# CLUSTERING DATING PROFILES

Peter Evans

Applied Data Science MSc | University of Buckingham

# Table of Contents


**EXECUTIVE SUMMARY** 3

**CHAPTER 1 – BUSINESS UNDERSTANDING** 4
- i) Introduction 4
- ii) Background and Relevant Literature 5
- iii) Ethical Considerations 7

**CHAPTER 2 - STRUCTURED DATA** 8

**2.1 - Data Understanding** 8
- i) Data Quality Assessment 8
- ii) Exploratory Data Analysis 11

**2.2 - Data Preprocessing** 14
- i) Feature Construction 14
- ii) Extreme and Erroneous Values 14
- iii) Feature Transformation and Encoding 15
- iv) Remaining Null Values 18
- v) Feature Selection / Extraction 19
- vi) Standardization 19

**2.3 - Modelling** 21
- Prefix - Cluster Evaluation Techniques 21
- i) K-Means 23
- ii) Reduced Data – K-Means 27
- iii) DBSCAN 29
- iv) Reduced Data – DBSCAN 33
- v) GMM 34
- vi) Shared Nearest Neighbours (SNN) 38

**2.4 Cluster Interpretation** 44

**CHAPTER 3 – TEXT DATA** 46

**3.1 – Data Understanding** 46
- i) Data Quality Assessment 46
- ii) Data Preprocessing (for EDA) 47
- iii) Exploratory Data Analysis 48

**3.2 – Preprocessing** 51

**3.3 - Modelling** 51
- i) Latent Dirichlet Allocation 51
- ii) Latent Semantic Indexing 57

**CHAPTER 4 – EVALUATION & NEXT STEPS** **60**

**CHAPTER 5 - CONCLUSIONS** **62**

**CHAPTER 6 - OUTPUT** **63**

**CHAPTER 7 - REFERENCES** **64**

**CHAPTER 8 - APPENDICES** **70**

**Appendix A – Structured Data EDA** **70**

# Executive Summary

This report presents an investigation into the clustering of dating app profiles, using both structured data features (that allow for selected input, i.e. height or age) and text features that are the users' responses to various in-app prompts designed to let the user reveal more about themselves.

The motivation for such an investigation is two-fold. Firstly, well-performing clustering algorithms could be used as part of a recommender system to present users with other profiles that they are likely to match. Secondly, it could provide a useful userbase breakdown for purposes such as targeted marketing or feature development.

Clustering was performed first on the text features, with most of the algorithms tested performing poorly as a result of the high dimensionality of the dataset after preprocessing. However, an algorithm based upon the shared-nearest-neighbour approach was able to produce what appear to be well-distinguished clusters, albeit with a large number of profiles discarded as noise.

In comparison, the segmentation of text features, which was performed on the output of the Latent Dirichlet Allocation and Latent Semantic Indexing topic modelling techniques, did not produce such well-defined clusters, with only the former resulting in any clusters at all.

In both cases, it is speculated that improvements in performance could be gained were a stricter preprocessing phase to be undertaken, guided by background domain knowledge.

# Chapter 1 – Business Understanding

## i) Introduction

The landscape of dating has been subject to a seismic shift in the past 10-15 years with the ubiquitous adoption of smartphones. Whilst online services were once a commitment that was reserved for the more seriously invested, nowadays meeting new romantic or sexual partners online has become, in many sociodemographic groups, the standard practice [1].

Resultingly, the market of online dating has grown to staggering levels, with 42 million users in the US alone in 2020 and the most popular application, Tinder, being valued by Morgan Stanley at over $40 billion [2, 3]. Thus, businesses in the space are constantly working to win users over the competition, and optimising their experience is a key step in doing so.

An obvious contribution to a user's satisfaction with a dating app is the potential partners that they are presented with. If they are repeatedly shown profiles that do not meet their preferences, then they are likely to lose interest quickly and move on. Therefore, it's imperative to learn what a user is looking for in a partner, and find profiles that meet their apparent requirements. Whilst it has been shown that typically this is someone with similar personality traits to themselves [4], dating apps also utilise the previous choices of a user to determine these preferences. In either case, it's necessary to establish a method to group profiles that are similar.

This project takes a dataset [5] of dating app users scraped from just under 60,000 profiles on OkCupid and examines the means by which they can be clustered into separate groups using established unsupervised learning techniques. Each profile consists of data of two different types: structured features that allow for a selected range of input (e.g. height, sex, orientation), and unstructured text features that are responses to prompts within the application (e.g. "I'm really good at" or "What I'm doing with my life"). The overall aim of the project is thus to complete a thorough investigation into how both of these types of data can be utilised within the clustering process, and observe how the performance of this clustering varies with the use of a variety of algorithms and techniques. This will require an exploratory data analysis for each set of features, some extensive preprocessing, and the use of appropriate optimisation methods for the chosen machine learning algorithms before a suitable evaluation of their performance.

The report structure is based upon the widely accepted standard for data mining investigations – the CRISP-DM framework [6], albeit with a few modifications to ensure a clear and logical composition for the task at hand. The data understanding, preprocessing, and modelling sections are all completed once for the structured data and then again for the unstructured data before a joint evaluation. Additionally, there is no deployment chapter as that is beyond the scope. The report then concludes with a summary and ideas for future work, followed by a description of the file structure of the code.

In terms of tools, the vast majority of the investigation will be carried out using Python and its open-source libraries Pandas [7] and Scikit-learn [8] for data manipulation and machine learning tasks respectively. Natural language processing of text data will be handled using spaCy [9] and NLTK [10] Additionally, some data analysis will also be carried out using Tableau, for its visualisations and dashboards.

## ii) Background and Relevant Literature

For as much as mobile dating apps have grown exponentially to generate huge user bases and revenue streams, there is still a surprising lack of publicly available research into the use of machine learning techniques in the domain. This is most likely a result of proprietary algorithms and pipelines created in-company being kept confidential from the competition.

It's found though that dating services have typically made use of both collaborative filter [11, 12] and content-based [12, 13] recommender systems to present users with items (in this case other users) that are likely to interest them. Such systems are also common in a wide range of industries such online retail and media consumption. In simplified terms, the former is a technique based solely off of the selections of other users – for each user, similar users are found using a similarity measure on their past choices, and then the top-rated items of those similar users are presented back to the user in question. Content-based systems, on the other hand, add features like age or location of the items so that the previous likes of the user indicate the type of profile they're interested in through metadata, rather than the activity of others. Of course, there are also many ways to combine these two methods together in hybrid systems [14, 15, 16].

One of the few recent insights into the workings of a major application was the presentation of TinVec, Tinder's own recommender system with neural-network trained embeddings, inspired by word2vec [17], that enable the representation of every profile with a vector [18]. User characteristics like their interests and career path are encoded as well as information from their profile's bio and images. A user is then

presented with similar profiles to those they have already 'liked' by taking the mean vector of the past 'likes' and finding profiles that have a high similarity with this mean. This inhouse system is able to predict the interest of a particular user ('swipe left' or 'swipe right') with an F1 score of 85%, demonstrating a very high performance level.

The focus of this report, however, is not on building a recommender system, but on the clustering of profiles into distinct groups using both text and non-text features. This is both to limit its scope considering the time constraints of a Masters project, and also a result of the data publicly available to work with – the dataset used is simply the content of profiles without any record of their historical choices. The motivation for such research is that a well-performing clustering technique could be utilised as part of the process of a recommender system. Additionally, it could provide useful knowledge of an application's userbase to inform future decisions around marketing and app design.

There are a myriad of algorithms that can be used for clustering, and the one that performs best varies between datasets [19]. Generally, they can be broken down into 4 main types. The first of these is centroid-based algorithms like the widely-used K-means [20], that are relatively simple to implement and scale well to datasets of any size [21]. Secondly, there are density-based algorithms such as DBSCAN [22] which have the advantage of being able to effectively handle noise and are not limited in the shape of clusters generated. Distribution-based methods like Gaussian mixture models [23] bring the option to undertake soft-clustering by assigning data points a 'probability' for each cluster, and finally hierarchical clustering methods [24] break clusters down into constituent groups.

Clustering research in the public domain has been performed mostly from a user behaviour or business standpoint rather than on the profiles themselves, with the use of K-means to group survey responses on Tinder use using revealing four main types of user in terms of problematic app use [25] and hierarchical clustering upon the top downloaded dating apps demonstrating that 42% of applications are aimed at a niche market (e.g. ethnic minorities or LGBTQ+) [26]. Analysis of text data has shown the difference in linguistic behaviour between those looking for long-term and casual relationships [27], with the former generally discussing personality traits and the latter using more diffuse topics.

The segmentation process in the case of text data can also utilise algorithms like K-means and agglomerative clustering [28], but a common way to handle such a task is to use some sort of topic modelling method [29]. These find not only a distribution of documents by topic but also a description of the different terms that comprise those topics. Often the former is performed after the latter to get topic-dependent clusters.

In either case, the transformation of text data into a matrix, like a bag-of-words representation or TF-IDF [30], is required.

It's important to consider that clustering is an ill-defined problem in comparison with supervised techniques like classification, as there is no gold standard to measure performance against. Whether or not the final result is desirable is totally dependent upon the purpose of the clusters, and thus most clustering algorithms optimise according to some pre-specified criterion [31]. Not only can the choice of algorithm and hyperparameters affect the 'goodness' of the results, but a change in this criterion can also mean that generated clusters are seen as good results when they were previously were not.

## iii) Ethical Considerations

Given the sensitive nature of the domain in question, appropriate cautions must be taken when undertaking any investigation. Not only is dating a personal topic that many people prefer to keep private, but the content of dating apps can strongly affect users' behaviour [32], which means the use of machine learning algorithms could have ethical implications [33].

In this case though, the dataset is sufficiently anonymised and was approved for distribution by the source application, OkCupid. Its use in clustering will not be problematic in this area as there is no personal information to be revealed nor have have any terms of service been broken. Additionally, the investigation is purely exploratory and will not involve production or deployment in any way, so there is no possibility of any behavioural manipualtion.

# Chapter 2 - Structured Data

## 2.1 - Data Understanding

### i) Data Quality Assessment

The dataset consists of 21 structured features in total, each with 59946 records. These features are detailed in Table 2.1. A more detailed description of the numerical features is found in Table 2.2 and the categorical features (the vast majority) in Table 2.3.

| **Feature** | **Categorical or Numerical?** | **Nominal or Ordinal? Interval or Ratio?** | **Datatype** | **Null Values** |
|---|---|---|---|---|
| **age** | Numerical | Ratio | int64 | 0 |
| **status** | Categorical | Nominal | Object | 0 |
| **sex** | Categorical | Nominal | Object | 0 |
| **orientation** | Categorical | Nominal | Object | 0 |
| **body_type** | Categorical | Nominal | Object | 5296 |
| **diet** | Categorical | Nominal | Object | 24395 |
| **drinks** | Categorical | Ordinal | Object | 2985 |
| **drugs** | Categorical | Ordinal | Object | 14080 |
| **education** | Categorical | Nominal | Object | 6628 |
| **ethnicity** | Categorical | Nominal | Object | 5680 |
| **height** | Numeric | Ratio | float64 | 3 |
| **income** | Numeric | Ratio | int64 | 48442 |
| **last_online** | Numerical | Interval | datetime | 8198 |
| **location** | Categorical | Nominal | Object | 0 |
| **job** | Categorical | Nominal | Object | 0 |
| **offspring** | Categorical | Nominal | Object | 35561 |
| **pets** | Categorical | Nominal | Object | 19921 |
| **religion** | Categorical | Nominal | Object | 20226 |
| **sign** | Categorical | Nominal | Object | 11056 |
| **smokes** | Categorical | Nominal | Object | 5512 |
| **speaks** | Categorical | Nominal | Object | 50 |

Table 2.1: General description of each of the features as in the original dataset. Datatypes are reported as those given by pandas on the initial dataset import. Null values for *income* include record with -1 listed, as this is assumed to be a default value.

| Feature | age | height | income | last_online |
|---|---|---|---|---|
| **mean** | 32.34029 | 68.29528 | 104395 | 2012-05-22 06:43:35.3 |
| **std** | 9.452779 | 3.994803 | 201433.5 | 77 days |
| **min** | 18 | 1 | 20000 | 2011-06-27 01:52:00 |
| **25%** | 26 | 66 | 20000 | 2012-05-29 20:37:15 |
| **50%** | 30 | 68 | 50000 | 2012-06-27 14:30:00 |
| **75%** | 37 | 71 | 100000 | 2012-06-30 01:09:00 |
| **max** | 110 | 95 | 1000000 | 2012-07-01 08:57:00 |

Table 2.2: Simple description of each of the numerical features in the dataset.

| Feature | Number of Unique Values | Top Value | Top Value Count |
|---|---|---|---|
| **status** | 5 | single | 55697 |
| **sex** | 2 | m | 35829 |
| **orientation** | 3 | straight | 51606 |
| **body_type** | 12 | average | 14652 |
| **diet** | 18 | mostly anything | 16585 |
| **drinks** | 6 | socially | 41780 |
| **drugs** | 3 | never | 37724 |
| **education** | 32 | graduated from college/university | 23959 |
| **ethnicity** | 217 | white | 32831 |
| **job** | 21 | other | 7589 |
| **location** | 199 | san francisco, california | 31064 |
| **offspring** | 15 | doesn't have kids | 7560 |
| **pets** | 15 | likes dogs and likes cats | 14814 |
| **religion** | 45 | agnosticism | 2724 |
| **sign** | 48 | gemini and it’s fun to think about | 1782 |
| **smokes** | 5 | no | 43896 |
| **speaks** | 7647 | english | 21828 |
| **education_status** | 3 | graduated from | 39376 |
| **education_level** | 7 | college/university | 31467 |

Table 2.3: Simple description of each categorical column in the dataset.

These descriptive summaries indicate some care needs to be taken when handling this dataset. There are a lot of null values present, far too many to simply exclude those records without valid data in every column. This incompleteness is a problem to be expected with social media profiles where the user has the choice of how much information they want to provide, as not everyone is going to fill in every option.

It is also to be expected with user-inputted data that not all of the information provided will be correct, as is the case. Evidence of this is found in the *height* column of Table 2.2, as the minimum value inputted is 1 inch, obviously not an accurate height input.

The reasons for these discrepancies will vary between each feature, and could be a combination of multiple. For example, it could be that a user has chosen to not fill an option because they believe listing this information will make them less attractive to other users, because they feel they don't fit into the given options, or it simply could be that they missed the option to provide this information. Likewise users' data may be incorrect because they were dishonest in an attempt to make themselves more appealing, they could have chosen to set the value as something completely unrealistic in jest, or it could have been a simple input error.

It's clear that the dataset's incompleteness and inaccuracy means each feature will have to be inspected on a case-by-case basis to determine what records to exclude, how to impute null values if necessary, and if any feature engineering can provide more insight.

On the other hand, there are no redundant features present in the structured dataset (each contributes its own unique information), and there are no duplicate records present.

In terms of timeliness, the profiles were all scraped in the early 2010s and although the landscape of online dating will have changed in terms of both users' attitudes towards dating apps as well as the design of the applications themselves, similar information is still collected in users' profiles today. This means the investigation and evaluation of segmentation techniques with both structured and non-structured data will still be relevant.

With regard to any sampling bias, the source [5] states that these profiles were scraped on June 30$^{th}$ 2012 and limited to those living within a 25 mile radius of San Francisco, with an 'active' profile on June 26$^{th}$ 2012 meaning that it had been online within the last year and had at least one photo. The dataset therefore has a very obvious bias in that it includes the individual particularities of San Francisco, and the demographic of

profiles present are likely to mirror these. One example is the larger proportion of men marked as gay (11%) compared with that for women (6.5%), reflecting the well-known above average population of gay men in the city [34]. It's not necessary to adjust or resample the data in any way for this, but it should be considered during the evaluation stage.

## ii) Exploratory Data Analysis

For the reader's convenience, the exploratory analysis is limited in the main report body to only a few key features. The analysis of the remaining features can be found in Appendix A – Structured Data EDA.

### Sex

The split of the users by gender is 59.8% male to 40.2% female, with 35,829 to 24,117 accounts respectively. Previous investigations [35, 36] have found men to be more common users of dating apps, so this is not a surprising split. It should be noted that the app, at the point of data collection, only permitted the selection of two gender choices [5].

### Location

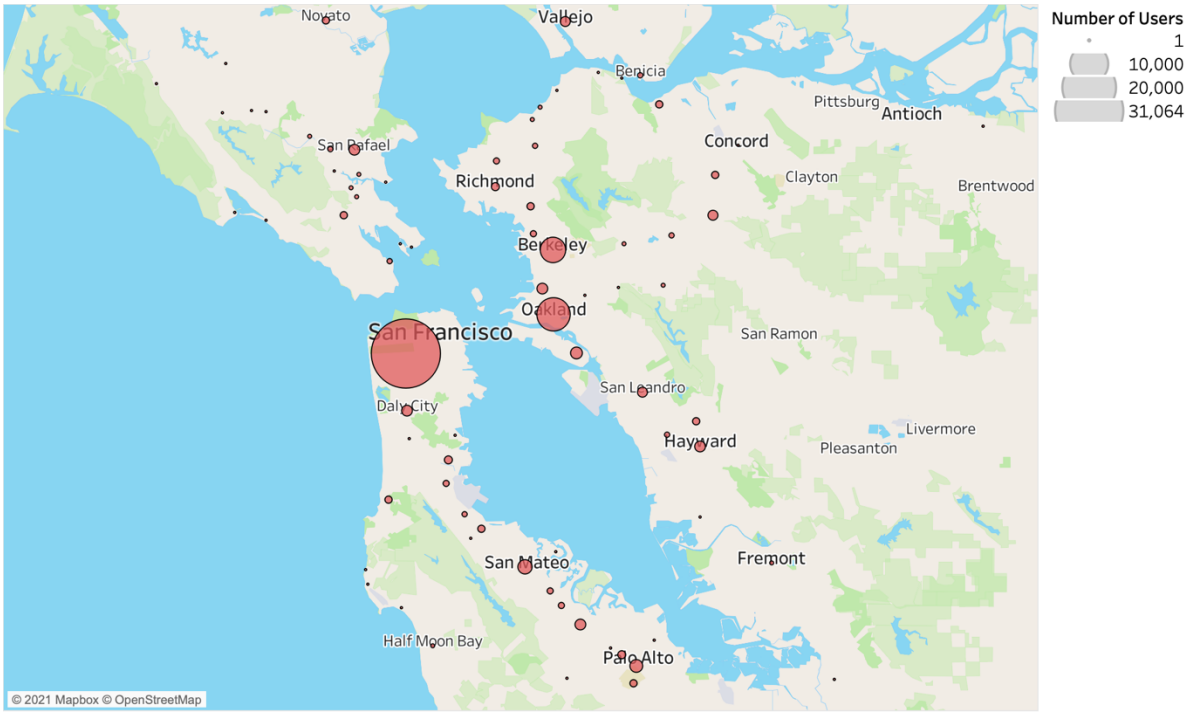


Figure 2.1: Map of users' locations around the San Francisco Bay Area.

As was stated in [5], the profiles scraped were chosen to be within a 25 mile radius of San Francisco, and it is seen in Figure 2.1 that this is true for the vast majority of the

users. However, there is a small proportion listed with locations around other areas of the US and the rest of the world. This could be perhaps a result of their profiles not updating after travelling, or it could be that they were able to choose their listed location and chose to input somewhere else.

### Age

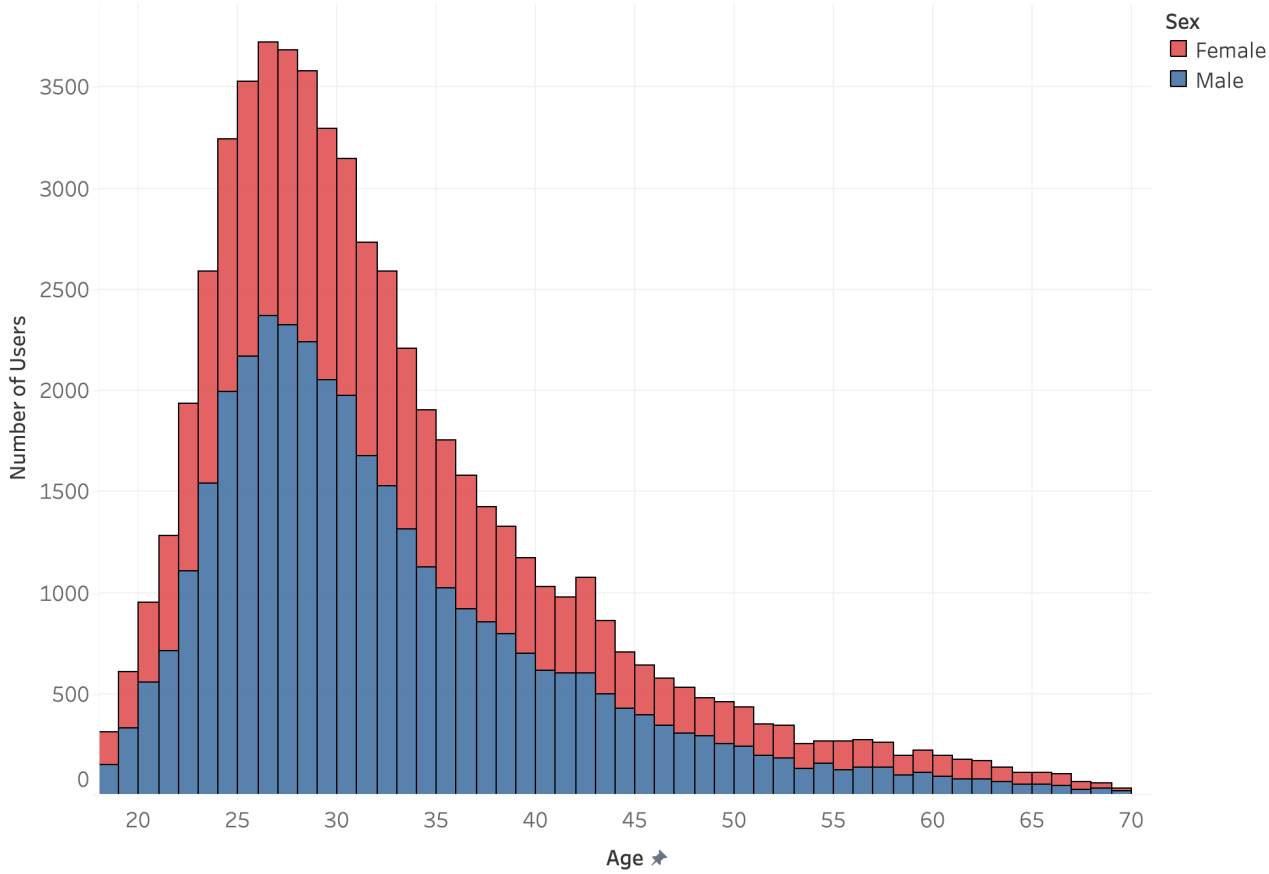


Figure 2.2: Histogram of all user ages, with each bar split by sex. Age axis limited to 70, as there are only 2 users above that range at 109, 110 respectively.

There is a very clear right-skewed distribution of *age* visible in Figure 2.2. This is unsurprising as dating apps like OkCupid have an age restriction of 18+ [37], thus giving the distribution a lower boundary while there is no such upper limit.
The modal value is 25 for women and 26 for men, while the overall median of 30 is to the right of both as would be expected for such a distribution. It makes sense for the peak of users to fall in the younger generation due to both their higher exposure to technology but also their higher likelihood to still be looking for a partner and not to be settled down. The gender split of each bin sits roughly around 60% male, there is no trend to a certain gender with age.

A small peak is visible at 42 years old, that is more prominent in the female distribution than the male. It is unlikely to a result of anything other than random variation.

### Status

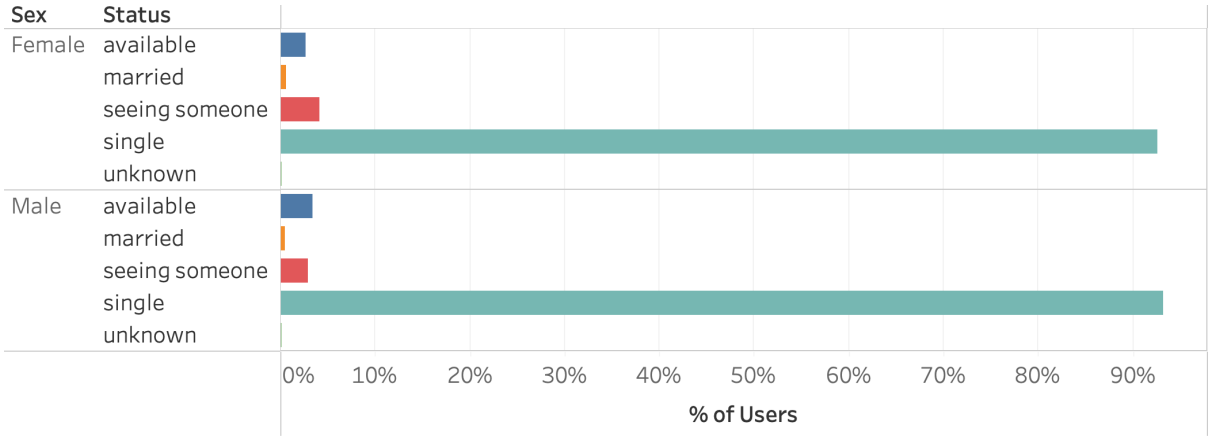


Figure 2.3: Percentage of users' relationship statuses by gender.

There are negligible differences between the status distribution of the genders seen in Figure 2.3. By far the modal value is *single*, which is logical given the aim of dating apps.

### Orientation

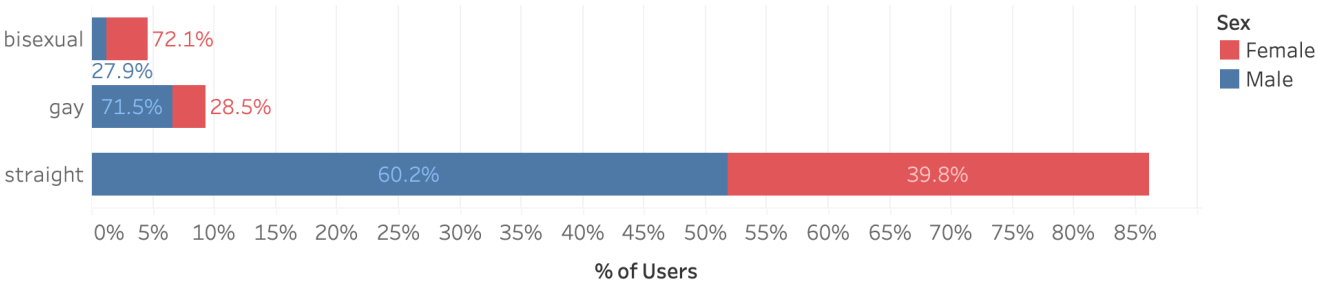


Figure 2.4: Percentage of users by orientation. Each bar split by gender, and labelled by its percentage split for that orientation.

The majority of users (86%) described themselves as straight, with a gender split mirroring that of the whole dataset. It is however interesting to note the majority of users listed as gay are men, and that the majority of users listed as bisexual are women (especially significant considering the larger proportion of men overall). The majority of gay men could be a result of numerous possible factors including the dismissal of a large number of gay men from the Navy in San Francisco [34]. Women's propensity to list themselves as bisexual fits with previous findings that they have a higher 'erotic plasticity', that is that their sexuality is influenced to a much larger extent by social factors such as cultural events and peer influence [38], and thus they are more likely

to be interested in same-sex relationships. Perhaps if there was a bi-curious category available to select, that would reduce the female majority of bisexuals.

## 2.2 - Data Preprocessing

In order to prepare the data for unsupervised modelling, it must be prepared in a suitable format. As mentioned in Chapter 1i) , the modelling will be undertaken using Python and its open-source library, Scikit-Learn. Models constructed with this library can only operate with numerical data and therefore the categorical features of the dataset must be encoded accordingly. Additionally, null and extreme values will be investigated and handled to address the problems discussed in Chapter 2.1i)

### i) Feature Construction

Multiple new features have already been created during the exploratory data analysis. These were all crafted by some string separation of the original feature. A summary is shown in Table 2.4. For those records where the original feature does not contain all the information required for the constructed features, then null values are imputed accordingly.

| **Original Feature** | **Constructed Feature(s)** |
|---|---|
| *diet* | *diet, diet_strictness* |
| *education* | *education_level, education_status* |
| *offspring* | *has_children, wants_children* |
| *pets* | *has_dogs, has_cats,*<br>*likes_dogs, likes_cats* |
| *religion* | *religion, religious_commitment* |
| *sign* | *sign, sign_seriousness* |

Table 2.4: Summary of constructed features.

### ii) Extreme and Erroneous Values

A number of records are excluded from the dataset based on their belief to be incorrect or extreme cases.

Firstly, there are users whose *location* value is very far outside the listed 25-mile radius from San Francisco. This is problematic as these records do not fit the intended collection criteria and thus indicate that they could be somewhat erroneous. Additionally, utilising profiles from other locations would mean that the scope of the

dataset would become unclear - it would not be specific to the San Francisco region nor generalised to a larger area. Exclusion of the 3926 profiles outside of this radius is completed using the GeoPy Python library.

The *age* feature appears to have been cut at 70, with around 30 profiles at 69 then no more until the two aforementioned at 109 and 110. It's clear that these ages are therefore erroneous and thus the two profiles are excluded from the dataset.

The *height* feature contains some non-sensical values as referenced in [5], which excluded values outside the range of 55 to 85 inches. The same range is utilised here, excluding 72 profiles in total.

In *education*, there are a total of 1672 containing 'space camp' as the level of education. It is assumed that this is chosen by users as more of a tongue-in-cheek option and thus is not an accurate representation of their true education. Therefore these values are set as null - to be addressed below.

## iii) Feature Transformation and Encoding

Two commonly encoding techniques, binary and one-hot encoding, are utilised to transform some of the categorical features into numerical data. The former is used on *sex*, *has_dogs*, *likes_dogs*, *has_cats* and *likes_cats*, while the latter is used for *orientation*, *body_type*, *status*, and *job* features. Null values in the one-hot encoding process are given a column just like any other value. This is because it is postulated that if someone chooses not to list a value for a certain feature, this information could be as indicative of their personality as whatever the true value is. Therefore this technique is also used for *has_kids* and *wants_kids*, as although the input values suggest that they could be represented ordinally, doing so would leave a very large proportion of the records null for these features (59% and 83% respectively) which would have to be deleted or their values imputed – either way losing a lot of information about the cohort of users. The same logic is applicable to all the features constructed from *pets*, *sign_seriousness* and *drugs* so these are one-hot encoded as well.

Whilst not quite one-hot encoded, a very similar process is performed on the *ethnicity* feature, where the newly generated columns are each individual ethnicity, and each record can be listed as 1 in as many ethnicities as is necessary, meaning that those people with 2+ ethnicities are accurately represented without the need to create a feature for every unique combination.

Other features have categories with a clear order that enables them to be ordinally encoded – this is true for *drinks* and *smokes*. These are changed to numerical values of 0 to 5 and 0 to 2 respectively. These transformations are summarised in Table 2.5.

| **Feature** | **Original Categories** | **Numerical Representation** |
|---|---|---|
| ***drinks*** | not at all | 0 |
| | rarely | 1 |
| | socially | 2 |
| | often | 3 |
| | very often | 4 |
| | desperately | 5 |
| ***smokes*** | no | 0 |
| | sometimes, when drinking, trying to quit | 1 |
| | yes | 2 |

Table 2.5: Summary of new ordinal features.

Multiple features contain values that are composed of a category and an order/severity level – for example *diet* contains multiple types of diet (vegetarian, halal, vegan etc.) and is accompanied by *diet_strictness* to which the user follows said diet (mostly, strictly). The same can be said for *religion* and *religious_commitment*, *education_status* and *education_level*, and *speaks.* Thus for all these variable pairs, a set of new features is created. This set is composed of a feature for each of the original categories, each of which contains numerical values according to the severity level/order. This should be made clear by the outline in Table 2.6 and the example of *diet*'s transformation in Table 2.7. It is noted that for *speaks*, as there are 77 unique languages, a feature is not created for each. Instead only the top 10 categories are taken, and any language not in this top 10 is set to 'other'.

Finally, due to the majority of users leaving their *income* blank, it is changed to a binary feature – becoming whether or not they listed their income, rather than the income itself. This is because it is unlikely that the income values listed are very accurate – there are only distinct inputs available and it's quite possible that users inflate this to try and appeal to others. It's deemed that the user's decision to list their income or not is a more useful feature.

| Original Feature(s) | New Features | Severity Level Values |
|---|---|---|
| ***diet, diet_ strictness*** | *diet_ anything*<br>*diet_ vegetarian*<br>*diet_ vegan*<br>*diet_ halal*<br>*diet_ kosher*<br>*diet_ other* | 'mostly' – 1,<br>'strictly' – 2<br><br>(*d_ anything* does not contain the severity and is simply set to 1 or 0. This is because 'mostly anything' and 'strictly anything' are nonsensical). |
| ***religion, religious_ commitment*** | all preceded by *religion_* :<br>*agnosticism,*<br>*atheism,*<br>*christianity,*<br>*catholicism,*<br>*judaism,*<br>*buddhism,*<br>*hinduism,*<br>*islam*<br>*other*<br>*nan* | 'laughing about it' – 1,<br>'not too serious about it' – 2<br>'somewhat serious about it' – 3<br>'very serious about it' - 4<br><br>(as for diet, one column does not use the level of commitment – the *nan* column) |
| ***education_ status, education_ level*** | *working on,*<br>*graduated from,*<br>*dropped out* | 'high school' - 1<br>'two-year college' - 2<br>'college/university' – 3<br><br>{'masters program', 'med school', 'law school', 'ph.d program} - 4 |
| ***speaks*** | *s_ english,*<br>*s_ spanish,*<br>*s_ french,*<br>*s_ chinese,*<br>*s_ german,*<br>*s_ japanese,*<br>*s_ italian, s_ c++,*<br>*s_ russian,*<br>*s_ portuguese,*<br>*s_ other* | 'poor' -1<br>'okay' – 2<br>'fluent' - 3 |

Table 2.6: Outline of new features created from original features with a combination of categories and a severity level.

| **Index** | ***anything*** | ***vegetarian*** | ***vegan*** | ***halal*** | ***kosher*** | ***other*** |
|---|---|---|---|---|---|---|
| 171 | 0 | 0 | 0 | 0 | 0 | 2 |
| 214 | 0 | 1 | 0 | 0 | 0 | 0 |
| 308 | 0 | 0 | 0 | 0 | 0 | 2 |
| 399 | 1 | 0 | 0 | 0 | 0 | 0 |
| 400 | 1 | 0 | 0 | 0 | 0 | 0 |

Table 2.7: Sample of transformed *diet* feature.

## iv) Remaining Null Values

Many of the feature engineering processes in the previous section accounted for the null values in the original variable, be that via one-hot encoding or adaptation of the variable to a binary form, as was the case for *income*. However, there are still many features that must be addressed so that the dataset can undergo further processing. For *height* and *speaks*, the number of null values is relatively small so these records are simply removed. For other features, the modal value is used to impute these values. This is true for *drinks* and *smokes*.

For the transformations using a severity level outlined above, the imputation of null values took place before the transformation was completed. For the records where *diet_strictness* was marked as null, the value was imputed as the mode of that specific diet, which for all apart from 'anything' was 'mostly'/1. For records with no *diet* listed, they were assumed to be 'anything' as this is the modal diet and also it is logical that users wouldn't list their diet if they do not follow one.

The same process was undertaken for religion, with the commitment level imputed as the modal value for the user's chosen religion before the transformation. However here, null values of the category itself, *religion*, were left and accounted for with a binary *religion_nan* column as they comprised a much bigger proportion of the profiles.

A null value of *education* does not signify the same thing as one in *diet* or *religion* - everyone has some level of education (bar extreme circumstances). The imputation here was more involved, and users with null values were assigned the modal value of their age bracket (created as 18-20, and then every 5 years up to 70 years old). Age was chosen as it's natural that those who are older have had more chance to advance in education.

Although all profiles with a null value of *speaks* were removed from the dataset, many of the users did not specify a fluency level with their listed languages. It's assumed that this is because they are native speakers, so the level was imputed as 3 in these cases.

## v) Feature Selection / Extraction

After all the feature engineering and data cleaning steps have taken place, the dataset now consists of 116 features and 55,903 records. Whilst the set does have many more rows than columns, the high level of dimensionality could still cause problems during the clustering process.

A common technique to reduce the dimensionality of a dataset is to extract the majority of the variation via principal component analysis (PCA). Often, the linear combinations of features generated through Singular Value Decomposition can represent a large proportion of this variation with a number of principal components much smaller than the number of original variables.

To investigate if this is the case, PCA is performed on a standardized version of the cleaned dataset to produce a scree plot, , and a graph of cumulative variation representation, Figure 2.5. It's clear from both of these figures that although a reduction in dimensionality could take place while representing almost 100% of the variation in the dataset, it would only reduce the number of features by about 15. Therefore, any improvement in clustering performance gained by using less dimensions would likely be relatively small, and not worth the reduced interpretability of the generated clusters.

Whilst a sacrifice in the amount of variance represented could be taken to reduce the dimensions further, there is no clear compromise. Even with the percentage of variance reduced to 80%, this would still require around 70 dimensions. Thus, the decision is taken, for now, to move forward without performing PCA on the data.

However, some features are removed as they do not fit the project goal: the variables *last_online* and *location* as they would likely be another part of the recommender system used by a dating app, and *sign* is also removed as it is determined to not be an important factor of a person's characteristics, and it would likely create unwanted clusters due to its spread of users over many categories as seen in Figure 8.17.

## vi) Standardization

As many of the algorithms used for clustering rely on distance measures, it's important to standardize the data beforehand so that each dimension has an equal contribution. This is performed using Scikit-learn's StandardScaler function to convert each value to a z-score.

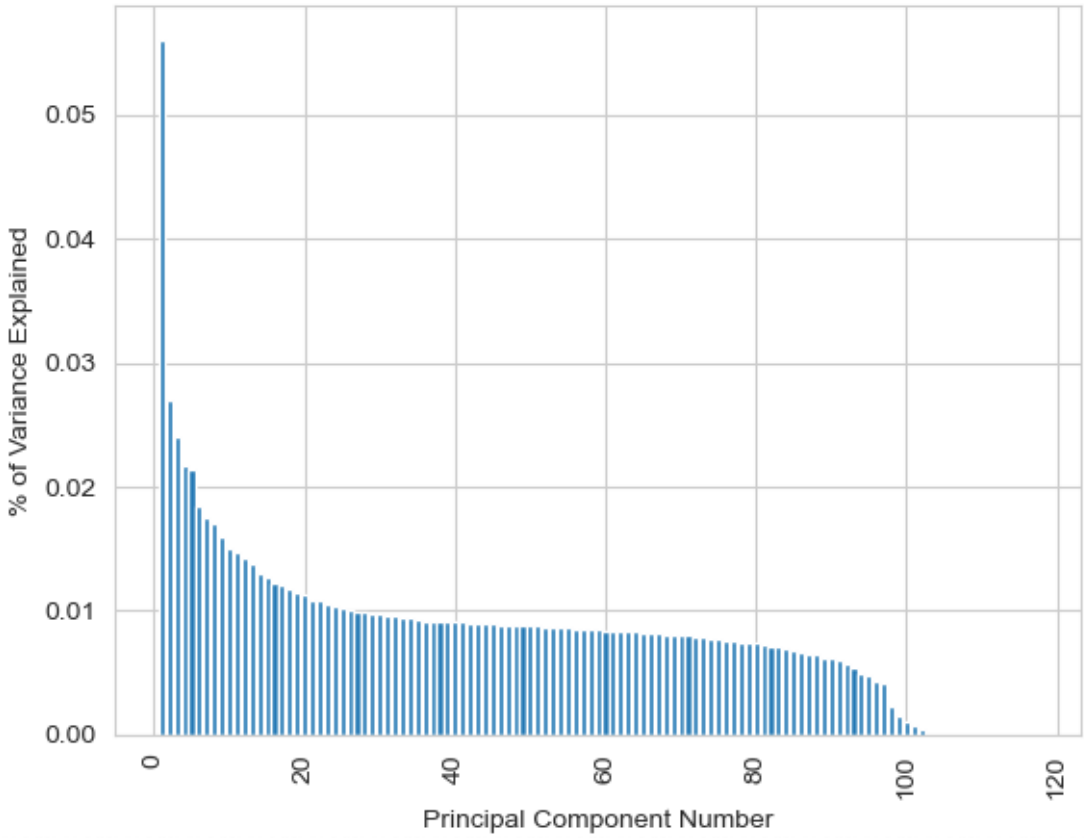


Figure 2.6: Scree plot of the PCA of the cleaned and encoded profile dataset.

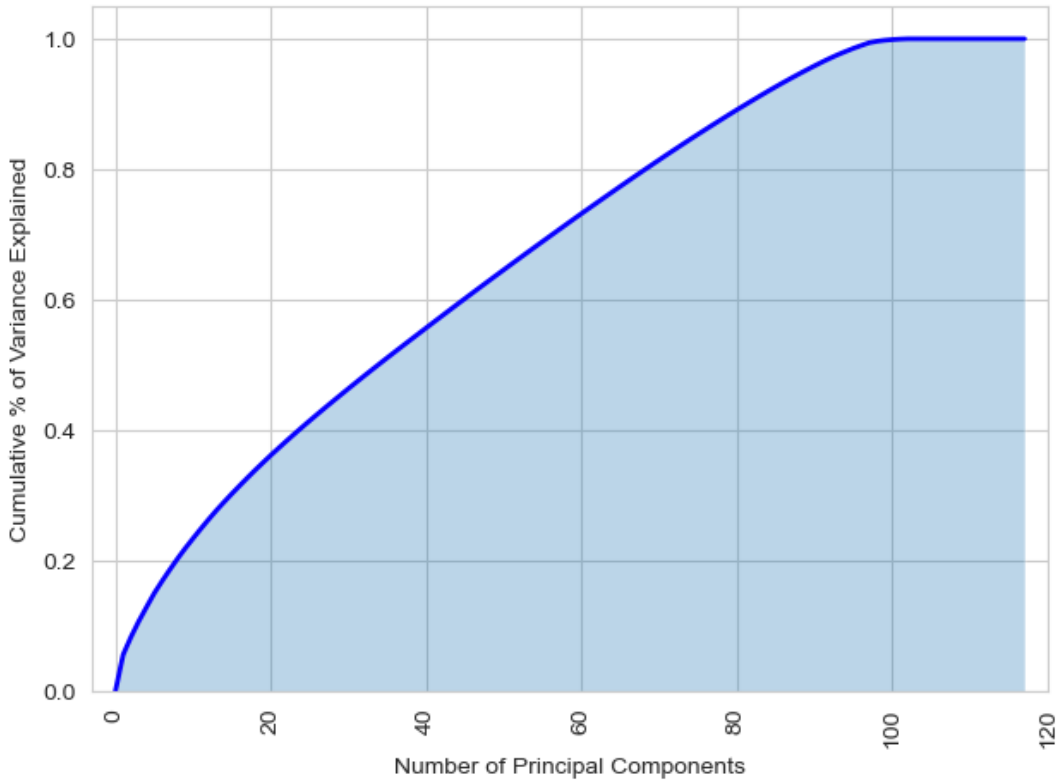


Figure 2.5: Cumulative plot of the total dataset variance represented with the principal components obtains through PCA.

## 2.3 - Modelling

### Prefix - Cluster Evaluation Techniques

#### t-SNE

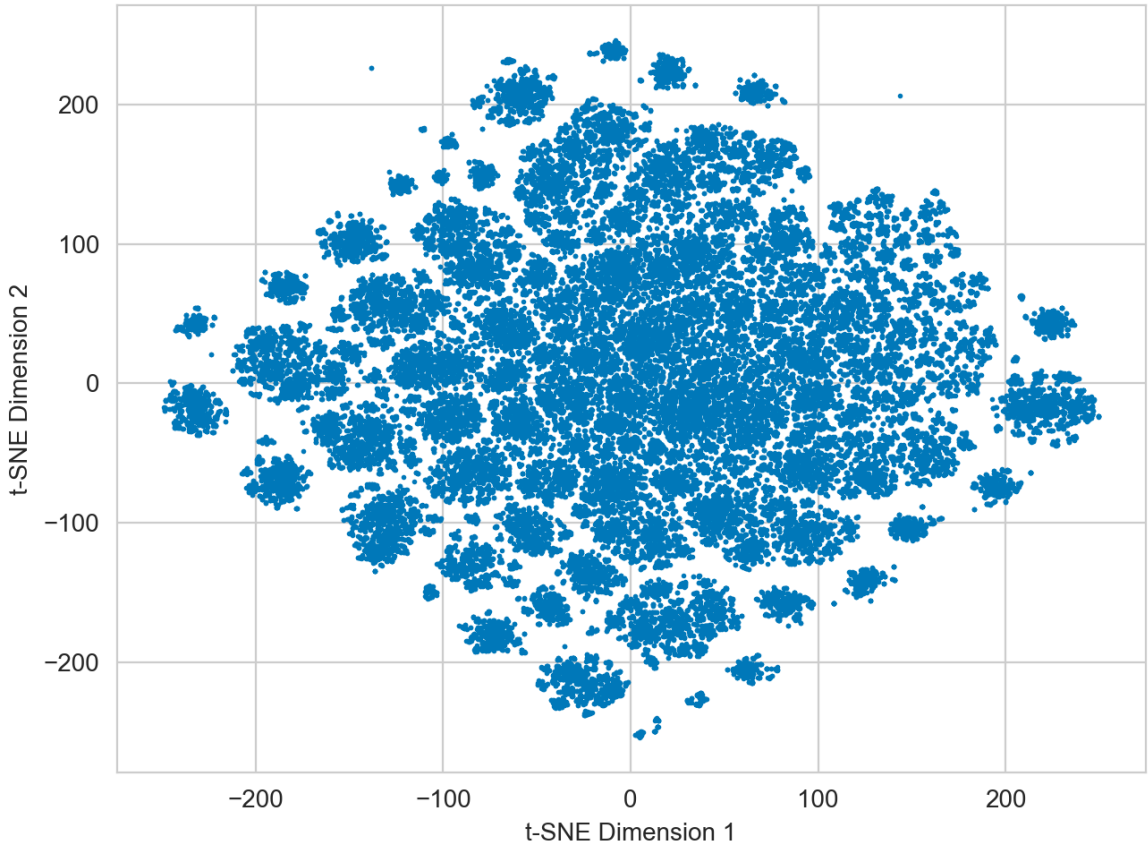


Figure 2.7: t-SNE plot of full, cleaned dataset of profiles after standardization. Generated using the Scikit-learn function, with perplexity = 10, and run for 5000 iterations. Other parameters left as function defaults.

As discussed in Chapter 1ii) , evaluating the performance of clustering techniques is more complex than would be the case in supervised learning tasks like classification or regression. There is no gold standard or absolute truth to compare the results against. It is for this reason that a more comprehensive approach must be taken.

There are a range of metrics that can be used to assess performance, one of which, the silhouette coefficient, shall be used here. However, it is also necessary to consider the interpretation of results by the reader and, as such, the t-SNE method [39] for visualising high dimensional data in a 2D plane is utilised. This is a very powerful tool as it can help give an intuition for a dataset that otherwise is very difficult to grasp given the complexity of a high N-dimensional space. In this section, the clusters

generated by each algorithm will be visualised using the t-SNE transformed form of the dataset. However, some caveats must be kept in mind when viewing it.

To begin with, the 2D representation of the dataset generated by t-SNE can vary massively with the choice of the perplexity input parameter. Stated in the original paper to be typically robust between values of 5 and 50 [39], lower values result in an increased attention of the algorithm on the data at a local scale, and higher values more on the global. Plots were generated for perplexity values of 5, 10, 20, 30, 40, 50 and 60, with the most 'natural-looking' clusters appearing in the plot with perplexity = 10. The reason for choosing this plot is that the data does not look clumped together as a ball but neither does it seem split into too many densely populated clusters. This plot is seen in Figure 2.7.

Additionally, the reader should be aware that due to the nature of how the method works, insight cannot be drawn from the cluster sizes nor their separation as the notion of distance in the plot is non-linear, it scales with the regional density variations of the data [40]. Also important to note is how the technique can result in random noise looking somewhat non-random at low perplexity values. However, given the choice to use a perplexity value equal to 10 (in the robust range), this shouldn't be too much of an issue.

### Silhouette Coefficient

The aforementioned silhouette coefficient can provide the quantifiable assessment required to be able to compare one clustering method against another objectively. It is a metric that is calculated for each sample in the dataset, by measuring the mean distance to members of its cluster (mean intracluster distance, $a$) and the mean distance to the members of the nearest cluster that the sample is not a member of (mean nearest cluster distance, $b$). The formula for that sample's coefficient is then given by the equation below.

$$s = \frac{b - a}{max(a, b)} \tag{2.1}$$

The resulting value ranges from +1 to -1, with a high positive value indicating that the point is well separated from its nearest cluster, a value around 0 indicating that it is on the boundary between its own and another cluster, and a value close to -1 indicating that it has been incorrectly assigned. These coefficients can be plotted all together on what is known as a silhouette plot, or they can be averaged for all points across a set of clusters.

However, this metric, along with many others, requires the use of a distance measure to evaluate the quality of the clusters produced. Such measures, particularly the Euclidean distance, lose their validity when applied to high-dimensional data. This is a problem for almost all cluster evaluation metrics, but an attempt will still be made to utilise it as there are a lack of alternatives.

### i) K-Means

One of the most well-used and simple-to-implement segmentation techniques is k-means clustering. The algorithm aims to form well partitioned clusters by minimising what is known as the inertia - the sum of squared distances of each cluster member to its 'centroid'.

Whilst this technique can often be used to produce high quality segmentation, there are many disadvantages to its use that will likely prove particularly problematic given our knowledge of the domain in question. Firstly, it requires that the number of clusters be specified before the algorithm is run, and works best with not too many [41]. It would be expected for this dataset, given the wide variation in the userbase and the high number of dimensions, that the ideal number of clusters would be quite large, and this expectation agrees with the 2D visualisation of the data seen in Figure 2.7. Additionally, this high level of dimensionality creates problems for the algorithm as it can cause the Euclidean distances used in the calculation of inertia to become inflated, and there is no option to change the distance measure in the sklearn function that will be used [41]. However, it could still be useful to test the algorithm on the dataset to provide a base point for comparison.

As aforementioned, the number of clusters is a parameter that must be inputted to the algorithm before running it, and to determine the best value of said parameter, $k$, an empirical technique known as the elbow method is utilised. This is simply plotting the inertia obtained for a range of values of $k$, and finding the 'elbow' in the plot where an increase in the $k$ value no longer results in a significant drop in inertia.

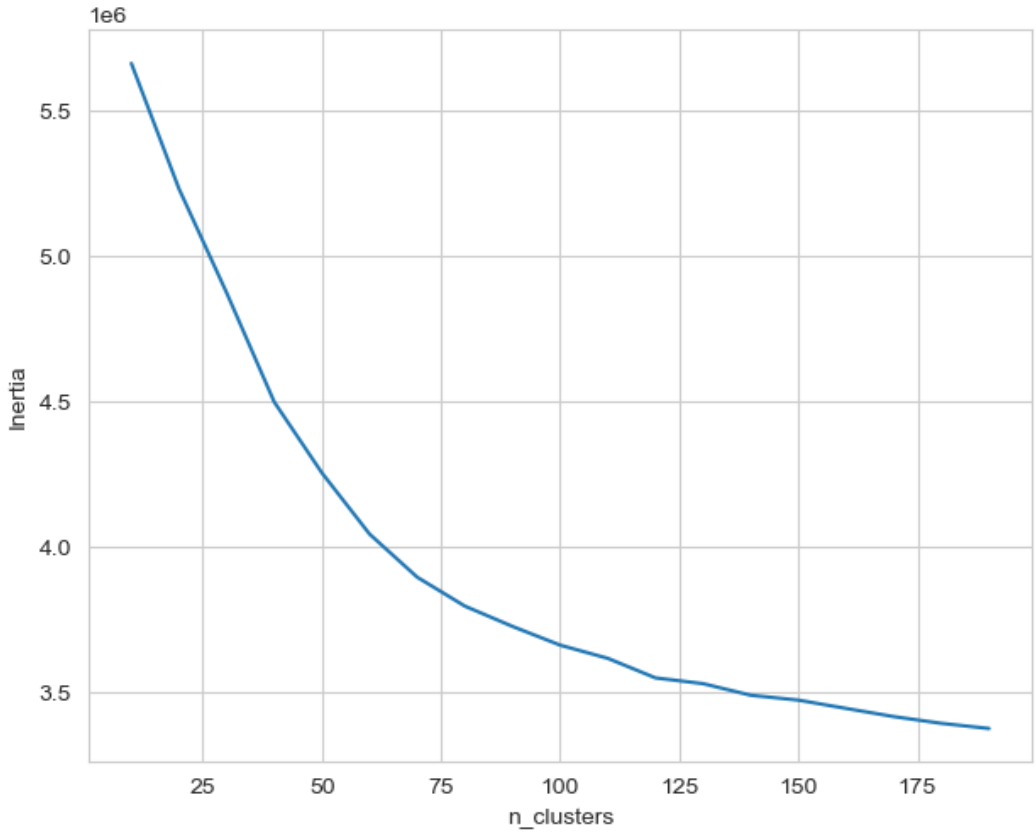


Figure 2.8: K-means elbow plot, with values of $k$ ranging from 10 to 190 in steps of 10, and other parameters left as Scikit-learn defaults.

The elbow plot for our data can be seen in Figure 2.8, with a range of $k$ values tested between 10 and 190. However, there is no clear elbow visible, indicating that likely the algorithm is not performing well as a result of 'the curse of dimensionality', as predicted. The plot was also generated for values of $k < 10$, and no elbow was seen there either.

Another technique to choose the value of $k$ is to look at the change in the overall silhouette score with different numbers of clusters. This would be expected to produce a peak value at the optimum $k$, but as we can see from Figure 2.9, the maximum score generated is with $k = 60$ at just under 0.05. This is a very small maximum value, so it's unlikely this truly corresponds to the ideal number of clusters. Additionally, the score was calculated for every value of $k$ under 10, and a slightly higher score was found at $k = 2$ of 0.06.

Finally, the silhouette scores are generated again, but with the $L_1$ (otherwise known as Manhattan) distance metric used in the calculations of $a$ and $b$. This step is undertaken as the $L_1$ distance has been found as preferable for high dimensional tasks [42]. The results of this can be seen in Figure 2.10. In this case, the silhouette score indicates a clear downtrend in cluster performance with k, with the maximum value also found at

the lowest value of $k = 2$ at 0.15. This is still very low, and once again it is unlikely this is a true insight into the structure of the data. Also it's expected and desired to be able to find a larger number of clusters than this.

Judging by the silhouette scores, it would seem that the use of the k-means algorithm to cluster this dataset has not been successful. The assigned clusters are shown in Figure 2.11, given the 'ideal' value of $k = 2$ identified. Again, although comparison to the t-SNE plot is not definitive, it appears that there is no agreement between the two techniques - almost all of the clusters isolated by the visualisation method are split between the k-means clusters.

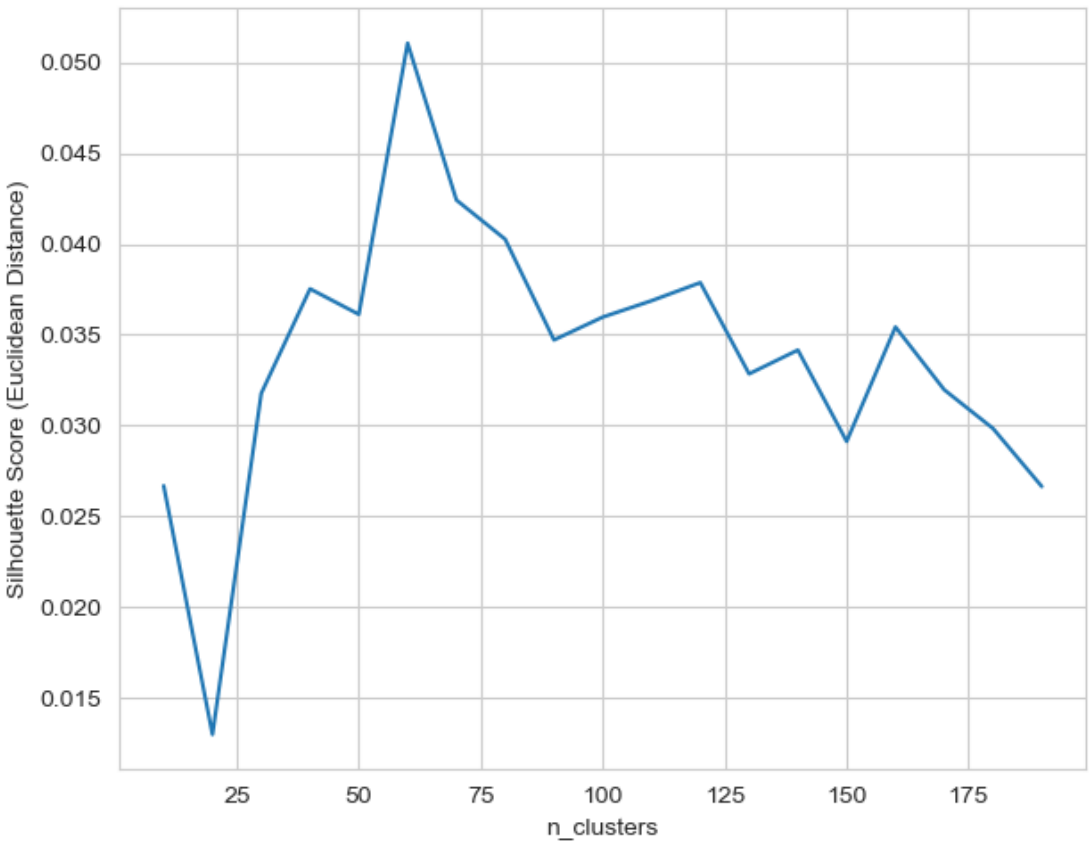


Figure 2.9: Overall silhouette score obtained with different k-values. Again, all the rest of the parameters in the Scikit-learn function were left as default.

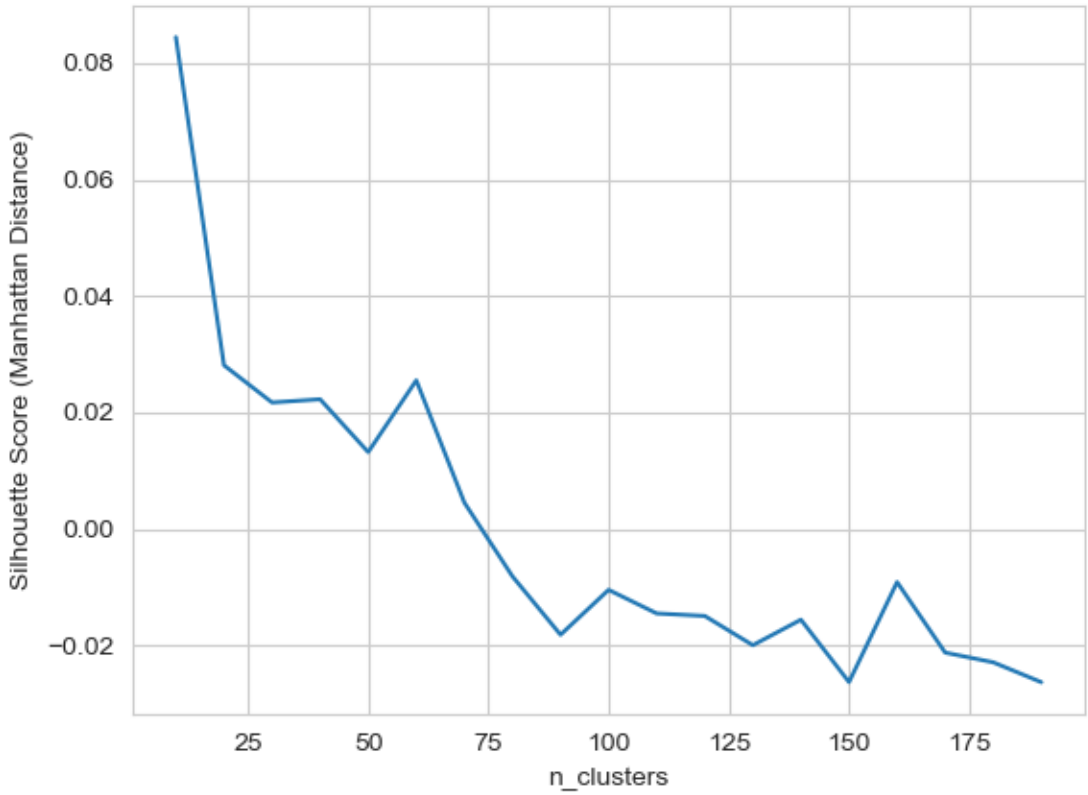


Figure 2.10: Silhouette score obtained with different values of k, calculated using Manhattan distance, averaged over all points in the dataset.

Figure 2.11: Clusters generated with $k$-means for $k = 2$.

## ii) Reduced Data – K-Means

Although the performance of clustering is always somewhat subjective, it's quite clear that in the case of K-Means above that it was poor, and it is believed that this poor performance is in large part due to the high dimensionality of the dataset after preprocessing.

Unsupervised machine learning in high dimensions is still an active area of research, and there are many different techniques suggested to handle the problem. For example, using distance metrics with a fractional norm [42, 43] or adapting clustering algorithms to account for 'hubness' [44]. However, applying such techniques to the dating app profiles would require a long investigation and diverge from the project's aim of comparing text-based and non-text-based clustering.

As such, the decision is taken to also use a reduced dimension version of the dataset and attempt the method again, despite the large loss of variation and cluster interpretability that this will impose. In order to minimise the problems caused by the high dimensionality of the dataset, it is transformed to only the first 8 principal components calculated through PCA. Although these represent only 20% of the variation in the data, it should ensure that the validity of the distance metrics is maintained.

The same processes of k-means optimisation are used as before, the elbow method and silhouette score plots are shown in Figure 2.12 and Figure 2.13 respectively. Note - here the silhouette score is only calculated with Euclidean distance as is standard with a normal level of dimensionality.

Once again, there is no clear elbow point visible, and the maximum silhouette score is found at $k = 2$. Figure 2.14 shows the clusters generated, which look very similar to those in part i)

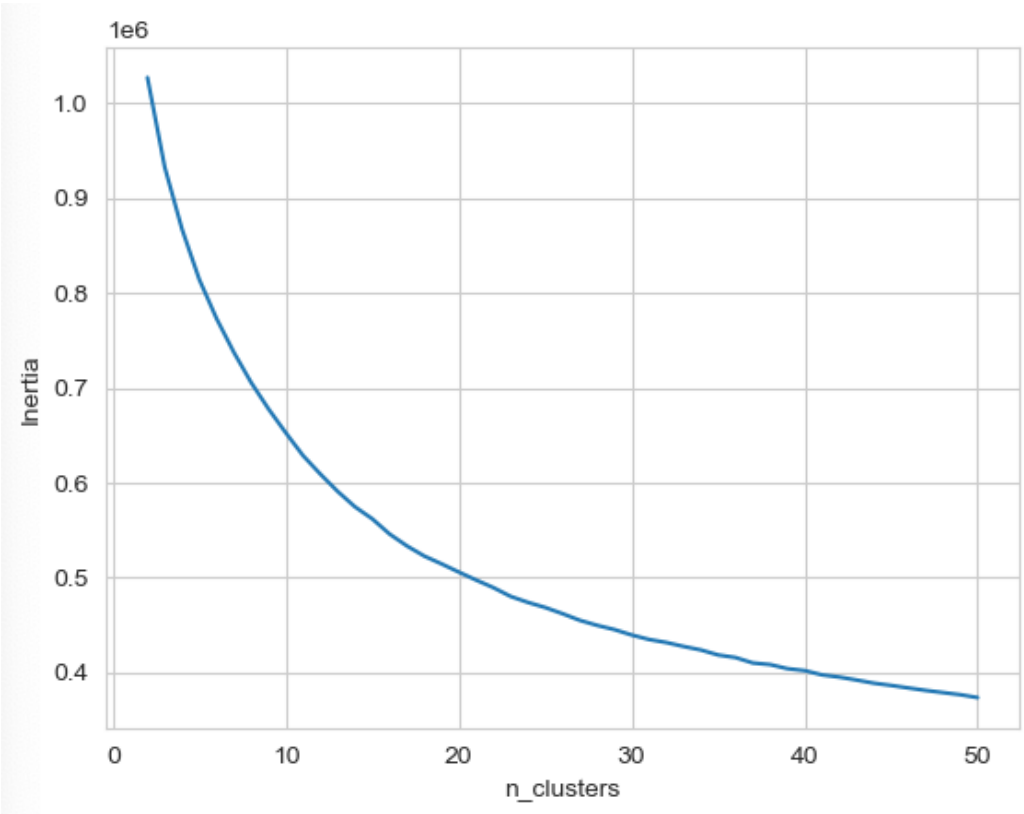


Figure 2.12: Plot of inertia with $k$ for $k$-means clustering on the reduced dimension dataset.

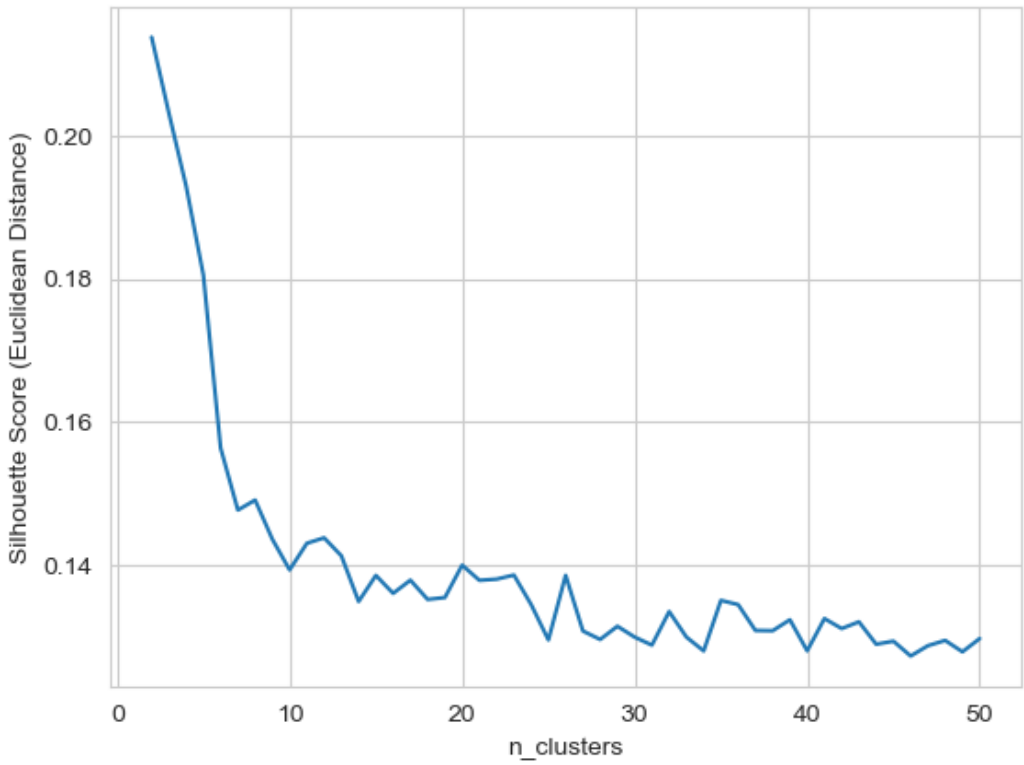


Figure 2.13: Silhouette score with $k$ for $k$-means on reduced dimension dataset.

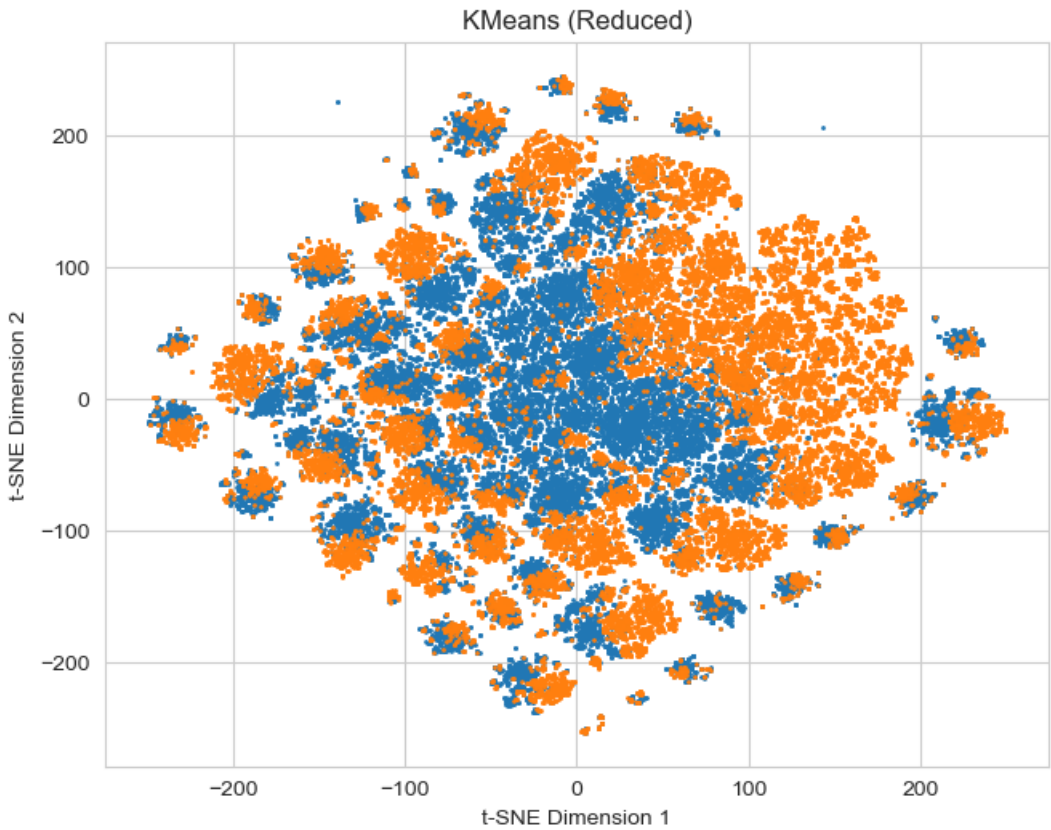


Figure 2.14: Cluster generated with $k$-means on reduced data, with $k$=2.

## iii) DBSCAN

A density-based algorithm, DBSCAN (Density-based Spatial Clustering of Applications with Noise) has the distinct advantage over K-Means in that it can identify clusters of any shape, while K-Means assumes that clusters are convex [41]. This is likely to be useful in the large, spread dataset that it will be used on. However, hyperparameter tuning is still required to produce optimal performance.

The algorithm operates by identifying core points – that is datapoints that meet a minimum neighbour criterion within a set radius around them. Both these parameters, the number of neighbours and the radius in which they must be found (*MinPts* and *eps* respectively), are set before the algorithm is run. In order to identify their ideal values, a process similar to the elbow method above is used. The nearest $n$ neighbours are found for each point and the average distance to them calculated. This is completed for all points in the dataset, and they are sorted in ascending order before plotting. Different values of $n$ can used, with a line produced for each value. The idea is, to find the best combination of *MinPts* and *eps*, an elbow point is found which will isolate noise and produce the best quality clusters. However, in this case only one value for $n$ will be tested that is twice the number of dimensions as is the recommended value for

*MinPts* [45]. As there are 116 dimensions in the dataset after preprocessing, the plots are generated using $n = 232$.

This method is completed twice, using both Euclidean and Manhattan distance for the same reasoning around dimensionality as explained in the k-means section. The results are shown in Figure 2.15 and Figure 2.16. In both plots, the elbow points are relatively clear, but are identified precisely using the kneed Python package. The determined values are 12.1 using the Euclidean distance, and 39.7 using the Manhattan. These are added to the plots as black, dashed horizontal lines.

Then, the algorithm is executed on the dataset and clusters generated. The achieved silhouette scores are 0.226 with Euclidean distance (calculated for the clusters obtained using Euclidean in the clustering) and 0.359 with Manhattan distance (calculated for the clusters obtained using Manhattan distance). These clusters are mapped onto the t-SNE representation of the data in Figure 2.17 and Figure 2.18, where it can be seen that the algorithm has only identified two clusters in the case of the Euclidean distance, and hasn't split up the data at all in the case of the Manhattan distance, bar isolating noise points. These plots make it clear that the silhouette score must be reevaluated in the case of the DBSCAN algorithm, as the Sci-kit Learn function counts the noise points as a cluster by default. By excluding these points we obtain final silhouette scores of 0.249 for the Euclidean distance and for the Manhattan distance the score becomes invalid as the data is not segmented at all.

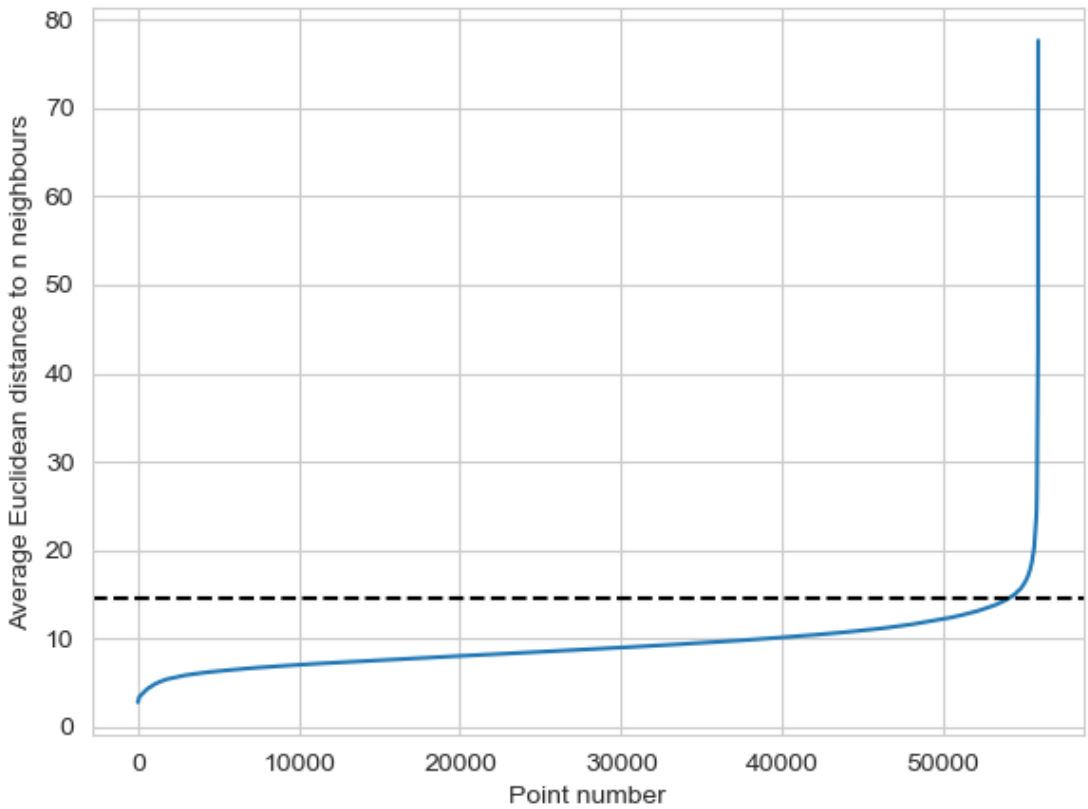


Figure 2.15: Heuristic method to find the optimal value of *eps* to use in the DBSCAN clustering algorithm. Black dotted line corresponds to the chosen value of *eps*, 12.1.

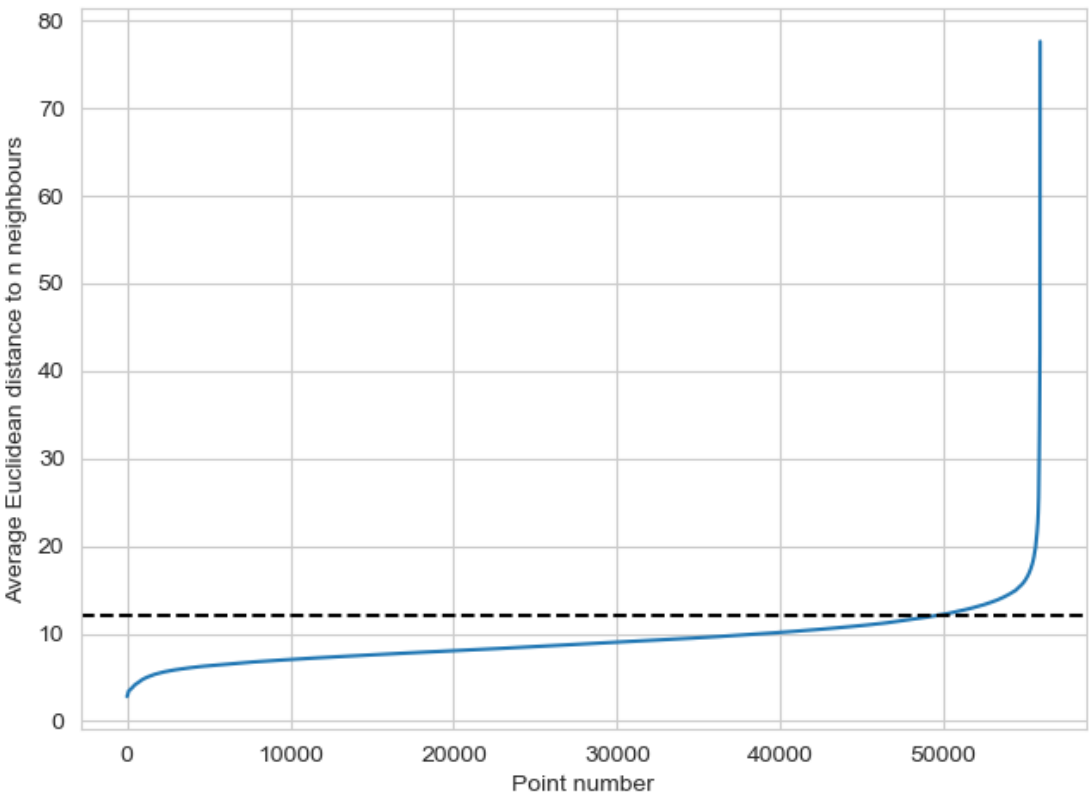


Figure 2.16: Repeat of the previous figure, but using Manhattan distance instead of Euclidean. Black dotted line corresponds to the chosen value of *eps*, 39.7.

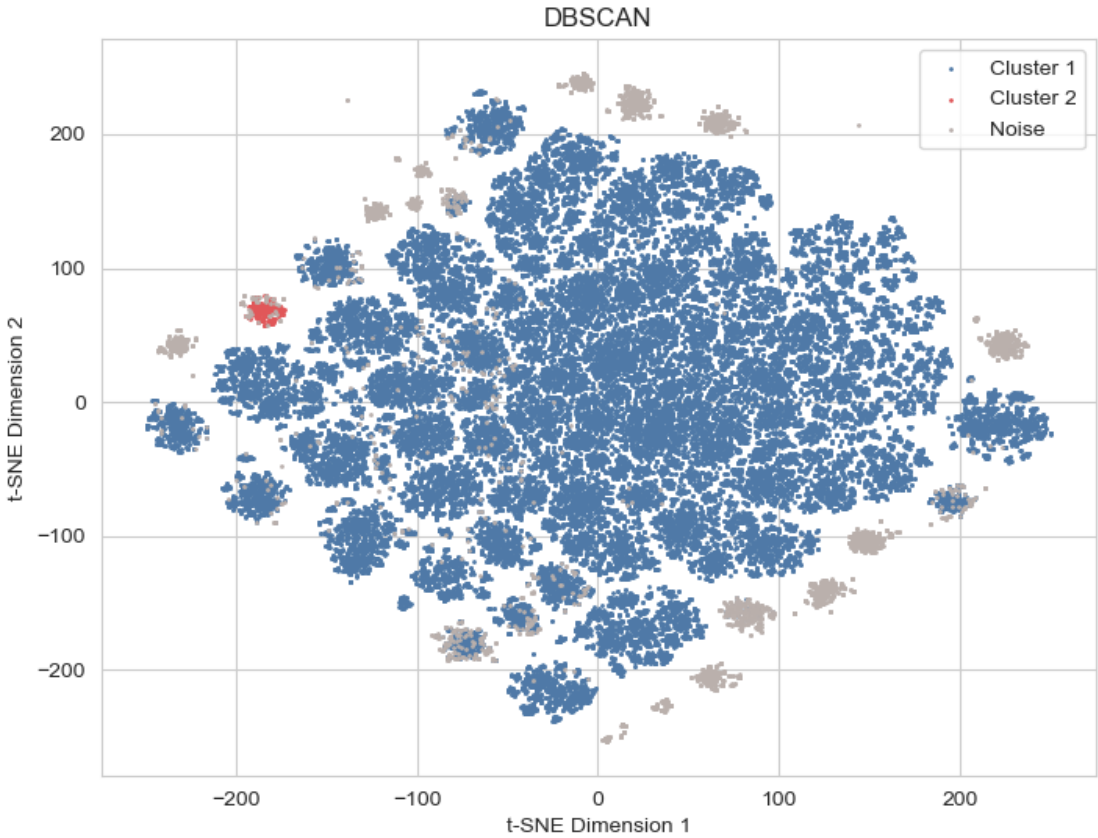


Figure 2.17: Clusters generated using DBSCAN with the Euclidean metric projected onto the t-SNE plot.

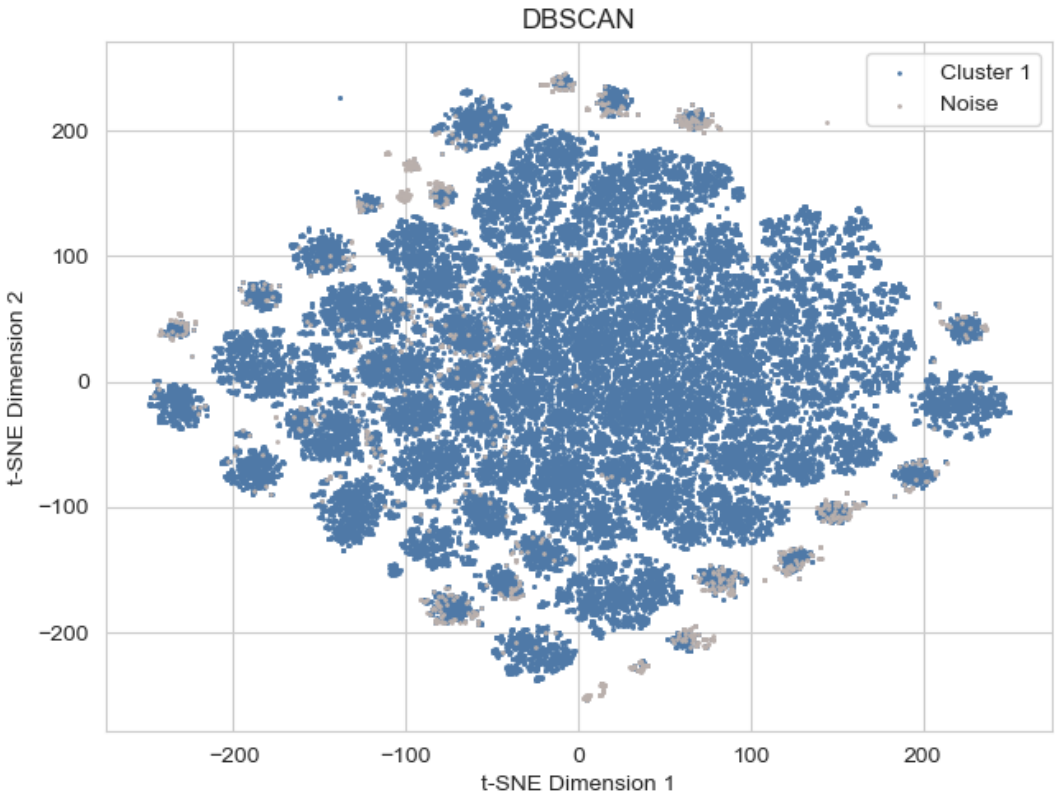


Figure 2.18: Clusters generated using DBSCAN with the Manhattan metric projected onto the t-SNE plot.

## iv) Reduced Data – DBSCAN

The same process is completed on the reduced data as it was for k-means. The value of $n$ is set to double the number of dimensions as before, which in this case this is 16. The same method is used to determine the *eps* value of 1.86, seen in Figure 2.19. As in the case of k-means on the reduced data, the distance measure is limited to the Euclidean distance. The final silhouette score obtained through this method after the noise points are discarded is 0.125. It appears that the reduction in dimensionality has harmed the performance of this clustering method, most likely because of the loss of variation. The clusters produced can be seen in Figure 2.20, and this time around many more of the points have been discarded as noise.

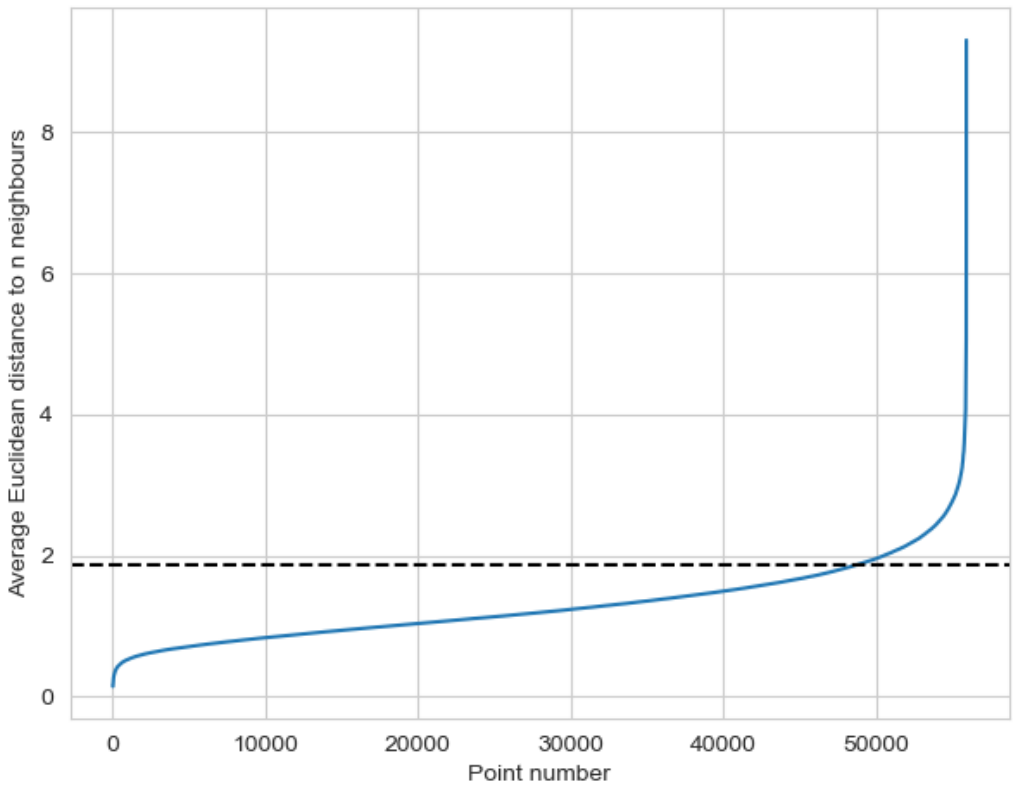


Figure 2.19: Heuristic method first used in 2.3iii) applied to the reduced dimension data. Black dotted line marks the chosen *eps* value of 1.86.

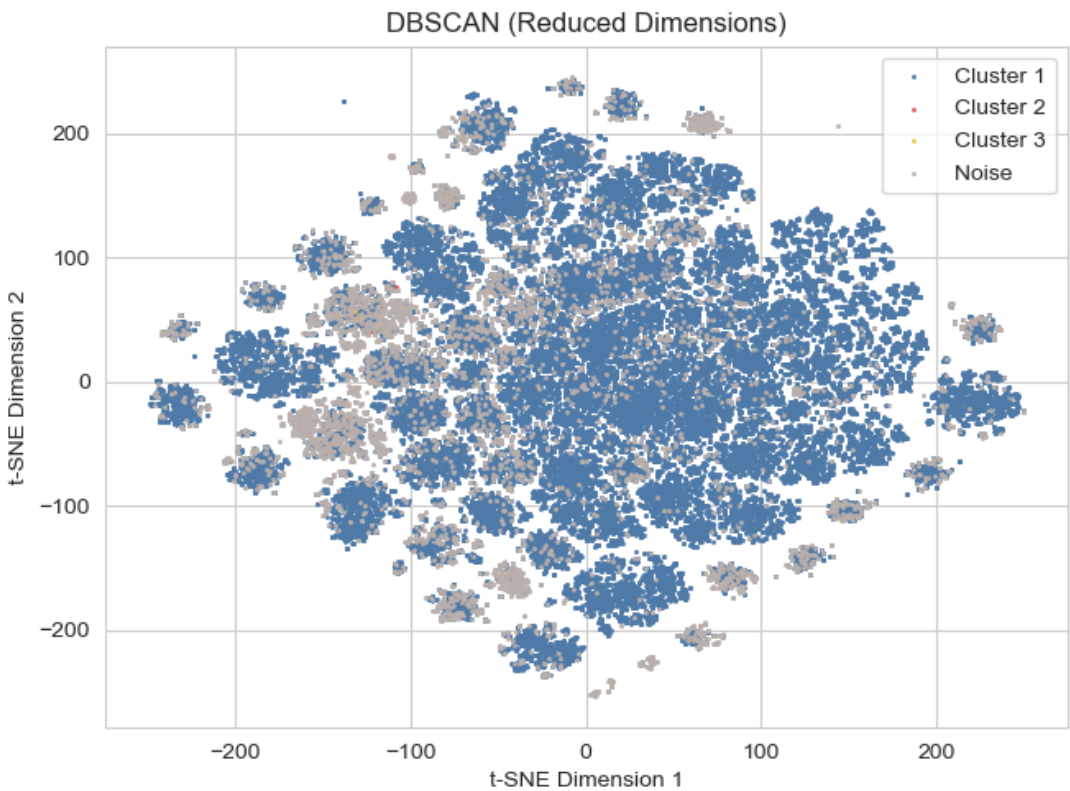


Figure 2.20: Clusters produced by DBSCAN on reduced dataset mapped onto the t-SNE plot.

## v) GMM

Gaussian Mixture Models (GMMs) are a different alternative to $k$-means clustering and their use offer its own set of advantages. The algorithm models the data by creating a set of Gaussian distributions, each comprised of a mean vector $\boldsymbol{\mu}$, a covariance matrix $\boldsymbol{\Sigma}$, and a probability factor $\boldsymbol{\pi}$ which defines the proportion of the sample represented by that particular distribution. These distributions are determined through the iterative Expectation-Maximisation algorithm, which aims to maximise the log-likelihood function [46].

This approach has two main differences over $k$-means. First of all, as $k$-means relies on the distance from a point to a centroid to determine its cluster membership, there is an assumption being made that the groups will be of a spherical shape. GMMs allow clusters to be elliptical in shape by finding the covariance matrix of each cluster. This should be more flexible when applied to real-world data.

The second difference is that GMM is a soft clustering algorithm. This means that its output is not solely the assigned cluster as is the case for $k$-means, a hard clustering algorithm, but a series of probabilities for each cluster. Obviously, these can become

hard cluster assignments by taking the maximum probability for each point (as is the case here), but the soft clustering can be useful to identify points that are in between two clusters, if desired.

Just as for $k$-means, the number of clusters is set before running the algorithm. So, as before, the silhouette score can be used in an attempt to find the optimum number of clusters, and is seen in Figure 2.21. However, just like in the plots for k-means, the 'optimum' number of clusters is the minimum of 2. As has been explained previously, any metrics relying on distance measures are likely to be very unreliable due to the high dimensionality of the data. Thus, another method to optimise the number of components is tested, using the average log-likelihood obtained for different values of $n_components$. For each value, the algorithm is fitted on a training set and the log-likelihood found for a separate validation set, which can be seen in Figure 2.22. It is clear that in general performance drops off as $n_components$ increases. A narrow window of the plot is presented in Figure 2.23, where it can be seen that the maximum log-likelihood of the validation data was achieved with $n_components = 23$.

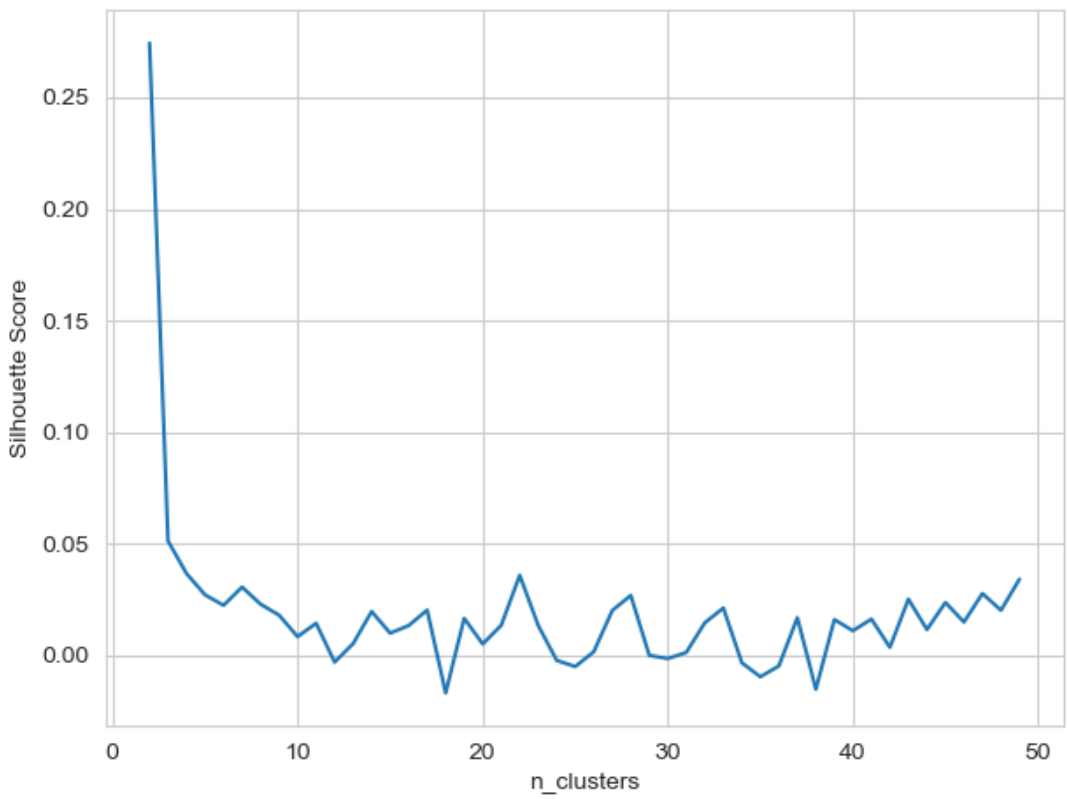


Figure 2.21: Average silhouette score obtained with GMM/EM using different numbers of components.

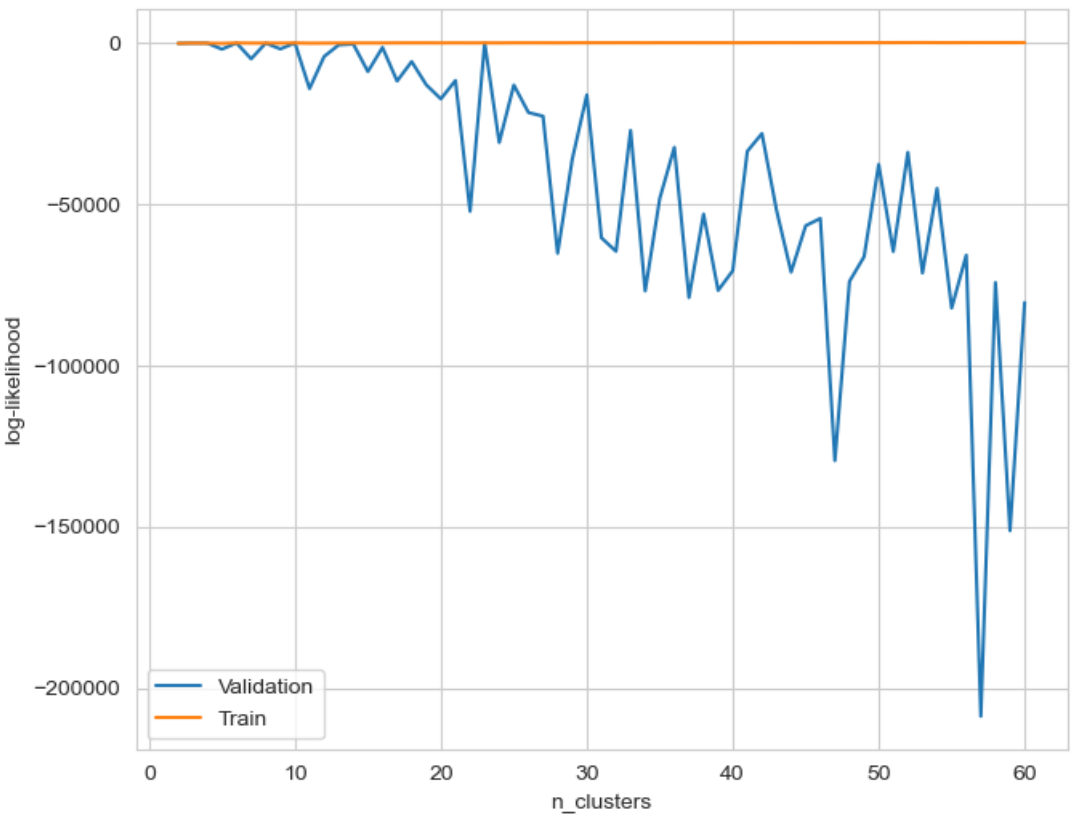


Figure 2.22: Average log-likelihood obtained through GMM both through the fitting and on an unseen part of the data, averaged over 3 runs.

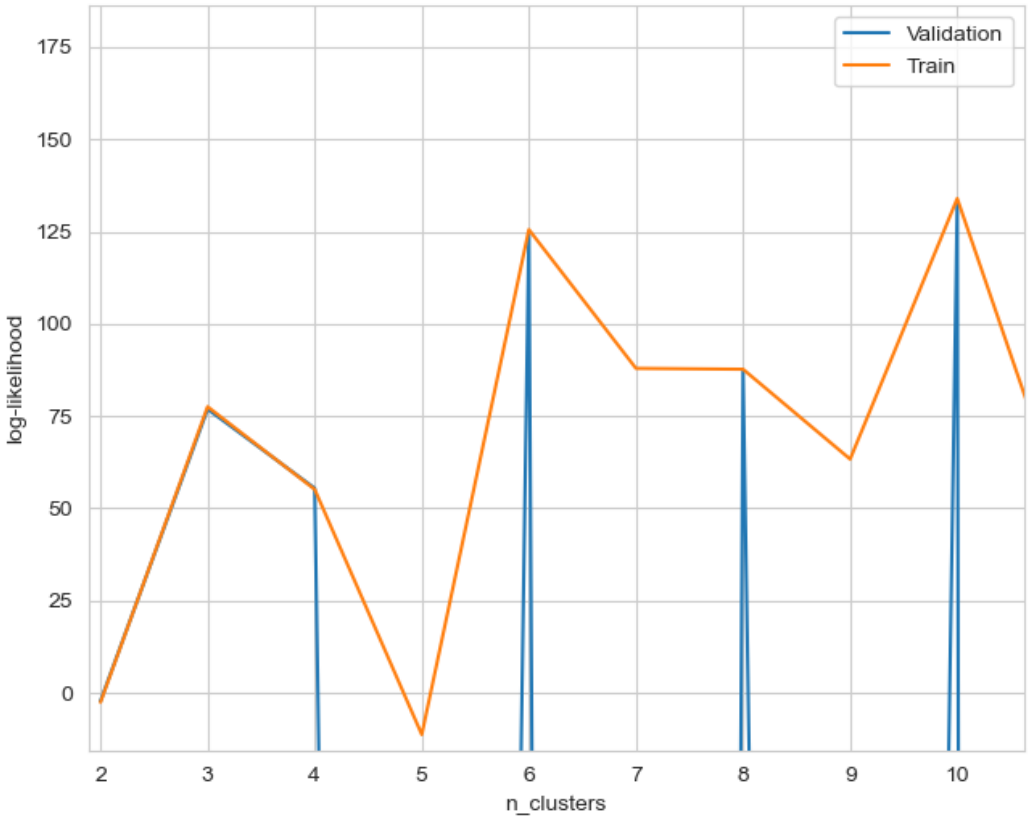


Figure 2.23: Close-up window of the log-likelihood plot shown in Figure 2.22.

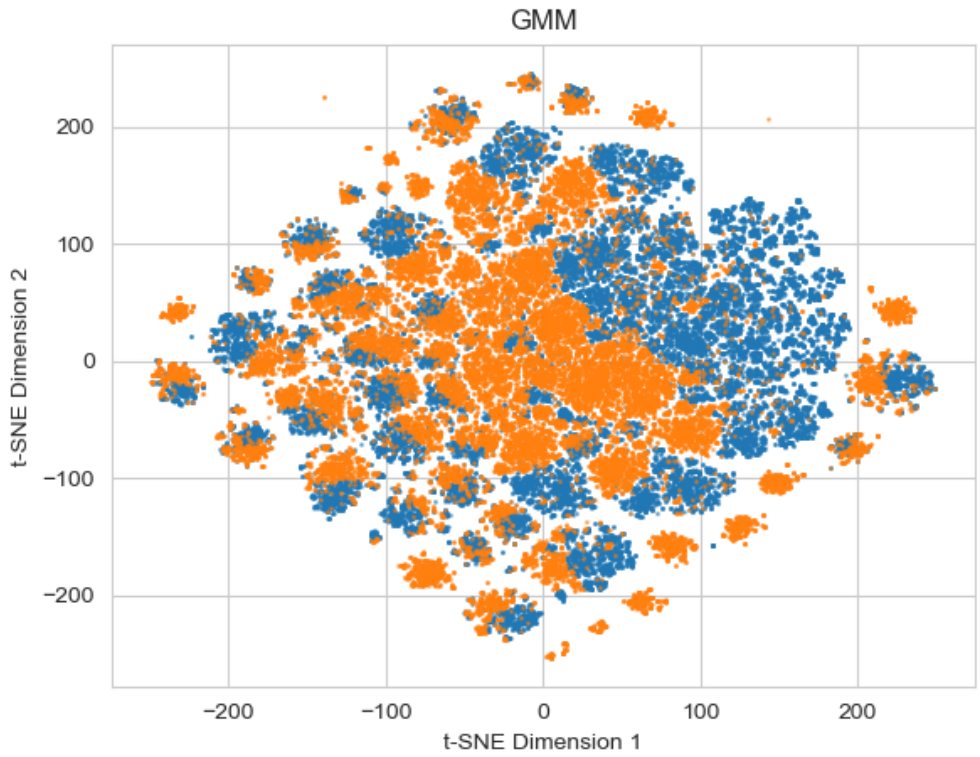


Figure 2.24: Determined clusters using GMM/EM with the number of components set to 2.

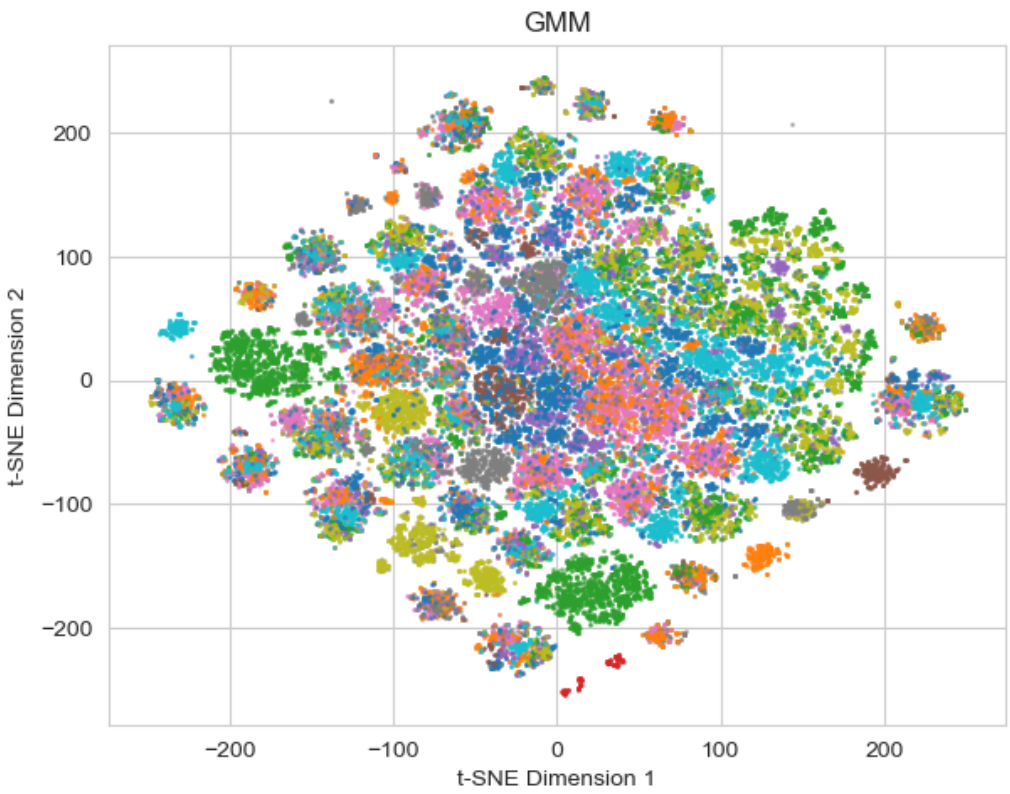


Figure 2.25: Determined clusters using GMM/EM with the number of components set to 23.

The clusters that are generated with $n_components = 2$, and $n_components = 23$ are shown in Figure 2.24 and Figure 2.25. The former presents clusters that look very similar to those found with k-means in section 2.3i) , with a lot of the clusters in the visualisation split between those determined by GMM. In the latter plot, whilst there are a lot more clusters, we still see the same problem. However, agreement between the determined clusters and the plot is not essential as it is not a ground truth. The silhouette scores are calculated as 0.05 and 0.14 with $n_components = 2$, and 0.01 and 0.05 with $n_components = 23$ for Euclidean and Manhattan distances respectively. It seems thus that once again, performance has been far from satisfactory.

## vi) Shared Nearest Neighbours (SNN)

As clustering with traditional distance measures has proven problematic in high dimensions, a method based upon the shared nearest neighbours among the dataset is tested [47, 48]. There are various algorithms that can use such a measure to perform clustering, but the one investigated in this case is based upon DBSCAN. An adaptation of the Sci-kit Learn method has been previously written and made publicly available online [49]. Utilising shared nearest neighbours as a clustering metric instead of distance has been found to have two main advantages that are relevant to the dataset used in this project. The first is that it often achieves better results in high dimensions, and the second is that it is better able to find clusters of varying densities [48].

The algorithm operates through first computing a similarity matrix using a given distance metric, sparsifying said matrix by keeping only the $k$ strongest links in order to create a shared nearest neighbour graph, and then defining the 'SNN density' of each point by counting the number of points that have a shared *eps* number of neighbours or greater with it. From there, the algorithm becomes just like DBSCAN as it identifies core points according to a threshold on the SNN density, *MinPts*, and so on [48].

As with the other methods, the difficult part is optimising the parameters to initialise it with. There is an example use of the algorithm in the original paper [48] which uses values of $k = 50$, $eps = 20$ and $MinPts = 34$. Whilst this is a smaller dataset of only 10,000 points and 2 dimensions, it's a starting point to test from. The result of using these parameters is seen in Figure 2.26. The method has identified 73 distinct clusters, with 30604 points assigned as noise, which are plotted in dark blue, and removed for closer inspection of the assigned clusters in Figure 2.27. It's important to note that as there are so many clusters there are repeating colours in the plot.

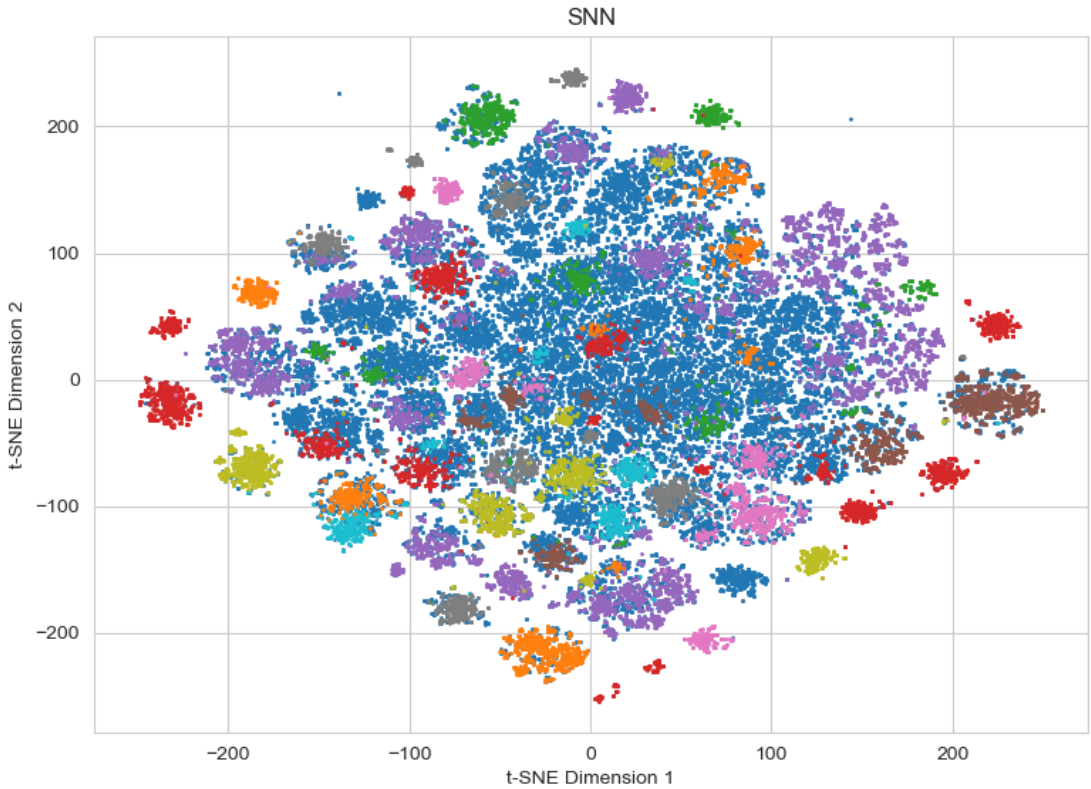


Figure 2.26: SNN generated clustering with k=50, eps=20, MinPts = 34.

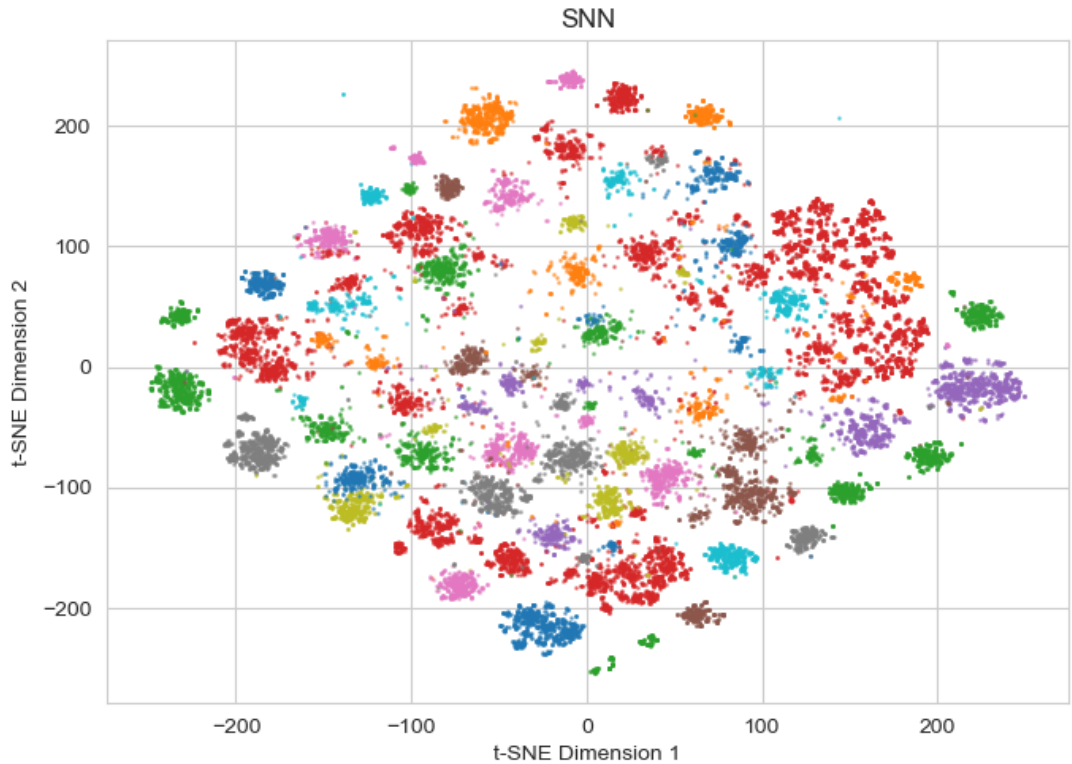


Figure 2.27: Figure 2.26 with the noise values removed.

It's interesting that the majority of globular clusters found around the edge with t-SNE are also almost all identified as their own clusters using SNN. As we start to approach the centre of the graph though, things look a bit more cluttered. However it should always be remembered that this visualisation cannot fully represent the nature of the data given its very high dimensionality. Thus it is hard to evaluate if the clusters have been identified correctly or not. The silhouette scores are calculated as -0.01 using Euclidean distance and -0.1 using Manhattan, after the noise points have been removed as was the case for DBSCAN. Judging by this alone, it would seem that this algorithm has performed just as poorly, if not worse, than those tested previously. However, as a significant number of clusters have been identified in this case, with many agreeing with the visualisation, it would appear that there is much more promise using this algorithm.

What is clear, however, is that there is an undesirable number of points discarded as noise using the original parameters. In order to reduce this, we can change *eps*, the number of shared nearest neighbours required for two points to be classed as similar. The higher this value, the higher the threshold for similarity, and this results in fewer points assigned to clusters, as seen in Figure 2.28. So by reducing it, less points will be identified as noise, but the clusters will have a lower intracluster similarity, as is seen in Figure 2.29 (in both cases the noise points are displayed as dark blue). Finding the balance between these is a difficult task, especially when the silhouette evaluation metric appears to be useless with the high dimensionality.

In an attempt to determine how the parameters can be optimised, the variation in the number of noise points excluded against the number of clusters generated is plotted, and shown in Figure 2.30. Whilst the number of clusters is not indicative of their quality, there is no clear way of measuring said quality, and this alternative still provides some useful insight into how changing the parameters affects the final result. In the plot, each colour represents the changing variable, with the other two held at their default values.

It can be seen that changing the value of either $k$ or *eps* has a large effect on both the number of clusters identified and the noise points excluded, with a low number of noise points obtained either by a higher $k$ or a lower *eps*. On the other hand, changing *MinPts* between 10 and 40 doesn't affect the number of clusters identified hugely, but the lower the value, the lower the number of noise points excluded.

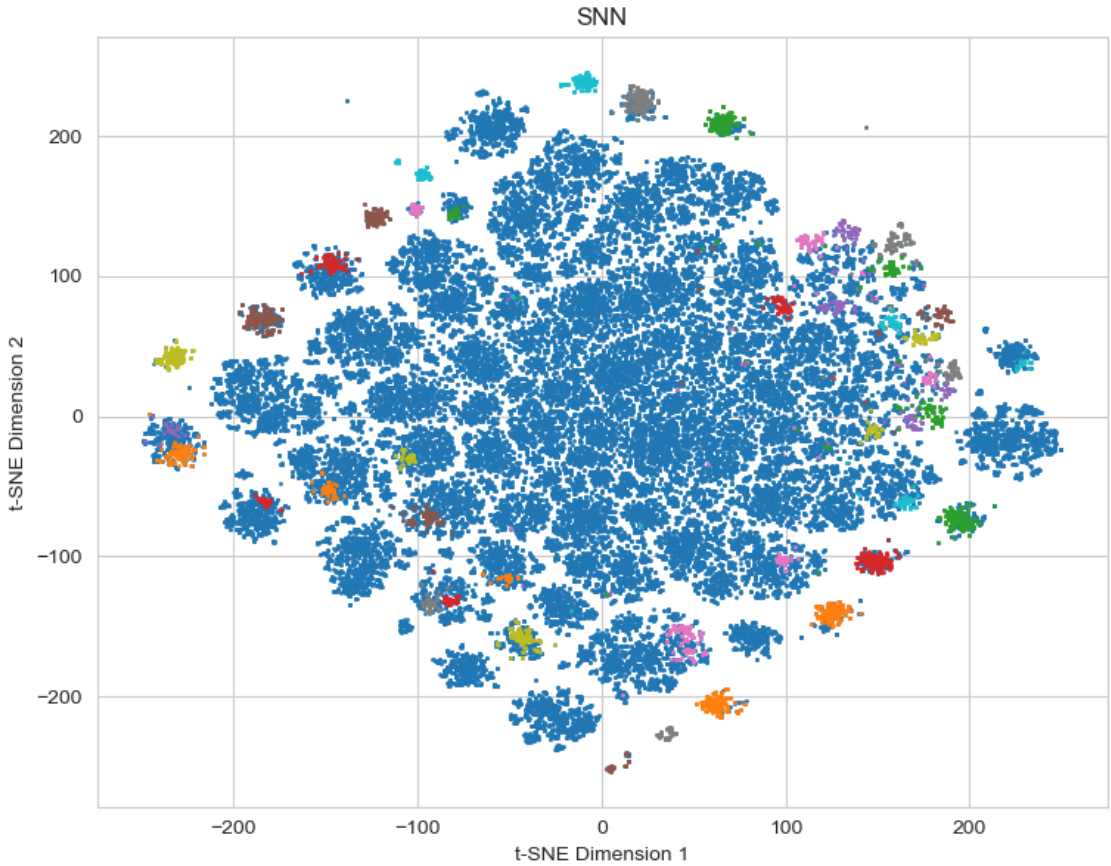


Figure 2.28: SNN generated clustering with k=50, eps = 30, MinPts = 34

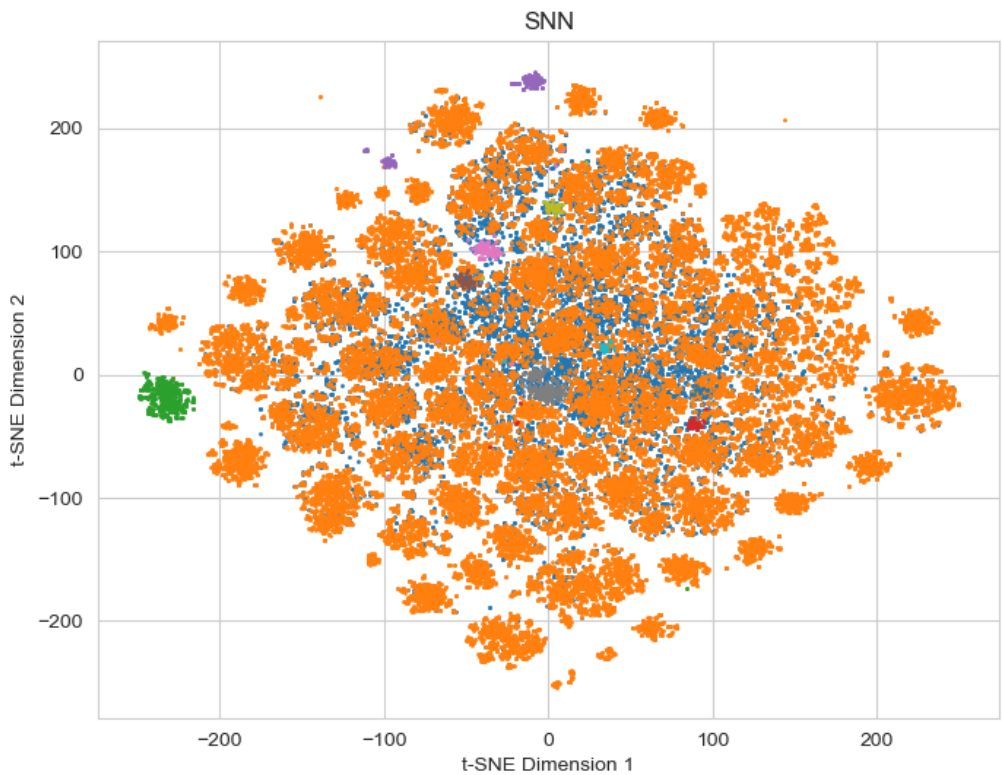


Figure 2.29: SNN generated clustering with k = 50, eps = 15, MinPts = 34

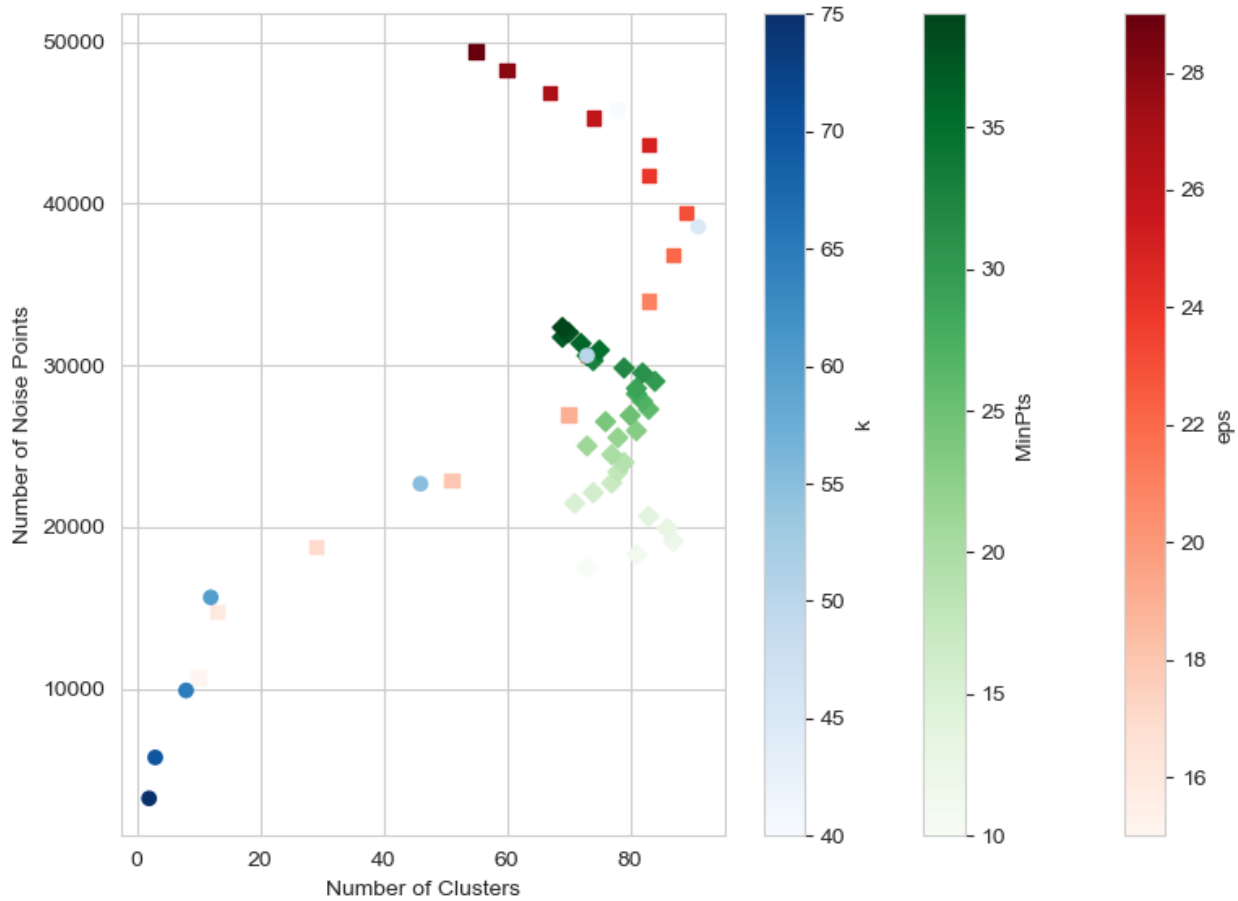


Figure 2.30: The change in noise points excluded against the number of determined clusters as $k$, $eps$ and $MinPoints$ vary. Each of the hyperparameters is changed as the other two are kept still as the default of $k = 50$, $eps = 20$ and $MinPts = 34$.

It seems natural then to attempt clustering again with a lower value of $MinPts$. Through testing a range of combinations and inspecting their plots to find the best combination of low noise and distinct clusters, parameters are chosen of $k = 54$, $eps = 21$ and $MinPts = 11$, with the clusters generated shown in Figure 2.31. These total to 53 individual clusters and 15988 points assigned as noise. It's clear that there is one dominant cluster in the set, and this is indicated by the member counts of the top 10 clusters in Table 2.8. A lot of the globular clusters on the t-SNE plot have been encompassed by this cluster 1, which is the price paid for reducing the number of noise points. This plot has silhouette scores of -0.168 and -0.275 using Euclidean and Manhattan distances respectively.

Whether or not this is better than using the original parameters would depend on the use of the clustering results in the final application. It might be preferable to use the latter results as there are a higher number of profiles classified into clusters, or it might be more useful to identify more distinct clusters as in Figure 2.26, which could enable a user's preferences of potential matches to be better tracked, as when they indicate an interest in another user from a very distinct cluster, it's easy to show them others that are sure to be similar.

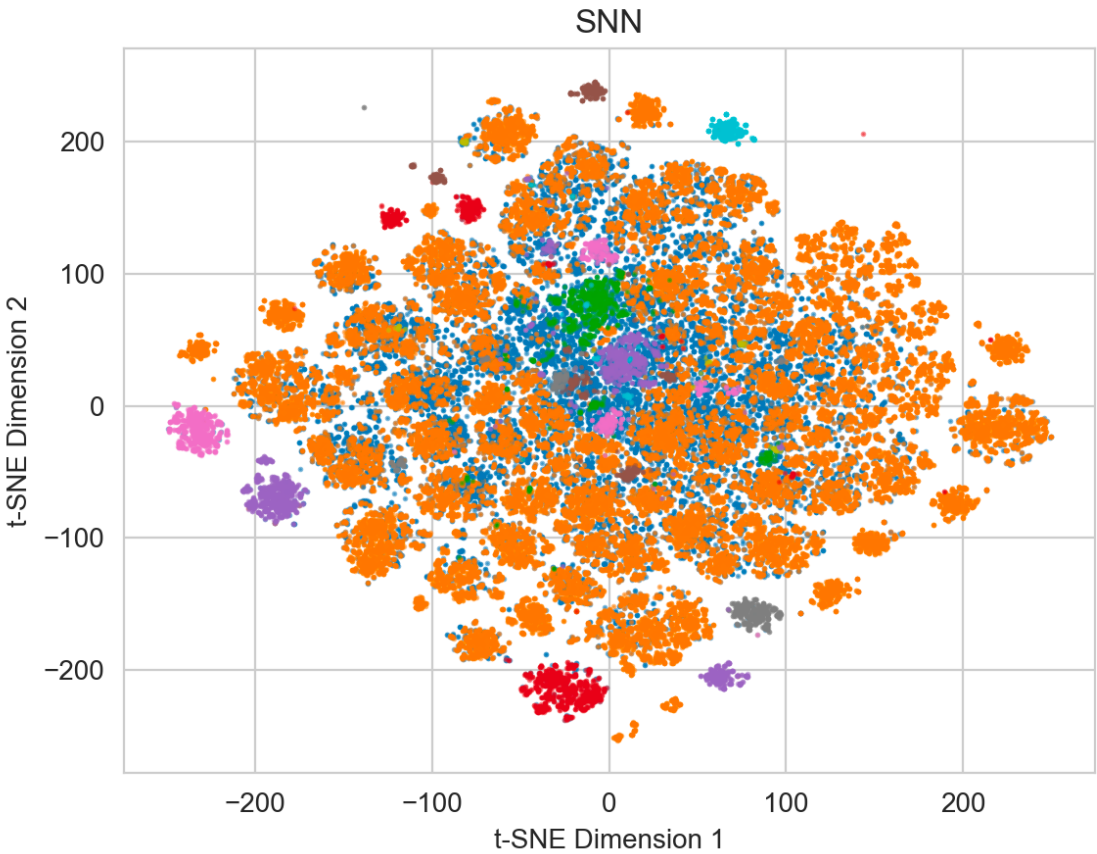


Figure 2.31: SNN generated clustering with k=54, eps = 21, MinPts = 11.

| Cluster | Number of Members |
|---|---|
| 1 | 33391 |
| Noise | 15988 |
| 2 | 867 |
| 3 | 719 |
| 4 | 688 |
| 5 | 616 |
| 6 | 602 |
| 7 | 404 |
| 8 | 279 |
| 9 | 268 |
| 10 | 251 |

Table 2.8: Cluster membership of Figure 8.6 for the top 10 most populous clusters.

## 2.4 Cluster Interpretation

The segmentation of a dataset is rarely useful without an understanding of the relationships between points that have caused them to be grouped together. In order to interpret the results, an external function is utilised, obtained from [50], which derives rules for the clusters utilising Sci-kit Learn's function for decision tree classifiers. The results are displayed for a sample of the clusters derived with SNN, with the default hyperparameters. The function does not produce the support of each rule or the confidence which are the typical evaluation metrics in association rule mining. Instead the metric is the percentage of datapoints following the given rule that are located in that cluster. For example, the first row of Table 2.9 shows that of all the user profiles that have *job_nan* less than or equal to 0.5 and *job_law / legal services* greater than 0.5, around 91% are in cluster 12.

Table 2.9 presents rules for a sample of the clusters, where it's clear that the main distinguishing factor between these clusters is the values of the *job* features. Whilst this is only a sample, almost all the rules generated include the *job* features. This makes sense as from the exploration in 2.1ii) it is known that the userbase is well spread in this area, and it was one-hot encoded. Whether this is desirable in the clusters depends very much upon if users usually have preferences about a possible match's job or not.

| Cluster | Number of Members | Rule Metric | Rule |
|---|---|---|---|
| 12 | 846 | 0.906 | job_nan <= 0.5<br>job_law / legal services > 0.5 |
| 18 | 753 | 0.941 | job_nan <= 0.5<br>job_law / legal services <= 0.5<br>job_construction / craftsmanship > 0.5 |
| 24 | 645 | 0.928 | job_nan <= 0.5<br>job_law / legal services <= 0.5<br>job_construction /craftsmanship <= 0.5<br>job_clerical / administrative > 0.5 |
| 10 | 593 | 0.941 | job_nan <= 0.5<br>job_law / legal services <= 0.5<br>job_construction / craftsmanship <= 0.5<br>job_clerical / administrative <= 0.5<br>job_banking / financial / real estate <= 0.5<br>job_political / government > 0.5 |
| 7 | 500 | 0.931 | job_nan <= 0.5<br>job_law / legal services <= 0.5<br>job_construction / craftsmanship <= 0.5<br>job_clerical / administrative <= 0.5<br>job_banking / financial / real estate <= 0.5<br>job_political / government <= 0.5<br>job_executive / management <= 0.5<br>body_type_full figured <= 0.5<br>job_hospitality / travel <= 0.5<br>job_transportation <= 0.5<br>ethnicity_indian <= 0.5<br>russian > 2.5 |
| 0 | 341 | 0.985 | job_nan <= 0.5<br>job_law / legal services <= 0.5<br>job_construction / craftsmanship <= 0.5<br>job_clerical / administrative <= 0.5<br>job_banking / financial / real estate <= 0.5<br>job_political / government <= 0.5<br>job_executive / management <= 0.5<br>body_type_full figured <= 0.5<br>job_hospitality / travel <= 0.5<br>job_transportation > 0.5 |

Table 2.9: 'Rules' generated for a small sample of the clusters generated with SNN.

# Chapter 3 – Text Data

## 3.1 – Data Understanding

### i) Data Quality Assessment

In addition to the features listed in 2.1i) , each profile contains the user's responses to 10 prompts intended to make them reveal more about themselves. These prompts are presented in Table 3.1.

| **Feature Name** | **Response to:** | **Null Entries** |
|---|---|---|
| *essay0* | My self-summary | 5488 |
| *essay1* | What I'm doing with my life | 7572 |
| *essay2* | I'm really good at | 9638 |
| *essay3* | The first thing people usually notice about me | 11476 |
| *essay4* | Favorite books, movies, show, music, and food | 10537 |
| *essay5* | The six things I could never do without | 10850 |
| *essay6* | I spend a lot of time thinking about | 13771 |
| *essay7* | On a typical Friday night I am | 12451 |
| *essay8* | The most private thing I am willing to admit | 19225 |
| *essay9* | You should message me if... | 12603 |

Table 3.1: Text features of the dating profiles.

The number of null values is significant for all features, with *essay8* unsurprisingly, given the nature of the prompt, having the most by far. As discussed in 2.1i) , this incompleteness is to be expected for user-inputted data. However, imputation is not possible for raw text, so records will now have to be excluded. To keep things simple, the investigation will move forward with profiles only where all essay features have an input.

The previous discussion around the data's timeliness and sampling bias remains relevant to the text features.

## ii) Data Preprocessing (for EDA)

Unlike for the exploratory data analysis on the structured features, significant preprocessing must take place before the analysis in order for the data to be interpretable. Firstly, records are removed where the language is not English using the fasttext Python library. Following that, a pipeline of the spaCy natural language processing library is used on each feature to handle many of the traditional preprocessing steps of NLP. The steps taken are the following [51]:

1. Tokenisation, splitting up the text into its individual words or punctuation (known as tokens). In spaCy, this is performed with a language model that is able to identify contractions like "I'm" and "can't" as separate words.
2. Part of speech extraction, the library is then able to predict for each token what its grammatical function is within a sentence.
3. Entity extraction, for each profile the library identifies notable entities within the essay. For example, different countries, cities or people.
4. Stop word removal, removing tokens that are common words of the language which contribute very little to the overall meaning of the text – e.g. 'that', 'are', 'and' etc.
5. Lemmatization, taking the remaining tokens after stop word removal and converting them into their base form. Examples of this are taking different forms of verbs and converting them to the infinitive form (e.g. 'rained' becomes 'rain) and taking comparatives and converting them to their base adjective (e.g. 'better' becomes 'good').
6. Identifying which records contain emojis.

After this, the lemmatized results are used to find collocations that provide insight into the content of each feature. Collocations are a sequence of words that occur together within a text more than would be expected by chance. In this case the top 10 bigrams and trigrams (collocations of 2 and 3 words respectively) are identified by both their raw frequency of occurrence and by the measure of pointwise mutual information (PMI).

PMI is dependent upon the probability of an n-gram occurring compared with the probabilities of its individual components occurring. The formula for a bigram's PMI is

$$PMI = \log\left(\frac{P(a,b)}{P(a)P(b)}\right), \tag{3.1}$$

which enables the 'most interesting' collocations to be found for each feature. It should be noted however that a filter is applied so that the top collocations by PMI must still meet a minimum frequency of 10. This process is completed with the NLTK library, as there is no collocation analysis feature in spaCy.

## iii) Exploratory Data Analysis

Using the processed data taken from the profile bios, a dashboard is constructed to examine how the contents of the text data changes between the different features, or essays. This is shown in Figure 3.1 for the first feature, 'My self-summary', but the whole dashboard is available online.

Looking at the different features reveals some expected information, such as the most common tokens in all of the different features being punctuation like commas and full stops, but also some surprising results, such as the top trigram by frequency for *essay7*, 'On a typical Friday night I am', being 'meow meow meow'.

In terms of the length of the answers, there are two prompts that produced much longer responses in general than the others – *essay0,* 'My self-summary' and *essay4*, 'Favorite books, movies, show, music, and food'. On the other hand the histogram of number of tokens for *essay8*, 'The most private thing I am willing to admit', is extremely right skewed with a peak at the 10-20 tokens bin.

The percentage of entities that are 'GPE' ('countries, cities and states') is much greater for *essay0* than for the rest of the features, which is logical as people are likely to mention their location when describing themselves. On the other hand, for *essay4*, the dominant entity category is 'PERSON', with some of the top examples being 'anderson', 'harry potter' and 'johnny cash'.

There's also a clear difference between the features in terms of their spread between the various parts of speech. Although in almost all of them the primary label is noun (with the top 3 most frequent examples being 'people', 'music' and 'friends), some features such as *essay3*, 'The first thing people usually notice about me' have a very high proportion of pronouns including 'I' 'my' and 'me', whilst others like *essay8*, 'The most private thing I am willing to admit' have a high proportion of verbs like 'be' 'want' and 'know'.

Table 3.2 shows a snapshot of the n-gram results with the top bigrams by frequency and pointwise mutual index for each prompt. The themes of the responses are obvious from the most frequent results, whilst the top results by PMI have 'harry potter' as

the top bigram on three occasions, which is logical given that the two individual components are not very common in text alone.

The final part of the EDA was to examine which profiles contained emojis, as these would be useful in differentiating the users if they were present. However, none of the users had any emojis in at all, possibly indicating they were removed before the dataset was made public, or that they weren't permitted as input into the application at the time of collection.

| **Prompt** | **Top Bigram (Frequency)** | **Most Interesting Bigram (PMI)** |
|---|---|---|
| My self-summary | san francisco | sri lanka |
| What I'm doing with my life | san francisco | carpe diem |
| I'm really good at | making people | harry potter |
| The first thing people usually notice about me | people notice | brad pitt |
| Favorite books, movies, show, music, and food | hip hop | pinkola estes |
| The six things I could never do without | family friends | harry potter |
| I spend a lot of time thinking about | time thinking | buenos aires |
| On a typical Friday night I am | hanging friends | meow meow |
| The most private thing I am willing to admit | open book | kelly clarkson |
| You should message me if... | sense humor | harry potter |

Table 3.2: Top bigrams for each feature.

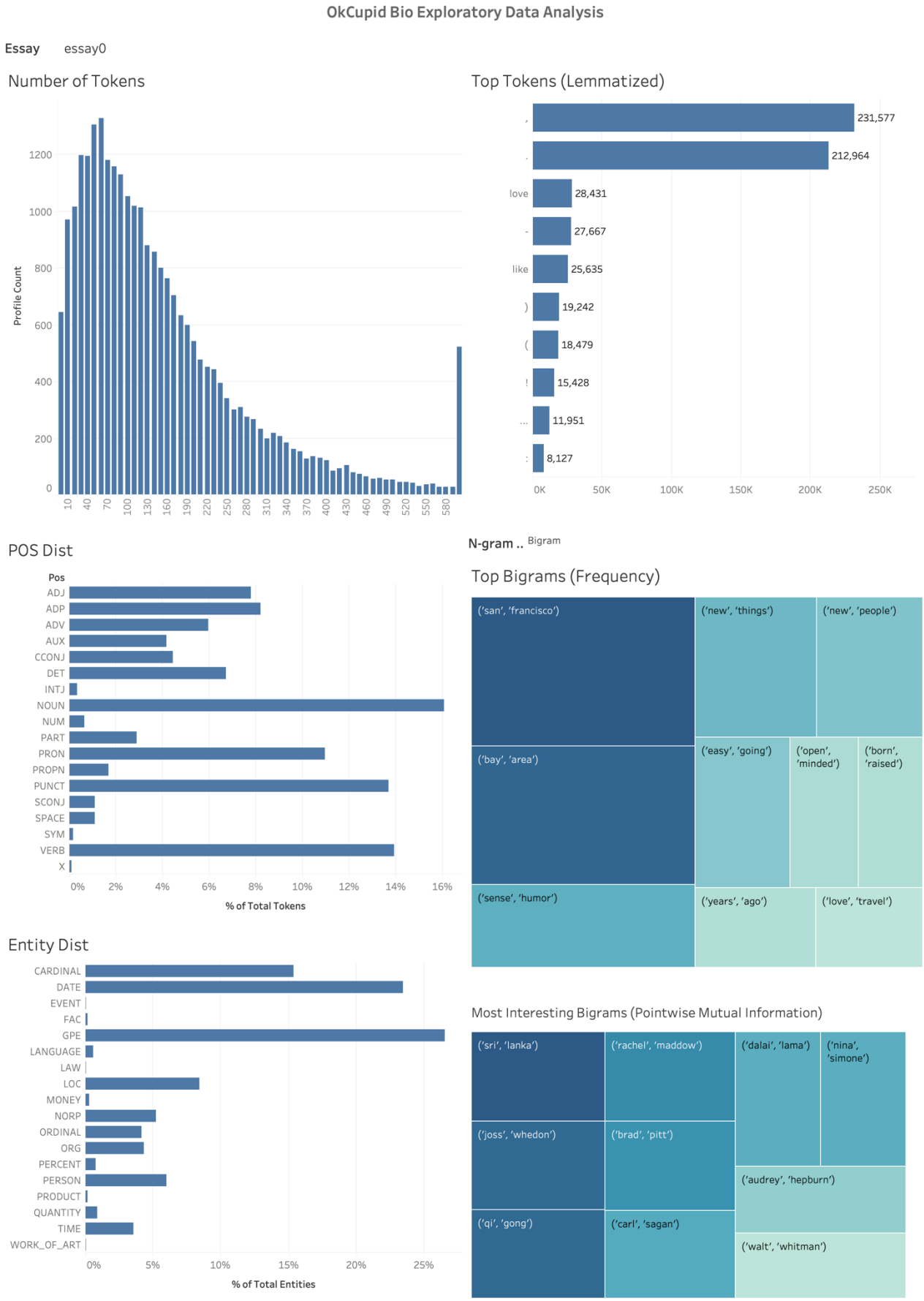


Figure 3.1: Dashboard of profile bio analysis, shown for feature *essay0*.

## 3.2 – Preprocessing

There is not much more preprocessing that needs to take place in addition to that already completed in order to make the text features suitable for machine learning models, with only a few extra steps executed at this stage.

The first of these is removing the punctuation. Obviously, there could be a way to utilise the differences in punctuation between profiles to cluster, but it is assumed in this case that it would have limited usefulness.

The second step is to create a document-word matrix, otherwise known as a bag-of-words representation of the features, undertaken using the CountVectorizer function of scikit-learn. The function allows for multiple arguments for filtering the data – so a minimum word length is set as 3 characters, and words that have a frequency of 50% of the maximum single word frequency or higher, or 1% of the maximum single word frequency or lower are not counted. The matrix is created for all of the features combined, i.e. each document is the concatenation of all the text responses for that profile.

## 3.3 - Modelling

### i) Latent Dirichlet Allocation

When entering into the world of clustering text, it's necessary to take into account the adjacent field of topic modelling, and one of its most popular methods: Latent Dirichlet Allocation (LDA). Described by its authors as "a generative probabilistic model for collections of discrete data such as text corpora" [52], when given a collection of documents it outputs different topics that are present in the corpus, as well as an allocation of these topics for each document.

The basic method can be summarised as follows [53]:

1. Every word in every document is randomly assigned as one of $k$ topics (the same word could be assigned to different topics in different instances)
2. Go through each document $d$ and in turn examine each word, $w$. Calculate two probabilities for every topic;

    - p(topic $t$ | document $d$), the proportion of words in document $d$ that belong to topic $t$, excluding the current word $w$

- p(word $w$ | topic $t$), proportion of words assigned topic $t$ are instances of word $w$, excluding current instance of $w$.

3. Update the probability of word $w$ belonging to topic $t$ using the multiplication of these two probabilities.
4. Assign word $w$ to the topic with the maximum new probability value, otherwise known as Gibbs sampling.
5. Repeat steps 2-4 for a set number of iterations, or until the assignment of topics among the words no longer changes.

The algorithm consists of three main hyperparameters that require tuning to withdraw the best performance, and create the most sensible topics. These are the document-word density factor, $\alpha$, the topic-word density factor, $\beta$, and the number of topics, $k$ [54]. In this instance, the computing power required to grid search all of 3 parameters would be excessive for this investigation, so $\alpha$ and $\beta$ are maintained as their default value in scikit-learn whilst the most important parameter, the number of topics, is varied.

In order to tune $k$ to the optimum value, models are generated with a range of values and the approximate log-likelihood output of each model is recorded, the same process as in [55]. The value of $k$ that produces the highest value of log-likelihood will be said to be 'optimum'. The results of this are displayed in Figure 3.2, where there's a clear peak reached at 6 topics, with a metric value of $-3.149 \times 10^7$. The top 10 keywords for each topic at this $k$ value are shown in Table 3.3. Although some of the topics do seem to have an intelligible theme through the words, with topic 4 leaning towards creativity for example, there are still some words that do not have a huge amount of significance, and words that repeat through multiple topics like 'lot'.

For this reason, the same process is completed again, but using only the tokens assigned as nouns by spaCy, as is often common practice to try and ensure the top words in a topic are meaningful. Results are found in Figure 3.3, with the top keywords for the peak of $k = 4$ in Table 3.4. But, given the fact that the peak is much less clear, and the keyword groups seem less coherent, these results appear worse than the previous set.

Another approach would be to topic model each feature individually and use the output of the document-topic distributions in a typical clustering model, like K-Means or DBSCAN. However, inspecting the same log-likelihood plots for each of the individual features reveals that the model seems to struggle with finding any distinct topics, as all plots are just a straight line decreasing in value from 2 onwards.

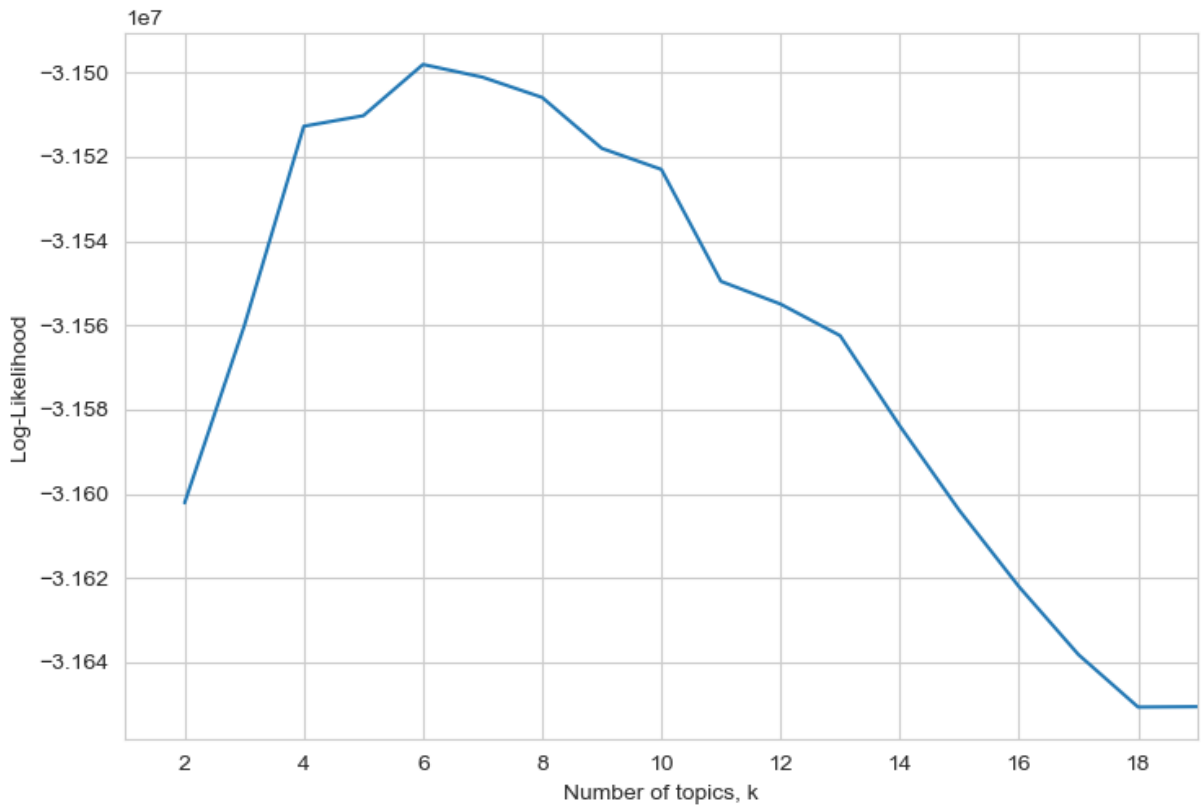


Figure 3.2: Log-likelihood of LDA model on the document-word matrix of all text features combined.

| **Topic 1** | **Topic 2** | **Topic 3** | **Topic 4** | **Topic 5** | **Topic 6** |
|---|---|---|---|---|---|
| read | great | year | dance | man | pretty |
| lot | open | travel | make | black | lot |
| way | find | city | eat | dead | guy |
| write | relationship | favorite | art | john | fun |
| find | world | san | cook | band | watch |
| world | person | fun | laugh | game | game |
| word | fun | lot | day | star | get |
| stuff | travel | bay | bike | david | read |
| probably | feel | watch | dog | show | favorite |
| art | share | great | dancing | big | meet |

Table 3.3: Top 10 keywords for each topic generated with LDA with $k = 6$.

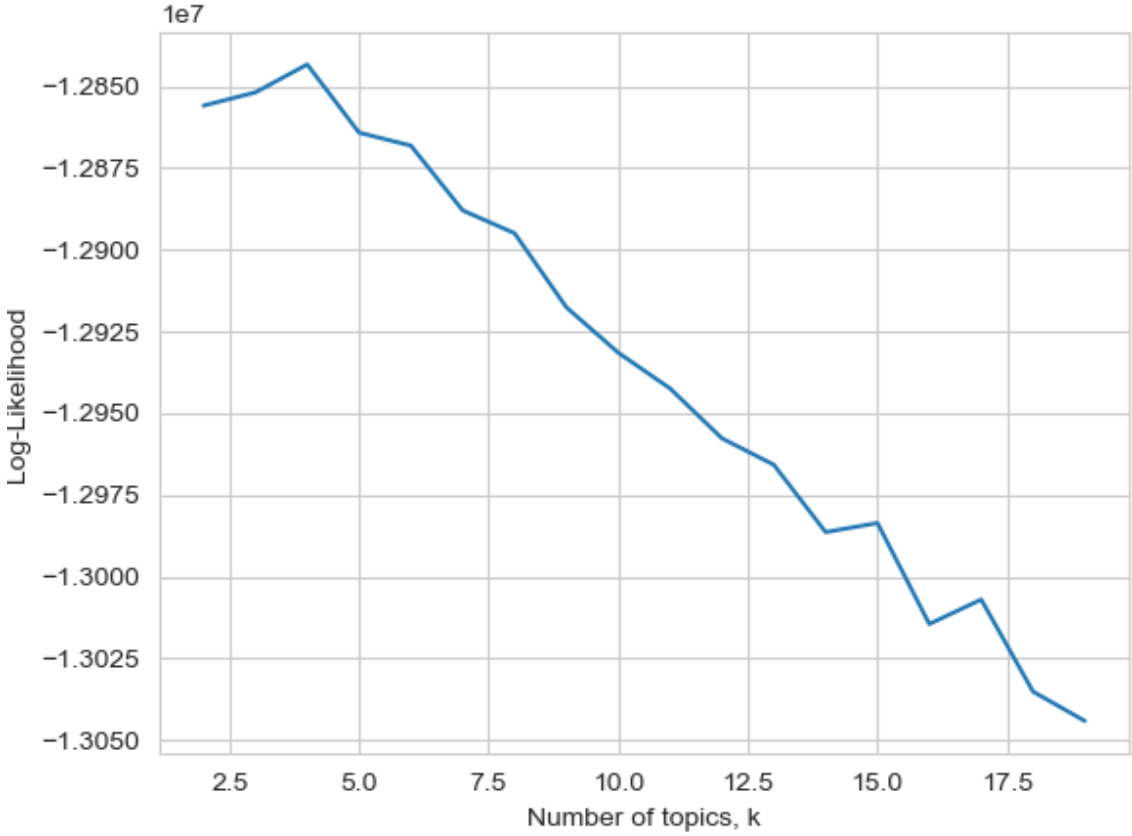


Figure 3.3: Log likelihood of LDA model on matrix of all features combined, but only nouns.

| **Topic 1** | **Topic 2** | **Topic 3** | **Topic 4** |
|---|---|---|---|
| year | guy | lot | love |
| city | fun | man | world |
| lot | school | game | way |
| day | person | art | art |
| work | lot | day | relationship |
| place | girl | stuff | year |
| wine | year | way | lot |
| game | day | world | person |
| fun | love | word | work |
| night | sport | girl | woman |

Table 3.4: Top 10 keywords for each topic generated with LDA with $k = 4$ on only nouns.

Nevertheless, this approach can still be used on the first set of distributions for all the features combined, and is first undertaken using K-means.

Given that there are only 6 dimensions, the issues encountered when clustering our structured data should not be a problem. The same method of choosing the optimal value of $k$ is utilised, resulting in the plots of Figure 3.4 and Figure 3.5, which both unsurprisingly agree that the optimal number of clusters is 6 – equal to the number of topics identified by LDA. A final silhouette score of 0.267 is not indicative of clearly separated clusters though.

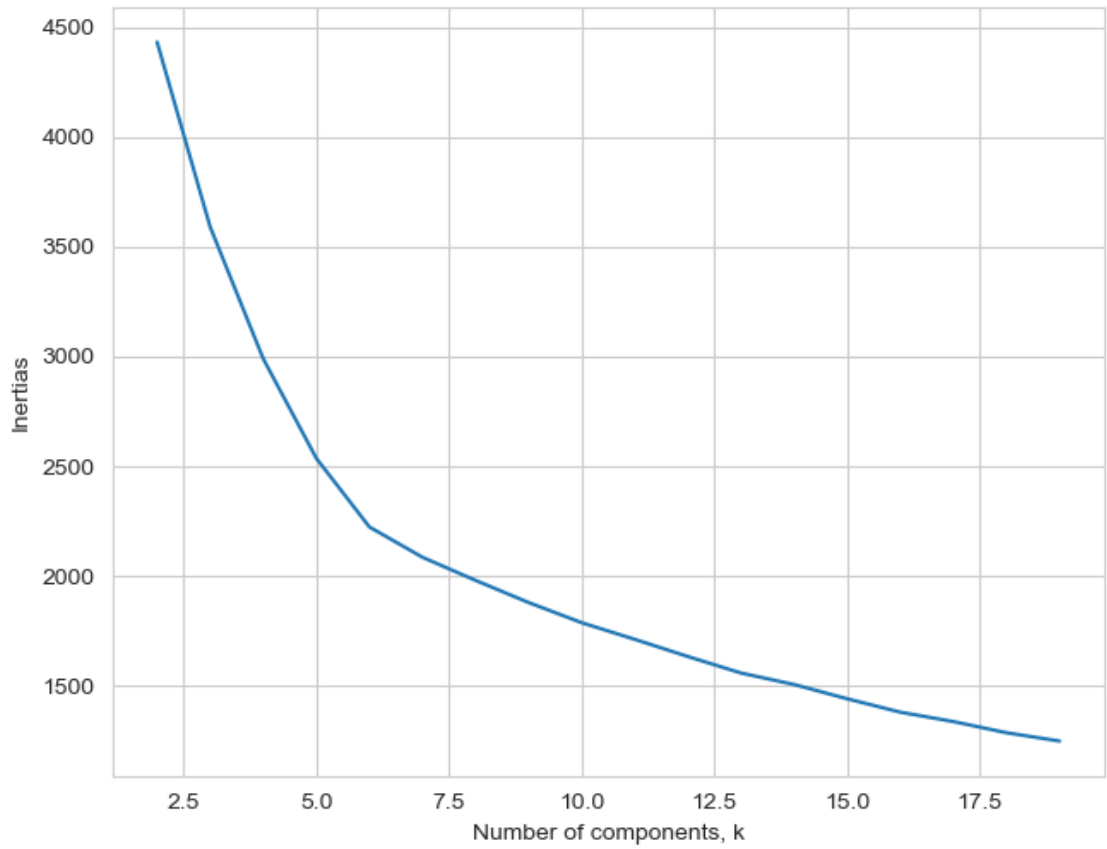


Figure 3.4: Elbow plot of k-means algorithm on LDA output.

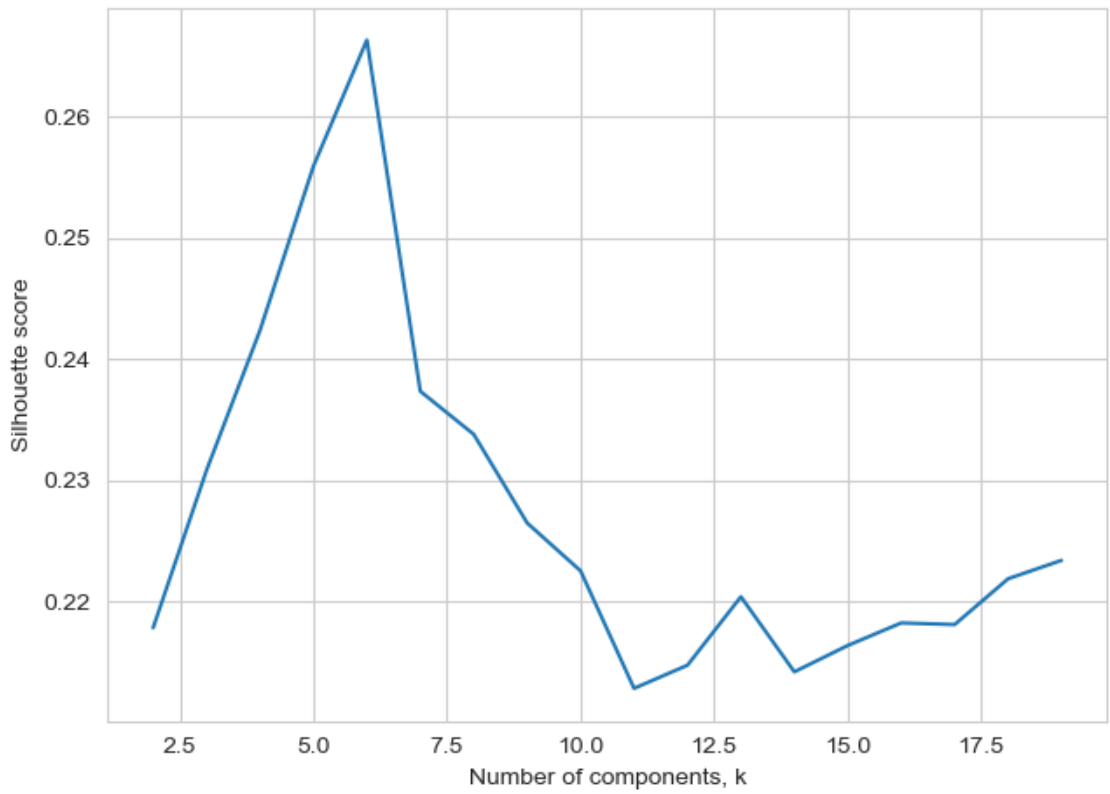


Figure 3.5: Silhouette scores of k-means algorithm on LDA output.

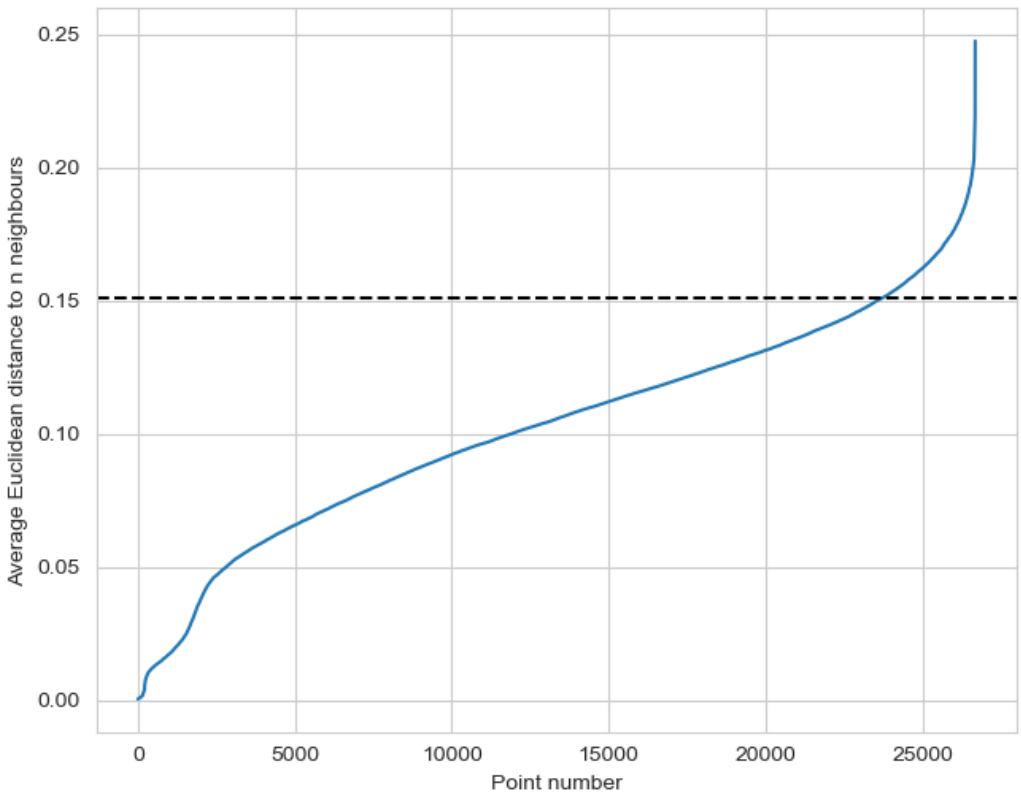


Figure 3.6: Heuristic method to find the optimal value of eps to use in the DBSCAN clustering algorithm. Black dotted line corresponds to the chosen value of eps, 0.15.

Following this, the output of LDA is also clustered using DBSCAN. Figure 3.6 shows the same plot as in 2.3iii) for this data. In comparison to Figure 2.15, a much more gradual increase in distance is shown, meaning the elbow point to isolate noise is less clear, and indicating that cluster will not be very clear. Again, the kneed Python library is utilised though, which determines the value of *eps* required to be 0.15. However, using this value groups together all the datapoints as one clusters, so this is certainly a worse method than K-means above.

## ii) Latent Semantic Indexing

Latent semantic indexing (LSI) is a similar approach to LDA, in that it returns both a set of topics and each document's topic membership, but it based around the concepts of term frequency-inverse document frequency (TF-IDF) and singular value decomposition (SVD), so is often compared to PCA as a dimensionality reduction algorithm.

TF-IDF is a representation of text similar to the document-word matrix used above, but instead of the raw frequency of each token stored, the value of each element in the matrix is equal to the term frequency (the count of term $t$ occurring in document $d$) multiplied by the logarithm of the total number of documents divided by the number of documents where term $t$ is present:

$$\text{TF-IDF} = f_{t|d} \ln\left(\frac{N}{|d \in D: t \in d|}\right) \tag{3.2}$$

This effectively brings a greater importance to terms that are more infrequent within the corpus, but frequent within a given document, whilst suppressing terms that are very common across the whole corpus.

LSI takes this TF-IDF matrix and reduces the dimensionality using the truncated form of SVD. This returns a term-concept matrix, a singular value (or eigenvalue) matrix and a concept-document matrix. Just like was the case for LDA, clustering can then be performed on the concept-document matrix.

The difficult part of this process is choosing how many dimensions to truncate the matrices to, especially in the case of clustering, where high dimensionality is a disadvantage as has been seen already. The plot of explained variance is shown in Figure 3.7. It is clear that only a very small amount of the variation is represented by this small a number of dimensions, but going into the hundreds or thousands would prove problematic, so for now it's chosen to test the clustering of the LSI output with 10 components, that explain just over 2.5% of the variance.

Unsurprisingly, given such a low percentage of variance explained with this number of components in SVD, the clustering is not successful, with no clear $k$ or *eps* values found for K-means or DBSCAN respectively (see Figure 3.8 and Figure 3.9). It would appear that the limited performance of SVD on this dataset limits this method's performance, and LDA was better suited to the task.

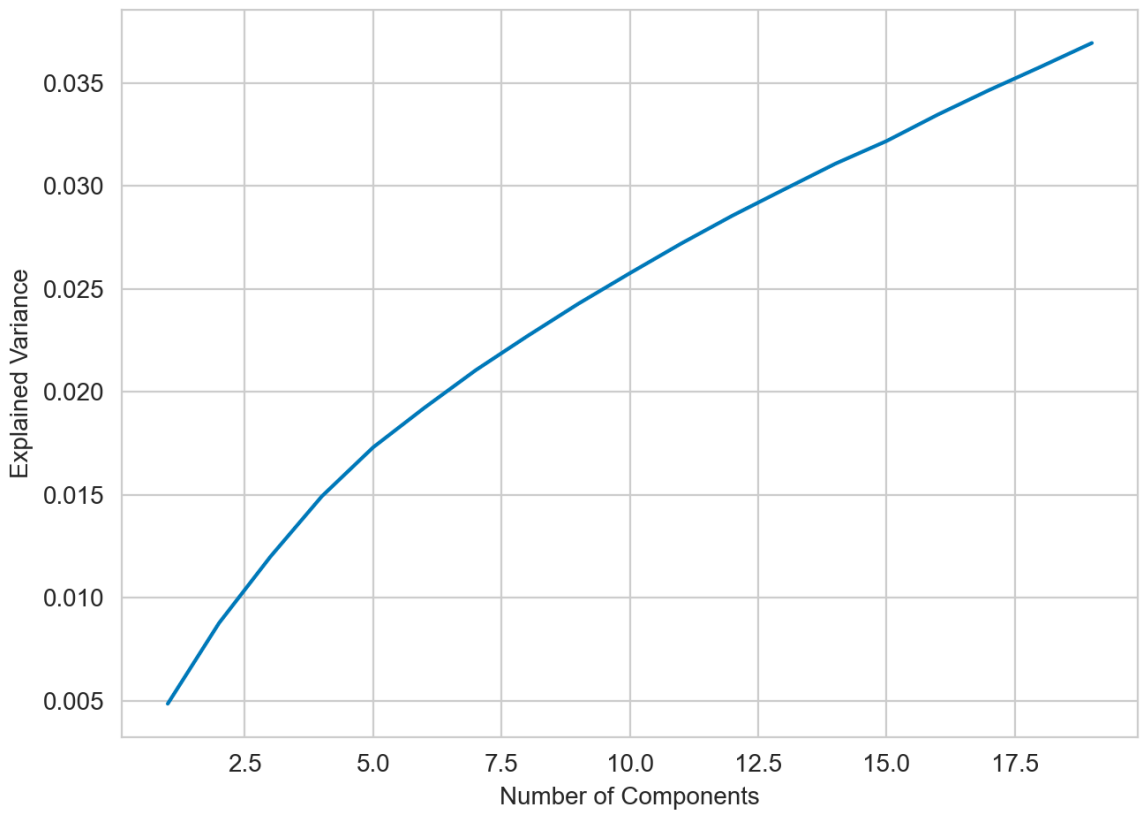


Figure 3.7: Variance of TF-IDF matrix explained in truncated SVD output, with the number of components.

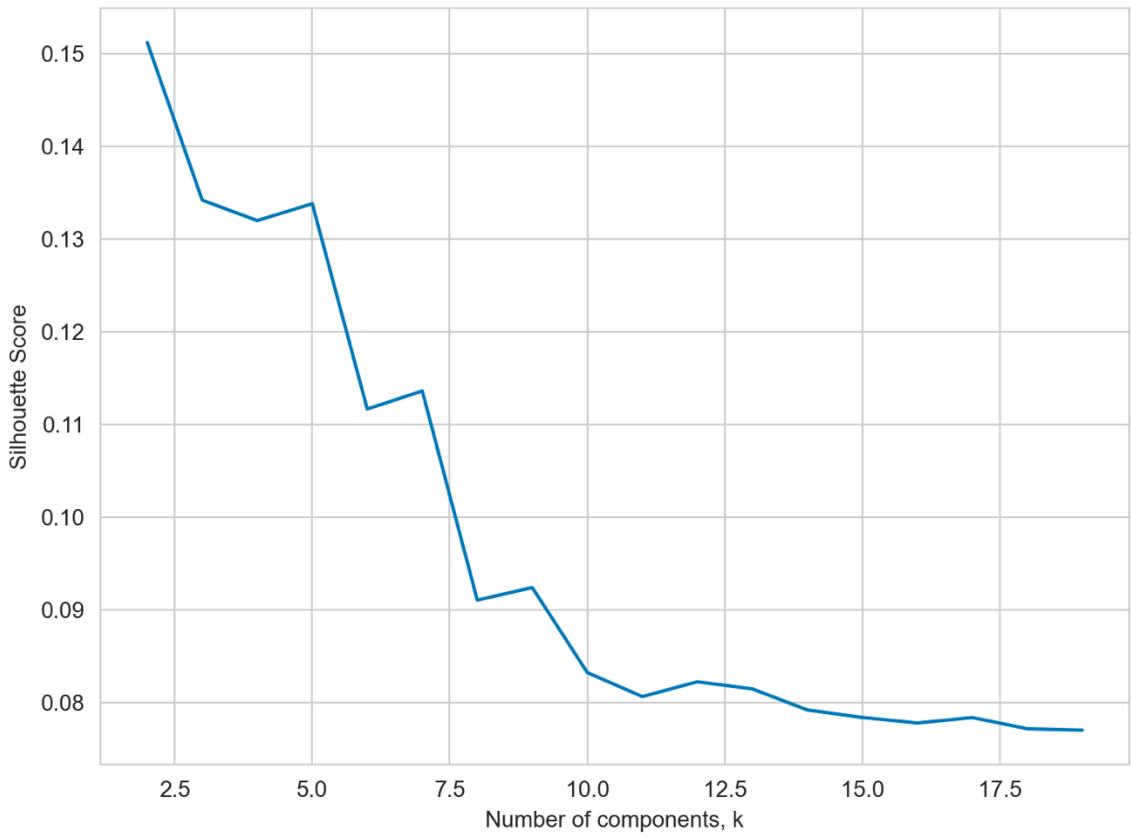


Figure 3.8: Graph of silhouette score of K-means obtained clusters on LSI output.

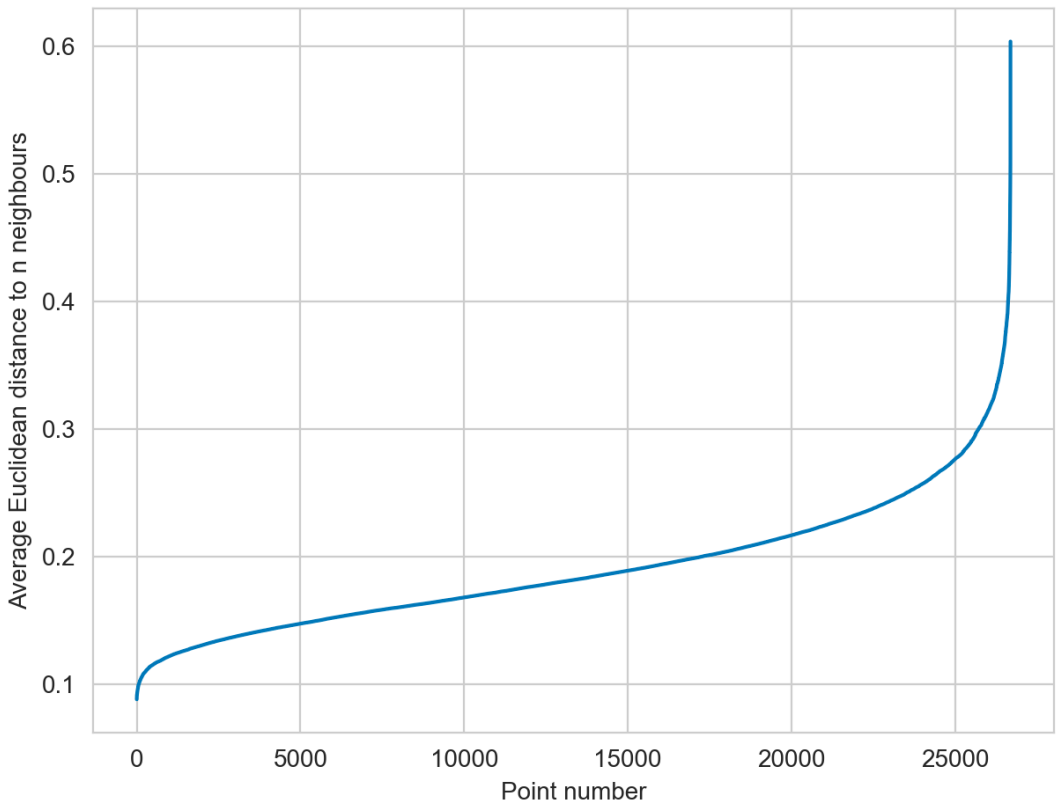


Figure 3.9: Heuristic method to determine *eps* value for DBSCAN on LSI output. No clear noise cutoff visible.

# Chapter 4 – Evaluation & Next Steps

Reviewing the outcome of the investigation, it does not appear that there was great success of profile segmentation using either the structured or the unstructured data. It is argued that the best performing approach was that of the shared-nearest-neighbour clustering on the structured features, given their apparent agreement with the t-SNE visualisation.

However, it is thought that some of the issues faced by the other algorithms would be mitigated if there was a different metric used for their optimisation processes. It's possible that such a metric, if based upon the nearest neighbours rather than an outright 'distance', would improve their performance. If more time were available, this would be the first of many next steps taken.

The clustering of text features was largely unsuccessful, with only the combination of Latent Dirichlet Allocation and K-means producing any output, but with a poor overall silhouette coefficient. It would appear that as a result of inputting the combination of all features together for each profile, both LDA and LSI were not able to distinguish very clear topics or concepts. It is thought that a large part of this problem is simply due to the input itself – in comparison with other domains on which these techniques are used like news articles or emails, these profile inputs are very informal and do not contain clear topics. Moving forward, it would be useful to perform a more complete optimisation of LDA including all its hyperparameters, not just the number of topics. However, use of these text features are likely to require either an algorithm capable of identifying very small differences, possibly based on neural networks, or a more targeted and rigid preprocessing, guided by specific domain knowledge of the types of users there are likely to be.

The latter is a point that is relevant in regard to the structured input too – after the preprocessing steps taken in this investigation, the dimensionality of the dataset caused issues for many of the approaches tested. If performed again, a more thorough feature selection process would be undertaken, after an evaluation of what are the most relevant features in relation with users' preferences in others.

Ideally, after better performance in both structured and unstructured data clustering has been achieved, they would be combined to make use of all the data available. This could be in the form of simply using all the features together in a clustering algorithm, or making use of a cluster ensemble technique to combine the output of two separate methods.

Obviously, clustering is just a one process of a dating application, which could possibly be part of a bigger recommender system. Future work would ideally use the clustering techniques from this investigation and incorporate them into said recommender system to produce a fully functional algorithm to match different users.

# Chapter 5 - Conclusions

An exploration of clustering on a dataset of 60,000 OkCupid dating profiles has been performed, utilising both the structured data, such as users' height and age, as well as the users' unstructured text input in response to the application's question prompts. This was performed in two steps, loosely following the CRISP-DM methodology's stages of data understanding, preprocessing and modelling for each type of data.

First, this was completed for the structured features, and the algorithms of K-means, DBSCAN, GMM and SNN were tested. It was found that the first three had very limited performance and did not appear to produce any meaningful clusters, but that the shared nearest neighbour approach was able to divide the data successfully, albeit with an undesirable number of datapoints discarded as noise. The clear limiting factor in the implementation of the first three algorithms was the high dimensionality of the dataset after preprocessing. Therefore, it would be desirable to repeat this part of the investigation with a stricter feature selection method, determined through domain expertise.

For the text features, the topic modelling techniques of Latent Dirichlet Allocation and Latent Semantic Indexing were applied to preprocessed text data before clustering. The former identified 6 topics, and each document's distribution among them. These distributions were then inputted into both K-means and DBSCAN clustering algorithms, with the former able to identify 6 clusters among the documents, although overall the output still had a poor overall silhouette coefficient.

Future work will thus focus on an implementation of stricter preprocessing to reduce the dimensionality of the structured dataset, more advanced algorithms for topic modelling to better divide the profiles by text, and use of a cluster ensemble method to combine the output of clustering on both types of data.

# Chapter 6 - Output

The tools used for this project were Python and its relevant libraries, including pandas, scikit-learn and spaCy. Thus, multiple scripts were written for the analysis, which were executed within the Spyder integrated development environment. It is noted that the aim of this code was solely investigation, and as such, it is not production or deployment ready.

The code filing has a structure that mirrors that of the report. It contains files for the two main chapters, `structured` and `unstructured`, that each consist of the following:

- `prep_for_eda.py`, the data manipulation required before the exploratory data analysis is to take place
- A Tableau workbook, `EDA.twb`, containing the exploratory data analysis
- `preprocessing.py`, which contains the steps taken in the preprocessing phase. In the structured version, this is split into `preprocessing_extremes.py` and `preprocessing_encoding.py`.
- A file for each of the models tested, e.g. `k-means.py` that consist of the optimisation steps, and the implementation and evaluation. In the case of the file `lda.py` it must be noted that it uses code from [55].
- A folder named `pickles` that contains all data generated by the scripts, including the preprocessed data and any graph data that took a long time to generated – i.e. the t-SNE plots.

`structured` also contains an `external` folder, within which are the functions used for SNN and the cluster interpretation in 2.4, taken from external sources.

# Chapter 7 - References


[1] A. Anzani, M. D. Sarno and A. Prunas, "Using smartphone apps to find sexual partners: A review of the literature," *Sexologies,* vol. 27, no. 3, pp. e61-e65, 2018.

[2] "Number of online dating users in the United States from 2017 to 2024," Statista Research Department, 05 07 2021. [Online]. Available: https://www.statista.com/statistics/417654/us-online-dating-user-numbers/. [Accessed 22 11 2021].

[3] D. Curry, "Dating App Revenue and Usage Statistics (2021)," Business of Apps, 15 11 2021. [Online]. Available: https://www.businessofapps.com/data/dating-app-market/. [Accessed 22 11 2021].

[4] B. Neyt, S. Baert and S. Vandenbulcke, "Never mind I'll find someone like me – Assortative mating preferences on Tinder," *Personality and Individual Differences,* vol. 155, 2020.

[5] A. Y. Kim and A. Escobedo-Land, "OkCupid Data for Introductory Statistics and Data Science Courses," *Journal of Statistics Education,* vol. 23, no. 2, 2015.

[6] R. Wirth and J. Hipp, "CRISP-DM: Towards a Standard Process Model for Data Mining," in *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, New York, 2000.

[7] The pandas development team, "pandas-dev/pandas: Pandas," *Zenodo,* 2020.

[8] Pedregosa et al, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 12, pp. 2825--2830, 2011.

[9] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017.

[10] E. Loper and S. Bird, "NLTK: The Natural Language Toolkit," in *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, Philadelphia, 2002.

[11] J. Akehurst, I. Koprinska, K. Yacef, L. Pizzato, J. Kay and T. Rej, "CCR - A Content-Collaborative Reciprocal Recommender for Online Dating," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, Barcelona, 2011.

[12] L. Brozovsky and V. Petricek, "Recommender System for Online Dating Service," 2007. [Online]. Available: https://arxiv.org/abs/cs/0703042.

[13] L. Pizzato, I. Koprinska and J. Kay, "RECON: A reciprocal recommender for online dating," in *Proceedings of the 2010 ACM Conference on Recommender Systems*, Barcelona, 2010.

[14] R. Burke, “Hybrid Recommender Systems: Survey and Experiments,” *User Modeling and User-Adapted Interaction,* vol. 12, pp. 331-370, 2002.

[15] G. Geetha, M. Safa, C. Fancy and D. Saranya, “A Hybrid Approach using Collaborative filtering and Content based Filtering for Recommender System,” *Journal of Physics: Conference Series,* vol. 1000, 2018.

[16] M. A. Ghazanfar and A. Prugel-Bennett, “A Scalable, Accurate Hybrid Recommender System,” in *2010 Third International Conference on Knowledge Discovery and Data Mining*, Phuket, 2010.

[17] T. Mikolov, K. Chen, G. Corrado and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” 2013. [Online]. Available: arxiv.org/abs/1301.3781.

[18] S. Liu, “Personalized User Recommendations at Tinder: The TinVec Approach,” in *MLconf*, San Francisco, 2017.

[19] M. Z. Rodriguez, C. H. Comin, D. Casanova, O. M. Bruno, D. R. Amancio, L. d. F. Costa and F. A. Rodrigues, “Clustering algorithms: A comparative approach,” *PLOS ONE,* vol. 14, no. 1, 2019.

[20] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967.

[21] Google Developers, “k-Means Advantages and Disadvantages,” Google, 13 01 2021. [Online]. Available: https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages. [Accessed 24 11 2021].

[22] M. Ester, H. P. Kriegel, J. Sander and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, 1996.

[23] G. J. McLachlan and K. E. Basford, “Mixture Models: Inference and Applications to Clustering,” *Journal of the Royal Statistical Society,* vol. 152, no. 1, pp. 126-127, 1989.

[24] S. Patel, S. Sihmar and A. Jatain, “A study of hierarchical clustering algorithms,” in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Dehli, 2015.

[25] L. Rochat, F. Bianchi-Demicheli, E. Aboujaoude and Y. Khazaal, “The psychology of “swiping”: A cluster analysis of the mobile dating app Tinder,” *Journal of Behavioral Addictions,* vol. 8, no. 4, pp. 804-813, 2019.

[26] G. Vojt, M. Smith, M. W. Owusu, F. Mapp, M. Pothoulaki, C. Estcourt and P. Flowers, “How do Dating Apps reflect the social organisation of sexual relationships? A review of dating apps and their key features,” 2021. [Online]. Available: https://doi.org/10.31235/osf.io/wf3vd.

[27] T. v. d. Zanden, A. P. Schouten, M. Mos, C. v. d. Lee and E. Krahmer, "Effects of Relationship Goal on Linguistic Behavior in Online Dating Profiles: A Multi-Method Approach," *Frontiers in Communication,* vol. 4, 2019.

[28] M. Steinbach, G. Karypis and V. Kumar, "A Comparison of Document Clustering Techniques," *Retrieved from the University of Minnesota Digital Conservancy,* 2000.

[29] I. Vayansky and S. A. Kumar, "A review of topic modeling methods," *Information Systems,* vol. 94, 2020.

[30] Scikit-Learn, "Tf–idf term weighting," [Online]. Available: https://scikit-learn.org/stable/modules/feature_extraction.html#tfidf-term-weighting. [Accessed 25 11 2021].

[31] R. Caruana, M. Elhawary, N. Nguyen and C. Smith, "Meta clustering," in *Sixth International Conference on Data Mining*, Hong Kong, 2006.

[32] P. Gao, X. Wang, H. Chen, W. Dai and H. Ling, "What is beautiful is not always good: influence of machine learning-derived photo attractiveness on intention to initiate social interactions in mobile dating applications," *Connection Science,* vol. 33, no. 2, pp. 321-340, 2021.

[33] D. Paraschakis and B. J. Nilsson, "Matchmaking Under Fairness Constraints: A Speed Dating Case Study," *Communications in Computer and Information Science,* vol. 1245, 2020.

[34] D. Black, G. Gates, S. Sanders and L. Taylor, "Why Do Gay Men Live in San Francisco?," *Journal of Urban Economics,* vol. 51, no. 1, pp. 54-76, 2002.

[35] Á. Castro, J. R. Barrada, P. J. Ramos-Villagrasa and E. Fernández-del-Río, "Profiling Dating Apps Users: Sociodemographic and Personality Characteristics," *International Journal of Environmental Health and Public Reserach,* vol. 17, no. 3653, 2020.

[36] D. A. Weiser, S. Niehuis, J. Flora, N. M. Punyanunt-Carter, V. S. Arias and R. H. Baird, "Swiping right: Sociosexuality, intentions to engage in infidelity, and infidelity experiences on Tinder," *Personality and Individual Differences,* vol. 133, pp. 29-33, 2018.

[37] OkCupid, "Community Guidelines," 2021 07 2021. [Online]. Available: https://help.okcupid.com/article/142-community-guidelines. [Accessed 05 09 2021].

[38] R. F. Baumeister, "Gender differences in erotic plasticity: the female sex drive as socially flexible and responsive.," *Psychological Bulletin,* vol. 126, no. 3, pp. 347-374, 2000.

[39] L. v. d. Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research,* vol. 9, no. 86, p. 2579−2605, 2008.

[40] Wattenberg et al., "How to Use t-SNE Effectively," Distil, 2016. [Online]. Available: https://distill.pub/2016/misread-tsne/. [Accessed 28 09 2021].

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau and Brucher, "Clustering," Scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/clustering.html. [Accessed 29 09 2021].

[42] C. C. Aggarwal, A. Hinneburg and D. A. Keim, "On the Surprising Behavior of Distance Metrics in High Dimensional Space," in *Database Theory — ICDT 2001*, London, UK, 2001.

[43] A. Flexer and D. Schnitzer, "Choosing $\ell$p norms in high-dimensional spaces based on hub analysis," *Austrian Research Institute for Artificial Intelligence,* vol. 169, pp. 281-287, 2015.

[44] N. Tomašev, M. Radovanović, D. Mladenić and M. Ivanović, "Hubness-Based Clustering of High-Dimensional Data," *Partitional Clustering Algorithms,* pp. 353-386, 2015.

[45] J. Sander, M. Ester, H.-P. Kriegel and X. Xu, "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications," *Data Mining and Knowledge Discovery,* vol. 2, pp. 169-194, 1998.

[46] O. C. Carrasco, "Gaussian Mixture Models Explained," Towards Data Science, 03 06 2019. [Online]. Available: https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95. [Accessed 17 10 2021].

[47] R. A. Patrick and E. A. Jarvis, "Clustering Using a Similarity Measure Based on Shared Near Neighbors," *IEEE Transactions on Computers,* Vols. C-22, no. 11, 1973.

[48] L. Ertöz, M. Steinbach and V. Kumar, "Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data," in *Proceedings of the 2003 SIAM International Conference on Data Mining (SDM)*, San Francisco, 2003.

[49] F. L. A. Vieira, "SharedNearestNeighbors," 15 03 2019. [Online]. Available: https://github.com/felipeangelimvieira/SharedNearestNeighbors. [Accessed 17 10 2019].

[50] W. Waisakurnia, "The Easiest Way to Interpret Clustering Result," Towards Data Science, 13 6 2020. [Online]. Available: https://towardsdatascience.com/the-easiest-way-to-interpret-clustering-result-8137e488a127. [Accessed 25 10 2021].

[51] J. M. Patel, "Setting up text preprocessing pipeline using scikit-learn and spaCy," Towards Data Science, 06 03 2020. [Online]. Available: https://towardsdatascience.com/setting-up-text-preprocessing-pipeline-using-scikit-learn-and-spacy-e09b9b76758f. [Accessed 12 11 2021].

[52] D. M. Blei, A. Y. Ng and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research,* vol. 3, pp. 993-1022, 2003.

[53] R. Kulshrestha, "A Beginner's Guide to Latent Dirichlet Allocation (LDA)," Towards Data Science, 19 07 2019. [Online]. Available:

https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2. [Accessed 15 11 2021].

[54] Great Learning Team, "Understanding Latent Dirichlet Allocation (LDA)," Great Learning Blog, 16 10 2020. [Online]. Available: https://www.mygreatlearning.com/blog/understanding-latent-dirichlet-allocation/. [Accessed 18 11 2021].

[55] S. Prabhakaran, "LDA in Python – How to grid search best topic models?," machinelearningplus, 04 04 2018. [Online]. Available: https://www.machinelearningplus.com/nlp/topic-modeling-python-sklearn-examples/#10diagnosemodelperformancewithperplexityandloglikelihood. [Accessed 18 11 2021].

[56] J. Wolstein, S. Babey and A. Diamant, Obesity in California, Los Angeles: UCLA Center for Health Policy Research, 2015.

[57] K. Asher, C. Green, D. H. Gutbrod, M. Jewell, G. Hale and B. Bastian, Study of Current and Former Vegetarians and Vegans, Humane Research Council, 2014.

[58] H. L. Corliss, M. Rosario, D. Wypij, S. A. Wylie, A. L. Frazier and S. B. Austin, "Sexual orientation and drug use in a longitudinal cohort study of U.S. adolescents," *Addictive Behaviors,* vol. 35, no. 5, pp. 517-521, 2010.

[59] J. A. Ford and J. L. Jasinski, "Sexual orientation and substance use among college students," *Addictive Behaviours,* vol. 31, no. 3, pp. 404-413, 2006.

[60] "Educational Attainment, Annual: Bachelor's Degree or Higher (5-year estimate) by County," FRED, Federal Reserve Bank of St. Louis, [Online]. Available: https://fred.stlouisfed.org/release/tables?rid=330&eid=391496. [Accessed 7 September 2021].

[61] Institute of Education Sciences, U.S. Department of Education, "Undergraduate Retention and Graduation Rates," National Center for Education Statistics, 2020.

[62] C. Gecewicz, "'New Age' beliefs common among both religious and nonreligious Americans," Pew Research Center, 1 10 2018. [Online]. Available: https://www.pewresearch.org/fact-tank/2018/10/01/new-age-beliefs-common-among-both-religious-and-nonreligious-americans/. [Accessed 08 09 2021].

[63] D. Looney and N. Lusin, "Enrollments in Languages Other Than English in United States Institutions of Higher Education, Summer 2016 and Fall 2016: Preliminary Report," Modern Language Association of America, 2018.

[64] (ACTFL), American Council on the Teaching of Foreign Languages, "Foreign Language Enrollments in K–12 Public Schools (Archived from Original)," 02 2011. [Online]. Available: https://web.archive.org/web/20140817144116/http://www.actfl.org/sites/default/files/pdfs/ReportSummary2011.pdf. [Accessed 09 09 2021].

[65] M. Balatsko, "Text preprocessing steps and universal reusable pipeline," Towards Data Science, 19 05 2019. [Online]. Available: https://towardsdatascience.com/text-preprocessing-steps-and-universal-pipeline-94233cb6725a. [Accessed 03 10 2021].

[66] M. Li, Y. Li, W. Lou and L. Chen, "A hybrid recommendation system for Q&A documents," *Expert Systems with Applications,* vol. 144, 2020.

# Chapter 8 - Appendices

## Appendix A – Structured Data EDA

### Body Type

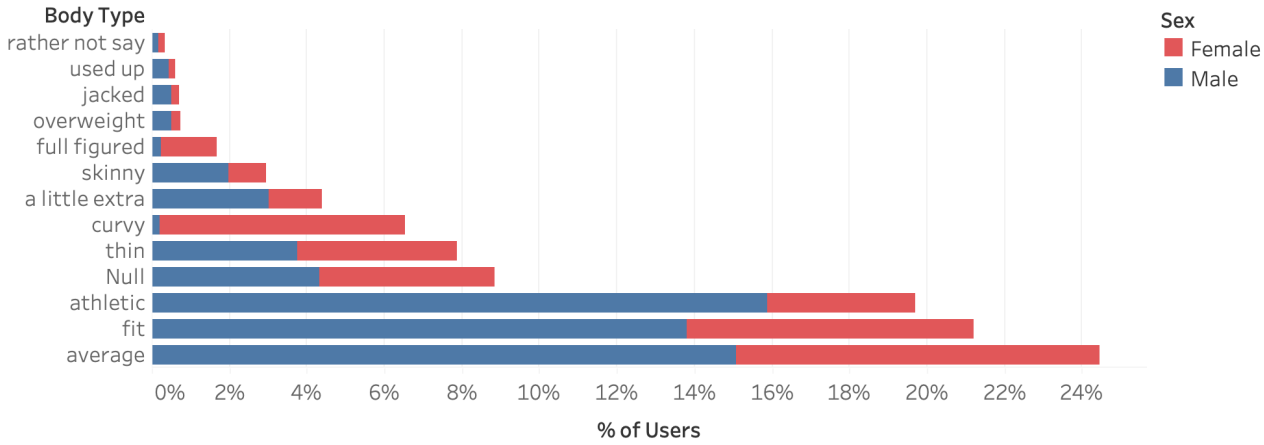


Figure 8.1: Percentage of users by listed body type.

The most notable characteristic of the split of body types in Figure 8.1 is the trend of users not to describe their body in a way that would be perceived as negative – as is logical. The overweight category comprises only 0.74% of the total userbase, while the adult obesity weight in 2011-2012 was 11% in the country of San Francisco (where the majority of users are), and 25% for the whole of California [56]. The options of 'full figured' and 'curvy' are most likely used as euphemisms for those that are overweight – both utilised almost entirely by female accounts. A similar category with a male majority is 'a little extra'.

Such tendencies of users to list themselves as a more appealing category, along with the multiple redundant categories in the feature (it's not clear what the difference is between 'athletic' and 'fit', for example) suggest that this variable may not be useful in the segmentation process.

### Diet

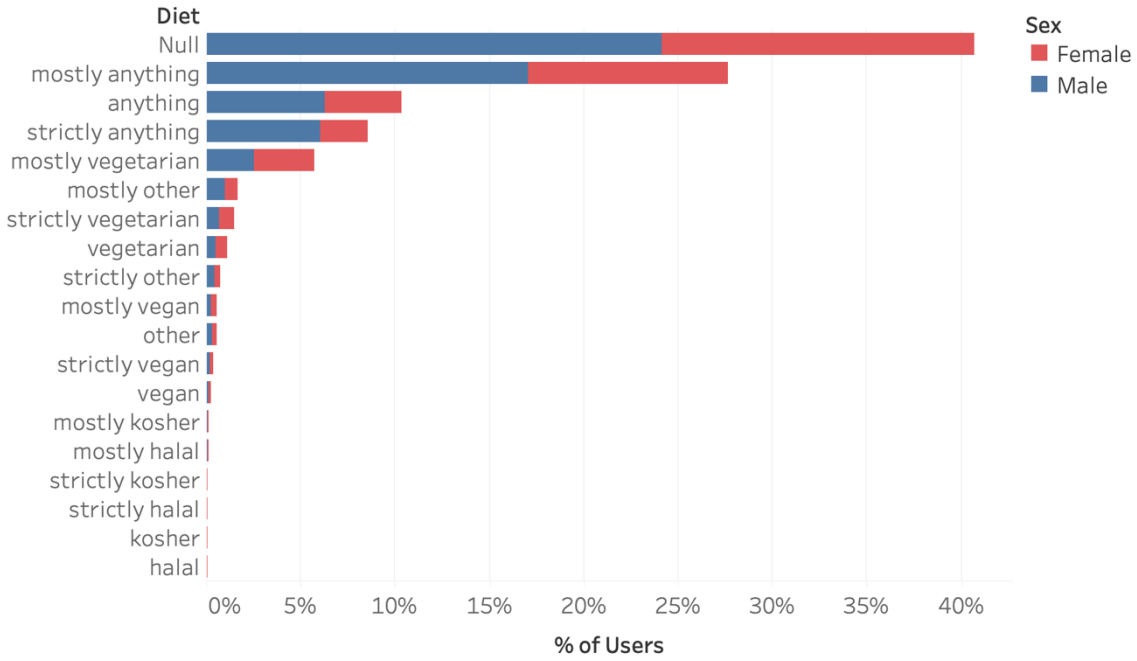


Figure 8.2: Split of users by listed diet.

It appears from Figure 8.2 that the majority of users do not have any dietary requirements. Although the modal value is no input, the next 3 top values are all variations of 'anything', and it is possible that a large proportion of those users without any input left the option blank because they had no restrictions - i.e. they also eat anything. However, it could also be that they do not want to reveal what their dietary choices are in order to remain unjudged, it's impossible to say with certainty.

It's also noteworthy that of the 6 diet options that are without meat (vegan and vegetarian), 5 are majority-female. This is known to be the case for the demographic of vegans/vegetarians in the US [57], and could be a result of a common link between meat-eating and masculinity.

### Drinks

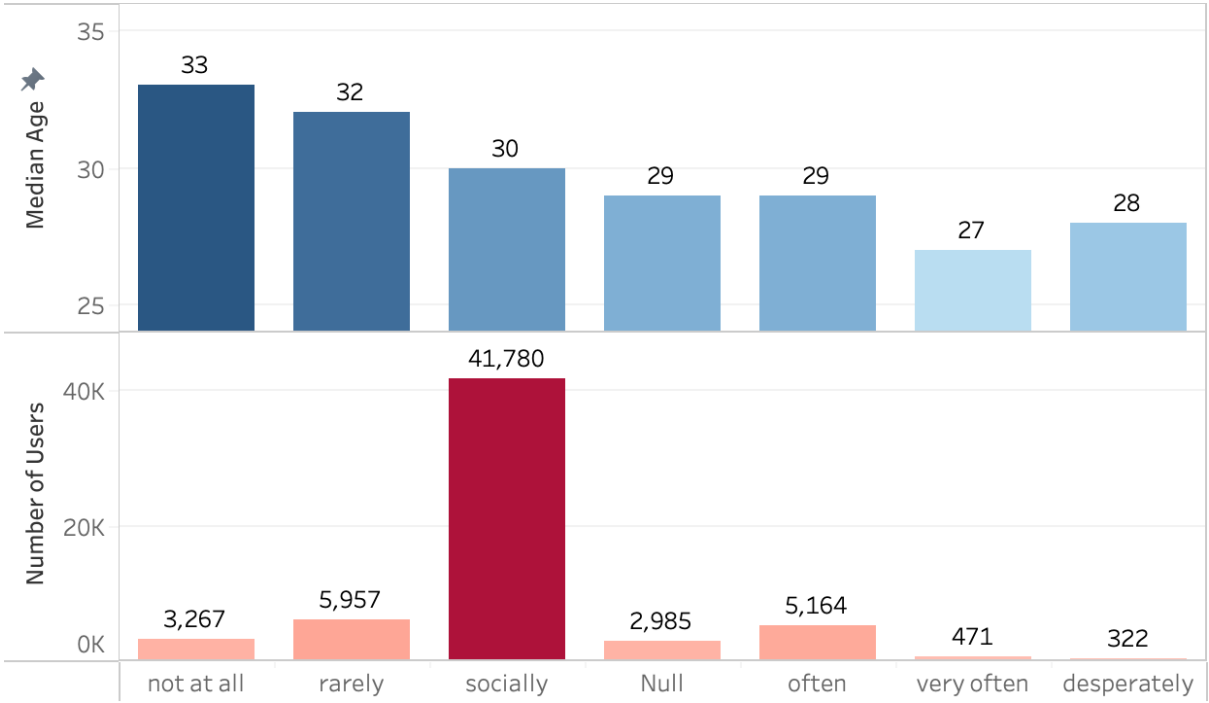


Figure 8.3: Number of users and median age for each drinking category.

The categories available in the *drinks* feature have a clear order among them, from 'not at all' to 'desperately'. The vast majority of users in the dataset have listed themselves as drinking 'socially' which, while not in itself a frequency, can be thought of as fulfilling the space left by the lack of an 'average' option. In Figure 8.3 the median age is also displayed for each category (a better descriptive statistic than the mean due to the right-skewed distribution of age). It's interesting to note the decline in age from those who don't drink at all to those drinking with the highest frequencies, although this could be a chance result due to the smaller number of users in 'very often' and 'desperately'.

### Drugs

The modal value by a long way of this feature is 'never', but there are a lot more null values present in the dataset than for drinking. This is likely due to the taboo nature of drugs meaning that users do not want to make their habits public, neither to potential partners nor to anyone else who might disapprove if they found their profile (e.g. employers or family).

Again there is a similar pattern in terms of median age seen in Figure 8.4, with those taking drugs more often having a lower age. Another interesting, and slightly unexpected relationship is found in Figure 8.5, where there is an almost identical split of *drugs* input for both straight and gay users, but for bisexuals the proportion of users stating they never take drugs is reduced from around 65% to under 40%, and there is

a large increase in the percentage of users who entered ‘sometimes’ or ‘often’. It must of course be considered that bisexual is the least common entry for orientation and therefore is the most subject to random variation in its distribution. However, with almost 2,800 bisexual users in the dataset this difference is not just by chance.

The reason for this difference is unclear, but previous evidence [58, 59] has agreed on a conclusion that bisexuals, especially females, are the most likely demographic to report drug use, and suggests that a contributing factor to this could be the higher rates of depression amongst sexual minorities.

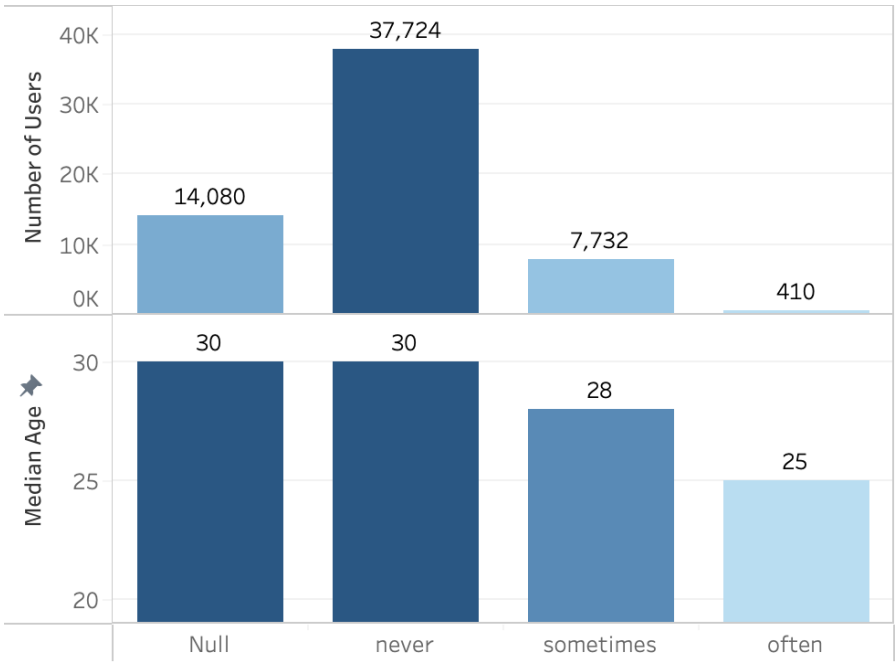


Figure 8.4: Number of users and median age for each response to the 'Drugs' feature.

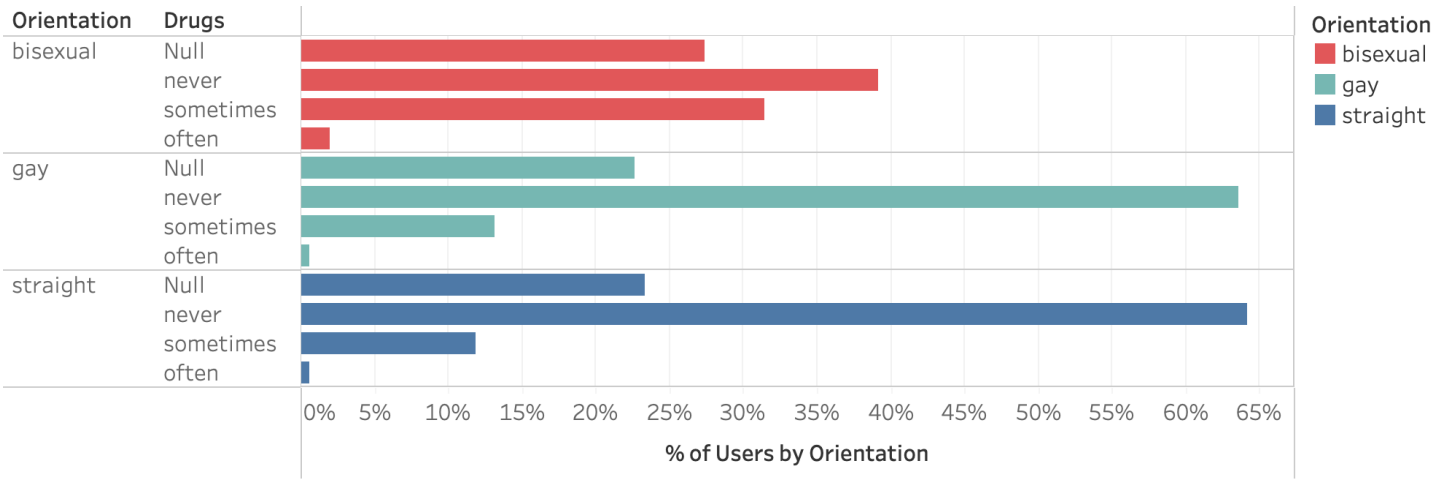


Figure 8.5: Percentage of users' response to the 'Drugs' option, for each sexual orientation.

### Education

The top 15 unique values of the *education* feature are presented in Table 8.1, out of the total 32 unique entries. It's seen that the majority of entries can be separated into an education level (i.e. high school or university) and their status in that educational institution (whether they are currently enrolled, have graduated or dropped out). Thus, for this analysis of the dataset it's logical to split the feature into these two parts. Figure 8.6 and Figure 8.7 show their distributions.

| Education | Number of Users |
|---|---|
| graduated from college/university | 23,959 |
| graduated from masters program | 8,961 |
| Null | 6,628 |
| working on college/university | 5,712 |
| working on masters program | 1,683 |
| graduated from two-year college | 1,531 |
| graduated from high school | 1,428 |
| graduated from ph.d program | 1,272 |
| graduated from law school | 1,122 |
| working on two-year college | 1,074 |
| dropped out of college/university | 995 |
| working on ph.d program | 983 |
| college/university | 801 |
| graduated from space camp | 657 |
| dropped out of space camp | 523 |

Table 8.1: Top 15 most selected inputs for Education.

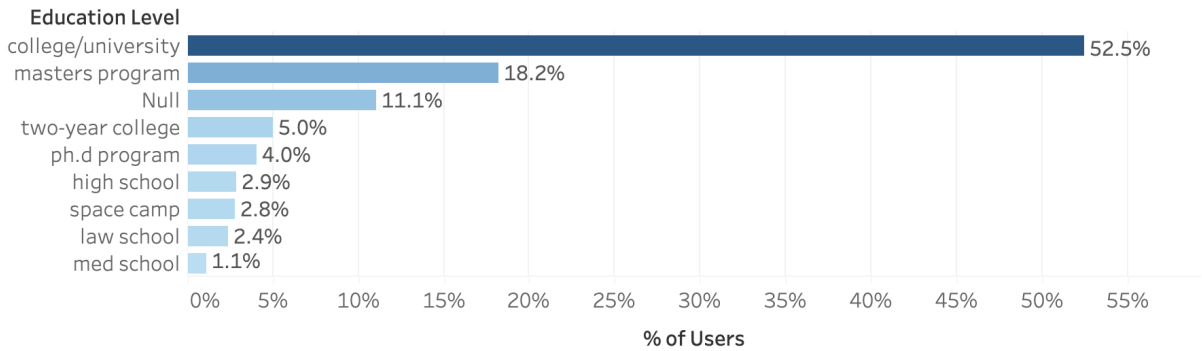


Figure 8.6: Division of users by education level.

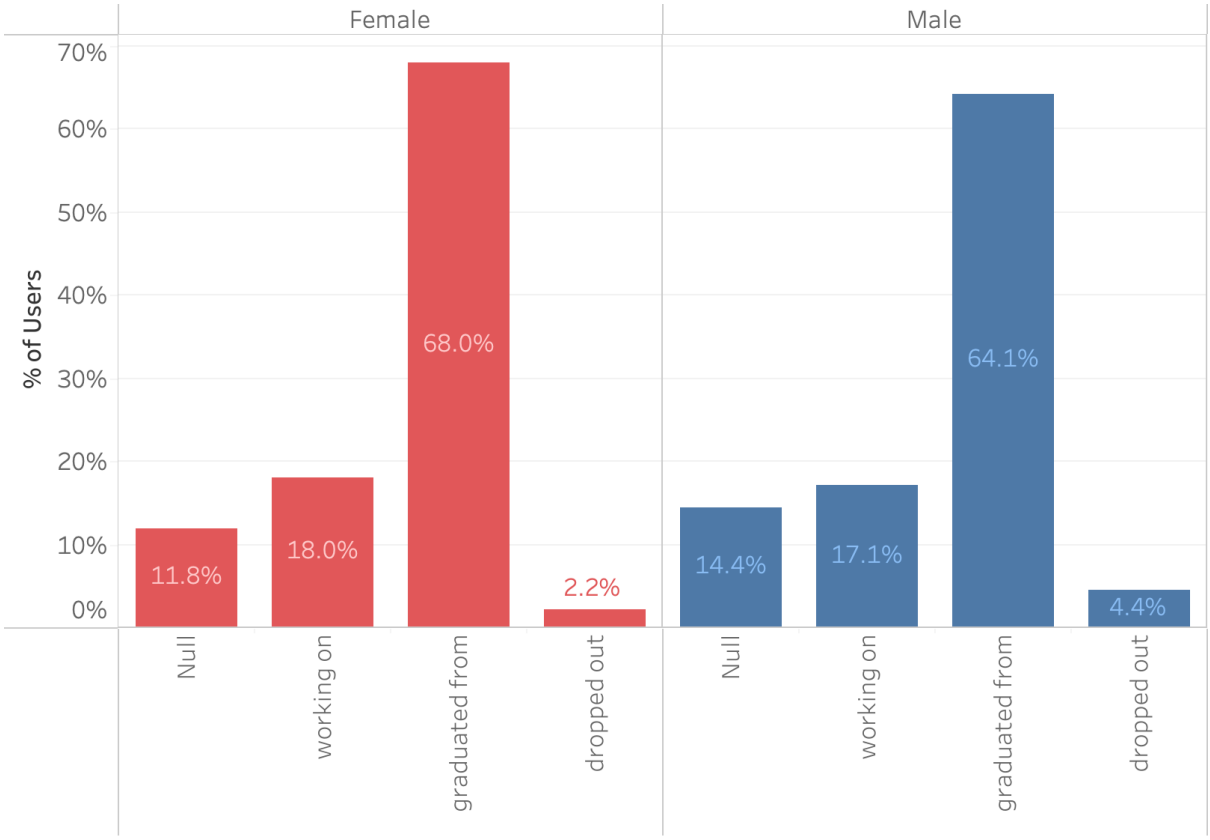


Figure 8.7: The division of users' education status for each sex.

The most common education level amongst the users is 'college/university', with just over half of the records, followed by 'masters program' with just under 20% of the userbase. San Francisco is known to be an area with much higher educational attainment than other parts of the US [60], so these numbers are not surprising.

One notable aspect of the education status though, is that the proportion of men who listed as 'dropped out' is double that of women. Again, although this relationship may at first appear odd, previous research has found a lower undergraduate graduation rate for males in comparison to females in the US (59% to 65%) [61].

### Ethnicity

There are 217 unique values for this feature due to the fact that the app has let users pick many different combinations of races. Whilst the modal value is 'white' with 58% of the records, there are a huge number of ethnicity combinations each with only a very small number of users.

| Ethnicity | % of Users |
|---|---|
| white | 57.8 |
| asian | 10.8 |
| Null | 10.0 |
| hispanic / latin | 5.0 |
| black | 3.5 |
| other | 3.0 |
| hispanic / latin, white | 2.3 |
| indian | 1.9 |
| asian, white | 1.4 |
| white, other | 1.1 |
| pacific islander | 0.8 |
| asian, pacific islander | 0.7 |
| native american, white | 0.6 |
| middle eastern | 0.6 |
| middle eastern, white | 0.5 |

Table 8.2: Percentage of users listed as each ethnicity combination, limited to the top 15.

### Height

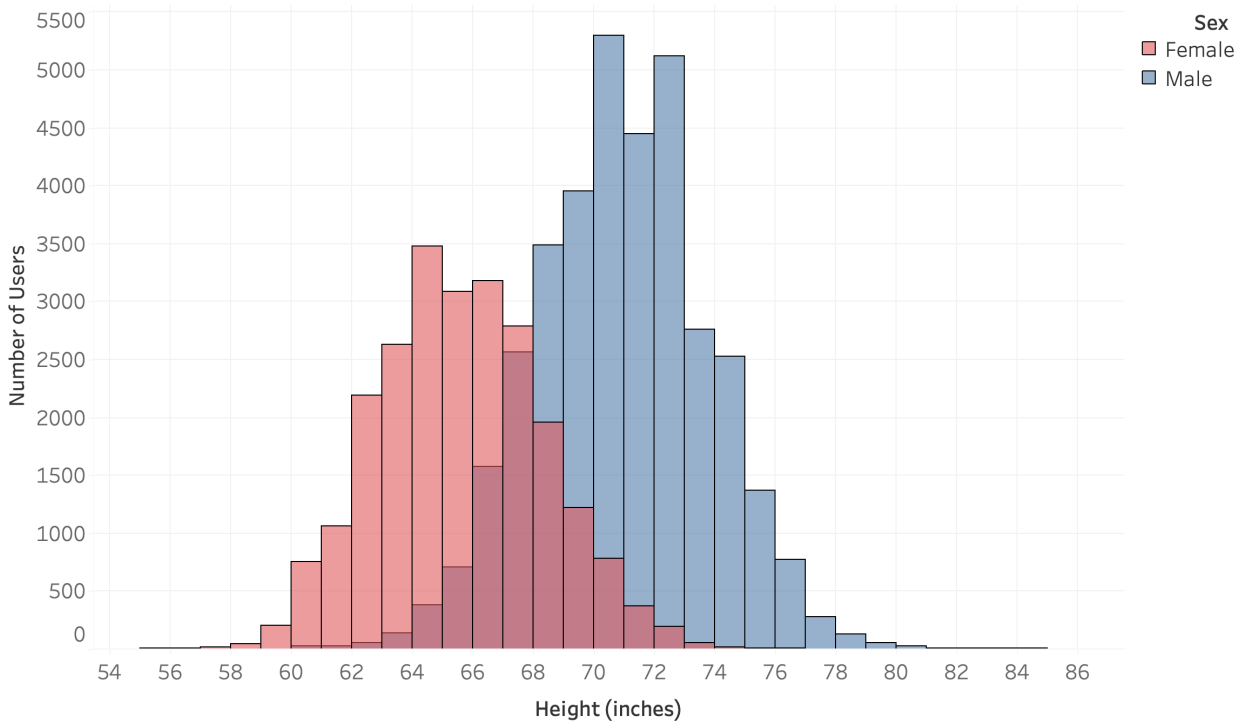


Figure 8.8: Distribution of height listed for each gender amongst the userbase. Limited to between 54 and 86 inches.

The distributions of *height* for each gender appear to be symmetrical, as would be expected, with a mean for males of 70.4 inches and 65.1 inches for females.

However, each sex contains its own anomalous peak. For males, this occurs at 72 inches – i.e. 6 feet. As mentioned in the original source [5], this is likely a result of users inflating their height to fit sociological perceptions like around round numbers and the widespread standard amongst women to not date men under this height. The peak for females is seen at 64 inches, or 5'4" for women, although the cause of this is less clear.

### Last Online

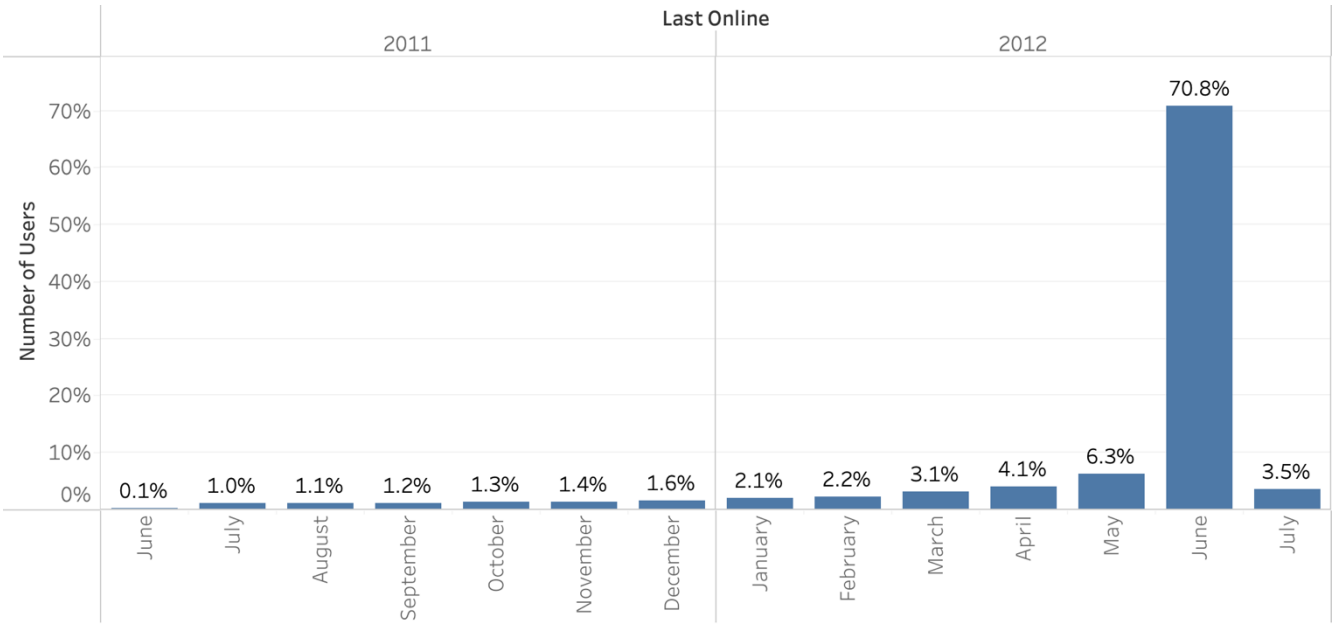


Figure 8.9: Split of users' profiles by their last login to the service.

It is seen in Figure 8.9 that the vast majority of users were last online during June of 2012, which is what is expected given that the profile scraping took place on the last day of that month [5]. The very small proportion of profiles online in July of that year must be a result of the scraping continuing into the early hours of July 1st.

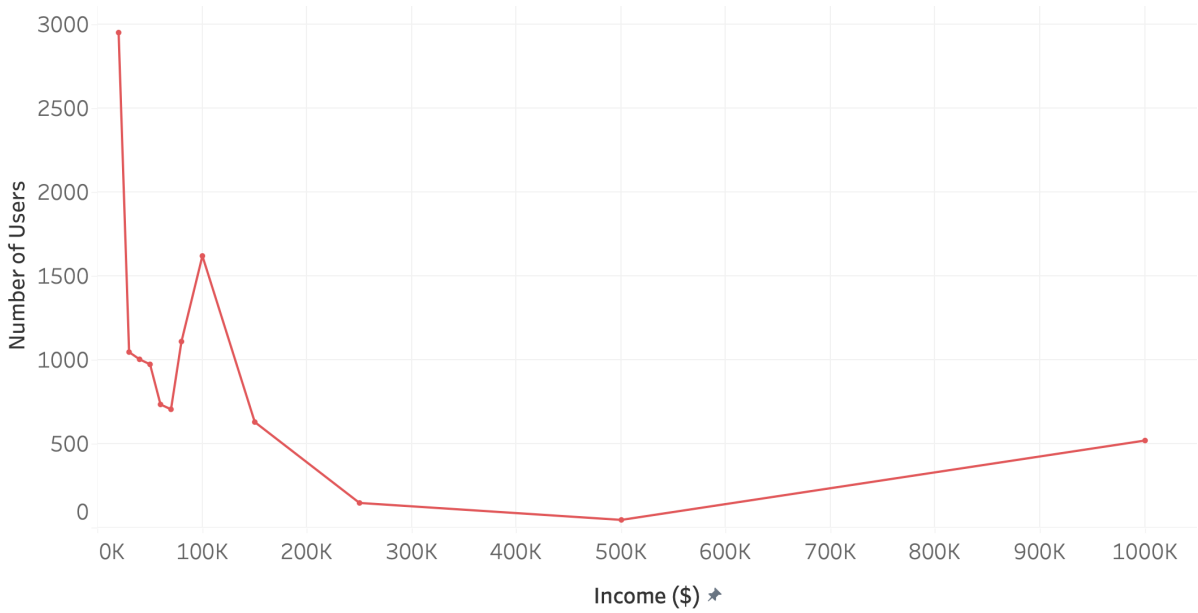


Figure 8.10: Number of users listed for each income level (null values not included).

### Income

Whilst it's possible to view the spread of incomes in the dataset, as has been visualised in Figure 8.10, it should be recalled that this is the feature with the largest number of null values – 48442 out of the total 59946 profiles. This, along with the fact there are limited input options available to users (and those provided are non-uniformly spread out) mean any conclusions drawn from this curve would be unreliable.

### Job

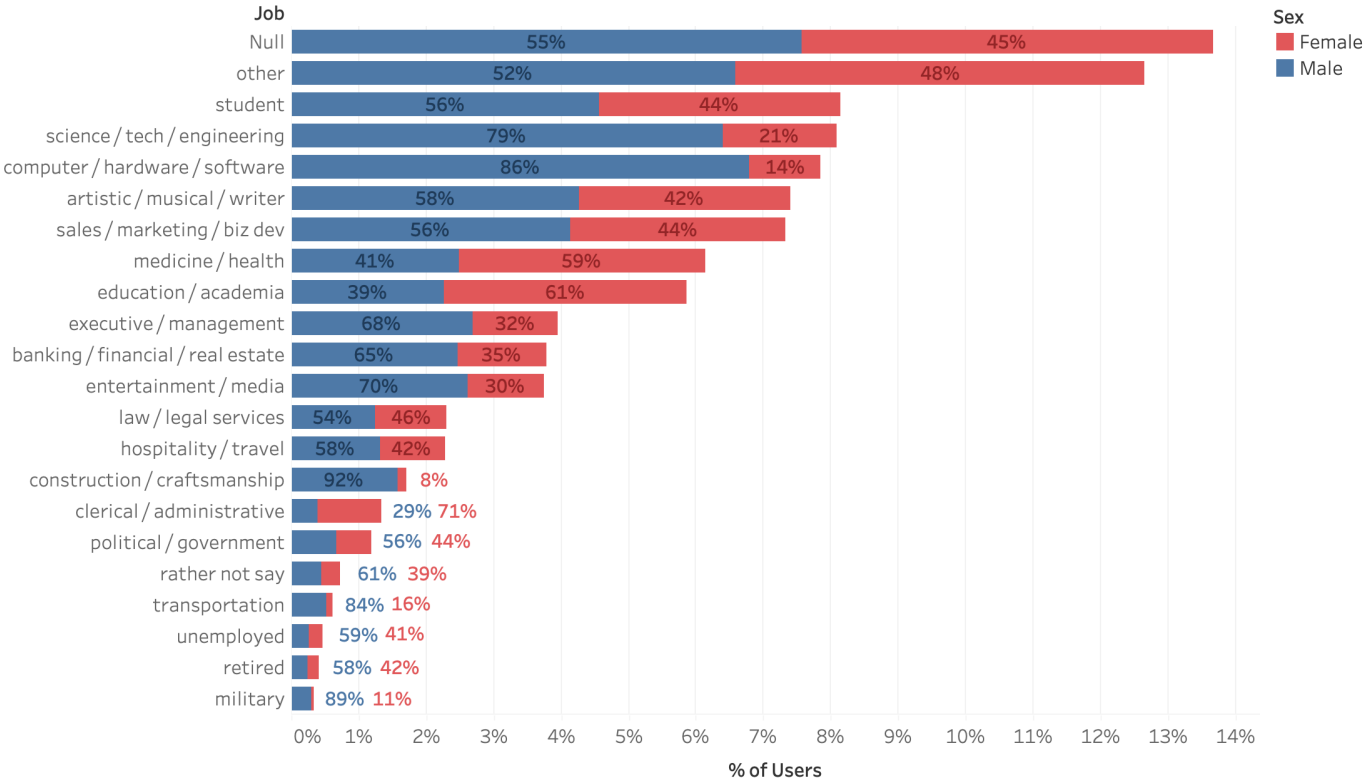


Figure 8.11: Proportion of users listed as each job category, and their split between the sexes.

The largest categories in the *job* feature are seen in Figure 8.11 are unspecific, with the top 3 being no listing at all, followed by other and then student. However, there are still a large number of profiles spread between the different industry options. It is perhaps unsurprising given the San Francisco locale that science and technology jobs are the top two industries, and that each of these has a majority of males. Other clearly male-dominated industries listed are construction/craftmanship, transportation and the military. However there are clearly female dominated industries including medicine/health, education/academia and clerical/administrative.

### Offspring

The input values for *offspring*, displayed with their proportion of the users in dataset in Figure 8.12, can be split into two parts – whether a user has children now, and whether they want to have children (more if they already have them). Therefore two new features are generated by splitting the original string. The results of this are found in Figure 8.13, and it's seen that while the majority of users did not give an input, there is a clear difference in the desire of users to have children depending on if they have them already – nearly 20% of those listed as not having children say they want to have some, but if they already do then this number drops to only 2%. Similarly the proportion who say they maybe want children drops, while the percentage who state that they do not almost triples from 6% to 16%.

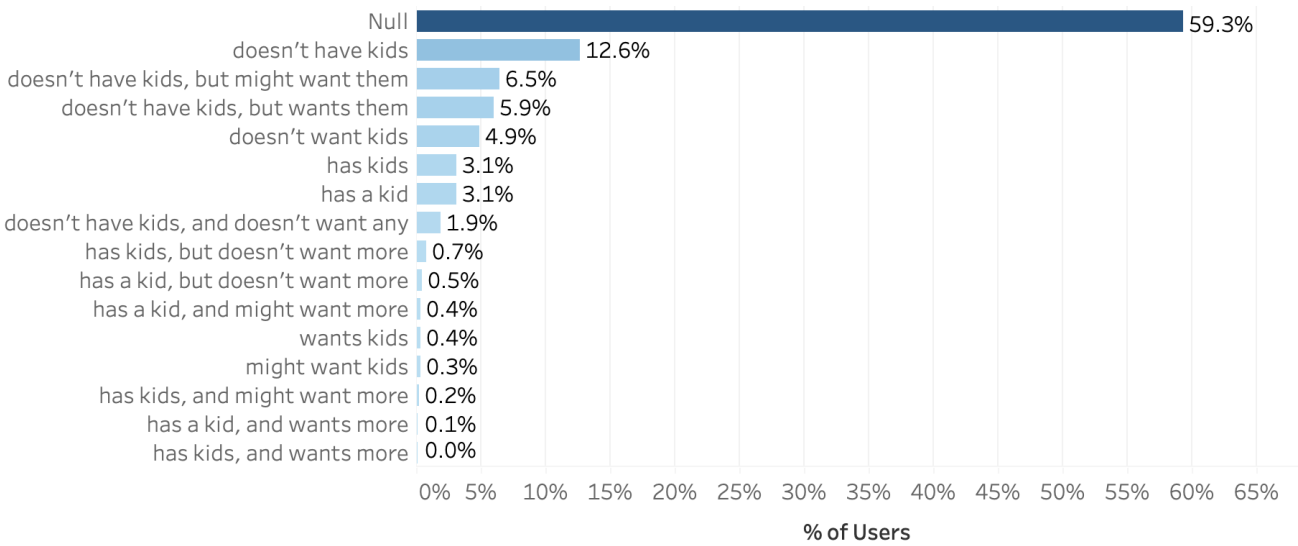


Figure 8.12: Proportion of users for each input of the offspring feature.

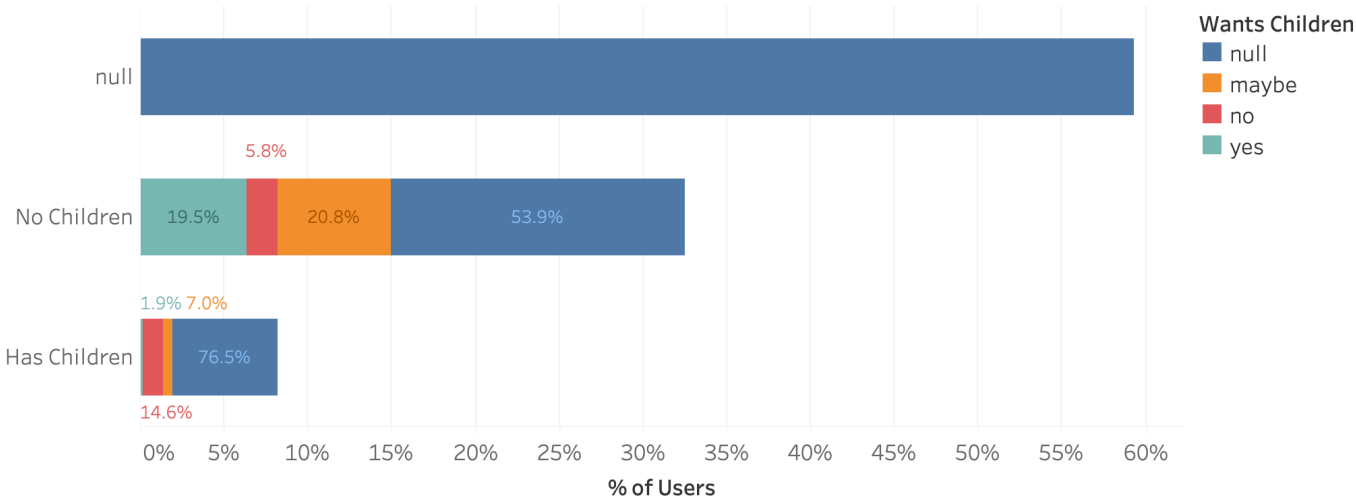


Figure 8.13: Users' inputs of the offspring feature broken into whether they have and whether they want children.

### Pets

A similar feature split can be performed on *pets* to that which was performed on offspring. Each user has been given the option for both cats and dogs, for whether they have, do not have, like or dislike them. The distribution of these options can be seen in Figure 8.14. It should be noted that it appears from the original string that if users listed themselves as having a dog or a cat, then they were not listed by the app as liking or disliking them. Thus any null values of *likes_ dogs* and *likes_ cats* where the user is listed as possessing the respective pet have been set so that they also are listed as liking them.

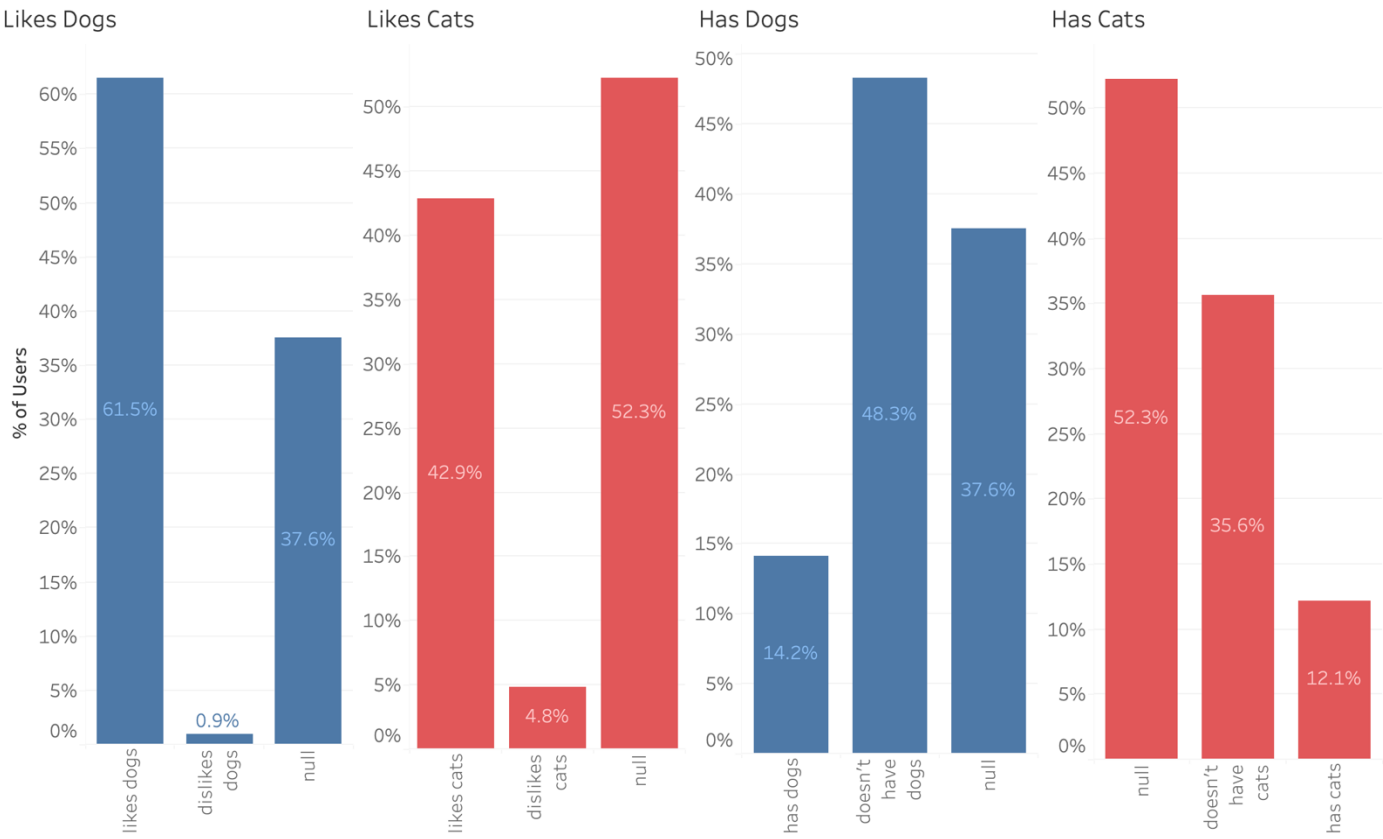


Figure 8.14: Distribution of users' opinions on cats and dogs and whether or not they have them.

From the figure it's clear that amongst this dataset, dogs are favoured over their feline counterparts, with a higher proportion of users liking and possessing them, and a much smaller percentage of dislikes.

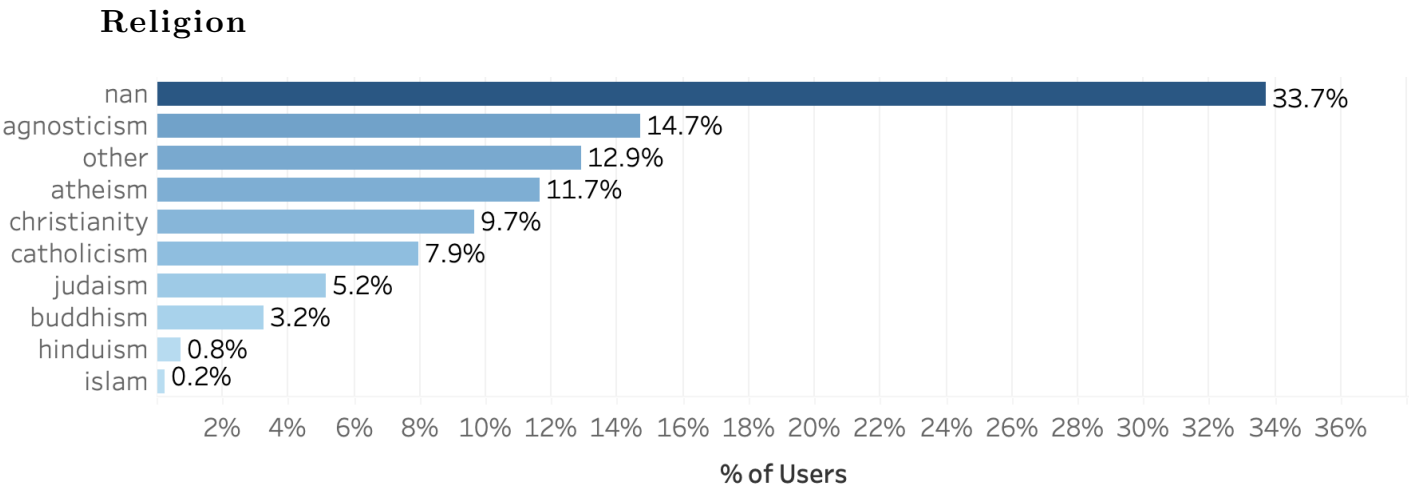


Figure 8.15: Proportion of users listed as each religion.

Once more the detail in this feature requires some manipulation in order to fully analyse it. It contains both the user's listed religion and their attitude towards it - Figure 8.15 depicts the former and Figure 8.16 the latter. It's notable that the highest chosen religion other than leaving the input blank was agnosticism, with atheism also with a high percentage. Perhaps this could be another characteristic of the city of San Francisco – not being very religious, or it could be that a lot of people choose not to list their religion as they do not believe it suitable for their dating profile.

With regard to the commitment for each religion, the standout details are that Judaism appears to have the least committed members, closely followed by Catholicism.

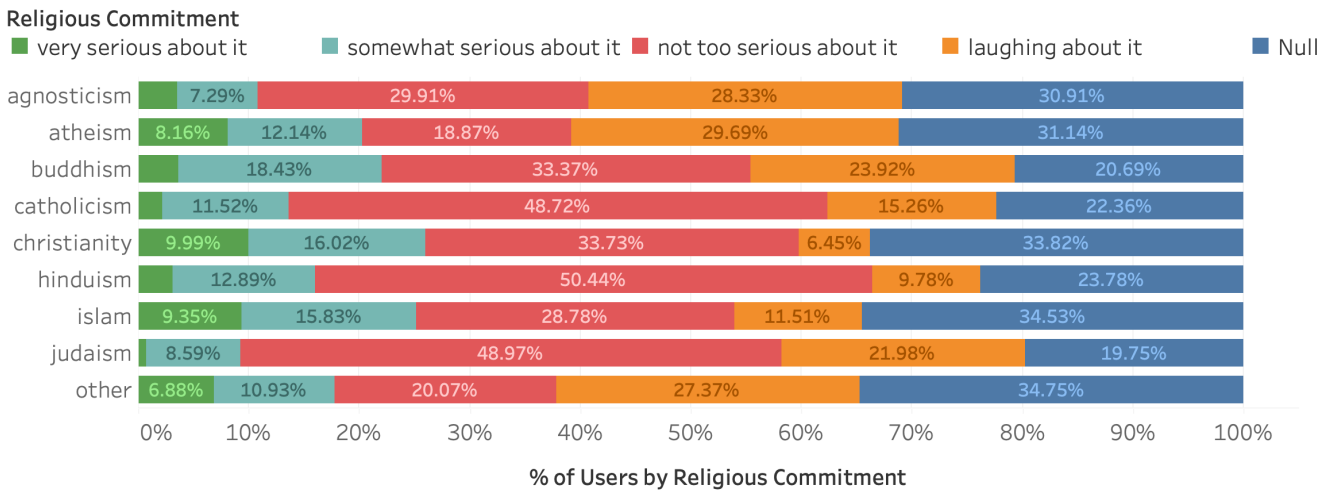


Figure 8.16: Distribution of users' reported commitment level among each religion.

### Sign

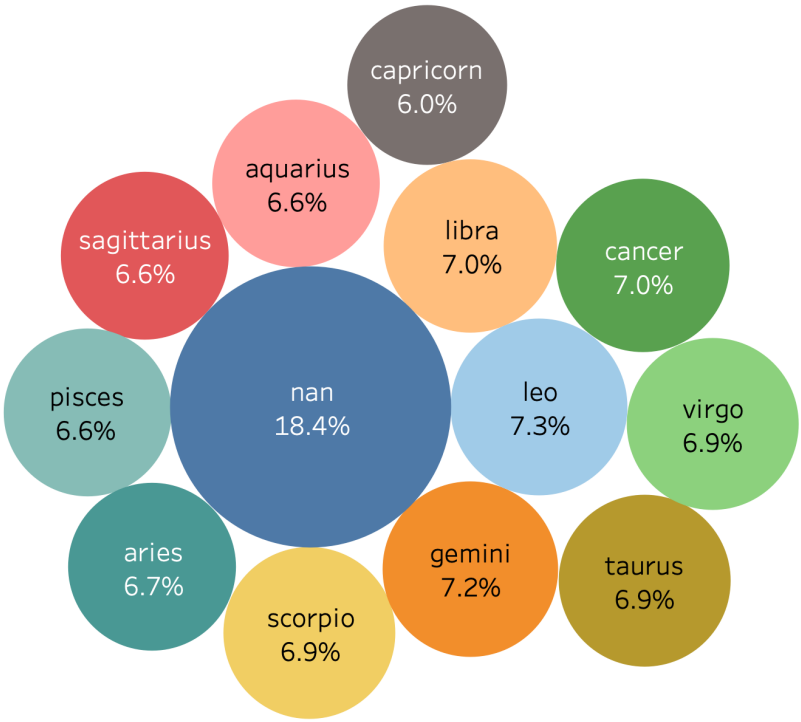


Figure 8.17: Distribution of the different star sign inputs. Area of bubble is proportional to the percentage of users.

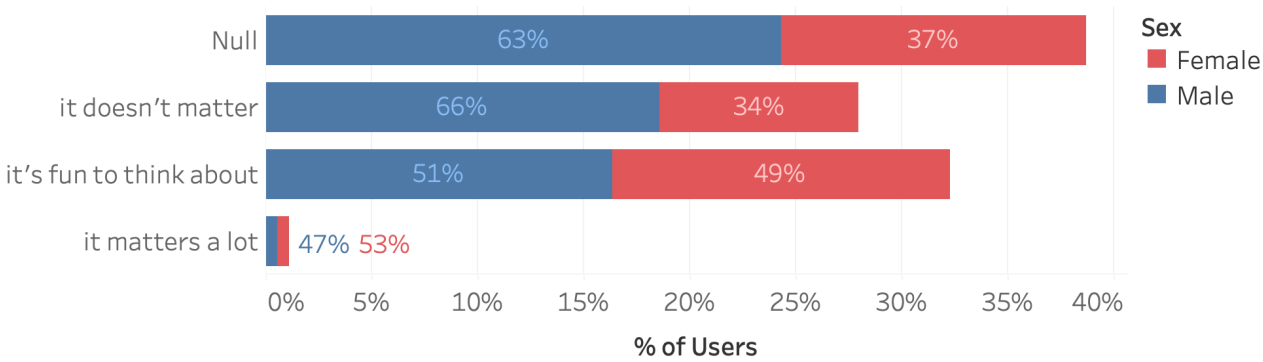


Figure 8.18: Proportion of profiles for each opinion on star signs. Each opinion is divided by sex.

There is a near-uniform split of star signs amongst the users who listed them, seen in Figure 8.17 – as would be expected. Similar as for religion, users were also given an option for how strongly they feel about star signs, depicted in Figure 8.18. The proportion that feels strongly about their importance is very small, but both options that think about them favourably ("it's fun to think about" and "it matters a lot") are favoured more by women than men (taking into account the overall gender split). This echoes previous research indicating that almost double the proportion of women in the US believe in astrology compared to men [62].

### Speaks

The values of this feature are lists of all the languages spoken by each user, and their fluency level for each language in brackets next to it, as seen in Figure 8.19. Therefore it's possible to rearrange the data to look at the breakdown per individual language, rather than per language combination, and this is depicted in Figure 8.20.

It is seen that, naturally for an English-speaking country, English is the highest spoken language. In fact, every single user has listed it. It has a much larger proportion of fluent speakers than any other language option – again unsurprising. It also contains a much higher proportion of null entries for fluency level than the other languages, perhaps indicating that those who do not list their fluency do so because they are natives and therefore don't see it as necessary (someone who is learning a language is more likely to think about their level than someone who is a native speaker).

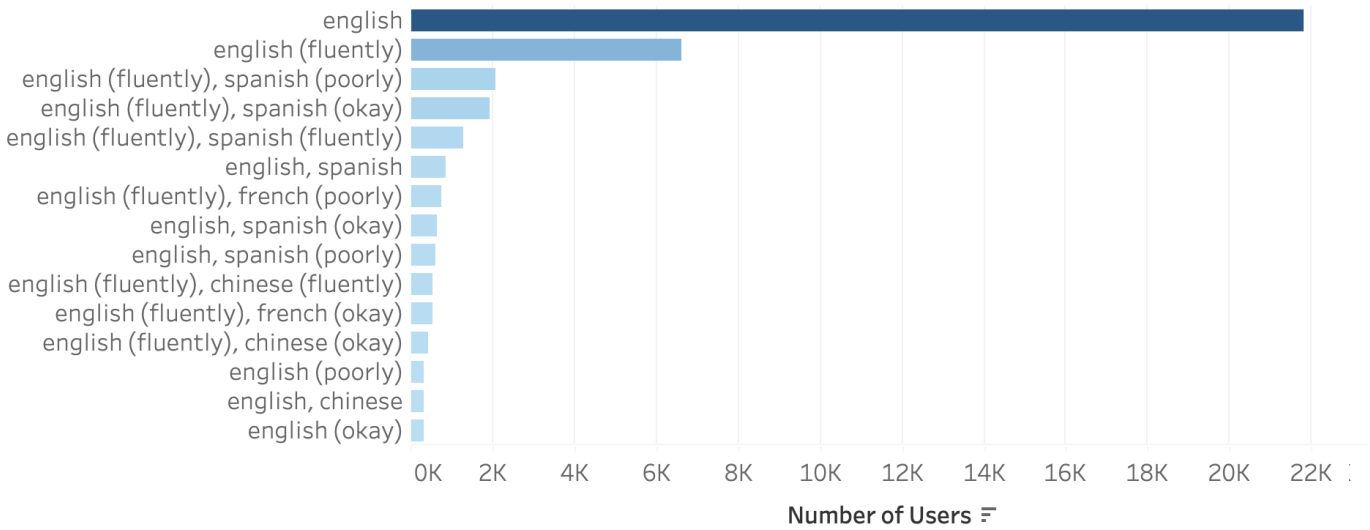


Figure 8.19: 15 most common inputs for the Speaks feature.

Another notable conclusion from this analysis is that the proportion of fluent speakers for both Chinese and Russian is around double that of the three most spoken European languages (31% and 41% vs 19%, 14%, 16% for Spanish, French and German respectively). It's theorised that this is a result of more Americans studying these European languages to a low or medium level at school than study Russian or Chinese, and this theory is supported by previous investigations into language learning in the US [63, 64].

Finally, it's worth paying attention to the inclusion of C++ as the only programming language. The reasons for which no others are listed are unclear, but nonetheless this could be a useful feature in distinguishing profiles.

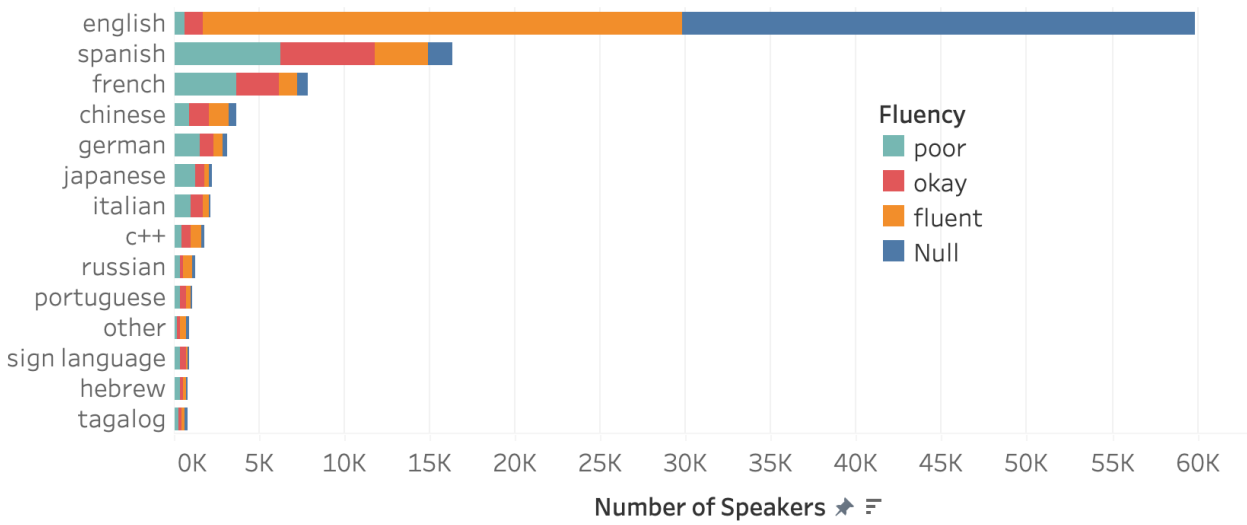


Figure 8.20: Breakdown of speakers of the top 14 languages in the dataset by fluency level.

## Smokes

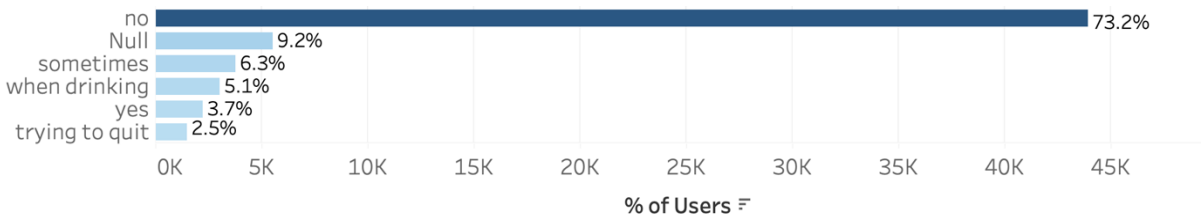


Figure 8.21: Distribution of users listed as each category of smoking.

There is a clear majority of non-smokers amongst the group, with a perhaps surprisingly low percentage of only 3.7% listing themselves as 'yes', although it's possible that a decent percentage of these are smokers but choose not to list it as it is seen as a traditionally unattractive quality.