Capstone on Clustering a University Dataset

Premise

The premise of this capstone was inspired by a clustering challenge on data.world. The idea is that you work as a data scientist for a fictitious non-profit organization whose goal is to increase graduation rates from universities for underprivileged populations. They provide you with a very rich dataset of university information collected by the US Department of Education, and the goal is to identify characteristically similar schools by performing some form of clustering on the dataset. With these clusters identified, the data scientist can leverage insights from the clusters to effectively inform the organization about patterns in the dataset or even suggest strategies that they can adopt in the form of programs or educational campaigns to aid in their mission of increasing graduation rates.

About the dataset

The College Scorecard dataset is quite a large dataset, with over 7000 observations and more than 1700 columns. This was part of the motivation for pursuing this project - with so many features, it's not immediately obvious how we can interpret clusters in the dataset - it's not like you can visualize 1700 dimensions and then coherently talk in terms of all of them! It is not the type of dataset where you can just immediately fit a model, get some results, and call it a day. It requires a lot of preprocessing work before it's ready for model fitting, particularly dimensionality reduction.

It's also not a very clean dataset - there is a lot of missing information and columns that need to be converted into different data types before being able to run machine learning algorithms on it.

Moving on to the type of information available in the dataset - you'll find data about the university's revenue, their student mix, graduation rates, and plenty of information relating to the finances of students - loans, repayment, debt, and income to name a few. The dataset also has many discrete variables that classify the universities into different categories, for example, whether the university is public, private for-profit, or private not-for-profit.

So clearly, this is a very rich dataset and there is plenty of relevant information our models can use while training. However, the challenge is really honing in on the most important information that best discriminates different classes of universities in the dataset, because it's difficult to describe clusters in terms of so many parameters. And that is precisely the challenge that we will be tackling in this exercise.

Processing steps

As part of the renowned Data Science Method™, this project was divided into 4 key stages:

1. **Data Wrangling** - This is where we try to get a raw understanding of what the data looks like and perform transformations and operations on it to bring it to a more usable state
2. **Exploratory Data Analysis** - Here, we try to understand the nature and distribution of our data by plotting visualizations of different features in the dataset. This can give useful insights prior to the modeling phase.
3. **Preprocessing** - The stage where we perform any other operations on the data, like scaling or normalizing, before it is ready to be fed into a machine learning model.
4. **Modeling** - The most important part of course, where we actually try fitting different models onto the preprocessed dataset and evaluate which model performs the best

In the context of this project, I shall summarize what specifically was done in each of these stages, attaching code snippets and visualizations where appropriate.

Data Wrangling

The key motivations in this phase of the project was:

1. Dimensionality reduction
2. Cleaning the data

As a reminder, the dataset had over 1700 features, and it needed to be brought down to a more manageable number before it was appropriate to fit a clustering model to it. This compelled me to be more aggressive with the dimensionality reduction

There are several techniques to perform dimensionality reduction on a dataset, but not all of them were applicable to this situation because clustering is an unsupervised learning task. For example, strategies like RFE (Recursive Feature Elimination) use ground truth labels to iteratively remove unimportant features from the dataset, but ground truth labels weren't available in our dataset.

So, these were the dimensionality reduction techniques I applied to the dataset:

1. **Removing columns with completely missing data.**

This is a no-brainer, columns with no information add no value to the dataset

```
empty_cols = [col for col in college_df.columns if
college_df[col].isnull().all()]
college_df = college_df.drop(columns=empty_cols)
```

2. **Removing columns with mostly missing data.**

What do I mean by 'mostly'? I decided to play around with threshold values for the proportion of missing data in a column, and then drop all columns that exceeded this threshold.

Keep in mind, I needed to be aggressive about the dimensionality reduction and therefore I could be stricter about the amount of non-missing information I wanted in a column.

```
college_df_test = college_df.loc[:, college_df.isna().mean() <
missing_threshold]
```

3. **Removing columns with constant data**

If a column has constant data, then that column isn't adding useful information across observations and therefore isn't changing the intrinsic dimensionality of the data. These features are also useless and could be dropped.

```
college_df_test = college_df_test.loc[:, college_df.nunique() > 1]
```

4. **Removing columns with low variance**

Similar to the previous step, columns with low variance also add very little useful information to the dataset, and therefore we would be able to capture most of the variance in the dataset with a machine learning model without these columns that just add unnecessary dimensionality to the dataset.

Again, this required playing with threshold values to achieve a balance between reducing dimensionality in the dataset but also not losing too much potentially useful information from the dataset either

```
from sklearn.feature_selection import VarianceThreshold
sel = VarianceThreshold(threshold=0.001)
sel.fit(college_df_test/college_df_test.mean()) #Divide by mean to normalize
the features
mask = sel.get_support()
reduced_college_df = college_df_test.loc[:, mask]
```

5. **Removing highly correlated columns**

If two columns are highly correlated, then we only need one of those columns to retain information about the dataset. So, by looking at pairs of highly correlated columns and dropping one of them, we can eliminate features from the dataset. A correlation of more than 70% was a threshold that I empirically decided to be best.

```
corr_df = reduced_college_df.corr().abs()
corr_mask = np.triu(np.ones_like(corr_df, dtype=bool))
tri_df = corr_df.mask(corr_mask)
to_drop = [c for c in tri_df.columns if any(tri_df[c] >  0.70)]
reduced_corr_df = reduced_college_df.drop(to_drop, axis=1)
reduced_corr_df.shape
```

As a result of all these operations, I managed to bring down the dimensionality from 1725 features all the way down to 222, which is far more reasonable

Next came cleaning the data. And while I was inspecting some of the columns, I noticed something interesting - columns that encoded numeric information were actually `'object'` data types in Pandas because some columns had the token *"PrivacySuppressed"* when information wasn't available for that observation column due to privacy reasons. It was the presence of this token that caused Pandas to parse the column as an `'object'` column instead of a numeric column.

Thus, I replaced all occurrences of *"PrivacySuppressed"* with `np.nan` and imputed the numeric columns with their median.

In the end, no columns had missing data, making the new dataset ready for visualization.

<u>Exploratory Data Analysis</u>

In this part of the project, my goal was to see if I could manually find some clusterings in the dataset by scatter plotting interesting pairs of regression variables and then hue-ing the data points with some discrete variable that encoded some kind of categories.

I was also keeping an eye out for potential outliers that could influence the efficacy of the clustering models (for example, K-Means is sensitive to outliers)

For instance, when I was plotting the proportion of undergraduates who were 25 and above against median debt, I noticed a clear outlier:

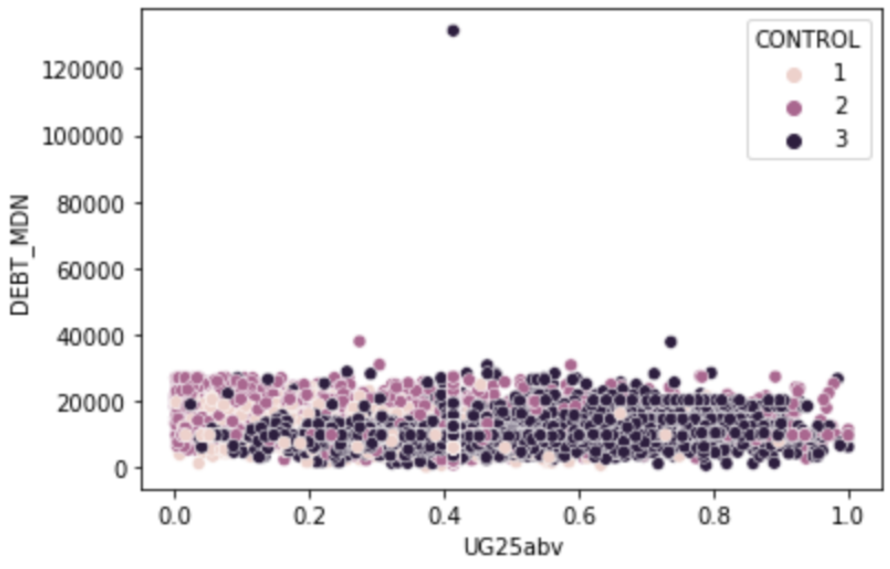


Fig 2.1: Plot of Median Debt vs Undergraduates above 25 rate

Which when eliminated, gave a more accurate picture:

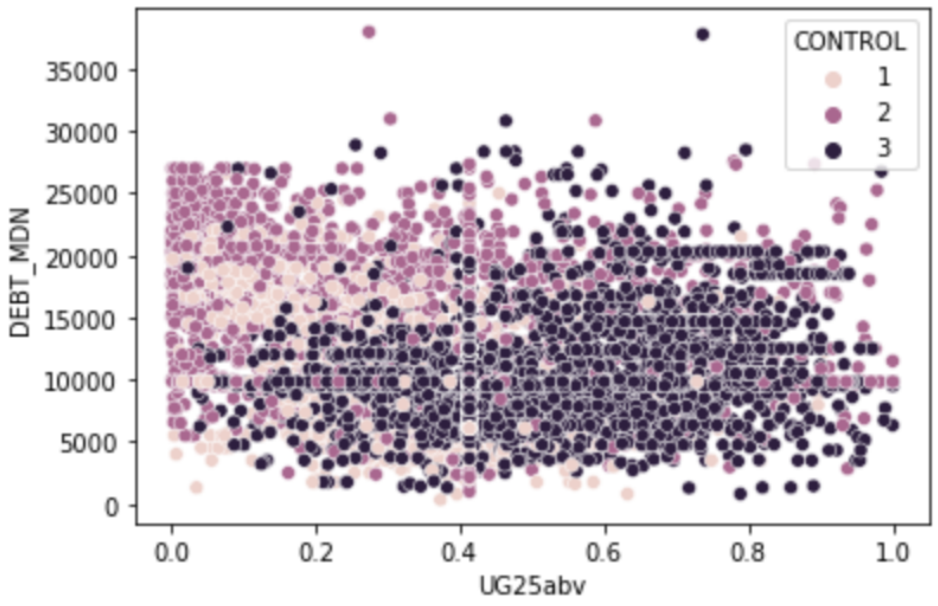


Fig 2.2: Plot of Median Debt vs Undergraduates above 25 rate, sans outlier

Manually choosing pairs of regression variables and a hue-ing variable to identify clusters wasn't producing many interesting results, so I decided to automate it by building a seaborn pairplot with a set of regression features and a hue-ing variable.

Although this didn't produce any obvious clusters either, I did notice that the `PCTFLOAN's` distribution was bimodal, which immediately indicates two distinct categories in the dataset. I thought that insight might be useful once I started collecting all my inferences from the modeling phase.

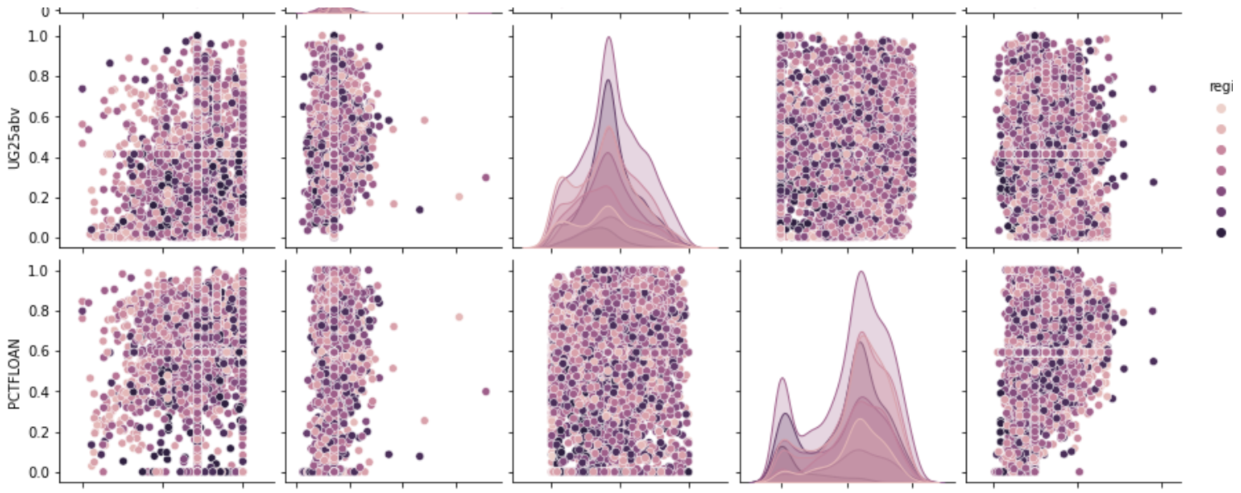


Figure 2.3 : Seaborn pairplot showing bimodal distribution of `PCTFLOAN`

I also wanted to explore if there were any interesting relationships between ethnicity or gender with respect to the other regression variables. I made some plots:

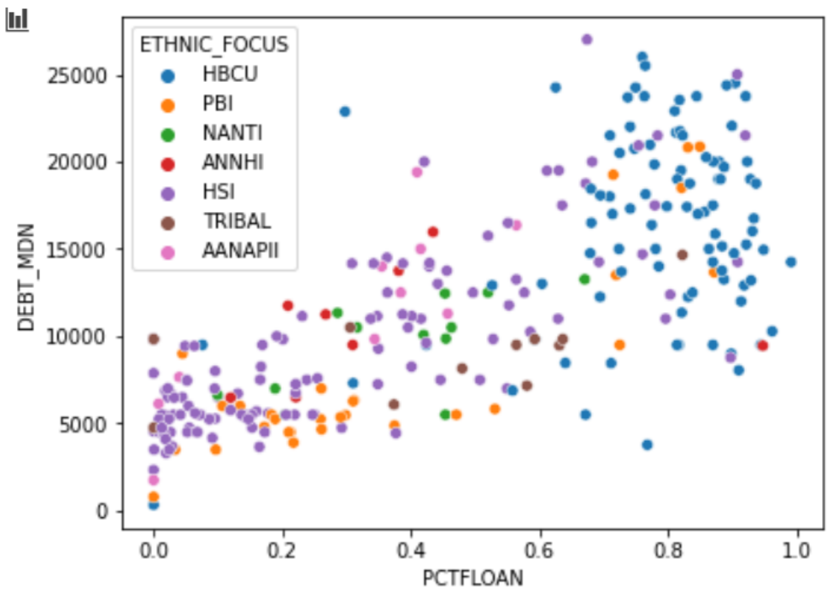


Figure 2.4: Median Debt vs Percentage Federal Loan, hued by ethnicity

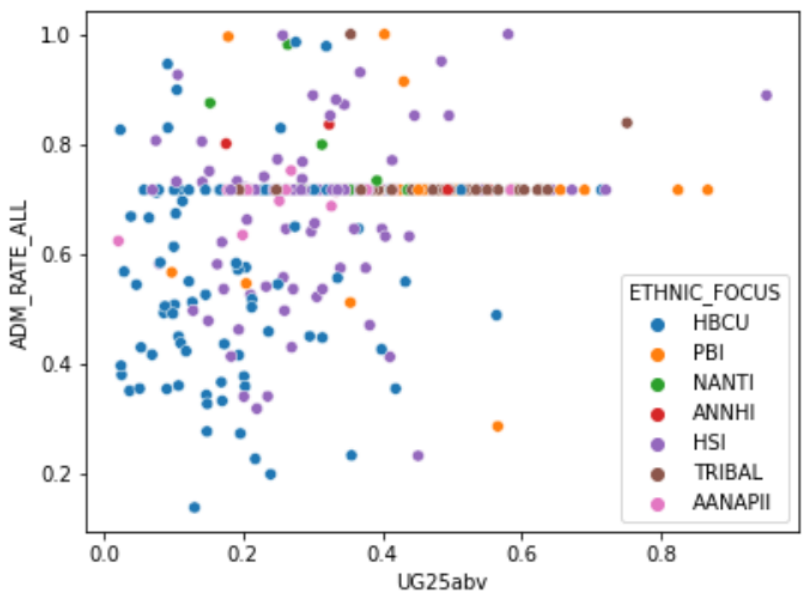


Figure 2.5: Admission Rate vs Undergraduates above 25 rate, hued by ethnicity

But soon realized that universities catered to specific universities or genders were a very small subset of the data and thus limited the amount of useful clustering insight that could be drawn from those features.

The outcomes of the EDA were mixed - I was able to eliminate some potentially problematic outliers by eyeballing some scatterplots, but wasn't able to manually find any interesting clusterings in the data. Though in hindsight, this might have been expected, since there were still about 200 columns in the dataset, and it would have been very noteworthy to have seen just 2 regression variables and a hue-ing variable, or 3 features overall, bucketize the data.

Preprocessing

For the preprocessing phase, I had 2 goals:

1. Remove more outliers
2. Normalize the data

I of course wanted to be more systematic about eliminating outliers and realized that normalizing the data into a fixed range would be important for clustering models like K-Means which are distance sensitive, so we would need all features to be in the same scale to give each feature an equal weighting in the algorithm.

I initially wanted to eliminate observations that had a column value that was more than 3 standard deviations away from that column's mean, but that ended up dropping the number of observations from 7803 to just 2577.

Thus, I needed to be more lax about my outlier criterion and empirically found that 8 standard deviations seemed to best balance retaining as many observations as possible while also eliminating notorious outliers.

```
new_df = numeric_df[(np.abs(stats.zscore(numeric_df)) < 8).all(axis=1)]
```

To normalize the data, I just used Scikit Learn's standard `MinMaxScaler`:

```
from sklearn import preprocessing

x = new_df.values #returns a numpy array
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
normalized_df = pd.DataFrame(x_scaled, columns=new_df.columns)
```

Modeling

Now that the data was finally ready, it was time to start fitting models to the dataset. I decided to try these 3 types of clustering models:

1)K-Means
2)Agglomerative Clustering
3)Gaussian Mixture Models

However, before fitting the models to the data, I decided to use PCA to reduce the dimensionality of the data to 2 PCA components. This would help with interpreting the clusters since it could be visualized on a 2D plot. And after the clusters had been determined, I would analyze the coefficients in the PCA components to determine how they related to original features in the dataset.

For each type of clustering model, I decided to manually perform a grid search over the hyperparameter space to find the best set of hyperparameters for the model. Scikit learn has modules like GridSearchCV and RandomSearchCV that find the best set of hyperparameters by performing cross-validation over the dataset, but since this was an unsupervised learning task with no ground-truth labels, I couldn't use these modules.

The way I quantified the betterness of a model was by using the _inertia attribute for K-Means which is a good metric for the goodness of a clustering since it measures the sum of squared distances of samples to their closest cluster, and also used
metrics like the silhouette score, calinski-harabasz, and david-bouldin to rate all 3 types of models.

I also ranked the 3 best models from each type and plotted their resulting clusterings.

Best K-Means:

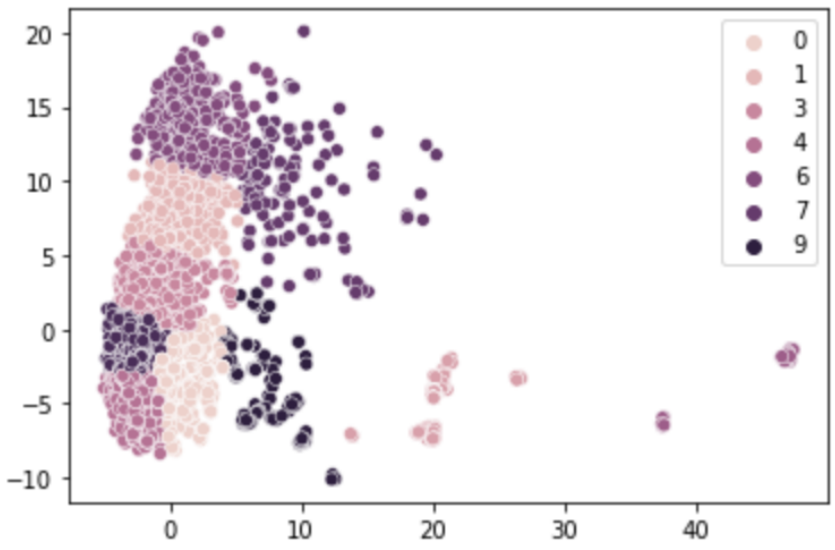


Figure 3.1: Best K-Means model, axes are PCA components

Best Agglomerative:

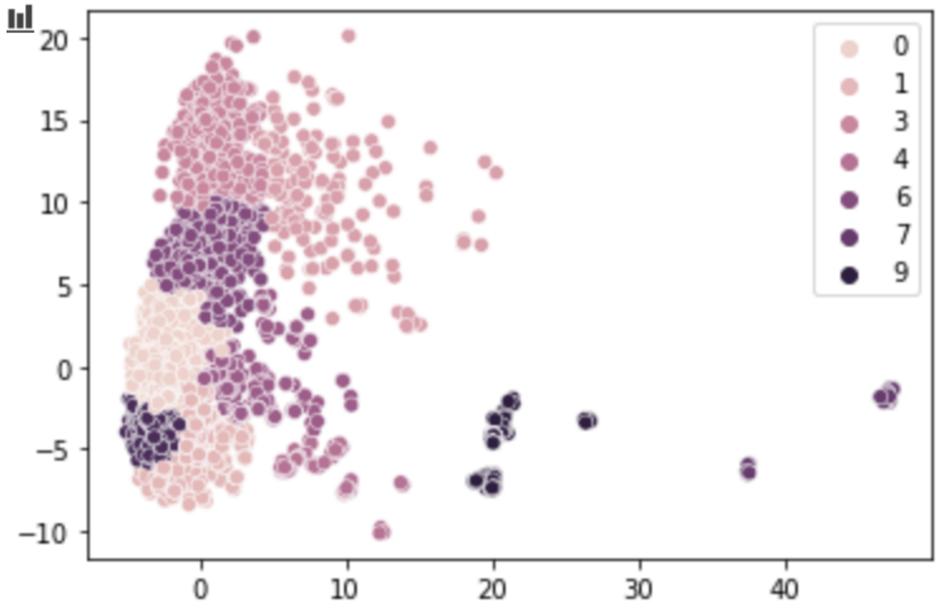


Figure 3.2: Best Agglomerative model, axes are PCA components

Best GMM:

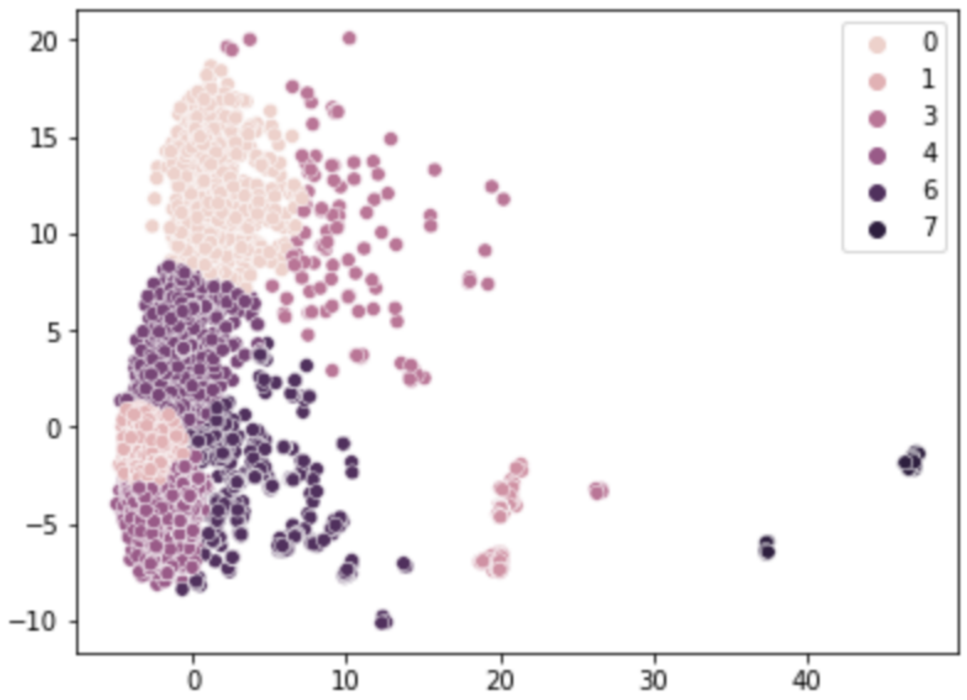


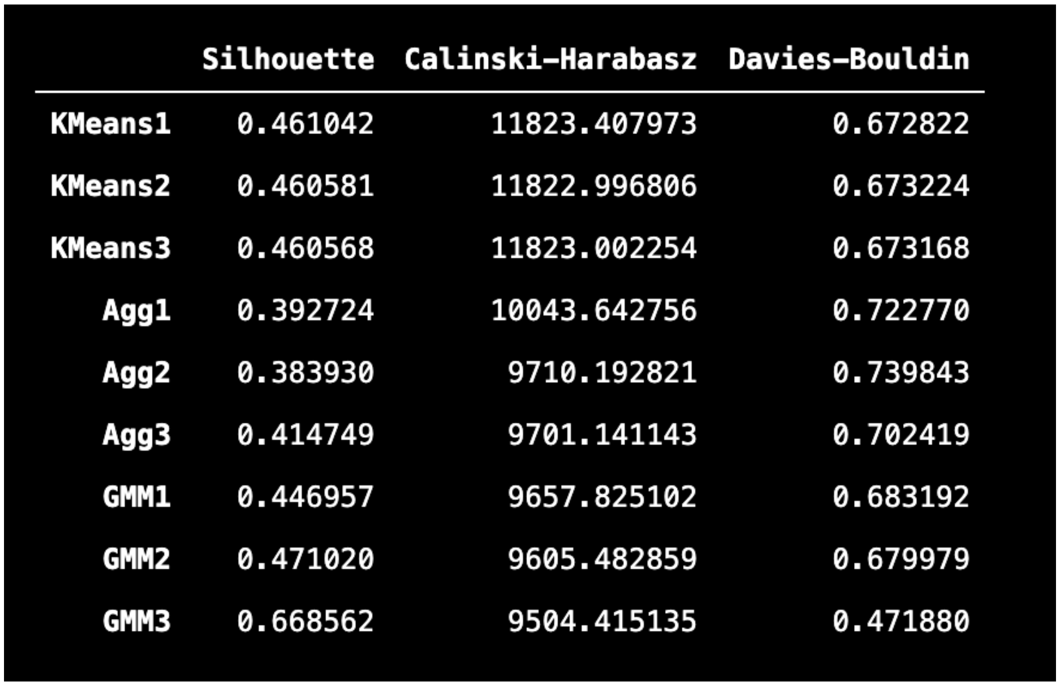
Figure 3.3: Best GMM model, axes are PCA components

After shortlisting the 9 best clustering models, I made a DataFrame of each model's score along the 3 metrics of silhouette, Calinski-Harabasz, and Davies-Bouldin

```
top_9_models = [best_K_means_models[0], best_K_means_models[1],
best_K_means_models[2], best_agglomerative_models[0],
best_agglomerative_models[1], best_agglomerative_models[2], best_GMM_models[0],
best_GMM_models[1], best_GMM_models[2]]

silhouette_scores = [silhouette_score(df_pca, model[1]) for model in
top_9_models]
CH_scores = [calinski_harabasz_score(df_pca, model[1]) for model in
top_9_models]
DB_scores = [davies_bouldin_score(df_pca, model[1]) for model in top_9_models]

index = ['KMeans1', 'KMeans2', 'KMeans3', 'Agg1', 'Agg2', 'Agg3', 'GMM1',
'GMM2', 'GMM3']
model_evaluation_dict = {'Silhouette' : silhouette_scores, 'Calinski-Harabasz'
: CH_scores, 'Davies-Bouldin' : DB_scores}
model_evaluation_df = pd.DataFrame(model_evaluation_dict, index=index)
model_evaluation_df
```



	Silhouette	Calinski-Harabasz	Davies-Bouldin
KMeans1	0.461042	11823.407973	0.672822
KMeans2	0.460581	11822.996806	0.673224
KMeans3	0.460568	11823.002254	0.673168
Agg1	0.392724	10043.642756	0.722770
Agg2	0.383930	9710.192821	0.739843
Agg3	0.414749	9701.141143	0.702419
GMM1	0.446957	9657.825102	0.683192
GMM2	0.471020	9605.482859	0.679979
GMM3	0.668562	9504.415135	0.471880

Figure 3.4: Table of model evaluation results across 3 different clustering metrics

It was upon analyzing this table that it seemed clear to me that the first K-Means model performed the best overall on the dataset. So with a winner selected, I assigned the cluster labels of that model to the original dataframe with the unscaled and unnormalized values.

To actually explain the best model's clustering however, I needed to better understand the generated PCA components. Since each PCA component is a linear combination of columns in the original dataframe, I decided to investigate what columns had the highest coefficients associated with them, which would indicate a higher degree of importance for that PCA component to explain variance in the dataset.

```
df_pca_components = pd.DataFrame(pipe.named_steps['reducer'].components_,
columns=df.columns, index=['PCA_1', 'PCA_2'])
df_pca_components.head()

df_pca_components.iloc[0].abs().sort_values(ascending=False)
df_pca_components.iloc[1].abs().sort_values(ascending=False)
```

From this, I realized that the most important columns from the first PCA component were:

1. NOPELL_DEBT_N
2. MD_INC_DEBT_N
3. NOTFIRSTGEN_DEBT_N

And the most important columns from the second PCA component were:

1. FEMALE_RPY_3YR_RT_SUPP
2. MALE_RPY_3YR_RT_SUPP
3. NOTFIRSTGEN_RPY_3YR_RT_SUPP

Interestingly, the first PCA component found columns relating to debt most important in explaining variance along its axis, while the second PCA component found columns related to repayment most important in explaining variance along its axis.

With the most important columns shortlisted, I was finally in a position to be able to explain the best clustering I could find from the dataset.

Results and recommendations

To explain each cluster using the top 6 most important columns, I decided to perform several aggregations of the 6 columns for each cluster :

```
aggregations_dict = {
    'FEMALE_RPY_3YR_RT_SUPP' : ['mean', 'median', 'min', 'max', 'std'],
    'MALE_RPY_3YR_RT_SUPP' : ['mean', 'median', 'min', 'max', 'std'],
    'NOTFIRSTGEN_RPY_3YR_RT_SUPP': ['mean', 'median', 'min', 'max', 'std'],
    'NOPELL_DEBT_N' : ['mean', 'median', 'min', 'max', 'std'],
    'MD_INC_DEBT_N': ['mean', 'median', 'min', 'max', 'std'],
    'NOTFIRSTGEN_DEBT_N' : ['mean', 'median', 'min', 'max', 'std']
}

aggregations = unnormalized_df.groupby('LABELS').agg(aggregations_dict)
```

Column encodings:

Loan repayment rates for females 3 years after graduating: FLR
Loan repayment rates for males 3 years after graduating: MLR
Loan repayment rates for non first-generation students 3 years after graduating: NFGLR

Unfortunately for the debt columns, there is no explicit mention in the data dictionary for what exactly the scale is for these values, or if they represent a cumulative debt value or some kind of average/median. Therefore, it is only appropriate to treat these values relative to each other for inter-cluster comparisons.

Debt quantification for students in families falling under the median income bracket - MID
Debt quantification for students who didn’t receive a Pell Grant - NPD
Debt quantification for non first-generation students - NFGD

Characterization of each of our clusters:

Cluster 1: (8.45% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.425	0.385837	0.448397	1231.939	840.590	2420.154
Median	0.456	0.405530	0.481179	1176.0	580.0	2089.0
Min	0.145	0.074249	0.154696	28.0	17.0	23.0
Max	0.773	0.751637	0.776990	4665.0	3824.0	8412.0

- Weak female loan repayment rates
- Weak male loan repayment rates
- Weak non-first loan repayment rates

Cluster 2: (8.53% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.818	0.779567	0.827951	556.89	630.253	933.284
Median	0.824	0.785915	0.829710	354.0	410.0	616.0
Min	0.519	0.507431	0.488636	28.0	18.0	41.0
Max	1.000	1.000000	0.996942	2824.0	2637.0	4745.0

- Pretty average across all dimensions

Cluster 3: (4.2% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.421	0.415953	0.449292	13094.90	8591.93	20478.91
Median	0.394	0.393501	0.427468	15958.0	10276.0	25955.0
Min	0.394	0.393501	0.427468	4762.0	2750.0	7675.0
Max	0.545	0.475300	0.555682	15958.0	10276.0	25955.0

- Very weak female loan repayment rates

- Very weak male loan repayment rates
- Very weak non-first generation repayment rates
- Very high debt for Non-Pell Grant students
- Very high debt for students from median income families
- Very high debt for non-first generation students

Cluster 4: (7.4% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.732	0.658101	0.744370	427.125	418.923	667.552
Median	0.742	0.638171	0.738095	271.0	410.0	322.0
Min	0.408	0.279279	0.402062	11.0	0.0	11.0
Max	1.000	0.985294	1.000000	2998.0	2510.0	5881.0

- Low debt for Non-Pell Grant students

Cluster 5: (18.46% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.419	0.404931	0.444704	284.364	311.26	406.038
Median	0.420	0.424779	0.451768	275.0	410.0	318.0
Min	0.060	0.043796	0.021277	0.0	0.0	0.0
Max	0.788	0.891892	0.830189	1663.0	1913.0	2278.0

- Weak female loan repayment rates
- Weak male loan repayment rates
- Weak non-first generation loan repayment rates
- Very low debt for Non-Pell Grant students
- Very low debt for students from median income families
- Very low debt for non-first generation students

Cluster 6: (0.556% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.497	0.494815	0.522184	17194.05	12211.771	24026.00
Median	0.546	0.542936	0.572239	18704.0	13561.0	24917.0
Min	0.357	0.355799	0.377580	12832.0	8314.0	21452.0
Max	0.546	0.542936	0.572239	18704.0	13561.0	24917.0

- Very high debt for Non-Pell Grant students
- Very high debt for students from median income families
- Very high debt for non-first generation students

Cluster 7: (6.72% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.909	0.889015	0.914052	587.3640	930.56264	1273.917
Median	0.916	0.896635	0.920091	399.0	612.0	892.0
Min	0.633	0.531250	0.659420	37.0	60.0	115.0
Max	0.987	1.000000	0.995146	2997.0	4304.0	6337.0

- Very strong female loan repayment rates
- Very strong male loan repayment rates

Cluster 8: (1.95% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.818	0.802410	0.836882	3822.7723	4808.520	7234.016
Median	0.828	0.807854	0.837394	3279.0	4130.0	6435.0
Min	0.551	0.524245	0.581233	482.0	410.0	505.0
Max	0.948	0.945972	0.951067	9707.0	10547.0	17168.0

- Strong female loan repayment rates
- High debt for students from median income families

Cluster 9: (40.03% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.620	0.559181	0.630510	265.426	307.127	362.723
Median	0.598	0.547753	0.612903	335.0	410.0	394.5
Min	0.217	0.196078	0.300000	0.0	0.0	0.0
Max	0.986	1.000000	1.000000	2205.0	2571.0	3620.0

- Very low debt for Non-Pell Grant students
- Very low debt for students from median income backgrounds
- Very low debt for non-first generation students

Cluster 10: (3.65% of data)

	FLR	MLR	NFGLR	MID	NPD	NFGD
Mean	0.411	0.397958	0.441000	4053.326	2693.50	7870.06
Median	0.425	0.343213	0.428122	3835.0	2681.0	8145.5
Min	0.228	0.195795	0.232003	925.0	410.0	889.0
Max	0.752	0.754185	0.758403	8886.0	8638.0	14174.0

- High debt for students from median income backgrounds
- High debt for non-first generation students

From this clustering data, here are some recommendations that can be made:

1. Cluster 6, although a very small cluster, represents a set of universities where students seem to be struggling with large amounts of debt. However, the smallness of this cluster means that the non-profit can be more targeted with its strategizing and perhaps try helping students with debt relief programs

2. Cluster 3 seems to be struggling across all dimensions - high debt and low repayment rates. This is another group that needs attention. Repayment rates could definitely be

helped with debt relief programs of course, but if the debt cannot be reduced, perhaps the non-profit should focus on helping students graduate with higher paying jobs. This could mean educating students better about available opportunities and potentially strengthening their profiles with professional career growth workshops.

3. Cluster 5 is interesting because even though they seem to have low levels of debt, they have weak repayment rates. Perhaps a closer look should be taken at universities in this cluster - are students perhaps having trouble finding jobs after graduation and therefore struggling to pay even small amounts of debt?

4. Cluster 9 seems to be a healthy performing cluster and eliminates a good 40% of the universities that the non-profit can choose to not focus on as much as universities in the other clusters. Finite resources that the non-profit has should definitely be dedicated to the more in-need clusters.

Conclusion and scope for further research

This was definitely a very interesting project to engage in and gave a good first impression of what it is like to deal with a relatively large volume of data. It also helped me realize that it’s extremely important to transform, visualize, and preprocess the data before even considering fitting a model. There were a lot of steps in the pipeline leading up to model fitting that were essential to getting sensible results, like dimensionality reduction and outlier elimination, so one cannot be hasty about that kind of thing.

Working in an almost real-world setting was also appreciated because it forces one to think in terms of a particular context or domain when analyzing the data, and that is an important skill for a Data Scientist to have and cultivate. And speaking of Data Scientists, this project helped me truly appreciate the difference between a Data Scientist and a Machine Learning Engineer, and how there’s so much more involved beyond just model-fitting that a Data Scientist needs to get right - responsibilities that might go unnoticed to an end-user or organization using a machine learning model.

That all being said, I must humbly admit that my clustering results, though satisfactory to me, may not truly reflect the possible characterization of the universities into groups, and that with such a rich dataset, there are plenty more opportunities or avenues to pursue to produce possible different equally or more insightful clusters than these. For example, other dimensionality reduction techniques could be explored, like Non-Negative Matrix Factorization. Or the thresholds during the Data Wrangling stage could be set in different combinations to produce a completely different modeling dataset. Maybe a different type of clustering model, like density-based approaches could be tried out as well. Several possibilities, of which I have explored the one that my data science intuition has guided me towards.