*Generating Wine Descriptions Using a First Order Markov Model*

*Anthony DeNiro, Hamilton Pitlik*

**Introduction and Motivation**

We created a Markov model to generate wine descriptions that cater to a given set of parameters. These two to three sentence scripts would hopefully reflect to some degree our three inputs of a rating, price, and wine variety. We looked to determine noticeable differences in the generated text descriptions that accounted for or even explained why the wine received a low or high rating or why it was being sold at a low or high price. Furthermore, we wanted to see if descriptions were more closely associated to the price of the wine or the rating. There is a practical use for people like us, who have very little to no knowledge of wine whatsoever; would we be able to discern distinguishing characteristics of the wine through the generated descriptions? We felt the scripts could certainly give us a basis with the criteria by which wines are generally talked about. What accents people are looking for in the taste, what food pairs well with it, etc. It’s surely not every day you’re thrown into the upscale world of wine tasting, but we just wanted to get our foot in the door and a leg up on the competition for when events like this inevitably occur. We figured that at the very least, the Markov model would output key buzzer words in describing different kinds of wine that we would be able to immediately use. Of course we figured the generated script in its entirety would not be ready to be thrown into conversation as it is only a first order Markov model after all.

From our assignment using first order Markov models to generate movie scripts we expected that the scripts in totality would be utter nonsense. Perhaps it works for absurdist humor but admittedly not much else when reading through them line by line. A Markov chain follows a mathematical rule known as the Markov property that involves using probabilities for transitioning states. It explains that the probability distribution of future outcomes is solely dependent on its current state and the amount of time that has elapsed and not the sequence of states preceding it. Meaning that if we are trying to predict a future outcome, which is what we are doing when determining the word that the Markov chain should choose next, it is only conditional on the previous state or word. Given the current state, probabilities are constructed representing the likelihood of the current state transitioning to another specific future state. A random variable can then be used to select the future outcome we are transitioning to from the previous state out of the distribution of other possible future states. A first order Markov model does exactly this. We have a vector containing all the first words used to start a sentence. Attaching probabilities in the same way, we can also use a random variable to determine how to start the sentence, and from there build the transition matrix laying out what was explained earlier in this paragraph. To construct Markov models of higher order, we would need to use multiple previous words to determine the future word instead of just one.

**Data**

The main reason we chose to use the wine dataset was because of its size. Containing 150930 rows and 9 columns, we felt we would have sufficient data to train almost any algorithm or model we wanted to use. Upon first glance, we were actually tempted to do a clustering problem. The dataset gave us multiple attributes for the location the wine came from such as county, province, regions, and even the winery where the wine was made. We thought we could use the given price and rating to determine if there is a specific area where better wine comes from. But we felt the overall scope of this idea was extremely limited in terms of practical use and quickly shifted to the Markov model. The dataset provided us with two to three sentence descriptions for nearly every observation and we figured this could be a more unique and perhaps a more informative way of helping us understand characteristics of the 632 variety of wines in a more implementable manner.

As previously mentioned, the majority of these attributes pertained to the location of the winery. Once we decided on using a Markov model, we were hesitant to and eventually decided against using the location attributes. The primary reason for this was that we feared our descriptions might become too catered to specific parameters and extremely limit the amount of data the Markov model would use to create its description. Another key reason being that with a large dataset, we were unsure how well our CPU’s would handle additional data. We will go into explaining the design of our method later, but essentially our set up was that we were only letting the model select data that matched our parameters of price, rating, and, variety. Adding another parameter and further limiting the data selection process would either yield a result that matched a too few descriptions or none at all. This was something we had underestimated in our initial thinking. What appeared to be a large enough dataset could quickly shrink enough to spark concern over the amount of data made available to the Markov model itself. So ultimately we decided to stay at three parameters instead of adding a fourth one in location.

The three attributes definitely sufficed in breaking down our dataset. By finding quartile ranges over our price and rating columns, we were able to split up these columns into working parameters that would be able to filter the descriptions to match the inputs. The ratings only ranged from 80-100 points and were split up the following way: 80-86 (1st quartile), 87-88 (2nd quartile), 89-90 (3rd quartile), 90-100 (4th quartile). The price had a much larger range and some noticeable outliers. The quartile ranges split the price in the following way: 4-16, 17-24, 25-40, 41-2300. The ranges are clearly not of equal size, but we figured as long as equal amounts of descriptions residing in each of these bins, we wouldn’t have an apparent problem in there being too much or not enough data for the Markov model to use to construct our wine descriptions. We also chose not to remove outliers because of the way our selective process was set up. The wine descriptions for these extremely high-priced wines would only be used for an input that called for the model to use the high quartile price range. There would not be a direct effect on the other three price bins. In addition, we felt the descriptions for our extremely high-priced outliers could actually be more discernible in highlighting distinguishing characteristics than their more moderately priced counterparts and thus help us in determining the differences between wines in our given parameters. Variety, as our third filter, was the most straightforward implementation-wise. We simply split the data by calling for specific name of the wine. We had no previous understanding of which varieties of wine were similar enough to group together (perhaps this is another clustering project) so we felt it was best to leave it as so instead of looking to further group varieties by something like color. At this point we would look to our results and generated descriptions to give us these distinctions.

**Methods**

In order to model the parameters of interest (variety, price, score) using a first-order markov model, the original data set was pruned. The majority of attributes were removed, as well as any entry that was missing data for the parameters of interest. Next, quartile ranges for the prices and scores within the data set were determined, and used in order to define the values that would belong to low, medium low, medium high, and high value ranges.

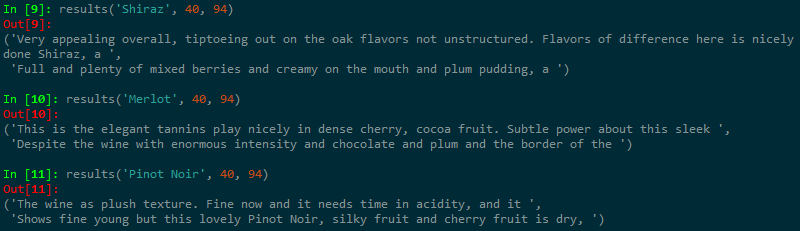
The cleaned data set was read into python using the Pandas library. This smaller data set only consisted of entries that met the criteria specified by the parameters of interest. Descriptions from these entries were concatenated into a single string. Using built in python and pandas functions, the concatenated string was processed in order to create a list of the unique words that were present in the descriptions. Two lists of unique words were created, one corresponding to all unique words, the other consisting only of words that start sentences. These lists were then used in order to create probability matrices.

The probability matrix of first words held the probability of each first word occurring, while the matrix of unique words, transition matrix, held the probability of a certain word occuring after a specific other word. Once these probability matrices were constructed, descriptions consisting of two sentences were written using these probabilities. A sentence consisted of a first word chosen randomly, using the random library in python, from the starting word matrix followed by words from the transition matrix. The order of words selected from the transition matrix depended on the word they followed when being added to the sentence, as such the probabilities from the transition matrix were used in order to determine successive words. Sentences were written to be as long as the average sentence length amongst all descriptions.

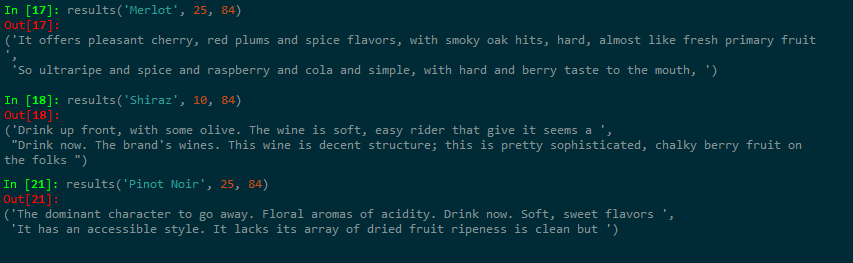
Written descriptions were then compared to descriptions within the original dataset. Key words found amongst the descriptions generated by the model and the original descriptions were noted. Based on the prevalence of shared terms, generated descriptions were deemed as accurate or inaccurate.

**Results**

Examples of the generated descriptions are provided in the following figures. In figure 1.1 and 1.2, descriptions were generated for a variety of wines. The descriptions in 1.1 were created using high price and high rating inputs, while those in 1.2 were made using low price and low rating. Based on these examples, it is clear that more positive adjectives are associated with the higher wine as compared to the lower wine. Across several varieties of wine, it was seen that adjectives such as ‘Flavorful’, ‘Beautiful’, ‘Outstanding’ and other positive adjectives were used in the generated descriptions of high rated and expensive wines. In contrast, generated descriptions for low rated wines used keywords such as ‘Light’, ‘Moderate’, ‘Wanting’. Overall, the adjectives used to describe low rated wines depicted a sense of underwelment. Considering the lowest rated wines were rated eighty out of a possible one-hundred, it was unsurprising that harsher adjectives were not seen.

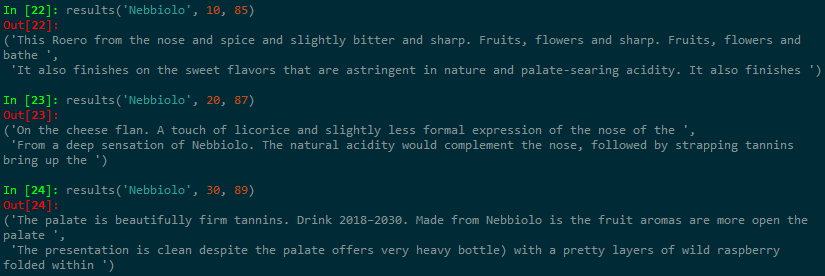


*Figure 1.1: Generated descriptions for three varieties of wine; Shiraz, Merlot, Pinot Noir. Price and rating range set to fourth quartile.*



*Figure 1.2: Generated descriptions for three varieties of wine; Shiraz, Merlot, Pinot Noir. Price and rating range set to first quartile.*

The model was used to generate basic descriptions for a variety of wines, given prices and points within the second and third quartile ranges in order to find words that described the flavor/profile of the wine rather than appeal. Varying wines of course consisted of different flavors; however, when descriptions were generated multiple times for the same variety of wine at different ratings and prices, consistent terms regarding flavor appeared across these descriptions. The results of which are presented in figure 1.3. As seen in the figure, terms such as ‘tannins’, ‘fruit’, and ‘acidity’ occur across the three different quartile ranges sampled. These results were consistent across all varieties tested. When compared to the descriptions provided in the original dataset, the flavors present in the generated descriptions were also present in the original.



*Figure 1.3: Generated descriptions for Nebbiolo. Price and rating range set to first, second, and third quartiles.*

**Conclusion**

Based on the results of this project, it is clear that a first order markov model will write descriptions containing keywords consistent with actual descriptions. While the generated descriptions my not be grammatically correct, the information they provide is more than enough to understand the flavors that portray a variety of wine. Future work should aim to quantify the specific terms that are found across all similarly rated and priced wines; this might be accomplished by implementing a frequent item set model.