Personality Disorder Classification and Evolution

1st Dhrishti Hazari
dhrishti.hazari01@gmail.com

I. Introduction

Medical terminology evolves at a turtle-like pace over time as well as location. This gradual change is minute and often undetectable to those who are not well versed in the linguistic history of such niche topics, both inside and out of that profession. Studying and understanding the evolution of language commonly associated with personality disorders can assist with detecting biases and stigmas, detecting misinformation, and even scientific communication by identifying outdated terminology.

The goal of this project is to identify terminology commonly associated with various personality disorders throughout time, unlimited by any person's scope of knowledge. The final output is a classification model capable of categorizing new articles into thematic cluster and predicted year based on historical patterns.

My interpretation of this model is that it helps determine whether a journal article is outdated. By combining these predictions about classification and year, we can assess not just what the article is about, but also how current or outdated it might be.

II. Exploratory Data Analysis (EDA)

There was not much exploratory analysis to be done here. Personally, I have a limited scope of knowledge surrounding personality disorders, and given that many professionals may not be able to understand context written as of 50 years ago, they may not either. I wanted to find a way to allow the journals to do most of the talking, so I decided to allow the Journals to decide potential classifications themselves.

To do this, I decided to parse journals from the Personality Disorders: Theory, Research, and Treatment.12

This journal is available on the Fordham online e-book portal, containing journals published between the years 2009 - 2025, each between 20 - 60 published articles.

While going through them, I noticed that titles in the early years seem to contain terms directly related to the "Classification" of the article while later years use more airy language, but ALL journals had a line labeled "Keywords" followed by a list of terms significant to the specified text. Noted this for future usage.

Another note was that while many files had HTML versions, there were also many PDFs scattered, namely towards the early years. Parsing PDFs means more of a need for cleanup moving forward.

A. Data Storage

All the Journals had to be organized in a way to make my goal achievable. I created a folder for the title of the journal, followed by sub folders of years, and each year folder contained the articles written in that year.

B. Dataframe Set Up

- **Journal name**: Used to store the name of the journal
- **Year**: Used to store the Year the of the article
- **Title**: Used to store the title of the Article
- **file_path**: The path of the folder, described in 1
- **Keywords**: Used to store keywords of the articles, found from within the text and the Title.
- **text**: The entirety of the article text
- **Sequences**: Terms or phrases that may be important to the text, calculated in the Sequential Data Mining section 2
- **lengths**: The number of times any given sequence appears in the text, calculated in the Sequential Data Mining section 2
- **correlations**: Most significant correlations to the keyword terms, calculated in the Association Rule Mining section 2
- **Classification** Final Classification of the article, identified in the Classification section 3

C. Parsing and Cleaning

When first parsing the text and storing in a list (to be used moving forward), I found many errors while parsing the data - I of course ensured the removal of stop words(using an NLP Library), but I found that there were random ”h”’s appended to some words and terms that I would identify as ”Stop Words” appearing throughout. I created a function to cleanse such abnormalities.

D. Lemmentization

After doing much of the code and attempting to stem the words to cut down on excessive finds, I found that stemming the words took away the ability to comprehend the results. Instead, I later ensured that all terms were lemmentized (AKA. cut down on ”-ly” and ”-s”), which improved the chance of finding repeated terms and reduced the number of ”important terms”. Once lemmentized, I was even able to increase the Minimum Support threshold to gather relevant data using sequential data mining2!

III. Finding Classifications

A. Identifying important phrases and keywords

An important step to classifying terminology is to extract important terms AND phrases. I used sequential data mining to pull the most repeated terms, using these terms as the most probable important terms.

1) Sequential Data Mining: I created a variation of sequential data mining. After parsing, cleaning, and splitting terms for EACH text, I set an initial min support threshold of 0.7% after testing multiple options. It uses this to parse all the terms that appear at least 0.7% of the time in a given text.

THEN, every time the length incremented above 2, I reduced the minimum support threshold by another 50%. This is because I found that important terms in Personality disorders can be cut down and referred to with shorter names, almost like nicknames - for example, ”Borderline Personality Disorder” can be referred to as simply ”borderline” or ”BPD”, which means there is a greater probability of identifying single term sequences over multiple term phrases. I wanted to capture these patterns as a precaution of my being unaware of potential references.

During this, I ensured I found PHRASES as opposed to random terms that appeared in a specific order. By this I mean that the terms had to be NEXT to each other to be considered a phrase.

Final phrases were saved in the ”Sequence” field, with the number of times each term appeared being stored in the ”lengths” column.

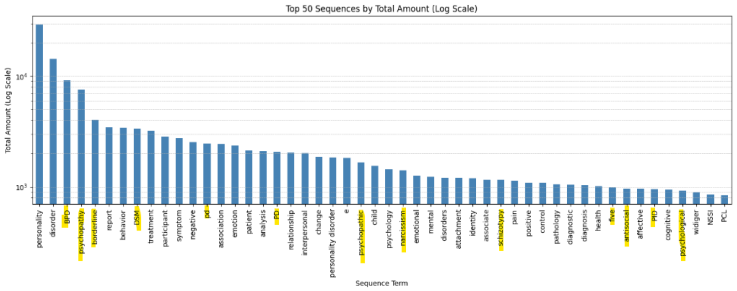

Fig. 1. Top 50 phrases found with Sequential Data Mining

I also used this method to parse ALL titles, first combining them and splitting into a list, parsing and cleaning before using a minimum support threshold of 0.2%. Once important terms were found, they were added to the ”Keyword” column as long as the term/phrase appeared in the ”Title” column. The keywords (mentioned in the EDM section 1) were also extracted from the ”Text” and appended to the ”Keyword” column.

2) Association Rule Mining: Why did I use association rule mining after sequential? It is part of the classification process. After performing sequential data mining, I found that there were way too many terms found to be able to just choose one from EACH list of sequences. Using association rule mining narrows down the options as it is created based on the ENTIRE dataset as opposed to the individual entries.

In order to set up the table for association rule mining, I created a new data frame. After first creating a 2-D array of appended keywords and sequences PER entry, I turned it into a Data Frame where ALL keywords and Sequences became Columns while the entries were ”True” or ”False”.

	AAQII	AB	ABCD	ADHD	AGGR	AIC	AIDA	AMPD	ANS	...	young	youth	youth psychopathology	z	zanarini	zeiglerh re
0	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
1	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
2	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
3	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
4	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
418	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
419	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
420	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
421	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal
422	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	Fal

423 rows × 2296 columns

Fig. 2. Association Rule Mining DataFrame Set up

Once this was completed, I used the apriori algorithm to generate rules with a min confidence threshold of 0.6 (Chosen after testing multiple thresholds)

Now going back to the original dataframe 1, for ANY antescent that appeared in the ”Keyword” column, its consequent was stored in the ”Correlation” column.

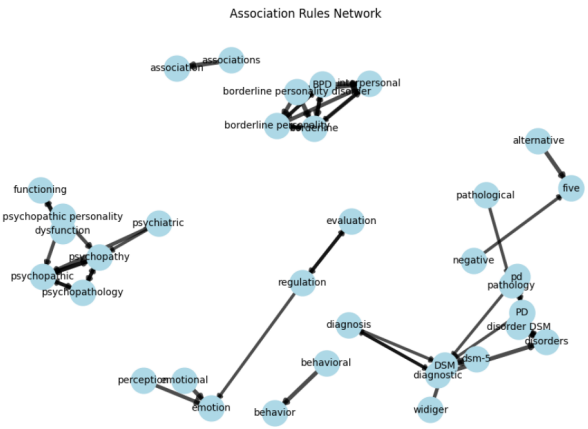

Fig. 3. Top 50 association rules sorted by lift

B. Classification

Now that I have important terms, phrases, keywords, AND correlations,it was time to do the hard part: classification.

If any ”Correlation” term appeared in either the ”Keyword” section OR the ”Sequence” section, it was saved as a possible classification. The possible classification the appeared the MOST in the text (found via. ”lengths” column), it was chosen as a final Classification.

I had to account for the chance that at times the correlation terms would NOT appear in the ”lengths” of an entry. In this case, the algorithm had already run once and while it classified many entries, it also generated a list of possible classifications. The list was collected and repeatedly randomized as the unclassified entries in the original data frame selected the first overlapping value between its ”correlations” and the now potential classifications. The randomization prevented the classifications from leaning too heavily towards one personality disorder. There were a few options manually added as the code was run repeatedly with varied outcomes, and I decided to add those outcomes as a possibility.

The final Classification list included: ’dementia’, ’borderline’, ’psychopathy’, ’DSM’, ’five’, ’BPD’, ’schizophrenia’, ’psychopathology’, ’neuroticism’, ’narcissistic’, ’antisocial’, ’psychopathic’, ’avoidant’ ’schizotypy’

Note: There are many more possible classifications, the list changes based on the run since there is a bit of a random factor included

Although not obvious, a couple of these classifications overlap with one another, such as ”BPD” and ”Borderline”, ”Five” and ”neuroticism” (five factor model includes neuroticism), and ”Psychopathy” and ”Psychopathology”. I manually cleaned these Classifications to get a final distribution of:

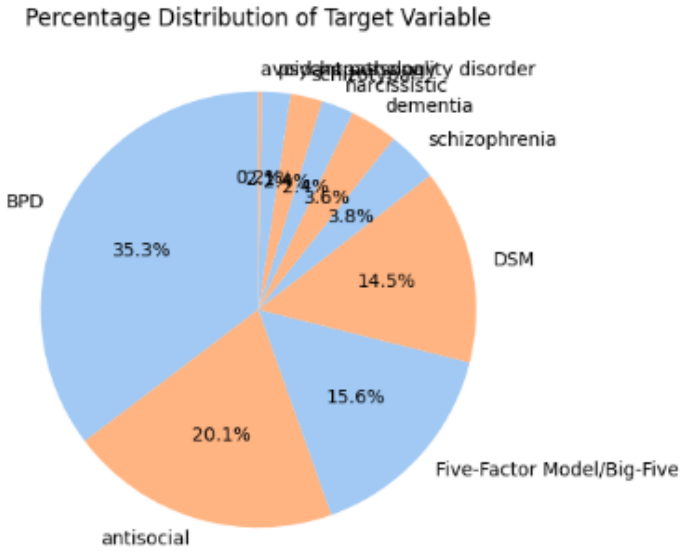

Fig. 4. Distribution of classifications after clean up

C. Notes

When already half way through my project, I came across a method called TF_IDF vectorization. This method allows you to convert ALL terms into vectors and use these to identify important terminology. When I briefly attempted to use it, I found that running this took up almost two hours without completion and that I would be unable to extract phrases, so I decided to continue and complete my current approach - however, I would like to figure out how to utilize this feature in the future.

IV. Analysis of Newly Created Data

A. Balancing

It is clear that the data is very unbalanced based on the above figure 3. This is most likely a result of the fact that terms like ”Schitzotypal” and ”Antisocial” were only studied in the recent years while ”narcissistic” has very limited studies, limiting the amount of data we have.

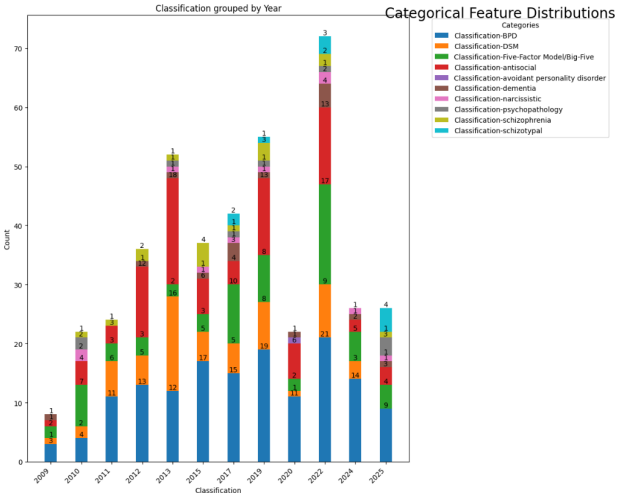

Fig. 5. Classification By Year

B. PCA

I created a PCA to better understand distribution of sequences among the Classifications. This was done by finding sequence embeddings and using variance as a measure of similarity. The below images help describe the PCA.

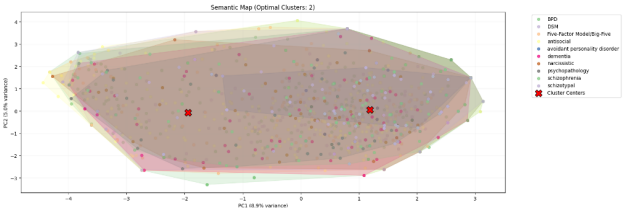

Fig. 6. PCA colored by Classification

```
Cluster-Classification Overlap:
Classification  BPD  DSM  Five-Factor Model/Big-Five  antisocial  \
Cluster
0                 0    0                           0           0
1                 1    1                           1           1

Classification  avoidant personality disorder  dementia  narcissistic  \
Cluster
0                                           0         1             0
1                                           1         0             1

Classification  psychopathology  schizophrenia  schizotypal
Cluster
0                             0              1            0
1                             1              0            1
```

Fig. 7. Overlap in PCA

As observed, there is clearly a LARGE amount of overlap in terminology used between classifications. My goal going forward is to cut down on that by creating some sort of probability weight-age for the terms..

C. Sequence Evolution

In Order to analyze the evolution of terms over time, I created a DataFrame utilizing Cosine similarity to find the sequences in the following year that is most similar to the sequences present in the current year. An example below:

	classification	sequence	year	next_year	best_match	similarity	is_same_sequence	is_semantic_shift
0	five	psychopathic	2009	2010	psychopathic	1.000000	True	False
1	five	temperament	2009	2010	males	0.753936	False	False
2	five	diagnostic	2009	2010	diagnostic	1.000000	True	False
3	five	electro dermal	2009	2010	functioning	0.389512	False	True
4	five	personality disorder	2009	2010	diagnosis personality	0.843102	False	False
...	...	...	...	...	...	...	...	...
56305	neuroticism	peer	2024	2025	orientation	1.000000	False	False
56306	neuroticism	fluctuations	2024	2025	variability	1.000000	False	False
56307	neuroticism	health	2024	2025	disorders	1.000000	False	False
56308	neuroticism	symptom	2024	2025	disorder	1.000000	False	False
56309	neuroticism	asian	2024	2025	girls	1.000000	False	False

56310 rows × 8 columns

Fig. 8. Sequence Evolution

I was planning on creating a directed graph to make the above results more readable. Unfortunately I was unable to do so in time, so it is now a ”To-Do”

However, I did graph the frequency of terms as used over the years (terms were grouped by cosine similarity). A few interesting results are displayed below:

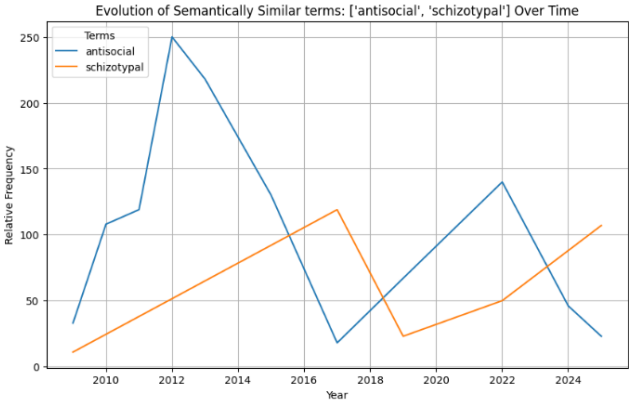

Fig. 9. Antisocial and schizotypal seem to be used in higher frequency at opposite times

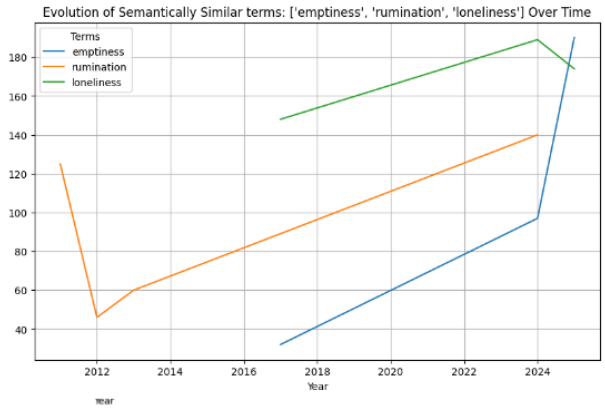

Fig. 10. Loneliness Epidemic is on the rise

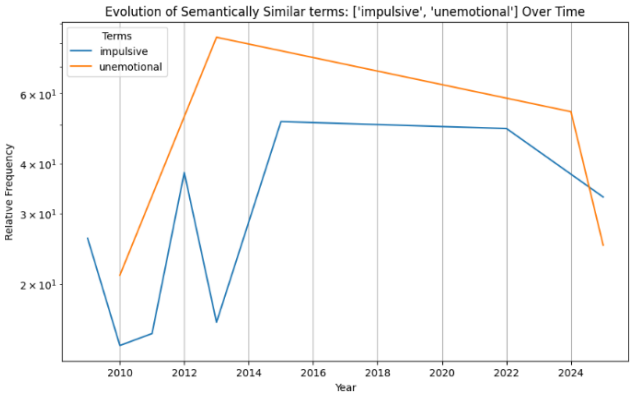

Fig. 11. Impulsiveness and Unemotional seem to follow similar patterns

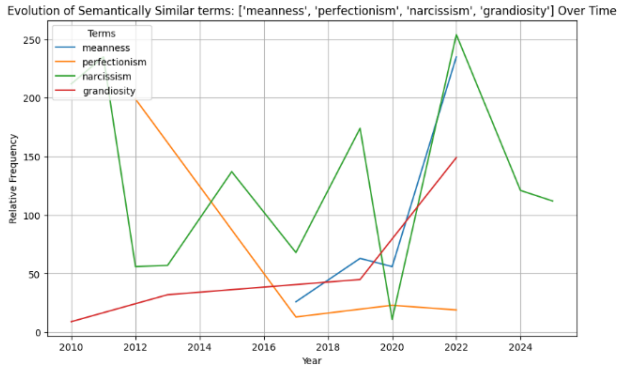

Fig. 12. Meanness and Narcissism rise while perfectionism goes down

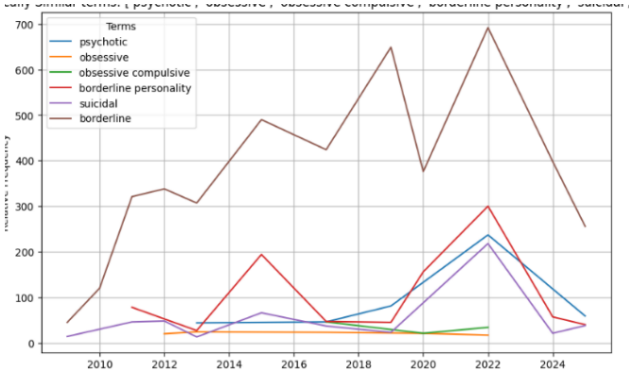

Fig. 13. Borderline and Suicidal follow a similar path

These figures can help evaluate the final findings. For example, an article about Schitzotypy is less likely to get classified as such in the year

V. Feature Evolutions

Now that we have the DataFrame filled and evaluated, we start on turning this into a trainable DataFrame.

A. DataFrame Creation

My first idea was in relation to having the sequences converted into columns while the entries would be binary values, and the prediction columns would be ”Classification” and ”Year”. However, I wanted the ability to apply weights to the features, so I dropped this idea.

Instead, I decided to use confidence as values and lift as weightage.

To do this, I first found all unique sequences grouped by the ”Classification” and ”Year”. I ensured to accumulate the counts per each and grouped sequence and store them. Once this was done, I proceeded to calculate the confidence and lift of each using the following equations.

$$\text{Confidence} = \frac{P(\text{Class}, \text{Sequence}, \text{Year})}{P(\text{Sequence})} = \frac{\text{Count}(\text{Class}, \text{Sequence}, \text{Year})}{\text{Count}(\text{Sequence})}$$

$$\text{Lift} = \frac{P(\text{Sequence} \mid \text{Class}, \text{Year})}{P(\text{Sequence})} = \frac{\frac{\text{Count}(\text{Class}, \text{Sequence}, \text{Year})}{\text{Count}(\text{Class}, \text{Year})}}{\frac{\text{Count}(\text{Sequence})}{\text{Total Count}}}$$

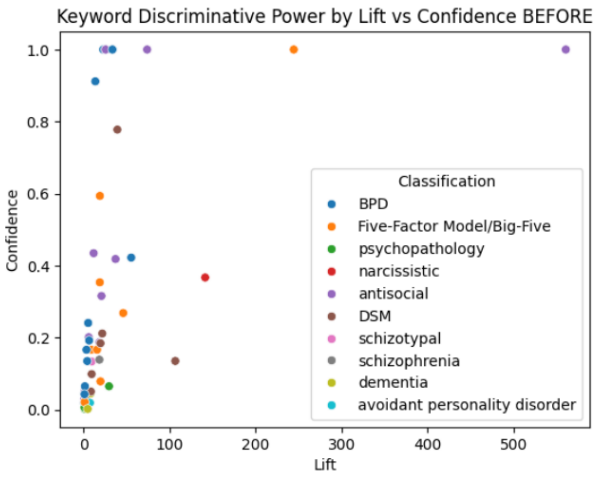

Fig. 14. 50 random terms

After plotting, it is observed that there are limited terms with both high confidence AND high lift thresholds. Fortunately, we have an exponential number of features, which means that having limited quality sequences is expected an welcome for the sake of feature selection! I filtered out sequences that had a confidence below 0.2 or lifts below 10 (values chosen based on observation of scatter plot).

	Classification	Year	Keyword	Count	P_keyword_given_class	P_keyword_global	Lift	Confidence
3	dementia	2009	semi	64	0.081529	0.000294	277.461221	0.752941
10	BPD	2009	questionnaire	13	0.008312	0.000218	38.166027	0.206349
11	BPD	2009	relative	16	0.010230	0.000194	52.845269	0.285714
14	BPD	2009	livesley	13	0.008312	0.000045	184.958440	1.000000
21	BPD	2009	characteristic	13	0.008312	0.000045	184.958440	1.000000
...	...	...	...	...	...	...	...	...
5192	schizotypal	2025	absorption	55	0.013174	0.000190	69.287425	1.000000
5195	schizotypal	2025	oe	156	0.037365	0.000671	55.715661	0.804124
5196	schizotypal	2025	kerns	31	0.007425	0.000176	42.115886	0.607843
5197	schizotypal	2025	schizotypal thinking	35	0.008383	0.000121	69.287425	1.000000
5198	schizotypal	2025	positive schizotypy	76	0.018204	0.000664	27.426272	0.395833

Fig. 15. DataFrame listing calculated values described

B. Feature Weighing

To make the lift values more interpretable and to avoid the extreme skew caused by high lift scores (ranging from 0 to 500), I applied a $\log_2$(lift) transformation. I had attempted to also scale the lift features with below-par results, most likely because the lift values were so skewed that scaling would render some lift values useless. Using a log transformation in comparison compresses the scale while preserving the relative differences and directionality. Since a lift value of 1 indicates statistical independence, sequences with lift values closer to 1 contribute less to the overall model. Using $\log_2$(lift) as a weighting factor helps emphasize significant patterns while reducing the influence of outliers or noise.

Once the above metrics were calculated, a new and FINAL DatFrame was created. First all possible sequences were transposed to be features. Then, using each entry in the ORIGINAL DataFrame 1, all the sequences present in EACH row had their confidence and lift searched up from the DataFrame5 created above given the classification and year. The entry was 0 if the feature did not appear as a sequence in the current rows entry or the classification*lift if it did. The lift acts as a weight, giving features with a greater lift more importance.

C. Feature Engineering

When starting out, I encountered many issues regarding a couple of labels. Since the data was so limited for classifications such as ”Narcissist” or ”Schitzotypy”. Logistic regression models were achieving a whopping accuracy of 18%, unable to properly run Cross Validation given the limited splits, and my models were failing. To remedy this, I grouped ANY label that had fewer than 10 classifications into a group labeled ”Other”. This was done prior to the train-test split.

The train-test split was then done with a 20% split

ALL X_train and X_test features were scaled based on the X_train data using StandardScalar! This allowed to limit any Bias.

I also attempted to perform SMOTE on training data by combining the ”Classification” and ”Year” as a single object, performing SMOTE, and splitting the object. Unfortunately, due to the limited data, if there were any times in which there was a classification with only one output, SMOTE was not performed.

D. Feature Selection

Given the sheer amount of sequences, the biggest challenge was feature selection. I combined the ”Classification” and ”Year” column for these steps, treating them as one entity before separating them at the end. ALL feature selection was run on X_train to avoid data leakage. The only changes to X_test were done based on dropped columns in X_train to avoid.

- I started off early, when calculating the Confidence and Lift values, as mentioned above in 5. This significantly improved the outcome of the Neural Network - most probably because the features dropped were misleading and introduced noise to the model.
- I attempted to use a variance threshold, which essentially drops features that do not meet a specified

variance threshold. HOWEVER, I found that this more detrimental than helpful because it resulting in removing features that had a significant impact on singular classifications while keeping features that had enormous overlap between all classifications. Lower variance was more harmful than helpful. I decided to alter this to calculate variance PER classification, and THEN apply the variance threshold! This way, I removed features that have high global variance but offered no class separation.

  - Final Variance Threshold:1e-3
  - Features before: 1642
  - Features After: 1249

- I then proceeded to cut down on features that were not highly associated with the Classification column - using the p-score found from using ANOVA. This was another filter to ensure the feature is associated with the target.

  - Final P-Value Threshold: 0.05
  - Features before: 1249
  - Features After: 257

- Originally, features with HIGH correlation (AKA. threshold of around 0.95) to each other were dropped to reduce redundancy. HOWEVER, this seemed to negatively effect the model - the features being dropped may not have only had high correlations to each other, but ALSO to the classifier. That is why I went with the option that would do the least harm - 1! Only features that were essentially exactly the same would be dropped.

  - Final Correlation threshold: 1
  - Features before: 257
  - Features After: 249

- Using SelectKBest, I selected a certain number of top features. After testing, it came to light that using this was more detrimental then helpful. Most likely the above functions were more than enough, and using this feature selection model caused the data to loose important classification metrics, hence I decided not to move forward with this approach.
- I also considered using a PCA for dimensionality reduction. However, the prime number of PCA features was determined to be 2 - much to limiting for the output. The data needed to be EVEN cleaner if there was hope for this method to work.

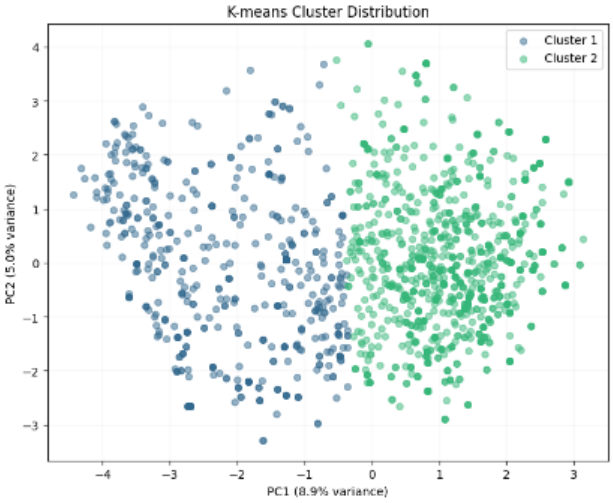

Fig. 16. PCA Cluster groups

VI. Model Trainings

ALL models EXCEPT neural networks use GridSearchCV to ensure the best parameters. GridSearchCV chooses the best model based on the 'f1_weighted" score since the set is imbalanced.

A. Regression

The first part was training models to predict the Target Year using regression.

The year column was converted into ordinal numbers based on year. This was so that order could be maintained.

The Evaluation metrics for regression were:

- Mean Absolute Error
- Mean Squared Error
- Root Mean Squared Error
- R^2 Score
- Cross-Validated R^2 Score

1) Linear Regression: Grid Search CV Parameters:

- 'fit_intercept': [True, False]

Chosen Parameters:

- 'fit_intercept': True

Results:

- Mean Absolute Error: 2.5007
- Mean Squared Error: 9.0015
- Root Mean Squared Error: 3.0003
- R^2 Score: 0.0177
- Cross-Validated R^2 Score: -0.0331 (+/- 0.1951)

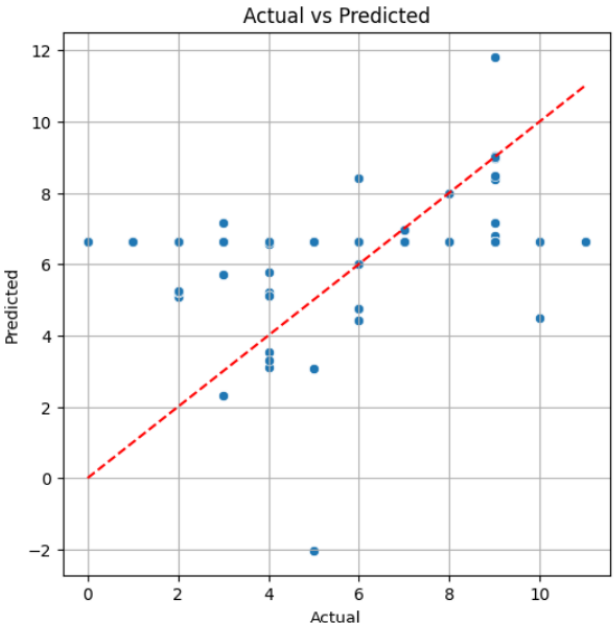

Fig. 17. Linear Regression Actual vs Predicted

Observations:

- WITHOUT using the lift as weights, the mean R^2 errors were below 0, indicating that the model performs worse than simply predicting the mean.
- When utilizing ANY of the other feature selection options, the MSE was 100 and R^2 errors were again below 0, indicating that the models predictions are EXTREMELY off.
- Overall this is off by around 2 years - not the best result

Given these observations, I decided to try fitting a model with a polynomial of degree 2 to see if there were any patterns not being captured. However, the results were even worse than Linear Regression, so I moved on to attempt ridge regression to apply a penalty in hopes that the model would be more stabilized.

2) Ride Regression: Grid Search CV Parameters:

- 'alpha': [0.01, 0.1, 1, 10, 100],
- 'fit_intercept': [True, False],
- 'solver': ['auto', 'svd', 'cholesky', 'lsqr', 'saga']

Chose Parameters:

- 'alpha': 100,
- 'fit_intercept': True,
- 'solver': 'saga'

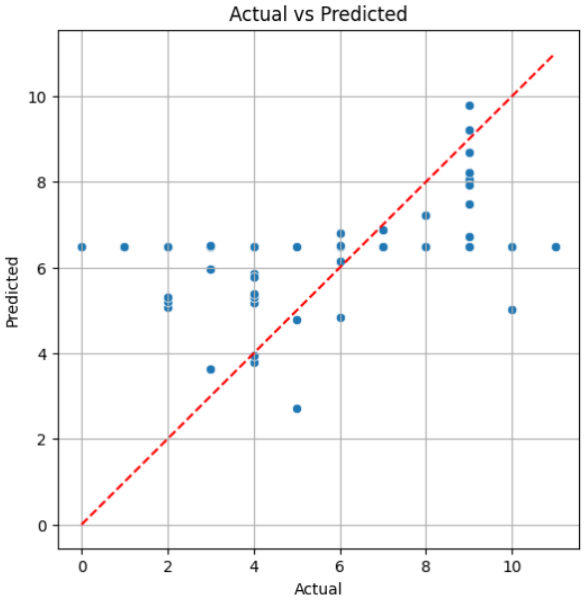

Fig. 18. Ridge Regression Actual vs Predicted

Results:

- Mean Absolute Error: 2.3851
- Mean Squared Error: 8.1020
- Root Mean Squared Error: 2.8464
- R^2 Score: 0.1158
- Cross-Validated R^2 Score: 0.1705 (+/- 0.0637)

Observations:

- The observations are a big improvement on the Linear Regression models given that the model is actually learning
- Unfortunately, the results were still very weak. The variance is most likely still very low, which is causing these lackluster results. The spread of year is less than ideal.

B. Classification

The second part was training models to predict the Target Classification using classification models.

The Classification column was converted to a multi-classification column using a label encoder.

The Evaluation metrics for classification were:

- accuracy
- Classification Report
- Confusion Matrix
- Although I attempted to use Stratified Cross validation, the lack of data once again came in the way of successfully performing this.

1) Logistic Regression: Grid Search CV Parameters:

- 'C': [0.001, 0.01, 0.1, 1, 10, 100]
- 'penalty': ['l1', 'l2']

- 'solver': ['lbfgs']
- 'multi_class': ['multinomial']
- 'class_weight': ['balanced']

Chosen Parameters:

- 'C': 100
- 'penalty': 'l2'
- 'solver': 'lbfgs'
- 'multi_class': 'multinomial'
- 'class_weight': 'balanced'

Results:

TABLE I
CLASSIFICATION REPORT: LOGISTIC REGRESSION

Class	Precision	Recall	F1-Score	Support
BPD	0.54	0.97	0.70	32
DSM	1.00	0.67	0.80	12
Five-Factor Model/Big-Five	0.20	0.09	0.12	11
Other	0.50	0.50	0.50	2
Antisocial	0.92	0.52	0.67	21
Dementia	0.00	0.00	0.00	5
Schizophrenia	0.00	0.00	0.00	2
Accuracy			0.61	85
Macro Average	0.45	0.39	0.40	85
Weighted Average	0.61	0.61	0.57	85

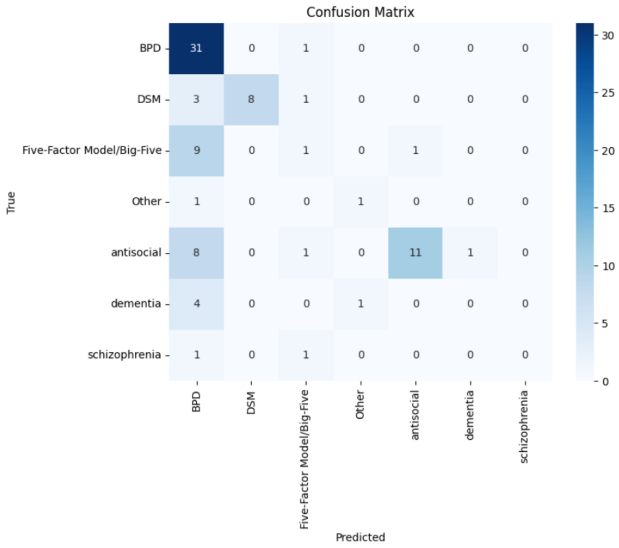

Fig. 19. Logistic Regression Confusion Matrix

Observations:

- When running Logistic regression without feature selection, the results were limiting. the accuracy ranged anywhere between 17% - 18%.
- When altering correlation thresholds, weather to each other or to the classifier, The improvement was minimal, going to between 21% - 33%.
- After more fine tuning, I found the culprit to be the original method of identifying variance thresholds. Once I changed the variance threshold to be BY CLASS rather than OVERALL, logistic regression so EXTREME improvement - the classification accuracy jumped up to 60%

2) XGBoost: Grid Search CV Parameters:

- 'n_estimators': [100, 200, 300]
- 'max_depth': [3,5,7]
- 'learning_rate': [0.01, 0.05, 0.1]
- 'subsample': [0.8,1.0]

Chosen Parameters:

- 'n_estimators': 200
- 'max_depth': 3
- 'learning_rate': 0.05
- 'subsample': 0.8

TABLE II
CLASSIFICATION REPORT: XGBOOST

Class	Precision	Recall	F1-Score	Support
BPD	0.46	1.00	0.63	32
DSM	1.00	0.58	0.74	12
Five-Factor Model/Big-Five	0.00	0.00	0.00	11
Other	0.00	0.00	0.00	2
Antisocial	1.00	0.43	0.60	21
Dementia	0.00	0.00	0.00	5
Schizophrenia	0.00	0.00	0.00	2
Accuracy			0.56	85
Macro Average	0.35	0.29	0.28	85
Weighted Average	0.56	0.56	0.49	85

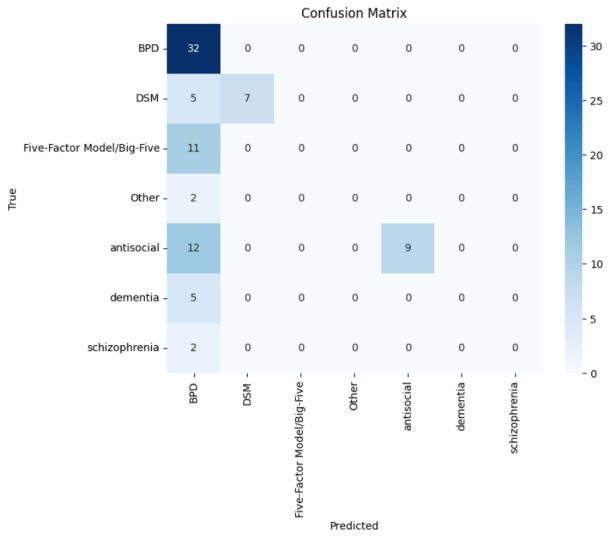

Fig. 20. XGBoost Confusion Matrix

TABLE III
CLASSIFICATION REPORT: RANDOM FOREST

Class	Precision	Recall	F1-Score	Support
BPD	0.51	1.00	0.67	32
DSM	1.00	0.67	0.80	12
Five-Factor Model/Big-Five	0.00	0.00	0.00	11
Other	1.00	0.50	0.67	2
Antisocial	1.00	0.48	0.65	21
Dementia	0.00	0.00	0.00	5
Schizophrenia	0.00	0.00	0.00	2
Accuracy			0.60	85
Macro Average	0.50	0.38	0.40	85
Weighted Average	0.60	0.60	0.54	85

Observations:

- When applying a global variance threshold, the terms were ALL being classified as BPD. This is another instance in which class imbalance is a negative impact.
- When the variance threshold was applied during feature selection, classes with fewer supports were surprisingly classified correctly! These classes are most likely more distinct in word choice than other groupings. However, most features with overlapping terms are still being classified as ”BPD”

3) Random Forest: Grid Search CV Parameters:

- ’n_estimators’: [100, 200, 300]
- ’max_depth’: [None, 10, 20, 30]
- ’min_samples_split’: [2, 5, 10]
- ’min_samples_leaf’: [1, 2, 4]
- ’class_weight’: [’balanced’]

Chosen Parameters:

- ’n_estimators’: 200
- ’max_depth’: None
- ’min_samples_split’: 5
- ’min_samples_leaf’: 1
- ’class_weight’: ’balanced’

Results:

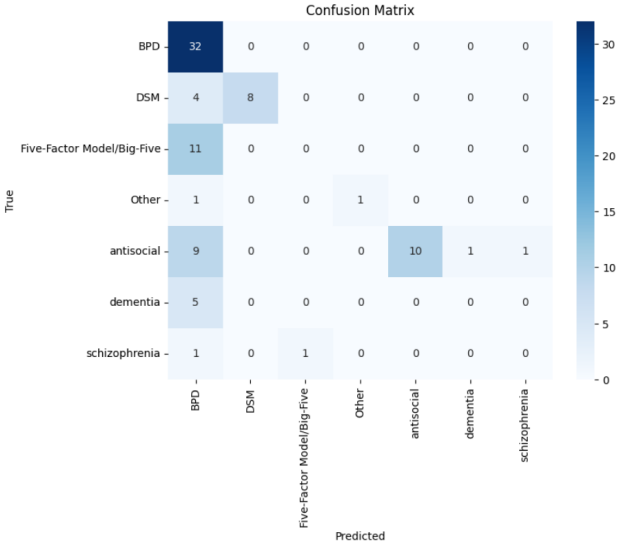

Fig. 21. Random Forest Confusion Matrix

Observations:

- At first, random forest was the poorest performing model with just a 0.07% accuracy.
- Surprisingly, adjusting the variance threshold did not help Random Forest as much as other models
- Instead, the ANOVA p-value feature selection helped. It was similar to the variance threshold, but has a more defined cut off (0.05 is well known to be the acceptance threshold). It helped random forest become one of the best performing models with a 60% accuracy.

C. Both

Neural Networks are able to classify both Classification AND regression targets!

1) Neural Network: My goal was to have the Neural Network reflect the BEST of the above models given that it could simultaneously train both the regression variable (year) AND the classification (Thematic groupings of personality disorders)

I used BatchNormalization to stabilizes the model and Dropout to prevent overfitting. Originally, I had a more generalized version, but it ended with around a 40% accuracy.

I used softmax to identify a multiclassification label

Results:

TABLE IV
CLASSIFICATION REPORT: NEURAL NETWORK

Class	Precision	Recall	F1-Score	Support
BPD	0.46	0.97	0.62	32
DSM	0.62	0.42	0.50	12
Five-Factor Model/Big-Five	0.00	0.00	0.00	11
Other	0.50	0.50	0.50	2
Antisocial	1.00	0.29	0.44	21
Dementia	0.00	0.00	0.00	5
Schizophrenia	0.00	0.00	0.00	2
Accuracy			0.51	85
Macro Average	0.37	0.31	0.29	85
Weighted Average	0.52	0.51	0.43	85

Classification Accuracy: 0.5058823529411764

Regression MAE: 2.4468846321105957

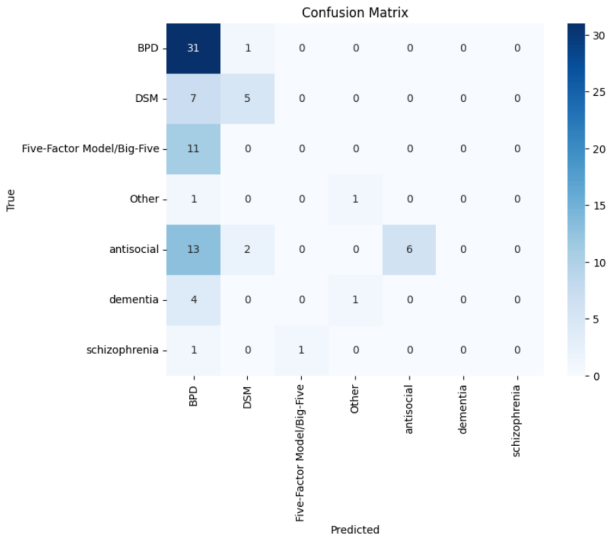

Fig. 22. Neural Network Confusion Matrix

Observations:

- This is a model that seemingly worked better WITHOUT feature selection, most likely because it performs feature selection by itself.
- This model surprisingly did not perform as well as individual models. However, that is most likely because of the tuning. Other models had higher levels of tuning
- This is also the only model to account for both year AND classification at once, which gives many brownie points.
- I still choose this model as the prime model.

D. Classification Analysis

- BPD
  - Across all reports, BPD has very high recall (close to or at 1.00), which means the classifier is very good at identifying true BPD cases.
  - However, its precision ranges from 0.46 to 0.54, indicating many false positives. It often over-predicts BPD. This is most likely because it is the Majority class
- DSM
  - Precision for DSM is consistently high (0.62 to 1.00), but recall is more variable (0.42 to 0.67).
  - This means DSM predictions are often correct when made, but the model misses some DSM cases. This is most likely due to the overlapping terms with other disorders, as DSM is a measure of personality disorders, almost like a container.
  - F1-scores for DSM are relatively high, often above 0.70, showing good balance when recall isn't too low

- Five-Factor Model/Big-Five Performs,Dementia, Other and Schizophrenia
  - Precision, recall, and F1-score are consistently 0.00 across all reports. It is EXTREMELY poor
  - This is most likely due to how it was classified. "Five" may not be an appropriate classification.
  - Many of the other items have VERY low support thresholds, which most probably contribute to the lack of classification
- Antisocial
  - Precision is very high (1.00) across all reports, meaning when it predicts "antisocial," it's usually correct
  - However, recall is low (0.29 to 0.52), so the model misses many actual antisocial cases. This is most likely due to the terms for this model being VERY distinct, most likely with extremely high lifts, causing unique terms to be inflated

VII. Analysis

Overall, the model is not performing completely up to par. There are a lot of possibilities as to how this model could be improved, starting with pulling in more data. Many attempting to improve the Accuracy scores were included but unable to function, such as using SMOTE or Stratified Cross Validation,all due to the limited splits in data.

- Lemmentization greatly helped with parsing the data to find potential Classifications. Not only did it clean messy text, but it allowed for a greater support threshold which in turn cut down on noise.
- The Sequential Data Mining could be improved by using a sliding window.
- Five-Factor Model and Avoidant Personality are never or barely predicted, likely due to severe class imbalance.
- Neural Net is under performing likely due to limited training data
- Logistic Regression seems to be working surprisingly well - most likely a result of it being a simpler model, hence reducing chances of overfitting
- The biggest issue is related to lack of data. It is preventing important methods such as Stratification and SMOTE from being run
- Feature selection, MAINLY selection of a variance threshold, made all the difference when it came to model training. The accuracies shot up by 20% for pretty much all models.

VIII. Improvements

- **JOURNAL**:The amount of data taken in is limited to one journal at the moment. My goal is to input more data. If you have any journals you may think will work feel free to upload here:
- **YEAR RANGE**: For starting purposes, I've limited the intake to 2009 - 2025. There is most probably a lot of data that I could pull in from years prior that would uncover interesting evolutions.
- **PARSING**: There is a method in NLP that extracts important terms using TF-IDF. However, it only extracts single terms. This is why I went with a variation of sequential data mining
- **DATAFRAME CREATION**: As opposed to using lift and confidence as entries, I could look back into using binary embeddings and using ChiSquare selector as a feature selection idea.
- **SEQUENCE EVOLUTION**: Finalize the graph to display evolution of terms over the years
- **YEAR PREDICTION**: Turn this into a classification! Bin and make it a RANGE as opposed to an ordinal value.
- **USAGE**: An interesting usage of this model would to be able to plug in a journal into any year (unlimited by the actual year) and be able to predict what the journal would be classified as in that year. It would require a few adjustments to the final model (such as allowing "Year" to be a feature as opposed to a prediction), but it would make for a very fascinating tell of how a journal would be classified given a year! I should test this!

References

[1] T. A. Widiger, "Editorial for personality disorders: Theory, research, and treatment," *Personality Disorders: Theory, Research, and Treatment*, vol. 1–15, no. 1, pp. 1–1, 2009. doi: 10.1037/per0000167