**Machine learning in ITGS**

*– Validating the easy way*

Jan Olav Rørhus, Statistics Norway, [jan.rorhus@ssb.no](mailto:jan.rorhus@ssb.no)

***Abstract***

*International trade in goods statistics (ITGS) is published with a very high level of detail. In the Norwegian customs tariff there are currently more than 7 000 different classifications codes which we release statistics for. In addition, the data material which the statistics is based on is large and contains a substantial amount of errors. Due to these circumstances, considerable resources are used in the editing. Our goal has been to edit data more effective, and we show that, given the correct type of data, machine learning models can be used to control the correctness of the classification of goods.*

*We have used machine learning to build a model that maps which classification code is most likely given a text string included in data received from Norwegian customs. The first step in the analysis is to split the data into a training and a test dataset. The training dataset is used to build the predictive model. The model is then applied to a test dataset.*

*The model creates a matrix containing the likelihood of each classification for all records in the test dataset. By looking at occurrences where the predicted classification deviates from the current classification code in use, we identify records which need to be investigated further. A big advantage of this model is that it provides not only an indication of suspected misclassifications, but also a suggestion for the correct classification. Also, the model provides quantified measures of suspicion which can be used in conjunction with other metrics to make objective and effect-driven priorities in our validation efforts.*

***Keywords:*** *Machine learning, register data, international trade*

1. **Background**

International Trade in Goods Statistics (ITGS) measures the value and volume of trade flows between countries, broken down by good classification. Figures are also published at a very detailed level. In 2016 the Norwegian custom tariff had almost 7 000 different commodity codes, there were 252 different countries and the import/export dimension of trade. All these categories are part of the dissemination and brings the potential number of combinations at 3.9 million. Of course, far from all these combinations have figures in them, as there is no trade for all combinations. In 2016, roughly 180 000 of the combinations had trade, which amounts to roughly 5 per cent of the possible.

In 2016 ITGS was based on more than 18 million trade observations. Some of these observations are reported by customs or the trader itself, but the largest proportion by far is filled out by declarants who work as agents for the traders. The declarant usually receives a fee per observation reported, which creates an incentive to process as many observations as possible using as little time as possible. A consequence of this is that data quality is often not prioritized.

|  |  |
| --- | --- |
| **Table 1: Number of observations in ITGS, 2015-2018** | |
| **Year** | **Observations** |
| 2015 | 17 821 364 |
| 2016 | 18 210 443 |
| 2017 | 19 086 032 |
| 2018 | 20 353 525 |

Source: Statistics Norway 2016

Due to the quality issues and the high number of observations and variables, it is demanding to maintain good quality of the statistics. This is especially relevant when breaking figures down to the most detailed level. To increase efficiency and to improve quality, statistical methods can be of great help in the data validation process. In this paper we will look at how machine learning can be utilized to validate the correctness of the classification of merchandise. Erroneous classification can have a large effect on the figures. Some products are very expensive and/or heavy compared to other products, and when erroneously classified these products can lead to misleading figures.

1. **International Trade in Goods data**

Statistics Norway receive the data for ITGS through the Norwegian customs’ electronic system for exchanging customs declarations (TVINN[[1]](#footnote-1)). In addition to a classification code, declarants also provide a short text describing the commodity. The text is filled out based on information from the invoice. It is primarily these two variables that are used in the machine learning process. The process itself is performed by trying to predict commodity codes based on the text field. This is basically the same approach as when data is manually validated.

In ITGS we use the 8-digit Norwegian Custom Tariff codes for classification purposes. It is this classification code we try to predict in this study. The first 6 digits of the code are given by the International harmonized system (HS), while the last 2 digits are national adaptions and differ between countries. Further, the first 2 digits of the classification specify which chapter a good belongs to, ranging from 01 - 97. As the code progresses from the lowest chapter, the degree of processing also increases. Chapter 01, for example, contains live animals, while the highest numbered chapters contain goods like vehicles, aircrafts and weapons. This arrangement makes it less likely that a correction of an erroneous classification code leads to change in chapters.

A change of the classification code is, as of today, a rather unlikely event. In 2016 less than 0.4 per cent of the observations had its classification changed. Table 2 shows the frequency and the severity of changes in classification. Approximately 6 per cent of the changes in classification code involves a change in chapter. In other words, 94 per cent of the changes are restricted to the same chapter, most changes appearing at position 5 and 6.

|  |  |  |
| --- | --- | --- |
| **Table 2: Change of classification, number of observations 2016** | | |
| **Position changed** | **Number of changes** | **Percentage** |
| 1&2 | 3 985 | 5.7 |
| 3&4 | 17 068 | 24.3 |
| 5 | 19 495 | 27.7 |
| 6 | 15 588 | 22.2 |
| 7&8 | 14 174 | 20.2 |

Source: Statistics Norway 2016

There can be several reasons for the overall low number of changed classifications. One likely reason is that most observations are probably correctly classified. Another possibility is that statisticians have been unable to identify all the errors. It is time consuming for a statistician to validate a classification code – observations must be compared with each other to assess the chance of the classification code being incorrect. Because of this, we usually only audit observations with large impact on an aggregate level - that means observations with large values. Thus, it is likely that there are a considerable number of misclassifications that are never identified. It is worth mentioning that if classifications are systematically false they can lead to erroneous prediction results.

* 1. *Data selection*

We have chosen to use 2016 data because this was the last finalized year at the time of writing. This means that we no longer edit these data, so it is most likely of better quality than data from later years. We have also chosen to use data from only one year to avoid confusing the model with changes in the classification. Each year there is a fair amount of splitting and merging of codes as well as discontinuations and new codes. When implementing the model in production, this is an issue that needs to be taken into consideration to harvest its full potential.

Due to the many observations, constraints in computing power and need to limit the analysis, we have restricted the data used in the study to chapters 19, 20, 25 and 57. These chapters have been chosen based on diversity in sub-classifications, number of unique words in the text variable and the share of observations where the classification has been changed. By incorporating chapters with a variety of these properties we hope to reveal how the model fares under different circumstances. The number of string combinations is included because we suspect that this has a negative effect on the prediction capability of the model. We believe that chapters with many different string combinations have a large variety in goods, which makes the job of the model hard. Detailed statistics about the selection criteria can be found in table A1 in the appendix.

Chapter 19 – “Preparations of cereals, flour, starch or milk; pastrycooks products” has been selected because it has many unique string combinations. In addition, 1.8 percent of the observations had the classification code altered.

Chapter 20 – “preparations of fruits, vegetables and nuts” has been chosen because it contains relatively many classification corrections and many unique string combinations. A total of 2.1 per cent of the observations have had the classification changed and there are 6 218 unique string combinations with a count of more than 1.

Chapter 25 – “Salt, Sulphur; earths and stone etc., “ is with 10.4 per cent, the chapter with the second highest degree of change in classification. In addition, this chapter, has more than 31 000 observations in the test data set.

Chapter 57 – “Carpets and other textile floor coverings” has been included because it has few corrections. In this chapter less than 0.1 per cent of the classifications has been changed. In addition, the chapter has enough observations. The challenge with chapters that has few corrections is that it is difficult to quality assure the validity of the occurrences where the prediction deviates from the current classification.

* 1. *Variable selection*

Of all the variables available in data only three are used in the analysis. The dependent variable in the analysis is as earlier mentioned the classification or commodity code (customs code).

Two explanatory variables are used, the most important one being the commodity text variable, which contains one or several words/strings describing the good. The declarant is free to type in any text in this field, thus misspellings regularly occurs. We have tried to remove some of the noise and special signs from the text variable using regular expressions, but it is difficult to say whether this has had a positive impact on the results. One goal of doing this is to make words more equal. This is necessary since the model will differentiate between two words, where one has a hyphen and the other does not. Traditionally the brunt of the work in machine learning lies in prepping the data, so there is a large potential in improving the text strings before feeding them to the model.

We have also included the 9-digit organization number for the trading enterprise as an explanatory variable. This is included to differentiate between enterprises in the model. We could also include the origin and destination country in the model, but this far we have not done this.

1. **Model**

We have limited the model to only search for misclassifications within the same chapter. We have done this because the matrix built by the model gets very large, and with large datasets this makes it difficult to process all the data simultaneously. This limitation means that the model will not identify erroneous classifications between chapters.

We have used Random Forest Model (RFM) to run the predictions in this study. Initially we also tested Support Vector Machines (SVM) Model, but there were little differences in the results between the two. We chose to use the RFM model since it is more familiar to us. The model has been implemented and run in Python[[2]](#footnote-2) with default parameters. The model is specified as follows:

(1)

Where the classification is the dependent variable, the “enterprise\_id” is the 9-digit enterprise number and “text” being the commodity text field.

We are not going to delve deep into the inner workings of the RFM model, as this is both beyond our knowledge and scope of this study. We will nevertheless give an overall explanation of the model.

The RFM can be said to be an extension of a more basic model called decision trees. In decision trees a hierarchy of if/else questions is built – where each clause/node limits the classification problem and pare off some of the possible codes. One major drawback of using decision trees is that it is prone to overfitting. Overfitting occurs when the model is trained to an extent where it is too detailed or well fitted to the training data. One example of this is that the model is built to accommodate outliers, while it probably would be better to ignore these. Overfitting usually leads to good prediction correctness on the training data, but these results seldom transfer to test data. It is after all the classification codes in the test dataset we want to predict, not the other way around. Overfitting can happen in any machine learning model and the usual way to avoid it is to limit the detail level of the model (Müller & Guido, 2016).

When building a Random Forest Model, multiple decision trees is built where each is randomized from each other. It is possible to specify how many trees the model should construct – we have chosen 10). The randomization process is implemented in two ways. First, each tree is built using a bootstrap sample of data. That is, if we have *n* data points in our training data set, we draw *n* observations from this data set with replacement. By “with replacement” we mean that each observation can be selected multiple times. We thereby end up with a sample that is as big as the original, but where some observations will be left out, while others will occur multiple times (Müller & Guido, 2016). This process is repeated for each tree in the forest. This way outliers are not included in all the trees/datasets, thus reducing the effect of overfitting.

Decision trees are then built for the different samples, but where the decision tree uses all the features of the dataset to construct the tree, only a subset of the features is used in the RFM. The model then constructs the best test using these features. Finally, the results from the different trees are averaged, leaving us with a likelihood for each classification.

The probabilities for all classifications for each observation end up in a probability matrix, which contains one column for each classification in the dataset and one row for each observation. The cells in this matrix contains the likelihood of each of the classifications. The classification with the highest likelihood is selected as the prediction . In addition, the likelihood of the current classification is extracted. By dividing by we can find the relative probability, describing the relationship between the predicted and the current probability (henceforth referred to as *Rc*).

**(2)**

The measure *Rc* is a measure of how likely the predicted code is relative to the likelihood of the currently used classification code. A value of 1 is the smallest possible value we can observe for the *Rc*, and it indicates one of two possibilities:

1. The predicted and current classification code coincides.
2. The predicted code differs from the current classification code, but their likelihoods are equal.

The higher *Rc* is, the more likely it is that the current code is incorrect. When *Rc* is infinite, which is common, this means one of two things.

1. The model is very uncertain about which classification code is correct. Therefore, no classification code has a high likelihood. On the other hand, the model is very sure that the current classification code is false – thus giving this code a likelihood of zero. This results in a *Rc* that is infinite.
2. The model is very certain that the predicted classification code is correct and very sure that the current classification code is false.

The challenge is to identify which of the observations, with a *Rc* equal to infinite, that belong in category (1) and which belong in (2). We want to study the observations in (2) further and abort any further investigation of the ones in (1). One way of doing this is to calculate a second relative value, one that describes the relationship between the likelihood of the predicted and second highest classification code. The formula for this is given below in (3).

**(3)**

By using both these relatives it is possible to separate the observations in (1) and (2) mentioned above. If we select the observations where both *Rc* and *Rs* approach infinity, we should only get observations where both the predicted code is likely and the likelihood for the current code is zero.

Input data for the model are randomly split into a training and test dataset, the former containing 75 per cent of the observations and the latter 25 per cent. The intention of the training dataset is to construct an algorithm that predict the classification codes in the test data set.

1. **Results**

To analyse the results, we have chosen to split the observations into two groups. Group 1 contains those observations where we have corrected the classification code, while group 2 contains the ones where it is unchanged. To compare our manual corrections with the prediction results we have split these groups further into the following groups:

* The predicted code equals the classification code used today.
* The predicted code differs from the classification code used today.

The prediction results are shown below in table 3 and 4

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 3: Prediction results, number of observations, 2016** | | | | |  | | |  | |  |
|  |  | | **Group 1** | |  | | | **Group 2** | |  |
| **Chapter** | **Total number of predicted observations** | | **Prediction equal current classification** | | **Prediction differs from current classification** | | | **Prediction equal current classification** | | **Prediction differs from current classification** |
| **19** | 43 863 | | 796 | | 13 | | | 41 464 | | 1 590 |
| **20** | 18 285 | | 285 | | 59 | | | 16 835 | | 1 106 |
| **25** | 7 696 | | 581 | | 11 | | | 6 589 | | 515 |
| **57** | 12 206 | | - | | 2 | | | 9 682 | | 2 522 |
| **Total** | 82 050 | | 1 662 | | 85 | | | 74 570 | | 5 733 |
| **Table 4: Prediction results, probability, 2016** | | | | | |  |  | |
|  | | **Group 1** | |  | | **Group 2** |  | |
| **Chapter** | | **Prediction equal current classification** | | **Prediction differs from current classification** | | **Prediction equal current classification** | **Prediction differs from current classification** | |
| **19** | | 98.4 | | 1.6 | | 96.3 | 4 | |
| **20** | | 82.8 | | 17.2 | | 93.8 | 6 | |
| **25** | | 98.1 | | 1.9 | | 92.8 | 7 | |
| **57** | | . | | 100 | | 79.3 | 21 | |
| **Total** | | 95.1 | | 4.9 | | 92.9 | 7.1 | |

Source: Statistics Norway

The total for group 1 shows that the model identifies most of the classification errors that have been audited manually. This is most prevalent in chapter 19 and 25 where the predictions match about 98 per cent of the manual corrections. Chapter 20 has, with 82,8 per cent, a considerably lower hit ratio, while chapter 57 is not suited for any analysis in group 1 since it has few corrections. There is a considerable diversity in unique strings used in the commodity text variable in chapter 20. More than 6 000 string combinations with a count larger than 1 are found in this chapter. The diversity is even larger for chapter 19, with 14 000 string combinations, and this chapter as mentioned earlier also has a better hit ratio. This makes it uncertain whether the number of strings in the text has a big impact for the prediction results.

In group 2 there are overall a lot more observations than in group 1. However, the percentage of observations where the predicted and current classification coincide are only marginally lower than for group 1. In chapter 19 for example, 96 per cent of the predictions correspond with the current classification code. The observations that differ from the current code still constitute almost 1 600 observations, which would take a considerable time to control manually. As a comparison we see that the total number of corrected observations for chapter 19 is almost 800. This means that we would have to control twice as many observations as we already have corrected if we were to control all the observations with differing prediction codes here. In chapter 57 approximately 80 per cent of the predictions hit the current classification code. This is considerably less precise than what we saw in the other three chapters. One reason for the low hit rate can be that the absence of corrections makes the data in this chapter less prepared for machine learning models. Data used in machine learning need to be consistent and prepared for the model to work properly.

In group 2 some of the predictions that differ from the current codes are likely to be misclassifications that auditors have been unable to identify earlier. Probably not all the differing predictions can be owed to this explanation, but it is difficult to draw any conclusions about the distribution without controlling the observations, which is not done here.

* 1. *Likelihood of prediction*

One very useful property of RFM is that it supplies a likelihood for each classification possibility. The classification with the highest likelihood is select as the prediction, but the corresponding likelihood does not need to be particularly high. The predicted code might just have the highest likelihood among many. It is therefore valuable to use the likelihood information to further validate the results. In table 5 we have presented the average likelihood for the groups presented in table 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table 5: Likelihood of prediction results, likelihood, 2016** | | | |  |  |
|  |  | **Group 1** |  | **Group 2** |  |
| **Chapter** | **Predicted observations** | **Prediction equal actual classification** | **Prediction differs from actual classification** | **Prediction equal actual classification** | **Prediction differs from actual classification** |
| 19 | 43 863 | 0.98 | 0.63 | 0.97 | 0.53 |
| 20 | 18 285 | 0.91 | 0.59 | 0.96 | 0.51 |
| 25 | 7 696 | 1 | 0.66 | 0.96 | 0.57 |
| 57 | 12 206 | . | 0.54 | 0.91 | 0.46 |

Source: Statistics Norway 2016

We can see that in group 1, where the prediction equals the current classification, the likelihood is overall very good. For chapter 19 for example the model is on average 98 per cent certain that the predictions are correct for this category. The corresponding likelihood for chapter 20 and 25 is 91 per cent and 100 per cent respectively. These results seem to imply that we should prerequisite that the prediction likelihood is high before we use any time on manual controls.

For group 2 we see that the model is also very certain of the results where the prediction equals the classification in use today, ranging from 91 to 97 per cent. In comparison the model is not as certain for the observations where the prediction and current classification code differ. Here the average likelihood lies around 50 per cent.

* 1. *Distribution of likelihood of prediction results*

Figure 1 below describe the likelihood distribution of the observations in group 2 where the predicted code differs from the current code in use. The x-axis shows likelihood intervals for the prediction being correct, while the y-axis shows the distribution of the observations.

**Figure 1: Likelihood distribution per chapter, per centage within chapter, 2016**



Source: Statistics Norway 2016

As we saw earlier in the results for group 1, it is not the best use of resources to control the observations with a low likelihood. After all, a low likelihood means that there are other classifications that probably have similar likelihoods. A prediction with a high likelihood (e.g. 90 per cent) does not only mean that this classification is likely. It also indicates that the model is unable to find any other classifications that are likely. If we were to demand that the likelihood of a predicted code had to be above 90 per cent before we performed any manual control we would trim the number of observations needed to control dramatically down from 5 733 to 395 (counting all 4 chapters). Given the high likelihood it is probable that we would end up correcting a high share of these observations. We have not done any analysis of what the actual hit rate on these variables, but this should be straightforward to perform given the time.

* 1. *Relative value*

The two relatives *Rc*  and *Rs,* can be of great help in validating the results from the model by summing up information about the likelihoods that are most important.

We have split the observations in group 2 where the predicted code differs from the current classification into three groups. The observations have then been plotted in a scatter plot where the x-axis shows , while the y-axis shows .

**Figure 2:** *Rc* **vs** *Rs***, 2016**



Source: Statistics Norway 2016

First, if both *Rc* and *Rs* are approaching infinity we can be sure that the predicted classification code has a high likelihood and that there is no other classification code competing. For 167 of the observations both relatives are approaching infinity. The likelihood for the predicted code for both observations is 100 per cent. For plotting purposes, we have set the maximum possible relative value to 100. This has been done to compress the plot and to improve the visualization of the plot. The observations where this apply are already flagged with a high priority. In addition to the observations mentioned above all observations where the likelihood of the predicted classification code is equal to 90 per cent or higher have been flagged with a high priority as we want to control these observations. The medium priority category includes the observations that should be checked, but where the model is not certain about the prediction. For these observations the likelihood for the current classification code can be significant. The specification of the medium group is given in table 6. The priority category low contains the remaining observations and for the most part include observations that we will not correct.

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 6: Definitions of priority groups** | | | |
| **Priority** | **Likelihood predicted classification code** | **Likelihood current classification code** | **Likelihood second highest classification code** |
| **High** | 0.9 <= | None | None |
| **Medium** | 0.6 <= < 0.9 | < 0.1 | < 0.3 |
| **Low** | Remaining | Remaining | Remaining |

Figure 4.2 show that we cannot blindly trust the relative , as a high value here does not necessarily mean that likelihood of the prediction is high. As we see there are lots of observations with a low priority that also has a high . By also demanding that the relative is high we include the observations with a high and medium priority.

1. **Conclusion**

Our work show that machine learning is an efficient method to predic erroneous classifications. This means that it is possible for a model to simulate manual work correcting classifications.

The challenge is that the predicted classification code deviate for a lot of the observations that are not audited (group 2). When using raw data we do not know the distinction between group 1 and group 2. One way to solve this is to use the likelihood estimator for the predicted classification code to separate the predictions that are likely to be correct from the unlikely. Results from group 1, where the predicted code coincide with the actual corrections, imply that this should be a safe presumption to take.

Further, to also cover the observations where the model is more uncertain about the predicted classification, but where the current classification code is unlikely - we apply the use of relative variables. The advantage of the relative values is that they convey more information than the likelihood does. By using this course of action, we can trim the results down to a feasible amount and simultaneously find identify the right observations. Either way we should save ourselves some tedious work and at the same time improve the figures.

1. **References**

Andreas C. Müller & Sarah Guido (2016), Introduction to Machine Learning with Python, O Reilly, CA

1. **Appendix**

|  |  |  |  |
| --- | --- | --- | --- |
| **Appendix 1: Selection criteria for chapters, 2016** | |  |  |
| **Chapter** | **Number of unique strings** | **Number of observations** | **Share classification corrected** |
| 14 | 85 | 661 | 0.132 |
| 25 | 2031 | 31812 | 0.104 |
| 05 | 1114 | 7967 | 0.086 |
| 01 | 1442 | 4495 | 0.048 |
| 89 | 2650 | 14445 | 0.046 |
| 02 | 1085 | 9610 | 0.042 |
| 24 | 490 | 24334 | 0.038 |
| 16 | 3004 | 24571 | 0.035 |
| 27 | 4690 | 95122 | 0.028 |
| 87 | 36447 | 943712 | 0.024 |
| 10 | 508 | 9488 | 0.023 |
| 20 | 6218 | 72327 | 0.021 |
| 22 | 9128 | 153328 | 0.02 |
| 19 | 14281 | 175667 | 0.018 |
| 03 | 5221 | 290105 | 0.016 |
| 12 | 1679 | 15469 | 0.015 |
| 86 | 664 | 9834 | 0.013 |
| 15 | 1706 | 22114 | 0.012 |
| 09 | 4257 | 48795 | 0.011 |
| 18 | 7858 | 91100 | 0.011 |
| 60 | 585 | 6524 | 0.01 |
| 21 | 13569 | 166509 | 0.01 |
|  |  |  |  |
| **Chapter** | **Number of unique strings** | **Number of observations** | **Share classification corrected** |
| 07 | 8343 | 138290 | 0.009 |
| 26 | 160 | 1638 | 0.008 |
| 11 | 826 | 7505 | 0.006 |
| 23 | 3042 | 67877 | 0.006 |
| 84 | 63338 | 2579582 | 0.005 |
| 49 | 5337 | 129426 | 0.004 |
| 63 | 10632 | 211671 | 0.004 |
| 47 | 422 | 27623 | 0.003 |
| 08 | 5998 | 101603 | 0.003 |
| 41 | 263 | 3210 | 0.002 |
| 13 | 449 | 1748 | 0.002 |
| 04 | 2252 | 33622 | 0.002 |
| 06 | 5262 | 103953 | 0.002 |
| 50 | 85 | 1031 | 0.001 |
| 78 | 93 | 864 | 0.001 |
| 75 | 226 | 4223 | 0.001 |
| 81 | 321 | 5553 | 0.001 |
| 55 | 481 | 8302 | 0.001 |
| 51 | 613 | 12797 | 0.001 |
| 54 | 759 | 12088 | 0.001 |
| 88 | 1018 | 27970 | 0.001 |
| 71 | 1429 | 49893 | 0.001 |
| 74 | 1879 | 40261 | 0.001 |
| 56 | 3109 | 43319 | 0.001 |
| 69 | 3847 | 106979 | 0.001 |
| 29 | 3981 | 35548 | 0.001 |
| 72 | 4918 | 74571 | 0.001 |
| 17 | 6096 | 85743 | 0.001 |
| 42 | 11630 | 248914 | 0.001 |
| 44 | 23915 | 313803 | 0.001 |
| 45 | 191 | 2551 | 0 |
| 53 | 269 | 6933 | 0 |
| 37 | 404 | 10126 | 0 |
| 66 | 458 | 9358 | 0 |
| 43 | 470 | 8416 | 0 |
| 79 | 510 | 8655 | 0 |
| 97 | 568 | 5058 | 0 |
| 46 | 675 | 13015 | 0 |
| 31 | 704 | 10112 | 0 |
| 52 | 962 | 16159 | 0 |
| 67 | 986 | 13793 | 0 |
| 92 | 1142 | 37152 | 0 |
| 58 | 1155 | 18052 | 0 |
| **Chapter** | **Number of unique strings** | **Number of observations** | **Share classification corrected** |
| 93 | 1270 | 8990 | 0 |
| 57 | 1540 | 48888 | 0 |
| 59 | 1652 | 28176 | 0 |
| 35 | 2484 | 34436 | 0 |
| 91 | 2704 | 45874 | 0 |
| 28 | 2748 | 44960 | 0 |
| 65 | 3380 | 57620 | 0 |
| 76 | 6068 | 163146 | 0 |
| 30 | 6459 | 128056 | 0 |
| 32 | 6843 | 142900 | 0 |
| 38 | 7355 | 155179 | 0 |
| 68 | 7617 | 110452 | 0 |
| 70 | 8088 | 162306 | 0 |
| 96 | 8577 | 132988 | 0 |
| 34 | 8700 | 177487 | 0 |
| 83 | 8725 | 180020 | 0 |
| 64 | 9422 | 295819 | 0 |
| 40 | 10304 | 221145 | 0 |
| 33 | 10928 | 256407 | 0 |
| 95 | 15256 | 351872 | 0 |
| 48 | 15846 | 311035 | 0 |
| 82 | 16344 | 312353 | 0 |
| 62 | 18932 | 881658 | 0 |
| 61 | 19387 | 788596 | 0 |
| 90 | 22043 | 742476 | 0 |
| 73 | 29909 | 627320 | 0 |
| 94 | 31085 | 1169362 | 0 |
| 39 | 39683 | 805516 | 0 |
| 85 | 60282 | 2888572 | 0 |
| 80 | 99 | 774 |  |
| 36 | 837 | 5767 |  |
| 99 | 3680 | 73313 |  |
| Selected chapters in grey  Source: Statistics Norway 2016 | |  |  |

1. Tolletatens system for utveksling av tolldeklarasjoner. [↑](#footnote-ref-1)
2. The packages used are numpy, pandas and the RandomForestClassifier included in the sklearn package. [↑](#footnote-ref-2)