Training Machine Learning Models to Recognize and Classify *Magic: The Gathering* Preconstructed Commander Decks

Devdan Ferguson

Introduction

Magic: the Gathering is one of the most prolific trading card games played in the world, with an estimated 50 million players worldwide (source: https://investor.hasbro.com/magic-gathering). Magic (or MTG) is built around a core deck-building system; players assemble cards with synonymous and synergistic abilities that complement one another, and then include the resources designed to best enable their deck to achieve the route they have chosen to attempt to win the game. The core of the game revolves around the "color wheel:" Magic's cards can be any combination (or none) of five colors: white, blue, black, red, and green, each of which have their own identities associated with them mechanically. Mana is the resource of the game, being spent to play cards and activate abilities, and can be any of these five colors, or colorless. Magic can be played in several formats, the most popular of which is Commander (or EDH), which sees a player have an eponymous "commander" for their deck, whose colors dictate what cards can be used in the deck. The decks can also only have one copy of each card (barring a few exceptions).

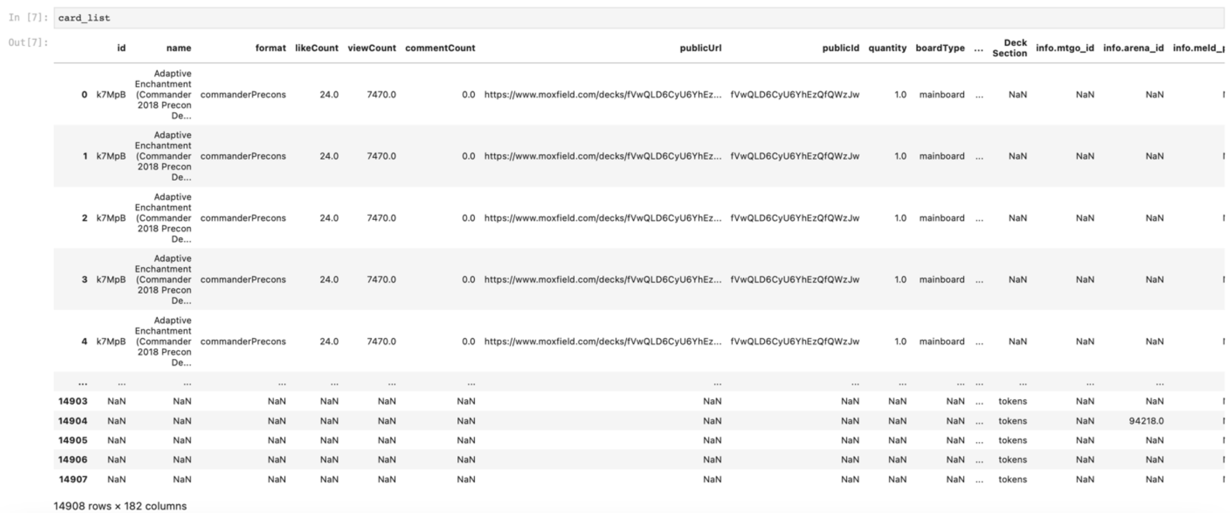
Although several formats have had them, EDH is also notable for the preconstructed, or "precon" decks released for the format. At what was originally a rate of once per year, but is now once per release, with several per year being released. Preconstructed decks are designed to be approachable, and generally represent common archetypes a player might build their own deck around. On Moxfield (a popular half-social media, half-database that users can submit their deck lists to), there is documentation of every preconstructed deck ever released.
At https://github.com/Westly/CommanderPrecons/blob/main/All_Precons_2024-11-11.csv, a web scraped CSV file from Moxfield of every card included in every preconstructed deck can be found, and in it lies rich trends and patterns of deckbuilding to be studied, which machine learning can take advantage of.

Within this study, two machine learning models are applied to learn and exploit patterns and similarities between cards. This is used to the end of training these models to recognize what deck each card can be taken from. Extending from this, this can be used to produce better recommendation systems and deck-building assistant tools for MTG fans looking to dig deeper into the tens of thousands of cards in existence to grow the power and fun of their collection.

Data

Data Introduction:

After importing our csv file as the Pandas DataFrame 'card_list', we can begin to explore it:



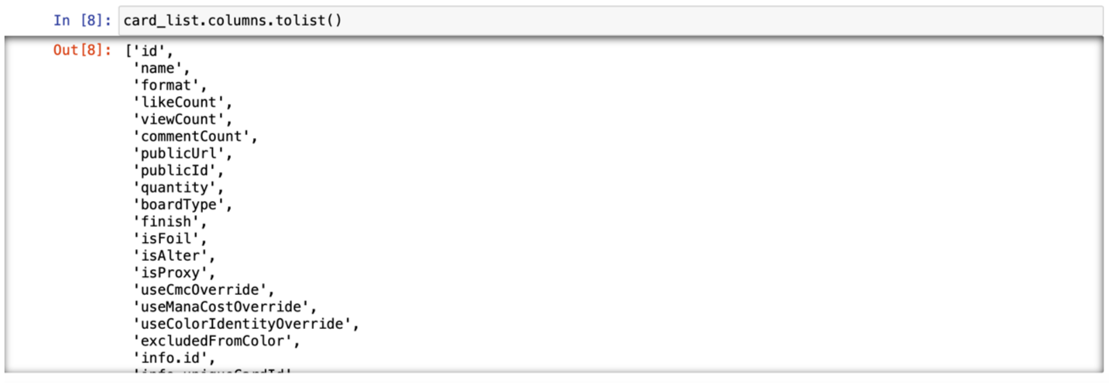
```
In [7]: card_list
```

Out[7]:

	id	name	format	likeCount	viewCount	commentCount	publicUrl	publicId	quantity	boardType	...	Deck Section	info.mtgo_id	info.arena_id	info.meld_
0	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	commanderPrecons	24.0	7470.0	0.0	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	1.0	mainboard	...	NaN	NaN	NaN	
1	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	commanderPrecons	24.0	7470.0	0.0	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	1.0	mainboard	...	NaN	NaN	NaN	
2	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	commanderPrecons	24.0	7470.0	0.0	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	1.0	mainboard	...	NaN	NaN	NaN	
3	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	commanderPrecons	24.0	7470.0	0.0	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	1.0	mainboard	...	NaN	NaN	NaN	
4	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	commanderPrecons	24.0	7470.0	0.0	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	1.0	mainboard	...	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
14903	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	tokens	NaN	NaN	
14904	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	tokens	NaN	94218.0	
14905	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	tokens	NaN	NaN	
14906	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	tokens	NaN	NaN	
14907	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	tokens	NaN	NaN	

14908 rows × 182 columns

Notably, the DataFrame has 182 columns/features:

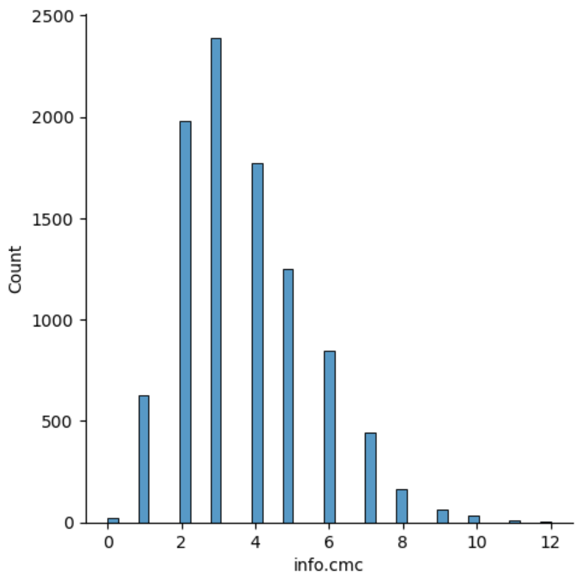


```
In [8]: card_list.columns.tolist()
```

```
Out[8]: ['id',
         'name',
         'format',
         'likeCount',
         'viewCount',
         'commentCount',
         'publicUrl',
         'publicId',
         'quantity',
         'boardType',
         'finish',
         'isFoil',
         'isAlter',
         'isProxy',
         'useCmcOverride',
         'useManaCostOverride',
         'useColorIdentityOverride',
         'excludedFromColor',
         'info.id',
```

Basic Visual Analysis

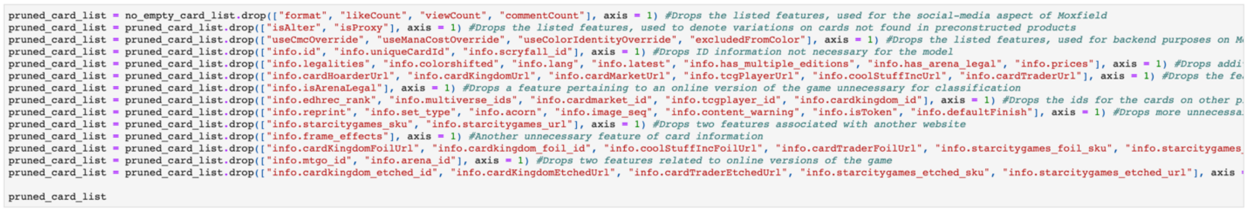
In Magic, cards are played by spending mana on them. All cards have a mana cost, and this value is included in the feature "info.cmc." An essential part of constructing a deck is producing a balanced "curve," which, when visualized, should form a bell curve:



Additional exploration and analysis of the data set can be found in the repository.

Preprocessing

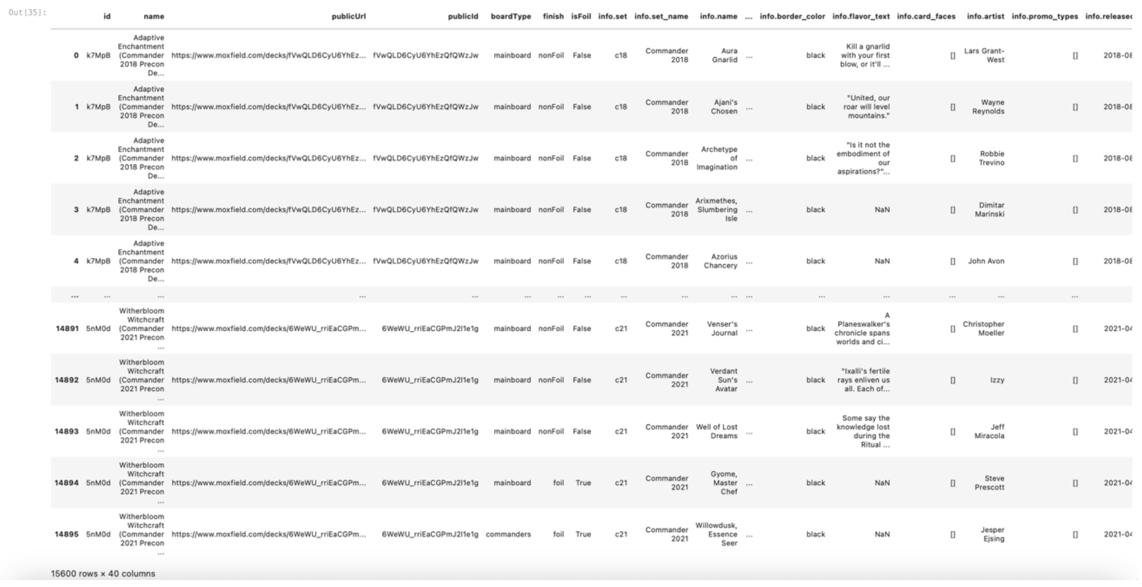
The most important step in data preparation for this data set is feature engineering. First, we remove any rows and columns from the DataFrame that have no values in them. As this data was scrubbed via the public tool Moxfield (which in-turn pulls data from several other platforms, including but not limited to Scryfall, EDHREC, etc), unnecessary features for the purposes of this project are dropped from the DataFrame:



```
pruned_card_list = no_empty_card_list.drop(["format", "likeCount", "viewCount", "commentCount"], axis = 1) #Drops the listed features, used for the social-media aspect of Moxfield
pruned_card_list = pruned_card_list.drop(["isAlter", "isProxy"], axis = 1) #Drops the listed features, used to denote variations on cards not found in preconstructed products
pruned_card_list = pruned_card_list.drop(["useCmcOverride", "useManaCostOverride", "useColorIdentityOverride", "excludedFromColor"], axis = 1) #Drops the listed features, used for backend purposes on M
pruned_card_list = pruned_card_list.drop(["info.id", "info.uniqueCardId", "info.scryfall_id"], axis = 1) #Drops ID information not necessary for the model
pruned_card_list = pruned_card_list.drop(["info.legalities", "info.colorshifted", "info.lang", "info.latest", "info.has_multiple_editions", "info.has_arena_legal", "info.prices"], axis = 1) #Drops addi
pruned_card_list = pruned_card_list.drop(["info.cardHoarderUrl", "info.cardKingdomUrl", "info.cardMarketUrl", "info.tcgPlayerUrl", "info.coolStuffIncUrl", "info.cardTraderUrl"], axis = 1) #Drops the fe
pruned_card_list = pruned_card_list.drop(["info.isArenaLegal"], axis = 1) #Drops a feature pertaining to an online version of the game unnecessary for classification
pruned_card_list = pruned_card_list.drop(["info.edhrec_rank", "info.multiverse_ids", "info.cardmarket_id", "info.tcgplayer_id", "info.cardkingdom_id"], axis = 1) #Drops the ids for the cards on other p
pruned_card_list = pruned_card_list.drop(["info.reprint", "info.set_type", "info.acorn", "info.image_seq", "info.content_warning", "info.isToken", "info.defaultFinish"], axis = 1) #Drops more unnecessa
pruned_card_list = pruned_card_list.drop(["info.starcitygames_sku", "info.starcitygames_url"], axis = 1) #Drops two features associated with another website
pruned_card_list = pruned_card_list.drop(["info.frame_effects"], axis = 1) #Another unnecessary feature of card information
pruned_card_list = pruned_card_list.drop(["info.cardKingdomFoilUrl", "info.cardkingdom_foil_id", "info.coolStuffIncFoilUrl", "info.cardTraderFoilUrl", "info.starcitygames_foil_sku", "info.starcitygames_
pruned_card_list = pruned_card_list.drop(["info.mtgo_id", "info.arena_id"], axis = 1) #Drops two features related to online versions of the game
pruned_card_list = pruned_card_list.drop(["info.cardkingdom_etched_id", "info.cardKingdomEtchedUrl", "info.cardTraderEtchedUrl", "info.starcitygames_etched_sku", "info.starcitygames_etched_url"], axis 

pruned_card_list
```

Next, we will drop each "token" card, as well as another similar one called emblems, as they will interfere with us having the right number of cards directed to each deck. After this, due to the Secret Lair commander decks, which come with bonus cards, we have 7 cards listed under the “sideboard” in the ‘boardType’ feature, and are, following research, confirmed to be bonus cards not part of the main deck lists. We remove these from the DataFrame and, finally, iterating over the “quantity” feature, for every card of which there are multiple included in the deck (only Basic Lands, a card type for which the “singleton” rule of the format does not apply), we duplicate the rows that many times, then drop the quantity metric. Finally, we are left with are pruned and cleaned data set:



Out[35]:

	id	name	publicUrl	publicId	boardType	finish	isFoil	info.set	info.set_name	info.name	...	info.border_color	info.flavor_text	info.card_faces	info.artist	info.promo_types	info.release
0	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	mainboard	nonFoil	False	c18	Commander 2018	Aura Gnarlid	...	black	Kill a gnarlid with your first blow, or it'll ...	[]	Lars Grant-West	[]	2018-0
1	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	mainboard	nonFoil	False	c18	Commander 2018	Ajani's Chosen	...	black	"United, our roar will level mountains."	[]	Wayne Reynolds	[]	2018-0
2	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	mainboard	nonFoil	False	c18	Commander 2018	Archetype of Imagination	...	black	"Is it not the embodiment of our aspirations?"...	[]	Robbie Trevino	[]	2018-0
3	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	mainboard	nonFoil	False	c18	Commander 2018	Arixmethes, Slumbering Isle	...	black	NaN	[]	Dimitar Marinski	[]	2018-0
4	k7MpB	Adaptive Enchantment (Commander 2018 Precon De...	https://www.moxfield.com/decks/fVwQLD6CyU6YhEz...	fVwQLD6CyU6YhEzQfQWzJw	mainboard	nonFoil	False	c18	Commander 2018	Azorius Chancery	...	black	NaN	[]	John Avon	[]	2018-0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
14891	5nM0d	Witherbloom Witchcraft (Commander 2021 Precon ...	https://www.moxfield.com/decks/6WeWU_rriEaCGPm...	6WeWU_rriEaCGPmJ2l1e1g	mainboard	nonFoil	False	c21	Commander 2021	Venser's Journal	...	black	A Planeswalker's chronicle spans worlds and ci...	[]	Christopher Moeller	[]	2021-0
14892	5nM0d	Witherbloom Witchcraft (Commander 2021 Precon ...	https://www.moxfield.com/decks/6WeWU_rriEaCGPm...	6WeWU_rriEaCGPmJ2l1e1g	mainboard	nonFoil	False	c21	Commander 2021	Verdant Sun's Avatar	...	black	"Ixalli's fertile rays enliven us all. Each of...	[]	Izzy	[]	2021-0
14893	5nM0d	Witherbloom Witchcraft (Commander 2021 Precon ...	https://www.moxfield.com/decks/6WeWU_rriEaCGPm...	6WeWU_rriEaCGPmJ2l1e1g	mainboard	nonFoil	False	c21	Commander 2021	Well of Lost Dreams	...	black	Some say the knowledge lost during the Ritual ...	[]	Jeff Miracola	[]	2021-0
14894	5nM0d	Witherbloom Witchcraft (Commander 2021 Precon ...	https://www.moxfield.com/decks/6WeWU_rriEaCGPm...	6WeWU_rriEaCGPmJ2l1e1g	mainboard	foil	True	c21	Commander 2021	Gyome, Master Chef	...	black	NaN	[]	Steve Prescott	[]	2021-0
14895	5nM0d	Witherbloom Witchcraft (Commander 2021 Precon ...	https://www.moxfield.com/decks/6WeWU_rriEaCGPm...	6WeWU_rriEaCGPmJ2l1e1g	commanders	foil	True	c21	Commander 2021	Willowdusk, Essence Seer	...	black	NaN	[]	Jesper Ejsing	[]	2021-0

15600 rows × 40 columns

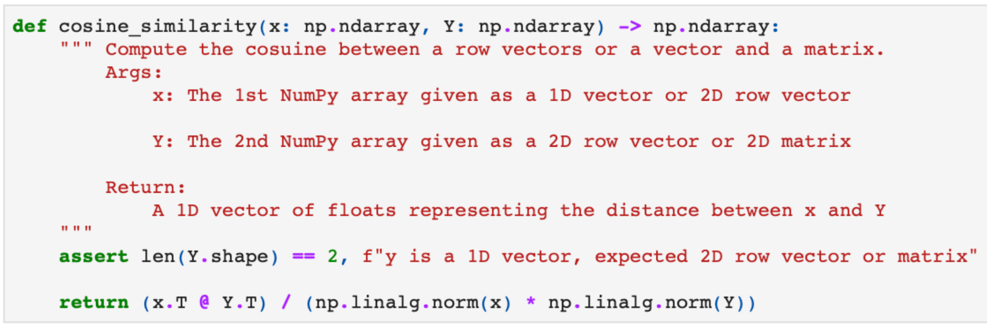
Next, for testing purposes, we can prepare a generalized method to generate our X and y, by setting the y equal to the name of the deck the card is found in, and the X equal to every other feature. While we are here, we drop columns that hold information equivalent to the name of the commander deck. This functions as an epilogue of sorts for feature engineering, dropping information that will interfere with testing. The reason we did not do this during feature engineering/data pruning is because these are still essential defining parts of a card.

After this we produce our get_preprocessed_data() method, again generalized. The method takes a dataframe as input, which is first passed to the prepare_X_y() method, generating our X and y. Column-by-column in X, we run algorithms like OneHotEncoding, TF-IDF (a technique I learned for vectorizing text documents in ITCS 3162), and Standardizing, with missing values being treated as the min - (2 * the max) (to keep them comfortably out of range). We then use OneHotEncoder to convert the labels y into one-hot encoding, for the sake of numerizing the data later. Then, we train-test-split, following the standards established throughout the semester. These techniques are derived from the assignment "[Homework] Neural Networks." Finally, we also return the one-hot encoder used during preprocessing to de-transform y-values later.

Method

The first algorithm we are using is **k-Nearest Neighbors**. Although this a simpler machine learning algorithm, we implement it for two key reasons. 1), runtime, as, knowing the size that is to be expected on the greater dataset, the 2020 MacBook Air this is being tested on can not be expected to process that much data in a feasible amount of time. 2), kNN executes the simple similarity logic we seek in this problem – decks are assortments of cards. If the majority of the k nearest cards are classed to a specific deck, then the idea of synergistic similarities between cards would tell us to assume the card we are classifying would be in the same deck as them.

First, we will produce our distance metric to be used for k-Nearest Neighbors. Unlike typical kNN, we will be implementing a slightly different distance metric than the usual Euclidean distance. Cosine similarity is the typical distance measure used when we are comparing text bodies converted using TF-IDF, as these vectors can balloon in length (or in our case, feature counts). This can make Euclidean distance unfeasible in our case. Thus, even at the sacrifice of quality, for the sake of making the program able to finish executing, we implement cosine similarity.



```python
def cosine_similarity(x: np.ndarray, Y: np.ndarray) -> np.ndarray:
    """ Compute the cosuine between a row vectors or a vector and a matrix.
        Args:
            x: The 1st NumPy array given as a 1D vector or 2D row vector

            Y: The 2nd NumPy array given as a 2D row vector or 2D matrix

        Return:
            A 1D vector of floats representing the distance between x and Y
    """
    assert len(Y.shape) == 2, f"y is a 1D vector, expected 2D row vector or matrix"

    return (x.T @ Y.T) / (np.linalg.norm(x) * np.linalg.norm(Y))
```

Next, we implement our accuracy metric, taken from "[Homework] Linear Algebra and KNN." Our kNN class itself is also derived from the code in said homework assignment however there are a few modifications, namely that we return a transformed form of the prediction; so, say, [0,0,0,1,0] would become 3) wherein the first (and only) index where 1 is found for each row of the OHE-formed 2-dimensional labels replaces that row in its entirety, transforming the data to a 1-dimensional form. Later on, we create use a method to transform y_tst for the same application. There is also a modification in the predict() method that, if cosine_similarity is the distance metric, we instead take the last k values of topk, as a higher similarity score would be equivalent to a nearer distance. Rather than testing kNN manually, we use an iterative for loop, starting with 1-Nearest Neighbor and traveling towards a ceiling.

The second algorithm we are using is **Logistic Regression**, using the softmax algorithm. This is the most complex algorithm that can handle multi-class scenarios we have access to that will not encounter runtime failings (that is, neural networks). Additionally, we chose this model to apply a gradient-descent-enabled model to this data, to explore how more computationally complex processes will interact with it.

First, we need our sigmoid activation function, taken directly from my implementation in "[Homework] Logistic Regression." Next, our Softmax equation and generalized Negative Log Likelihood (NLL) implementations, also from the same homework, as well as its batch-procurement method. Finally, from the same homework assignment, we transport our SoftmaxRegression algorithm.

Results

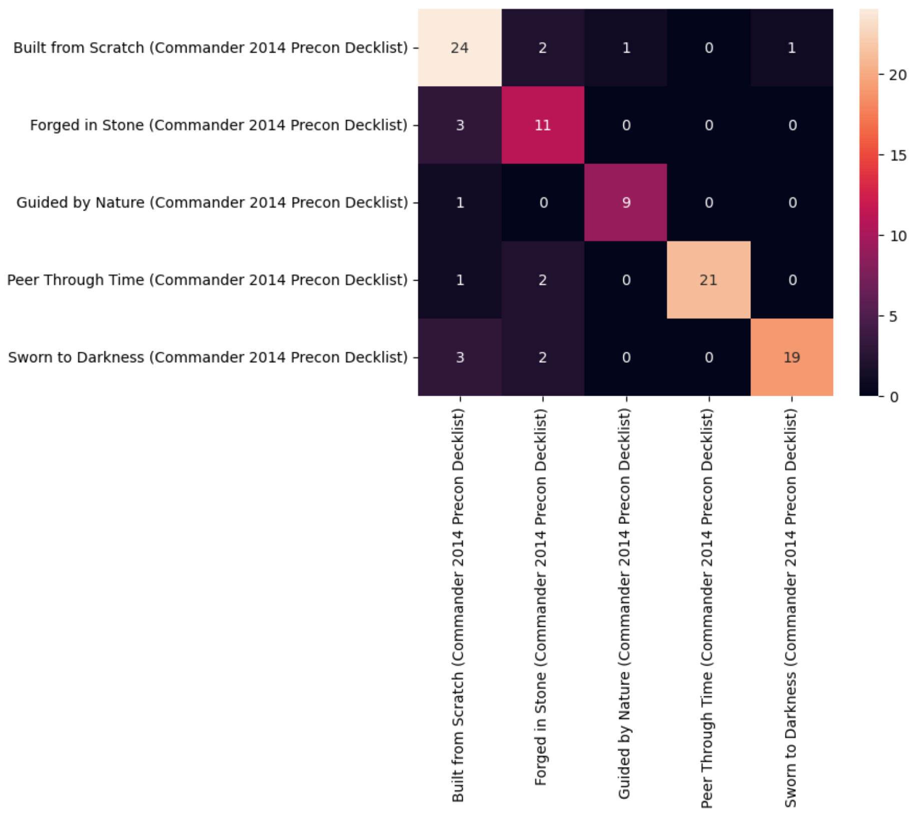
Demonstration of Methods Using a Subset of the Data

Experimental Setup

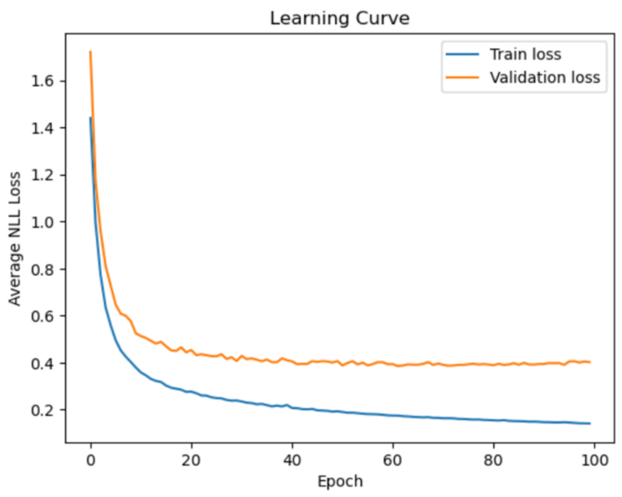
Before we work with the full data set, we want to test with a subset of our data. One set of commander precons was a quintet released in 2014, the aptly-named "Commander 2014" decks. Each of the decks represented one of the game's five colors of cards, and were helmed by a special set of "Planeswalker" cards that could be used as your commander. Because these decks have no overlap in terms of colors nor themes, they are a good benchmark to begin training with.

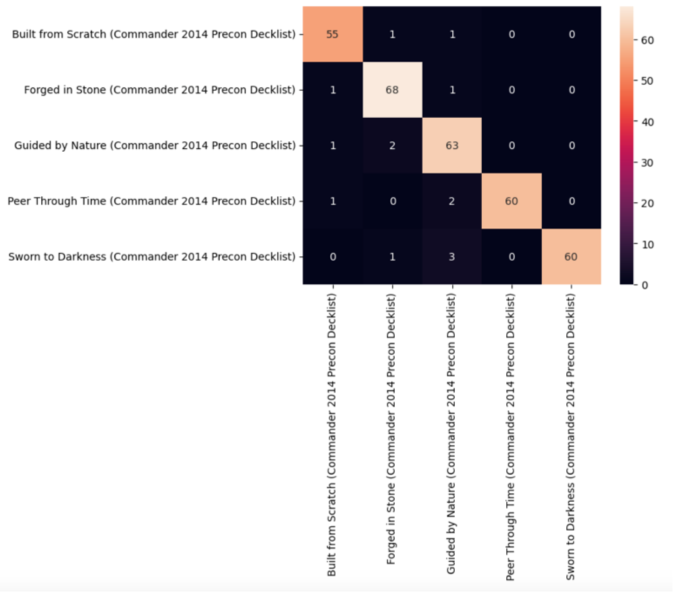
Results

Upon running k-NN, we are returned the best k found as k=10, and an accuracy score of 84%, and the results are modeled in the below confusion matrix:

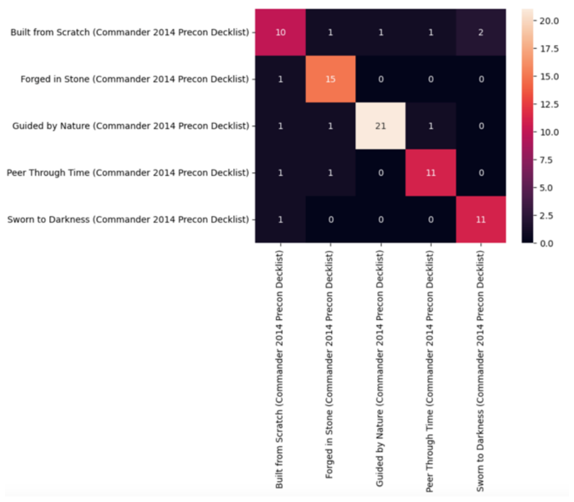


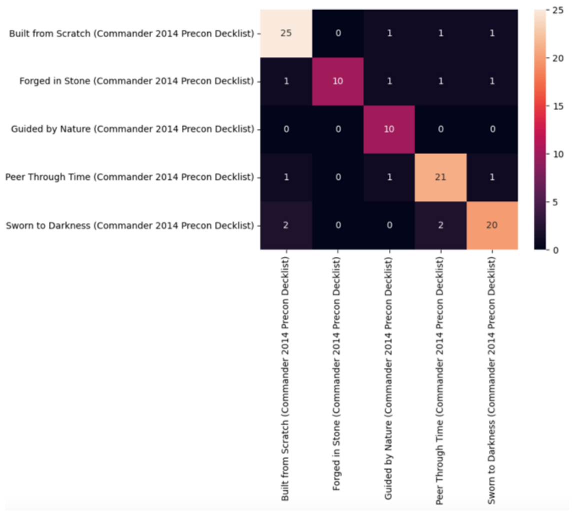
Not to be outdone, Softmax Regression gave us a training accuracy of 95.625%, and the following learning curve and confusion matrix:





And with the following validation accuracy of 85% and testing accuracy of 86%, the respective confusion matrices:





Analysis

k-Nearest Neighbors: 84% Accuracy, while not fantastic, is an acceptable starting place. Noticeably, outside of small misclassification, the Guided by Nature, Peer Through Time, and Sworn to Darkness decks had the vast majority of their cards predicted properly, with Peer Through Time having all of its actual cards classified properly. However, there are considerably more incorrect predictions for those that were actually in the Built from Scratch and Forged in Fire decks, both of which seem to have their error distributed without pattern.

Softmax Regression: At the training level, our softmax model managed to achieve 95.625% accuracy, with a few errant classifications in the first 3 decks, and the latter 2 having accurately had every single one of their cards accurately classified (although there were additional cards misclassified as them). At the validation and testing levels, accuracy was 85% and 86%, respectively; errors at these levels appear to have random distribution with no discernible patterns.

Main Data Set Testing

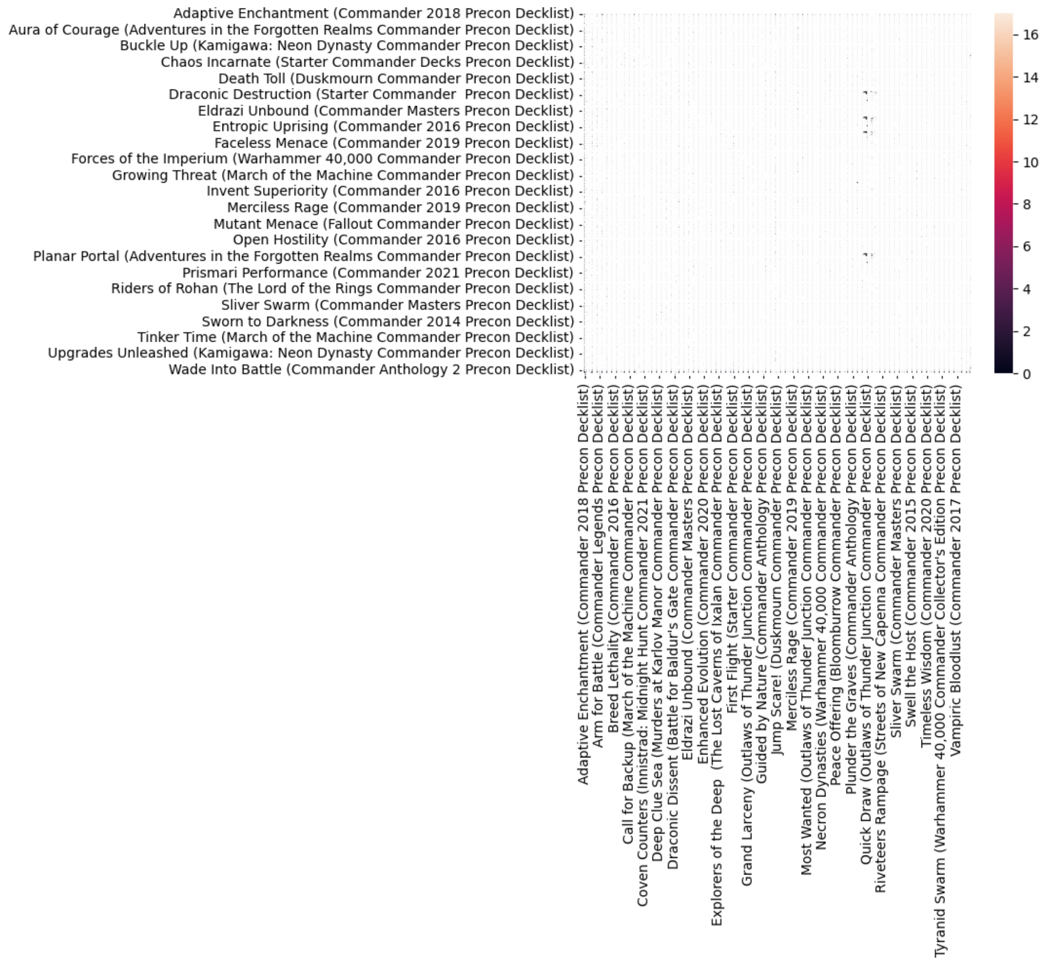
Experimental Setup

The main preprocessed data set has the shape of (9984, 11926). That is, once TF-IDF and OHE have transformed single features into hundreds of columns each, we now have more features than these algorithms are typically equipped for. Thus, thus we need to be careful and conservative with the implementations we do.

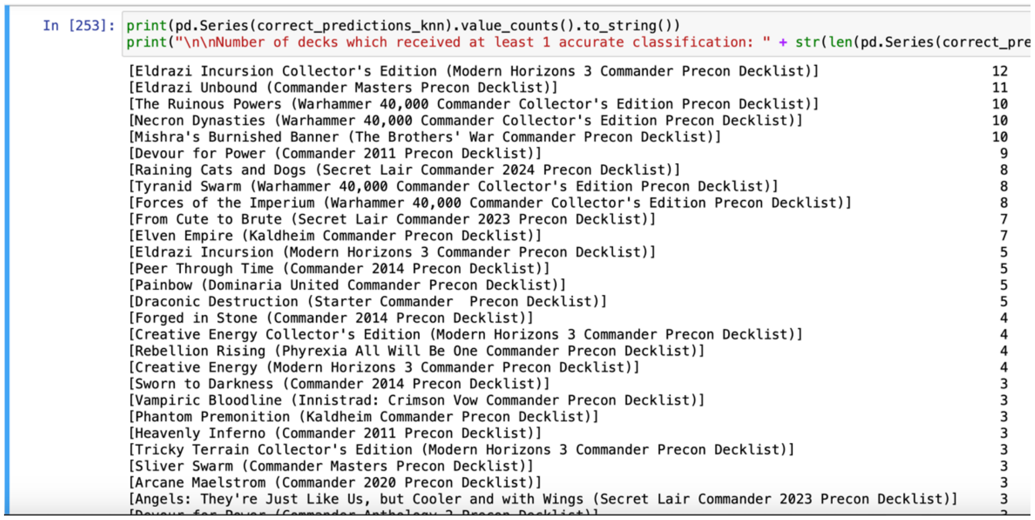
With about ~11 minutes to complete one iteration of kNN our implementation wanting us to iterate from 1 to len(X_tst) (which is 3120), which would take 34,320 minutes, or 572 hours, or almost 24 days. No dice. We'll cut our iterations to len(X_tst) / 100, which is (when converted to an integer), 31 runs, or about 6-7 hours of uninterrupted runtime. We will only investigate further if the highest accuracy found is 31.

Results

The best k found by kNN was 29, with the best accuracy of a whopping 9%. Attempting a confusion matrix yields the following:



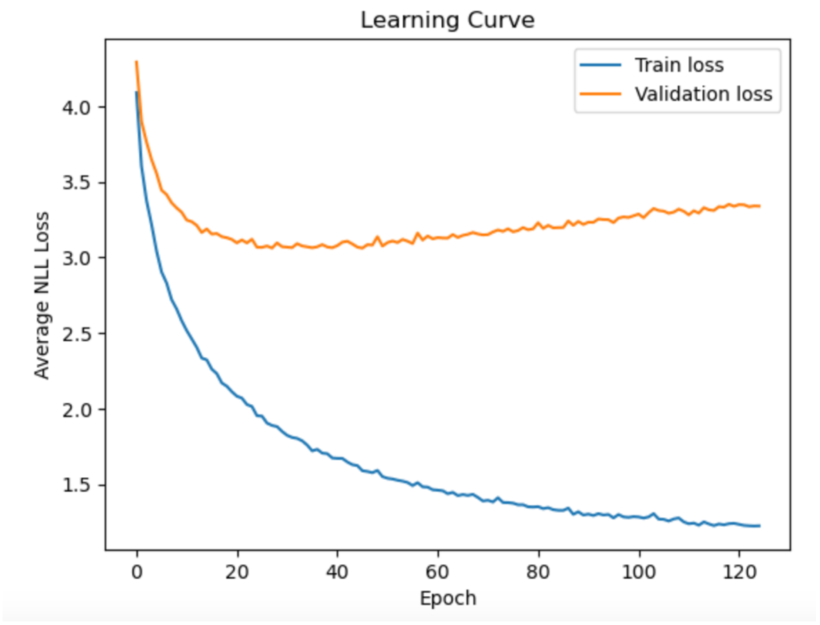
Which is less than opportune. Instead, we do the following, showing how many cards each deck was correctly classified:

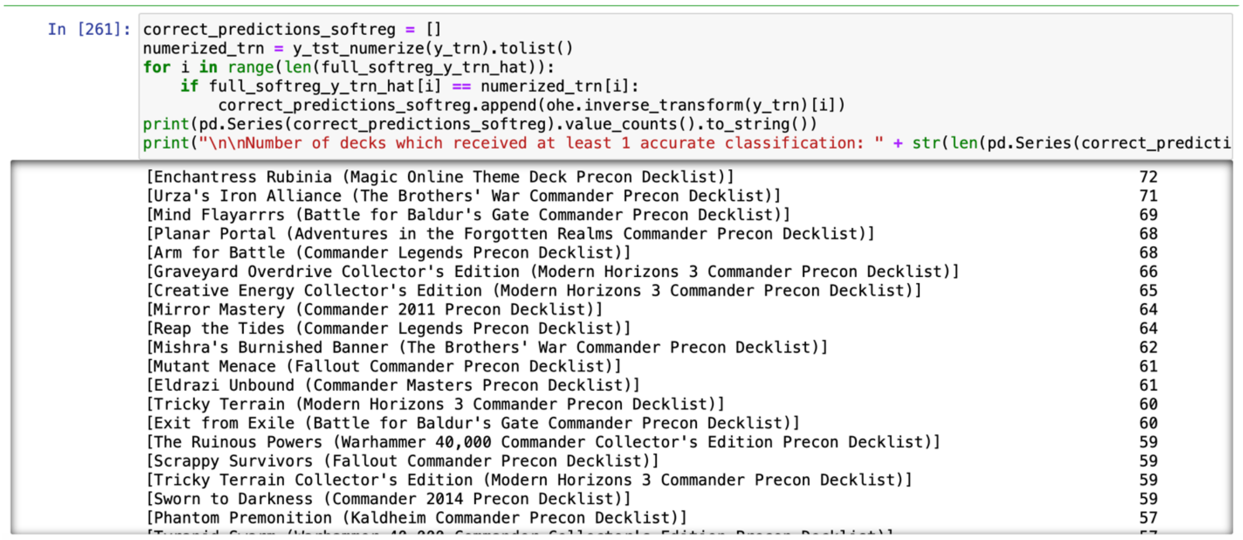


```
In [253]: print(pd.Series(correct_predictions_knn).value_counts().to_string())
          print("\n\nNumber of decks which received at least 1 accurate classification: " + str(len(pd.Series(correct_pre

          [Eldrazi Incursion Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]            12
          [Eldrazi Unbound (Commander Masters Precon Decklist)]                                            11
          [The Ruinous Powers (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]            10
          [Necron Dynasties (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]              10
          [Mishra's Burnished Banner (The Brothers' War Commander Precon Decklist)]                        10
          [Devour for Power (Commander 2011 Precon Decklist)]                                               9
          [Raining Cats and Dogs (Secret Lair Commander 2024 Precon Decklist)]                              8
          [Tyranid Swarm (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]                  8
          [Forces of the Imperium (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]         8
          [From Cute to Brute (Secret Lair Commander 2023 Precon Decklist)]                                 7
          [Elven Empire (Kaldheim Commander Precon Decklist)]                                               7
          [Eldrazi Incursion (Modern Horizons 3 Commander Precon Decklist)]                                 5
          [Peer Through Time (Commander 2014 Precon Decklist)]                                              5
          [Painbow (Dominaria United Commander Precon Decklist)]                                            5
          [Draconic Destruction (Starter Commander  Precon Decklist)]                                       5
          [Forged in Stone (Commander 2014 Precon Decklist)]                                                4
          [Creative Energy Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]               4
          [Rebellion Rising (Phyrexia All Will Be One Commander Precon Decklist)]                           4
          [Creative Energy (Modern Horizons 3 Commander Precon Decklist)]                                   4
          [Sworn to Darkness (Commander 2014 Precon Decklist)]                                              3
          [Vampiric Bloodline (Innistrad: Crimson Vow Commander Precon Decklist)]                           3
          [Phantom Premonition (Kaldheim Commander Precon Decklist)]                                        3
          [Heavenly Inferno (Commander 2011 Precon Decklist)]                                               3
          [Tricky Terrain Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                3
          [Sliver Swarm (Commander Masters Precon Decklist)]                                                3
          [Arcane Maelstrom (Commander 2020 Precon Decklist)]                                               3
          [Angels: They're Just Like Us, but Cooler and with Wings (Secret Lair Commander 2023 Precon Decklist)]   3
```

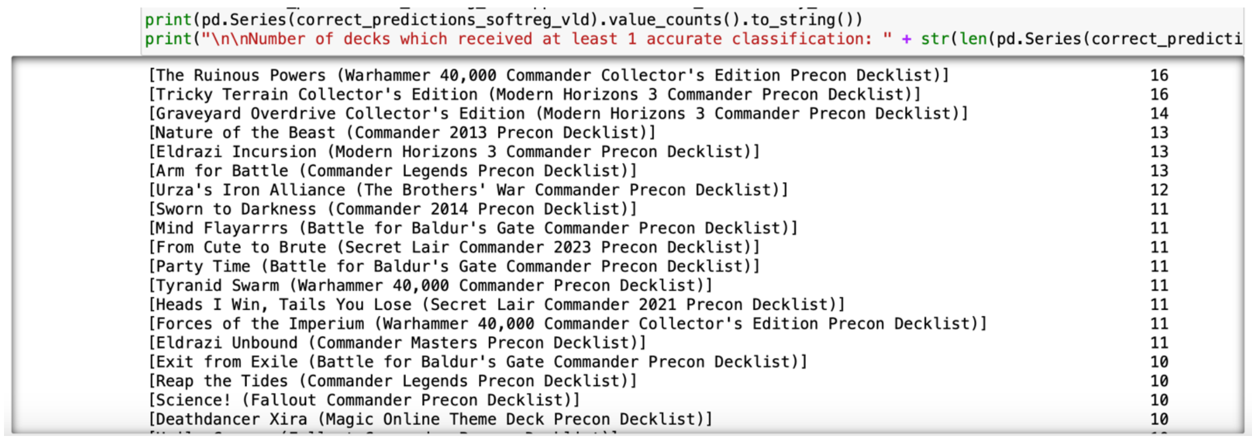
As for softmax, like kNN (although thankfully not as rough), running regressions will balloon in runtime if we are not careful. The following results are at the greatest I was comfortable increasing the runtime to, and the validation and test accuracy appeared to plateau as well. The training data managed to achieve an accuracy of 62.5%, with the following learning curve and deck-accuracy textual visualization:




```
In [261]: correct_predictions_softreg = []
          numerized_trn = y_tst_numerize(y_trn).tolist()
          for i in range(len(full_softreg_y_trn_hat)):
              if full_softreg_y_trn_hat[i] == numerized_trn[i]:
                  correct_predictions_softreg.append(ohe.inverse_transform(y_trn)[i])
          print(pd.Series(correct_predictions_softreg).value_counts().to_string())
          print("\n\nNumber of decks which received at least 1 accurate classification: " + str(len(pd.Series(correct_predicti
```

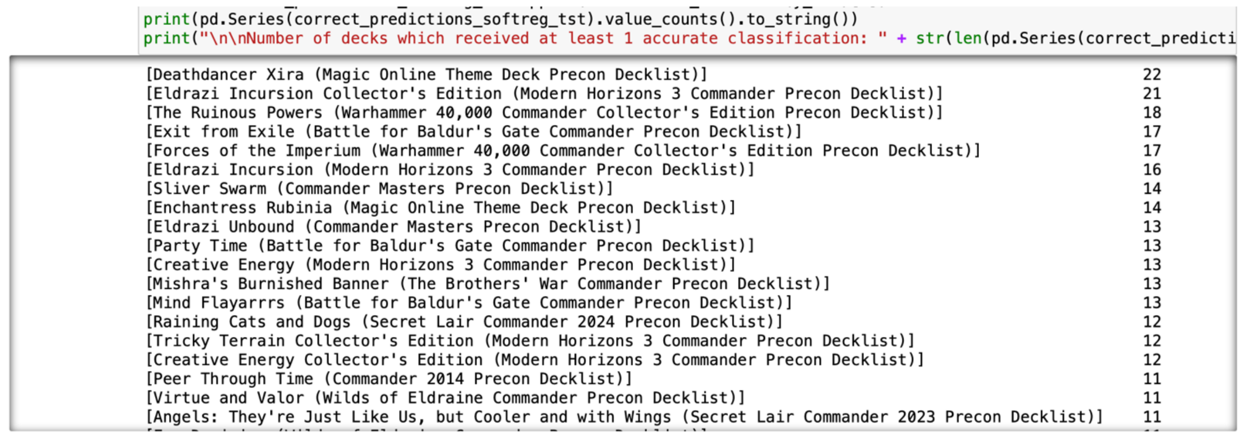
```
[Enchantress Rubinia (Magic Online Theme Deck Precon Decklist)]                                       72
[Urza's Iron Alliance (The Brothers' War Commander Precon Decklist)]                                  71
[Mind Flayarrrs (Battle for Baldur's Gate Commander Precon Decklist)]                                 69
[Planar Portal (Adventures in the Forgotten Realms Commander Precon Decklist)]                        68
[Arm for Battle (Commander Legends Precon Decklist)]                                                  68
[Graveyard Overdrive Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]               66
[Creative Energy Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                   65
[Mirror Mastery (Commander 2011 Precon Decklist)]                                                     64
[Reap the Tides (Commander Legends Precon Decklist)]                                                  64
[Mishra's Burnished Banner (The Brothers' War Commander Precon Decklist)]                             62
[Mutant Menace (Fallout Commander Precon Decklist)]                                                   61
[Eldrazi Unbound (Commander Masters Precon Decklist)]                                                 61
[Tricky Terrain (Modern Horizons 3 Commander Precon Decklist)]                                        60
[Exit from Exile (Battle for Baldur's Gate Commander Precon Decklist)]                                60
[The Ruinous Powers (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]                 59
[Scrappy Survivors (Fallout Commander Precon Decklist)]                                               59
[Tricky Terrain Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                    59
[Sworn to Darkness (Commander 2014 Precon Decklist)]                                                  59
[Phantom Premonition (Kaldheim Commander Precon Decklist)]                                            57
```

The validation and testing sets had accuracy 25.8% and 24.7%, respectively, and the following visualizations:



```
print(pd.Series(correct_predictions_softreg_vld).value_counts().to_string())
print("\n\nNumber of decks which received at least 1 accurate classification: " + str(len(pd.Series(correct_predicti
```

```
[The Ruinous Powers (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]          16
[Tricky Terrain Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]             16
[Graveyard Overdrive Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]        14
[Nature of the Beast (Commander 2013 Precon Decklist)]                                         13
[Eldrazi Incursion (Modern Horizons 3 Commander Precon Decklist)]                              13
[Arm for Battle (Commander Legends Precon Decklist)]                                           13
[Urza's Iron Alliance (The Brothers' War Commander Precon Decklist)]                           12
[Sworn to Darkness (Commander 2014 Precon Decklist)]                                           11
[Mind Flayarrrs (Battle for Baldur's Gate Commander Precon Decklist)]                          11
[From Cute to Brute (Secret Lair Commander 2023 Precon Decklist)]                              11
[Party Time (Battle for Baldur's Gate Commander Precon Decklist)]                              11
[Tyranid Swarm (Warhammer 40,000 Commander Precon Decklist)]                                   11
[Heads I Win, Tails You Lose (Secret Lair Commander 2021 Precon Decklist)]                     11
[Forces of the Imperium (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]      11
[Eldrazi Unbound (Commander Masters Precon Decklist)]                                          11
[Exit from Exile (Battle for Baldur's Gate Commander Precon Decklist)]                         10
[Reap the Tides (Commander Legends Precon Decklist)]                                           10
[Science! (Fallout Commander Precon Decklist)]                                                 10
[Deathdancer Xira (Magic Online Theme Deck Precon Decklist)]                                   10
```



```
print(pd.Series(correct_predictions_softreg_tst).value_counts().to_string())
print("\n\nNumber of decks which received at least 1 accurate classification: " + str(len(pd.Series(correct_predicti
```

```
[Deathdancer Xira (Magic Online Theme Deck Precon Decklist)]                                          22
[Eldrazi Incursion Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                 21
[The Ruinous Powers (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]                 18
[Exit from Exile (Battle for Baldur's Gate Commander Precon Decklist)]                                17
[Forces of the Imperium (Warhammer 40,000 Commander Collector's Edition Precon Decklist)]             17
[Eldrazi Incursion (Modern Horizons 3 Commander Precon Decklist)]                                     16
[Sliver Swarm (Commander Masters Precon Decklist)]                                                    14
[Enchantress Rubinia (Magic Online Theme Deck Precon Decklist)]                                       14
[Eldrazi Unbound (Commander Masters Precon Decklist)]                                                 13
[Party Time (Battle for Baldur's Gate Commander Precon Decklist)]                                     13
[Creative Energy (Modern Horizons 3 Commander Precon Decklist)]                                       13
[Mishra's Burnished Banner (The Brothers' War Commander Precon Decklist)]                             13
[Mind Flayarrrs (Battle for Baldur's Gate Commander Precon Decklist)]                                 13
[Raining Cats and Dogs (Secret Lair Commander 2024 Precon Decklist)]                                  12
[Tricky Terrain Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                    12
[Creative Energy Collector's Edition (Modern Horizons 3 Commander Precon Decklist)]                   12
[Peer Through Time (Commander 2014 Precon Decklist)]                                                  11
[Virtue and Valor (Wilds of Eldraine Commander Precon Decklist)]                                      11
[Angels: They're Just Like Us, but Cooler and with Wings (Secret Lair Commander 2023 Precon Decklist)] 11
```

Analysis

k-Nearest Neighbors: Obviously, 7% accuracy is disappointing to say the least. However, there is valuable innformation to be mined from our output value_counts. The kNN model is incapable of understanding that each deck can only have 100 cards classified to it; thus, if we ever have a deck that is a "similarity black hole," so to speak, it will keep having cards classified to it regardless of if it is a "finished" deck. However, we do see some trends in the (relatively) high-performing decks: the two highest scoring decks, Eldrazi Incursion and Eldrazi Unbound, are both themed around the eponymous Eldrazi creatures. The Eldrazi are notable for being colorless and utilizing their own unique kind of "colorless mana," and for being big, dumb, game-ending creatures that are incredibly lethal if they enter play. Because of this, they're easily seperable from other card types, and it appears the model was able to learn and recognize this. The other major factor we can glean, from the next two highest scoring decks, as well as subsequent results, is that "Universes Beyond" decks (that is, those that are crossovers with other franchises) are much easier for the model to recognize the similarities between. This is likely because, in both name and text body, these cards have more uniqueness to them than those from the main Magic world, and also only exist in sets of 4 (or 8, for those with collectors variants). Secret Lair decks, with their scattershot printings (thanks to the unique reprint system called "The List"), were also generally higher-performing. In spite of the 7% accuracy of the model, there are still visibly learned trends in the data.

Softmax Regression: 62.5%, for the scale of our data set, is thrilling to see for our training data. Additionally, we see that all 156 decks had a card correctly classified to it, a milestone compared to kNN barely reaching halfway to this point. Additionally, there seem to be very few trends as to these classifications; the top performing deck being the Magic Online precon that predates all others seems to have had its greatest success because of this, however beyond that, there appear to be no tendencies outside of potentially collector's editions of decks having greater success. The validation and testing data both had only about 25% accuracy; validation achieved 25.8%, whereas testing achieved 24.7%. However, something that immediately stands out is that testing managed to classify at least one card to 136 of the decks, whereas validation, even with its higher accuracy, classified correctly at least once to only 126 of the decks. Ironically, both validation and testing reaffirm the trends we discussed earlier; the 3 highest scoring decks for the validation set were collector's editions. Additionally, the testing data saw its highest scoring deck be the other, even older Magic Online precon, and the aforementioned one was its 8th highest. Ultimately, while overall accuracy for validation and testing were only around 25%, we can conclude that the model was able to learn trends in the decks.

Conclusion

Ultimately, while I do wish that my model had greater accuracy, I can not say that I am disappointed in it. To begin, the accuracy levels are deceptively low: some cards appear in multiple decks, however they may be classed to the "wrong" deck, yet it was actually proper classification, the model just had the printings switched. Additionally, I have learned a valuable lesson, which is that it is not the number of rows a data set has that affects the most quality of your results, but rather the features. I do wish I had been able to implement a neural network via TensorFlow as well, to see its quality of classifications. However, I am proud of the fact that these models managed to find patterns in such a ridiculously large number of features.

References and Citation

- https://stackoverflow.com/questions/9057379/correct-and-efficient-way-to-flatten-array-in-numpy-in-python
- https://www.geeksforgeeks.org/convert-numpy-array-to-dataframe/
- https://stackoverflow.com/questions/19815335/similarity-between-two-data-sets-or-arrays
- https://stackoverflow.com/questions/17388213/find-the-similarity-metric-between-two-strings
- https://scikit-learn.org/dev/modules/generated/sklearn.preprocessing.OneHotEncoder.html
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shape.html
- https://stackoverflow.com/questions/11620914/how-do-i-remove-nan-values-from-a-numpy-array
- https://stackoverflow.com/questions/3437059/does-python-have-a-string-contains-substring-method
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html
- https://www.turing.com/kb/guide-to-numpy-array-slicing
- https://numpy.org/doc/stable/reference/generated/numpy.concatenate.html
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.replace.html
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.mean.html
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.astype.html
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.at.html
- https://scikit-learn.org/1.5/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html#sklearn.feature_extraction.text.TfidfVectorizer.fit_transform

- https://www.w3schools.com/python/python_ref_string.asp
- https://www.w3schools.com/python/ref_string_rfind.asp
- https://www.w3schools.com/python/ref_string_rindex.asp
- https://www.geeksforgeeks.org/string-slicing-in-python/
- https://docs.python.org/3/library/re.html#re.sub
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.replace.html
- https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.cosine.html
- https://datascience.stackexchange.com/questions/26446/can-i-use-cosine-similarity-as-a-distance-metric-in-a-knn-algorithm
- https://scikit-learn.org/dev/modules/generated/sklearn.preprocessing.OneHotEncoder.html#sklearn.preprocessing.OneHotEncoder.inverse_transform

Acknowledgement

Attempts were made to use ChatGPT to debug this project, however there was never any substantial progress made, and brute force was required instead. I would like to thank Andy Ha for his tireless assistance in finding my plethora of typos throughout my programming, and Professor Jake Lee for enabling this project.

Source Code

https://github.com/mystr0man/ITCS-3156-Final-Project