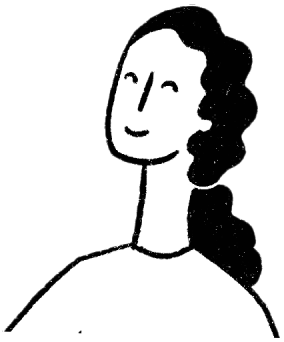
The Digital Sommelier (DRAFT)

A wine recommendation engine based on Machine Learning

Sebastiaan Dijkstra (12251267)
Dilyar Muslem Buzan (12645311)
Lowie de Beer (12674680)
Jacco Kortman (12603503)



Final report for
Tweedejaarsproject BSc KI

Universiteit van Amsterdam
Netherlands
23 - 06 - 2021

Contents

1 Introduction

1.1 The Complex World of Wine

Wine is an incredibly complex world with an infinite number of varieties, regions, and styles. However, navigating this vast sea of choice can be a daunting task to the novice wine drinker. The choice between a bottle of red or white wine is just the tip of the iceberg. It is not just about the wine but also the people who make it and its culture. From vineyards in France to wineries in Napa Valley, many aspects of this drink are taken for granted. The people that study this complex world are called sommeliers.

1.2 Sommeliers

Sommeliers are the people that study wine and give advice on which wines to buy. They can be found in restaurants, fine dining establishments, or at a vineyard. They are not only experts at identifying wines but can tell each one apart by holding them up against the light or swirling it in front of their nose so that they may expose any imperfections before taking a sip themselves. Unfortunately, this complexity leads to two problems: first, people stick to the same bottles of wine they have known their whole life, missing out on a whole variety of different flavor profiles they might like but do not know how to find. Second, this lack of experimentation and discovery hurts the wine merchants' conversions, as people do not know what to buy. Fortunately, Vyno has a solution to this problem.

1.3 Vyno

Vyno is a consumer insight platform for retailers, powered by Naomi, the artificial intelligence (AI) virtual sommelier. Naomi automates the wine profile of a merchant's portfolio and inputs them into a recommendation engine. Positioned as a chatbot on merchants' websites, Vyno enables intelligent wine recommendations for customers, collects consumer data, and produces actionable insight for retailers.

To test the market and gain initial traction, Vyno has currently built a recommendation engine. This engine uses data built from a machine learning model that predicts the core attributes of wine profiles, refined with Master of Wine knowledge.

It works as follows: when a user visits a wine merchant that implemented Naomi, the chatbot shows up in the bottom right corner, telling the user what Naomi can do. Naomi then asks the user a couple of questions regarding specific tastes wrapped in a way that is easy to understand. Naomi will then recommend six wines, ranked from most to least, from the merchant's website.

The downside with the current system is that it is primarily rule-based, which means much time-consuming manual work is involved. For this reason, Vyno is looking to further develop the recommendations into the realm of AI. This project focuses on a small part of the overall vision of Vyno, namely that of live user interactions. Vyno wants a system that can utilize descriptions of wines to find similar products.

The final implementation they had in mind would consist of two phases. The first phase is to let the existing rule-based system narrow down the wine choice to 12 bottles. The second phase is a more modern AI solution that can reduce those 12 bottles to six wines ordered from most to least recommended. Vyno wanted this reduction to occur after the customer picks from eight words related to the taste of the wines. These words should be primary flavors like *lime* or *melon*, easily imagined by the average person reading them. This project will be implementing the second stage of this process.

2 Method

After careful consideration, a clustering-based solution was the most logical for this project because the recommendations were to be made only from the descriptions. However, using only descriptions meant that all the wine characteristics had to be inferred from the text. Thus, the first step in the process will be creating embeddings that describe the different flavor profiles of the wines. Fortunately, Vyno possesses lots of wine data.

2.1 Dataset

Vyno provided two datasets. The first one contained the country, description, name, province, region, subregion, grape, and vineyard for over 150,000 wines. The second dataset contained the same information, plus a column containing the wine title for an extra 130,000 wines. This column is a more detailed description of the wine name. Furthermore, the column order of both datasets was not equal, which resulted in some preprocessing before the datasets could be combined. Removing the duplicates from the merged dataset resulted in a dataset containing 169,000 wines.

2.2 Preprocessing

The first step in the embedding creation pipeline is the preprocessing of the wine descriptions. Taking a random wine sample from the dataset shows the following:

> "This tremendous 100% varietal wine hails from Oakville and was aged over three years in oak. Juicy red-cherry fruit and a compelling hint of caramel greet the palate, framed by elegant, fine tannins and a subtle minty tone in the background. Balanced and rewarding from

> start to finish, it has years ahead of it to develop further nuance. Enjoy 2022–2030."

Studying the descriptions shows that the text contains much information about the specific wine. Keywords such as oak, fine tannins, and minty tones should accurately represent that particular wine. However, a couple of problems also appear. First of all, many stopwords have to be removed. Additionally, because wine descriptions have a distinct and creative language, many of the words in the description that look different describe the same taste. After researching the literature and the web of previous work regarding natural language processing on wine descriptions, the following workflow from Schuring (2019) is adapted for the requirements of this project.

First, the descriptions are stripped of stop words and punctuation. Next, a stemmer is applied to get words back to their root form. Stemming words will keep the fluctuation of different words to a minimum, resulting in better embeddings later. In addition, many of the words in the description are combination words, such as *red cherry*. A set of bi- and tri-grams are produced using the entire corpus of wine descriptions to estimate these combination words. These n-grams are saved for future use when processing new, unseen descriptions. The wine description now looks as follows:

> "tremend, 100_variet, wine, hail, oakvill, age, three_year, oak, juici, redcherri, fruit, compel, hint, caramel, greet, palat, frame, eleg, fine, tannin, subtl, minti, tone, background, balanc, reward, start, finish, year, ahead, develop, nuanc, enjoy, 2022–2030"

The processed description shows that the stop words are correctly removed and that combination words such as red cherry and three years are combined into a single descriptor. Furthermore, words are brought back to their root form and are now ready for further processing.

2.3 Wine Aroma Wheel

As Schuring (2019) describes in his article, wine's language is very distinct and creative. Many of the words that seem different actually describe the same flavor and aroma. For example, *white peach*, *peachy*, and *peach nectarine* all describe the same flavor: *peach*. Bringing these words to their common denominator will significantly improve the embeddings. Fortunately, there has been much research around this area. Chen, Rhodes, Crawford, and Hambuchen (2014) have developed a computational wine wheel that maps words with the same meaning to a set of descriptors. Schuring (2019) developed this further by combining the research from Chen et al. (2014) with Wine Folly and UC Davis contributions. The result is a CSV file containing over 1000 descriptors that is of excellent use for this project. Applying the descriptor map on sample descriptions yields the following result:

> "tremend, 100_variet, wine, hail, oakvill, age, three_year, oak, juicy, cherry, fruit, compel, hint, caramel, greet, palat, frame, elegant, fine, tannin, subtl, mint, tone, background, balanc, reward, start, finish, year, ahead, develop, nuanc, enjoy, 2022–2030"

Careful readers may notice that the *redcherri* is transformed to *cherry*, and rooted words like *eleg* are transformed to *elegant*. Finally, after applying the descriptor mapping to all the descriptions in the dataset, they are ready for embedding.

2.4 Word2Vec

Word2Vec is used to retrieve the semantic meaning between different wine descriptors. Word2Vec works as follows: a two-layer neural network is trained on all the wine descriptions to reconstruct the contexts of the words. Then, the network is thrown away, and the network's hidden layer weights are the resulting embeddings. For example, table 1 shows the output of the model when searching for the descriptor lime. As shown, these words are undoubtedly related to lime.

Descriptor	**Similarity**
citrus	0.759
lime_peel	0.745
lime_kiwi	0.696
nectarine	0.657
lemon	0.655

Table 1: Descriptors and similarity scores of the word *lime*.

2.5 TF-IDF

Because descriptors like *dry* and *fresh* are more common than descriptors such as *syrup* and *lemongrass*, the embeddings need a weighting to influence distinctive descriptors. These weights are created by calculating all descriptors' term frequency-inverse document frequency (TF-IDF) (1) scores. The TF-IDF process works as follows: first, a multiplier is calculated for every descriptor. Then, this multiplier is calculated with the number of times a particular word can be found in a description combined with the amount it appears in all the descriptions.

$$w_{t,d} = \mathrm{tf}_{t,d} \times \mathrm{idf}_t \tag{1}$$

For example, if a word is mentioned ten times in one description and three times in another, the weight of every instance will be lowered so that the overall weight of the descriptor is decreased. Finally, the sum of all the vectors in a description multiplied by their TF-IDF score gives a combined vector for the full description. These are used to compare the wine descriptions to see which ones are similar.

2.6 API

Vyno wanted to interact with this model through an API. The API takes as input the 12 wines and returns the six most recommended wines. The API is developed in Python using the Flask framework. The input to the API is a JSON object with 12 wine descriptions, and the final output is a JSON object with the six recommend wines.

There are two main request calls where Vyno will interact with the API to get six recommended wines at the end. The first is a post request to the server with a JSON file containing 12 wines and their descriptions. Because user interaction is required, the API returns the eight descriptors that the user uses to create a recommendation. After the user selects the descriptors of their choosing, it is sent back to the server. After processing, the API sends back a JSON object with the six best wines that match the chosen descriptors.

2.7 Making Recommendations

Now that all the models are ready, the only thing left is to create the actual recommendations. As mentioned previously, Vyno's rule-based system inputs this system consisting of 12 wines and their description:

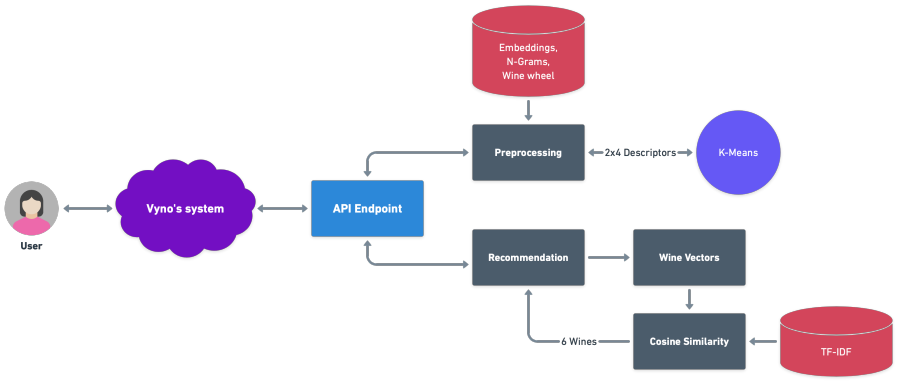
1. Those wine descriptions all get preprocessed the same way as mentioned before.
2. The list of descriptors gets combined into a single set. Because the user needs to choose from two sets of four descriptors, the descriptors are clustered using k-means with eight clusters.
3. The descriptor closest to the centroid is chosen for every cluster, resulting in a list of eight descriptors.

Those descriptors are randomly split into two separate lists and shown one for one to the user. The user picks two of the four descriptors for both lists, which the system uses for creating the wine vector. Creating the wine vectors is done by looking at the embedding vector for every chosen descriptor and multiplying it by the respective TF-IDF weight. Then all the vectors are summed to create a single wine vector indicating the user preference. This wine vector is then compared with the 12 input wines based on cosine similarity. The six most similar wines are returned from most to least similar. Figure 1 shows the complete framework.

3 Results

This section compares three different inputs: 12 white wines, 12 red wines, and a mix of red and white wines.

Figure 1: Flowchart of complete system.



3.1 White Wines

Figure 2 shows the output of the system for the 12 white wines. For explanation's sake, all the descriptors are printed to the terminal to clarify the system. This information will not be returned to the user in production.

Figure 2: System output of the white wines.

```
All descriptors: ['salt', 'asparagus', 'clean', 'herb', 'yellow', 'peach', 'candy', 'green', 'lime', 'cream', 'white', 'com
plex', 'oak', 'weight', 'ripe', 'refreshing', 'nectarine', 'fresh', 'medium_bodied', 'crunchy', 'soft', 'saline', 'light_bo
died', 'stone_fruit', 'smoke', 'supple', 'depth', 'honey', 'bright', 'lemon', 'classy', 'minerality', 'powerful', 'tropical
_fruit', 'fruit', 'tree_fruit', 'guava', 'dry', 'gold', 'round', 'gooseberry', 'full_bodied', 'rich', 'french_oak', 'crisp'
, 'apple', 'grapefruit_pith', 'citrus', 'grapefruit', 'spice', 'elegant', 'shine', 'lemongrass', 'pear', 'straw', 'flower',
 'lime_peel', 'smooth']

All user descriptors: ['asparagus', 'peach', 'lime', 'oak', 'nectarine', 'smoke', 'lemon', 'tropical_fruit', 'apple', 'citr
us', 'grapefruit', 'spice', 'lemongrass', 'pear', 'lime_peel']

The wine descriptors are:  ['pear', 'tropical_fruit', 'nectarine', 'lime']
Please choose first descriptor from list: pear


The wine descriptors are:  ['grapefruit', 'oak', 'lime_peel', 'lemon']
Please choose second descriptor from list: lemon


Based on your preference we recommend these wines:

1: Lembranzas Albariño, Rias Baixas, Spain

2: Domaine de Fussiacus, Mâcon-Fuissé, Burgundy, France

3: Stella Bella, Semillon, Sauvignon Blanc, Margaret River, Australia

4: Rallo Azienda Agricola, Al Qasar Zibibbo, DOP Sicily, Italy

5: Stella Bella, Suckfizzle Chardonnay, Margaret River, Australia

6: Iona, Sophie Te'Blanche Sauvignon Blanc, Elgin, South Africa
```

When comparing the descriptions in the appendix with the extracted keywords, the resulting user descriptors make much sense. For example, descriptors such as *grapefruit*, *tropical fruit*, and *lemon* are easy to imagine. Furthermore, the two sets of four descriptors returned to the user are distinct and easy to understand.

In this example, the user chooses *pear* and *lemon*, resulting in six wine

recommendations. Looking at the description of the most recommend wine:

> "Lemon yellow in colour with hints of gold. Citrus fruit hits the nose followed by apple, pear and dried candied fruits with a mineral note...".

This description seems to correctly fit the chosen descriptors, seeing that *pear* and *lemon* are found in the description. Looking at the description of the wine at second place:

> "Fragrant and fresh yet well rounded with subtle peachy fruit and touches of mineral character."

On the first impression, the second wine does not seem to match the descriptors as well as the first description. However, analyzing the profiles of the wines reveal more information. For example, the albariño that occupies the first place has a beautiful round flavor with a slight minerally aftertaste. The second place is a chardonnay, often a more fatty round flavored wine with a minerally character than other white wines. Considering these facts, it seems likely that the Domaine de Fussiacus is similar to the Lembranzas Albariño. Looking at the description of the wine at the third spot in the list:

> "A charming combination of citrus blossom, lemon, lime and guava with a hint of musk, candied apple and floral aromas."

Comparing this wine with the other two, it seems that it is at the right spot in the list, as this wine is dryer and more floral than the higher-ranked wines. Because Sauvignon Blanc is usually a dry white wine and contains sour and floral elements, the system used the lemon descriptor to place this third.

3.2 Red Wines

Figure 3 shows the output of the system for the 12 red wines. Descriptors such as *spice*, *velvety*, *oak*, and *blackberry* are common characteristics of red wines. Furthermore, the descriptors shown to the user are all easy to imagine for the average person.

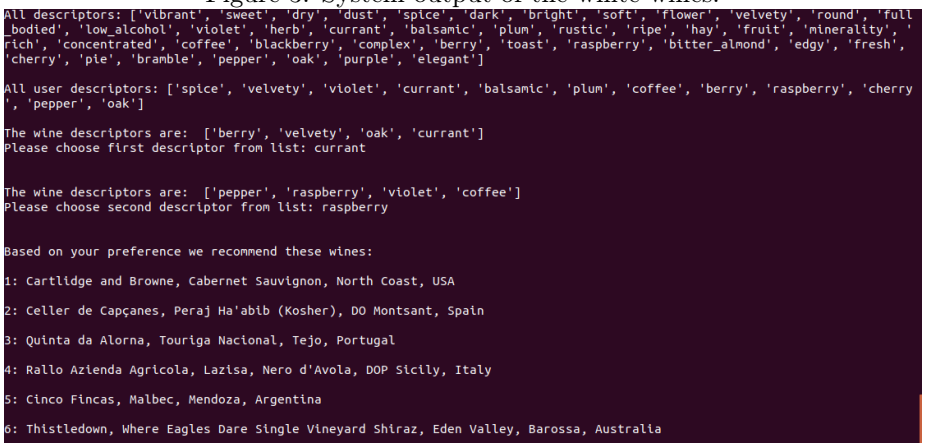
In this example, the user chooses the following descriptors: *currant* and *raspberry*, both forest fruits. Looking at the description of the most recommend wine:

> "This Cabernet starts out with rich, delicious red fruits on the nose that flows into cherry, watermelon, candied plum and crème brûlée."

The first placed Cabernet Sauvignon description involves much red fruit, so it should be compared to the second and third place to see if it deserves the number one spot. The second-place descriptions read as follows:

> "Dark, full red yet vibrant colour. Bright aromas of cherry, plums, currant and toasty, coffee, mocha nuances."

Figure 3: System output of the white wines.



The description describes the wine as having a bright cherry aroma, so forest fruits are present but less than in the Cabernet Sauvignon. Seeing that the aroma of a wine is a big part of the taste, it is not surprising that this wine got the number two spot. Looking at the third spot:

> "This purple coloured wine has a floral aroma with strong notes of violets and red and black fruits, like raspberry and blackberry."

This wine has forest fruits as strong notes, which is less present than the aroma. So the ranking of red wines can be considered accurate as well.

3.3 Mixed Wines

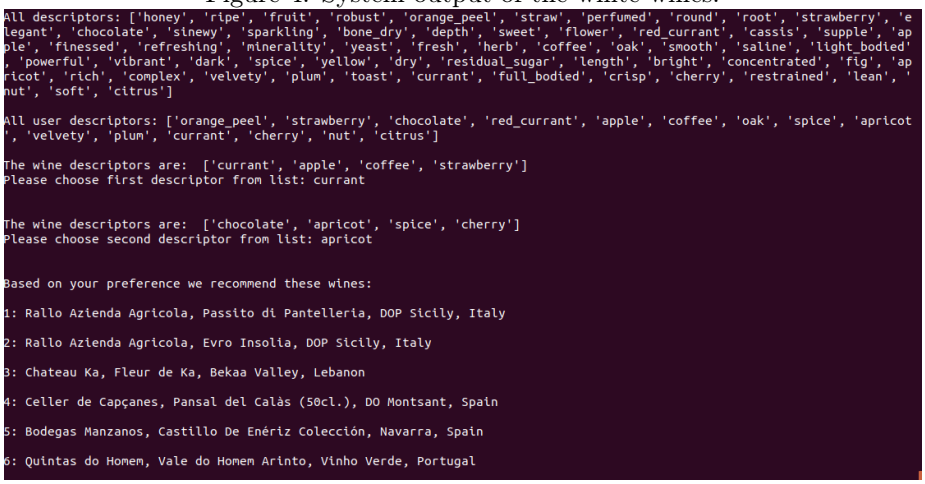
Since customers of Vyno can indicate they prefer both red and white wine, it is essential to have a look at the combined results as well. Figure 4 shows these results.

The chosen descriptor *currant* is mainly used for red wines and the chosen descriptor *apricot* primarily for white wines. This combination will make it particularly interesting to see how the ranking will be calculated and which descriptor ranks higher. Looking at the description of the most recommend wine:

> "A light amber colour with intense yet sweet, elegant and well-balanced with notes of orange peel, apricots, dried figs and honey with impressive acidity."

This description describes the wine as having notes of apricot, which is not significantly dominant, but the wine does include apricot. Now it is essential to see if the number two wine has either currant or apricot as a descriptor but less presently:

Figure 4: System output of the white wines.



> "A straw yellow colour wine, with fresh citrussy notes and crisp apple. Persistent harmanious and mineral on the finish."

Unfortunately, this wine does neither include apricot or currant, making it hard to see why the recommendation engine has placed this wine second. It is plausible that it has calculated currant as a dry and sour fruit that, collectively with the apricot's vector, makes this sour and dry wine suitable. The third-place must make less sense than the second place to see if the engine is entirely accurate:

> "Bright red fruits jump out of the glass along with a hint of oak. The palate is rich, smooth and supple with bright cherry and redcurrant fruit."

This wine contains the word redcurrant and seems to fit the forest fruits taste very well. However, the apricot is missing. The missing apricot is probably why this wine ranks third: it fits half of the chosen descriptors, so it is a pretty good recommendation but not fitting for both descriptors.

4 Discussion

In conclusion, it is safe to say the algorithm operates rather accurately. When analyzing the results, it is hard to determine whether it works perfectly. The difficulty in this is because the taste of wine can be very subjective, and it is impossible to check if wines are similar enough based solely on the description. Luckily this project was mainly about making a product for Vyno that was easy to use and would contribute to their virtual sommelier. Since day one, there

has been intensive contact between the company and the engineers, which has contributed to a streamlined end product that should tick off many boxes from Vyno's expectations.

There is always room for improvement, so here a few points of improvement will be discussed. The first point is that some descriptors in the wine wheel can be viewed as too vague or simply unsuitable to present to a customer. Furthermore, Vyno might decide that a descriptor is missing, but this is easy to deal with since the wine wheel is a CSV that can be altered. Anyone using the recommendation engine can open the CSV file and add or delete words without any problem since it does not alter the code.

Another point of improvement can be found when a customer selects both white and red wines as the desired output. The descriptors shown are mixed with white and red wines and do not necessarily feel natural in combination. When reading *cherry* and *apricot* next to each other, it is hard to imagine a good wine using both these descriptors. The optimal solution would be to use only red wine descriptors for the first four words and solely white wine descriptors for the second four words.

References

Chen, B., Rhodes, C., Crawford, A., & Hambuchen, L. (2014). Wineinformatics: applying data mining on wine sensory reviews processed by the computational wine wheel. In *2014 ieee international conference on data mining workshop* (pp. 142–149).

Schuring, R. (2019, Dec). *Robosomm chapter 3: Wine embeddings and a wine recommender.* Towards Data Science. Retrieved from `https://towardsdatascience.com/robosomm-chapter-3-wine-embeddings-and-a-wine-recommende`