from 1 to 9. Typically, these attributes are used in psycholinguistics to characterize the emotional impact of a word (ibid.):

Arousal (AROU): it defines the degree of emotional excitement the word conveys. With an average score of 2.057, *dull* is the word that conveys the lowest emotional arousal in the dataset. Instead, the highest level of *arousal*, 8.18, was found in the word *passionate*.

Valence (VAL): it defines the degree of positivity of a word. In the dataset, the word with the lowest valence value, 1.03, is *rape*. On the other hand, the highest valence is contested between the words *love* and *peace*, both with a value of 8.647.

Dominance (DOM): it defines the level of dominance that a word provokes. With a value of 1.941, the word with the lowest *dominance* is *powerless*, while the word with the highest value of *dominance*, 8.371, is *dominate*.

The remaining PSY variables were evaluated by individual participants on a scale of integers from 1 to 7:

Concreteness (CNC): it is a measure of how concrete a word is. Specifically, it represents how much something can be perceived by our senses. The most abstract word in the dataset is *infinite*, with a value of 1.636. The most concrete is *bridge*, with a value of 6.938.

Imageability (IMAG): it measures how easy something is to imagine. The word with the lowest *imageability*, 1.737, is *quiescent*. *Carrot*, instead, with a value of 6.941, is the easiest word to imagine.

Familiarity (FAM): it is a measure of how familiar something is and in particular, how much a word is close to subjective experience. With a value of 1.647, the least familiar word is *zephyr*. Both with a value of 6.939, the two most familiar words are: *music* and *university*.

Age of acquisition (AOA): it measures the estimated age at which a word was first learned. For the first 6 values, the scale is defined as a series of consecutive 2-years periods, from the age of 0 to 12 years; a score of 7, on the other hand, indicates a period of time ranging from 13 years onwards. The word *mum* it is the first to be learned with a value of 1.219, whereas *Twitter* with a score of 6.971, is the latest word learned by the participants.

Semantic size (SIZE): it is the measure of the physical or conceptual dimension of what the word refers to. *Crumb*, with a value of 1.375, is the "smallest" word, while *universe*, with 6.912, is the "biggest" one.

Gender association (GEND): it measures the degree of "masculinity" of a word. With a value of 1.0, the most feminine word is *lady*. *Male*, instead, with a value 6.971, it turned out be the closest word to the masculine gender.

In addition to the nine attributes just presented, the dataset provides three more variables:

Length (LEN): it indicates from how many letters a word is formed. The words *tv* and *up*, with only two letters, are the shortest. *Intercontinental*, instead, with 16 letters, is the longest word.

Polysemy (POLY): it is a binary attribute that indicates whether, in the dataset, the word can have more than one meaning. In particular, the dataset contains:

- 289 words with two meanings;
- 69 words with three meanings;
- 19 words with four meanings;
- 2 words with five meanings.

In total the ambiguous words are 379, while the non-ambiguous words are 871. Since only the 8.81% of the records present a value of *polysemy* equal to 1, it can be considered an asymmetric binary attribute. This is also known as the *class imbalance problem*, and we will discuss it extensively in section 3.

Web Corpus Freq (FREQ): is the absolute frequency of a word in the Google Newspapers Corpus. The least and most frequent words are, respectively, *enthrall* and *all*.

Here, the original dataset of 5553 words is presented into two distinct sets: the first one contains 4682 unambiguous words whereas the second is formed by the remaining 871 non-ambiguous words (the latter does not contain LEN, FREQ and POLY attributes). For the moment the analysis will be centered on the first dataset.

1.2 Assessing and improving data quality

Psycholinguistic varaibles are syntactically accurate, since data are all in the range $[1, 9]$ – with respect to dimensions: *arousal*, *valence*, *dominance* – and in the range $[1, 7]$ – the others. Furthermore, data are collected through a dedicated web portal and since there are no qualitative variables there can be no transcription errors. Besides, given the absence of duplicates, the dataset is free from redundancies.

In addition, for each dimension, the semantic accuracy of the words that received the highest scores and those that received the lowest scores were evaluated. For instance, considering the *arousal* attribute, the five terms with the worst scores are: rape, genocide, cancer, racist and murderer; whereas those with the highest scores are: happiness, trustworthy, loving, peace and love. In general, all of the scores assigned (the inspected ones) appear to be consistent. The only oddity found concerns the word "building". It received an average score of about 1.94 on the *familiarity* dimension. The Cambridge Dictionary defines it as "a large building, especially an impressive one". Perhaps this is due to the fact that there are few buildings with these characteristics in Glasgow. Finally, the real length of the words corresponds to that declared in the *length* field of the dataset.

As regards the completeness of the attribute values, the Psycholinguistic variables do not have missing values. The only variable affected by this issue is *web corpus frequency*, with 14 missing values and thus an irrelevant overall weight on the dataset, equal to approximately 0.3%.

Instead, the presence of missing values indicated with different characters should be excluded: the domain of the attributes has been checked and there are no negative values nor special characters.

The detected missing values seems to be of MCAR type, i.e. missing completely at random and therefore independent from other observable variables or unobservable parameters. This impression is also supported by the fact that, as already noted, they only have a negligible impact on the dataset.

As can be seen, almost all of them seem to be fairly common words, therefore, these are presumably high frequency terms: burgle, Christmas, Dad, Dame, Facebook, FALSE, Mom, Mum, Mummy, skijump, TRUE, TV, Twitter, yo-yo.

A solution to this problem might be to simply delete these records or try interpolating the missing data with linear regression. The latter option seems to be interesting because the *web corpus frequency* attribute appears to be moderately correlated with some variables of the dataset (Table 1). The interesting thing is that it is a monotone non-linear correlation. This can depend on several factors such as: the skewness of the distribution and the presence of numerous outliers. Indeed,

Table 1. *Web corpus frequency* correlations

	LEN	AROU	VAL	DOM	CNC	IMAG	FAM	AOA	SIZE	GEND	POLY
Pearson	-0.158	-0.010	0.137	0.103	-0.038	-0.056	0.278	-0.229	0.081	-0.037	0.101
Spearman	-0.244	0.069	0.300	0.223	0.025	0.023	0.580	-0.413	0.161	-0.075	0.194

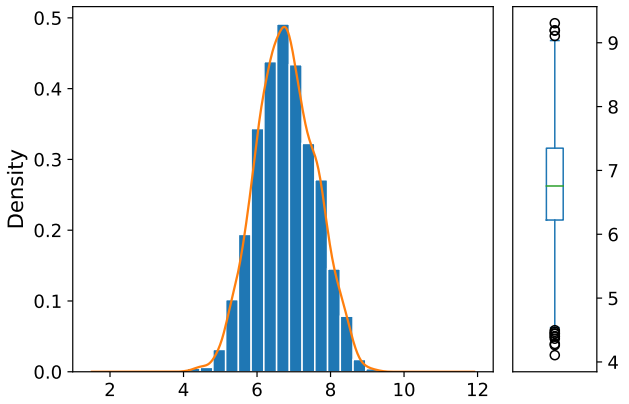

Figure 1. *Web corpus frequency* distribution after logarithmic transformation.

the *Spearman correlation* between two variables is equal to the *Pearson correlation* between their rank values, thus it is less sensitive to outliers. This analysis will be performed later in this chapter.

Variable transformations

The huge difference between average (5 702 981.5) and median (29 889 759.53) confirms that the attribute *web corpus frequency* presents a lot of outliers and that its distribution is very asymmetric. The statistical tool that will be used to measure the distortion of each distribution is the *Fisher-Pearson* standardized third moment adjusted by a factor that accounts for sample bias (Joanes and Gill 1998):

$$\mathrm{S}[X] = \frac{\sqrt{n(n-1)}}{n-2}\mathrm{E}\left[\left(\frac{X-\bar{x}}{\sigma_X}\right)^3\right] \qquad (1)$$

where $\bar{x}$ and σ_x are, respectively, the sample mean and the standard deviation.

For unimodal distributions usually hold the following properties: negative values of skew indicate that the left tail of the distribution is longer than the right, whereas positive values suggest the opposite. The normal distribution is perfectly symmetrical, thus its *skew* equals 0. The *web corpus frequency* attribute has a very high and positive distortion, equal to 9.49, indeed its right tail is very long.

Neither the traditional *z-score* nor its theoretically more robust variants are able to reduce the outliers of the *web corpus frequency* attribute. For this type of distributions with positive skew a particularly suitable transformation is the logarithmic one. After the base 10 log-transformation the *skew* of the *web corpus frequency* variable is equal to 0.04 and the distribution assume a pseudo-normal shape (figure 1). In addition, the extreme values have been reduced significantly. Furthermore, can be shown that the *Pearson* and *Spearman* correlation coefficients now are almost equivalent.

The *familiarity* attribute has a *skew* value equal to -0.77 and thus it is slightly biased to the left. By applying a cubic transformation to it, the distortion can be almost totally eliminated (*skew* = 0.01). Figure 2 compares the histogram of the *familiarity* attribute before and after the transformation. For ease of comparison, data were normalized in the range [0, 1] by means of the min-max method.

The *arousal* variable, instead, suffers from a slight positive distortion (*skew* = 0.42). This bias can be compensated through the application of a base 2 log-transformation (*skew* = -0.11). As for the *familiarity* attribute, the figure 3 shows the result of such transformation.

The skewness of all the psycholinguistic variables are summarized in table 2.

1.3 Distribution of the variables and statistics

A helpful tool to analyze the distributions of the variables and detect potential outliers is kurtosis. It is defined as the expected value of the standardized data raised to fourth power. The only observations that contribute significantly to kurtosis are those very far from the mean, that is outliers. The functions

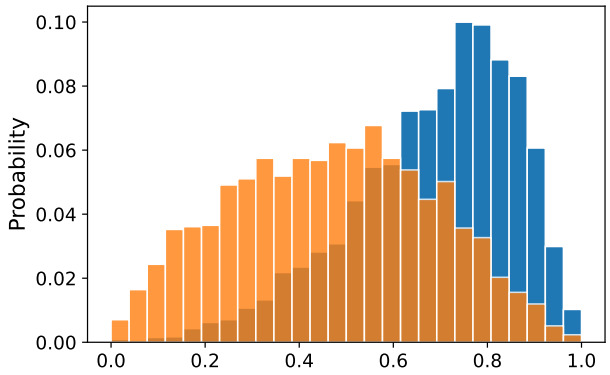

Figure 2. *Familiarity* distribution before (blue) and after (orange) the cubic transformation.

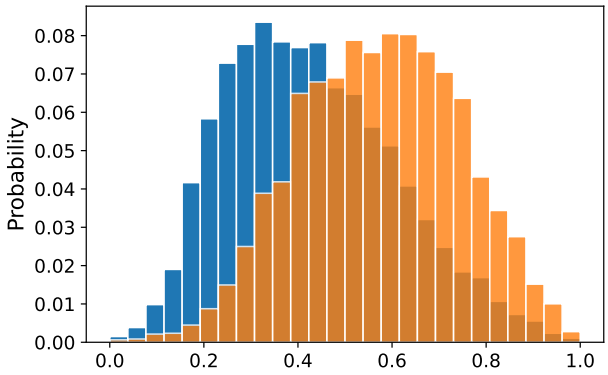

Figure 3. *Arousal* distribution before (blue) and after (orange) the a log-transformation.

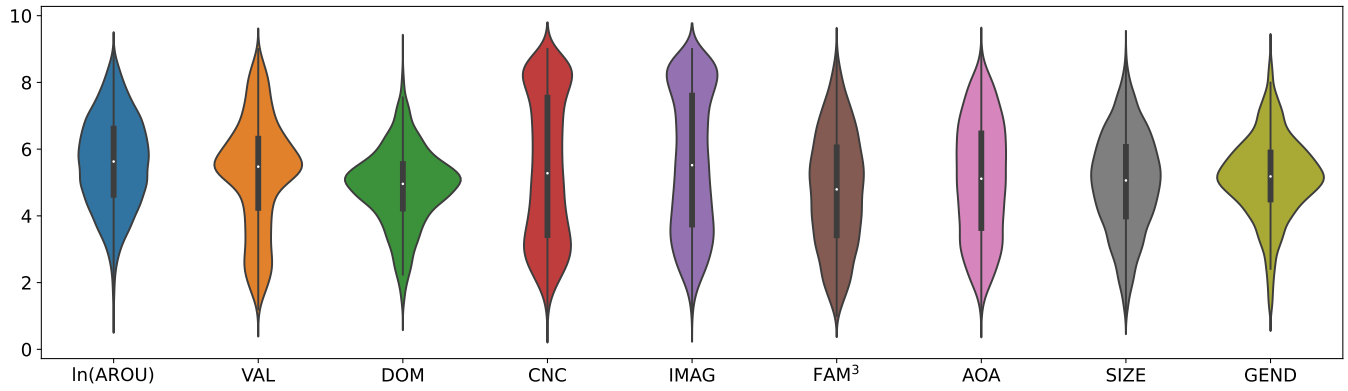

Figure 4. Violin plots of min-max normalized PSY variables

used are the followings (Joanes and Gill 1998):

$$\mathrm{K}[X] = \mathrm{E}\left[\left(\frac{X-\bar{x}}{\sigma_X}\right)^4\right] - 3 \tag{2}$$

$$\bar{\mathrm{K}}[X] = \frac{n-1}{(n-2)(n-3)}\left[(n+1)\mathrm{K}[X]+6\right] \tag{3}$$

where the equation (3) is the unbiased form of the equation (2). Thus defined, the kurtosis of the normal distribution is equal to 0 (Fisher's definition). A negative kurtosis indicates a platykurtic distribution, characterized by few outliers; while a positive kurtosis is typical of leptokurtic variables, characterized by a higher number of outliers.

To detect outliers, LEN, POLY and FREQ were not taken into account as the set of words in the original study was defined in the design phase of the experiment where these information were already known.

From table 2 we can note that for most attributes kurtosis is low, an indication that outliers should not be a major problem. Indeed, this is supported by the last row of the table that show the percentage of outliers over the whole dataset as computed in the boxplot (see leter). Also, looking at the violin plot in figure 4 we can observe the shape of the distributions of the PSY variables; they are min-max normalized in the range $[1, 9]$.

The attributes ln(AROU), VAL, FAM^3, AOA and SIZE have a pseudo-normal course and a negative kurtosis, hence their distributions are platykurtic. These type of distributions present a number of outliers lower than a Gaussian distribution, for which we expect outliers to represent, approximately, 4.8% of all observations.

CNC and IMAG accentuate these characteristics even more, because in addition to being platykurtic, they have an almost uniform distribution (to be precise they have bimodal symmetric distributions as also suggested by the null skewness). Not surprisingly, these types of distributions do not exhibit outliers, however, they have a wider variance than the other variables which are characterized by a smaller dispersion around the mean.

Finally, we can observe that there are only two leptokurtic variables: DOM and GEND. As expected, these are the attributes that contain the most extreme observations. Nevertheless, the percentage of outliers is still less than 4.8% of the Gaussian distribution and this, probably, is due to the fact that their distributions are not perfectly normal.

Outlier detection for single attributes

A way to detect potential outliers is the z-score method. To apply this technique, first we need to normalize our data with the standard score. Considering the dataset as matrix data with dimension $n \times m$, then each value $|z_{ij}| > k$ with $1 \leq i \leq n$, $1 \leq j \leq m$ can be considered as an outlier. The threshold k is usually taken in the set $\{2.5, 3, 3.5\}$.

Following this strategy, for each of the thresholds in the set above, it was found that:

- 230 rows (5 of which with missing values) with at least one value greater than 2.5;
- 38 records (3 of which with missing values) with at least one value greater than 3;
- 1 row with at least one value greater than 3.5.

Table 2. PSY variables statistics

	ln(AROU)	VAL	DOM	CNC	IMAG	FAM^3	AOA	SIZE	GEND
Std dev.	1.37	1.67	1.16	2.16	2.10	1.72	1.74	1.48	1.22
Kurtosis	-0.44	-0.44	0.29	-1.36	-1.29	-0.79	-0.97	-0.44	0.60
Skewness	-0.11	-0.30	-0.26	0.03	-0.03	0.01	-0.04	-0.17	-0.25
Outliers	0.17%	0.04%	2.84%	0.00%	0.00%	0.00%	0.00%	0.00%	3.50%

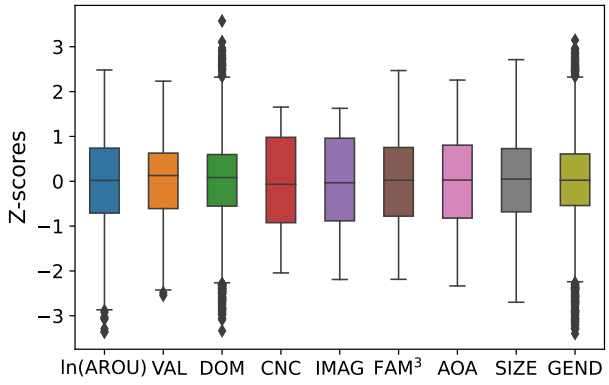

Figure 5. Each psycholinguistic attribute is associated with its own boxplot. Variables are standardized.

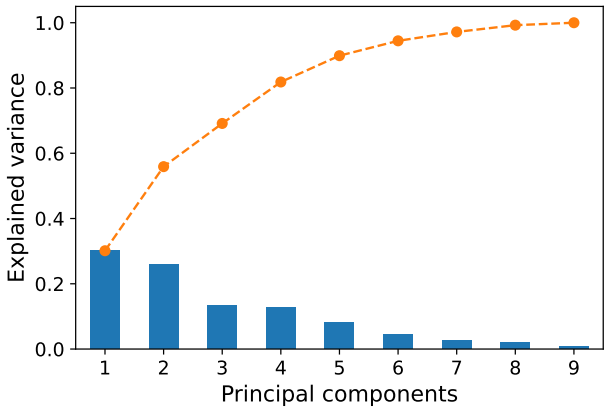

Figure 6. Individual component (in blue) and cumulative (in orange) explained variance.

Even if we deemed these values as outliers and removed them, the dataset would still show extreme values. Indeed, after their elimination, the positional statistics would be altered. Namely, some values that were not previously considered as outliers, now become so. The solution to this problem is to iterate the above strategy until all outliers are removed. For example, imposing k = 3, the algorithm needs five iterations to find all (57) extreme values. However, in literature this is not considered a good practice,one should stop after the first iteration.

Outliers can also be detected by means of box plots. This visual tool enables us to understand which distributions have extreme observations and also to get an idea of their frequency and magnitude.

From the box plot in Figure 5 can be seen that the attributes with outliers are four: AROU, VAL, DOM and GEND. Generally speaking, for each variable X_i in the dataset, with $1 \leq i \leq m$, the box plot marks all values in the following set as outliers:

$$\{x_{ij} \in X_i \mid x_{ij} < Q_1 - 1.5\,\mathrm{IQR} \ \vee \ x_{ij} > Q_3 + 1.5\,\mathrm{IQR}\}$$

where IQR is the interquartile range, i.e. $\mathrm{IQR} = Q_3 - Q_1$. Applying this method to the Glasgow norm dataset, 303 extreme values were detected.

As with the z-score method, this strategy has the drawback that it needs to be iterated several times to detect all the extreme values. Indeed, every time the algorithm is used, the dataset shrinks, hence the positional statistics used to compute the outliers in the previous step also change. Thus, it is necessary to normalize the dataset at each iteration. In this way, after five iterations, 404 records with one or more extreme values were found. The recursive procedure is not a good practice here too. Nevertheless, even if it was, the results obtained in terms of domain restrictions and variance reductions are marginal, except for attributes with many outliers, that is DOM and VAL.

Outlier detection for multidimensional data

Expressly for multidimensional data, outliers can be detected using the principal component analysis (PCA). The objective is to reduce to two or three attributes the dimensionality of the dataset and then to perform a visual inspection using the scatter plot (Berthold et al. 2020, pp. 68–69). There is a trade-off between dimensionality reduction and data quality, therefore for good results, the former must not worsen the latter too much.

Despite the many tests carried out, the cumulative explained variance is not high enough with 2 or 3 principal components. As shown in the scree plot in figure 6, the first two principal components, jointly, account for only 56% of the total variance, whereas the first three components, explain around 69%. Those values do not appear satisfactory for an accurate representation of the data.

The results were obtained using the standardized dataset with all the original records, but even using the dataset without the outliers discovered in the previous section, there were no significant improvements.

For multidimensional data, another technique that can be used to detect outliers is DBSCAN, the density-based clustering algorithm. Since it uses the concept of noise points, this algorithm appears to be particularly suitable to discover extreme values. After setting the two hyperparameters, ε (the radius) and the $minPts$, the algorithm categorizes each point as either a core or a border or a noise point. Core points are all the objects that within their ε-neighbourhood contains at least $minPts$ points (the point itself is counted). Objects that are not core points but are in the neighbourhood of a core point are border points. Finally, objects that are neither core nor border points are considered outliers and do not join any clusters.

This algorithm will be discussed in more detail in section 2.3; for now it is essential to note it does not work well on this dataset. Indeed, no matter how we set the hyperparameters, for each clustering the algorithm discovers either a single cluster or few clusters with many outliers. More then 7920 configurations were tested on a min-max normalized dataset in

Table 3. DBSCAN results

ε	$minPts$	n. clusters	n. noise points
2.9	10	2	15
3	20	2	14
3.2	20	2	7
3.55	60	2	2
3.55	100	2	6
4.1	450	2	4
4.2	400	2	3

the range $[1, 9]$; some with the higher silhoutte scores (actually the one with $\varepsilon \geq 3.55$) are summarized in table 3 (note that the set of noise points is considered as a cluster).

For instance, with $\varepsilon = 3.55$ and $minPts = 100$ the following words are considered as outliers: fuck, Mum, tsunami, university, wank and war.

For the sake of completeness, for all the analyses carried out in this project, both the original and the cleaned version of the dataset were used. Nevertheless, no interesting findings were observed. For this reason, unless otherwise specified, data refers to the entire dataset.

Linear regression to predict missing values

To try to predict the missing values of Log(FREQ) we will use a linear regression on the standardized data. However, some variables in the dataset are strongly correlated with each other and this would lead the model regressors to be, albeit imperfectly, multicollinear. Consequently, the estimated coefficients would be biased, characterized by a high standard error.

This problem was addressed by using a random partitioning of the standardized dataset, which divided it into two parts: 80% intended to train the model and the remaining 20% to assess the reliability of the predictions.

Before presenting the results, some useful statistics are introduced. The definition of Residual Sum of Squares is $\text{RSS} = \sum_i^n e_i^2$, where each error is the difference between true and predicted values, that is $e_i = y_i - \hat{y}_i$; whereas the Total Sum of Squares is defined as $\text{TSS} = \sum_i^n (y_i - \bar{y})^2$. The coefficient of determination, $\bar{\text{R}}^2$ (aka adjusted r-squared), is the goodness of fit and it tells us how well the regression function fit the data. It is equal to $1 - \frac{\text{RSS(n-1)}}{\text{TSS(n-k)}}$ (k is the degree of freedom, in this case the number of regressors) and it varies between 0 and 1. The RMSE instead is equal to $\sqrt{\frac{\text{RSS}}{n-k}}$ and represents the standard deviation of the unexplained variance; it can be seen as a measure of how well the model explain a given set of observations.

Overall the model performs almost identically on both training and test sets, thus there should be no overfitting problems. Specifically, the adjusted r-squared is 0.51 for the training set while 0.47 for the test set. Instead, regarding the RMSE, they respectively received a score of 0.70 and 0.72.

The same analysis was conducted on the dataset without extreme values but the results are nearly the same.

Eventually, the estimated regressors obtainend form the model were used to predict the missing values of the web corpus frequency attribute; the result is shown in table 4. As suspected, excluding *burgle*, *dame*, *skijump* and *yo-yo*, the others are all high-frequency words. Given that the predicted values seem reasonable, these new pieces of information were incorporated into the dataset.

Table 4. Predicted values of Log(FREQ)

burgle	6.43	Mum	8.45
Christmas	7.71	Mummy	7.44
Dad	8.61	skijump	6.48
Dame	6.36	TRUE	8.06
Facebook	7.49	TV	7.85
FALSE	7.22	Twitter	7.06
Mom	8.24	yo-yo	5.99

1.4 Correlations analysis

In subsection 1.2 correlations concerning the *web corpus frequency* attribute were already discussed. In the following, the relations involving the other non-binary variables are explored.

After the transformations applied to the attribute *arousal* and *familiarity*, the Pearson and Spearman correlation coefficients appear almost equivalent for all the variables. Indeed, since Spearman's correlation is rank-based, monotone transformations have no effect on the coefficients (for Pearson's correlation this is only true for linear transformations). Subtracting the two correlations matrices, Pearson exhibits slightly higher values on almost all pairs of variables. However, these are tiny differences, less than 100 basis points, except for three pairs of attributes with a maximum peak of 200 basis points. Nevertheless, Spearman shows a more significant correlation for the attributes VAL and GEND (these differences ranging from 250 to 550 basis points). For these reasons (and for space-economy), only the latter matrix is displayed (see the heatmap in figure 7).

Looking at the heatmap, the prevalence of cool colors suggests that negative correlations are more common than positive ones. The relations characterized by a relatively high Spearman's coefficient ($r_s > 0.5$) are:

- CNC and IMAG, with $r_s = 0.9$. This implies that the more concrete a word is considered, the easier it is to imagine. At the same time, words easier to visualize are more concrete.
- VAL and DOM, with $r_s = 0.70$. It suggests that words conveying a higher degree of positivity generally provoke a higher level of dominance. The opposite is also true.
- FAM and AOA with $r_s = -0.67$. The words closer to one's subjective experience are those that are generally learned at a young age. Moreover, this relation also

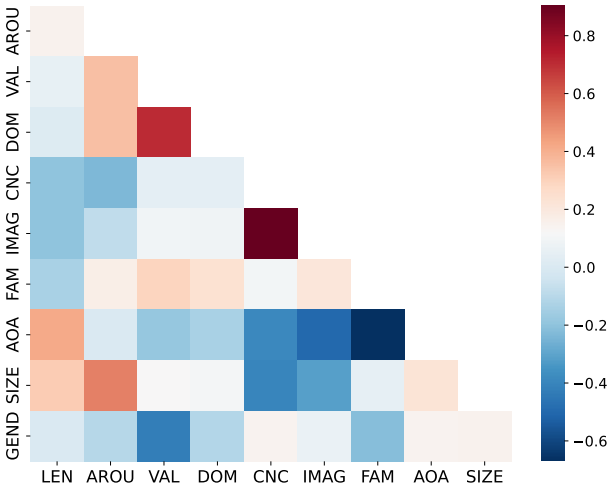

Figure 7. Heatmap—Spearman's correlations.

suggests that the more unfamiliar a word is, the higher the age at which it is learned.

- AROU and SIZE, with $r_s = 0.51$. This means that words conveying a higher degree of emotional excitement are those physically or conceptually larger, and *vice versa*.
- IMAG and AOA, $r_s = -0.50$. Finally, the easier a word is to imagine, the smaller, usually, the age of acquisition, and the other way around.

The hexagonal bin plot in figure 8 depicts the above correlations and, in addition, shows the linear regression function. It was used in place of the scatter plot because the latter does not work well when the number of objects to display is too high: points start overlapping, and the file size increases problematically. Objects are binned into hexagons, whereas colors distinguish denser regions from less dense ones. The figure shows that, despite the dispersion, areas with high density are those near the regression line. The quadratic fit was also tested, nonetheless for crucial correlations, it performs almost identically to the linear one (due to visualization issues, the quadratic curves were not plotted).

Finally, it is necessary to note the presence of several pairs of attributes characterized by a lower but not negligible correlations, ranging from 0.49 to 0.32. This is particularly true for the following variables: VAL and GEND; CNC and SIZE; LEN and AOA; CNC and AOA; AROU and DOM; AROU and VAL; IMAG and SIZE; LEN and SIZE.

1.5 Discretization of non-psycholinguistic variables

For further analysis, it might be helpful to focus on psycholinguistic variables and discretize the others to see how they distribute within clusters. For this purpose, let us focus on the attributes *length* and *web corpus frequency*.

The domain of the LEN variable is the set of integers in the range $[2, 16]$. Its distribution has a slightly positive skewness, with the majority of values concentrated in the range $[4, 8]$.

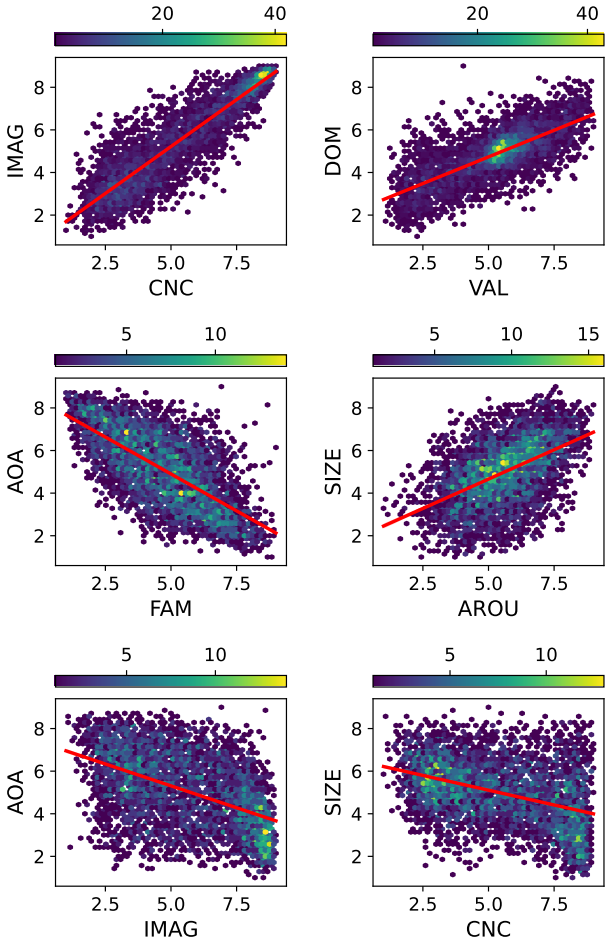

Figure 8. Hexagonal bin plot and linear regression line.

Many unsupervised discretization techniques were tested. The *equal width* approach splits the domain of the attribute into a specified number of intervals, each having the same width. This method performs well only with two nominal categories for the LEN variable. Indeed, more groups generate classes that are too unbalanced. For instance, with three categories, it produces the following groupings: 2713 objects in $[2, 6]$, 1914 in $[7, 11]$, and 55 in $[12, 16]$.

Instead, the *equal frequency* approach tries to put the same number of objects into each interval. However, this causes the groups to have almost the same cardinality, for this reason it is not suitable for highlighting interesting correlations. With three categories, it generates the following partition: 1803 objects in $[2, 5]$, 1687 in $[6, 7]$, and 1192 in $[8, 16]$.

Lastly, the k-means algorithm was used. Precisely, it was tested both stochastic—k-means++ algorithm—and deterministic initialization—with the initial centroids determined starting from equal width intervals. The former method has a better Total Sum of Squared Error (SSE) for every number of clusters; for example, with $k = 3$, the former has an SSE = 2857 while the latter has an SSE = 3551. Three clusters (bins) seem a quite natural choice for this attribute (this is also supported by the

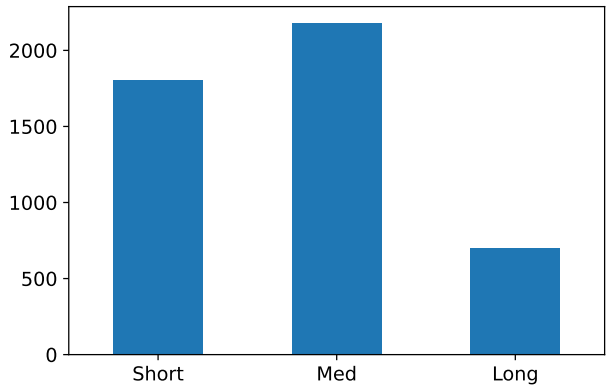

Figure 9. LEN discretization.

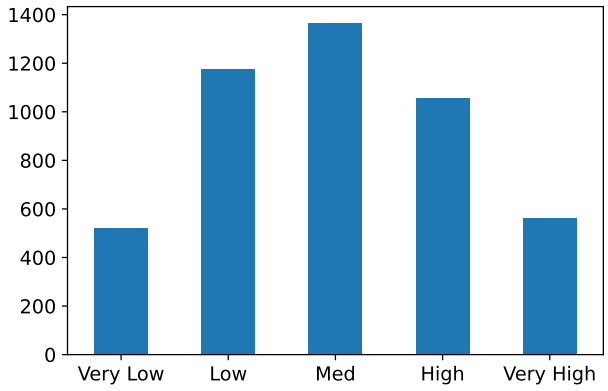

Figure 10. Log(FREQ) discretization.

elbow method, not shown here to save space). Hence, objects was grouped in short, medium, and long words. Stochastic initialization produced the following bins (see figure 9): 1803 in $[2, 5]$, 2179 in $[6, 8]$, and 700 in $[9, 16]$.

Regarding the attribute Log(FREQ), the same consideration holds. Its demain is composed by the real numbers in the interval $[4.11, 9.31]$. Here stochastic and deterministic initialization produce essentially identical results. Presumably, this happens because, unlike LEN, Log(FREQ) has a pseudo-normal and unbiased distribution (see figure 1). Considering the SSEs and the silhouette scores computed in all tests performed, the numbers of bins that appear most plausible are: 3 and 5. Both are candidate knee points ($SSE_{k=3} = 521$; $SSE_{k=5} = 217$) and also have the same silhouette score (roughly 0.51), therefore the choice is not clear-cut. Eventually, it was decided to discretize the variable Log(FREQ) into five bins: 523 objects in $[4.1, 5.81)$, 1175 in $[5.81, 6.48)$, 1365 in $[6.48, 7.09)$, 1055 in $[7.09, 7.77)$ and 564 in $[7.77, 9.31)$. The result is shown in figure 10.

In the next chapter we will use unsupervised classification trying to figure out if the data in the Glasgow Norms dataset tend to groups according to some specific patterns.

2. Clustering

We are now ready to perform the cluster analysis. For this purpose, the following sections will present the results obtained over the min-max normalized dataset as modified in the previous chapter. Both the standardized and the outliers-free datasets were also analyzed, nonetheless, the differences are—in the broadest sense—not significant. Many clustering techniques were tested: hierarchical; k-means with stochastic initialization; k-means with deterministic initialization, using the hierarchical algorithm as a subroutine; k-means with stochastic initialization followed by hierarchical clustering; DBSCAN. In the next subsection, the stochastic k-means was the first algorithm to be discussed.

2.1 K-means

K-means is a greedy algorithm that uses a partitional, exclusive, prototype-based (center-based) approach. This technique attempts to find a user-specified number of clusters, which are represented by their prototypes (centroids or medoids). After seletcing the initial k-centroids with a stochastic or a deterministic approach, the algorithm assigns each point to the closest cluster, where the closeness is measured with rispect to the prototype object. After all points are assigned, the prototypes are computed again within each cluster. The algorithm stops when prototypes don't change anymore or when an ending condition is reached. Therefore, other then the number of clusters, there are at least three more hyperparameters to set: the number of initialization to attempt, the maximum number of iterations to perform and the level of tollerance accepted to interrupt the algorithm.

For the probabilistic initialization many configurations were tested: twenty models are presented, with a number of clusters ranging from 2 to 20; 100 initializations performed with the k-means++ algorithm (this method tries to find the initial prototypes in such a way that they are far away from each other); a number of iterations capped to 1000 and a tollerance equal to 10^{-9}. The results are displayed in figure 11 and 12: the former shows—for each model ran—the total SSE, while the latter shows the silhouette score, an overall statistic summarizing both cohesion and separation measures. A good clustering is caracterized by a low SSE and a high silhouette score (it may assume all real values in the range $[-1, 1]$).

Both the graphs can help us to find the "optimal" number of clusters for the k-means algorithm. Using the elbow method on figure 11, it seems reasonable that the ideal number of clusters is between 3 and 8. The SSEs are high due to the min-max normalization in the range $[1, 9]$; the same analysis conducted on the standardized dataset shows halved values, however the shape of the curve does not change. Instead, looking at figure 12, we can notice that the silhouette score immediatly drops after computed the first model with two clusters. The dataset without 404 extreme values computed in section 1.3 even exhibit a lower score equal to 0.251 and the same occur

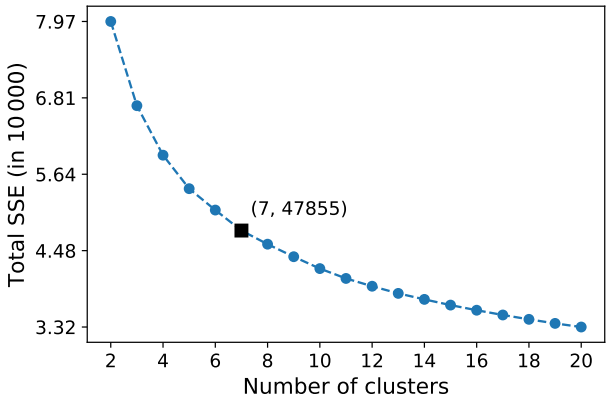

Figure 11. SSE versus number of clusters. The black square represents the knee point as computed by the *KneeLocator* function provided by the *kneed* Python library.

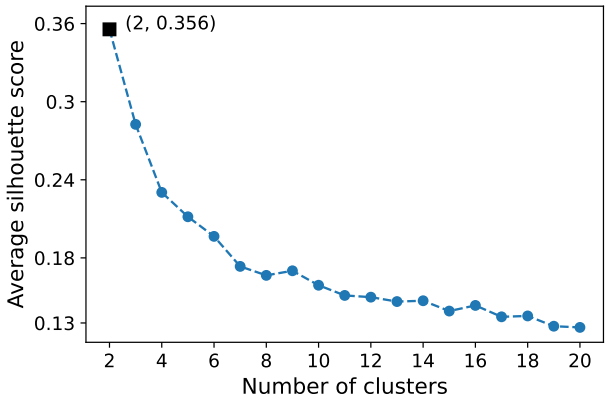

Figure 12. Average silhouette coefficient versus number of clusters. The black square indicates simply the point with the highest silhouette score.

with the whole standardized dataset (0.219). Moreover, for these two cases, the silhouette curve has a peak, respectively, with three and four clusters. Nonetheless, whatever model we consider the avarege silhouette score still remains low, and this might indicate that clusters are close to each other. Putting all this information together, it seems reasonable to narrow down the number of clusters: the "optimal" number could be within the range $[2, 4]$. For this reason and for space-economy, in the next figures only the graphs related to models producing a number of clusters belonging to this range will be shown.

Until now we have considered the total SSE (overall validity function for clusters cohesion), however it can be helpful to compare models with respect to the intracluster SSE (cohesion for each cluster). In figure 13, we can see that the SSE is distributed approximately equivalently within each cluster. Generally speaking, the homogeneity among clusters might indicate that they have similar sizes and densities. The cluster C2 in the second model seems to be an exception since it has an SSE much lower than the other two clusters, meaning that there the objects are more closely related. On the other hand, the most balanced results are obtained with the model with four clusters. The same analysis can be conducted regarding the silhouette score. Looking at figure 14, for each k-means model we can analyze the average intra-cluster and inter-cluster silhouette scores as well as the silhouette coefficient for each object.

The silhouette plot can be seen as an horizontal barplot, where each object is associeted with a horizontal bar of length equivalent to its silhouette coefficient. In each cluster, the objects are sorted in acending order on the y-axis with respect to their silhouette coefficients. Points with a negative silhouette coefficient are badly allocated. This phenomenon occurs when an object is closer to the nearest cluster then to the one to which it belongs. It is clear from the image that the more the number of clusters increases, the more frequently and with greater intensity this problem occurs. Perhaps, it is possible to try to reassign the misallocated points in order to maximize their silhouette coefficient, so that the model also generates an higher overall average score, however, this analysis was not conducted.

For single objects, the model that seems to produce the better outcome is the one with only two clusters.

Continuing with the analysis, the figure 14 suggests that the first and the third model have a more balanced intra-cluster silhouette scores, meaning that for those models clusters lies at roughly the same distance with respect to each other.

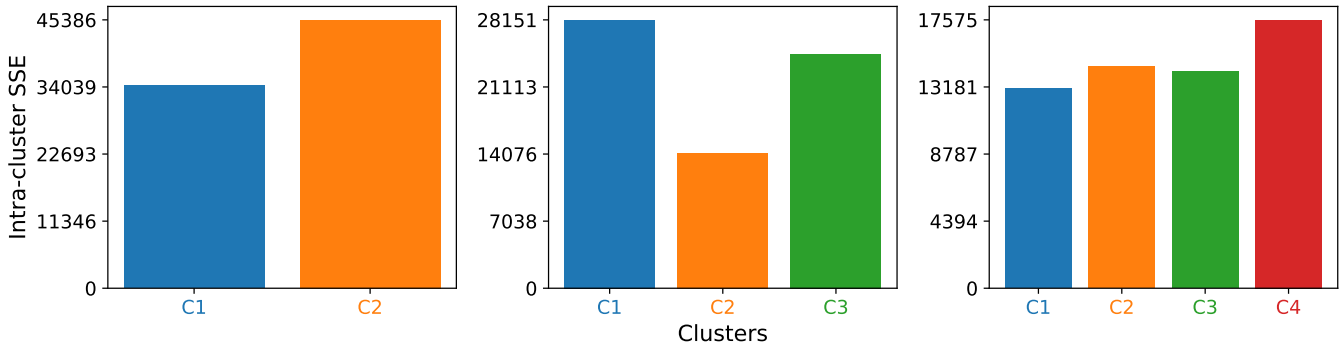

Figure 13. Intra-cluster SSE for each k-means model.

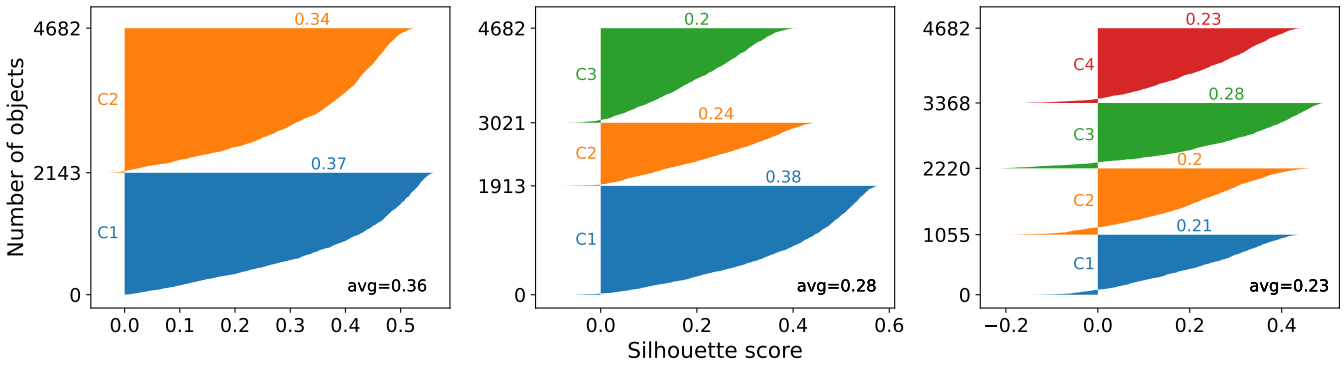

Figure 14. Silhouette analysis. For each model, on the y-axis are represented the individual objects sorted with respect to their silhouette coefficients. Points belonging to the same cluster share the same color. Along with the silhouette coefficients of each object, the intra-clusters and the overall average silhouette scores are also reported.

Furthermore, looking at the length of the base of the pseudo-triangular shape, we can note that, for these two models, clusters have approximately the same cardinality.

Looking at figure 15, for each model we can appreciate how the centroinds distribute in the space with respect to each other. For all the shown models, the graph does not show any issues. Generally speaking, two clusters overlap each other when they have very similar values for all the centroids dimensions; however, due to the curse of dimensionality problem, with nine attributes it is improbable that this might happen here. Nevertheless, in the model with three clusters, the centroids seems better distributed in the space. Instead, in the model with two clusters, for five out of nine dimensions, the two centroids have almost identical values.

2.1.1 K-means: hierarchical initialization

The most critical aspect of the kmeans algorithm is the initialization: different initial centroids can (and usually do) lead to different clusterings. As already mentioned, the k-means++ inizialization is a good way to address this problem. Another strategy that can be used is the following: compute the initial centroids by means of the hierarchical algorithm applied over a random sample of the data, and then use k-means.

Generally speaking, the agglomerative clustering is a deterministic algorithm, however, since it is used on a random sample, this version of k-means is in turn not deterministic. The hierarchical algorithm will be discussed in more detail in section 2.2, for now we just observe the results it produces when used as a subroutine of kmeans.

The random sample was generated with 25% of objects randomly drawn from the Glasgow Norms dataset. The draw was without replacement. For the hierarchical algorithm the following linkage criterion were used: single, complete, group average and Ward. The model was tested for all the positive integers $k \in [2, 20]$, with k number of clusters. Moreover, all four criteria were compared to each other as well as to the k-means++ algorithm.

Since no notable advantages were found in using the alternative initialization approaches, it was decided not to show the results. Indeed, for both silhouette score and especially SSE, for all the initialization methods the curves almost overlap each other in every region of the domain.

2.1.2 K-means and hierarchical clustering

In the previous subsection, we tried to use the hierarchical clustering to initialize the k-means algorithm; here, instead,

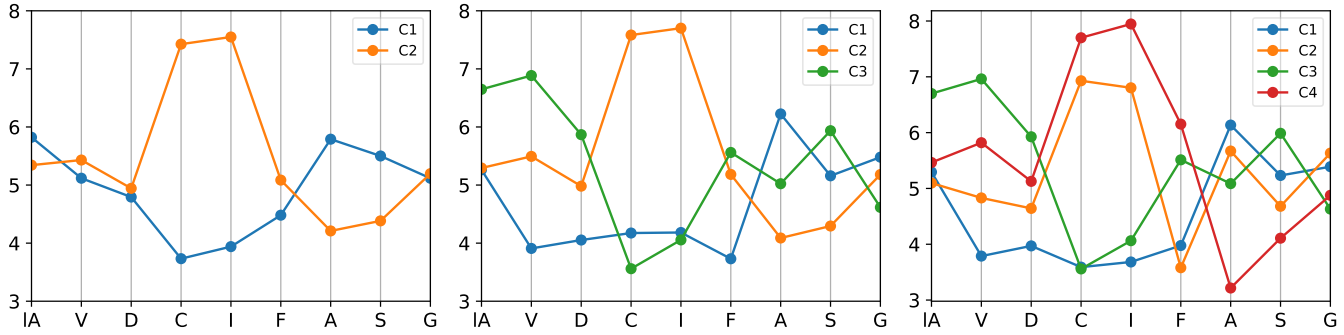

Figure 15. For each k-means model, the graph shows the spatial distribution of centroids. To better fit the figure, only the first character of the attribute's name is reported on the x-axes.

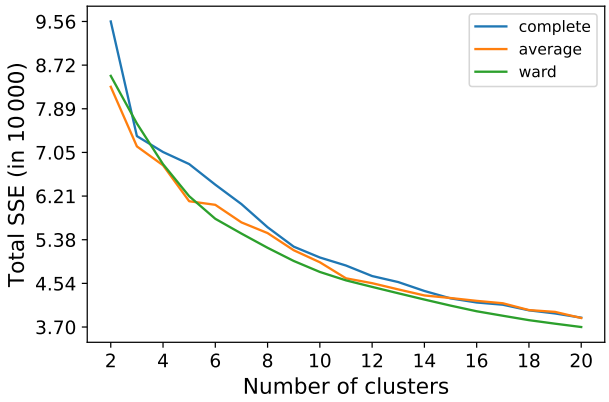

Figure 16. K-means followed by agglomerative hierarchical clustering. Number of clusters versus total SSE.

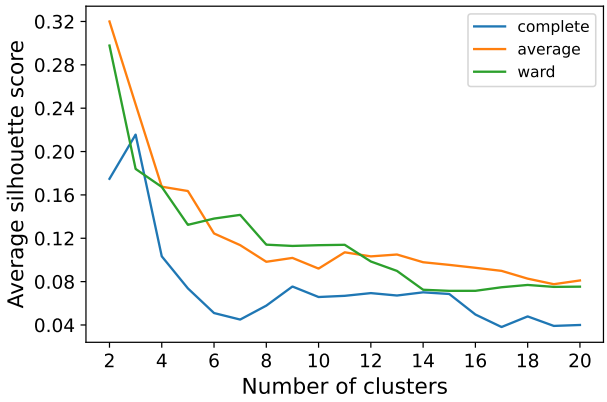

Figure 17. K-means followed by agglomerative hierarchical clustering. Number of clusters versus average silhouette score.

the opposite strategy is discussed. The first step is to run the k-means algorithm so as to produce a large number of clusters. This phase can be seen as a kind of pre-processing step to compress the data volume. Then, the agglomerative hierarchical algorithm is applied to group all the clusters until a specified condition is reached.

The strategy just described was applied to a number of 68 clusters obtained from the k-means algorithm with the following additional hyperparameters: 100 initialization with k-means++, iterations capped to 600 and tollerance equal to 10^{-9}. To allow the hierarchical algorithm to work on an already partially clustered dataset, a personal implementation of the algorithm capable of working with all four linkage methods was used. Figures 16 and 17 summmarize the analysis performed. For a better graphical representation, the three curves have been drawn as if they were continuous, however they are only defined for natural numbers. Note that the single link method was removed from the graphs owing to its poor results.

In general, these linkage methods behave similarly. Regarding the total SSE, figure 16 shows that when the number of clusters is small, the average link method is preferable because is characterized by a lower SSE. Instead, models with more than five clusters benefit from using the Ward method.

We can observe a similar pattern also in figure 17: when the number of clusters is small or high, average link has the highest silhouette score; instead, for an average number of clusters the Ward's method performs better.

However, these results strongly depend on the initial k-means clustering. Due to the stochastic initialization, each time the algorithm is used the relative performances among link methods might change.

Nevertheless, it appears clear from the two figures that for the Glasgow Norms dataset, overall, the described algorithm (k-means followed by hierarchical) does not produce a good clustering. Indeed, it produces clusters of different sizes and, as already seen, all the link methods used produce higher SSEs and lower silhouette scores than the pure k-means algorithm (compare figures 16 and 17 with 11 and 12).

2.2 Hierarchical clustering analysis

For the sake of comparisons, we will use the same dataset with the same attributes analyzed in the k-means clustering as well as the same metric, Euclidean distance, since this is the configuration that produced the best results.

Hierarchical clustering analysis yields a hierarchy of clusters (nested clusters) without making any *a priori* assumption about their numbers. It is a deterministic greedy algorithm, hence it can fall into sub-optimal solutions. There are two main approaches: agglomerative and divisive. The former starts assuming each object as a singleton cluster and, following a bottom-up approach, merges the closest pair of clusters at each iteration. On the other hand, the latter adopt the opposite strategy, using a top-down approach: it starts with only one big cluster containing all the observations and then builds a hierarchy by splitting one cluster at a time, according to specific criteria.

For the first family of techniques, different cluster proximity criteria were tested. While for the second group, a personal implementation of bisecting k-means was used.

2.2.1 Agglomerative hierarchical clustering

The agglomerative hierarchical algorithm requires defining a notion of cluster proximity. Especially four linkage criteria were tested: single link, complete link, group average, Ward's method. The outputs are shown in figuree 18. The single link method was removed due to its poor results: indeed, it can find only one big cluster (the other clusters are all singleton), and the same thing also happens with the cleaned dataset. As we know from the previous section, this might depend on the fact that, in the Glasgow Norms dataset, clusters are close to each other. Generally speaking, the single link defines the distance between two clusters as the distance between their nearest two points. Thus, this distance represents the maximum similarity among those objects. The single link approach behaves well in handling non-elliptical shapes and with well-separated clusters, but it is sensitive to noise and outliers and tends to produce

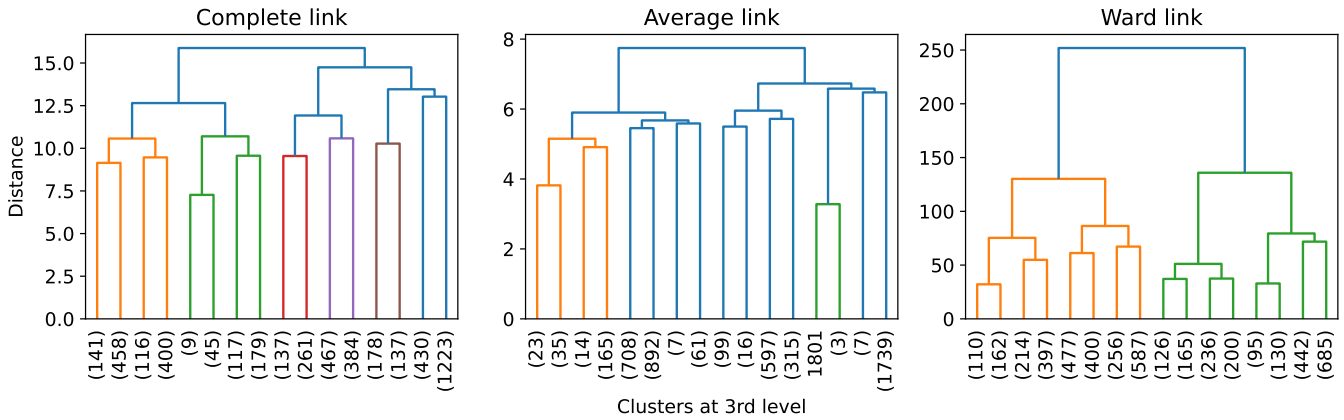

Figure 18. Dendrograms for different linkage criterions. Each vertical line is a cluster. For each of the 16 clusters at the 3rd level, the x-axis shows the cardinality (if the number is within parentheses) or the index (for singleton). Two clusters merge whenever an horizontal line links them. For each horizontal line, the value on the y-axis indicates the value of the proximity function (the distance for complete and average link) at which the two clusters merge.

large clusters.

The three dendrograms in figure 18 were truncated at the 3rd level, where they all produce 16 clusters. We can select the preferred clustering by simply cutting the tree with a horizontal line at a specific height.

Among the three methods shown in the figure, the group average link produces the worse output. This linkage method defines the proximity between two clusters having cardinalities, respectively, n and m, as the average pairwise distances among all the $n \times m$ pairs of points. Despite, it is less sensitive to outliers than the single link criterion, and although it is biased towards globular clusters, for $k > 3$, the algorithm builds many empty or almost empty clusters. For instance, to generate a clustering with five sets of objects, we need to cut the tree in figure 18 to a height of approximately 6, thus producing a cluster with four points and one of seven (respectively, the third last and the second last). On the other hand, if we wanted four clusters, we would have to cut the tree to a slightly higher height, making the second last cluster to have four objects.

The complete link assumes that the proximity between two clusters is determined by their most distant points. Therefore, this distance could also be seen as the diameter of the united cluster. Unlike the single linkage method, the complete link is biased towards tight and compact clusters. Even though it behaves slightly better than the other two criteria analyzed earlier, it remains unbalanced when the number of clusters is greater than 3. For example, looking again at figure 18, if we cut the tree to a height of approximately 12.5—thus generating a clustering with 5 sets of objects—the third and the second last clusters would have cardinalities very small compared to that of the other three clusters, equal, respectively, to 315 (i.e. 178 + 137) and 430. Moreover, if we cut the tree to a height of about 13, we would eventually have 4 clusters, where the second last is still the one with 315 points.

Finally, Ward's method measures the proximity between two clusters in terms of the increase in the SSE that results from merging the two clusters. Like the k-means algorithm, it attempts to minimize the sum of squared distances of objects from their cluster centroids. The Ward method, consistently with the other two methods analyzed, tends to prefer globular clusters and is less susceptible to noise than the single linkage criterion. Moreover, it usually produces more balanced clusters than the other linkage methods, as in the present case. Indeed, looking at its dendrogram, we can appreciate the symmetry of both the branches of the tree. However, in the latter aspect, it is not as good as the k-means algorithm.

All the linkage methods discussed were also evaluated with respect to the cophenetic correlation coefficient (CPCC), a global standard measure of how well a hierarchical clustering fits the data (see table 5). This coefficient measures how well the clustering produced by the algorithm matches actual distances. Besides the dissimilarity matrix, we also need the cophenetic distances among objects to compute this measure. For each pair of points, the cophenetic distance is the value computed by the proximity function the first time they merged in the same cluster. In the dendrogram, this is given by the height of the horizontal segment connecting their respective clusters. Therefore, all objects that merge in the same iteration step will mutually share the same cophenetic distance. Then the cophenetic coefficient is computed using the Pearson correlation between the distance and the cophenetic distance matrices. It can assume all real values in the range $[-1, 1]$: the closer to 1 is the coefficient,

Table 5. Cophenetic correlation coefficient for the 4 agglomerative hierarchical clustering techniques.

Linkage criterion	CPCC
Single link	0.143
Complete link	0.515
Group average	0.575
Ward's method	0.532

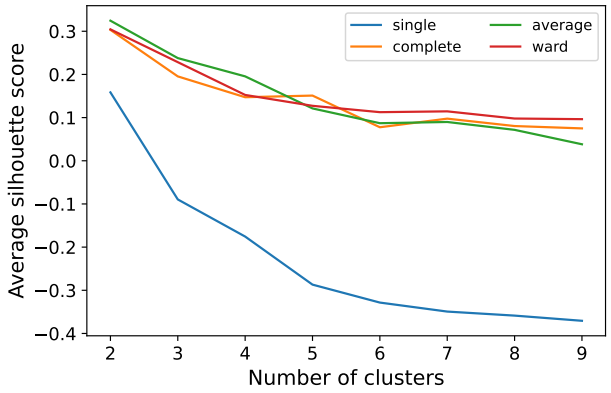

Figure 19. Average silhouette coefficient versus number of clusters. Single, complete, group average and Ward's linkage criteria compared.

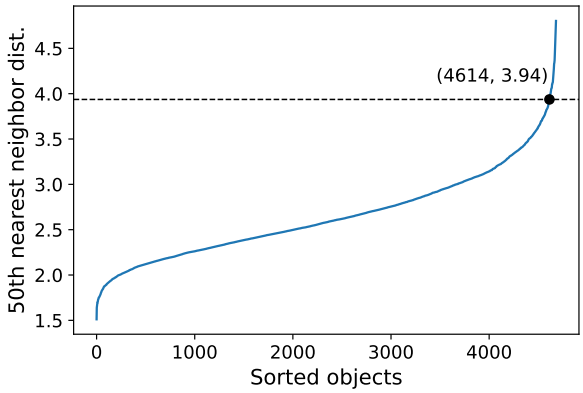

Figure 20. The figure shows the distance between each point and its 50th nearest neighbor. On the x-axis, objects are sorted by distance to their 50th nearest neighbor. The black dot on the curve represents the knee point found by the KneeLocator.

the higher is the clustering quality.

Table 5 corroborates what previously said about the single linkage: it is undoubtedly the method with the worst results. The other three criteria approximately share the same medium cophenetic coefficient. Theoretically, this means that they all produce decent clusterings. However, it is slightly odd that the group average obtained the highest score since it generates many empty clusters for $k > 3$.

Silhouette analysis confirms what the cophenetic coefficients highlighted. Looking at figure 19, it appears that single, complete, and average links all have a similar level of effectiveness, although the k-means algorithm still has a higher score. Finally, the single link criterion is so bad that it even produces a negative average silhouette score.

2.2.2 Bisecting k-means

As already mentioned in the introduction of this section, the bisecting k-means is a divisive hierarchical algorithm. The criterion used to split the clusters is based on the intra-cluster SSEs and cardinalities: at each step, the algorithm selects and splits the cluster with the highest SSE; ties are solved by picking the one containing more objects.

This algorithm does not behave badly at all with the Glasgow Norms dataset. Indeed, the total SSE is very close to the one produced by the k-means initialized with k-means++. On average, for each number of clusters, its SSE is larger by about 3000, whereas the overall silhouette score is nearly the same.

For this reason, it was also tried to initialize the k-means algorithm with the bisecting k-means. Nevertheless, like the other initialization approaches, it does not improve the pure k-means.

2.3 DBSCAN

The last clustering algorithm to be discussed is DBSCAN, the deterministic density-based algorithm described in section 1.3. Generally speaking, it is able to determine the clusters by using the contrast between high and low-density areas, therefore it is relatively resistant to noise and can handle clusters of arbitrary shapes and sizes. DBSCAN does not work well with high-dimensional data and when the actual clusters have widely varying densities.

For instance, consider figure 20. It shows a common heuristic to decide the radius value once the number of neighbors is chosen ($minPts$). The curve is determined by computing, for each point, the distance to the kth (here $k = 50$) nearest object. The best ε candidates are those points that lie on the curve where the "knee is bent". For example, if we consider the black point suggested by the *kneelocator* function ($\varepsilon = 3.96, minPts = 50$), we *a priori* know that there are exactly 4614 core points, whereas the other $4682 - 4614 = 68$ objects are either border or noise points. Unfortunately, these hyperparameters put all the objects in one unique cluster. Moreover, if we try to set $\varepsilon = 3$, DBSCAN produces one cluster with cardinality 4644 and marks 38 objects as noise points. Changing the $minPts$ parameter doesn't give much better results.

Overall, more than 7920 $(\varepsilon, minPts)$ combinations were tested, however none of them produced acceptable results. Indeed, either it marks all or most of the objects as noise points or it only builds one big cluster. To be specific, for the radius hyperparameter all the values in range $[0.1, 10, \text{step} = 0.1]$ were considered, while for minPts all values in range $[5, 400, \text{step} = 5]$ were taken into account.

In conclusion, we can say that DBSCAN, together with the hierarchical agglomerative algorithm with single link, is the worst clustering algorithm for the Glasgow Norms dataset.

2.4 Clustering analysis: final considerations

All the analysis carried out seems to suggest that the best clustering model for the Glasgow Norms dataset is the k-means initialized either with k-means++ or with one of the agglomer-

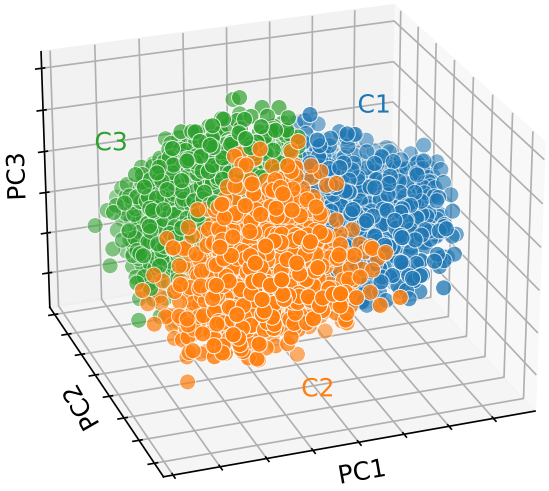

Figure 21. 3D Scatter plot for the k-means model with $k = 3$. The colors of the three clusters are consistent with those used in all other figures.

ative hierarchical algorithms seen in the previous sections. Table 6 summarizes some helpful clustering evaluation measures by assuming the scenario with three clusters. Agglomerative hierarchical algorithms are not prototype-based, since they do not have the concept of "centroids", thus these models are not usually evaluated with respect to the SSE. However, since the three linkage methods reported in table 6 are usually biased toward globular clusters, and since this is precisely the case for the analyzed dataset, the *within cluster sum of squares* were computed—*a posteriori*—for these models as well.

In addition to the measures already discussed, the *between cluster sum of squares* (SSB) is also present. It is a measure of separation between clusters (Tan et al. 2019, p. 359) and is computed as:

$$\text{SSB} = \sum_{i=1}^{k} |C_i| \, dist(c_i, \mu)^2 \tag{4}$$

where $|C_i|$ is the cardinality of cluster i, c_i is its centroid, and μ is the "centroid" of all the data in the dataset. The higher the total SSB of a clustering, the farther apart the clusters are. Once again, k-means triumphs over the other algorithms.

Concerning the silhouette score, overall it is rather small. As already said, this may mean that clusters are close to each other or poor quality clustering, but it might also indicate that there are no actual clusters in the data. Indeed, clustering algorithms, like humans, suffer from apophenia: they might find false patterns in random information.

Table 6. Clustering algorithms comparisons: summary evaluation measures for $k = 3$.

Clustering algorithm	SSE	Silhouette score	SSB
K-means	66 879	0.283	49 002
Bisecting k-means	67 855	0.280	48 022
Complete link	74 286	0.196	41 595
Group average	71 231	0.238	44 650
Ward's method	74 930	0.228	40 951

In order to reject (although not definitively) the latter hypothesis, the dataset was tested with the Hopkins statistic (ibid., p. 371). It substantially measures the clustering tendency by comparing n randomly generated data with a sample of the dataset consisting of n objects. The mathematical formula is the following:

$$H = \frac{\sum_{i=1}^{n} w_i}{\sum_{j=1}^{n} u_j + \sum_{i=1}^{n} w_i} \tag{5}$$

where w_i is the distance from the generic point i of the sample to the nearest neighbor in the original data set; likewise, u_j is the distance from the generic random point j to the nearest neighbor in the original data set. This statistic may range between 0 and 1. Values near 0.5 indicate the absence of actual clusters.

For the Glasgow Norms dataset, the Hopkins statistic is moderately far from the critical value, indeed $H = 0.259$. This "confirms" the existence of actual clusters in the data.

Therefore, the ultimate question is: how many clusters should we consider? In section 2.1, talking about the k-means algorithm, we supposed that the best clusterings were the ones with two, three, or four clusters. Afterward, in section 2.2, we discovered that all the linkage criteria behave similarly when the number of clusters is smaller than or equal to three. Instead, for $k \geq 4$, their performances worsen, especially those of the group average method. Given all of these considerations, it was deemed reasonable to set the number of clusters at three and accept the clustering produced by k-means.

Figure 21 depicts the three clusters in a simplified three-dimensional space. The 3D scatter plot was generated using the dimensionality reduced dataset computed during the principal component analysis performed in section 1.3. The observations we had made in that circumstance are still valid: the three principal components account only for 69% of the total variance, thus the representation should be taken with a pinch of salt. That being said, the figure confirms our hypotesis: clusters are compact, globular, and very close to each other.

Figure 23 shows the clusters distributions of the nine quantitative PSY variables used as inputs to produce the clustering. It was constructed in the following way: after grouped data by labels, an aggregation function was applied to each cluster to compute the third quartile of each variable. Hence, within each cluster, 75% of all the objects have a magnitude smaller than or at most equal to the one shown in the heatmap for that attribute. The k-means algorithm partitioned the data in the following way:

- in cluster C1 it put the words rated as concreate, easy to imagine, and therefore familiar.
- Words with high levels of arousal, valence, dominance, familiarity, and size were mainly allocated in cluster C2.

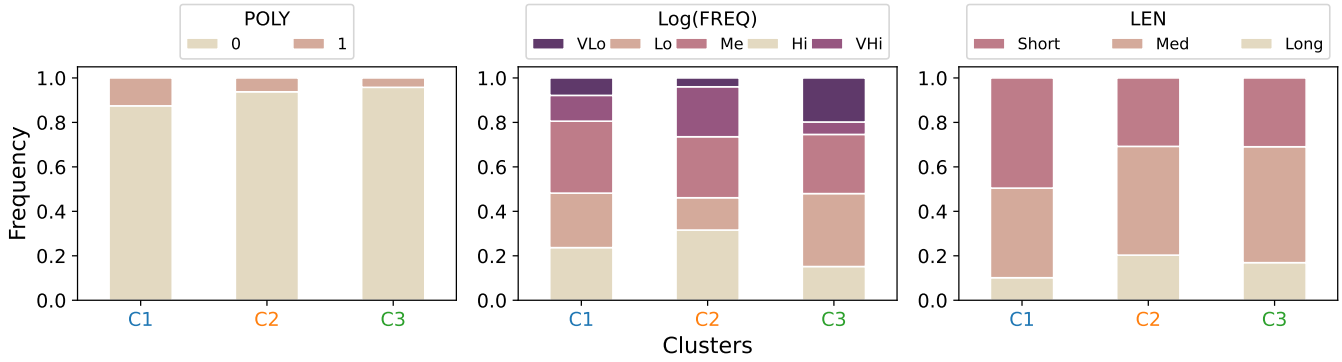

Figure 22. Distribution of categorical variables within clusters for the k-means model and $k = 3$. The first bar plot refers to the POLY attribute, the second to the Log(FREQ) attribute, and the third to the LEN attribute.

The majority of them are also feminine, abstract, and difficult to imagine.

- Lastly, C3 is characterized by words that are presumably considered more complex because they were learned at a more adult age. Indeed, these words also have a low level of valence, dominance, concreteness, and imageability.

Finally, in figure 22, we can observe how the categorical variables are distributed within the clusters. Regarding the POLY attribute, C2 and C3 have very few polysemic words (respectively, $\simeq 4.2\%$ and $\simeq 6.2\%$), whereas C1 has a more balanced makeup ($\simeq 12.5\%$). In general, since POLY is an asymmetrical attribute (as mentioned in section 1.1, only 8.81% of the words in the dataset have a value equal to 1), each cluster has an unbalanced composition within itself.

Concerning the FREQ attribute, cluster C1 has an internal composition consistent with the FREQ distribution, that is: Med > Low = High > Very Low = Very High. Instead, C2 and C3 are specular: the former, for more than 50%, is

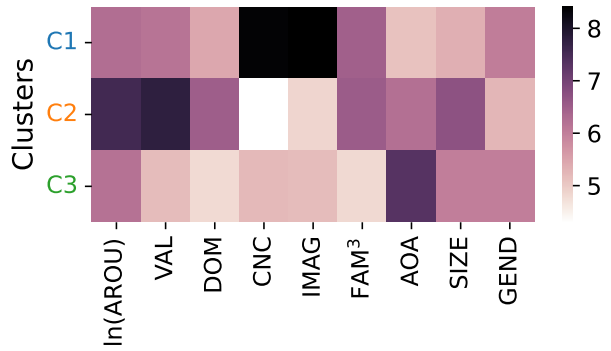

Figure 23. Distribution of quantitative variables within clusters for the k-means model with $k = 3$. The heatmap was obtained using third quartiles.

composed of words with a high or very high frequency, while for C3 the opposite is true.

Eventually, we consider the LEN discretized attribute. The variable overall has a skewed distribution (short = 38.5%, med = 46.5%, long = 15%), however most of the long words was clustered within C2 and C3, whereas C1 is highly biased towards short-medium words.

3. Classification

The general idea behind classification is to find a set of attributes that can be used to predict the value of a qualitative target variable for each desired record defined over that same set of attributes. Compared to the clustering analysis, the main difference is thus the pre-existence of this set of labels.

The first goal is to identify a suitable dependent variable. Regarding this project, the choice was quite forced: POLY seems the most suitable for this task since it is a binary variable, whereas the other attributes are continuous. Alternatively, we could have used the discretized FREQ variable, however, the problem would have become multiclass.

Another relevant aspect concers the selection of the best subset of attributes to use as independent variables. Theoretically, we would like to choose the variables with the highest predictive power. In practice, the choice is strictly dependent on the classification method utilized. The techniques explored here are substantially three: decision tree, k-nearest neighbors, and a dummy classifier.

3.1 Decision tree

Decision trees are very effective classifiers, with many advantages. Some of them pertains precisely to a point already touched upon: the choice of the attributes set. Indeed, they can easily handle redundant or irrelevant attributes and are robust to noise. Notwithstanding, different configurations were tested. Firstly, due to the Scikit-learn implementation, we were forced to return to using the (min-max normalized) dataset at a state before the discretization of the variables LEN and FREQ.

Like in the clustering analysis, we tried to use only the nine psycholinguistic variables. Then we also tried to use a smaller subset, composed of only the Gini important features as determined by the algorithm (that is, the attributes that generated the highest reduction of Gini impurity measure). Moreover, we

tried to remove the features that might be considered redundant like IMAG, FAM e DOM since high correlated, respectively, with CNC, AOA, VAL and also because they led to smaller improvements in terms of impurity measures. However, the best results were obtained using all eleven variables of the dataset. This behavior was also noted with the other classifiers discussed later. Hence, unless otherwise specified, we will refer to this specific dataset from here on out.

As already noted in section 1 our target variable is highly skewed and this is a crucial problem in classification analysis. Indeed, many classifiers work well only when the training set has a balanced representation of both classes. If they are trained over imbalanced datasets they may not be able to detect in an effective way objects belonging to the minority class.

Speaking specifically about the Glasgow Norms dataset, we tried to run a five-times stratified 10-fold cross-validation algorithm over the training set without using any sampling technique before. The best strategy identified by the algorithm was to make always negative predictions, that is the majority class (actually, nine out of 1405 were positive predictions).

To obtain more satisfactory results, two sampling techniques were tested: undersampling and oversampling (specifically the SMOTE technique). Moreover, in order to make more meaningful comparisons between the different models, the dataset was initially partitioned into two parts (in a stratified way): 70% of the data reserved to the training set and the remaining 30% to the test set. All the models were trained and tuned using the stratified cross-validation technique over the training set (5-fold in the case of undersampling, 10-fold otherwise).

Particularly, this strategy consists in split the training set into k parts: at each iteration $k-1$ of them are used to train the model and the remaining one is used as validation set to tune the hyper-parameters and computing performance measures. In the end, each fold is used exactly $k-1$ times to train each model and exactly one time to validate the model.

After the best model was identified, it was also refitted over the entire training set. The resulting classifier was tested both on the training and the test set. Finally, all the classifiers discussed in this project were compared over the same test set having a cardinality of 1405 records (this same test set was also used for the classifier discussed above).

3.1.1 Undersampling

The frequency of the majority class was reduced to match the frequency of the minority class. Specifically, the training set defined above contains 3012 objects belonging to the negative class and only 265 to the positive class. In order to balance the two classes, 265 instances were randomly drawn from the set containing only negative elements. Then we concatenated the two sets and shuffled the output so to have a better worst case. This strategy introduces two more randomness factors: the first when negative elements are drown within their class; the second when the final training set is shuffled. In theory, the first point is the more potentially problematic because the

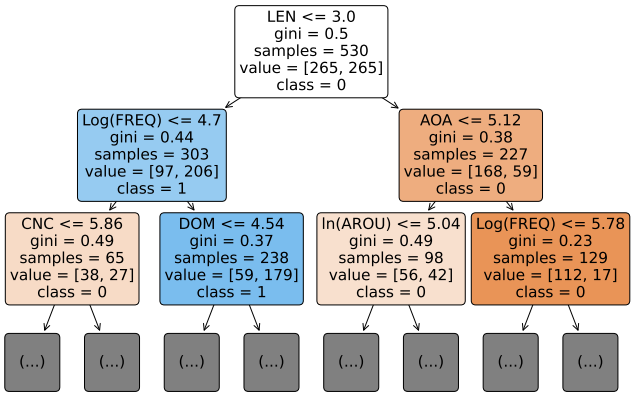

Figure 24. Decision Tree Classifier. Blue leaves contain mostly records belonging to the positive class. The orange ones, on the other hand, are mostly composed of items from the negative class.

sampled negative instances may have a higher variance than the previous dataset. We checked this aspect, but the change in variance between the two datasets are minimal. In the end, the decision tree classifier was trained with a stratified 5-fold cross-validation over this dataset with cardinality 530. The score maximized during the greedy search was the F_1 since it is a suitable measure of models tested with imbalance datasets. Indeed, by using this measure to select the best model, we are implicitly stating that we want the classifier that maximizes the worst measure between "precision" and "recall". This choice was also applied for the models described later.

The hyper-parameters were tuned considering the following domains:

- `max depth:[2 to 8]`
- `min samples split:[40 to 80, step = 10]`
- `min samples leaf:[20 to 60, step = 10]`

The resulting tree has `8 leaves`, a `depth=3`, `min samples leaf=20`, and `min samples split=40`. These parameters act as pre-pruning conditions and are particularly helpful to control the model overfitting over the training set since they do not allow the tree to grow uncontrollably.

The Gini coefficient was used to compute the impurity. The first three levels of the decision tree are shown in figure 24.

At the root level, for all the attributes, the Gini coefficient is maximized since records are equally distributed among all classes. The first (best) splitting attribute to be selected is LEN, which means that the test condition of this attribute produces, overall, the highest gain in purity, that is:

$$G_{sx} = 1 - \left((97/303)^2 + (206/303)^2\right)$$
$$G_{dx} = 1 - \left((168/227)^2 + (59/227)^2\right)$$
$$\text{Gain} = 0.5 - (303/530)G_{sx} - (227/530)G_{dx} = 0.0864$$

where G_{sx} is the Gini index for the left child. It is equal to $\simeq 0.44$ as indicated in the figure. Equivalently, $G_{dx} = 0.38$ represents the Gini index of the right child. After the split, the 291 records concerning words smaller than or equal to 3 characters go to the left child, whereas the 239 instances regarding words longer than 3 go to the right child. Generally speaking, these attribute tests partition the space with border lines parallel to the axes and are known as *decision boundaries*.

The order in which nodes are visited is not a critical aspect of this implementation since the constraints specified above only limit the expansion of individual nodes.

Now let's focus on the way the records descend through the tree. The blue color indicates the positive class while the orange the negative one. After the first split at the root level, the tree can ideally be divided into two halves: the left prevalently containing objects belonging to the first class (206/303); the second characterized by mostly negative instances (168/227). The resulting decision tree classifies all the words that simultaneously have a *length* ≤ 3, a *frequency* ≤ 4.7, and a *dominance* ≤ 4.54 as positive instances. Conversely, whenever an object does not presents all these characteristics at the same time it is classified in the negative class. This happens due to the default threshold set to 50%. In particular, whenever we want to classify a new record, this descends through the tree until it reaches a leaf. Then the classifier assigns it the majority label of the instances in such a leaf.

Furthermore, by looking at these 3 levels, we can see that the Log(FREQ) attribute appears twice. This pattern also repeats for the other level not shown in the image. Indeed, figure 25 shows the most important features, i.e. the attributes that contribute the most to reducing impurity. Surprisingly, the variable that contributes most to the construction of the tree and thus to discern between records, is LEN. Actually, this is not completely unexpected. Indeed, feature importances are strongly related to the Pearson correlations between the POLY attribute and the other variables (see table 7). Moreover, for each decision tree classifier tested, the LEN attribute is always at the top of the list of feature importances. This might depend on the fact that it is always chosen as a splitting attribute at the root level. Indeed, the first split makes a binary partition of the whole dataset, consequently, LEN can discern between a very

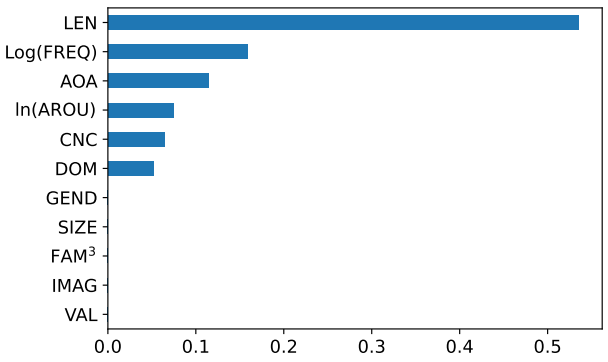

Figure 25. Feature importances.

Table 7. POLY correlations. Values are sorted with rispect to their modulus.

LEN	-0.231	IMAG	0.104
Log(FREQ)	0.190	ln(AROU)	-0.084
AOA	-0.179	DOM	0.052
SIZE	-0.133	GEND	0.025
CNC	0.113	VAL	0.021
FAM^3	0.104		

high number of objects (in absolute terms).

Let's now briefly discuss about the performances of this model over the test set. The confusion matrix in figure 26 can help us in this regard. The first thing that jumps out is the high number of false positives. Indeed, the model in order to try to predict more positive cases, makes a lot of mistakes. This is also the reason for the very poor performaces obtained with respect to the precision measure (16%). However, none of the classifiers tested for this project were able to perform better in terms of recall of the positive class (81%) than the one considered here.

It is possible modifying the hyper-parameters to obtain a model more accurate (here, accuracy = 64%), by ensuring, for instance, that the classifier predicts the positive class more rarely. Trivially, this can be obtained by a dummy model that always predicts false. Considering that the test set used here is composed of 1291 negative instances and 114 positive instances, such a model would obtain an accuracy of $\simeq 92\%$. However, it should be noted that the accuracy measure is not suitable for handling datasets with imbalanced class distributions as it tends to favor classifiers that correctly classify the majority class. For this reason was used the *balanced accuracy score*, consisting in the unweighted average between the two recall measures (one for each class), that for the decion tree discussed here was $\simeq 72\%$.

If we analyze, instead, the performance over the training set the results are of course better. This happens for two reasons: the first is that the model is biased toward that dataset since it

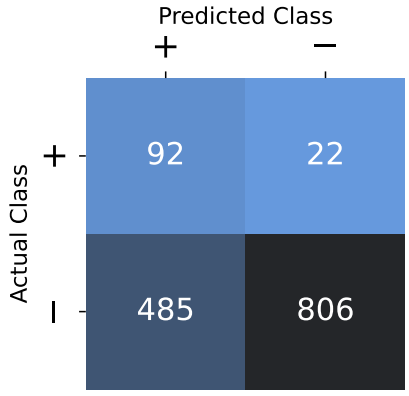

Figure 26. Confusion matrix for the decision tree trained over the undersampled training set.

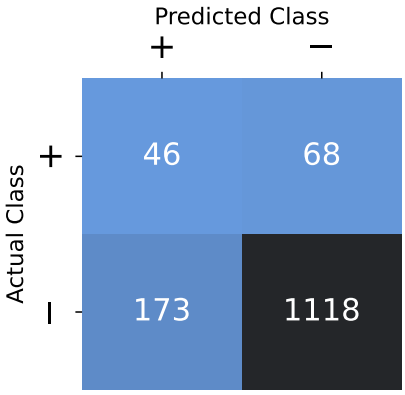

Figure 27. Confusion matrix obtained applying the decision tree computed from the oversampling technique to the test set.

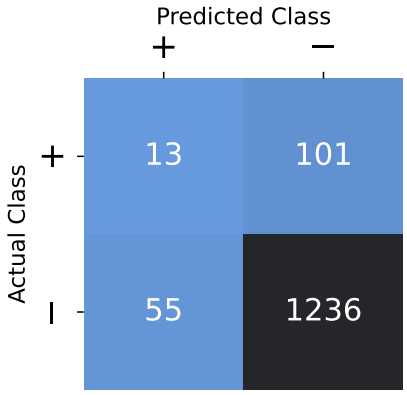

Figure 28. Confusion matrix for the KNN classifier applied over the test set.

was trained over it; secondly, unlike the test set, the training set is balanced because of the undersampling. Despite that, the performance measures are not too high, and this probably depends on the fact that the model is not adequately complex (small underfitting).

3.1.2 SMOTE

Another approach that can be used is oversampling, and in particular the Synthetic Minority Oversampling Technique (SMOTE). Instead of simply duplicating positive records, we can identify the k-closest positive neighbors of each positive instance, and then generate new positive instances by choosing a neighbor at random and computing a linear combination between these two vectors. For our training set, for example, this procedure must be repeated $3012/265 \simeq 11$ times for each positive record so that the two classes become balanced. At the end of this procedure, we end up with a training set consisting of 6024 elements.

Also on this dataset, to select the "best" model, we applied a 10-fold stratified cross-validation on the training set. The sets of hyperparameters that have been taken into account to perform the greedy search were the following:

- `max depth:[2 to 10]`
- `min samples split:[ 60 to 100, step=10]`
- `min samples leaf:[30 to 80, step=10]`

Since the high cardinality of this dataset, we avoided using the parameter `None` for the max depth, so as not to generate a too complex classifier. The best parameters discovered by the algorithm were: `max depth=10`, `min samples leaf=30`, `min samples split=70`. This produced a binary tree composed by 74 leaves.

Although the decision tree built using the undersampling technique is much less complex than the one just computed here, they share roughly the same characteristics. The most crucial features in terms of impurity reduction are still the same, with LAN and Log(FREQ) dominating over all the other variables. For this reason and to save space, this time we did not report the decision tree figure and the feature importances barplot. Instead, the confusion matrix generated by applying the classifier over the test set is shown in figure 27.

This decision tree, unlike the previous one, tends to make more negative predictions, and since the test set is biased toward the negative class this generates a lower misclassification error with respect to the undersampling technique. For this same reason, it tends to produce fewer true positives and more false negatives. This means a smaller recall, that is the model has less probability to guess the class of a record when this is positive. In general, as for the previous model, all the measures associated with the imbalanced class are very poor whereas the ones relative to the major class are quite high.

3.2 K-nearest neighbor

K-nearest neighbor (KNN) is a “model-free” classifier in the sense that it does not build a global model, like decision tree, but uses the training examples to make predictions for a test instance, thus acting locally. The idea behind this algorithm is quite simple: each time it must predict the label of an instance, its k closest elements are scanned and then the classifier predicts the class according to a majority or weighted majority voting schema.

The most important hyper-parameter to tune is the number of the k-nearest objects to check at each prediction. In order to find the best value of k, we performed the same cross-validation analysis made for the decision tree. The best k parameter selected during the greedy search was $k = 1$, However, since such low values can generate overfitting, the search was repeated by setting a higher range of candidates values, i.e. $[3, 10]$. This led to the final value of $k = 4$.

Not surprisingly, as shown in figure 28, the KNN classifier does not have better performances with respect to the positive class. Indeed, it tends to predict almost always the negative one. The results in figure 28 were obtained by training the model on the whole training set. The decision tree trained over the same

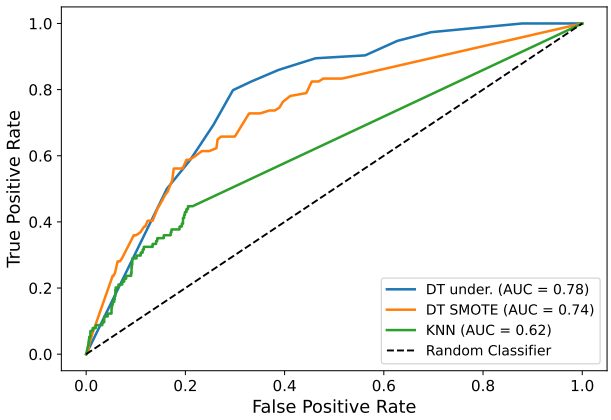

Figure 29. ROC curves for different classifiers.

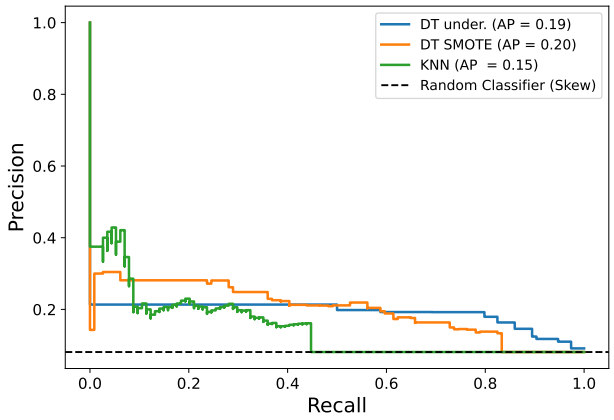

Figure 30. PR curves for different classifiers.

dataset showed very similar scores as we already discussed at the beginning of this section. In addition, our efforts to use the two already seen sampling techniques with the KNN classifier did not yield better performances; for these reasons, we do not report the results.

3.3 Classifier Comparisons

In this section, we will compare the three analyzed classifiers presented in the previous pages. For this purpose, we will make use of two widely used tools: the ROC curve and the Precision-Recall curve (PR).

The three classification models tested before, in addition to predicting the class to which a certain record belongs, can also predict the probability with which that same record is classified with that specific label.

By default, a classifier makes a positive prediction whenever the percentage associated with the record evaluated is greater than 50%. However, it is possible to change this behavior by manually modifying this specific threshold. The ROC curves in figure 29 show exactly what happens in this circumstance. On the y-axis, we have the True Positive Rate (TPR, i.e. recall), while on the x-axis we have the False Positive Rate (FPR) defined as FP/(FP+TN), that is the probability with which the model makes a positive prediction whenever the actual class is negative.

The point $(0, 0)$ indicates what happens to TPR and FPR when the threshold is maximum, i.e. when the classifiers always make a negative prediction. On the other hand, at the point $(1, 1)$ the threshold is set to the minimum. In this circumstance, the classifiers always predict the positive class. Therefore, each ROC curve shows a clear trade-off between TPR and FPR: if we want our model to make more positive predictions when the actual class is positive then we must accept an increase of false positives (and *vice versa*). Generally speaking, models placed near the upper left border of the figure are considered the best ones because they are subject to a lower trade-off.

As it is evident from the image, and as already observed previously, among the three evaluated classifiers, the two decision trees have similar behaviors, especially for high levels of the threshold. This is also supported by the similar Area Under the Curve score (AUC). Overall they always seem to perform better than the KNN classifier. However, it should be noted that for mid-small values of the threshold, the decision tree trained over the undersampled dataset always outperforms the decision tree trained over the oversampled one.

We may decide to manually adjust the threshold depending on the problem under investigation by doing a cost-benefit analysis. For instance, if we were more interested in predicting the positive class than the negative one we should use the former classifier with a lower value of the threshold since, with the same FPR, the TPR would be greater.

On the other hand, the Precision-Recall curve (figure 30) is a particularly suitable tool in the presence of class imbalance. The principle behind the PR curve is the same as the ROC curve, however, now the trade-off is between precision, $p = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$, and recall (TPR).

The dashed line in figure 30 represents the skew of the POLY variable. Unlike the ROC curve, the ideal value of the PR curve is reached in the upper right corner, where precision and recall are both maximized, This occurs for low threshold values.

If we want to have an higher probability to predict positive instances we should increase the frequency with which the classifier predicts positive, thus sacrificing the precision due to more false positives. On the other hand, to make more accurate positive predictions, we should reduce the frequency with which the classifier predicts positive, thus worsening the recall.

In the case of Glasgow Norms, there seem to be no particular reasons to change the standard behavior of the classifier.

4. Pattern Mining

In order to perform an association rules analysis, we first need to discretize all non-categorical variables. In the previous sections, we had already discretized the variables LEN and

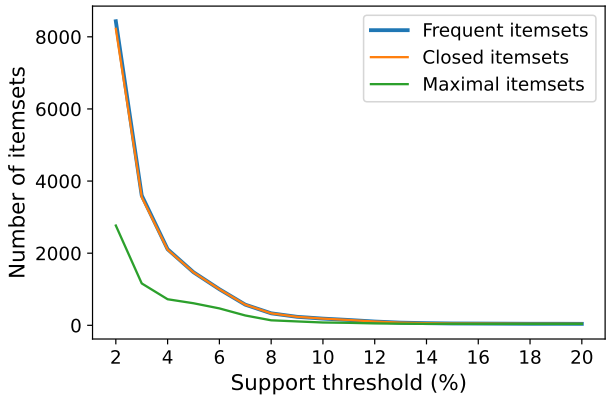

Figure 31. Frequent, closed frequent, and maximal frequent itemsets compared when support threshold is changed.

Log(FREQ) with the k-means clustering technique. Specifically, the domain of the LEN variable became of cardinality three, whereas that of Log(FREQ) flattened out to a total of five categorical values. Instead, for the remaining variables, a quantile-based discretization was used. With this technique we partitioned the domain of each variable into four equal-sized buckets, each pertaining to one quartile. All these operations turned the Glasgow Norms into a transactional dataset.

The objective of the pattern mining analysis is to find significant association rules. Potentially discovered rules do not express a cause-effect relationship between the implicating and the implicated itemsets, instead, they suggest a certain probability of co-occurrence between the elements of the two itemsets.

4.1 Frequent itemsets

The first task performed was to apply the *apriori* algorithm to the dataset to discover frequent itemsets. Several different configurations were tested. The largest one was obtained using a support threshold of 2%. In addition, only itemsets with cardinality greater than or equal to two were considered. These parameters generated a large set of frequent itemsets consisting of 8432 elements. However, the number of frequent itemsets decreases sharply when starting to increase the threshold. Indeed, just setting the minimum support value to 10% produces approximately a 98% reduction in the number of frequent itemsets, lowering its cardinality down to 186 elements. With a threshold equal to 20% the number decreases to 41 and with a `minsup` of 30% only 2 itemsets survive.

This phenomenon can be better explained by looking at figure 31. The three depicted functions show how frequent itemsets, closed frequent itemsets, and maximal frequent itemsets respond on varying of the support threshold.

It seems interesting to observe that the closed and the frequent itemsets curves overlap. It means that almost all the itemsets have a unique support count concerning their immediate supersets and, also, that they are all considered frequent. Instead, regarding the maximal frequent itemsets, the spread between this and the other two curves is meaningful at first, where roughly 50% of the frequent itemsets are also maximal. However, when the threshold exceeds 13%, all the remaining frequent itemsets become both closed and maximal. These behaviors are perfectly normal because the set of closed frequent itemsets is a subset of the frequent itemsets. On the other hand, the set of maximal frequent itemsets is in turn a subset of the frequent closed one. Under a theoretical point, these two sets are useful because they allow computing all the other itemsets thus significantly reducing the computational costs of the pattern mining algorithm.

By setting the threshold at 25% causes only 3 frequent itemsets to remain. In particular, they all are 2-itemsets and are constructed by involving only three different attributes and four categorical values. With a support of 27.08%, the third most frequent itemset is: {`med_Log(FREQ)`, `No_POLY`}. These categorical values are very frequent within the dataset: `med_Log(FREQ)` represents the mode of its distribution, while `No_POLY` is the majority class of the POLY attribute. This suggests us that many words in the dataset have a medium frequency and are non-polysemic. Not an interesting information. The second and the first most frequent itemsets, with a support of of 32% and 45% are, respectively, {`Short_LEN`, `No_POLY`} and {`Med_LEN`, `No_POLY`}. Again, not an interesting information, since we already knew that the LEN variable is skewed toward words having a short/medium length.

4.2 Association Rules

In order to extract the association rules (AR) from the itemsets, a greedy search was performed, counting the number of rules generated for each combination imposed by the support and confidence thresholds. In particolar, all supports in the range $[2, 22]$ as well as all confidence thresholds in range $[60, 80]$ was used in order to see how many rules were possible to extract from each combination. In total, 441 combinations were generated. Results are summarized in the heatmap in figure 32.

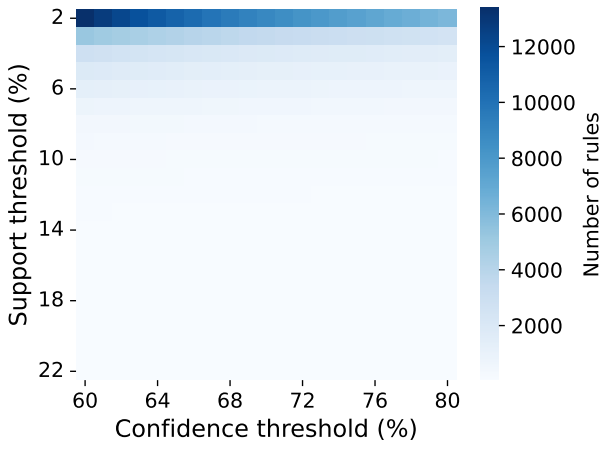

Figure 32. Association rules extraction.

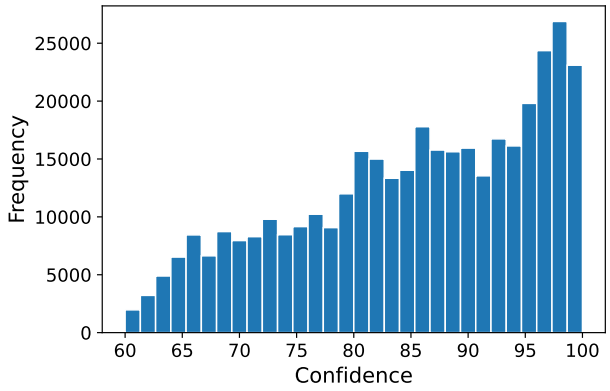

Figure 33. Histogram of rules' confidence.

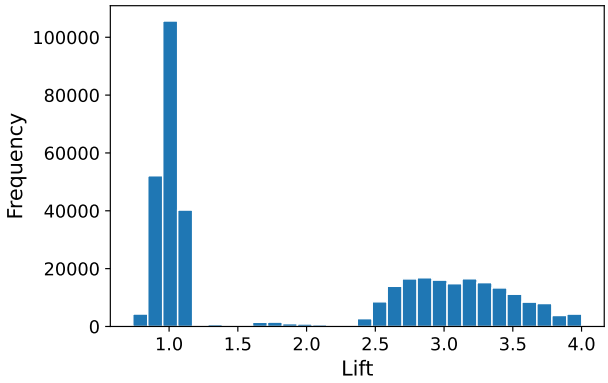

Figure 34. Histogram of rules' lift.

As expected, following the analysis carried out in the previous subsection, for low values of the support threshold, a greater number of itemsets and thus of rules is obtained. For the confidence threshold the same principle applies: if we want to extract stronger rules we must necessarily increase this threshold and thus give up most of the rules.

Let's try to analyze some rules. For example, setting a minimum number of two items in the antecedent itemset, the rules with the highest values of support and confidence are the following three:

- $\{\texttt{High\_CNC}, \texttt{High\_IMAG}\} \longrightarrow \{\texttt{No\_POLY}\}$;
- $\{\texttt{High\_CNC}, \texttt{No\_POLY}\} \longrightarrow \{\texttt{High\_IMAG}\}$;
- $\{\texttt{High\_IMAG}, \texttt{No\_POLY}\} \longrightarrow \{\texttt{High\_CNC}\}$.

All three rules have a support of 19.2%. The first one has a confidence of 89.4%, the second one 86.1% and the third one 85.7%. Again, we found something quite expected. As already analyzed in subsection 1.4, CNC and IMAG variables are those with the highest positive linear correlation and it is not surprising to find them in the same association rules. This is also highlighted by the lift measure of the last two association rules: both are equal to 3.4 and this suggests us that they are positively related. The same for the non-polysemic words, since as already recalled, it is the major class of the heavily skewed POLY variable.

Finally adopting the same approach followed for depicts figure 32, as well as the same search intervals, we generated the histogram of both confidence and Lift measure. From figure 33 we can see that despite the 379 152 rules generated, the distribution is still skewed towards those with higher confidence. This tells us that consequent itemsets often appear in transactions that contain the antecedents of the rule. However, as we have already seen, many rules are trivial and do not allow us to extract any insights from the data.

The lift measure intead has a bimodal distribution (see figure 34). The majority of the association rules generated in this interval established by different confidence and support thresholds are distributed around the value lift = 1 thus antecedents and consequents can be considered formed by independent variables. There is then a minority part of the distribution, but still numerous, which presents a positive and high value. This suggests that these rules are positively related. For this reason, they should be investigated more deeply. For instance, the three association rules showing the strongest positive relationships are:

- $\left\{\begin{array}{l}\texttt{MLow\_ln(AROU)},\\ \texttt{Low\_VAL},\\ \texttt{Low\_FAM}^3,\\ \texttt{No\_POLY}\end{array}\right\} \longrightarrow \{\texttt{VLow\_Log(FREQ)}\}$
- $\left\{\begin{array}{l}\texttt{High\_SIZE},\\ \texttt{High\_FAM}^3,\\ \texttt{Short\_LEN}\end{array}\right\} \longrightarrow \{\texttt{VHigh\_Log(FREQ)}\}$
- $\left\{\begin{array}{l}\texttt{High\_SIZE},\\ \texttt{High\_FAM}^3,\\ \texttt{Short\_LEN},\\ \texttt{No\_POLY}\end{array}\right\} \longrightarrow \{\texttt{VHigh\_Log(FREQ)}\}$

Particularly, their lift value range from 4.7 to 4.8. The last two rules are almost identical. The only difference is that the last AR antecedent is a superset of the second rule. They all are characterized by tiny support (1.1%-1.3%) and medium confidence (53%-57.8%). Nevertheless, apart from the `No_POLY` class, what these rules suggest is not trivial. Specifically, the probability that a word has a very high frequency is roughly 5 times higher if the word is short, with a high semantic size and a high familiarity. On the other hand, a word is nearly 5 times more probable to be very infrequent if it has a low valence, a low familiarity, and e medium-low arousal.

AR Predictions

As mentioned in the introductory section, association rules indicate co-occurrence and cannot be intended as predictive models such as classification models. Nevertheless, In this

section, we want to use some of the generated rules to try to see how they perform in this task.

We will apply these rules to the same test set used during the classification task. For this purpose, we use only those that have the POLY variable as consequents and in particular the positive class. To accomplish this task we select three association rules: the one with the highest confidence, the one with the highest support, and the one with the highest lift. However, the former and the latter is the same rule, that is:

$$\left\{\begin{matrix}\texttt{High\_Log(FREQ)},\\ \texttt{High\_CNC}, \texttt{High\_IMAG},\\ \texttt{Low\_ln(AROU)}, \texttt{Short\_LEN}\end{matrix}\right\} \longrightarrow \{\texttt{Yes\_POLY}\}$$

Once applied over the test set it produces the following results with respect to the positive class: accuracy = 91%, precision = 29%, recall = 4%, F_1 = 8%. For the negative ones, instead, precision = 92%, recall = 99%, F_1 = 95%. As the classifier already analyzed, this association rule has amazing performance over the negative class. The reason is that, as we already know, the distribution of the target variable is highly biased toward the negative class. Indeed, the 1388 predictions out of 1405 are all negative.

The second rule is characterized by the highest support, however the antecedent is composed of just a singleton itemset:

$$\{\texttt{Med\_LEN}\} \longrightarrow \{\texttt{Yes\_POLY}\}$$

Here, apart from a high precision on negative class, the rule performs quite badly in classifying both classes. Its results indeed are comparable to the random classifier: positive and negative votes are distributed equally. Therefore it produces 635 false positives, 656 true negatives, 25 true positives, and 89 false negatives.

References

Berthold, Michael R, Christian Borgelt, Frank Höppner, Frank Klawonn, and Rosaria Silipo (2020). *Guide to Intelligent Data Science*. Springer.

Giovanardi, Claudio (2010). *L'italiano da scrivere. Strutture, risposte, proposte*. Liguori Editore.

Joanes, D. N. and C. A. Gill (1998). "Comparing measures of sample skewness and kurtosis". In: *Journal of the Royal Statistical Society* 47.1, pp. 183–189.

Scott, Graham G, Anne Keitel, Marc Becirspahic, Bo Yao, and Sara C Sereno (2019). "The Glasgow Norms: Ratings of 5,500 words on nine scales". In: *Behavior research methods* 51.3, pp. 1258–1270.

Tan, Pang-Ning, Michael Steinbach, Anuj Karpatne, and Vipin Kumar (2019). *Introduction to Data Mining*. Pearson Education Limited.