Machine Learning assisted Gross Weight Prediction for Alcoholic Beverages in Retail

Christian Schorr

Trier University of Applied Sciences, Environmental Campus Birkenfeld
c.schorr@umwelt-campus.de

Abstract.

In this paper we present a machine learning pipeline developed specifically for the product group of alcoholic beverages with focus on the two segments wine and beer which constitute the major part of a retailer's alcoholic beverages inventory. We focus on exploiting expert knowledge about the data domain to engineer features tailored to prediction of the important attribute gross weight. Experiments with data from a major retail company show that our proposed machine learning approach with feature enriched data achieves superior results which are more robust than those obtained by traditional heuristic approaches on the original data. In practical terms this is a step towards fully automated product data generation and maintenance reducing manual effort and thus costs for a retail company.

Keywords: Product Classification, Feature Engineering, Machine Learning

1 Introduction

Retail companies dealing in alcoholic beverages are faced with a constant flux of products. Apart from general product changes like modified bottle designs and sizes or new packaging units two factors are responsible for this development. The first is the natural wine cycle with new vintages arriving at the market and old ones cycling out each year. The second is the impact of the rapidly growing craft beer trend which has also motivated established breweries to add to their range. The management of the corresponding product data is a challenge for most retail companies. The reason lies in the large amount of data and its complexity. Data entry and maintenance processes are linked with considerable manual effort resulting in high data management costs. Product data attributes like dimensions, weights and supplier information are often entered manually into the data base and are often afflicted with errors. Another widely used source of product data is the import from commercial data pools. A means of checking the data thus acquired for plausibility is necessary. Sometimes product data is incomplete due to different reasons and a method to fill the missing values is required. All these possible product data errors lead to complications in the downstream automated purchase and logistics processes.

We propose a machine learning model which involves domain specific knowledge and compare it a heuristic approach by applying both to real world data of a retail company. In this paper we address the problem of predicting the gross weight of product items in the merchandise category alcoholic beverages. To this end we introduce two levels of additional features. The first level consists of engineered features which can be determined by the basic features alone or by domain specific expert knowledge like which type of bottle is usually used for which grape variety. In the next step an advanced second level feature is computed from these first level features. Adding these two levels of engineered features increases the prediction quality of the suggestion values we are looking for. The results emphasize the importance of careful feature engineering using expert knowledge about the data domain.

2 Related Work

Feature Engineering is the process of extracting features from the data in order to train a prediction model. It is a crucial step in the machine learning pipeline, because the quality of the prediction is based on the choice of features used to training. The majority of time and effort in building a machine learning pipeline is spent on data cleaning and feature engineering [Domingos 2012]. A first overview of basic feature engineering principles can be found in [Zheng 2018]. The main problem is the dependency of the feature choice on the data set and the prediction algorithm. What works best for one combination does not necessarily work for another. A systematic approach to feature engineering without expert knowledge about the data is given in [Heaton 2016]. The authors present a study whether different machine learning algorithms are able to synthesize engineered features on their own. As engineered features logarithms, ratios, powers and other simple mathematical functions of the original features are used. In [Anderson 2017] a framework for automated feature engineering is described.

3 Data Set

The data set is provided by a major German retail company and consists of 3659 beers and 10212 wines. Each product is characterized by the seven features shown in table 1. The product name obeys only a generalized format. Depending on the user generating the product entry in the company data base, abbreviating style and other editing may vary. The product group is a company specific number which encodes the product category - dairy products, vegetables or soft drinks for example. In our case it allows a differentiation of the product into beer and wine. Additionally wines are grouped by country of origin and for Germany also into wine-growing regions. Note that the product group is no inherent feature like length, width, height and volume, but depends on the product classification system a company uses. The dimensions length, width, height and the volume derived by multiplicating them are given as float values. The feature (gross) weight, also given as a float value, is what we want to predict.

3.1 Feature description

Feature	Type	Unit	Example
Product name	string	-	MW PFUNGSTAED EDEL PILS EXCLUSIV
Product group	int	-	47114
Length	float	mm	69.0
Width	float	mm	69.0
Height	float	mm	270.0
Volume	float	ml	1285470.0
Gross Weight	float	g	915.0

Table. 1. Data set features, corresponding types, units and example

3.2 General Pre-Processing

As is often the case with real world data, a pre-processing step has to be performed prior to the actual machine learning in order to reduce data errors and inconsistencies. For our data we first removed all articles missing one or more of the required attributes of table 1. Then all articles with dummy values were identified and discarded. Dummy values are often introduced due to internal process requirements but do not add any relevant information to the data. If for example the attribute weight has to be filled for an article during article generation in order to proceed to the next step but the actual value is not know, often a dummy value of 1 or 999 is entered. These values distort the prediction model when used as training data in the machine learning step. The product name is subjected to lower casing and substitution of special German characters like umlauts. Special symbolic characters like #,! or separators are also deleted. With this pre-processing done the data is ready to be used for feature engineering.

Following this formal data cleaning we perform an additional content-focused pre-processing. The feature weight is discretized by binning it with bin width 10g. Volume is likewise treated with bin size 10ml. This simplifies the value distribution without rendering it too coarse. All articles where length is not equal to width are removed, because in these cases there are no single items but packages of items.

4 Feature Engineering

Often the data at hand is not sufficient to train a meaningful prediction model. In these cases feature engineering is a promising option. Identifying and engineering new features depends heavily on expert knowledge of the application domain. The first level consists of engineered features which can be determined by the original features alone. In the next step advanced second level features are computed from these first level and the original features.

For our data set the original features are product name and group as well as the dimensions length, width, height and volume. We see that the volume is computed in the most general way by multiplication of the dimensions. Geometrically this corresponds to all products being modelled as cuboids. Since angular beer or wine bottles are very much the exception in the real world, a sensible new feature would be a more appropriate modelling of the bottle shape. Since weight is closely correlated to volume, the better the volume estimate the better the weight estimate. To this end we propose four first level engineered features: capacity, wine bottle type, beer packaging type and beer bottle type which are in turn used to compute a second level engineered feature namely the packaging specific volume. Figure 1 shows all discussed features and their interdependencies.

4.1 Capacity

Let us have a closer look at the first level engineered features. The capacity of a beverage states the amount of liquid contained and is usually limited to a few discrete values. 0.33l and 0.5l are typical values for beer cans and bottles while wines are almost exclusively sold in 0.75l bottles and sometimes in 0.375l bottles. The capacity can be estimated from the given volume with sufficient certainty using appropriate threshold values. Outliers were removed from the data set.

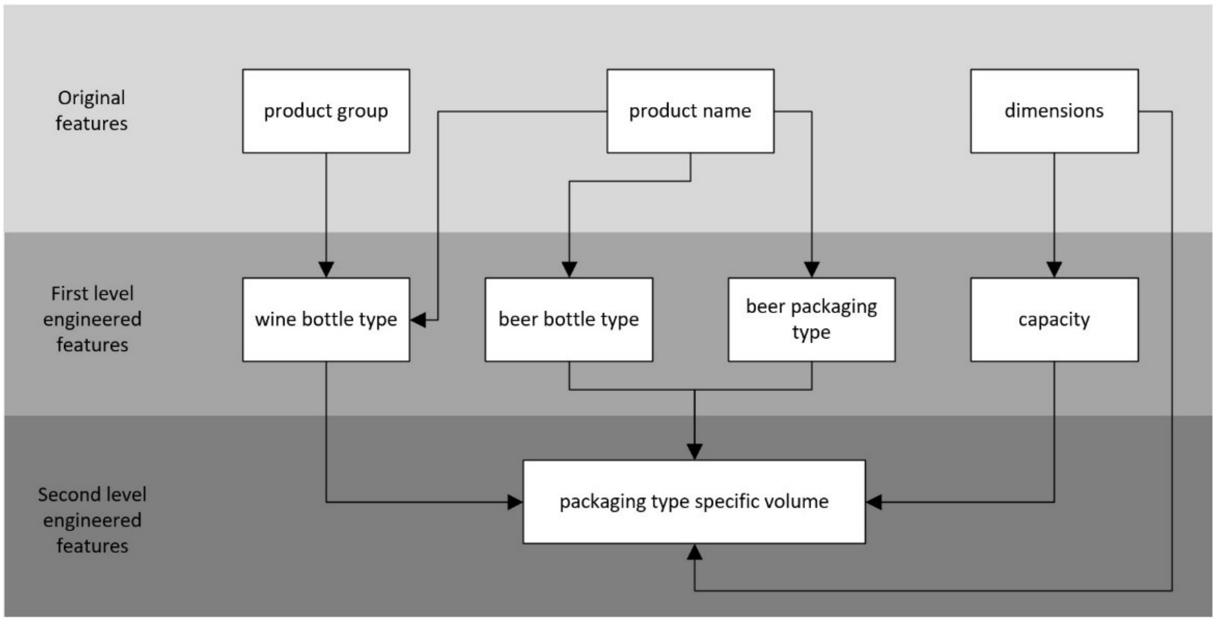

Figure. 1. First and second level engineered features

4.2 Beer packaging and bottle types

There are three main beer packaging types in retail: cans, bottles and kegs. While kegs are mainly of interest to pubs and restaurants and are not considered in this paper, cans and bottles target the typical super market shopper and come in a greater variety. In our data set, the product name in case of beers is preceded by a prefix denoting whether the product is packaged in a can or a bottle. Extracting the relevant information is done using regular expressions. Not, though, that the prefix is not always correct and needs to be checked against the dimensions.

The shapes of cans are the same for all practical purposes, no matter the capacity. The only difference is in their wall thickness, which depends on the material, aluminium and tin foil being the two common ones. The difference is weight is small and the actual material used is impossible to extract from the data. A further distinction for cans in different types like for beer and wine is therefore unnecessary.

Regarding the German beer market, the five bottle types shown in figure 2 are pre-dominant: longneck, NRW, Euro, Steini and NRW-Vichy. While not conforming to a DIN (the DIN 6199 Packmittel – Flaschen, Steinieform was withdrawn in 1997) standard anymore, the variations in length, width and height and weight are small. Therefore, bottles of the same type and brand may have different weights, even if they are sold in the same crate. Measurements for a crate of 18 filled Steini bottles resulted in 20 different weight values. A tolerance of +/- 2% was observed. As a pre-processing step, the weights were therefore rounded to the nearest 10g.

This holds true for all of the five bottle types in question. Since for beers there are no comparably meaningful reference points like color, grape or region of origin, the bottle type is determined by height and diameter using the minimum of the Euclidean distance to a set of idealized bottles with dimensions given in table 2. Bottles which were differing by more than 3% from these dimensions were removed from the data set.

Bottle type	Capacity	Height	Diameter
NRW	500 ml	260.0	67.0
Euro	500 ml	230.0	70.6
Longneck	500 ml	270.0	68.3
Longneck	330 ml	238.0	60.0
NRW-Vichy	330 ml	233.0	61.0
Steini	330 ml	174.0	70.0

Table. 2. Beer bottle types and corresponding typical dimensions

The engineered feature *beer packaging type* assigns each article identified as beer by its product group to one of the classes bottle or can. The feature beer bottle type contains the most probably member of the five main beer bottle types. Packages containing more than one bottle or can like crates or six packs are not considered in this paper and were removed from the data set.

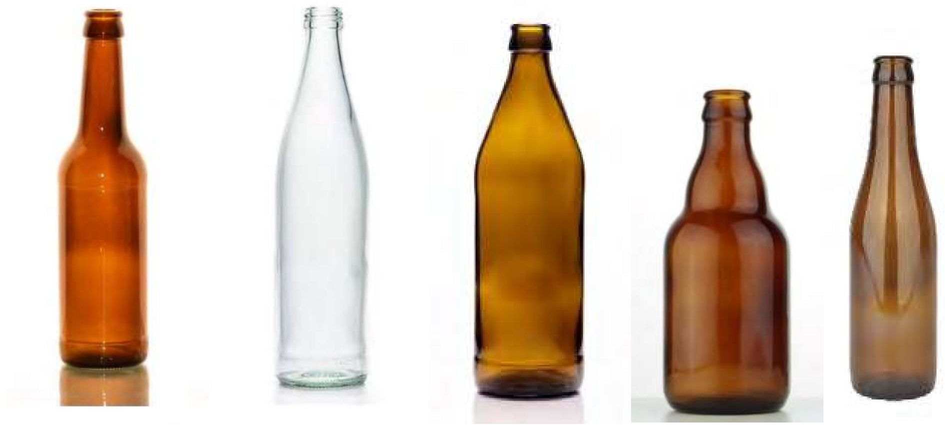

Figure 2. Beer bottle types, from left to right: Longneck, NRW, Euro, Steini, NRW-Vichy

4.3 Wine packaging types

Compared to beer the variety of commercially sold wine packagings is limited to bottles only. A corresponding packaging type attribute to distinguish between cans and bottles is not necessary. Again there are a few bottle types which are used for the majority of wines, namely Schlegel, Bordeaux and Burgunder (Figure 3). Deciding what product is filled in which bottle type is a question of domain knowledge. The original data set does not contain a corresponding feature. From the product group the country of origin and in the case of German wines the region can be determined via a mapping table. This depends on the type of product classification system the respective company uses and has not to be valid for all companies. Our data set uses a customer specific classification with focus on Germany. A more general one would be the Global Product Classification (GPC) standard for example. To determine wine growing regions in non-German countries like France the product name has to be analyzed using regular expressions. The type of grape is likewise to be deduced from the product name if possible. Using the country and specifically the region of origin and type of grape of the wine in question is the only way to assign a bottle type with acceptable certainty. There are countries and region in which a certain bottle type is used predominantly, sometimes also depending on the color of the wine. The Schlegel bottle, for example, is almost exclusively used for German and Alsatian white wines and almost nowhere else. Bordeaux and Burgunder bottles on the other hand are used throughout the world. Some countries like California or Chile use a mix of bottle types for their wines, which poses an additional challenge. With expert knowledge one can assign regions and grape types to the different bottle types. As with beer bottles this categorization is by no means comprehensive or free of exceptions but serves as a first step.

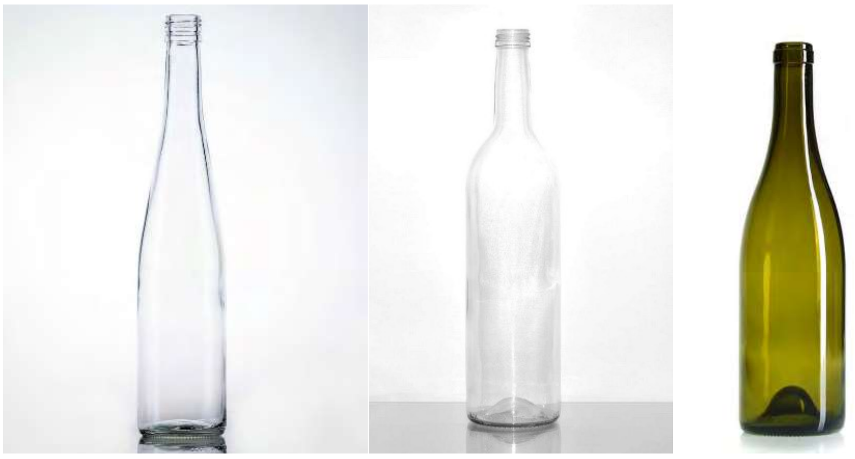

Figure 2. Wine bottle types, from left to right: Schlegel, Bordeaux, Burgunder

4.4 Packaging type specific volume computation

The standard volume computation by multiplying the product dimensions length, width and height is a rather coarse cuboid approximation to the real shape of alcoholic beverage packagings. Since the volume is intrinsically linked to the weight which we want to predict a packaging type specific volume computation is required for cans and especially bottles.

The modelling of a can is straightforward using a cylinder with the given height h and a diameter of the given width w and length l. Thus the packaging type specific volume is:

$$V_{Can}^{C} = 2\pi \cdot l \cdot w \cdot h \tag{1}$$

A bottle on the other hand needs to be modelled piecewise. Its height can be divided into three parts: base, shoulders and neck as shown in figure 4. Base and neck can be modeled by a cylinder. The shoulders are approximated by a truncated cone. With the help of the corresponding partial heights h_{Base}, $h_{Shoulders}$ and h_{Neck} we can compute coefficients k_{Base}, $k_{Shoulders}$ and k_{Neck} as fractions of the overall height h of the bottle. The diameters of the bottle base and the neck opening are given by d_{Base} and d_{Openi} and are likewise used to compute the ratio $k_{opening}$. Since bottles have circular bases, the values for width w and length l in the original data have to be the same and either one may be used for d_{Base}. These four coefficients are characteristic for each bottle type, be it beer or wine (table 3). With their help, a bottle type specific volume from the original data length, width and height can be computed which is a much better approximation to the true volume than the former cuboid model.

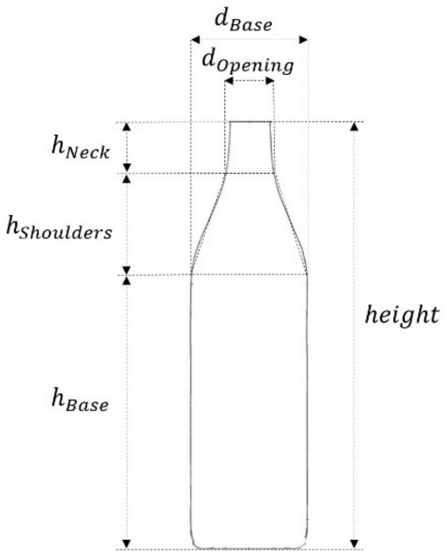

Figure 4. General parametric bottle description

Coefficient	Burgunder	Bordeaux	Schlegel	NRW	Longneck	Steini	Euro
k_{Base}^{Btype}	0.48	0.60	0.34	0.59	0.52	0.52	0.62
$k_{Shoulders}^{Btype}$	0.32	0.13	0.44	0.28	0.13	0.29	0.34
k_{Neck}^{Btype}	0.20	0.27	0.22	0.13	0.35	0.19	0.24
$k_{Opening}^{Btype}$	0.40	0.40	0.40	0.37	0.37	0.43	0.35

Table 3. Empirically measured coefficients for different bottle types

The bottle base can be modelled as a cylinder as follows:

$$V_{Base}^{C} = 2\pi \cdot l \cdot w \cdot h \cdot k_{Base}^{Btype} \tag{2}$$

The bottle shoulders have the form of a truncated cone and are described by formula 3:

$$V_{Shoulders}^{C} = \frac{1}{12}\pi \cdot h \cdot k_{Shoulders}^{Btype} \cdot l \cdot \left(l + l \cdot k_{Opening}^{Btype} + l \cdot k_{Opening}^{Btype}{}^{2}\right) \tag{3}$$

The bottle neck again is a simple cylinder:

$$V_{Neck}^{C} = 2\pi \cdot l \cdot w \cdot k_{Opening}^{Btype} \cdot h \cdot k_{Neck}^{Btype} \tag{4}$$

Summing up all three sections yields the *packaging type specific volume* for bottles:

$$V_{Bottle}^{C} = V_{Base}^{C} + V_{Shoulders}^{C} + V_{Neck}^{C} \tag{5}$$

5 Experiments

The experiments follow the multi-level feature engineering scheme as shown in figure 1. First, we use only the original features product group and dimensions. Then we add the first level engineered features *capacity* and *bottle type* to the basic features. Next the second level engineered feature *packaging type specific volume* is used along with the basic features. Finally all features from every level are used for the prediction. After pre-processing and feature engineering the data set size is reduced from 3659 to 3380 beers and from 10212 to 8946 wines.

5.1 Algorithms and metrics

For prediction of the continuous valued attribute gross weight, we use and compare several regression algorithms. Both the decision-tree based Random Forests algorithm (Breimann, 2001) and support vector machines (SVM) (Cortes, 1995) are available in regression mode (Smola, 1997). Linear regression (Lai, 1979) and stochastic gradient descent (SGD) (Taddy, 2019) are also employed as examples of more traditional statics-based methods. Our baseline is a heuristic approach taking the median of the attribute gross weight for each product group and use this value as a prediction for all products of the same product group. Practical experience has shown this to be a surprisingly good strategy.

The implementation was done in Python 3.6 using the standard libraries sk-learn and pandas. All numeric features were logarithmized prior to training the models. The non-numeric feature *bottle type* was converted to numbers. The final results were obtained using tenfold cross validation (Kohavi, 1995). For model training 80% of the data was used while the remaining 20% constituted the test data.

We used the root mean square error (RSME) (6) as well as the mean and variance of the absolute percentage error e_{ap} (7) as metrics for the evaluation of the performance of the algorithms.

$$RSME = \sqrt{\frac{\sum_{n=1}^{N}\left(v_n^{predicted} - v_n^{true}\right)^2}{N}} \qquad (6)$$

$$e_{ap} = \frac{1}{N}\sum_{n=1}^{N}\frac{\left|v_n^{predicted} - v_n^{true}\right|}{V_{true}} \qquad (7)$$

5.2 Results

Features	Algorithm	RSME	Mean e_{ap}	Var e_{ap}
Product group	Baseline	216.40	14.28	319.00
Dimensions, product group	Linear Regression	135.12	8.89	114.78
Dimensions + capacity	Linear Regression	142.97	9.19	137.32
Dimensions + bottle_type	Linear Regression	152.39	9.02	127.72
Dimensions + bottle_type + capacity	Linear Regression	136.45	8.19	115.92
Dimensions + packaging_type_specific_volume	Linear Regression	181.30	8.33	114.12
Dimensions + all engineered features	Linear Regression	133.91	8.08	122.09
Dimensions, product group	SGD	142.81	10.10	124.41
Dimensions + capacity	SGD	124.68	8.09	129.19
Dimensions + bottle_type	SGD	143.38	10.39	105.47
Dimensions + bottle_type + capacity	SGD	176.30	8.96	142.60
Dimensions + packaging_type_specific_volume	SGD	135.59	10.34	142.80
Dimensions + all engineered features	SGD	154.62	9.37	138.82
Dimensions, product group	Random Forest	**91.73**	6.72	121.45
Dimensions + capacity	Random Forest	**98.75**	**4.98**	**106.12**
Dimensions + bottle_type	Random Forest	**91.75**	**4.59**	**80.94**
Dimensions + bottle_type + capacity	Random Forest	**83.25**	**3.76**	**71.17**
Dimensions + packaging_type_specific_volume	Random Forest	**111.72**	5.68	113.28
Dimensions + all engineered features	Random Forest	**99.93**	**3.62**	81.77
Dimensions, product group	SVM	114.48	**5.36**	**95.73**
Dimensions + capacity	SVM	118.00	5.08	160.38
Dimensions + bottle_type	SVM	140.16	5.13	88.88
Dimensions + bottle_type + capacity	SVM	143.62	5.63	94.00
Dimensions + packaging_type_specific_volume	SVM	118.87	**3.88**	**94.16**
Dimensions + all engineered features	SVM	125.67	3.99	**80.47**

Table. 4. Weight prediction result metrics for beer with different feature combinations

All machine learning algorithms deliver significant improvements regarding the observed metrics compared to the heuristic median approach. The best results for each feature combination are highlighted in bold script. The results for the beer data set in table 4 show that the RSME can be more than halved, the mean e_{ap} almost be reduced to a third and the variance of e_{ap} quartered compared to the baseline approach. The Random Forest regressor achieves the best results in terms of RSME and e_{ap} for almost all feature combinations except basic features and basic features combined with the packaging type specific volume, in which cases Support Vector Machines prove superior. Linear regression and SGD are are still better than the baseline approach but not on par with the other algorithms. Linear regression shows the tendency to improved results when successively adding features. SGD on the other hand exhibits no clear relation between number and level of features and corresponding prediction quality. A possible cause could be the choice of hyper parameters. SGD is very sensitive in this regard and depends more heavily upon a higher number of correctly adjusted hyper parameters than the other algorithms we used. Random Forests is a method which is very well suited to problems, where there is no easily discernable relation between the features. It is prone to overfitting, though, which we tried to avoid by using 20% of all data as test data. Adding more engineered features leads to increasingly better results using Random Forest with an outlier for the *packaging type specific volume* feature. SVM are not affected by only first level engineered features but profit from using the bottle type specific volume.

Features	Algorithm	RSME	Mean e_{ap}	Var e_{ap}
Product group	Baseline	38931.0	13.10	1125.00
Dimensions, product group	Linear Regression	**30653.4**	7.55	144.10
Dimensions + capacity	Linear Regression	36168.2	7.41	121.60
Dimensions + bottle_type	Linear Regression	37132.2	7.56	143.80
Dimensions + bottle_type + capacity	Linear Regression	40533.5	7.40	120.50
Dimensions + packaging_type_specific_volume	Linear Regression	42894.9	7.48	**125.47**
Dimensions + all engineered features	Linear Regression	34322.8	7.54	134.60
Dimensions, product group	SGD	34079.4	8.12	131.04
Dimensions + capacity	SGD	**32039.8**	7.83	**95.69**
Dimensions + bottle_type	SGD	**32448.1**	10.31	137.76
Dimensions + bottle_type + capacity	SGD	43545.4	11.69	178.17
Dimensions + packaging_type_specific_volume	SGD	37905.3	8.43	144.81
Dimensions + all engineered features	SGD	**31762.2**	11.57	146.07
Dimensions, product group	Random Forest	30999.9	**6.98**	121.08
Dimensions + capacity	Random Forest	38385.0	**6.84**	124.00
Dimensions + bottle_type	Random Forest	34032.1	**6.65**	**118.17**
Dimensions + bottle_type + capacity	Random Forest	39852.3	**6.45**	**107.87**
Dimensions + packaging_type_specific_volume	Random Forest	35509.5	**7.18**	143.60
Dimensions + all engineered features	Random Forest	34986.3	**6.67**	**118.83**
Dimensions, product group	SVM	39030.4	7.07	**102.15**
Dimensions + capacity	SVM	38132.8	7.18	118.53
Dimensions + bottle_type	SVM	38843.5	7.18	128.90
Dimensions + bottle_type + capacity	SVM	**30109.9**	7.23	140.37
Dimensions + packaging_type_specific_volume	SVM	**35044.4**	7.47	159.21
Dimensions + all engineered features	SVM	38993.4	7.35	135.40

Table. 5. Weight prediction result metrics for wine with different feature combinations

Regarding the wine data set the results depicted in table 5 are not as good as for the beer data set though still much better than the baseline approach. A reduction of the RSME by over 29% and of the mean e_{ap} by almost 50% compared to the baseline were achieved. The variance of e_{ap} could even be limited to under 10% of the baseline value. Again Random Forests is the algorithm with the best e_{ap} metrics. Linear regression and SVM are comparable in terms of e_{ap} while SGD is worse but shows good RSME values. In conclusion the general results of the wine data set show not much improvement when applying additional engineered features.

6 Discussion and Conclusion

The experiments show a much better predicting quality for beer than for wine. A possible cause could be the higher weight variance in bottle types compared to beer bottles and cans. It is also more difficult to correctly determine the bottle type for wine, since the higher overlap in dimensions does not allow to compute the bottle type with the help of idealized bottle dimensions. Using expert knowledge to assign the bottle type by region and grape variety seems not to be as reliable, though. Especially with regard to the lack of a predominant bottle type in the region with the most bottles (red wine from Baden for example), this approach should be improved. Especially Bordeaux bottles often sport an indentation in the bottom, called a 'culot de bouteille'. The size and thickness of this indentation cannot be inferred from the bottle's dimensions. This means that the relation between bottle volume and weight is skewed compared to other bottles without these indentations, which in turn decreases prediction quality.

Predicting gross weights with machine learning and domain-specifically engineered features leads to smaller discrepancies than using simple heuristic approaches. This is important for retail companies since big deviations are much worse for logistical reasons than small ones which may well be within natural production tolerances for bottle weights. Our method allows to check manually generated as well as data pool imported product data for implausible gross weight entries and proposes suggestion values in case of missing entries.

The method we presented can easily be adapted to non-alcoholic beverages using the same engineered features. In this segment, plastics bottles are much more common than glass ones and hence the impact of the bottle weight compared to the liquid weight is significantly smaller. We assume that this will cause a smaller importance of the bottle type feature in the prediction. A more problematic kind of beverage is liquor. Although there are only a few different standard capacities, the bottle types vary so greatly, that identifying a common type is almost impossible. One of the main challenges of our approach is determining the correct bottle types. Using expert knowledge is a solid approach but cannot capture all exemptions. Especially if a wine growing region has no predominant bottle type and is using mixed bottle types instead. Additionally many wine growers use bottle types which haven't been typical for their wine types because they want to differ from other suppliers in order to get the customer's attention. Assuming that all Rieslings are sold in Schlegel bottles, for example, is therefore not exactly true. One option could be to model hybrid bottles using a weighted average of the coefficients for each bottle type in use. If a region uses both Burgunder and Bordeaux bottles with about equal frequency, all products from this region could be assigned a hybrid bottle with coefficients computed by the mean value of each coefficient. If an initially bottle type labeled data set is available, preliminary simulations have shown that most bottle types can be predicted robustly using classification algorithms. The most promising strategy, in our opinion, is to learn the bottle types directly from product images using deep neural nets for example. With regard to the ever increasing online retail sector, web stores need to have pictures of their products on display, so the data is there to be used.

References

1. Domingos, P. (2012). A few useful things to know about machine learning, Communications of the ACM, vol. 55, no. 10, 78
2. Zheng A, Casari A. (2018) Feature Engineering for Machine Learning, Sebastopol: O’Reilly
3. Heaton (2017). An Empirical Analysis of Feature Engineering for Predictive Modeling
4. Anderson (2016). Input Selection for Fast Feature Engineering
5. Breiman, L. (2001). Random forests, Machine learning, vol. 45, no. 1, 5–32
6. Smola, A., Vapnik, V. (1997). Support vector regression machines, Advances in neural information processing systems, vol. 9, 155–161
7. Lai, T., Robbins, H., Zi Wei, C. (1979). Strong consistency of least squares estimates in multiple regression II. Journal of Multivariate Analysis 9.3, 343-361
8. Taddy, M. (2019). Stochastic Gradient Descent. Business Data Science: Combining Machine Learning and Economics to Optimize, Automate, and Accelerate Business Decisions. New York: McGraw-Hill. 303–307
9. GS1 Germany, Global Product Classification (GPC) (2018). https://www.gs1-germany.de/gs1-standards/klassifikation/produktklassifikation-gpc/ (11.03.20)
10. Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection". Proceedings of the 14th International Joint Conference on Artificial Intelligence. 2 (12): 1137–1143