**Utilizing Machine Learning in the Consumer Price Index**

Kristian Harald Myklatun, Statistics Norway, khm@ssb.no[[1]](#footnote-1)

**Abstract**

*The availability of new data sources for use in official statistics has been, and still is, increasing in the recent years. At Statistics Norway, as well as other NSO’s, the use of transaction data has replaced a large share of the more traditional data collection method in the Consumer Price Index (CPI). Machine learning techniques have proved to be an important tool when making use of transaction data.*

*Transaction data provides greater amounts of data, increasing both coverage and accuracy to better capture the dynamic universe of consumption. However, some challenges arise when utilizing transaction data, one of them being classification of data. Data used in the CPI follows the international classification standard of consumption; COICOP. One advantage of the traditional data collection method is that the data is already classified prior to the actual data collection. Transaction data, however, must be classified after the data is collected.*

*Since 2005 data collection for the consumption group Food and non-alcoholic beverages has been fully based on transaction data. Each month 400-1200 new items (new article codes) are introduced to the market of food and non-alcoholic beverages. Previously, the classification of new items was performed in a time consuming and labor-intensive manner. Introducing a new method to automatically classify new items greatly reduced the manual burden. Utilizing machine learning techniques, we are able to automatically link new items to the COICOP.*

*Machine learning has both increased the quality of our CPI, and substantially reduced the time spent on a tedious task at a crucial time in the production cycle. We also believe that machine learning can offer similar effects not only for different sub-indices in the Norwegian CPI, but also for other statistics, and for NSOs in other countries. In this paper we will share our experiences with utilizing machine learning techniques in the production of the CPI.*

**Keywords**: CPI, transaction data, machine learning, SVM, artificial intelligence

## Introduction

The availability of new data sources for use in official statistics has been, and still is, increasing in the recent years. At Statistics Norway, as well as other National statistical offices (NSOs), the use of transaction data has replaced a large share of the more traditional data collection methods in the Consumer Price Index (CPI).

Transaction data provides greater amounts of data, increasing both coverage and accuracy to better capture the dynamic universe of consumption, and it also reduces the burden on respondents, who otherwise would have to fill in a survey each month.

Since 2005, the sub-index of food and non-alcoholic beverages has been based exclusively on transaction data.

Transaction data has the disadvantage of not being labelled according to our classification system. This means that we have to classify the data ourselves. This will quickly become very time-consuming as the mass of data increases.

Our usage of transaction data will only increase in the coming years. This necessitates an effectivization of the classification process. Since early 2017 we have used a form of supervised machine learning called support vector machine (SVM) to assist us in classifying new items in the sub-index of food and non-alcoholic beverages. Our experiences with this tool will be presented in this paper.

## Consumer price index

The consumer price index is a monthly statistic that details the change in prices on goods and services purchased by consumers. We usually receive all our required data before the 1st of each month, and we publish the CPI 10 days after the reference period.

The items in the CPI basket is grouped according to the international standard Classification of individual consumption after purpose (COICOP). The COICOP is a five-tier structure with individual sub-indices at each tier. We have 12 different divisions at the second-tier level. This level is quite generic and includes for example “food and non-alcoholic beverages”, “clothing and footwear” and “education”. For “Food and non-alcoholic beverages”, we use two third-tier groups, 11 fourth-tier classes and 57 sub-classes in the fifth tier. In addition, we have created a sixth tier of very detailed categories under this, as an unofficial COICO6-level. Traditionally, most data used in the consumer price index was either based on questionnaires or price collectors. However, in the last 20 years, advances in technology, as well as changes in consumer behaviour has made it both possible and necessary to explore new data sources.

One of these data sources is transaction data. Transaction data[[2]](#footnote-2) is data extracted from the stores own systems, bases on transactions made by individuals. The use of transaction data has several advantages, both for us, in increasing the quality of the index, and for the data reporters, who get a reduced reporting burden.

Statistics Norway started exploring the possibility of using transaction data from grocery stores in the late 1990’s, and in 2005 it was introduced in full scale for the sub index for food and non-alcoholic beverages, which has a weight of about one-eighth of the total CPI basket. Since then, we have started using transaction data in other areas, and at present about one fifth of the CPI is based on transaction data.

**Figure 1: Data sources used in the CPI, by weight.**

## Classifying transaction data

As mentioned, the index for food and non-alcoholic beverages has been exclusively based on transaction data since 2005. Here I will briefly describe the method we have used previously, which is partly still in use, and in what way machine learning helps us.

All items sold in the grocery stores have a unique code. For most items, this is a chain-independent GTIN-code[[3]](#footnote-3), which uniquely identifies items, by product, manufacturer and type of packaging. For some other goods, mostly fruit and vegetables, a chain-dependent PLU-code[[4]](#footnote-4) is used.

The large grocery stores in Norway all use some sort of classification standard for goods. The most common one is ENVA, which is used by several of the largest chains in Norway. The ENVA-codes are very helpful for us, as they tell us roughly what the items are, and using a mapping catalogue that we have created, they can be used to correctly classify many of the items. However, the ENVA-codes has some limitations.

The ENVA-codes do not follow the COICOP-structure, and it is not detailed enough to be used directly. This means that some ENVA-groups spans more than one COICO6 sub-class. To be able classify these items, we had to supplement the rule-based approach with a text search. This is of the form “if ENVA = XXXX, and TEXT contains “XXXX”, then COICOP6 = XXXXXX”. This text search had to be continuously updated with new words that helped split the ENVA-groups. We also had to manually update the mapping catalogue between ENVA and COICOP when new ENVA codes were put into use.

The biggest challenge with our previous method was to manually check all new items whether they were correctly classified.

Machine learning helps us conduct this task more efficiently. The chains own classification standard is an important input for the model, but we do not have to manually maintain the model, since it is self-learning. It is not able to classify all items correctly, but it gives us an estimate of how sure it is, which means that we can limit the amount of items we check to the ones the model is uncertain about.

Several other NSOs are experimenting with using machine learning for classification purposes. Some recent examples are Martindale, Rowland and Flower (2019) in the UK HICP/CPI and Roberson (2019) in the US using it for product categorization. Yung et al. (2018) gives a general overview about the possibilities for machine learning in official statistics, and Beck, Dumpert and Feuerhake (2018) gives an overview of which NSOs are using machine learning, and for what purpose.

## Bag of words

Before we begin explaining the machine learning technique itself, we need to explain how we are able to work with text data. This requires us to first transform the data to a format that we can work with. We do this by using a so-called bag-of-words model (BoW).

Using the bag-of-words model, we create a matrix containing all unique words in the training set, and the frequency with which each of them occurs in each item. We can illustrate this with a very small sample from our training set.

Table 1 shows four different observations for pizza, represented in a matrix format. In this paper, a row in this matrix will be referred to as either an item or a document, and a column will be referred to as a word or a term.

**Table 1: Matrix of chosen training set items.**



It is important to note that the computer does not understand any of the words, or see any meaning in them. For the computer, the word “VEGETARPIZZA” (vegetarian pizza) has no connection to the words “VEGETAR” and “PIZZA”. It also has problems with abbreviations and spelling errors, both of which are common in the GTIN texts.

## Machine learning for classification

There are many different methods for classifying text data. In addition to Support vector machines, logistic regression and naïve Bayes are some simple and popular baseline methods. I will briefly introduce these methods before focusing on Support vector machines in next section.

### Logistic regression

Logistic regression is familiar to many as a regression method for binary response variables. It can also be used to classify items. In general, classification- and regression tasks have a similar objective – to find the mapping function between the predictors and the response, so that we can make a good prediction of what the response would be with a new set of predictors. With logistic regression, the output is the odds of a certain event occurring. When we use logistic regression for classification, this event is a class.

As mentioned, logistic regression is used when you have a binary response variable, while we in this case want to classify new items into one of many classes. There are several ways to do this, but the conceptually simplest one is a one-vs-all approach. With this approach we separately estimate the probability of the item belonging to either class 1 or any other class, class 2 or any other class, and so on. In the end we are left with the probability of the new item belonging to each possible class. The item is then given the class with the highest probability.

### Naïve Bayes

Naïve Bayes is a simple and fast method for classification, and is therefore often used as a baseline method for comparing performance.

The Naïve Bayes method is based on the Bayes’ theorem, and much of the calculation is done simply by counting word frequencies. The Bayes’ theorem is the following:

$$P\left(B\right)= \frac{P(B|A\_{i})×P(A\_{i})}{P(B)}$$

In this setting, we can interpret Ai as a class, and B as a feature, or a set of features. So we calculate the likelihood of a new item being in a certain class, given a certain set of features. This is equal to the likelihood of having those set of features given that the items belong to that class, multiplied by the likelihood of having the class, divided by the likelihood of having the features. However, since we are only using $P(A|B)$ for comparison between different classes (values of A), we can ignore the divisor.

To use a practical example, lets say we have a new item with the text “jasmine rice”. To classify this item to the correct class, we calculate the above for all classes. We also assume that every word in the document is independent, so that we get:

$$P\left(A\_{i}\right)=P(jasmine|A\_{i})×P(rice|A\_{i})$$

Where for example $P(jasmine|A\_{i})$ is simply the share of words in class *i* that is “jasmine” in the training set. We then simply calculate the following for all classes:

$$P(jasmine|A\_{i})×P(rice|A\_{i})×P(A\_{i})$$

Where$ P(A\_{i})$ is the share of all classes in the training set that is class *i*. The document is then classified to this class.

## Support vector machine

Support Vector Machine (SVM) is a method of statistical learning developed by Vapnik (1979)[[5]](#footnote-5), but not popularized until the 1990’s, with Cortes & Vapnik (1995). According to Joachims (1998), SVM is particularly well-suited for classifying text data. The basic objective of SVM for classification is to optimally separate classes based on a set of features. We can illustrate this in a simple two-class problem as in Figure 2. To the left we have the simplest case, where the classes are linearly separable. The SVM algorithm separates the classes by maximizing the distance to the closest items of each class. These closest items of each class on each side of the line are called support vectors.

We are not always able to separate the classes with a straight line[[6]](#footnote-6). Non-separability complicates the problem somewhat, as illustrated in the figure to the right. To solve this problem we have to introduce an additional constraint: that the overlap, $C=\sum\_{}^{}α\_{j}$ should not exceed a given limit. The size of C is a trade-off between bias and variance; setting it too high will give a very good fit for your training set, but will not perform well with your actual data (overfitting), while setting it too low might lead to the model being too simple (learning very little from the training set), and therefore not being very accurate (underfitting). The appropriate value of C is set by using cross-validation. For text data, we often, but not always, have linearly separable classes (Joachims, 1998).

**Figure 2: Illustration of 2-class support vector machine**

$$α\_{1}$$

$$α\_{2}$$

When we have more than two classes, we start by first repeating the above process between each class. We call this a 1-versus-1 approach (1-v-1). This divides our feature space into several different regions (sub-sets). The class awarded to new items in a certain region is then decided by a form of majority vote. In this context this means that for each one-versus-one division, we repeat the 1-v-1 approach as detailed above, and the region on the “home side” of the line receives a vote for the relevant class. The class that receives the most votes in a region “wins” the region. This is illustrated in the figures below.

**Figure 3: Illustration of 3-class support vector machine**

1

2

3

4

5

6

A

B

C

Table 2 shows the voting matrix of Figure 3 For example, the 1-v-1 between blue and red draws line A, showing that blue receives a vote in areas 1, 5 and 6, while red receives a vote in areas 2, 3 and 4. The class that receives the most votes in each area “wins” the area, which means that all new items in this region receives this class.

**Table 2: The voting matrix for Figure 3**

|  |  |  |  |
| --- | --- | --- | --- |
| Area | Blue | Red | Green |
| 1 | 2 | 1 |  |
| 2 | 1 | 2 |  |
| 3 |  | 2 | 1 |
| 4 |  | 1 | 2 |
| 5 | 1 |  | 2 |
| 6 | 2 |  | 1 |

This results in Figure 4, where the areas have been divided according to the voting matrix above. The standard SVM is non-probabilistic, and therefore does not give a probability estimate. The R package we use works around this to provide us with an estimate of how sure it is that an item belongs to a certain class, but I will not go into the technicalities of how this is done, other than to provide a simplified and intuitive explanation.

In the example illustrated in Figure 4, two new items (X and Y) enter the feature space. We see that they are both awarded the green class. However, the model is less certain about its classifications near the border regions (where Y is), than it is for the observation X, which is more similar to the existing observations in the green class. It will therefore assign a greater probability for X belonging to the green class than for Y.

**Figure 4: 3-class problem, with each region divided.**



## SVM for food and non-alcoholic beverages

As mentioned, we have created 117 different categories of goods at the lowest level within the division of food and non-alcoholic beverages, ranging from rice to fruit juice. The approach we use compares each of these groups against every other group. This means that we train almost 7000 different models each month.

When we started using SVM, we first had to create a high-quality training set. We started with a few thousand pre-classified unique items. The variables we used were the GTIN, the text of the items, the ENVA code, and the COICO6-code.

After running the SVM, each new item receives a probability that it belongs to each of the 117 classes. The class with the highest assigned probability is predicted to be the correct class for the item.

We then have to decide which items to check. The intuitive approach may consist of only checking the items where the highest assigned probability is below a certain threshold. To see why this is insufficient, imagine an item that is assigned a very high and similar probability for two different classes, and very low for the rest. In this case the probability for the first choice is high, but the certainty about the classification is low.

For this reason, the criteria is a combination of low probability and a low relative between the probability of the first and second choice.

After we have checked the uncertain items, the new items are then added to the training set with the correct classification. In this way, the accuracy of the model improves over time. Our training set now contains all items that we have previously classified with the help of the model since early 2017, at present amounting to about 30 000 unique items.

## Workflow integration

The machine learning model is well integrated with our regular procedure. This limits the amount of staff training necessary to make use of the benefits machine learning offers. Here I will briefly show how the machine learning fits into our regular production process for this index. The purpose of this is to give readers an insight in to what the practical implications might be of using machine learning in the production of official statistics.

As with most other elements of the Norwegian CPI, the production process for the index of food and non-alcoholic beverages is programmed in SAS. The first steps are to read in the transaction data from the different grocery store chains and gather all items in a single file. We then merge this file with the existing production data set. For the items with GTIN-codes that have already been classified, this implies merging it with a pre-existing entry, adding a new price for the latest month. The new items are separated in a different file and given a format that fits our SVM training set.

We then run the machine learning program. The program is written in R, but we simply run it via the Linux console, without actually opening any R software. This makes it very easy to use, even for co-workers who have never used R. The R program reads the SAS-dataset of new items directly, runs the support vector machine, and then adds the predicted COICO6, the probability estimate, and the relative between the probability of the first and second choice, and writes a new SAS-dataset which can then be used directly in SAS.

We use the information provided by the R program to select which items to check. The procedure of checking the items is the same as before, except that there are a lot fewer items to check. We check items using the FSEDIT procedure in SAS, which gives us all the relevant information in a single window, including the information the machine learning procedure provides. An example of what the interface looks like is included in the appendix, along with an example of what the output from the machine learning model looks like.

## Effects

Using only the rule-based classification with manual checks on all new items, the classification work took experienced co-workers 6-12 hours of tedious work each month, depending on the number of new items in the given month. With the help of machine learning we have reduced this substantially, and the amount of time it takes is now usually between 2 and 6 hours. This is a substantial improvement which allows us to spend more time on other more rewarding tasks.

As previously mentioned, the weight of the index of food and non-alcoholic beverages is about one eighth of the total in the CPI. This means that inaccuracies in this index is likely to have an impact on the CPI total. With transaction data for this type of goods, one of the biggest sources of error is misclassification of items.

Our previous approach was decent, and automatically classified most items correctly. However, when manually checking up to 1200 items for up to 12 hours, some misclassifications are unavoidable. With the new approach, we substantially reduce the number of items we check manually. This not only reduces time spent checking the items, but it also allows us to spend more time on each item. The result is that the chance of misclassification is substantially reduced.

## Performance metrics

Two different metrics are important to us. Firstly, how good the model is at classifying items correctly, and secondly, how many of the items are picked for individual checks (the share of items the model is uncertain about). For the first, we check the performance of a support vector machine model compared to that of logistic regression and naïve Bayes, and for the second, we look at the distribution of certainty for the model we currently use.

We can check the performance of a model by repeatedly splitting the training set, and training on one part and testing it on the other part. One way to do this is in a procedure called the Monte Carlo cross-validation, which is detailed in Kuhn & Johnson (2013). This is what we use for naïve Bayes. For SVM and for logistic regression we use a repeated K-fold cross validation.

The three algorithms have very similar rates of accuracy. We use a different implementation of the support vector machine here from the one we use in production. The one used here is closer to an out-of-the-box algorithm, increasing reproducibility and comparability with the other algorithms. It also gives slightly higher accuracy than our previous algorithm. Unfortunately, it is also considerably slower.

It is not surprising that SVM gives the highest accuracy, as it is generally considered state of the art for this type of problem and training set size. The good performance of Naïve Bayes is more surprising. The difference in performance between SVM and naïve Bayes is near-negligible in this context, and its conceptual simplicity and speed makes it a good candidate for someone considering a similar project. Note that using term frequency-inverse term frequency[[7]](#footnote-7) significantly increases the performance of naïve Bayes, as detailed in Rennie et al. (2003). Logistic regression is also a very high-performing algorithm for this data, but is also by far the slowest. Note that these training times could be significantly improved, for example by using a sparse matrix representation (which is used in our old SVM implementation).

**Table 3: Accuracy and training time of chosen models**

|  |  |  |
| --- | --- | --- |
| Algorithm | Accuracy | Time |
| Support vector machine (new) | 90,2% | 24,7 minutes |
| Support vector machine (old) | 86,1% | 4,2 minutes |
| Naïve bayes (tf-idf) | 87% | 2,1 seconds |
| Regularized Logistic Regression | 89,3% | 6,5 hours |

We should note that the accuracy rate here will probably never reach 100 percent, because some items could fit in more than one sub-class, some item texts do not have enough information to classify it correctly, and some items may be given the wrong ENVA-group by the chains.

With the SVM implementation we use today, we achieve an overall accuracy rate of 86,1 percent. We manually check items where the SVM assigns a probability rate less than 20 percent, or alternatively a relative probability between the first and second choice of class that is less than four. The accuracy varies with the strictness of our control criteria.

We see that with our current criteria, the accuracy rate is 95,4 percent for items not picked for individual checks. The accuracy increases as we tighten the criteria, meaning that more items are selected for checks, but it is doing so at a decreasing rate. The criteria is a trade-off between accuracy and time spent doing manual checks.

**Table 4: Accuracy of SVM model, and share of items controlled under different control criteria.**

|  |  |  |
| --- | --- | --- |
|  | Accuracy | Share of items manually controlled |
| Relative > 1 and probability > 0.1 | 86,8% | 2,6% |
| Relative > 2 and probability > 0.15 | 91,4% | 12,8% |
| Relative > 4 and probability > 0.2 | 95,4% | 19,7% |
| Relative > 5 and probability > 0.2 | 96,2% | 21,9% |
| Relative > 5 and probability > 0.3 | 96,2% | 23,1% |
| Relative > 10 and probability > 0.5 | 97,5%  | 33,3% |
| Relative > 15 and probability > 0.5 | 98,0% | 38,4% |
| Relative > 20 and probability > 0.5 | 98,4% | 42,8% |

The share of items picked for a manual check also varies with our criteria. With our current criteria, we manually check just less than 20 percent of the new items, constituting a major reduction in time compared to not using machine learning.

Although the accuracy is very high for items not picked out for individual checks, we see that there are some errors in this material as well. As mentioned, some of these items could fit into more than one sub-class, and should therefore probably not be counted as errors, but there are some errors in the set. This is an issue for us, so we abate this in two ways. Firstly, we very briefly look over the list of items not picked for individual checks. Because we are only looking for some very few errors, generally in the borderline of our checking criteria, this check only takes about ten minutes, and we believe we are able to find most errors this way. Secondly, we do a yearly thorough clean-up of our production data and our training set[[8]](#footnote-8).

## Conclusion

Machine learning has both increased the quality of our CPI, and substantially reduced the time spent on a tedious task at a crucial time in the production cycle. We also believe that machine learning can offer similar effects not only for different sub-indices in the Norwegian CPI, but also for NSOs in other countries.

However, machine learning is not a magic pill. It does not eliminate the need for manual controls, and it can’t be used for all data sources and indices. It also requires that at least some people have the skills needed to set it up and maintain it. If something goes wrong you can go back and manually classify all items. However, this would require substantially increasing the amount of man-hours spent on the production of the CPI, at short notice in a strained period. While there are limitations to the usefulness, and considerations that should be taken when considering cost of skills training and work-planning, I do not believe that these issues should stop anyone from exploring the use of machine learning in official statistics.

As this is a method that offers the possibility of producing better statistics at a lower cost, and considering the ever-increasing usage of transaction data and other non-classified data, I believe that exploring the possibilities that machine learning offers for the purpose of classification is a must for all NSOs handling this sort of data.

## References

Beck, M., Dumpert, F., Feuerhake, J. (2018). Machine Learning in Official Statistics. arXiv preprint arXiv:1812.10422.

Caruana, R., Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd international conference on Machine learning (pp. 161-168). ACM.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), (pp. 273-297).

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? The Journal of Machine Learning Research, 15(1), (pp. 3133-3181).

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning (pp. 137-142). Springer, Berlin, Heidelberg.

Kuhn, M., & Johnson, K. (2013). Applied predictive modelling (Vol. 26). New York: Springer.

Martindale, H., Rowland, E., Flower, T. (2019). Semi-supervised machine learning with word embedding for classification in price statistics. For Ottawa group meeting, 8-10 May, Rio de Janeiro. Available online: <https://eventos.fgv.br/sites/eventos.fgv.br/files/arquivos/u161/semi-supervised_ml_for_price_stats-ottawa_group.pdf>

Rennie, J. D. M., Shih, L., Teevan, J., Karger, D. R. (2003). Tackling the Poor Assumptions of Naive Bayes Text Classifiers. Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC.

Roberson, A. (2019). Applying Machine Learning for Automatic Product Categorization. New Techniques and Technologies for Statistics (NTTS), 12-14 March, Brussels. Available online: <https://coms.events/ntts2019/data/abstracts/en/abstract_0156.html>

Vapnik, V. (1979). Reconstruction of Dependences from Empirical Data. Nauka, Moscow.

Vapnik, V. (1963). Pattern recognition using generalized portrait method. Automation and Remote Control, 24, (pp. 774–780).

Yung, W., Karkimaa, J., Scannapieco, M., Barcarolli, G., Zardetto, D., Sanchez, J. A. R., Braaksma, B., Buelens, B., Burger, J. (2018). The Use of Machine Learning in Official Statistics. UNECE Machine Learning Team report.

## Appendix

**Figure 5: An example of an output showing the results from the machine learning.**



**Figure 6: The *FSEDIT* window we use for checking individual goods.**



1. I am grateful for useful comments by Randi Johannessen and Morten Madshus. Credit to Leiv Tore Salte Rønneberg for the idea and the original implementation of the algorithm. [↑](#footnote-ref-1)
2. In this paper, transaction data means scanner data, which is defined as follows by OECD: detailed data on sales of consumer goods obtained by ‘scanning’ the bar codes for individual products at electronic points of sale in retail outlets. [↑](#footnote-ref-2)
3. GTIN: Global Trade Item Number, standard barcode. [↑](#footnote-ref-3)
4. PLU: Price look-up code. [↑](#footnote-ref-4)
5. The history goes back to Vapnik and Lerner (1963), and further, but the method was not called by its current name at that point, and it was not as refined. [↑](#footnote-ref-5)
6. Technically a hyperplane when we are talking about higher dimensions. [↑](#footnote-ref-6)
7. Downweighing terms that occur in many documents. [↑](#footnote-ref-7)
8. The index is chained monthly, and calculated via a geometric average of relatives (Jevons index), where only items present in both months are included. This means that misclassifications only affect our index if the price development of the item is different than the sub-class it has been put in, and considering the mass of our data, the difference must be substantial to affect the index significantly, and this would be discovered in controls. [↑](#footnote-ref-8)