NHL Draft Predictions

Members:

- Quoc-Huy Nguyen
- Ryan DeSalvio

Github: https://github.com/quochuyn/nhl-draft-predictions

1. Introduction

One of the most exciting events in the National Hockey League's (NHL) calendar is the annual draft where teams pick and choose hockey star prospects. Attempts at predicting the pick order use number-based quantitative features from player statistics or qualitative features such as the player's perceived strengths and weaknesses as well as the compatibility with the respective teams. What sets this project apart is the use of text-based Natural Language Processing (NLP) techniques to extract each player's qualities from player scouting reports. Solving this problem reveals novel techniques to make draft predictions and describe players using word embeddings.

Both the supervised and unsupervised tasks go through three different data pipelines for text preprocessing: (1) Natural Language Toolkit (NLTK) and TF-IDF vectorization, (2) Sentence Transformer's BERT embeddings, and (3) OpenAI's Large Language Model embeddings. The first technique establishes a baseline for the downstream analysis and model performance while the last two techniques are recent developments in the NLP space that are trained on large-scale high quality datasets. The other dataset features go through separate pipelines for standardization/normalization and one-hot encoding (for categorical features). Finally, the supervised model follows an ordinal regression model which is a sequence of binary classifiers (e.g., Random Forests) that follows a ranking rule for determining the draft order. The main unsupervised methods pass through KMeans clustering and a comparison between TSNE vs UMAP visualizations.

2. Related Work

Similar analyses have been conducted throughout the sports analytics landscape, such as Chris Zaire's efforts to evaluate the 2020 NFL draft class. He created word clouds describing the most salient words for a select few players and performed sentiment analysis by using NLP techniques to determine how positive or negative a player review was (i.e., to determine if the pros outweigh the cons). Our project differs by not using sentiment scores, but instead the word embeddings from the text for the explicit task of predicting draft positions as well as clustering players into similar groups.

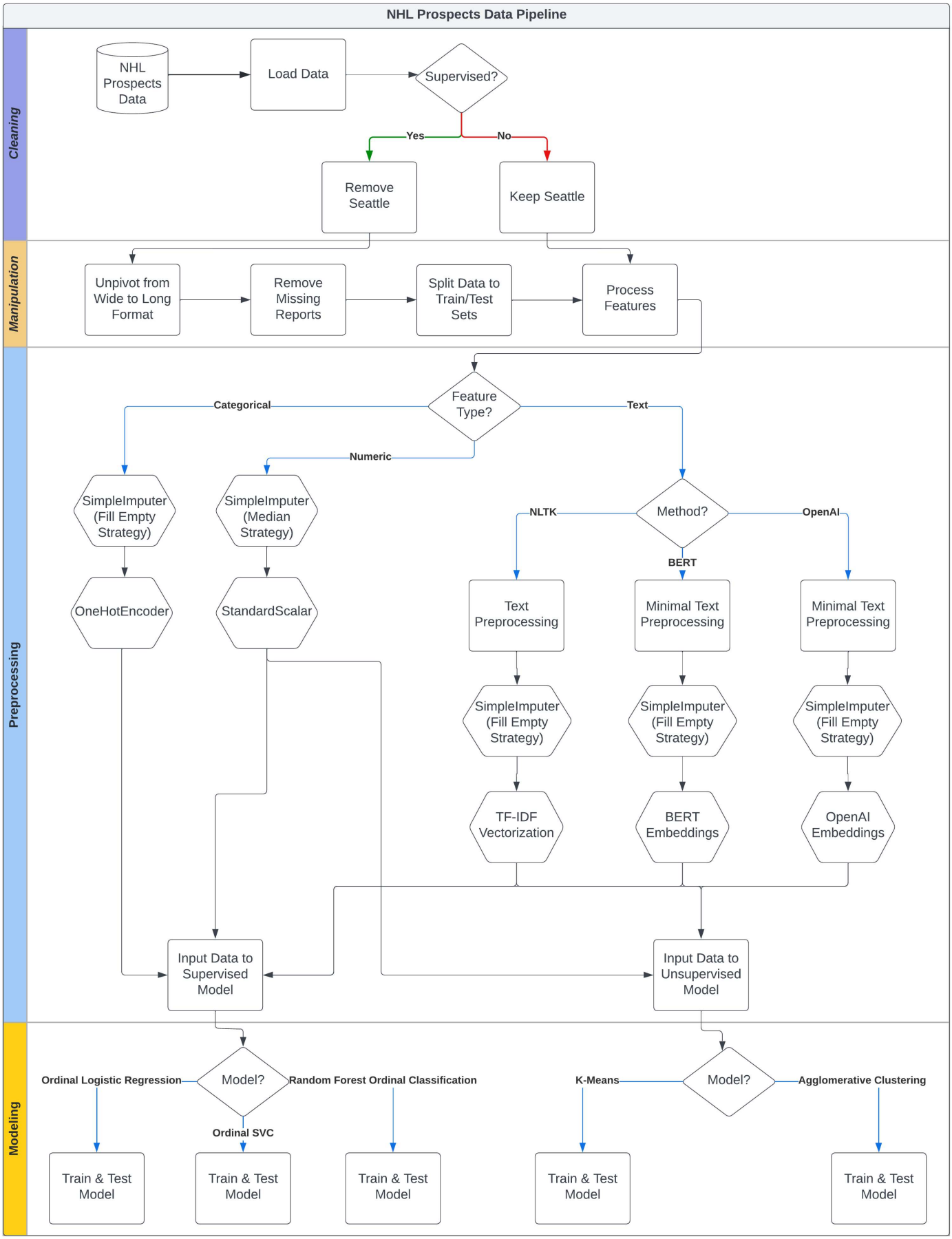
Brandon Petrosilli uses mock drafts as his text data to perform an analysis on the 2022 NFL draft class. A key difference between his project and our project is his prior knowledge of the precise order teams are picking. This data is not possible with the NHL draft because the team ordering is not known prior to the actual event. In fact, there is a lottery system for the first round of the NHL draft and our data focuses on the first round.

One study by Benjamin Hendricks uses BERT-based sentiment analysis of tweets about players for picking winning players in a fantasy sports betting environment. Our project also uses one of the BERT models, but for analyzing scouting reports.

3. Data Source

A scouting report is a textual writeup of a player's strengths and weaknesses by an NHL scout. Each NHL team has an entire scouting department made up of many different scouts specializing in different regions who all create detailed scouting reports on different players which are then used to evaluate whether a team should draft a player or not. This work is kept private by teams so as to not provide a competitive advantage to the other teams they are competing against. Some public news and blogging sites also have "prospect experts" or scouts of their own which perform this work for the public eye so the public can be informed about these player's their favorite teams may be drafting. Our dataset is composed of player information and scouting reports from these public reports for each of the top 40 picks in the NHL draft from the years 2014-2023. The information was gathered manually via public sporting news outlets such as The Athletic, ESPN, FC Hockey, Smaht Scouting, The Daily Faceoff, and EliteProspects. Player biometrics such as height(inches) and weight(pounds) were taken from official measurements via the NHL Scouting Combine which is held yearly for the top 100 or so NHL prospects. In total, there are 320 prospects represented with an average of 3.4 scouting reports per player. All of this information is aggregated into a csv file which was used for the modeling attempts.

4. Feature Engineering


NHL Prospects Data Pipeline
Cleaning
Manipulation
Preprocessing
Modeling
NHL Prospects Data
Load Data
Supervised?
Yes
No
Remove Seattle
Keep Seattle
Unpivot from Wide to Long Format
Remove Missing Reports
Split Data to Train/Test Sets
Process Features
Feature Type?
Categorical
Numeric
Text
SimpleImputer (Fill Empty Strategy)
SimpleImputer (Median Strategy)
Method?
NLTK
BERT
OpenAI
OneHotEncoder
StandardScalar
Text Preprocessing
Minimal Text Preprocessing
Minimal Text Preprocessing
SimpleImputer (Fill Empty Strategy)
SimpleImputer (Fill Empty Strategy)
SimpleImputer (Fill Empty Strategy)
TF-IDF Vectorization
BERT Embeddings
OpenAI Embeddings
Input Data to Supervised Model
Input Data to Unsupervised Model
Model?
Ordinal Logistic Regression
Ordinal SVC
Random Forest Ordinal Classification
Model?
K-Means
Agglomerative Clustering
Train & Test Model
Train & Test Model
Train & Test Model
Train & Test Model
Train & Test Model

Since our prospects data are obtained manually, most cleaning steps have already been conducted. Some cleaning instances include missing values for the height and weight of a player and an invalid value for a player name. When branching between the supervised and unsupervised tasks, some cleaning steps differ. For example the Seattle Krakens, founded in 2018, are a recent addition to the NHL. Their recency may pop up in the scouting reports and potentially bias the supervised models. Afterwards, data manipulation includes unpivoting the data frame from wide to long format (the columns to pivot are the scouting reports) and removing rows with missing text. Increasing the sample size for each draft position label alleviates the issue of having such a large set of labels for a small dataset which bodes poorly for a classification problem. Before processing the features, the data is split into train and test sets based on a group splitting strategy where the groups are the player names. This prevents data leakage: the issue of the training set spilling into the testing set.

With the completion of data cleaning and manipulation steps, feature preprocessing goes through separate sequences depending on the feature type. For numerical features, we employed a simple imputer following a median strategy proceeded by a standardization which transforms the feature to have a mean of 0 and a standard deviation of 1. For categorical features, we employed a simple imputer following a fill empty strategy proceeded by one-hot encoding. The explanation for using one-hot encoding instead of label encoding is to prevent the unintended consequence of creating an order between the categories. For example, with label encoding a defenseman (label=2) may be encoded to be "greater" than a forward (label=1). One-hot encoding fixes this issue and creates binary variables for each category. For text features, there are 3 different pipelines that we have implemented:

1. A basic text processing with NLTK where we remove player names, remove unnecessary whitespace, remove English stop words, remove domain-specific hockey words, and normalize the text with Porter stemmer. The reason for removing player names is to prevent classifying players with similar names which was revealed in our explorations with the unsupervised task. Note that we remove domain-specific hockey words such as locations of interest, nationalities, and etc. to prevent grouping of players from the same leagues. This then passes through a TF-IDF vectorizer layer. Of particular interest are the `ngram_range` parameter for including bi-grams (2 word sequences; e.g., machine learning) and tri-grams (e.g., graph neural networks), the `min_df` parameter for setting a lower bound for how frequent words should appear throughout the corpus, and the `max_df` parameter for setting an upper bound.
2. Minimal text processing with NLTK where we remove player names, remove unnecessary whitespace, and remove domain-specific hockey words. We employ an encoding transformation with a state-of-the-art large language model (LLM) from Hugging Face to obtain the so-called BERT embeddings.
3. Also, minimal text processing with NLTK where we remove player names, remove unnecessary whitespace, and remove domain-specific hockey words. This time we employ an extremely popular LLM, Chat-GPT, from OpenAI to obtain word embeddings.

For the supervised learning exploration task, we wanted to know how well these word embeddings from all 3 pipelines would perform for predicting the player's draft position. Additional features such as the player's height, weight, and position were also included, but the main focus were the word embeddings. For the models, we modified Logistic Regression, K Nearest Neighbors Classification, and Random Forest Classification to be an ordinal regressions task that allowed us to rank the players. This was a balance between a purely classification task and a purely regression task.

For the unsupervised learning portion of the exploration, we wanted to evaluate how player's could be clustered based on similarities in their scouting reports and profiles. To accomplish this, we decided to use the K-Means Clustering algorithm on vectors created from each player's prospect profile information. K-Means was chosen as our clustering algorithm because of its simplicity to use as well as its speed since we were clustering high dimensional embedding vectors.

5. Unsupervised Learning

In order to tackle this clustering problem, we first had to turn our textual scouting reports into numbers which can then be clustered into similar groupings. This process, known as embedding, is at the core of the now booming Natural Language Processing(NLP) domain in machine learning. There are a variety of ways to do this and we explored a couple of the more popular options available, including our own method from scratch, as well as using a more advanced embedder.

5.1 Term-Frequency Inverse-Document Frequency

To begin, we created our own embeddings by using the Term Frequency - Inverse Document Frequency(TF-IDF) algorithm. This process involves mapping every word in the training vocabulary and then attempts to assign the importance of a word in a document. This "importance" number is determined for each word in the document until the collection of words is turned into a numerical vector. This vector is then what is fed into the clustering model. The algorithm for measuring how important a word is, is described below:

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

5.2 BERT Embeddings

While TF-IDF is a very powerful algorithm, we had some concern because our dataset is limited in size. In an attempt to combat this, we decided to also use some pre-trained embedding models trained on extremely large datasets. The first of these models is the Bidirectional Encoder Representations from Transformers(BERT) embedding model. BERT, trained by Google in 2018, is a publicly available Large Language Model(LLM). There are a lot of things unique about LLM's, but at it's core, they are trained on massive amounts of textual data which helps them to understand context in words much better than anything we could do on our own. In theory, this leads to embeddings that are more true to meaning and help to accurately classify sentences as being similar to one another.

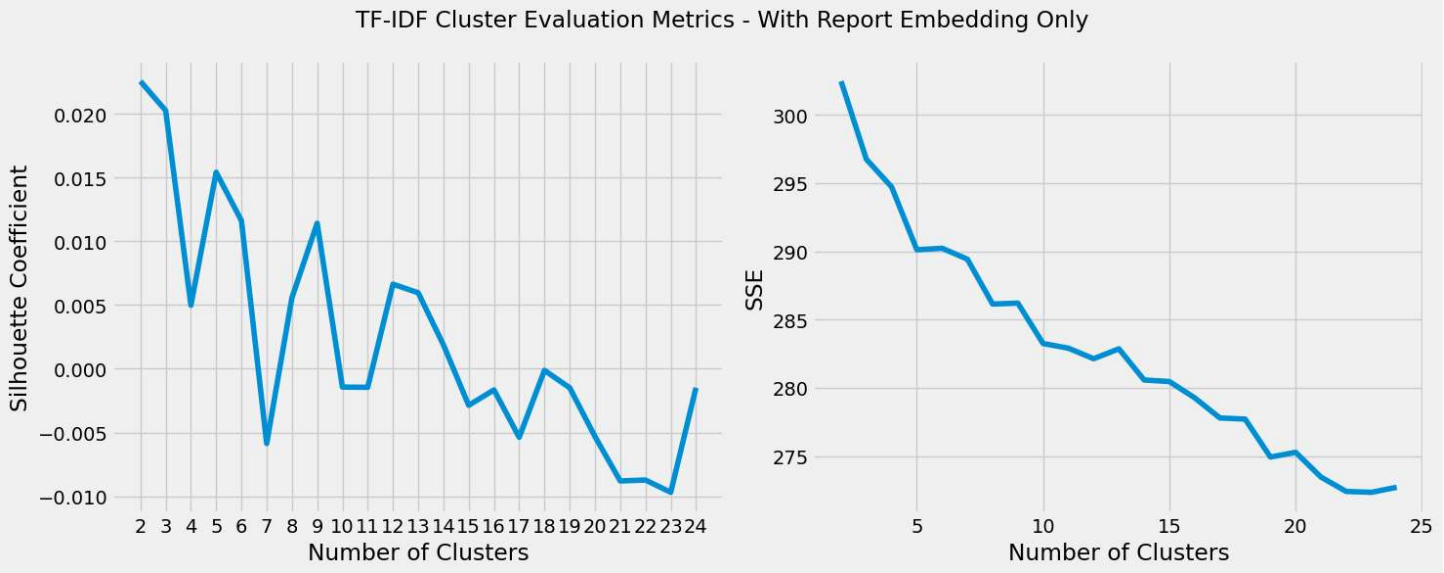
5.3 OpenAI Embeddings

While TF-IDF is a very powerful algorithm, we had some concern because our dataset is limited in size. In an attempt to combat this, we decided to also use some pre-trained embedding models trained on extremely large datasets. OpenAI has been dominating the news for it's extremely popular large language model(LLM), Chat-GPT. LLM's have their own embedding models which are trained during the creation of the LLM and is used for understanding input to the model. OpenAI has made their embedding models available for use to the public via their API and we were also able to take advantage of this powerful pre-trained model for clustering our scouting reports. The embedding vectors from OpenAI were retrieved via their API by passing our scouting reports in an API call and receiving the embedding vectors back. We wanted to make sure that we covered all of our bases during model evaluation so we could see really see how effective our model could be. This also helped to partially solve our small data size issue.

5.4 Unsupervised Models

After the embeddings were created for each of the reports, we then had to perform the actual clustering of these embedding vectors. To do so, we explored two extremely common clustering models, K-Means Clustering and Agglomerative Clustering. We chose both of these models because they were relatively easy to understand, were easy to use, and are very popular in clustering algorithms for many NLP tasks.

K-Means clustering attempts to create k clusters from n samples were each cluster is defined by having a mean value. The samples are then grouped by their nearness to the cluster's mean value. One of the most important, and difficult, parts of K-Means is deciding what the value k should be. There exists a wide variety of methods to decide this value.. We began by evaluating how K-means performed over a range of 2-25 different clusters and took the Silhouette Score and the Sum Squared Error(SSE) values at each cluster to evaluate performance of the model. Silhouette Coefficients attempt to measure how good and well-separated the clusters you created are. SSE is a similar metric in that it measures the distance between a data point and the center of the cluster it is assigned. You want to minimize the error(SSE) and maximize the cluster separability(Silhouette Coefficient). The "Elbow" Method, is a common technique used with the Silhouette Score and SSE values to determine the proper k value. A plot where the x-axis is the cluster values and the y-axis is the score value you evaluating is created. The correct cluster number is the "elbow" of the graph or where the decrease in score relative to the k value is so sharp a visible elbow is seen. An example with our data can be seen in the image below.



The second method we chose to use for clustering is called Agglomerative Clustering. This model is a clustering algorithm which is termed "bottom-up". The model begins with a single cluster and then begins to compare each value provided to it with the previous values. A distance calculation is done, and then depending on how far the current value is from the previous, it is determined whether the new value is placed into the initial cluster or a new cluster needs to be created. This process repeats for every data point until all of the data is placed into clusters. The main difference between this process K-Means is the sequential nature of the algorithm. This can be useful when you want to be fine grained with the level of clustering you want in your data and provides a fresh perspective on the process compared to K-Means.

5.5 Unsupervised Hyperparameter Tuning

For both models, we exhaustively explored the different options available to us. We began by tuning just using the parameters available to us in TF-IDF function. Mainly, the min_df and max_df parameters which allow you to control the minimum and maximum amount of mentions a word must appear in a document for it be considered. From there, we explored whether adding height and weight values to our embedding vector improved the cluster definition for our models.

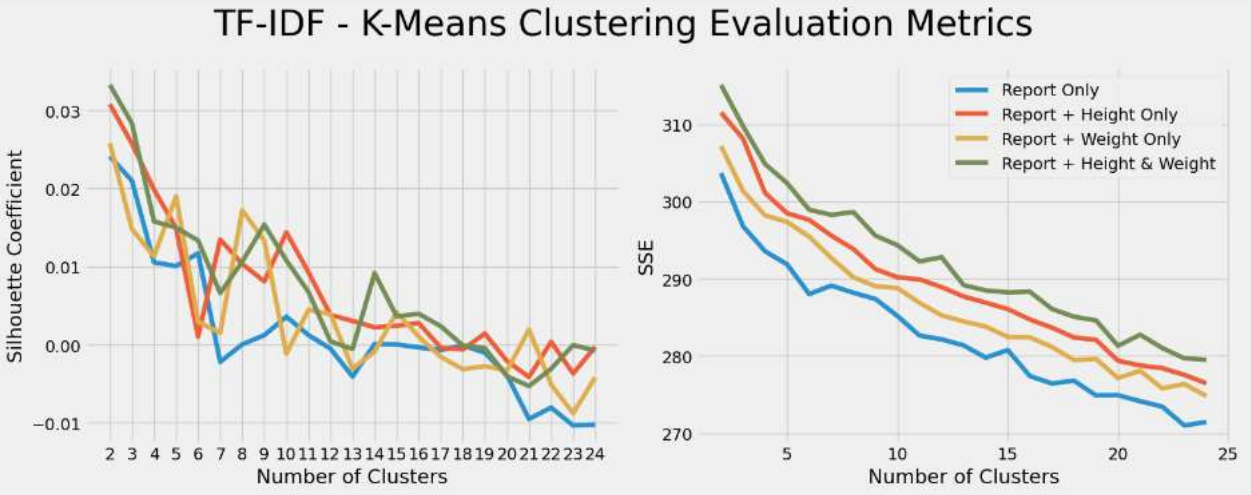
1. Clusters were well defined
2. There was no obvious underlying theme to the cluster

Evaluating a "well defined" cluster was mostly straightforward. We used the silhouette score/SSE as a barometer for cluster division and then visualized the clusters via the UMAP algorithm. Evaluating whether there was an underlying theme turned out to be a major issue. We found that the clustering algorithms tended to converge on clusters representing position even though we had made every attempt to remove positionality from the dataset. For instance, the scouting report and the words used to describe a goalie do not have very much in common at all with a forward. To combat this, we attempted to remove any positionality words from dataset along with as much location data as possible. After clustering, we used domain knowledge and the player position field in our dataset to compare whether the clusters had eventually been narrowed down to position.

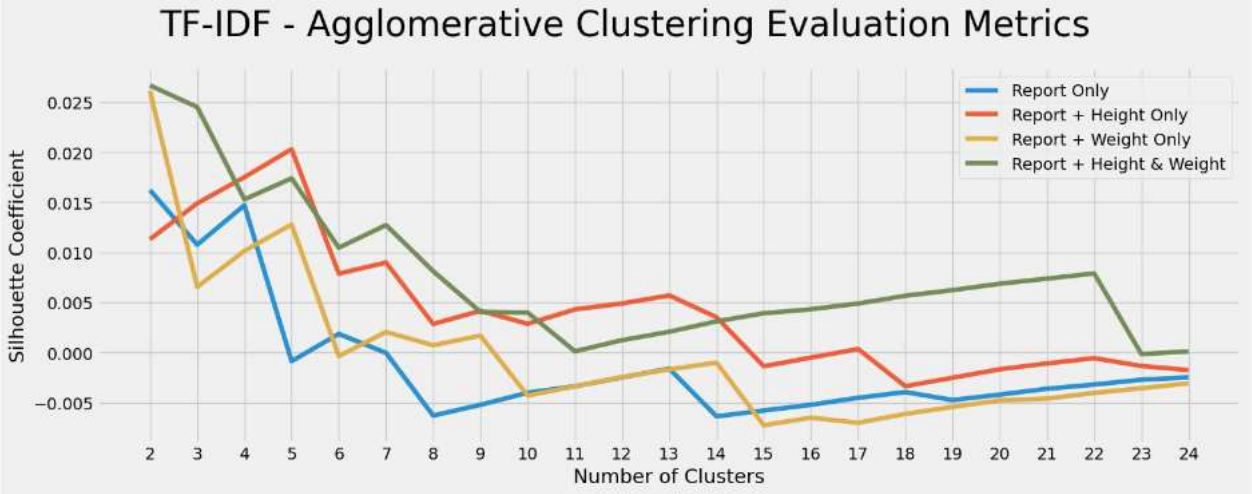
5.6 Unsupervised Evalation

5.6.1 TF-IDF Evaluation

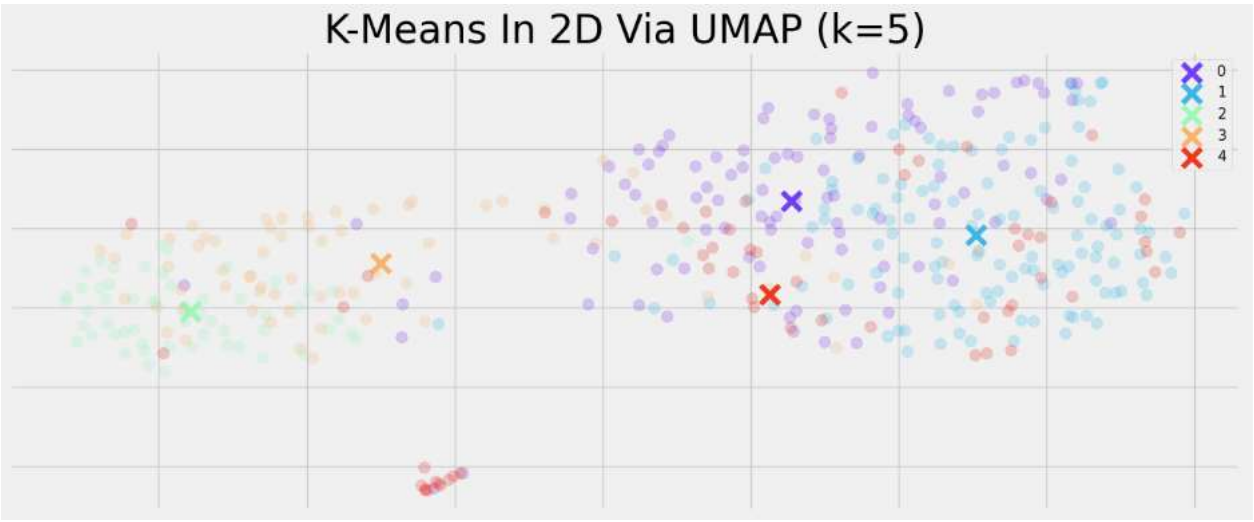
The TF-IDF model performed well for the being the least "advanced" of the models that we used. We evaluated the embedding model by testing the clustering values for different variations of embedding data including, using the report embedding only, appending height to the embedding data, appending weight to the embedding data, and appending height and weight to the embedding data. The results for each can be seen below.

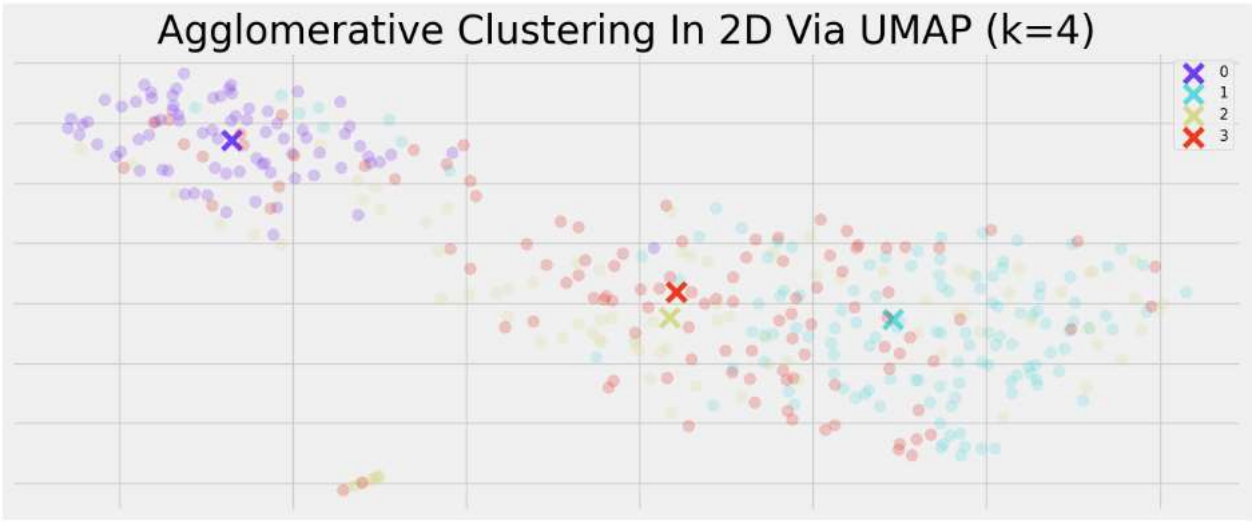


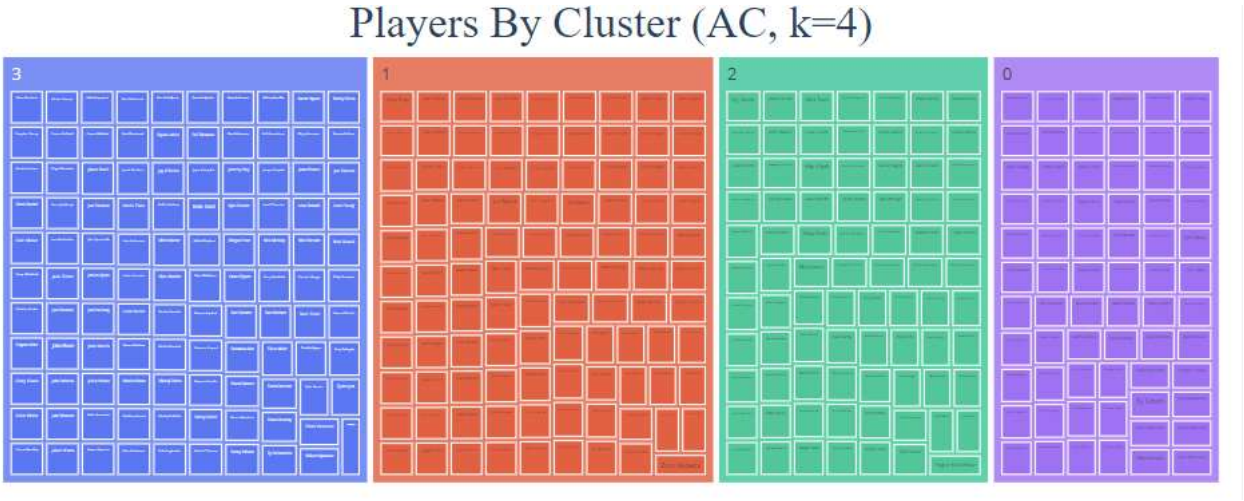
Using our elbow method and comparing the scores, one of the most promising cluster values was at k=5. We followed the same evaluation process with Agglomerative Clustering model as we did with K-Means model. The only difference is that SSE is only available for the K-Means model so we were unable to use that to make our judgement on the best k value. Here is the Silhouette Coefficient plotted using the Agglomerative Clustering model the same builds as we used for the K-Means model:



At agglomerative clustering, the best value for k proved to be 4. Once we have decided upon the final embedding and what values we include in it, we run the embeddings through k-means and the agglomerative clustering algorithms to deter bin the players into the cluster value we decided upon. Finally, plotting the clusters produced from these methods result in the following



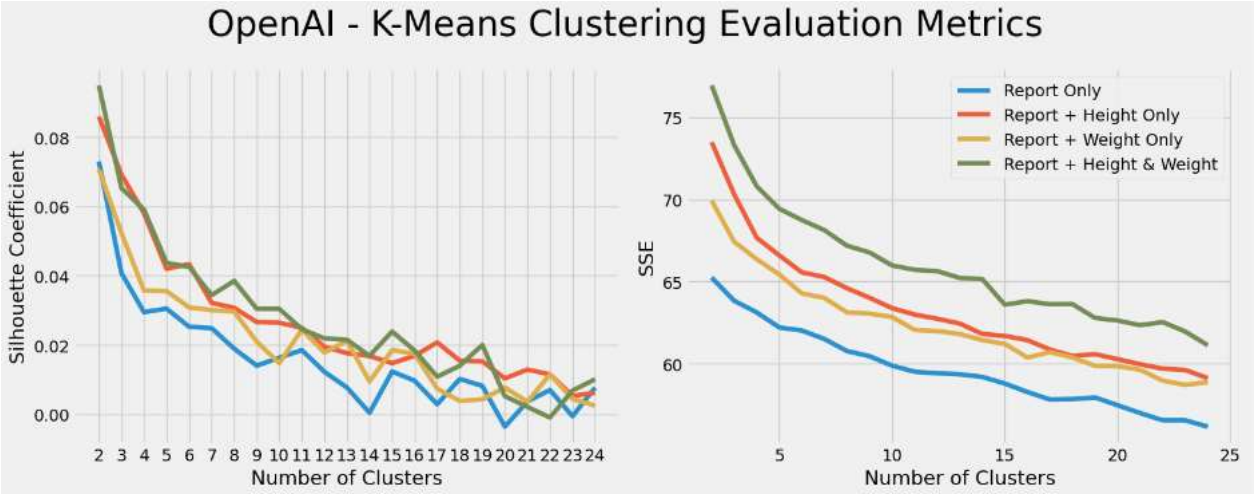




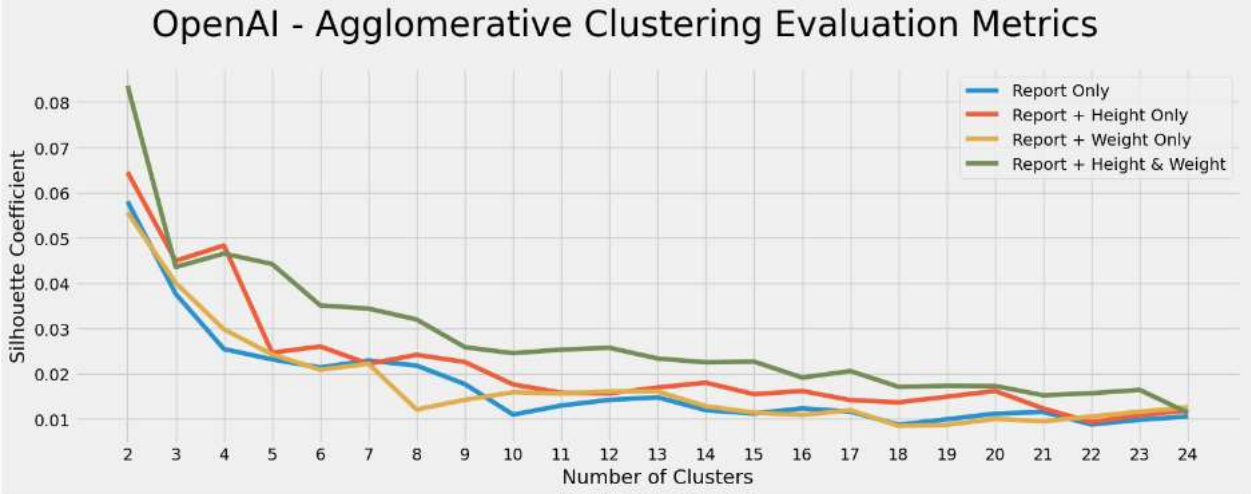
The resulting clusters appear to relatively evenly distributed but when viewing the 2D representation, you can see that there is a lot of overlap and the clusters are not well defined. These results were the best we were able to achieve with the TF-IDF embedding model so we decided to move on to our next option for embedding.

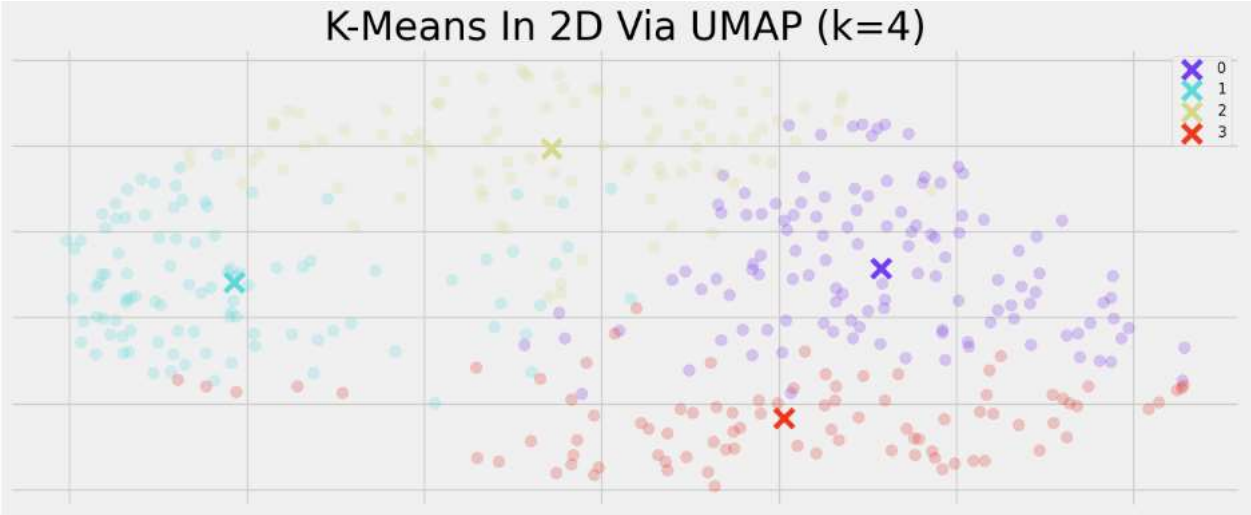
5.6.2 OpenAI Evaluation

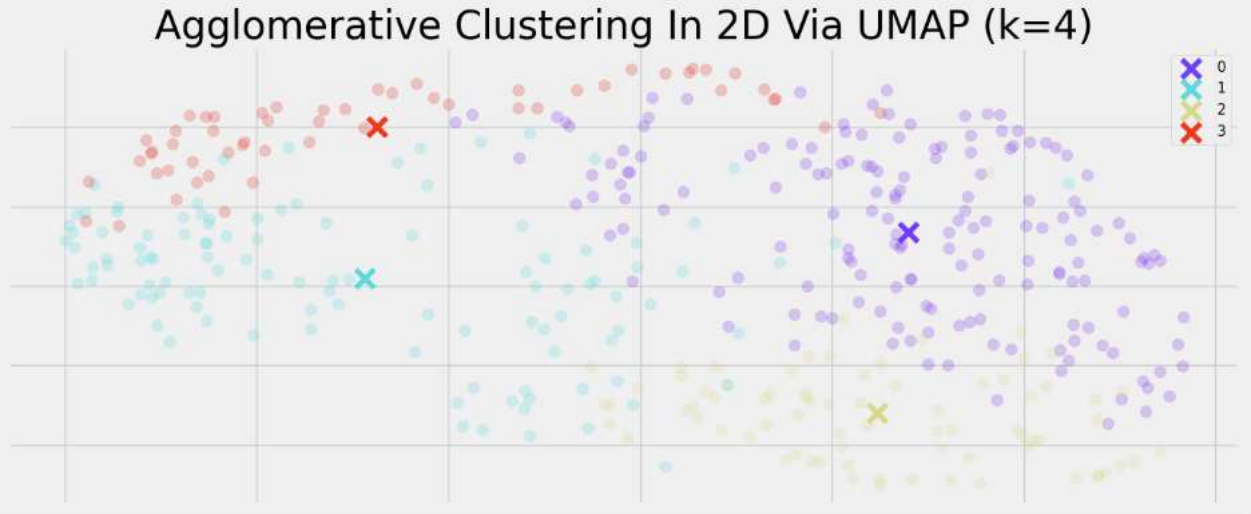
Evaluating the OpenAI embeddings were extremely easy following the setup that we did for the TF-IDF embeddings. The only difference in the process was how the emebdding vector was obtained. Since both embedding models output a vector, we were able to use the exact same process to compare how OpenAI's embeddings improved the process. We begin in the same way we did before, by attempting to determine what the best combination of embedding values we need to provide and deciding on a k value.



Interestingly, similar to TD-IDF, the most performant k value was 4. The main difference here is that the cluster definition was much more profound, most likely due to the much more advanced embedding model provided by OpenAI. The final results of the clustering can be seen here







Unsurprisingly, much better definition is achieved using the OpenAI embeddings. Clusters can be seen much better, and while there is not great separation between clusters, there is significantly less overlap. To see an example of what scouting reports are considered "similar", here are 2 examples of scouting reports for players placed in the same cluster from the OpenAI embeddings. Both reports were written by Corey Pronman of The Athletic:

- **Jacob Larsson**: "Larsson is a very interesting prospect who played at the top level at times for Frolunda this season, and was a top-four defenseman in Sweden's under-18 lineup. He's got a lot of physical tools, moves effortlessly in all directions and evades pressure pretty well for a big guy. He's not an overly gifted puck mover, but has solid to above-average skill. Defensively, Larsson can make some stops. He uses his body well to win battles and shows good effort in battles. He's not great in his own end, as he could clean up his reads, reactions and overall positioning. His skating allows him to make up for some errors."
- **Jeremy Roy**: "Roy has been a top prospect for years, and has consistently displayed his very impressive offensive upside. He's also the rare right-handed defenseman who plays the left side with higher frequency. He's a very good skater in all directions, with nice edge work and a quality top gear. His puck skills are clearly above average, and he's a coordinated puck handler who can lead a rush with ease. His puck movement is high-end, as he thinks quickly, is creative and doesn't force plays. Roy is slightly undersized and could tighten his gaps defensively at times, but he makes some defensive stops and battles well."

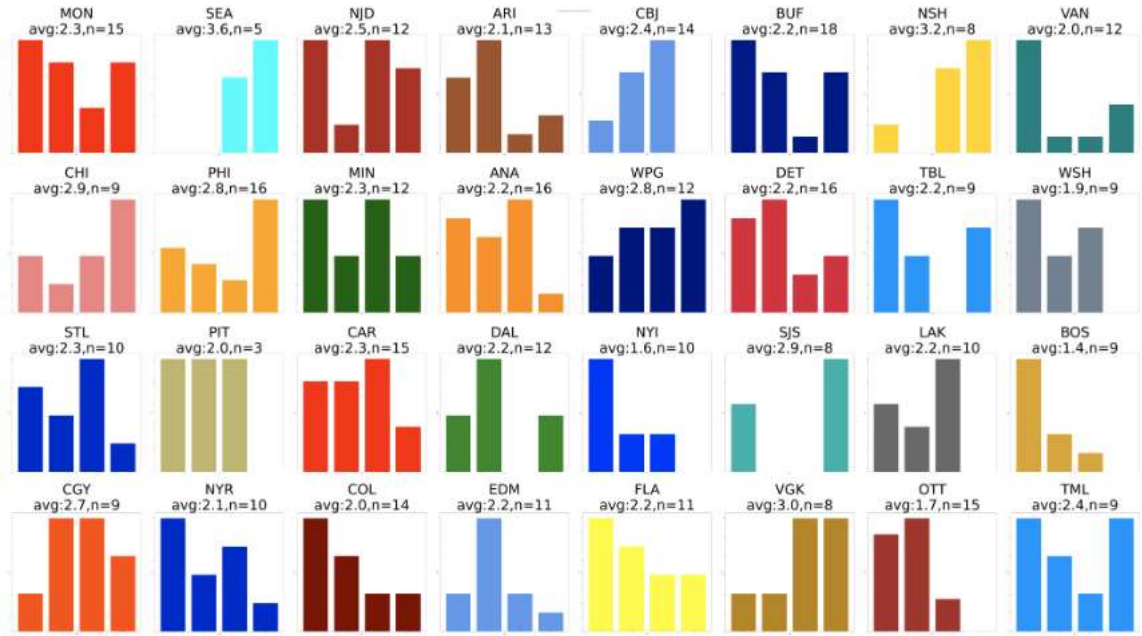
Below is a summary table showing the exact numbers we achieved with each modelling attempt and combination that we tried.

TF-IDF - K-Means	Best K	Silhouette Coefficient	SSE	OpenAI - K-Means	Best K	Silhouette Coefficient	SSE
Report Only	6	0.0188	290.1365	Report Only	3	0.0406	63.6241
Report + Height Only	4	0.0212	301.4021	Report + Height Only	4	0.0569	67.7662
Report + Weight Only	6	0.0116	292.3889	Report + Weight Only	5	0.0377	65.1025
Report + Height & Weight	5	0.0246	302.4338	Report + Height & Weight	4	0.0585	70.8705
TF-IDF - Agglomerative Clustering	Best K	Silhouette Coefficient	SSE	OpenAI - Agglomerative Clustering	Best K	Silhouette Coefficient	SSE
Report Only	6	0.0147		Report Only	4	0.0255	
Report + Height Only	4	0.0203		Report + Height Only	4	0.0483	
Report + Weight Only	6	0.0101		Report + Weight Only	3	0.041	
Report + Height & Weight	5	0.0245		Report + Height & Weight	4	0.0465	

As a final thought, we were interested in seeing if there were any patterns in how teams draft. There are, obviously, a great multitude lot of major caveats/issues to investigating that question using our results, including, but not limited to:

- Imperfect clustering
- Team's making certain picks based on which position they are drafting in
- Draft strategies changing from year to year
- The quality of the draft affecting which players are chosen any given year

But, it still seemed like a fun exercise. So, with all those problems in mind, here is the team by team breakdown of how teams have drafted over the past 8 years



6. Supervised Learning

Recall, the primary objective of the project was to determine the predictive value of using word embeddings behind scouting reports. Specifically, we wanted to answer how well the word embeddings predicted the player's draft position. Additional features such as the height, weight, and position of the player were also included as complementary features. The workflow closely follows the visualization for the NHL prospects data pipeline described in the feature engineering section. Since a majority of high quality scouting reports are private, solving this problem using publicly available scouting reports paves the way for a novel workflow for sports analytics.

Some of the algorithms used are Logistic Regression, K Nearest Neighbors Classification (KNN), and Random Forest Classification. These methods were adapted to the task of ordinal regression or ranking players against one another. Ordinal regression strikes a balance between the spectrum of a purely classification task and a purely regression task. The idea is to train a sequence of binary classifiers for each label. (In other words, train a binary classifier with a one-vs-rest strategy for label=0, then train another binary classifier for label=1, and so on until all the labels are exhausted).

The choice of methods span a diverse range of model families that are not as computationally intensive. Logistic Regression, an example of a probabilistic model, is one of the most basic classification algorithms, most commonly used as a binary classifier due to the output space ranging from 0 to 1. To create predictions, a typical cutoff point is established at 0.5 to separate the data into two classes. The most interesting parameter for scikit-learn's Logistic Regression is `C` : the regularization parameter to limit overfitting the training data. Another model is KNN which chooses the K nearest neighbors determined by proximity/distance and selects the most common label. A parameter of interest for scikit-learn's KNN is the parameter `n_neighbors` that determines how many neighbors to determine the label. Lastly, Random Forest Classification is an ensemble of tree-based models. Each tree is given a random sample of the data, called a bootstrap sample when trained. For making predictions, the ensemble of trees each votes on the predicted label and the label with the most votes is outputted from the Random Forest Classifier. Interesting parameters of scikit-learn's Random Forest Classifier is `n_estimators` , the number of trees which needed to be lowered due to the small dataset size, and `max_depth` , which prunes the trees at a certain depth to lower overfitting.

Hyperparameter tuning was conducted with a grid search that exhaustively searches the respective parameter grid for each model on a 3-fold cross validation splitting strategy. A major blocker with hyperparameter tuning was the extensive amount of time required for each training fit. Two major contributors that increased the training time was the lengthy process for creating word embeddings from the aforementioned LLMs and the increased model complexity.

6.1 Supervised Evaluation

The evaluation metrics for the supervised learning task are the accuracy, f1_score (macro strategy), recall (macro strategy), and precision (macro strategy). For each year, we selected the top 40 players in the first round so a majority of the players tend to have a draft position in the top 40. Even among the top 40 draft positions, there is a bit of variance.

The data pipeline splits when deciding between the choice for text preprocessing, either NLTK, BERT, or OpenAI, and another split on the choice for classification modeling, either Logistic Regression, KNN, or Random Forest Classification.

The table below shows the evaluation metrics for all 9 models. Values are the means of 3-fold cross validation value while the values in quotes represent the standard deviation.

Model	accuracy	f1	precision	recall
NLTK_log_reg	0.0158 (0.0017)	0.0158 (0.0017)	0.0188 (0.0020)	0.0182 (0.0022)
NLTK_KNN	0.0114 (0.0014)	0.0114 (0.0014)	0.0117 (0.0003)	0.0164 (0.0072)
NLTK_rand_forest	0.0148 (0.0018)	0.0148 (0.0018)	0.0183 (0.0025)	0.0182 (0.0040)
BERT_log_reg	0.0097 (0.0040)	0.0097 (0.0040)	0.0126 (0.0078)	0.0142 (0.0019)
BERT_KNN	0.0160 (0.0020)	0.0160 (0.0020)	0.0159 (0.0027)	0.0226 (0.0068)
BERT_rand_forest	0.0123 (0.0029)	0.0123 (0.0029)	0.0152 (0.0010)	0.0154 (0.0066)
OpenAI_log_reg	0.0297 (0.0216)	0.0297 (0.0216)	0.0327 (0.0298)	0.0378 (0.0199)
OpenAI_KNN	0.0121 (0.0040)	0.0121 (0.0040)	0.0146 (0.0069)	0.0179 (0.0061)
OpenAI_rand_forest	0.0301 (0.0193)	0.0301 (0.0193)	0.0283 (0.0193)	0.0346 (0.0190)

Recall the main focus was to see how predictive the different types of word embeddings performed on predicting the draft position. In the table below, we iteratively remove certain features and observe the downstream analysis results for the testing metrics. We are pleased to see that the word embeddings by themselves did outperform any permutation of feature selection. An unsurprising result is the predictive power of simply using Height and Weight alone to predict the draft position. This makes sense since the recent trends for top draft picks tend to be bigger and taller players.

OpenAI - Random Forest	accuracy	f1	precision	recall
Report Only	0.0508	0.0313	0.0326	0.0368
Position Only	0.0254	0.0020	0.0012	0.0078
Height Only	0.0169	0.0023	0.0014	0.0057
Weight Only	0.0085	0.0211	0.0157	0.0337
Height + Weight Only	0.0339	0.0124	0.0150	0.0117
Report + Position	0.0169	0.0224	0.0260	0.0223
Report + Height	0.0169	0.0049	0.0037	0.0074
Report + Weight	0.0169	0.0057	0.0045	0.0076
Report + Weight + Position	0.0085	0.0163	0.0159	0.0205
Report + Height + Position	0.0085	0.0156	0.0298	0.0148
Report + Height + Weight + Position	0.0339	0.0258	0.0192	0.0462

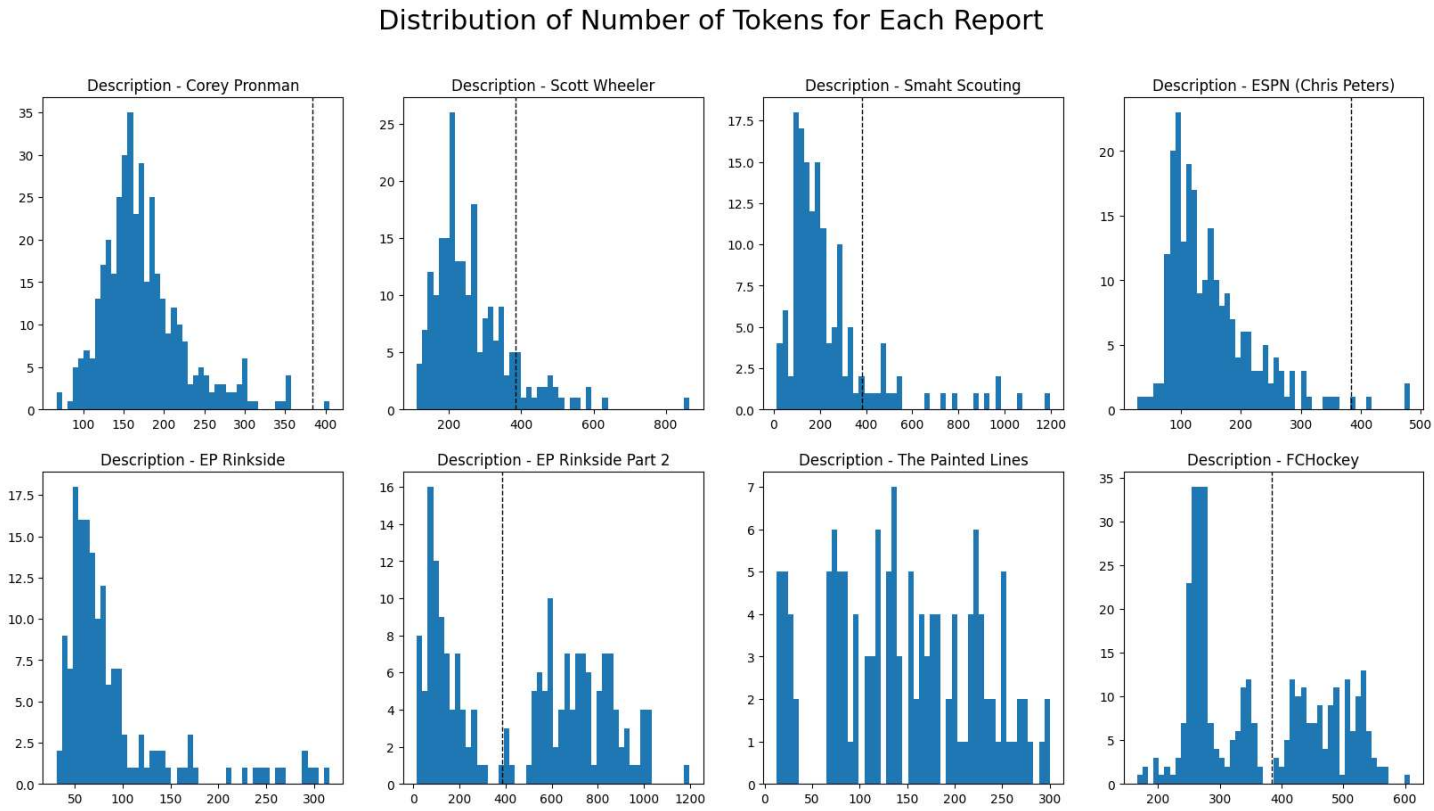
Note that the f1, precision, and recall metrics use a macro strategy. The choice to use the macro strategy instead of the micro strategy is due to the class imbalance within the draft position labels. From literature there is the game of balancing between optimizing precision or optimizing recall. This tradeoff is confirmed with our model results and best seen in the better performing models. There will either be a stronger inclination towards precision or recall. However, the poorer performing models have low values across all evaluation metrics.

6.2 Supervised Failure Analysis

Initially, the supervised learning task was tackled as a purely classification task. Due to the large number of labels (drafted position ranged from about 1-70) being predicted compounded with a small number of data samples, models typically performed poorly. We alleviated this issue by treating each individual report for a player as a single sample instead of aggregating all reports for a single player together.

One major issue was overfitting. To resolve this issue, we included testing regularization parameters so that the models generalized better to unknown data. This helped the evaluation metrics to a moderate extent. However, we believe the biggest contributing factor to overfitting was the overwhelming number of features in the latent embedding space. For BERT embeddings, the embedding space was 784 while OpenAI's Chapt-GPT model had an embedding space of 4096. A future improvement is to use dimension reduction techniques on the word embeddings themselves. However, a better idea from related works would be to use the large language models to calculate a sentiment analysis score. This would simply be an extra numerical feature instead of using the entire word embedding.

Another failure was the realization that the token limit for documents being passed into the BERT models was at most 512 tokens. One preprocessing step previously taken was joining all the reports for a single player into one large report totaling to at least a couple thousand tokens. For individual tokens, a histogram visualization on the distribution of the number of tokens reveals most reports are under the limit. Therefore, preprocessing steps were modified to process each report individually.



7. Discussion

Part A

A major lesson with the supervised learning task was the complexity of designing and implementing a complete end-to-end data pipeline and the difficulty of predicting labels with word embeddings even when using state-of-the-art large language models. Initially, the data pipeline was overengineered with scripts to be run on the command line. This allowed for a sequence of commands to be run described in a makefile. However, due to the project requirements and the increased project complexity, this came as a sunk cost. Moving past this blocker, was the extremely long training times for each model. Even a simple model such as logistic regression took several hours to undergo a 3-fold test/train split followed by an additional 3-fold cross-validation split to determine the best hyperparameters. A major challenge was overfitting and the extremely low values for testing metrics. We overcame the issue of overfitting by tuning regularization parameters, but the testing metrics still stayed low. This wasn't surprising due to the lack of data samples and high embedding feature space. Funnily enough, a model trained only on the height and weight features performed incredibly well. This makes sense since the recent trend for top picks tends to be bigger and taller players. It was even observed to perform better than the word embeddings by themselves, but this proved false as we had a data leakage issue where the training set was spilling into the testing set. We are proud of our current work and progress towards a very difficult problem in the sports analytics space. Possible future work include (1) performing dimension reduction techniques on the word embeddings, (2) use a state-of-the-art convolutional neural network model (e.g., AlexNet) on the word embeddings or (3) scrapping the word embeddings and instead using the large language models for a sentiment analysis score. These options require additional time and, for the case of using a pre-trained deep learning model, additional computing resources.

Part B

One of the major takeaways we had after the unsupervised learning aspect was how difficult of a problem it was to parse out intent from the scouting reports. The embedding and clustering algorithms picked up on nationality and then position and focused on clustering based on those values. We were able to move past nationality but the position aspect was very hard. As a consequence, I was surprised with just how easily the embedding/clustering algorithms could identify position just based on the scouting reports. Even the less sophisticated TF-IDF was able to parse out position with very little effort. We attempted to overcome this by removing the position values from the reports we were reading and we even tried at one point to break up the data we viewed into just one position at a time. Removing the position had some success but the separate models based on position never converged to a useable k value for any algorithm we tried, most likely due to a lack of data. With some more time, I think we might be able to figure this problem out. Another big blocker to us being able move past this was the amount of data that we had. It was a lot of work to even get the data that we have now and I suspect for better results, we may need to get much more data. Unfortunately, we were bounded by the people who are making public scouting reports for NHL prospects. The good news is that this group of people is growing and their work is becoming more popular. We have hopes that with some more time, there may be enough data to revisit this and do a deeper dive.

8. Ethical Considerations

Some things to keep in mind when viewing our data is that there is almost always an exception or outlier who bucks the trends. Every year in the NHL draft, or any professional sports draft for that matter, there is a player who slips under the radar and beats the odds to make it to the league. This is important because that player who beats the odds must deal with a lot of adversity as a result of not getting the opportunities which may arise from being a high draft pick. High draft picks are given every opportunity to succeed, and this does not carry over to later round picks. As a result, when viewing draft predictions such as what we provide in part A, while there is some utility in the predictions themselves, the predictions are based on imperfect selections in the past which may have hidden bias or flawed reasoning behind the picks made/the scouting reports written about said player. This may end up with some players receiving preferential treatment and leaving out others and harming those player's chance at success in the process. A very similar issue arises with the clustering data in part B, but perhaps more directly. That "outlier" player may physically manifest itself in the clustering data as an outlier which may be placed in a poor cluster as a necessity to satisfy the criteria of the clustering algorithm. As a result, that specific player may not be considered relevant to a team, even though they might be of similar or greater talent than another player.

Options to reduce this bias and provide a better picture to the user of the models which we propose would be twofold. First, adding more data. Including players beyond the top 40 players in each draft might help to paint a better picture of the entirety of players available in each draft. It also might help to bucket lower ranked players, since our models now might struggle to do so. Second, adding some sort of uncertainty measure to our data. For instance, making some changes to the supervised results to relay some sort of uncertainty in whatever draft position is predicted via the model. A similar sort of measure could be shown in the model to perhaps relay distance of a point from the centroid and the next closest centroid to provide context with what label is given to a player and how close they were to a different label.

9. Statement of Work

Ryan DeSalvio was in charge of topic selection, data collection, exploratory data analysis, supervised learning code, and report write-up. The specific report sections that Ryan led were: Data Source, Unsupervised Learning and Ethical Considerations sections.

Quoc-Huy Nguyen was in charge of draft proposal write-up, exploratory data analysis, supervised learning code, and report write-up. The specific report sections that Quoc-Huy led were: Introduction, Related Work, Feature Engineering, and Supervised Learning sections.

10. References

A massive thank you to everyone writing prospect reports in the public space. Their works was obviously invaluable to this project.

- "2019 NHL draft rankings: Chris Peters' final prospect board." ESPN, ESPN Internet Ventures, 2019, www.espn.com/nhl/insider/story/_/id/26942547/2019-nhl-draft-top-100-prospect-rankings-peters-final-draft-board.
- "2020 NHL draft rankings: Final top 100 prospects in the class." ESPN, ESPN Internet Ventures, 2020, www.espn.com/nhl/insider/story/_/id/29994322/2020-nhl-draft-rankings-final-top-100-prospects-class-chris-peters-plus-position-rankings.
- "Cam Robinson's 2018 Draft Rankings: Top 130 Final Edition." Dobber Prospects, 15 June 2018, dobblerprospects.com/2018/06/15/cam-robinsons-2018-draft-rankings-top-130-final-edition/.
- "Cam Robinson's 2019 NHL Draft Rankings - April 2019." Dobber Prospects, 15 May 2019, dobblerprospects.com/2019/05/15/cam-robinsons-2019-nhl-draft-rankings-april-2019/.
- "Connor McDavid, Jack Eichel lead list of top 100 prospects for 2015 NHL draft." ESPN, 2015, insider.espn.com/nhl/insider/story/_/id/12819334/connor-mcdavid-jack-eichel-lead-list-top-100-prospects-2015-nhl-draft.
- "Elite Prospects 2020 Draft Guide." Elite Prospects, 2020, www.eliteprospects.com/2020draftguide.
- "Elite Prospects 2021 Draft Guide." Elite Prospects, 2021, www.eliteprospects.com/2021draftguide.
- "Elite Prospects 2022 Draft Guide." Elite Prospects, 2022, www.eliteprospects.com/2022draftguide.
- Hendricks, Benjamin. Sports Analytics with Natural Language Processing: Using Crowd Sentiment to Help Pick Winners in Fantasy Football. Diss. Harvard University, 2022 https://dash.harvard.edu/handle/1/37371598.
- "NHL Chris Peters' top 80 prospects in the 2018 NHL draft class big board." ESPN, ESPN Internet Ventures, 2018, www.espn.com/nhl/story/_/id/23782880/nhl-chris-peters-top-80-prospects-2018-nhl-draft-class-big-board.
- "NHL draft 2021 rankings: Top 50 prospects in the class, plus scouting reports." ESPN, ESPN Internet Ventures, 2021, www.espn.com/nhl/insider/insider/story/_/id/31811931/nhl-draft-2021-rankings-top-50-prospects-class-plus-scouting-reports.
- "NHL draft prospects ranking 2021: Corey Pronman's final top 151." The Athletic, 15 June 2021, theathletic.com/2620093/2021/06/15/nhl-draft-prospects-ranking-2021-corey-pronmans-final-top-151/.
- "NHL draft prospect rankings 2022." The Athletic, 5 July 2022, theathletic.com/3393872/2022/07/05/nhl-draft-prospect-rankings-2022/.
- "NHL draft top 100 prospects ranking." The Athletic, 6 June 2022, theathletic.com/3306659/2022/06/06/nhl-draft-top-100-prospects-ranking/.

- "NHL draft rankings 2023: Bedard, Pronman's top 151 prospects." The Athletic, 30 May 2023, theathletic.com/4538998/2023/05/30/nhl-draft-rankings-2023-bedard-pronman/.
- "NHL draft 2023 ranking: Wheeler's top 100 prospects." The Athletic, 13 June 2023, theathletic.com/4575346/2023/06/13/nhl-draft-2023-ranking-wheeler/.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2014.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2015.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2016.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2017.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2018.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2019.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2020.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2021.
- "NHL Entry Draft Rankings." NHL Entry Draft, nhlentrydraft.com/rankings/. 2022.
- "NHL Top 100 Draft Prospect Rankings." ESPN, 2017, www.espn.com/nhl/insider/story/_/id/19416323/nhl-top-100-draft-prospect-rankings.
- "Patrik Laine, Auston Matthews lead list of top 100 prospects for 2016 NHL draft." ESPN, 2016, www.espn.com/nhl/insider/story/_/id/15506495/patrik-laine-auston-matthews-lead-list-top-100-prospects-2016-nhl-draft.
- "Patrik Laine, Auston Matthews lead list of top 100 prospects for 2016 NHL draft." ESPN, 2016, www.espn.com/nhl/insider/story/_/id/15506495/patrik-laine-auston-matthews-lead-list-top-100-prospects-2016-nhl-draft.
- Petrosilli, Brandon. "2022 NFL Mock Draft Analysis." LinkedIn, 28 April 2022, https://www.linkedin.com/pulse/2022-nfl-mock-draft-analysis-brandon-petrosilli/.
- "Pronman's 2018 NHL Draft Board." The Athletic, 21 May 2018, theathletic.com/342438/2018/05/21/pronmans-2018-nhl-draft-board/.
- "Pronman's 2019 NHL draft board: Top 107 prospects." The Athletic, 21 May 2019, theathletic.com/970746/2019/05/21/pronmans-2019-nhl-draft-board-top-107-prospects/.
- Pronman's 2020 NHL draft board: Top 122 prospects." The Athletic, 16 June 2020, theathletic.com/1769140/2020/06/16/pronmans-2020-nhl-draft-board-top-122-prospects/.
- "Wheeler: Final ranking for the 2018 NHL draft's top 100 prospects." The Athletic, 8 May 2018, theathletic.com/342459/2018/05/08/wheeler-final-ranking-for-the-2018-nhl-drafts-top-100-prospects/.
- "Wheeler: Final ranking for the 2019 NHL draft's top 100 prospects." The Athletic, 6 May 2019, theathletic.com/947751/2019/05/06/wheeler-final-ranking-for-the-2019-nhl-drafts-top-100-prospects/?source=dailyemail%20target=
- "Wheeler: Final ranking for the 2020 NHL draft's top 100 prospects." The Athletic, 1 June 2020, theathletic.com/1736440/2020/06/01/wheeler-final-ranking-for-the-2020-nhl-drafts-top-100-prospects/.
- "Wheeler: NHL draft's top 100 prospects for 2021 sees Michigan players top the ranking." The Athletic, 22 June 2021, theathletic.com/2569641/2021/06/22/wheeler-nhl-drafts-top-100-prospects-for-2021-sees-michigan-players-top-the-ranking/.
- Zaire, Chris. “Evaluating the 2020 NFL Draft Class, Using NLP.” Medium, 10 April 2020, towardsdatascience.com/evaluating-the-2020-nfl-draft-class-using-nlp-dc77513cbffe.