An Investigation into Predicting Reading Preferences

Maths AI HL Internal Assessment

November 28, 2023

Contents

1 Introduction

Reading is one of those particular hobbies that has stuck with me over the years, despite the changes I have undergone as a person. My fervent passion for this hobby is demonstrated through my purchase and constant use of a Kindle, the second-hand bookshop loyalty cards stashed in my pencil case, and my notebook dedicated to logging words, quotes and the books I have read. Naturally, my taste and reasons for reading have changed over time, and I have attempted to find ways in which I could express this development concretely.

Although I have an intuitive sense of whether I will enjoy a book, it is challenging to reify something as abstract and complex as book preference – there are many contributing factors and they all have varying influences. Although there exists a variety of recommendation engines online, these often rely on majority preference, and in my personal opinion, are not nuanced or representative enough of an individual's personal tastes.

For this reason, the focus of this IA will be to build a simplified, however, more personalised book preference model, where a mixture of both primary and secondary data will be collated to create a model that can predict whether I will enjoy a book based on my past read books.

The more technical objective of this IA is to create a graphical representation where my past read books are represented as points; the Euclidian distances between them represent similarity (in terms of the likelihood of enjoying a book), which are then used to form "likelihood of preference" groups.

1.1 Aim

The objectives detailed above can be simplified into the following aim:

Attempt to graphically represent my books based on factors that personally determine a book's "quality" and categorize these into "likely to be enjoyed" and "unlikely to be enjoyed" groups.

2 Dataset

Overall, twelve factors were determined to be the characteristics I personally found were important in determining whether I would enjoy a book. This information (primary and secondary) was collated and presented here below:

Primary Data	**Secondary Data**
1. Book Title	4. Year Published
2. Book Author	5. Average Rating
3. Personal Rating	6. Number of Votes
	7. Number of Pages
	8. Dark %
	9. Challenging %
	10. Reflective %
	11. Sad %
	12. Tense %

Table 1: Tabled Primary & Secondary Data

The primary data comes from a personal book log I have kept for the past 1.5 years. It includes the **book titles**, their respective **authors**, and my **personal rating** of each book.

The secondary data was collected from Goodreads and TheStoryGraph – two popular websites used to track books read and obtain recommendations. From Goodreads, the **year published, number of pages, average rating** as well as the **number of users** that provided that rating. As for TheStoryGraph, the website's reading analysis function was utilized to indicate my top read and preferred moods/themes in books. The top five were selected and used as part of the model: **Dark, challenging, reflective, tense**, and **sad**. For each book on the site, users can submit a review where they rate certain moods' presence in a book. For instance, a book dealing with themes of remorse and grief would have high mood percentages in "sad", "dark", and "tense". For each book in the log, the mood percentages for the 5 mentioned moods were obtained.

Overall **43 book entries** are used to create the model. A sample snippet of the tabled data is presented here below:

	title	author	year	average rating	personal rating	number of pages	number of votes	dark	challenging	reflective	tense	sad
0	Beneath the Wheel	Hermann Hesse	1909	3.87	3.75	192	1029	0.54	0.29	0.80	0.11	0.73
1	A Grain of Wheat	Ngũgĩ wa Thiong'o	1967	3.87	3.00	247	5557	0.41	0.64	0.71	0.35	0.41
2	Milk Fed	Melissa Broder	2021	3.58	2.50	304	47350	0.35	0.25	0.57	0.12	0.25
3	Down & Out in Paris & London	George Orwell	1933	4.09	3.00	213	84225	0.38	0.22	0.76	0.05	0.29
4	The Invisible Man	H.G. Wells	1897	3.64	3.00	192	186223	0.53	0.12	0.11	0.44	0.06

Figure 1: Raw Data Snippet

3 Methodology

Math Employed - Linear combinations, covariance matrices, eigenvectors, eigenvalues, Voronoi diagrams, PCA, KMeans clustering.

3.1 Technology

Due to the dataset's relatively large size, it would be impractical and inefficient to carry out on every individual entry the calculations needed to address this IA's aims. Hence, *Python 3.12* is used alongside *Jupyter Notebooks* to facilitate this process. The following 3rd party external libraries were also used as part of generating the graphics and performing specific calculations: *Sklearn, Numpy, Matplolib* and *Pandas*. Throughout this IA, comments explaining the code snippets will be denoted by the # preceding the explanations.

3.2 Procedure Overview

Modelling Steps

1. Load, organise data into tables and standardise.
2. Perform PCA (Principal Component Analysis) with $n = 2$.
3. Scatter plot the 2 principal components.
4. Determine outlying points.
5. Scatter plot the 2 principal components again, without the outliers.
6. Visually analyze book scatterplot to estimate book distribution and groups.
7. Use KMeans clustering to generate cluster centre points.
8. Plot cluster centre points.
9. Construct a Voronoi diagram to form category cells.

4 Steps

4.1 Step 1: Load, organise data into tables and standardise

All the primary and secondary data were manually collated and stored in a CSV file. This was then loaded into the Python file to yield the table shown in the *Dataset* section.

As can be seen from the table output, there is a mixture of quantitative and qualitative data. Furthermore, the numerical data varies in units and magnitudes, which will affect PCA later on, as it works based on the data's variability, so leaving the data unprocessed will yield problems later on. This issue is addressed by centring and standardising the data. This involves making the mean of the data 0, and the standard deviation 1.

Firstly, all the numerical data from the dataset is extracted and stored separately (10 factors overall). For each factor in the dataset, all the values were taken and their mean and standard deviation were calculated. The mean was then subtracted from each value and was then divided by the standard deviation. The programmed process and output table are presented here below:

```
# Standardising data (calculating and applying z-scores for each value)


# Storing numerical and qualitative data into separate lists
numerical_data = data[['year', 'average rating', 'personal rating', 'number of pages', 'number of votes', 'dark',
'challenging', 'reflective', 'tense', 'sad']]
qualitative_data = data[['title', 'author']]
scaled_data = numerical_data


# For each column, the mean and standard deviations are calculated, in order to find the z scores for each
for column in scaled_data:
    values = scaled_data[column]
    mean = sum(values / len(values))
    standard_dev = np.std(values)

    for i in range(0, len(scaled_data[column])):
        scaled_data[column][i] = (scaled_data[column][i] - mean) / standard_dev
```

Figure 2: Code for standardisation of data

	year	average rating	personal rating	number of pages	number of votes	dark	challenging	reflective	tense	sad
0	-1.768311	-0.148125	0.320340	-0.683910	-0.489138	-0.003234	0.111585	1.161723	-0.857035	1.957224
1	-0.264674	-0.148125	-0.296434	-0.206082	-0.484213	-0.364739	2.019941	0.863390	0.072020	0.580363
2	1.135264	-0.927501	-0.707616	0.289121	-0.438762	-0.531588	-0.106513	0.399318	-0.818324	-0.108068
3	-1.146117	0.443126	-0.296434	-0.501466	-0.398658	-0.448164	-0.270086	1.029130	-1.089298	0.064040
4	-2.079409	-0.766251	-0.296434	-0.683910	-0.287731	-0.031042	-0.815331	-1.125491	0.420415	-0.925579

Figure 3: Snippet of resulting scaled data table

4.2 Step 2: Perform PCA with n = 2

PCA (Principal Component Analysis) is a dimensionality reduction technique. The first aim of this IA involves graphically presenting the books, however, currently, each book is associated with 10 different variables. It can be said that the dataset has 10 dimensions. Since humans are only able to visually process a maximum of 3 dimensions, graphing 10 different variables would prove to be difficult to understand. PCA addresses this problem by **reducing the number of variables while preserving as much data as possible**. [1] The method in which this process is carried out is as follows:

1. Identify the number of dimensions to reduce the data to (2 or 3 for visualisation). For the purposes of this IA, 2 dimensions will be chosen.

2. This technique involves examining how variables relate to each other which often includes additional redundant information which can be taken out to reduce the dataset's dimensionality. Firstly, to identify correlations between the variables, a covariance matrix is calculated. This is a square matrix, where each entry is the covariance of two variables. Overall, this matrix will contain all the covariances of all the possible variable combinations from the dataset.

3. To obtain the 2 dimensions that are then plotted, the eigenvectors and eigenvalues are computed from that previously computed covariance matrix. This yields the **principal components** which are then graphed. These in themselves do not hold real meaning as they do not represent a particular unit or magnitude. Rather, these are **linear combinations** of the original data.

The programmed process and principal component coordinates obtained for each book are presented below:

```
# Choosing the number of dimensions to be reduced down to
pca = PCA(n_components=2)

# Fitting PCA to scaled data
principal_components = pca.fit_transform(scaled_data)

# For each row of data, the 2 principal components were obtained
principal_df = pd.DataFrame(data = principal_components, columns = ['pc 1', 'pc
2'])
```

Figure 4: Code for PCA

	pc 1	pc 2
0	-1.697147	-1.540824
1	-1.086245	-1.232695
2	0.537538	0.565115
3	-1.002464	0.150641
4	1.199924	-0.062672

Figure 5: Snippet of resulting principal components table

4.3 Step 3: Scatter plot the 2 principal components

The obtained principal component coordinates for each book were then plotted. The two plots, one with and one without the labelled book titles are presented below:

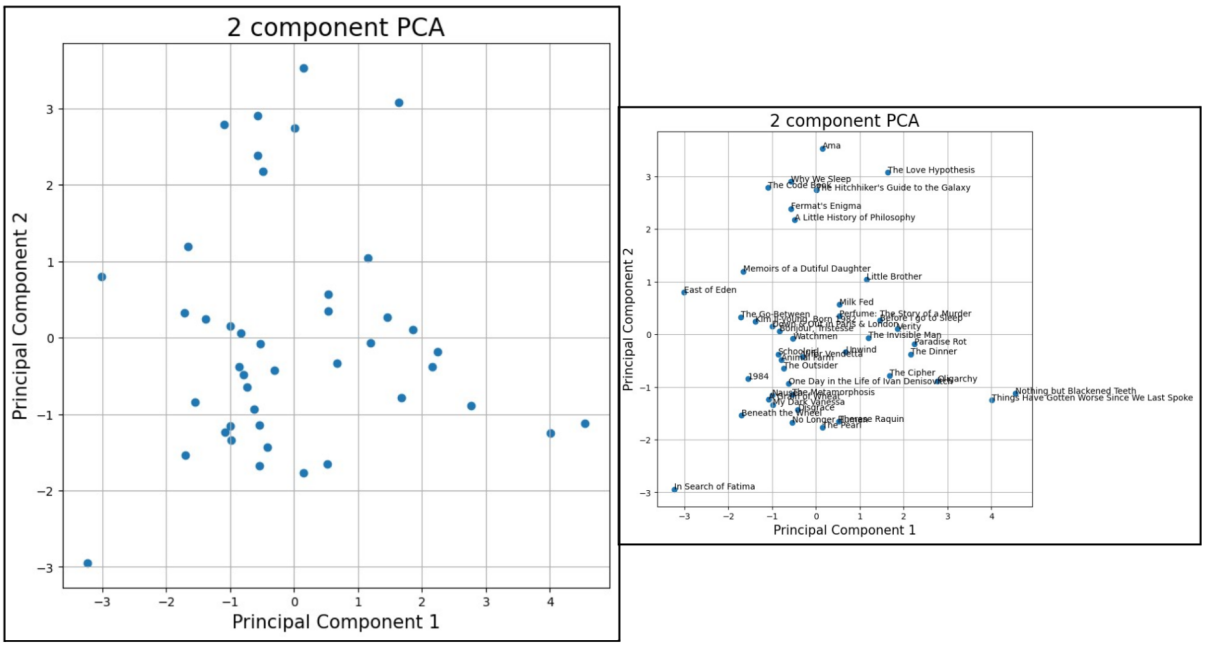

Figure 6: PCA unlabelled & labelled plots

4.4 Step 4: Determine outlying points

Visually, from the plot above, it can be seen that certain points (such as "In Search of Fatima") are generally much further away from the rest of the points. Thus, an outlier test was performed to confirm this. Since the year published, the number of pages, number of votes, average rating, and my personal rating are constants, only the outliers for the mood percentages are calculated. The interquartile range was calculated for the 5 moods and the outliers identified by the books that deviated from that range. The programmed process, as well as the identified outliers, are presented below:

```
# Determine outliers (IQR)
def find_outliers(df):
    q1 = df.quantile(0.25) # Lower quartile
    q3 = df.quantile(0.75) # Upper quartile
    iqr = q3-q1

    outliers = df[((df < (q1 - 1.5 * iqr)) | (df > (q3 + 1.5 * iqr)))]
    return outliers

# Calculate overall genre compatibility score
genre_scores = []

for dark, challenging, reflective, tense, sad in zip(numerical_data['dark'], numerical_data['challenging'],
numerical_data['reflective'], numerical_data['tense'], numerical_data['sad']):
    score = (dark + challenging + reflective + tense + sad) / 5
    genre_scores.append(score)

genre_scores = pd.DataFrame(genre_scores, columns=['score'])
outliers = find_outliers(genre_scores).dropna()
```

Figure 7: Code for determining outliers

	score
19	-1.386242
32	2.001958
33	-1.318024
38	-1.308381

Figure 8: Resulting identified outliers table

4.5 Step 5: Scatter plot the 2 PCs without outliers

The 4 books identified to be outliers are then omitted from the dataset and replotted. Below is presented the programmed process, the "before" scatterplot with outliers highlighted, and the "after" graph with outliers omitted:

```
# Prune data (taking out outliers from dataset)

pruned_principal_df = principal_df.copy()
pruned_qualitative_data = qualitative_data.copy()

for i, row in outliers.iterrows():
    pruned_principal_df = pruned_principal_df.drop(i)
    pruned_qualitative_data = pruned_qualitative_data.drop(i)

pruned_qualitative_data =
pruned_qualitative_data.reset_index(drop=True)
```

Figure 9: Code for pruning data

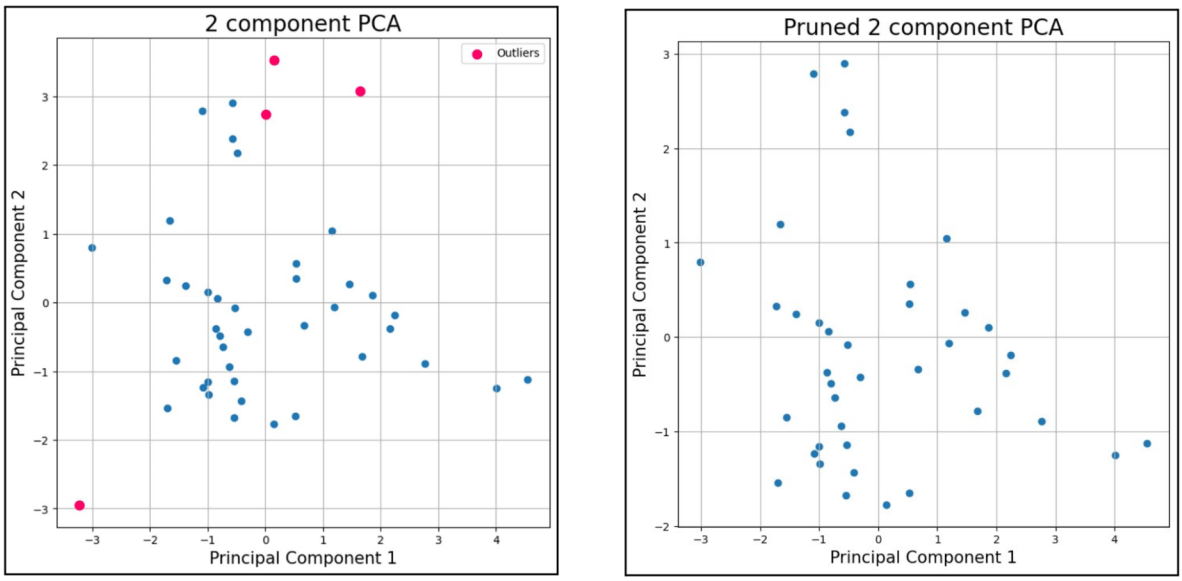

Figure 10: Resulting before & after PCA plots

4.6 Step 6: Visually analyze book scatterplot to estimate book distribution and groups

Overall, after pruning the data, it is remarke that the points tend to cluster around certain areas. The final aim of the model is to classify books into ones that are "likely to be enjoyed" and "unlikely to be enjoyed" by me based on the factors I personally found influenced whether I would gravitate towards a book or not. Hence, it can be assumed that overall, the book points should fall into those two categories.

Visually examining the plot, however, it is noticed that there seems to be an additional point cluster towards the top. Referencing back to the book title labelled plot, manual estimates were made as to which category the books would fall into. The general areas are then highlighted, and the final image is presented here below:

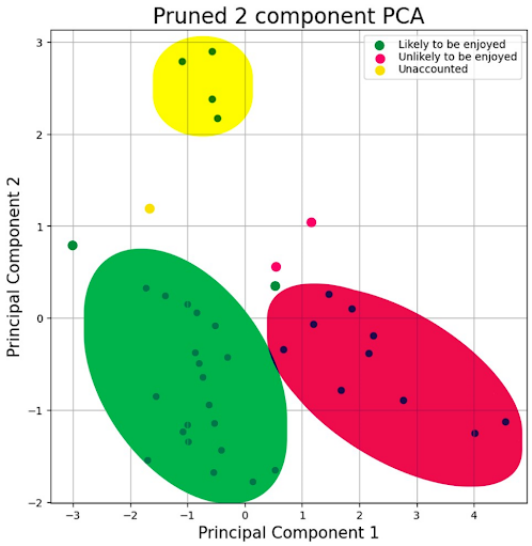

Figure 11: Highlighted estimated cluster regions for pruned PCA plot

4.7 Step 7: Use KMeans clustering to generate cluster centre points

This visual trend of points grouping, or clustering about specific points incites the use of a clustering algorithm such as KMeans to identify those coordinates. [3] This process is as follows:

1. Firstly, the number of clusters (k) is chosen. Based on the visual analysis, k is determined to be 3.
2. Three random coordinates are then generated and plotted. All the points are assigned to one of these cluster centres based on which one is the closest.
3. The cluster centre coordinates are re-computed and all the points re-assigned.
4. This process is repeated again and again until the following criteria are met:

   (a) The new re-computed centre coordinates do not change.
   (b) The points remain assigned to the same centre point.

These steps were applied to the dataset to yield the following centrepoints:

```
# Estimate cluster centers

kmeans = KMeans(n_clusters=3, init='k-means++')
kmeans.fit(pruned_principal_df)
cluster_centers = pd.DataFrame(kmeans.cluster_centers_, columns=['x',
'y'])
```

Figure 12: Code for KMeans

	x	y
0	-0.798506	-0.797883
1	1.909366	-0.206786
2	-1.234391	2.040684

Figure 13: Calculated KMeans cluster centre coordinates

4.8 Step 8: Plot cluster centre points

The obtained coordinates of the cluster centre points from the previous step were then plotted and labelled with their appropriate category titles:

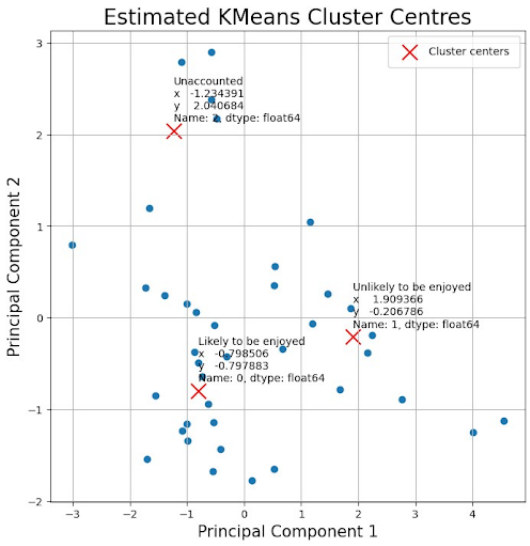

Figure 14: Plotted KMeans cluster centres

4.9 Step 9: Construct a Voronoi diagram to form category cells

As opposed to manually iterating through each point, determining its Euclidian distance to each centrepoint, and evaluating which category the point should be assigned to based on which is closest, a Voronoi diagram can be constructed. In the context of this IA, the centrepoints represent the Voronoi sites. Firstly the line intersections between the 3 points are calculated and plotted. The programmed process and output graph is presented here below:

```
# Calculate line intersections

def line(c1, c2):
    m = (c2[1] - c1[1]) / (c2[0] - c1[0])
    c = c2[1] - (m * c2[0])
    return [m, c]

cluster_coords = cluster_centers.to_numpy()
a, b, c = cluster_coords[0], cluster_coords[1], cluster_coords[2]
lines = [line(a,b), line(a,c), line(b,c)]

# Calculate perpendicular bisectors

def perpend_bisector(c1, c2, m):
    midpoint = [(c2[0] + c1[0]) / 2, (c2[1] + c1[1]) / 2]
    new_m = -1 / m
    new_c = midpoint[1] - (new_m * midpoint[0])
    return [new_m, new_c]

Perpendicular_bisector_lines = [perpend_bisector(a,b,lines[0][0]), perpend_bisector(a,c,lines[1][0]),
perpend_bisector(b,c,lines[2][0])]

# Calculate vertex of diagram

m1 = perpendicular_bisector_lines[0][0]
m2 = perpendicular_bisector_lines[1][0]
c1 = perpendicular_bisector_lines[0][1]
c2 = perpendicular_bisector_lines[1][1]

vertex_x = (c2 - c1) / (m1 - m2)
vertex_y = (m1 * vertex_x) + c1
```

Figure 15: Code for constructing voronoi diagram

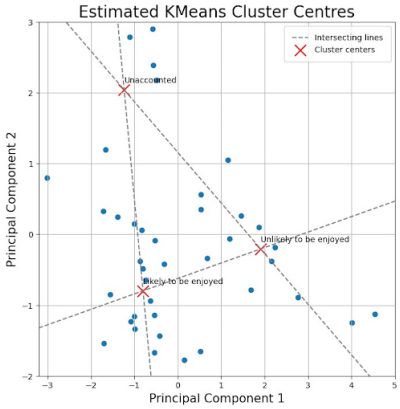

Figure 16: Plotted intersecting lines

After obtaining the line equations of the intersections centrepoint intersections, the perpendicular bisectors are calculated and plotted:

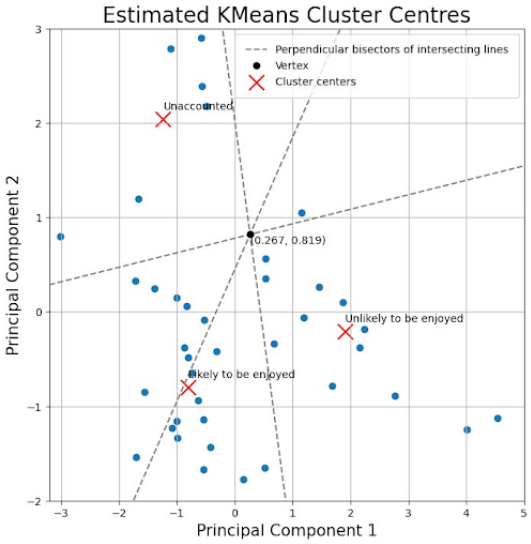

Figure 17: Plotted perpendicular bisectors of intersecting lines

Finally, the unused lines are removed, and the different cells colour-coded for ease of identification. The final model is presented here below:

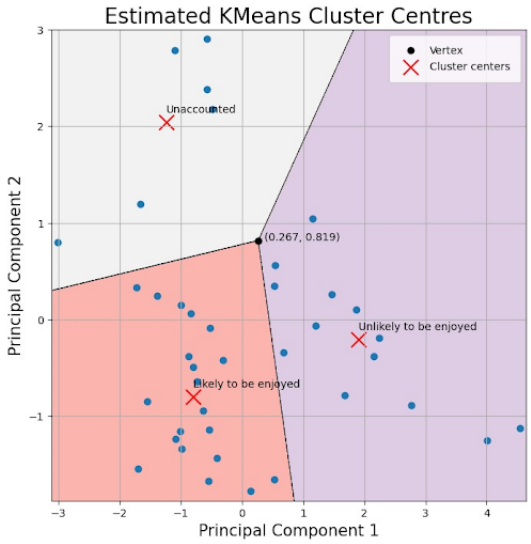

Figure 18: Final voronoi diagram

5 Reflections

- Despite the model being personalized to my preferences, the subjectivity of the choice of factors determining the "quality" of books may signify that the model cannot be used with other individuals and their own booklists, due to the high specificity of the model and its' parameters.

- Following on from this, the model assumes that all the factors have equal "weighting" in the model (meaning that each variable is given the same degree of influence). A modification to improve this would be to introduce an option to vary the weighting of each variable.

- [2] To understand how much information was retained from performing PCA, the explained variance ratio is calculated. This represents the "usefulness" of the principal components. As a general rule, the percentage sum of the principal components should be around 0.8. In total, the obtained variance ratio was found to be 0.52, 0.29 under the ideal value. Since this is not drastically low, this means that linear combinations of the variables can explain the factors' variability only to an extent, so it is important to take this into consideration when determining the accuracy of the model.

- In the visual analysis of the scatter plot, an additional cluster of points was remarked and formed the "unaccounted" group. The presence of these points suggests that not all my preference factors were taken into account. Upon closer examination of the book titles, it was noticed that they all were non-fiction books. A modification that can be made to account for this is to add an additional factor: the"informative" mood percentage.

6 Further Research

- The model's accuracy can be tested with new points (recently finished books). The same process detailed in the procedure is applied to the points to yield principal components, which are then plotted onto the Voronoi diagram.

- Since the outcome of whether the book is enjoyed is known, the model's accuracy can be determined by whether the points are plotted in the appropriate cells. This can be further extended by determining the Euclidian distance between the test data points and its' corresponding group cluster centre point. As a general rule, the larger the distance between the two, the less accurate the model.

- As the model was simplified, it assumed equal weighting or "influence" of all the utilised factors on book preference. However, as was mentioned in the introduction, different factors hold varying importance for different individuals. Hence, it would prove interesting to explore how controlling the magnitude of the "influence" of certain factors on the model affects its' predictive accuracy.

7 Conclusion

In conclusion, a model was constructed to address the aims detailed in the introduction. The steps which were followed were explained in terms of how they related to the aim, and how they would be carried out. Reflections were made on the weaknesses of the choices made, and modifications were suggested to improve on them. Finally, suggestions for further investigation and research are given that may improve the model.

References

[1] URL: https://builtin.com/data-science/step-step-explanation-principal-component-analysis.

[2] Isabella Lindgren. *Dealing with highly dimensional data using principal component analysis (PCA)*. Apr. 2020. URL: https://towardsdatascience.com/dealing-with-highly-dimensional-data-using-principal-component-analysis-pca-fea1ca817fe6#:~:text=The%20explained%20variance%20ratio%20is,or%2080%25%20to%20avoid%20overfitting.

[3] Pulkit Sharma. *The Ultimate Guide to K-means clustering: Definition, methods and applications*. Nov. 2023. URL: https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/.