Discussion paper

Evaluating and improving a text classifier for subpopulations: the case of cybercrime

Arnout van Delden,
Dick Windmeijer

July 2021

Content

Summary

There is considerable public interest in official figures about cybercrime. In 2019, Statistics Netherlands (CBS) developed a beta productto derive cyber-related crimes from texts in police reports using text mining. This beta product was used to estimate the proportion of cyber-related crimes for a set of standard crime types (SCM). Given the public interest in cybercrime figures, CBS wanted to evaluate what needs to be done to develop this beta product further into official statistics. In the current report, we treat three of the issues that have to be addressed for that purpose: missingness of text fields, bias in population estimates due to modelling errors and variability in model performance over SCM codes. Each of those issues are analysed and solutions are proposed. The variability in model performance over SCM codes appears to be the most difficult issue to tackle. The methodology that is proposed in the current discussion paper to evaluate the model performance over subpopulations - in our case the SCM codes - might also be useful in other situations where one is interested to move from a beta product to official statistics.

Keywords

Text mining, subpopulations, model evaluation, domains, model adaptation

1. Introduction

In the current study, we aim to analyse whether a preliminary version of a developed machine learning classifier by Statistics Netherlands (in Dutch: CBS) (CBS, 2019) - a so-called beta product - to predict cybercrime within a police registration for the Netherlands as a whole, can be developed further to produce official statistics on cybercrime by crime category. It concerns a text mining classifier that uses texts of records in a police registration of potential felonies (crime offences) as input. In the remainder of this paper, we will refer to these as 'crimes', although these potential crime have not been brought to court. The background behind this specific case study will be given in section 3. First analyses into this topic by Hooijschuur et al. (2020) showed that there we a number of issues to address (see below). Although our study focuses on cybercrime, we aim to use an approach to analyse the issues that may also be useful for other cases where one is interested to move from a beta product to official statistics.

The first issue concerns the coverage of the available textual data, subdivided into missing codes of a crime offences and missing texts for a given crime. First of all, the police registration will not contain all instances of cyber-related crimes since not everybody will report a cyber-related crime to the police. CBS publishes a survey of crime victims that includes cybercrime. In the past, differences have been studied between the survey and the police registration on frequencies of reported crimes (Reep, 2017; Reep, 2014). Part of the crimes are only reported in the survey and others only in the registration, but there is also a considerable overlap. These coverage differences have a large set of reasons. So far, no estimator has been developed that tackles the coverage issues in those two sources. In the present study, we limit ourselves to the register data, and only pay attention to the issue of absence of texts for a part of the reported crimes. This latter issue is a potential source of bias in the estimated proportion of cyber-related crimes, even when one defines the set of records in the police registration as the target population.

Coverage issues also play a role in other examples where one is interested to publish population parameters based on text mining. An example is the estimation of sentiment in a population (O'Connor et al, 2010) based on text mining of social media data. Available texts of social media data may not always be representative for the target population of interest. Another example is a text classifier that aims to predict whether enterprises are innovative or not, using website data (Daas and van der Doef, 2020). Not all enterprises have a website, and maybe enterprises with a website are more innovative than enterprises that do not have a website.

The second issue, mentioned in Hooijschuur et al (2019) refers to the computation of the output that one is interested in. The usual procedure is that one first predicts a label (cybercrime yes/no) for each individual records and then one sums all the relevant records. That is also referred to as the 'classify-and-count' estimator. However that may not be the most accurate estimator, because the predicted labels are prone to errors (false positives and false negatives) and those errors might not cancel out. The effect of those errors, when they do not cancel out, is that the population estimates are biased. The estimation of the accuracy of population estimates as affected by errors made by machine learning classifier can be found for instance in Meertens et al. (2019) and in Scholtus and Van Delden (2020). When the classification task concerns a large, known, target population then the classification errors may result in biased estimates. Hooijschuur et al. (2019) found that the bias varied for different variations of the model. We are interested to understand this model sensitivity.

The third issue we address is how we can evaluate the model performance for subpopulations. Most official statistics are partitioned into geographical regions or subpopulations. The question then rises what the quality of the machine learning model is for the different geographical regions or subpopulations. In order to answer this question one needs to have a measure to evaluate the model performance for subpopulations. A machine learning model is usually evaluated on a test set by using performance scores such as the recall, precision and the F_1 score. When we are interested in the performance of subpopulations we could of course also compute those scores, but the number of test units per subpopulation may be much smaller than for the original set, affecting their accuracy. In the current paper we therefore propose an alternative measure to test whether the model performance is affected by the subpopulations or not.

Various papers, e.g. Lewis et al., (2019), Talby (2018) and Ziegler and Czyż (2019) mention that when a model has been trained for a population, it may not necessarily perform well for classes underlying that population. A fourth issue mentioned in Hooijschuur et al (2019) that we explore further is to what extent the model performance per subpopulation can be improved by retraining.

A final point that we address is interpretability: how the model takes its decisions. We aim to understand why the model works better for one subpopulation compared to the other. We aim to use this to find ways to further improve the model in future.

The remainder of this paper is organised as follows. The methodology that we use concerning the different issues is given in section 2. In Section 3 we introduce the case study. Subsequently, we give the results on missingness (section 4), on model sensitivity for the bias (section 5), on retraining per cybercrime (section 6), on retraining per cyber type (section 7) and on evaluating model decisions (8). Finally, section 9 discusses what are the next steps we would like to do to reliably estimate cybercrime related aspects for subpopulations. Furthermore, we discuss the general approach we used to analyse and retrain models for subpopulations.

2. Methodology

2.1 General notation

We will use the following notation throughout this paper. Let $i = 1, \ldots, N$ stand for the units in the target population of size N. In the case study on cybercrime one unit i stands for one record - with a unique identification - in the police registration, see also section 3.

Furthermore, let $s_i = h$ denote that unit i has stratum h, with $h = 1, \ldots, H$. The strata stand for the subpopulation that one is interested to use for the publication. Let y_{hi} be a binary target variable for unit i within stratum h, which is 1 when the target class (e.g. cybercrime) is present and 0 otherwise.

We are interested in the proportion of units per stratum for which the binary target variable is 1. The total number of units in stratum h for which the target variable equals 1 is denoted by Y_h and its average by $\bar{Y}_h$ which equals the proportion that we are interested in. This proportion is given by:

$$\bar{Y}_h = \frac{1}{N_h} \sum_{i=1}^{N_h} y_{hi} = \frac{Y_h}{N_h}.$$

The values for variable y are determined manually in a sample of size n, the labelled set. The unit i in the sample has weight w_{hi}. The exact form of the weights depends on the evaluation that is done, and will be treated in section 3.5. With the notation weight w_{hi} we just express that the weights may vary within stratum h for unit i.

The Hajek estimator of the population proportion in stratum h is given by

$$\hat{\bar{Y}}_h^{HJ} = \frac{\sum_{i=1}^{n_h} w_{hi} y_{hi}}{\sum_{i=1}^{n_h} w_{hi}} \tag{1}$$

Note that for a stratified random sampling $w_{hi} = N_h/n_h$ $\hat{\bar{Y}}_h^{HJ}$ can be written as $N^{-1} \sum_{i=1}^{n_h} w_{hi} y_{hi}$, which equals the Horvitz-Thompson estimator. In the case study on cybercrime however, one of the samples that we use is a selective sample. We will try to correct for this selectivity by using estimated weights that may differ for class 1 and 0. These weights are computed retrospectively, by making use of the labelled data, this is explained in section 3.5. For the situation of this selective sample, the Hajek estimator cannot be simplified to the Horvitz-Thompson estimator.

In practice, the values for variable y that are not included in the sample will be estimated by means of a machine learning model that will be trained on a sample.

The vector of k features of the model are denoted by $\boldsymbol{u}$, with $\boldsymbol{u} = (u_1, u_2, \dots, u_k)^T$ (all vectors in this paper are column-vectors).

2.2 Analysis of missingness

For some records in the police registration no texts are available, see section 4.1, implying that the target variable cannot be estimated by text mining. In order to quantify whether this missingness is likely to be selective with respect to the target variable, we first introduce some additional notation. Let z_i be an indicator variable that denotes whether the data to derive the features for the machine learning model are present ($z_i = 1$) or absent ($z_i = 0$) for unit i. When the values of variable z are unrelated to y, we refer to this missingness as *missing completely at random* (MCAR), following the terminology of Little and Rubin (2002). The MCAR assumption may not hold in practice. A less restrictive assumption is that the values of z are independent of y given a set of p background variables $\boldsymbol{x}$. The vector of background variables $\boldsymbol{x}_i$ is given by $\boldsymbol{x}_i = \left(x_{i1}, x_{i2}, \dots, x_{ip}\right)^T$. This situation is referred to as missing at random (MAR). When neither MCAR nor MAR holds then we state that the data are missing not at random (MNAR).

Generally speaking, there are three options to deal with the missingness in the data. The first option is not to publish outcomes of strata for which the proportion of missingness is above a certain threshold. Second, for strata where the proportion is missingness is limited one could decide to publish the outcomes and report the proportion of missingness in those strata. The third option is to try to correct for the selectivity in the missingness. Such a correction can be done in different ways; in the current paper we limit ourselves to imputing missing values. Of course, also combinations of these three options is possible. For instance, one could impute missing data, but only for those strata where the proportion of missingness is below a certain threshold.

We analysed the extent and nature of the missingness as follows. First we computed the percentage of missingness per stratum h, in order to judge whether the missingness is concentrated in certain strata. Next, we analysed whether the missingness could be explained by background variables in the data set. We applied a logistic regression of variable z on background variables $\boldsymbol{x}$ in the full data set:

$$\text{logit}\left\{\hat{P}(z_i = 1)\right\} = (\boldsymbol{x}_i)^T \widehat{\boldsymbol{\beta}}_{zx}, \text{with } i = 1, \dots, N, \quad (2)$$

where $\boldsymbol{\beta}_{zx} = \left(\beta_{zx_1}, \beta_{zx_2}, \dots, \beta_{zx_p}\right)^T$ is the vector of regression coefficients of variable z on x (including a constant). The goodness of fit of the logistic regression was analysed by the Akaike Information Criterion (See Appendix 1). We selected the best fitting background variables by using a forward selection procedure. We use forward and not backward selection because we wanted to have the model as parsimonious as possible. We could also have used a stepwise selection procedure that is a modification of the forward selection. In stepwise selection one tests, after adding the variables for each of the added variables whether they are significant or not. We used the simpler procedure because we had to program the whole procedure ourselves in Python. Nevertheless, we did test the significance of the parameters of our final model and those results are reported, see sections 4.2 and 4.3.

Further, we applied a logistic regression of the target variable y on the background variables $\boldsymbol{x}$ within the labelled set:

$$\operatorname{logit}\left\{\hat{P}(y_i = 1)\right\} = (\boldsymbol{x}_i)^T \hat{\boldsymbol{\beta}}_{yx}, \text{with } i = 1, \dots, n \tag{3}$$

and $\boldsymbol{\beta}_{yx} = \left(\beta_{yx_1}, \beta_{yx_2}, \dots, \beta_{yx_p}\right)^T$ is the vector of regression coefficients of variable y on x (including a constant). Within this analysis, we account for the sample weights w_{hi}, so we aim to predict the relation at population level.

When both the missingness (z_i) as well as the target variable (y_i) are well explained by background variables then it might be feasible to impute the missing records.

These analyses were performed in R using the survey package (Lumley, 2019).

2.3 Accuracy of level estimates affected by classification errors

We are interested to evaluate the accuracy, i.e. the bias and the variance, of the estimator for the population proportion $\bar{Y}_h$. Recall that n_h stands for the number of manually labelled records with value y_{hi}. Let $\tilde{y}_{hi}$ denote the predicted label of the target variable by the machine learning model and let $\bar{\tilde{y}}_h$ stand for the predicted proportion of the target variable in stratum h. The estimated proportion of the target variable is then given by:

$$\hat{\bar{Y}}_h = \frac{1}{N_h}\left(\sum_{i=1}^{n_h} y_{hi} + \sum_{i=n_h+1}^{N_h} \tilde{y}_{hi}\right) = \frac{n_h \bar{y}_h}{N_h} + \frac{(N_h - n_h)\bar{\tilde{y}}_h}{N_h}, \tag{4}$$

with $\bar{y}_h = \frac{1}{n_h}\sum_{i=1}^{n_h} y_{hi}$ and $\bar{\tilde{y}}_h = \frac{1}{N_h - n_h}\sum_{i=n_h+1}^{N_h} \tilde{y}_{hi}$

In our situation, $n_h \ll N_h$, which implies that $\hat{\bar{Y}}_h \approx \bar{\tilde{y}}_h$. In the remainder of this paper when we compute the accuracy of the estimates, we ignore the small labelled part where y_{hi} is known.

When using a supervised machine learning model the errors made by the machine learning model in stratum h can be estimated by predicting the target variable on an independent test set with known inclusion probabilities. The frequency counts of the labelled versus the predicted values in stratum h are presented in a so-called confusion matrix, see Table 1. We denote these frequencies by f_{jkh}, for true label $y_i = j$ and predicted label $\tilde{y}_i = k$ within stratum $s_i = h$. To be more precise, the machine learning model produces a confidence that the predicted label is 1. When this confidence is above a threshold of 0.5, the label is set to 1 and 0 otherwise. One can vary the threshold above which the label is set to 1 in order to influence the model performance. This is further discussed in section 5.

Table 1. Example of a confusion matrix of the sample units in stratum h.

	Predicted label	
True label	1	0
1	f_{11h}: true positives	f_{10h}: False negatives
0	f_{01h} : False positives	f_{00h}: True negatives

Let superscript '*' denote that it involves the test set, and n_h^* stands for the size of the test set. It then follows that $\sum_{j,k} f_{jkh}^* = n_h^*$. In the case study, we will use all units of the labelled set as test set, which is made possible because we used a cross-validation procedure with hyperparameter settings that have been fixed beforehand; this will be explained in sections 3.4 and 3.6. That means that the labelled set and the test set concerns the same set of units.

The frequencies were weighted to obtain frequency estimates at population level. Let $I_{hi}(y_i = j, \hat{y}_i = k\ )$ be an indicator variable that equals 1 when the true label $y_i = j$ and the observed label $\hat{y}_i = k$ and is 0 otherwise. The population frequencies denoted by $\mathcal{F}_{jkh}^*$, are then estimated by $\hat{\mathcal{F}}_{jkh}^* = \sum_{i=1}^{n_h^*} I_{hi}(y_i = j, \hat{y}_i = k\ )\, w_{hi}$. Further denote $\hat{\mathcal{F}}_{j\bullet h}^* = \sum_k \hat{\mathcal{F}}_{jkh}^*$, where a '•' denotes that one sums over a class.

The frequencies given in Table 1 can also be transformed in terms of a matrix $\mathbf{P}_h$ with probabilities of the form:

$$\mathbf{P}_h = \begin{pmatrix} p_{11h} & 1 - p_{11h} \\ 1 - p_{00h} & p_{00h} \end{pmatrix},$$

where the rows represent the labels and the columns represent the predictions. These probabilities can be estimated by $\hat{p}_{jkh}^* = \hat{\mathcal{F}}_{jkh}^* / \hat{\mathcal{F}}_{j\bullet h}^*$. The symbol p_{11h} stands for the recall of label 1 in stratum h, so the fraction of cases predicted as 1 given that the true label is 1. Other terms for the recall of label 1 are 'true positive rate' and 'sensitivity'. Similarly, p_{00h} stands for the recall of label 0 in stratum h. Other terms for the recall of label 0 are 'true negative rate' and 'specificity'.

The bias and variance of $\hat{\hat{Y}}_h$ are now given by (see Scholtus and van Delden, 2020):

$$B\left(\hat{\hat{Y}}_h\right) = (p_{11h} - 1)\bar{Y}_h + (1 - p_{00h})(1 - \bar{Y}_h),$$

and

$$V\left(\hat{\hat{Y}}_h\right) = \frac{p_{11h}(p_{11h} - 1)\bar{Y}_h + p_{00h}(1 - p_{00h})(1 - \bar{Y}_h)}{N_h}.$$

where $\bar{Y}_h$ stand for the true proportion of the target variable in the population, which is unknown.

One possibility to estimate the bias and variance is to use the estimated proportion of cyber in the population according to the machine learning model, $\hat{\bar{Y}}_h$ as defined in (4), as plug-in estimator for $\bar{Y}_h$:

$$B^{POP}\left(\hat{\bar{Y}}_h\right) = (\hat{p}^*_{11h} - 1)\hat{\bar{Y}}_h + (1 - \hat{p}^*_{00h})(1 - \hat{\bar{Y}}_h), \tag{5}$$

where superscript 'POP' means that the predicted population mean is used as a plug-in estimator for $\bar{Y}_h$. Van Delden et al. (2016) showed that this estimate works reasonably well for the variance, but the bias estimate is biased itself, because $\hat{\bar{Y}}_h$ is a biased estimator for $\bar{Y}_h$.

As an alternative, we used the weighted fraction of cyber per stratum h of the manually labelled data, $\hat{\bar{Y}}_h^{HJ}$, as defined in equation (1). Since the labelled sample data that we used for $\hat{\bar{Y}}_h^{HJ}$ were obtained from a stratified random sample with strata h, see section 3.3–3.5, $\hat{\bar{Y}}_h^{HJ}$is an unbiased estimator for $\bar{Y}_h$. We therefore used $\hat{\bar{Y}}_h^{HJ}$as plug-in estimator for $\bar{Y}_h$ for the bias and variance estimates, denoted by $\hat{B}^*\left(\hat{\bar{Y}}_h\right)$ and $\hat{V}^*\left(\hat{\bar{Y}}_h\right)$, where the asterix denotes that all estimates are obtained from the test set (recall that all units in the labelled set are used for the test set by using a cross validation procedure).

We now obtain as the bias estimate:

$$\begin{aligned}\hat{B}^*\left(\hat{\bar{Y}}_h\right) &= (\hat{p}^*_{11h} - 1)\hat{\bar{Y}}_h^{HJ} + (1 - \hat{p}^*_{00h})\left(1 - \hat{\bar{Y}}_h^{HJ}\right) \\ &= \left[\hat{p}^*_{11h}\hat{\bar{Y}}_h^{HJ} + (1 - \hat{p}^*_{00h})\left(1 - \hat{\bar{Y}}_h^{HJ}\right)\right] - \hat{\bar{Y}}_h^{HJ} \\ &= \hat{\bar{\bar{Y}}}_h^* - \hat{\bar{Y}}_h^{HJ} \text{ ,with}\end{aligned} \tag{6}$$

$$\hat{\bar{\bar{Y}}}_h^* = \hat{p}^*_{11h}\hat{\bar{Y}}_h^{HJ} + (1 - \hat{p}^*_{00h})\left(1 - \hat{\bar{Y}}_h^{HJ}\right) \tag{7}$$

where superscript '*' indicates that all parameters are estimated from the same weighted confusion matrix.

With respect to equation (7): $\hat{p}^*_{11h}$ and $1 - \hat{p}^*_{00h}$ are the estimated probabilities of the matrix $\mathbf{P}_h$ of the labelled set and $\hat{\bar{Y}}_h^{HJ}$ is the true proportion of cyber according to the labelled set. From this it follows that the predicted label 1 according to the confusion matrix (Table 1) is given by $\hat{p}^*_{11h}\hat{\bar{Y}}_h^{HJ}$ (upper left cell: predicted as cyber and in reality it is cyber) plus $(1 - \hat{p}^*_{00h})\left(1 - \hat{\bar{Y}}_h^{HJ}\right)$ (lower left cell: predicted as cyber and in reality it is no cyber).

Alternatively, we also could have also estimated the bias by $\hat{B}\left(\hat{\hat{Y}}_h\right) = \hat{\hat{Y}}_h \ - \hat{\hat{Y}}_h^{HJ}$. In practice both the estimates $\hat{B}^*\left(\hat{\hat{Y}}_h\right)$and $\hat{B}\left(\hat{\hat{Y}}_h\right)$are expected to be reasonably close to each other, but the estimator $\hat{B}^*\left(\hat{\hat{Y}}_h\right)$is easier to obtain since only the data of the test set are needed.

Note that these bias and variance formulas assume that the probabilities in matrix $\mathbf{P}_h$ are fixed. The uncertainties in estimating these probabilities are not accounted for because they do not express the bias and variance of the estimator $\hat{\hat{Y}}_h$ but they express our uncertainty about that bias and variance.

2.4 Evaluating model accuracy

We evaluate the model accuracy by computing the following measures within each stratum h:

$$\text{Recall}_h(y=1) = \frac{\hat{\mathcal{F}}_{11h}}{\hat{\mathcal{F}}_{1\bullet h}};\ \text{Recall}_h(y=0) = \frac{\hat{\mathcal{F}}_{00h}}{\hat{\mathcal{F}}_{0\bullet h}},$$

$$\text{Precision}_h(y=1) = \frac{\hat{\mathcal{F}}_{11h}}{\hat{\mathcal{F}}_{\bullet 1h}};\ \text{Precision}_h\ (y=0) = \frac{\hat{\mathcal{F}}_{00h}}{\hat{\mathcal{F}}_{\bullet 0h}},$$

$$\text{F}_{1h}(y=1) = \frac{2}{\text{Recall}(y=1)^{-1} + \text{Precision}(y=1)^{-1}},$$

and $\text{F}_{1h}(y=0)$ is defined similary. A '•' denotes that one sums over a class: $\hat{\mathcal{F}}_{1\bullet h} = \hat{\mathcal{F}}_{11h} + \hat{\mathcal{F}}_{10h}$ and so on. We computed those scores for each of the strata h and combined them by using a micro- and a macro average value. The macro average of a score S_h per stratum h is given by $S(macro) = \frac{1}{H}\sum_{h=1}^{H} S_h$. The micro-average is obtained by ignoring the stratum indicator and simply pooling the results at micro level over all strata.

In practice, to summerise our results, we will limit the $\text{F}_{1\text{h}}$ scores to class 1 for which we will use the shorthand notation F_{1h}, $[\text{F}_{1\text{h}} = \ \text{F}_{1h}\ (y=1)]$. The F_{1h} performance measure combines three cells of the confusion matrix into a single measure. We are interested to compare the model performance over different strata. Burger and Meertens (2020) show and discuss that the performance measures cannot directly be compared over different strata, because the outcomes of the performance measures depend on the true Y_h which may vary for the different strata. Burger and Meertens (2020) propose to compute min-max performance measures, which are measures that are normalised with respect to their minimum and maximum value. The minimum value is determined by "guessing", and they propose different possible "guessing" options. A natural candidate would be the proportion of cyber found in the labelled set, but some of our results are for the Netherlands as a whole and others are per stratum h. One then has to choose between those two options. Since the cyber proportions in the labelled set vary considerably per stratum h it seems most appropriate to account for these differences, although the estimates of those proportions are based on a limited number of units (the labelled set).

In the current paper we account for a fair comparison in model performance between strata in the following way. Let $p_{jk} = P(\hat{y}_i = k|\, y_i = j)$ be the probability that the true class j is predicted as class k. Now, additionally we have strata h and the proportions $p_{jh} = P(y_i = j|s_i = h)$ vary per stratum h. We are interested in having a machine learning classifier whose performance is not affected by the subpopulation that we are interested in. That is, the prediction probabilities only depend on the true codes and not also on the strata, that is $p_{jkh} = P(\hat{y}_i = k|y_i = j, s_i = h) = P_{jk}(\hat{y}_i = k|y_i = j)$. This means that we want the recall of class 0 and 1 to be constant for the strata h .

So far, we have only considered to directly predict a value 0 or 1. Instead we can also predict the confidence of the machine learning model in label 0 or 1, and incorporate that value in an evaluation measure. While only probabilistic machine learning models (such as Naïve Bayes and logistic regression) directly estimate a posterior probability most machine learning models give a confidence measure. A confidence measure can be calibrated to approximate a posterior probability.

Let $\hat{P}(\hat{y}_i = j)$ denote the posterior probability to predict class label j, with $j = \{0,1\}$. Furthermore, let $I(y_i = j)$ be an indicator variable which is 1 when the class label of unit I equals j and is 0 otherwise. The $\log_2$ cross-entropy expresses the probability mass to predict the correct binary label and is given by:

$$\hat{H}(I,P) = -\sum_{i=1}^{n_h^*}\sum_{j=0}^{1} I(y_i = j)\log_2\{\hat{P}(y_i = j)\} \tag{8}$$

When there is perfect agreement between $I(y_i = j)$ and $\hat{P}(y_i = j)$ the cross-entropy is 0. In fact, the cross-entropy is the Kullback-Leibner distance between the distribution of $I(y_i = j)$ and $\hat{P}(y_i = j)$. This distance is defined as

$$\begin{aligned}\hat{\mathcal{K}}(I,P) &= \sum_{i=0}^{n_h^*}\sum_{j=1}^{1} I(y_i = j)\log_2\left\{\frac{I(y_i = j)}{\hat{P}(y_i = j)}\right\}\\ &= \sum_{i=0}^{n_h^*}\sum_{j=1}^{1} I(y_i = j)\log_2\{I(y_i = j)\} - \sum_{i=0}^{n_h^*}\sum_{j=1}^{1} I(y_i = j)\log_2\{\hat{P}(y_i = j)\}\\ &= -\sum_{i=0}^{n_h^*}\sum_{j=1}^{1} I(y_i = j)\log_2\{\hat{P}(y_i = j)\} = \hat{H}(I,P),\end{aligned}$$

where in the second line we defined $0\log_2(0) = 0$. So, a distance of 0 implies that those two distributions are identical. We consider the binary classifier to be completely uninformative when all predicted probabilities are 1/2, implying that $\hat{H}(I,P) = -n_h^*\log_2\left(\frac{1}{2}\right) = n_h^*\log_2(2) = n_h^*$. Analogous to the relative entropy (Dias and Vermunt, 2008), we introduce the relative cross-entropy defined as:

$$\widehat{RH}(I,P) = 1 - \frac{\widehat{H}(I,P\,)}{n_h^*}. \tag{9}$$

which equals 1 in case of perfect agreement and it equals 0 when the classifier is completely uninformative. Negative values of $\widehat{RH}(I,P)$ indicate the performance is poorer than when just taking 1/2, which is a very unlikely situation. The relative entropy in (9) is referred to as entropy R^2 in Boeschoten et al. (2017). The relative entropy is very similar to accuracy: accuracy measures the fraction of correctly predicted cases, while the relative entropy measures the (overall) probability to predict the correct class.

2.5 Testing whether model accuracy is affected by stratum

Furthermore, we were interested to test whether the association between the true and the predicted labels differed among the strata h. We did so by applying a log-linear model on the counts in the confusion matrix. We start by defining this model for the unweighted frequency counts of the units in the test set. The log-linear model for the frequency counts f_{jkh} for true label (L) $y_i = j$, predicted category (P) $\hat{y}_i = k$ and stratum (C) $s_i = h$ is:

$$\text{Model 1}: ln f_{jkh} = u + u_j^L + u_k^P + u_h^C + u_{jk}^{LP} + u_{jh}^{LC} + u_{kh}^{PC}. \tag{10}$$

This model contains - apart from a constant term (u) - the main effects for each variable (u_j^L, u_k^P, u_h^C) and the interactions between pairs of variables (u_{jk}^{LP}, u_{jh}^{LC}, u_{kh}^{PC}). If this model describes the data well, then the model performance, the association between label and predicted values does not depend on the stratum (the subpopulations).

However, the sampling frequencies are not representative for the population, and we are interested in this relationship at population level. We have defined the frequencies at population level before as $\hat{\mathcal{F}}_{jkh}^* = \sum_{i=1}^{n_h^*} I_{hi}(y_i = j, \hat{y}_i = k\,)\, w_{hi}$. Let $w_{hi} = \omega_{jkh}$, where ω_{jkh} denotes the weight for $y_i = j$, $\hat{y}_i = k$ and $s_i = h$. Then $\hat{\mathcal{F}}_{jkh} = f_{jkh}\omega_{jkh}$. Using $\ln f_{jkh} = ln\hat{\mathcal{F}}_{jkh} - \ln \omega_{jkh}$ we obtain model 2:

$$\text{Model 2}: ln f_{jkh} = -\ln \omega_{jkh} + u + u_j^L + u_k^P + u_h^C + u_{jk}^{LP} + u_{jh}^{LC} + u_{kh}^{PC}. \tag{11}$$

The term '$-\ln \omega_{jkh}$' is also referred to as the offset of a log-linear model.

Inspection of the results showed that it is not only interesting to test the overall association between predicted and true label, but also look into the effect of a stratum on prediction errors that are made given the true label $j = 0$ or the true label $j = 1$. In order to do so we used model 3:

$$\text{Model 3}: ln f_{kh|j} = -\ln \omega_{jkh} + u + u_k^P + u_h^C \tag{12}$$

In model 3 all terms with label (L) are dropped, so u_{jk}^{LP} and u_{jh}^{LC}, and also u_{kh}^{PC} is dropped because for a given true label, the model $= -\ln \omega_{jkh} + u + u_k^P + u_h^C +$

u_{kh}^{PC} is the fully saturated model. If model 3 describes the data well, then the error made by the model does not depend on the stratum.

We tested the fit of the log-linear model by its deviance (see Appendix). A significant deviance in case of model 2 means that the model without the three-way interaction deviated significantly from the saturated model, implying that stratum h significantly affects the association between the labels and their predicted values.

Note that in equations (2) and (3) we were interested to find a parsimonious model to describe the missingness and cybercrime as a function of background variables, where we used a penalty for the number of parameters according to AIC criterion. In the log-linear analysis however, we are explicitly interested to compare models 2 [equation (11)] and 3 [equation (12)] with their saturated equivalents in order to test whether the difference is significant. If not, this means that the model performance is not affected by crime type.

A significant effect of stratum h also means that the error matrix $\mathbf{P}_h$ (see section 2.3), needed to compute the bias and variance of $\hat{\hat{Y}}_h$ differs among the strata h. If it holds that $\mathbf{P}_h = \mathbf{P}$ for all strata, then the corresponding probabilities p_{00} and p_{11} can be used for the bias estimates $\hat{B}^*\left(\hat{\hat{Y}}_h\right)$, given in section 2.4.

When the cell frequencies of the confusion matrix are significantly affected by stratum, then it is also possible to test *which* of those strata are responsible for that effect. To describe this test, we first write down an expression for the deviance in case of frequency counts. The deviance, denoted by D, between the observed frequency counts f_{jkh} (equivalent to a saturated model) and the expected frequency counts based on the model, denoted by $\hat{f}_{jkh}$, over all combinations $(j, k, h)$is given by:

$$D = 2\sum\nolimits_{jkh} f_{jkh} \ln\left(\frac{f_{jkh}}{\hat{f}_{jkh}}\right)$$

It can directly be seen that the total deviance is the sum of the contributions per stratum. The deviance for a given stratum h can be tested against 1 degrees of freedom. We considered strata for which the p-value was smaller than a threshold $\alpha = 0.05$, to have a significantly different confusion matrix. In the application of cybercrime there are about 20 strata, so when using $\alpha = 0.05$ one expects that at least one stratum is significantly affected just by chance. We decided not to correct the threshold, first of all because we only test individual strata when the overall test is significant and second because we are interested to find any stratum for which the model predictions may be deviating. When we would have used an adjusted threshold, a much smaller α would have been obtained with the risk of overlooking differences. This can be seen as follows. A simple adjustment formula is $\alpha = 1 - \left(1 - \alpha_{adj}\right)^m$ where m is the number of tests performed, α is the overall probability of rejecting and α_{adj} is the threshold per stratum. The idea behind the formula is that $\left(1 - \alpha_{adj}\right)^m$ is the probability of not rejecting m independent tests so $1 - \left(1 - \alpha_{adj}\right)^m$ is the probability of rejecting m

independent tests. This results in $\alpha_{adj} = 1 - \sqrt[m]{(1-\alpha)} \approx \alpha/m$ for small α. α/m is known as the Bonferoni correction. Setting $\alpha = 0.05$ yields $\alpha_{adj} \approx 0.0025$.

2.6 Retraining strategies

We analysed whether retraining the model with the binary variable on different groupings of the strata h improved the model performance. We used two groups of retraining strategies: a) retraining with respect to the subpopulations we intend to publish: the strata h, and b) retraining with respect to different types of cybercrime.

Ad a). We compared the following training strategies with respect to the strata h:

- BASIC (BAS): model trained on full labelled set;
- BINARY (BIN): model trained on two groups: one group consist of all strata whose deviance-statistics are significant and a group with all other strata;
- DEVIATING (DEV): each stratum whose deviance-statistics was significant is trained separately, as well as a rest group consisting of all non-significant strata
- ALL: each stratum is trained separately.

The settings BINARY and DEVIATING were retrained for two types of deviance tests: Model 2 and model 3 of section 2.5.

Ad b) Furthermore, we also tested the effect of combining all strata h, like in the model BASIC, but now we trained on different categories of variable y. So rather than training and predicting a binary variable y, we used three categories:

- pure cybercrime, such as ransomware and hacking
- cyber-related crimes such as identity fraud
- no cyber-related crime.

We will further explain these categories in section 7.1.

2.7 Analysis of model decisions

In addition to the analyses described above we wish to understand how the model has taken its decision in individual records, and to what extent this varies over the strata. Hopefully that helps us to know how the model performance can be improved.

In general, a supervised machine learning classifier is a mapping of some set of features, denoted as $\boldsymbol{u}_i$, to the target variable y_i: $y_i = f(\boldsymbol{u})$. Specifically, for the model that we used in our case study, a support vector machine, the output variable is linearly related to the target variable:

$$y_i = \operatorname{sign}\left(\boldsymbol{u}_i^{\boldsymbol{T}} \hat{\boldsymbol{\gamma}}\right)$$

so the contribution of each feature to the output value can easily be derived given a fitted model. This can be done by ordering the estimated coefficients $\hat{\gamma}$ of the features: the most negative coefficients refer to the features that most contribute to predicting non-cyber and the most positive coefficients refer to the features that contribute most to predicting cybercrime.

3. Background to the case study

3.1 Context

There is great interest in society to have accurate data on computer crimes. Alkaabi et al. (2011) divide these computer crimes into two main types. The first type concerns crimes where the computer is the target of a criminal activity. Examples are hacking, ransomware and DDos attacks. This is also referred to as 'pure cybercrime' by Tollenaar et al. (2018). The second type concerns crimes where the computer is the tool to commit a crime. Examples are online identity fraud, purchase fraud, defamation and stalking. This is also referred to as 'digital crimes' by Tollenaar et al. (2018). Both types were further classified in Alkaabi et al. (2011), giving a total classification of seven main types.

CBS already publishes yearly data on victims of cybercrime, based on survey data. Furthermore, the police publishes monthly data on pure cybercrime by region (data.politie.nl). These data are based on a national registration (Dutch: Basisvoorziening Handhaving, BVH) that holds incidents and potential crimes reported by individuals and found by police work. The BVH data are collected to support police work and are not originally intended to be used for official statistics. In this registration, each incident (record) obtains one code that best describes the offence. In practice, incidents that are reported often concern multiple issues and can also contain cyber-related aspects. The publications by the police only concern incidents with the (main) code of pure cybercrime.

Recently, CBS as well as the Research and Documentation Centre (in Dutch: WODC) explored, with help of the police, whether it is possible to classify cyber-related crimes by means of text mining algorithms, using texts in the BVH registration. Those cyber-related aspects concern both main types of cybercrime: pure cyber and digital crimes. Furthermore, it concerns any offence that also has a cyber-related aspect, it does not necessarily have to be the main offence that is reported. This cyber-related aspect should be a potential criminal offence.

CBS decided to start with predicting a single binary variable for cyber-related crimes (yes, no) and used a support vector machine (SVM) algorithm in combination with a bag of words model. The CBS study is limited to entries within the BVH that concerns potential felonies (crime offences). This involves on a yearly basis approximately 1 million records out of a total of about 4 million records within the BVH. As a test set, CBS drew a random sample from this subset. This resulted in promising results where 96 per cent of predicted cybercrime was indeed true cybercrime (precision), whereas 85 per cent of true cybercrime as indeed found (recall). The results can be found at the CBS innovation website, see http://www.cbs.nl/en-gb/our-services/innovation.

The use of other feature types than just a bag of words model to predict cybercrime has been tested by Tollenaar et al. (2018). They compared bag of words with the use of meta-textual information, syntactical information and semantical information and surprisingly found that results by the bag of words approach were not improved by using the other feature types. Therefore we limit the features in the current paper to the use of bag of words features.

Rather than publishing a single national proportion of cybercrime, CBS is interested to divide cyber-related crimes into subpopulations such as characteristics of the victim and the potential perpetrators. The current report tests whether cyber-related crimes can be predicted with sufficient quality when it is partitioned by the standard classification crime types (Dutch: standaard classificatie misdrijven, SCM). Although crime type is not the classification variable that CBS is most interested in, we wanted to look into this SCM classification since the published beta product (CBS, 2019) partitioned the results by crime types while Hooijschuur (2020) found that the model performance varied with SCM code. If we can find a way to improve the model results, that might also be useful when cybercrime is partitioned onto other classification variables. Rather than focusing on the quality of predictions at the individual level, we are interested to test whether the estimated proportion of cybercrime per crime type will be accurate enough.

3.2 The police registration data

We used the BVH registration data of 2016. The full data set consists of approximately 3.9 million records with an average of 4.2 entries per BVH registration (see Tollenaar et al., 2018, p. 14). In total CBS received a subset which concerns potential crimes. That data set consisted of roughly 990 000 records of which roughly 13 000 with an unknown crime type. The analysis on the current report used a number of selections:

– records with an unknown crime type were excluded;
– records were both the declaration and the clarification field were empty were excluded;
– records that could not uniquely be linked to the police annotations were excluded
– duplicate records were excluded.

Selections of the BVH data are also used by CBS to make (yearly) output on crime frequencies by SCM codes, resulting in official CBS figures. However, the selections made by CBS for those official figures differ on some points from the selections made in the current report, therefore differences may occur. The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

The data contains two fields with a unique identification number: 'original registration number' and 'identification field'. A unique 'original registration number' may have multiple records with a different 'identification field'. These records are used to register different crimes that may occur for the same incident.

Furthermore, the data contains three text fields: a declaration field, a clarification field and an observation field. The declaration field concerns mostly declarations by a victim or declarant who reports a crime to the police. The clarification field is filled in by the police. A part of records in the data are the result of effort by the police to find crimes rather that declarations by victims. The 'detective work' by the police usually results in records with a filled clarification field combined with an empty declaration field. The observation field is hardly filled and does not contain text that can be used to derive whether the incident concerns a cyber-related task or not. In the remainder of the paper we only use the declaration and the clarification fields. Some basic information on the BVH data can be found in Table 18 in Appendix 2: the description of the SCM code, number of records per SCM code and completeness of the declaration and the clarification text fields.

Apart from the identification variables and the text fields, the following variables can be found in the data:
– Police corps: which police corps registered the incident;

Furthermore, the following binary variables (yes/no) can be found in the data:
– Crime: whether it concerns a potential crime or not;
– Declaration: whether it has been declared by a victim or not;
– Main: whether it concerns the main issue of the incident or not;
– Series: whether it concerns a series of issues for the same incident or not;
– Most severe: whether it concerns the most serious issue within the incident or not;
– Attempt, whether it concerns an attempt to commit a crime or not;
– Clarified: whether the incident has been clarified or not. Examples of 'yes' are when a suspect is registered or when, missing goods are found;
– Outside Region: whether crime was within or outside region of the police corps where the incident has been reported;

3.3 Creating annotated data

The beta-product developed in CBS (2019) predicted cybercrime using only the declaration field as the text source, since that field is most informative with respect to cybercrime. Hooijschuur et al (2019) extended the original study and also took the clarification field into account because not all records contain a filled declaration field.

CBS received a labelled set of about 5300 records, that was annotated by the police on different types of cybercrime, see Table 3. We recoded them into a binary label (cybercrime yes/no). Because cyber-related crimes are on average found in about 9% of the records – see Table 12 and CBS (2019) – the police decided to use a selective sample to create a labelled set. They used search terms to find records that potentially concerned cyber, see section 3.1.2 in Tollenaar et al. (2018) for more information. The police data set is also referred to as the selective sample. This selective sample cannot be directly used to assess the overall model performance.

For that purpose, CBS (2019) subsequently drew four sets of completely random samples without replacement of 150, 1000, 350 and 300 units respectively, resulting in a total of 1800 units. The sample size was increased stepwise to improve the model performance. A small sample of 300 units acted as an independent test in CBS (2019). The samples were drawn from records with a filled declarations field. Next Hooijschuur et al. (2019) complemented the sample of 1800 units with records with a filled clarification field, such that the resulting sample was a complete random sample from records with either a declaration of clarification field. To that end, a gross sample of 1057 units was needed.

For some SCM codes the sample size was still rather small. Within each SCM code with more than 1000 records in the population with a declaration or clarification field we used a minimum sample size of 50. That way we hoped that we would be able to publish for more separate SCM codes. To achieve this, we drew an additional sample of 346 units. The total gross CBS sample size was 1800 + 1057 + 346 = 3203 records. This CBS data set is also referred to as the random sample. The combination of the police and CBS data set is also referred to as the full sample.

The sampled CBS data were manually annotated. Three of the samples, namely of the samples size 150, 346 and 1057, were labelled by different annotators in order to analyse their agreement. The sample of size 150 was judged by four different annotators, the sample of size 346 was annotated by two annotators and the sample of size 1057 was split into three parts, each part was judged by two annotators, see the second column of Table 2. The annotators were asked to appoint one of three labels to each records: '0' (no cyber), '1' (cyber) or '9' (not enough information in the text). In practice sometimes the annotators were uncertain about their labels and wrote down '0?', '1?' or '?'. Cases where the annotators disagreed and cases where the annotators were uncertain were discussed among the annotators; after which they took a final decision.

We computed the fraction of agreement of the annotators' decisions with the final label. We used '0?' and '0' to be a (preliminary) decision for 0 , '1?' and '1' as a preliminary decision for a 1, and '?' and '9' as a preliminary decision for a 9. A case was counted to 'agree' in Table 2 when the preliminary assignment of all annotators corresponded with the final label.

From Table 2 we conclude that the annotation task was rather difficult. First of all, in more than 10% of the cases a label 9 was assigned implying that there was not enough information to decide whether the crime had a cyber-related aspect. Second, the agreement for label 1 was rather low. On average there was less than

60% agreement among the annotators. We conclude that even for a human judging whether an incident has cyber-related aspects is difficult. The high agreement for label '0' implies that there were also a lot of records for which it was clear that there was no cyber-related aspect involved.

Table 2. Agreement of annotators decisions with the final labels in different CBS samples

Source	Annot.	label 0		label 1		label 9	
		size	agree	size	agree	size	agree
150	A,B,C,D	104	0.990	15	0.733	28	0.000
349	E,F	284	0.968	21	0.381	40	0.825
1057a	E,F	130	0.977	7	1.000	12	0.917
1057b	E,G	490	0.990	21	0.381	37	0.730
1057c	F,G	315	0.962	12	0.750	33	0.667
Total	A-G	1323	0.977	76	0.566	150	0.620

In the remainder of this paper, units with unknown label were dropped. The final labelled set partitioned by source type is given in Table 3.

Table 3. Labelled set by source

Source	label 0	label 1	total
Police	746	4555	5301
S1000	873	101	974
S1057	935	40	975
S150	104	15	119
S300	264	26	290
S349	284	21	305
S350	309	26	335
S-total	2460	203	2663
Total	3515	4784	8299

3.4 Split of annotated data in training and test data

In the current paper, we first tuned the hyper parameters by using all labelled data (including the police data) to train the model and the random sample of 300 units to test the model, see section 3.6 for more details.

Unfortunately, this set of 300 units was too small to test whether the association between label and predicted classes was affected by SCM code. Therefore, we kept the hyper parameters fixed, and we used a five- fold cross validation procedure on the complete labelled set (including these 300 units) for the remainder of our experiments. In each fold we fitted the data on 80% of the records and predicted the remaining 20%. Next, we analysed the results on the pooled set of all predicted units.

For the evaluation of the predictions, we distinguish among 1) the full set of labelled units (incl. the selective sample) and 2) the random sample. The full set was used to test the deviance as given in section 2.5. This was done because enough labelled data were needed for this. Judgement about the overall model performance is only based on the predicted units within the random sample.

Table 4. SCM codes that meet the criteria given above, their number of labelled examples in the full labelled set and their total weight. Total weight corresponds with an estimated population total.

SCM code	SCM Weight	Matched criteria	Number, label 0	Weight, label 0	Number, label 1	Weight, label 1
0	yes	both	677	112607	93	2540
101	yes	both	1375	494457	85	4995
102	yes	both	32	1429	674	44768
103	yes	both	152	9447	2205	15988
104	yes	both	38	5252	7	568
107	no	both	21	3493	7	191
201	yes	both	263	96307	41	434
202	yes	both	90	8915	1403	2865
203	yes	test	48	4829	2	112
301	yes	both	228	45740	19	2095
302	yes	both	207	25287	107	4503
303	yes	both	55	7281	13	910
306	no	test	12	1996	1	27
307	yes	both	44	1090	5	26
400	yes	both	66	9702	88	1990
601	yes	test	51	5453	1	107
602	yes	test	50	7743	1	155
603	no	both	34	5655	6	164
700	yes	both	43	4282	6	598
902	yes	both	29	2807	20	561
Sum			3515	853772	4784	83597

When the model is retrained - as described in section 2.6 - then the model is fitted on (3-digit) SCM code strata, or combinations thereof. We derived a set of SCM strata for which we had enough labelled data to train and test then as separate strata. For this we used the following criteria:
- for testing: a SCM code should have at least 5 records with at least 1 record per class within the CBS-sample.
- for training: an SCM code should have 40 records with at least 5 records per class (cyber/no cyber), within the full sample and it should also be a code that matches the criteria for testing.

There were 16 SCM codes that fulfilled both criteria and 4 codes that could be tested separately, but not trained separately, see the third columns of Table 4. The remaining codes (see Table 18) were joined into a code '0'. The four codes that could not be separately used for training, since they had fewer than 5 codes per class were included in the codes for testing because CBS was interested to know whether the cyber-related crimes could be separated out (preferably all) 3-digit crime codes. Although these codes did not result in significant effects of the deviance - the number of crime codes as simply too small to detect this - we did inspect their model performance results. The second, fifth and last column of Table 4 refers to computation of sampling weights which is explained in section 3.5.

3.5 Computation of sampling weights

The units in the labelled set have unequal inclusion probabilities. To account for this, model performance measures are computed accounting for the sampling weights, as described in section 2. We now first describe the weights for the random sample and then the weights for the full sample.

Sampling weights when using the random sample only. Let n_h^{CBS} denote the size of the CBS sample for stratum h, and let N_h be the population stratum h. The sampling weight for unit i of stratum h is then given by

$$w_{hi} = \frac{n_h^{CBS}}{N_h}.$$

The strata h for which the sampling weights were computed are shown in column 2 of Table 4. It concerned all SCM codes that could be used for testing (first column of Table 4) and for which at least 1000 records with a declaration or clarification text was available, which was not the case for SCM code 107, 306 and 603. These codes were combined into the stratum '0'. The sum of all weights columns 5 and 7) are an estimate of the population size. For the codes 107, 306 and 603 the average weight of stratum 0 is used and as a result the estimated population totals in Table 4 were larger than the actual population number.

Sampling weights when using the full sample. For the selective sample, the proportion of units with label 1 is higher than in the population. We derived a pseudo-design weight as follows. Recall that $y_i = j$ stands for the true label. We assume that the manually annotated label corresponds with the true label. Further let n_{hj}^{ALL} denote the number of sampling units in the full sample for stratum h with label j. Similarly let n_{hj}^{CBS} denote the number of sampling units in the final random sample for stratum h and with label j. The weight of the units in the full sample is then given by:

$$w_{hji}^{ALL} = \frac{n_{hj}^{ALL}}{\widehat{N}_{hj}}, \text{with } \widehat{N}_{hj} = \frac{n_{hj}^{CBS}}{n_h^{CBS}} N_h.$$

where $\widehat{N}_{hj}$ stands for the estimated size of the population in stratum h and label j. An alternative approach to compute a pseudo-design weight when combining a probability with a non-probability sample is given in section 2.1.1. of Vaillant (2020).

3.6 The basic text classifier for cybercrime

The text classifier developed by CBS (2019) concerned an SVM. The features consisted of bag of words, no word stemming nor lemmatisation was used. The basic model was meant to explore whether prediction of cybercrime would be possible or not. Adaptations were left for a later stage. In the current paper we apply exactly the same text preparations steps as was done for the basic model, except when mentioned otherwise (in section 8.1).

For the current paper, the text preparation step consisted of extracting the words in the text (see below), down casing, and removing a list of words from a list of stop words. This list consisted of some Dutch stop words and a partial set of numbers, regions, names of persons and time concepts. Furthermore, we removed texts with 15 or less characters as well as texts with more than 10000 characters (including spaces and punctuation marks). We extracted the words in Python by using a standard regular expression in Python ('(?u)\\b\\w\\w+\\b') which selects tokens of 2 or more alphanumeric characters; punctuation is ignored and always treated as a token separator.

The SVM model uses a linear kernel, no category weighting, L2-penalty and word-weighting (see below). In the present study we explored different settings to optimise the model parameter. First of all, we varied the value of the tuning parameters: the penalty parameter C, word n-gram range and minimum and the maximum document frequency for the tf-idf weighting. Furthermore, we varied two settings to determine the optimal fit: we use F1 or the MCC as the score function and we compared the presence or absence of 'balancing' penalty parameter C (see below). To set the hyper parameters, we split the full labelled set into the 300 sample as the test set and all other units (including the selective sample) as the training set. Within this training set a six-fold cross validation was used, together with a grid-search to find the optimal hyper parameters.

For the original published beta product a six-fold cross validation was used to set the tuning parameters and we maintained this in our situation in which the number of annotated records was larger - we added the sample S1057. With the sample size of the beta product 8299 - 975 = 7324) approximately 6100 records were available to train on in each fold and 1220 records to test on. In the current paper approximately 6900 records were available to train on in each fold and 1380 records to test on. The hyper parameters that gave the best result for the score function as averaged over the folds were chosen.

We found that these latter two settings (F1 or MCC; balancing class weights or not) hardly affected the results. Therefore, we decided to use the basic settings for the remainder of this paper: score function = F1 and class weight = False. We selected the tuning parameters when the data set was trained with all SCM codes combined. The selected parameters are found in the final column of Table 3. These hyper parameters yielded an F1 score of 0.93 on the training set. Thereafter we fixed the hyper parameters. The model that is used has no lemmatisation, a relatively small value for the maximum document frequency (implying that there is a lot of word variation) and no limit on the maximum number of features. The actual amount of features of the model was 1 786 312 ngrams. It would have been interesting to play more with those settings. We come back to that issue in the discussion.

- intermezzo on balancing penalty parameter C - A large value of penalty parameter C implies a small margin around the separating planes, with the risk of overfitting the test set. With small margins, more features are included in the model, with the risk of overfitting on the training set. With balancing the parameter C is set differently per class j, according to $C_j = wht_j * C$, with $wht_j = \frac{1}{J}\frac{1}{\alpha_j}$ where α_j is the proportion of units with class j and is the number of classes. So an underrepresented class wil use a larger C value.

Table 5. Setting to optimise the fit of the SVM algorithm

Tuning parameter settings		Selected
Penalty parameter C	[0.01, 0.23, 0.45, 0.67, 0.89, 1.12, 1.34, 1.56, 1.78, 2, 5, 10, 50, 100]	0.23
Tf-idf: minimum df Ignore words with lower absolute frequency than threshold	[1, 3, 5, 7, 9, 11, 13]	9
Tf-idf, maximum df Ignore words with higher relative frequency than threshold	[0.10, 0.15, 0.20, 0.25, 0.3, 0.35, 0.4, 1.0]	0.25
Word n-gram range	[(1,2),(1,3),(2,2),(2,3)]	(1,2)
Different optimisation settings		
Score function	F1 or MCC	F1
Class weight	False or Balanced	False

We used a term-frequency inverse document frequency (tf-idf) word-weighting. The term-frequency (tf) refers to how often does a token (word) occurs in the text. The inverse document frequency measures in how many documents a token occurs. The idea behind the inverse document frequency is that a token that occurs in more documents is less unique for the target variable for which it is used to predict. In the original model (CBS, 2019) and to determine the hyper parameters, the tf-idf values were computed on the training data and recomputed when predicting the values for the whole population. That implies that the tf-idf values are changing when moving from the training set to the predictions. Especially the tuned thresholds for the minimum and maximum document frequency may not be representative for the full population, which is especially a problem when the model is trained per stratum. For cybercrime we have the situation that we have a fixed, known, population and our only interest is to predict an unknown label. In such a situation one can first compute the tf-idf values for all units in the population, and use those for later training and test steps. That is what we did in the current paper.

The estimated model performance of the original model (CBS, 2019) which was tested on the CBS sample of 300 units, is given in Table 6. Results showed that only 6 out of 290 test records were misclassified. This results in an F1 score for label 0 of 0.99 and an F1 score for label 2 of 0.88. The average proportion of cybercrime was estimated to be $\hat{\bar{Y}} = \frac{26}{290} = 9.0$ % cyber. Because the precision and recall are exacly identical the estimated bias is exactly zero: $B^*\left(\hat{\bar{Y}}\right)$ = 0.0. The estimated standard deviation for the proportion cybercrime was $2.42\ 10^{-6}$, which is extremely small. This is due to the large population size. In the remainder of this document we will only consider the bias because the variance is negligible.

Table 6 Confusion matrix of the basic classifier on the independent test set

	Predicted 0	Predicted 1
True 0	261	3
True 1	3	23

4. Missingness of text fields for cybercrime

4.1 Overall missingness of text fields

The original model (CBS 2019) only used the declaration field. This declaration field is missing in 16.2 % of the records in 2016. Therefore, we decided to merge the text of the declaration and the clarification field. For 4.3 % of the cases neither of the two fields are filled.

Inspection of the data showed that we can further reduce this missingness. For part of the data we found that there are multiple records with a different 'identification number' that share a common 'original registration number'. Records with the same 'original registration number' concern the same incident, but they register different crimes for the same incident. Usually these crimes are separated into a variable 'most offensive crime (yes / no)' with one record for the most offensive crime = yes, and one or more records with most offensive crime = no. In those cases, there is a text about the incident but it is not always repeated for all of those crimes. Overall in 3.1 % of the cases with a missing value this concerns the situation that there is another records with the same 'original registration number' - that does contain a text. Only in the remaining 1.2 % of the cases those conditions are not met, which involves cases for which cyber-related crimes cannot be predicted.

4.2 Missingness as affected by background variables

The extent of missingness was analysed in three ways: 1) for the set of units where the variable 'Original registration number' has multiple records (columns 'multiple' in Table 7), 2) for the set of units where the variable 'original registration number' has no duplicate records (columns 'unique' in Table 7) and 3) for all units in the data set thus ignoring whether the variable 'original registration number' has multiple records (columns 'full' in Table 7).

Table 7 clearly shows that the missingness is higher when the 'original registration number' had multiple records than for the set of unique records. Missingness varied clearly with SCM codes. When focusing on the unique records, it varied from almost 30% for SCM code 204 to 0,2% for SCM codes 102 and 201.

Table 7. Missingness for different variables per SCM code and per background variable for three cases of 'original registration number' (see text).

SCM	full	unique	multiple	Bg-var	Cat	full	unique	multiple
101	0.013	0.006	0.254	Declaration	Yes	0.027	0.001	0.387
102	0.005	0.002	0.321	Declaration	No	0.146	0.087	0.476
103	0.044	0.015	0.448	Own corps	Yes	0.054	0.031	0.422
104	0.393	0.156	0.661	Own corps	No	0.043	0.012	0.411
105	0.139	0.016	0.561	Main	Yes	0.010	0.009	0.026
106	0.068	0.058	0.400	Main	No	0.669	0.450	0.702
107	0.434	0.203	0.767	Series	Yes	0.380	0.216	0.412
201	0.027	0.002	0.446	Series	No	0.009	0.009	0.338
202	0.134	0.029	0.457	Most severe	Yes	0.016	0.011	0.140
203	0.023	0.004	0.262	Most severe	No	0.625	0.632	0.625
204	0.492	0.299	0.753	Attempt	Yes	0.028	0.003	0.368
301	0.064	0.006	0.310	Attempt	No	0.044	0.013	0.414
302	0.093	0.010	0.384	Clarified	Yes	0.138	0.048	0.427
303	0.069	0.025	0.432	Clarified	No	0.009	0.002	0.342
304	0.137	0.029	0.301	Obs field	No	0.046	0.013	0.434
305	0.224	0.029	0.467	Obs field	Yes	0.011	0.007	0.060
306	0.144	0.108	0.361	Online	No	0.068	0.020	0.415
307	0.106	0.022	0.512	Online	Yes	0.000	0.000	0.000
400	0.180	0.061	0.539	Fraud	No	0.045	0.013	0.412
501	0.005	0.002	0.205	Fraud	Yes	0.000	0.000	0.000
502	0.065	0.032	0.319					
503	0.112	0.047	0.257					
504	0.276	0.200	0.299					
505	0.209	0.069	0.714					
506	0.194	0.054	0.541					
507	0.358	0.091	0.574					
508	0.054	0.009	0.224					
601	0.325	0.166	0.599					
602	0.163	0.073	0.313					
603	0.240	0.160	0.532					
700	0.317	0.112	0.666					
902	0.205	0.062	0.665					

Furthermore, missingness also varied clearly with the categories of the background variables (right column of Table 7). High levels of missingness occurred for the

'Variable: Category' combinations: Declaration: No, Main: No, Series: Yes, Most severe: No and Clarified: Yes.

In the forward model selection procedure of the logistic regression (see equation (2) in section 2.2) we found that the best fitting model missingness was explained by the following variables (in order from most to least important): Main, SCM, Declaration, Police corps, Clarified, Most severe, Series and Outside Region. The AIC value was 116 807 at 979 701 residual degrees of freedom. The fitted model was not significantly different from the saturated model using a chi-squared test (p-value = 1.0), implying that the model described the data well. Nearly all model coefficients were highly significant, this concerned all categories of the variables (not shown). Exceptions were SCM code 505 and 506 whose coefficients were not significant.

Table 8. Coefficients and their significance, of the logistic regression of cyber-related crimes versus background variables (see text).

	Estimate	Std.Error	t.value	Pr.t.value
(Intercept)	-5.37	0.66	-8.20	0.00
101	0.27	0.67	0.40	0.69
102	8.23	0.84	9.84	0.00
103	5.54	0.63	8.79	0.00
104	3.46	0.81	4.29	0.00
107	4.60	0.73	6.31	0.00
201	-0.53	1.18	-0.45	0.65
202	4.31	0.71	6.09	0.00
203	1.17	1.18	0.99	0.32
301	2.21	0.72	3.06	0.00
302	3.48	0.66	5.27	0.00
303	3.23	0.73	4.44	0.00
306	3.02	1.25	2.41	0.02
307	1.18	1.17	1.01	0.31
400	3.94	0.70	5.65	0.00
601	2.09	1.20	1.75	0.08
602	1.85	1.18	1.57	0.12
603	3.94	0.72	5.44	0.00
700	4.02	0.77	5.22	0.00
902	3.86	0.77	5.02	0.00
Declaration:No	-0.67	0.30	-2.22	0.03
Clarified:No	0.64	0.31	2.05	0.04

4.3 Cyber as affected by background variables

Using a logistic regression model according to equation (3) in section 2.2, we analysed to what extent differences in cyber-related crimes could be explained by the background variables in the data set. We restricted this analysis to the annotated random sample. We only allowed for main effects in the logistic regression, because preliminary analysis showed the inclusion of interactions lead to unstable parameter estimates. Probably that instability occurred because certain combinations of background variables rarely occurred.

We found that for the model with the best AIC, the cyber-related crimes were explained by the variables SCM code, Declaration and Clarified. The AIC was 680 at 2964 residual degrees of freedom. The fitted model was not significantly different from the saturated model using a chi-squared test (p-value = 1.0), implying that the model described the data well.

Many of the individual coefficients of SCM code were significant, see Table 8. Thus, the differences in percentages of cyber-related crimes are well explained by the variables SCM code, by Declaration and by Clarified.

4.4 Conclusions

We found that missingness is affected by nearly all background variables, whereas cyber-related crimes are only related to three background variables: SCM code, Declaration and Clarified. Of course, the latter relation can only be determined for records with text data. That means that, when one decides to correct the estimates for missingness, one needs to assume that the relation of the background variables with cyber-related crimes in the records with missing text data is the same as for the records with text data. We have no direct reason to believe that the variables Main, Police corps, Most severe, Series and Outside Region would be (causally) related to the extent of cyber-related crimes. Therefore, we expect that we can use these three background variables to correct for the missingness, at least for the strata where the missingness is small.

5. Sensitivity of bias for retraining

5.1 Understanding the bias sensitivity

The bias estimates were computed for the various retraining strategies explained in section 2.6. Recall that for the main strategies BINARY and DEVIATING, two models were used to test for effects of crime type: model 2 that included both labels in the same model and model 3 in which label 0 and label 1 were tested separately, see section 2.6. The bias estimate, $\hat{B}^*\left(\hat{\bar{Y}}_h\right)$ according to equation (6), varied over the different retraining strategies from 0,018 to 0,032 for the full sample and from -0,02 to -0,01 for the random (CBS) sample. Similar ranges for the bias were found for the confusion based estimate of the bias,

$\hat{B}^{CF}\left(\hat{\bar{Y}}_h\right)$, according to equation (5). This result is more stable than we found previously in Hooijschuur et al. (2020).

Table 9 Estimated bias for when retraining the model.

Retraining		Full sample		Random sample	
Strategy	Model	$\hat{B}^{CF}\left(\hat{\bar{Y}}_h\right)$	$\hat{B}^{*}\left(\hat{\bar{Y}}_h\right)$	$\hat{B}^{CF}\left(\hat{\bar{Y}}_h\right)$	$\hat{B}^{*}\left(\hat{\bar{Y}}_h\right)$
BASIC		0.026	0.030	-0.012	-0.017
BINARY	2: both labs	0.028	0.032	-0.012	-0.016
	3: per label	0.024	0.028	-0.014	-0.020
DEVIATING	2: both labs	0.016	0.019	-0.011	-0.015
	3: per label	0.015	0.017	-0.008	-0.010
ALL		0.015	0.018	-0.008	-0.011

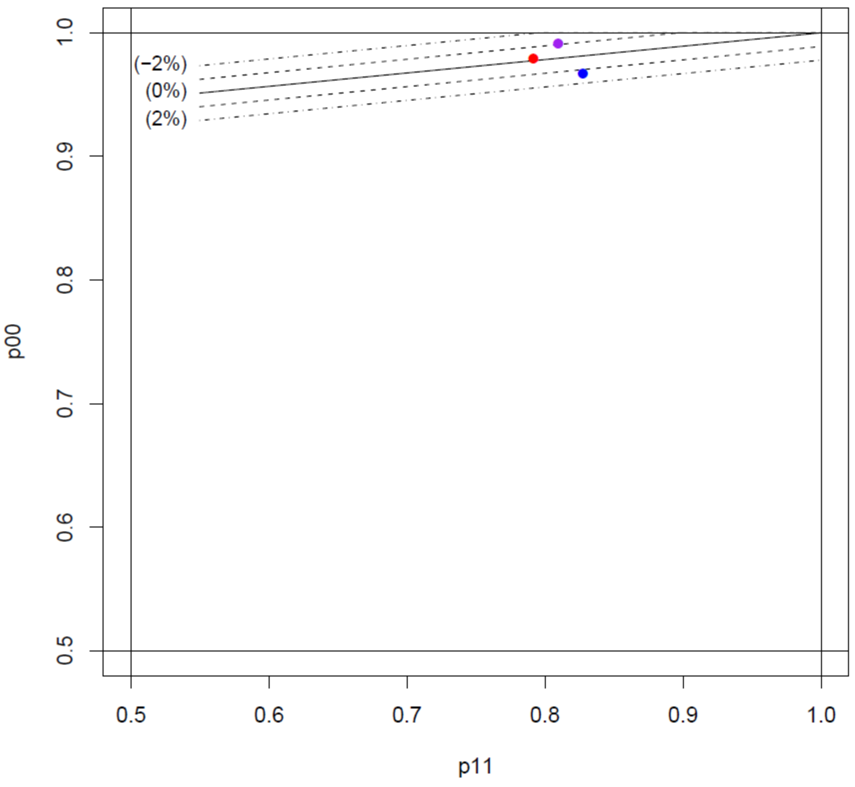


Figure 1. Isolines $(\boldsymbol{p11}, \boldsymbol{p00})$ such that $\boldsymbol{B}(\hat{\boldsymbol{\alpha}}\boldsymbol{1q})$ $=\boldsymbol{\beta}$ for the cybercrime application with $\boldsymbol{\alpha 1q}$=$\boldsymbol{9.8}$% (solid line: $\boldsymbol{\beta}$=$\boldsymbol{0}$; dashed lines: $\boldsymbol{\beta}$=±$\boldsymbol{1}$%; dot-dashed lines: $\boldsymbol{\beta}$=±$\boldsymbol{2}$%). The red, blue and purple points indicate $(\boldsymbol{p11}, \boldsymbol{p00})$ for the basic, retrained and hypothetical model, respectively.

We have looked into the cause of the model sensitivity of retraining on the bias. This has been documented in a separate discussion paper: Scholtus and Van Delden (2020). Based on the analytical formulas for the bias, they showed that relatively small changes in the error probabilities $\mathbf{P}$ can already lead to considerable changes in the bias, when the proportion of the target variable in the population is small.

This is illustrated in Figure 1. The lines in Figure 1 represent combinations of p_{00} and p_{11} which lead to the same bias. The red dot represents a basic model from Hooijschuur et al. (2020). The purple dot represents an improved model with better values for both p_{00} and p_{11}, but still the bias is increased.

There are different options to reduce the bias of the output. The first option is to choose a threshold for the predicted probability (a calibrated confidence) such that the predicted false positives and false negative exactly balance each other. With this threshold we mean the threshold that is used to decide whether a unit belongs to class 1 (cyber-related aspects) or class 0 (no cyber-related aspects). The second option is, for a given classifier, to correct for the classification errors. Different ways to correct for classification errors are compared in Kloos et al. (2020). Correction for classification errors usually lead to an increase in variance of the estimator. Kloos et al. (2020) worked out which correction works best under which conditions. Last but not least, we are interested to publish outcomes for different subpopulations. In this situation, the first thing to do is to develop a text mining classifier that separates the cyber from non-cyber as good as possible for the different subpopulations. Thereafter, we have to estimate the bias for each of the subpopulations and correct for them. For subpopulations with very small proportions of cyber one could also opt for a model with a high recall and then check the uncertain cases manually.

5.2 Conclusion

The bias of the estimated cyber proportion is sensitive for small changes in the proportions of false positives and false negatives, because the proportion of cyber in the population is small. This has been shown in a separate discussion paper by Scholtus and Van Delden (2020). Different ways to reduce the bias are discussed in the main text of section 5.1.

6. Model performance affected by retraining on standardised crime type

6.1 Retraining

We tested to what extend the performance of the originally developed model BASIC) varied with SCM code or not according to the deviance measure. First of all, we tested whether the association between true and predicted label was affected by SCM code, (log linear model 2, equation (11)). We found that the G^2 of 208 which was highly significant ($P = 0$). We found that the SCM codes 0, 101, 201, 302 and 400 had a significantly different deviance. That implies that these groups were separately trained - either combined into one group (BINARY) as opposed to all others, or each SCM code individually (DEVIANCE).

Next, we tested whether for true label 0 the association between predicted fraction of label 0 was affected by SCM code. We found a G^2 of 1052 which was highly significant ($P = 0$). Except for the SCM codes 104, 306 and 902 all SCM codes were deviating. Finally, we repeated this test for true label 1 and found a $G^2$of 650 which was also highly significant ($P = 0$). Again all codes were affected, except 0, 201, 203, 307 and 902. SCM codes that were significantly affected for both label 0 and 1 were trained separately, see Table 10.

Table 10 SCM codes of the BASIC mode that have a significantly affected deviance (see text).

SCM code	P < 0.05 of deviance per code			
	model 2	model 3, label 0	model 3, label 1	model 3, intersection
0	yes	yes	no	no
101	yes	yes	yes	yes
102	no	yes	yes	yes
103	no	yes	yes	yes
104	no	no	yes	no
107	no	yes	yes	yes
201	yes	yes	no	no
202	no	yes	yes	yes
203	no	yes	no	no
301	no	yes	yes	yes
302	yes	yes	yes	yes
303	no	yes	yes	yes
306	no	no	yes	no
307	no	yes	no	no
400	yes	yes	yes	yes
601	no	yes	yes	yes
602	no	yes	yes	yes
603	no	yes	yes	yes
700	no	yes	yes	yes
902	no	no	yes	no

We evaluated the model performance according to the retraining strategies with respect to crime type as described in section 2.6. The evaluation measures we used are given in section 2.4. First, in section 6.2 we concentrate on the results at national level, then, in section 6.3, we look into describe the results per crime type.

6.2 Results at national level

The model performance with respect to predictions at national level were tested on the random sample, see Figure 2. The Macro F1 score (about 0.90) was found to be smaller than the micro F1 Score (about 0.97), because recall and precision of class 1 (cyber-related crimes) were clearly smaller than recall and precision of class 0 (no cyber-related crimes). Both the macro and micro F1 scores were hardly affected by the retraining strategies. Both the recall of class 0 (0.99) and its precision (0.98) were high and hardly affected by the retraining per crime type. The recall of class 1 slightly increased with more detailed retraining from 0.74 (BASIC) to 0.78 (ALL). The precision of class 1 slightly decreased with more detailed retraining from 0.91 (BASIC) to 0.89 (ALL). For completeness, the bias $\hat{B}^*\left(\hat{\hat{Y}}_h\right)$ is given in the bottom- left panel of Figure 2; it has also been given in the last column of Table 9 and has been discussed in section 5. The entropy R^2(bottom-right panel) was 0.84-0.85 and was hardly affected by the retraining strategies.

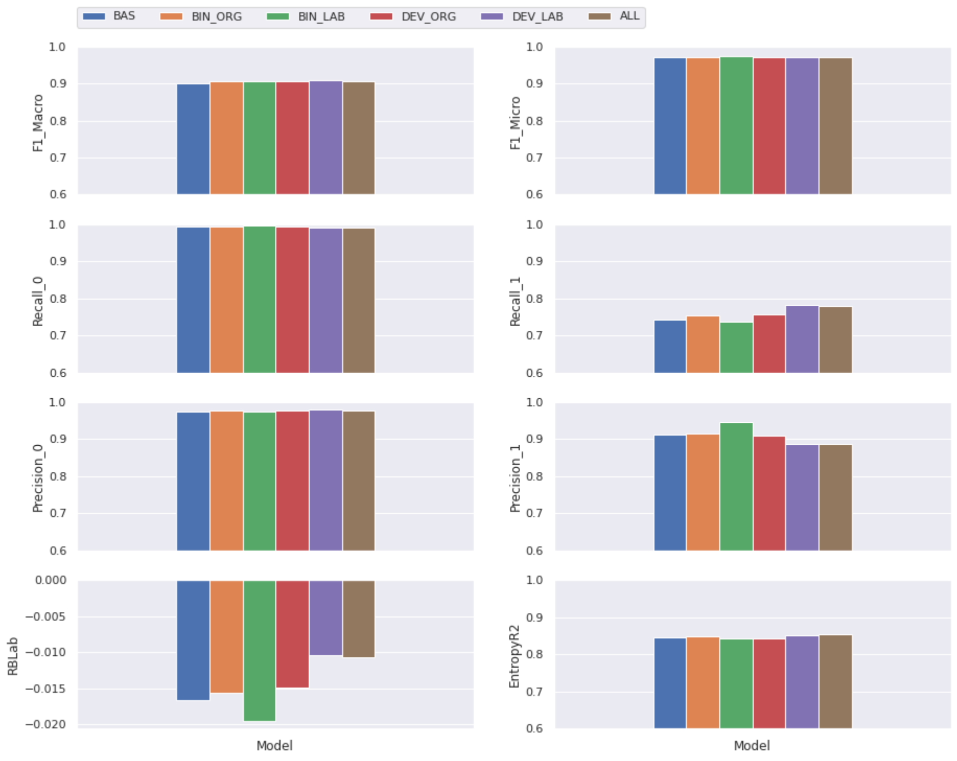
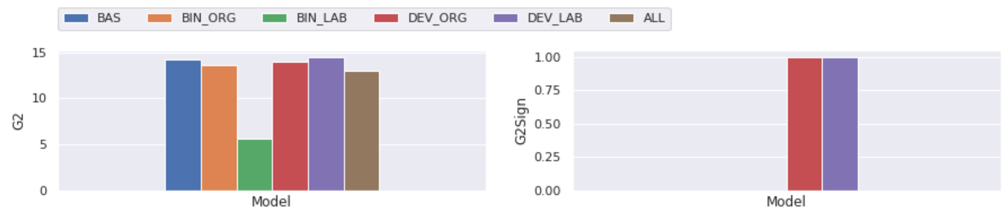


Figure 2. Model performance measures at national level tested on the random sample for six retraining strategies.

The deviance results are given both for the random sample and for the full sample in Figure 3. For the random sample, the deviance value (G^2) was around 13-14 for all retraining strategies except for the BIN_LAB retraining for which the G^2 dropped to a value of 6. In two retraining strategies there was a single class for which crime type significantly affected model performance: DEV_ORG and DEV_ALL. Since the number of units to test the effect of crime type on model performance was in fact too small for the random (CBS) sample, we tested the deviance also on the full sample (Figure 3). Here we found the deviance value to be highest for the retraining strategies BASIC (208) and BIN_LAB (220), while the

G^2 values of the other strategies were in the range 158 - 167. Depending on the retraining strategy, we found 4 - 6 categories for which the G^2 values were significant at $\alpha = 0.05$. Unfortunately, a more detailed retraining did not reduce the number of significant crime categories. To better understand these results, we will look into the model performance per crime category, section 6.3.

source: random sample



source: full sample

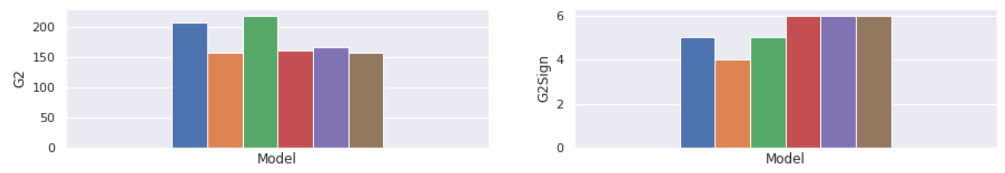


Figure 3. Deviance results tested at national level tested on the random sample (upper row) and on the full sample (lower row) for six retraining strategies.

6.3 Results per category, model 2

The results per category were tested on the full sample. We first concentrate on the recall of class 0 and 1 and the deviance values for the four basic retraining strategies BASIC, BINARY, DEVIATING and ALL of model 2.

The recall of class 0 (no cyber) is high for most of the tested categories, with exceptions for SCM code 102, 103, 202 and 400, see Figure 4 and Figure 5. SCM code 102 has a high proportion of cybercrime which may have caused this effect. But that does not hold for categories 103, 202 and 400. The recall of class 1 (cyber) varied greatly with SCM code. The deviance values varied greatly with SCM code, high scores were found for the SCM codes 0, 201, 302 and 400. It is difficult to directly explain why these cell have high scores since high scores are the results of a three-way interaction and the score level depends on the size of the cell. Nonetheless, the main conclusion is clear: the recall of class 0 and 1 varied greatly with SCM code. Furthermore, as we have seen before, the recall of class 1 per SCM code did not consistently improve with more detailed retraining: for some codes it improved, for others it became worse with retraining.

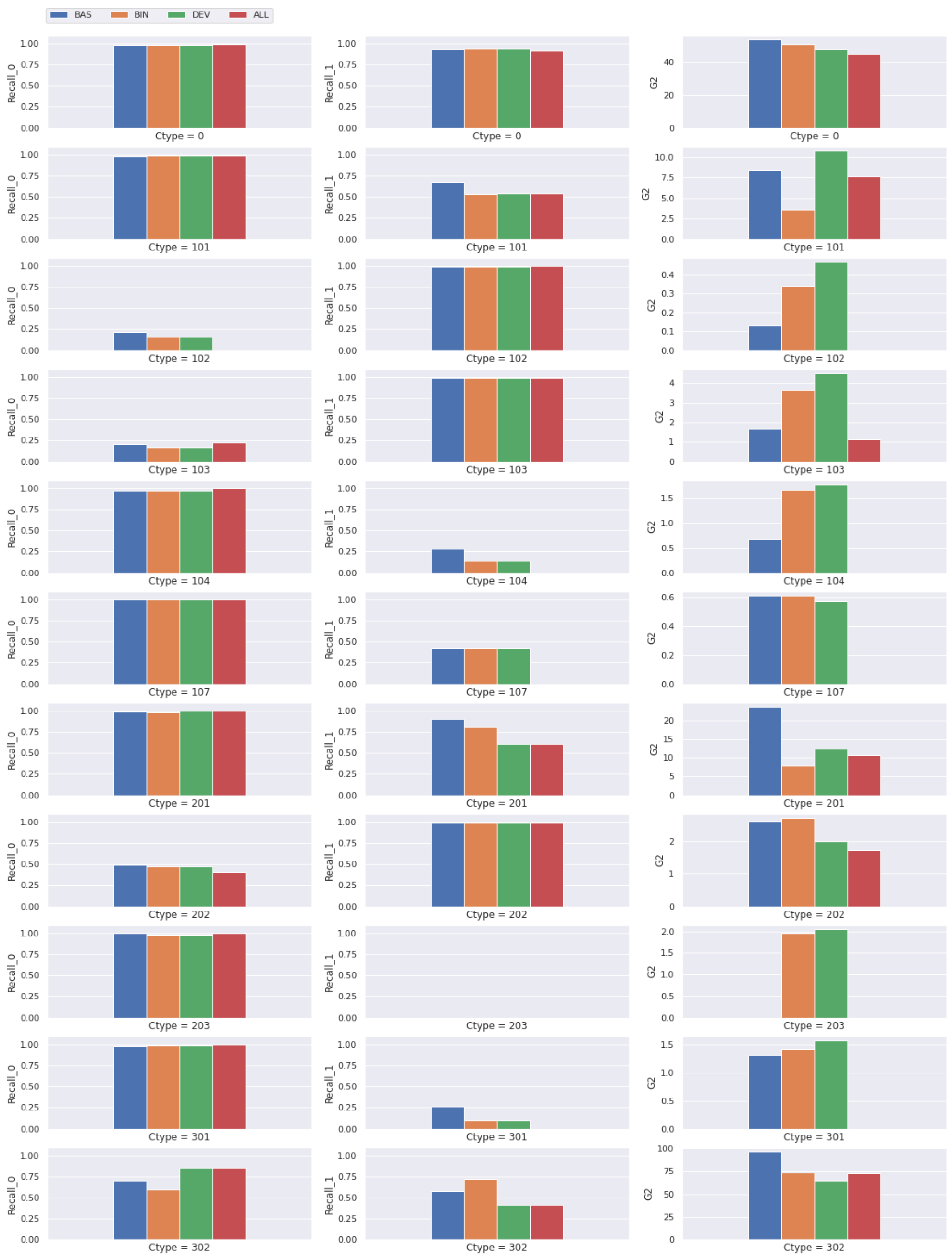
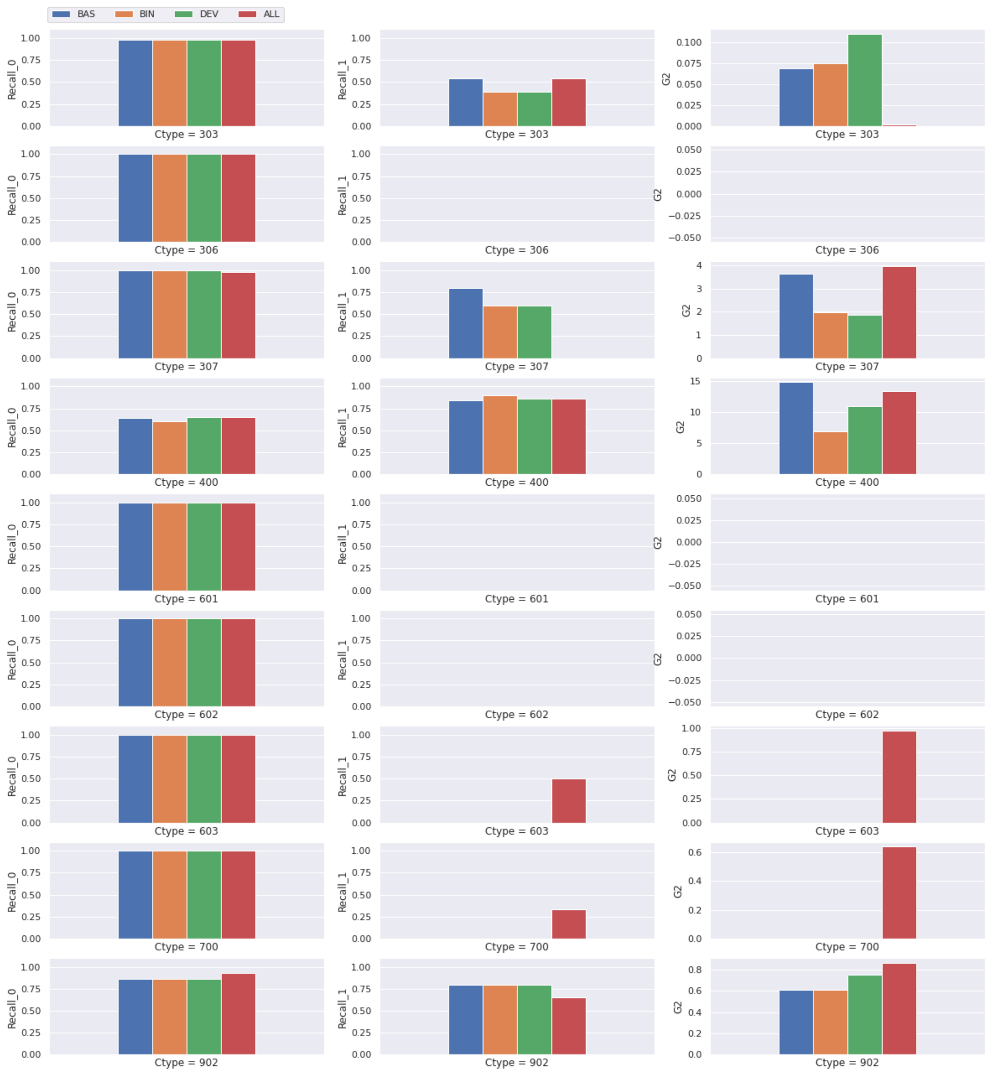


Figure 4. Recall of class 0 and 1 and G2 score (model 2) per crime category tested on the full sample for four retraining strategies.



.

Figure 5. (continuation of Figure 4) Recall of class 0 and 1 and G2 score (model 2) per crime category tested on the full sample for four retraining strategies.

6.4 Results per category, model 3

The deviance values that were found per category, for the four retraining strategies BASIC, BINARY, DEVIATING and ALL according to log-linear model 3 are given in Table 11. Most of the deviance values per code were significantly affected, which confirmed the results found in section 6.3 that the recall of the model for a given label varied considerably with SCM code. As we have seen before, retraining the model did not reduce the number of SCM codes that were significantly affected.

Table 11 The deviance values per label (log linear model 3) per SCM code. All values of 4.0 and larger are significant at $P < 0.05$.

CTest	BAS		BIN		DEV		all	
	label 0	label 1	label 0	label 1	label 0	label 1	label 0	label 1
0	72.5	1.3	84.4	0.9	71.2	0.6	67.6	2.6
101	130.4	78.1	120.8	82.9	155	127.3	156.5	125.6
102	86.9	27	96.5	31.6	163.1	63.5	162.6	64.9
103	429.2	82.9	417.8	112.2	445.3	119	443.6	123.1
104	2.7	24.2	7.0	32.0	6.2	31.3	6.2	42.8
107	4.1	16.6	3.9	44.0	3.4	43.1	3.4	42.8
201	36.5	2.6	25.8	2.2	42.9	46.0	43.2	45.4
202	103.1	37.3	106.8	36	154.2	73.9	153.5	76.5
203	9.3	12.9	8.9	12.6	7.8	12.3	7.9	12.2
301	22.1	69.1	28.5	94.2	37.1	117	37.4	116.2
302	68.1	150.6	63.6	165.3	10.4	240.5	10.2	237.8
303	5.2	21.5	4.8	20.4	3.9	19.7	4.0	19.4
306	2.3	6.5	2.2	6.3	2.0	6.2	2.0	6.1
307	8.5	1.8	8.1	6.1	2.6	30.8	2.6	30.6
400	36.0	19.4	34.3	30.1	38.9	11.0	38.6	10.6
601	9.9	6.5	9.4	6.3	8.3	6.2	8.4	6.1
602	9.7	6.5	9.2	6.3	8.1	6.2	8.2	6.1
603	6.6	38.8	6.3	37.7	5.5	10.4	5.6	10.3
700	8.3	38.8	7.9	37.7	7.0	17.2	7.1	17
902	0.6	7.2	1.3	10.2	1.0	9.7	0.0	18.2

6.5 Conclusions

At national level, the recall of class 1 is clearly smaller than the recall of class 0. The recall of both classes varies considerably with SCM. In that sense the model performance is not uniform over the SCM codes. Also the deviance tests show a significant effect. Retraining of the SVM model on more detailed SVM codes does not reduce the variability of the recall with SCM codes. The recall of class 1 is not improved by (just) increasing the total size of the training set, nor by lemmatisation nor by under- or oversampling. In section 9 we will discuss what might be done to improve the recall of class 1 and to reduce the variability of the recall with SCM codes.

7. Model performance affected by retraining on cybercrime type

7.1 Explain cyber type and the labelled data

We explored whether we could improve the model performance by distinguishing among different forms of cyber-related crimes. As explained before in section 2.6, there are two main types of cybercrime: pure cybercrime - where the computer is the target of the criminal activity, and digital crimes 0 where the computer is the tool to commit a crime. Some forms of pure cyber are hacking (HA), malware (MA), ransomware (RA) and DDos attacks (DD). Some forms of digital crimes online purchase fraud (OP), online identity fraud (OI), online abuse (OA), see Alkaabi (2021) for a complete overview.

Table 12. Estimation of the population distribution of main cyber type by SCM code see text) $^{(a)}$.

SCM	Pure Cyber	Digital crimes	Other Cybercrime	No Cybercrime
0	2718	203	0	123751
101	2572	2194	227	494457
102	14845	27824	2098	1428
103	5519	10325	142	9447
104	567	0	0	5252
201	349	84	0	96307
202	1838	756	22	9162
203	112	0	0	4828
301	1611	322	161	45740
302	2086	2251	164	25286
303	260	650	0	7280
307	25	0	0	1090
400	477	1512	0	9701
902	400	160	0	2806
total	33379	46281	2814	836535

(a) The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

To get an impression of how cyber type was distributed over the SCM codes, we estimated a frequency table of cyber type by SCM codes. This table was estimated as follows. First, using the random sample and using the estimated population weights per sampling unit, we first estimated the fraction of cyber per SCM code. Second, we restricted the police sample to those units with a label cybercrime and per SCM code we estimated the distribution of type of cyber (given the record has cyber-related aspects). Third, we combined the two findings. The set of SCM codes for which we estimated the proportions of cyber type was smaller than the ones used in section 6. This was due to two reasons: a) we included only SCM codes for which we could compute a separate sample weight (see second column of Table 4) and b) we could only include SCM codes for which there were annotated police records.

The estimated distribution of main cyber type per SCM code is given in Table 12. Pure cyber is dominating in SCM codes 104, 201, 203, 301, 307 and in the rest category 0. Digital crimes are not really dominating any SCM codes, but it concerns the majority cyber type in SCM code 102, 103 and 400. In some of the SCM codes there were also forms of cybercrime that were not further specified. Apparently it was unclear to the annotators which exact form of cybercrime it concerned.

Table 13. Estimation of the population distribution of main cyber type by SCM code see text). $(a)(b)$

SCM	Pure Cyber				Digital crimes			OT	NO
	HA	RA	DD	MA	OS	OI	OA		
0	135	2480	101	0	0	67	135	0	123751
101	2118	454	0	0	832	1059	302	227	494457
102	13368	1010	310	155	24948	2875	0	2098	1428
103	4143	1203	52	120	6858	3406	60	142	9447
104	567	0	0	0	0	0	0	0	5252
201	108	180	24	36	48	36	0	0	96307
202	866	771	156	43	561	148	45	22	9162
203	0	112	0	0	0	0	0	0	4828
301	1611	0	0	0	161	0	161	161	45740
302	1812	219	54	0	54	54	2141	164	25286
303	130	130	0	0	0	0	650	0	7280
307	6	19	0	0	0	0	0	0	1090
400	398	79	0	0	0	0	1512	0	9701
902	160	120	120	0	160	0	0	0	2806
Total	25422	6777	817	354	33622	7645	5006	2814	836535

(a) Hacking (HA), Malware (MA), online sales fraud (OS), online identity fraud (OI), online abuse (OA), ransomware (RA), DDos attacks (DD), other cybercrime (OT), no cybercrime (NO).
(b) The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

Pure cyber and digital crimes per SCM codes is further broken down into subtypes in Table 13. Interestingly we now see that some of the SCM codes have a clear peak on a specific cyber type, while others show a mixture. The rest category, SCM 0, involves mainly ransomware, SCM 102 (swindle) involves mainly hacking and online purchase fraud, SCM 104 (extortion) only involves hacking, while SCM codes 203 (arson, explosion) and 307 (violence crime) mainly involve ransomware. SCM code 302 (threatening and stalking) has a peak at hacking and at online abuse. SCM code 303 (sexual crime) has peaks at hacking, ransomware and online abuse. SCM 400 (other crimes WvSr) also has a peak at online abuse. At least in part of these cases there appears to be a plausible relation between the SCM code and the cyber type. For SCM code 202 (public order and safety offence) all kinds of cyber were found.

7.2 Results

7.2.1 Effect training on cyber type

We investigated whether training the model on three classes: pure cyber, digital crimes and no cyber improves the scores on model performance measures as opposed to training on a binary variable. In total there were 4784 records with a label 1 in the annotated data set and 3515 of label 0 (see Table 3 and Table 14). The training set for cyber type contained 4405 records for which we also had an annotated cybercrime type from the police sample and 3497 records of label 0 (see Table 14). A few records were lost because we could not uniquely link them to the original BVH data and further for some records the police annotators did not fill in a cyber-type classification.

Table 14. Original and annotated set for cyber type.

Source	label 0 (original)	Label 0 (cyber type set)	label 1 (original)	Label 1 (cyber type set)
Police	746	728	4555	4405
S-total	2460	2460	203	0
Total	3515	3497	4784	4405

Table 15. Annotated set that could be used for cyber type, broken down by SCM code. (a)

SCM code	Pure cyber	Digital crimes	No cyber	Total
0	1	2	959	962
101	28	36	1372	1436
102	179	376	30	585
103	576	1539	149	2264
105	7	78	14	99
201	14	22	261	297
202	560	804	87	1451
301	10	2	228	240
302	35	45	205	285
303	1	6	55	62
307	1	3	44	48
400	15	61	64	140
902	7	7	29	43
total	1434	2981	3497	7912

(a) The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

As criteria to determine which SCM codes to distinguish for testing we used that a code should have at least 10 records with at least 1 record for each of the three classes. These criteria were met by the SCM codes 0, 101, 102, 103, 105, 201, 202, 301, 302, 303, 307, 400 and 902. Note that this set slightly differs from the set in section 6. The final set for training , broken down by SCM code is given in Table 15.

Using this labelled set, we compared the two training strategies. The units in the labelled set were weighted, in order to correct for selectivity in the data set as much as possible by using the weights given by $w_{hji}^{ALL} = \left(\widehat{N}_{hj}\right)^{-1} n_{hj}^{ALL}$ as described before in section 3.5, where n_{hj}^{ALL} is given the by available records per SCM code h and label j in the current labelled set.

Using the same hyper parameters as for the retraining strategies of section 6 (Table 5) we used a five-fold cross validation to predict all cases in the set. After training the model on the classification variable with three classes, we transformed the results into a binary classification variable and then computed the same model performance measures for both types of training. We only tested the results on the full sample (for which cyber type is available), since for the units in the random sample the cyber type label is not available.

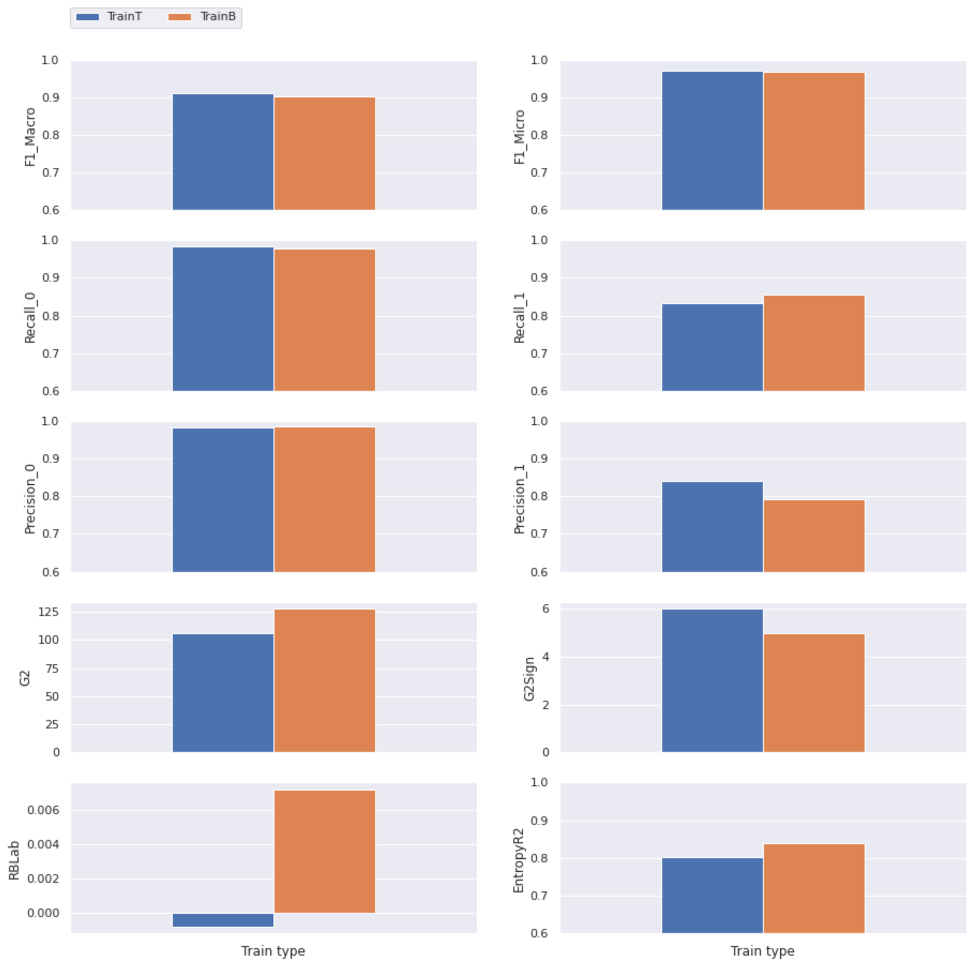


Figure 6. Model performance at national level, tested on the random sample for main cyber type versus binary.

Results on the recall and deviance values per SCM code for both models

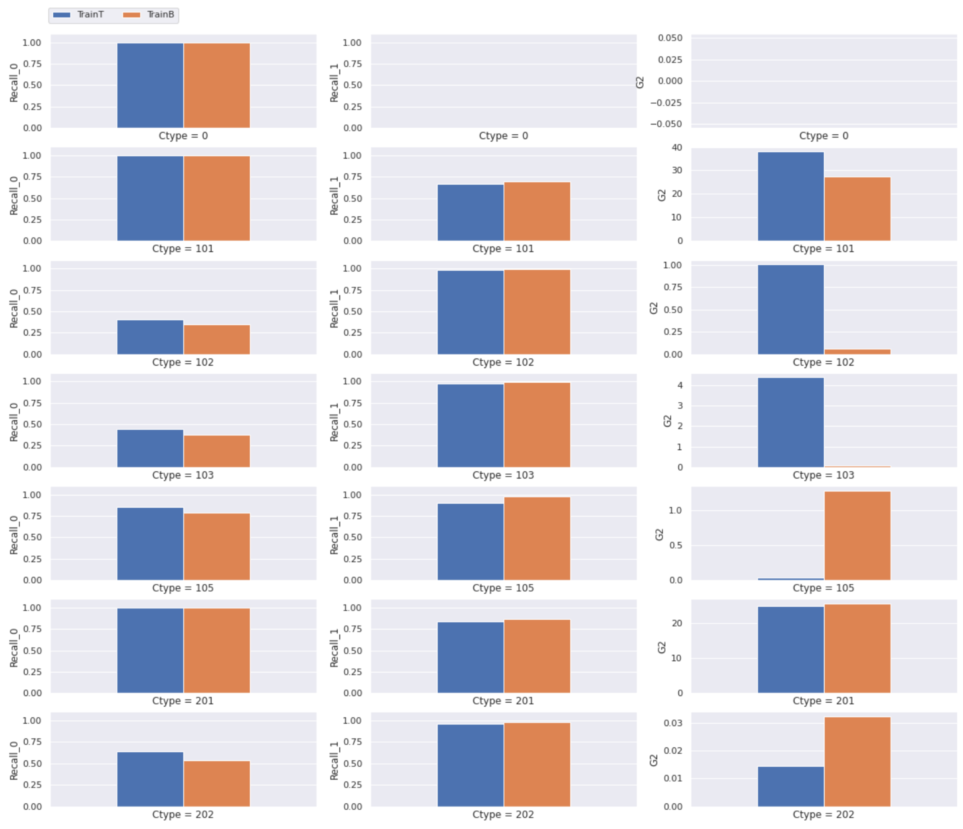


Figure 7. Recall of class 0 and 1 and G2 score (model 2) per crime category tested on the full sample and for main cyber type versus binary.

We found that the macro-F1 scores for training per cyber type (TrainT) was nearly the same as for training for the binary variable (TrainB). The recall and precision of class 0 hardly differed between TrainT and TrainB, while the recall of class 1 was slightly lower and that of class 0 slightly higher for TrainT compared to TrainB. The deviance of TrainT was somewhat smaller that for TrainB which indicated an improvement, although the number of significantly affected classes was higher for TrainT than for TrainB. The overall entropy R^2 of TrainT was smaller than for TrainB. In more detailed results we found that the model that was trained on cyber type often mixed up pure cuber and ditigal crimes. Finally, we found that the absolute value of the bias of the TrainT model was smaller than that for TrainB, but for both models the estimated bias was small.

Recall and deviance scores per SCM codes are given in Figure 7 and Figure 8. Generally, the recall of class 0 is high for all SCM codes except for SCM 102, 103, 105, 202 and 400 while difference between TrainT and TrainB were small. The SCM code effects gave the same result as found before in section 6.3, except that in that section 6 SCM code 105 was not separately considered. The recall of class 1 (cyber) varied greatly with SCM code. The recall was especially good for SCM codes 102, 103, 202 and 903. Other codes gave poorer results. Differences between TrainT and TrainB were small for most of the SCM codes, except for SCM 307 with poorer results for TrainT.

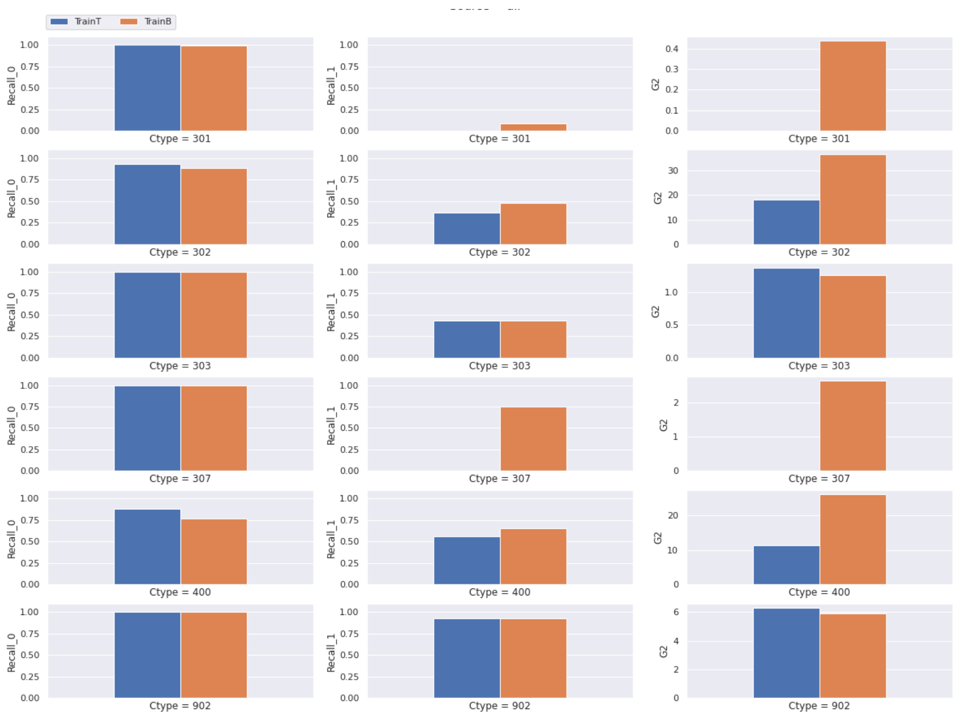


Figure 8. Recall of class 0 and 1 and G2 score (model 2) per crime category for source = full sample and for main cyber type versus binary. (cont.)

7.3 Conclusions

Cyber-related crimes in fact consist of a mixture of cyber types. Some of those cyber types have a peak at certain SCM codes. We estimated that, in absolute numbers, online purchase fraud occurs mainly in SCM codes 102 and 103, online identity fraud at SCM codes 101, 102 and 103 and online abuse in SCM codes 302 and 400. Hacking has a peak in SCM code 102 but occurs in all other SCM codes. Although cyber types have a peak at certain SCM codes, in many different SCM codes different types of cyber occur. We therefore expected that training on cyber type helps to improve the recall of class 0 and 1 and its stability over SCM codes. We restricted the retraining distinguishing among to pure cyber versus digital crimes. Unfortunately results showed no real improvement for the recall of class 1 (cyber) compared to training a binary variable.

8. Evaluating model decisions

We performed a number of small experiments to better understand the results of the model that predicts cybercrime as a binary variable and to try to improve the recall of class 1.

8.1 Effect of lemmatisation

In the first experiment we have altered the text processing. The model of the beta product was a bag of words model without any word standardisation step. We tested whether lemmatisation of the text before training and testing the model would improve the results. Lemmatisation is a method to normalise the original words in the text to their lemma, that is to their dictionary form. We used a Dutch language model (nl_core_news_sm, version 2.00) in Spacy 2.1 for the lemmatisation. Unfortunately, results after lemmatisation were nearly identical to the results that were obtained without lemmatisation.

8.2 Effect of under- and oversampling

In the second experiment we tested the effect of using under- and oversampling of label 0 or 1 during training of the model on the recall of label 0 and 1 in the test set. For the test set, we used the random sample of 1000 units (CBS sample of 1057 units). The over- and under sampling was done per SCM code, using the set of SCM codes that were also used for testing. In this experiment the sample size of label 0 and 1 are made equally large: with oversampling additional units are drawn with replacement from the smallest sample, with under sampling units are dropped from the largest sample. First results showed that oversampling yielded better results than under sampling, so we will only present results of oversampling. Results of oversampling compared to training without over- or under sampling were very similar for the full training set (see Figure 9, fraction 1.0).

8.3 Effect of size of the training set

In the third experiment, we tested the effect of the size of the training set on recall of label 0 and 1. We did so for the situation without under- and oversampling and for the situation with oversampling. We reduced the size of the training set stepwise, stratified by the label. That way we created a learning curve. We repeated the procedure of randomly removing the units five times each time determined the recall of class 0 and 1 and then the average result, which is shown in Figure 9.

First of all, Figure 9 shows that when we trained without under and oversampling, that - surprisingly - the recall of class 0 and 1 was hardly affected by the size of the training set. Only in case of oversampling, the recall of class 1 reduced with size of the training set. Furthermore, we found that, oversampling has a very limited effect of the results. Oversampling slightly reduced the recall of class 1 and slightly increased the recall of class 0 when the training set is about 70 percent or less of the full training set.

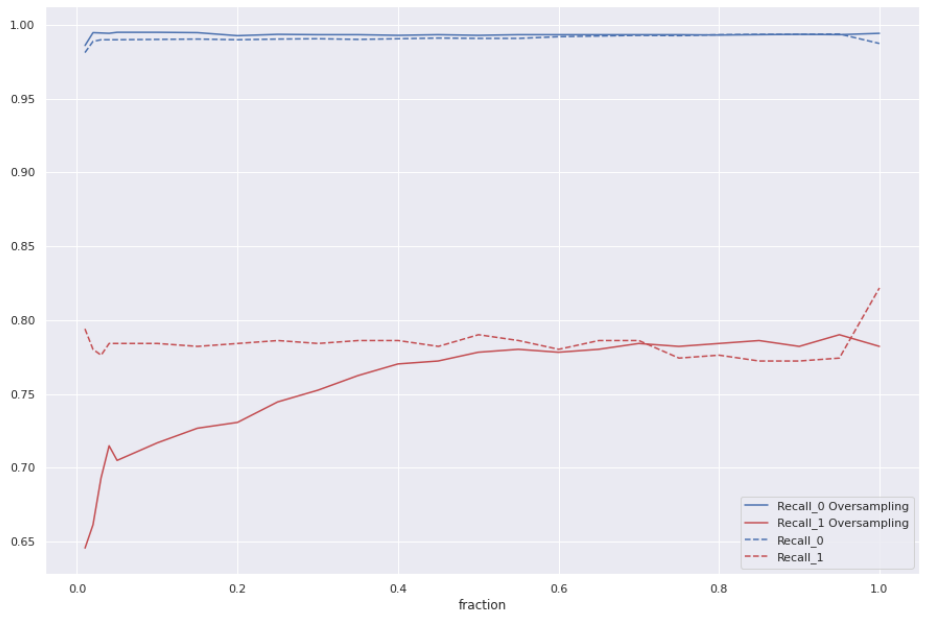


Figure 9. The recall as a function of the fraction of the full sample size (7240) for the BASIC model without (broken lines) and with oversampling (straight lines) (see text).

In order to understand why the recall of both class 0 and 1 was insensitive to the training size, we further looked into predictions in the test set, see Table 16. Results show that the recall of label 1 depends on SCM codes 102 and 103, and recall of label 0 depends on SCM codes 0, 101 and 201. If we would give each SCM code an equal weight, then a smaller value for the recall of both classes would have been obtained.

Table 16. Original and annotated set for cyber type.

CTest	Size 0.005 (73 label 0, 40 label 1)				Size: 0.001 (3 label 0, 5 label 1)			
	Label 0		Label 1		Label 0		Label 1	
	Pred 0	Pred 1	Pred 0	Pred 1	Pred 0	Pred 1	Pred 0	Pred 1
0	95	2	0	0	92	5	0	0
101	565	15	5	4	513	67	2	7
102	0	1	1	51	1	0	2	50
103	4	8	0	26	1	11	2	24
105	1	1	0	0	0	2	0	0
201	103	0	0	0	97	6	0	0
202	4	0	1	2	2	2	0	3
301	42	1	0	0	27	16	0	0
302	22	0	3	3	13	9	4	2
303	2	0	0	0	1	1	0	0
307	2	0	0	0	1	1	0	0
400	4	0	3	2	4	0	1	4
902	1	0	0	0	1	0	0	0

8.4 Most influential features

We looked into the words that were predictive for cybercrime based on the coefficients of the SVM model, see section 2.7. We purposely selected the model with an absolute minimal number of training examples: 3 examples of no cyber and 5 examples of cyber, to understand why this still yields a high recall (the model of size 0.001 shown in Table 16). We looked both into the model with normal training compare to the model trained with oversampling.

Table 17. Most influential features for cybercrime (in Dutch), using just 5 training examples.

Normal training
computer, virus, kaarten, bestanden, primera, account, kreeg, systeem, software, bitcoins, mld, toilettas, codes, rekening, stagiair, pop ups, bitcoin, microsoft, persoon, mld geeft, pop, cybercrime, hoofdkantoor, code, tune, ing bank, bedrag, versleuteling bestanden, overgemaakt, afgeschreven, bank, geld, view, invoeren, gepoogd.

Oversampling
bestanden, mail, phishing, stroomstootwapen geadresseerde, aangetroffen stroomstootwapen, vernielde bestanden, geadresseerde standaard, seponeringsbrief gestuurd, standaard seponeringsbrief, seponeringsbrief, pakket aangetroffen, nl pakket, abn amro, amro, ziggo, abn, uur post, geadresseerde, stroomstootwapen, creditcard aangevraagd, link, pincode, site, herstellen, express, amro bank, post nl, hacking, bestanden computer, vrouw, computersysteem, bestanden usb, kpn, site abn, afgeschreven.

Some of the most predictive words for the model with normal training and for oversampling (see Table 17) are definitely related to cyber-related crimes, probably mainly online sales fraud, but the other words seem mainly predictive for SCM code 102 (swindle) and 103 (forgery). The words for non-cyber (not shown) in case of normal sample were really specific for the three examples and not for non-cyber. The words of non-cyber in case of oversampling also contained words like 'fietsen', 'fietsen dagen', 'staan fietsenhok' which might be related to theft of bicycles. It also contains 'computercriminaliteit', a feature which one expects to be positive for cybercrime.

So the non-cyber was predicted because the cyber words were not found (rather than that the non-cyber words were found). Many of the declaration and clarification field of SCM 102 and 103 (also those used for prediction) contain words like computer, (e)mail, account and so on, which explains the high proportion of cyber predicted in those SCM codes and the high recall even when a very small training set is used.

8.5 Conclusion

In the current section we found that the model performance was not improved by including a lemmatisation step. By analysing the learning curve we discovered that a recall around 0.8 can already be achieve with extremely small number of training examples (less than 10). This result was achieved because the model predicts the occurrence of the SCM codes 102 and 103 rather than predicting cybercrime. In the next section we propose a number of measures that hopefully improves the model and solves that issue.

9. Discussion

In the current study we have investigated three issues that need to be addressed before the beta product on cybercrime can be brought to official statistics. The first issue concerned the missingness of the textual data for some of the records. We showed that the missingness as well as the target variable are related to some background variables: SCM code, Declaration and Clarified. These background variables can be used to correct for the missingness, by imputing a value for cybercrime yes/no, as long as SCM codes with high levels of missingness are excluded from the output.

The second issue concerned the bias in the population estimates due to the classification errors made by the machine learning model. We found that this bias is sensitive to small changes in the machine learning model. We have shown in a separate report that this occurs because the phenomenon that we are interested in concerns a small proportion of the potential crimes. CBS can apply bias-correction methods which have been developed to achieve unbiased estimates, see Kloos et al. (2020) and Scholtus and Van Delden (2020). This correction is easiest in practice, when the model performance does not vary with the subpopulation that one aims to publish, which brings us to the third issue.

The third issue refers to the question whether the model performance is comparable for the different crime types. It was shown that this comes down to the question whether the recall of class 0 and 1 is affected by the subpopulations or not. Unfortunately, the recall of class 1 clearly varied with subpopulation. This variation was not reduced by applying retraining strategies. Further inspection strongly suggests that the trained model can easily learn the relation between features (words) and cybercrime for the SCM codes 102 and 103 but less easily so for other SCM codes. At least one of the reasons is that the clarification and declaration texts in SCM code 102 and 103 very frequently use terms like computer, laptop, email and so on, which is far less so in the other codes.

The finding that model performance varies with SCM codes is also relevant when one wants to classify the cyber-related crimes by other classification variables such as by characteristics of the victim and by characteristics of the perpetrators. For instance, one expects to find that women may be more often victim of cyber online abuse than man. That form of cyber-crime is mainly found in the SCM codes 302 and 400, but the current model has a limited performance for those SCM codes (as also holds for other SCM codes).

The current model has a recall of 0.85 for cybercrime, implying that about 15% of cyber-related crimes are not detected by the model. On a yearly total of about 1 million potential crimes and 10% cyber-related crimes, this implies that about 15.000 cases of cybercrime are missed. One could imagine that a limited number of cases is checked manually, for instance cases where the probability to be cybercrime is somewhat below 0.5. However, say that 50% of the checked records is found to be cybercrime, then still 30.000 cases should be checked manually, which is far too much. This implies that the model accuracy has to be improved before the beta product can go to official statistics.

We suggest that the following approaches are worthwhile to investigate in order to improve recall of the predicting cybercrime and to achieve that the model performance is not affected by subpopulation:

- conceptualise the input words before using them as features,
- further tuning of number of features,
- train the model per cyber type, make it binary afterwards,
- disentangle the association between cyber and SCM code,
- test other ML models, possibly combine with rule-based approach,
- restrict the number of crime categories.

Each of these will subsequently be briefly discussed.

Conceptualisation. The texts that are used to predict cybercrime have a lot of word variations that refer to the same aspects of cybercrime and we expect that it would be worthwhile to standardise them before training the model. For instance, the terms Facebook, Instagram, Twitter, LinkedIn, SnapChat and so on could be replaced by a more general term like #SocialMedia (for instance). Likewise, we the words computer(s), laptop(s), tablet(s), smart phone(s) could be replaced by one term, for instance #ElectronicDevice. We propose to derive which aspects ('concepts') are related to different forms of cyber and which descriptive words are synonyms or hyponyms of those concept words. By replacing the actual descriptive words (D-words) by the corresponding concept words (C-words) one could considerably reduce the word variation. This data preparation is also referred to as conceptualisation in Wang et al. (2017). We believe that this is very helpful to accurately predict different forms of cybercrime, also forms that are rarer.

Feature number. We found to our surprise that using lemmatisation hardly changed our results. Maybe that result was obtained, because we had a maximum tf-idf of 0.25

The model that we used so far used had no lemmatisation, a maximum document frequency of 0.25 combined with a high number of features (1 786 312 ngrams). It would be interesting to play a bit more with these settings and look into their combined effect. It would be interesting to test whether a combination of lemmatisation (besides conceptualisation), a lower total number of features (say 1000 or less) and a higher maximum document frequency improves the model performance.

Cyber type. Furthermore, cyber-related crimes concern a rather broad set of crimes (Alkaabi et al., 2011). Tollenaar et al. (2018) show in their tables 4.4. and 4.5 that those cyber types are associated with different words in the texts. Although, we found no clear improvement in model performance by separation the two different main cyber types (section 7) we still believe that it is worthwhile to have a closer look into predicting different forms of cyber. Our results in section 7 might have been obtained because we were confronted with multiple factors that limit the recall at the same time (factors discussed in the current section). One of the manual annotators mentioned that disagreement with respect to the manual labelling also occurred especially for some forms of cybercrime, mainly forms of digital crimes. Therefore, it might be worthwhile to test whether it is easier to predict certain forms of cybercrime.

Confounding. We have seen that the training set is very unbalanced with respect to SCM codes, because many cyber examples come from SCM codes 102, 103 and 202, whereas less examples come from the other codes. Furthermore, we have found that certain forms of cyber are associated with certain SCM codes. For instance, online abuse is concentrated in SCM codes 302, 303 and 400. It is important that the model is really trained on cyber type rather than that it has learned to the SCM code itself. We therefore believe that it is worthwhile to balance the training material such that within each of the SCM codes there are equal amounts of non-cyber and cyber examples. Even further, for the training set there should also be more balancing in types of cyber within each SCM code. This balancing might be achieved with (combinations of) under and oversampling, or by creating artificial examples, also referred to as augmentation.

Algorithm. We so far have restricted ourselves to the SVM algorithm which was used as a beta product. Tollenaar et al. (2018) compared four machine learning models and found that a classifier chain with random forest gave the best results, irrespective of other conditions they tested. A classifier chain method (Reid et al., 2011) is a way to handle multiple classes and as opposed to multiple binary classifications accounts for interactions between the labels. The algorithm estimates a separate model for each label. This is estimated in a sequential process where the other labels are also used. In the first model only one other label is used, in the second model two other labels are used and so on. The order of the labels matter: the label that is predicted most easily is places first in the row. It is might also be useful when a rule-based step is included for instance to select relevant parts of the text rather than the whole text, especially when the input text is very long.

Subpopulations. The full set of 3-digit SCM codes consists of about thirty categories. Some categories suffer from a large proportion of reports for which no texts are available, e.g. SCM code 104, 204, 601 and 604. It might be best to exclude those codes from official statistics. Furthermore, SCM codes vary considerably in estimated proportion of cybercrime. It might be rather difficult to reliably estimate the cyber proportion in SCM codes where cyber is very rare. It might therefore be useful to combine the 3-digit SCM codes based on substantive considerations into a limited set of five to ten groups for instance, for which a reliable publication can be constructed.

Although this paper has concentrated on cybercrime, the issue of publishing output of a text mining model by subpopulations is a more general one. In section 2 we presented methods that can be used to evaluate whether the model performance is affected by subpopulation or not. It would be interesting to investigate whether this approach is indeed also appropriate in for cases where one is interested to publish on subpopulations. We used a deviance test and focused on comparing the recall per class over subpopulations, while Burger and Meertens (2020) propose to compute min-max performance measures. For future research it is interesting to work out which approach to evaluate model performance of subpopulations is most suitable for which kind of situation.

Acknowledgements

We thank Lisanne Jong and Carlijn Verkleij for annotating an additional set of records and we are gratefull to Joep Burger, Lisanne Jong, Carlijn Verkleij and Nick de Wolf for their useful comments an earlier version of this paper.

References

Alkaabi, A., Mohay, G., McCullagh, A., and N. Chantler (2011). Dealing with the problem of cybercrime. Pages 1-18 in Baggili, I. (ed). Digital Forensics and Cyber Crime. Second international ICST Conference, 2020, Abu Dhabi, United Arab Emirates, 0ctober 2010.

Burger, J. and Q. Meertens (2020). The algorithm versus the chimps: On the minima of classier performance metrics. BNAIC/Benelearn, Leiden, 19-20 November 2020.

CBS (2019). Cybercrime achterhalen in aangifte (in Dutch). www.cbs.nl/nl-nl/oner-ons/innovatie/project/cybercrime-achterhalen-in-aangiften.

Daas, P.J.H. and S. van der Doef (2020). Detecting Innovative Companies via their Website. Accepted for publication in the Statistical Journal of the IAOS.

Hooijschuur, E., Delden, A. van, Verkleij, C., Windmeijer, D. and Jong L. (2019) Towards implementing a text mining model to detect cybercrime in police reports. Paper for the CBS advisory board, 17 September 2019. (available upon request).

Kloos, K., Meertens, Q., Scholtus, S. and Karch, J. (2020). Comparing Correction Methods to Reduce Misclassication Bias. Paper presented at the BNAIC/Benelearn conference, Leiden 19 and 20 November 2020.

Lewis, G.A., Bellomo, S. and A. Galyardt (2019). Component Mismatches Are a Critical Bottleneck to Fielding AI-Enabled Systems in the Public Sector. Archiv 1910.06136v1.

Little, R.J.A., and D.B. Rubin (2002). Statistical Analysis with Missing Data (Second Edition). New York, John Wiley and Sons.

Lumley, T. (2019) "survey: analysis of complex survey samples". R package version 3.35-1.

Meertens, Q.A., Delden A. van, Scholtus S. and F.W. Takes (2019). Bias Correction for Predicting Election Outcomes with Social Media Data. 5th International Conference on Computational Social Science IC2S2 July 17-20, 2019, University of Amsterdam, The Netherlands.

O'Connor, B., Balasubramanyan, R., Routledge, B. R., and Smith, N. A. (2010). From tweets to polls: Linking text sentiment to public opinion time series. In Proceedings of the 4th International AAAI Conference on Weblogs and Social Media, pages 122–129.

Reep, C. (2017). Fraude met online handel Antwoorden uit de Veiligheidsmonitor vergeleken met het politieregister. Internal CBS report (in Dutch).

Reep, C. (2014). Slachtoffer geweest? Antwoorden uit de Veiligheidsmonitor vergeleken met politieregister. Internal CBS report (in Dutch).

Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. Machine learning, 85(3), 333-359.

Scholtus, S. and A. van Delden (2020). On the accuracy of estimators based on a binary classifier. CBS discussion paper Feburary 2020.

Talby, D. (2018). Lessons learned turning machine learning models into real products and services. Blog available at: https://www.oreilly.com/ideas/lessons-learned-turning-machine-learning-models-into-real-products-and-services (accessed August 2019)

Tollenaar, N., Rokven, J., Macro, D., Beerthuizen, M. and A. M. van der Laan (2018). Predictieve textmining in politieregistraties: cyber- en gedigitaliseerde criminaliteit. Rapport van het Wetenschappelijk Onderzoek- en Documentatiecentrum, Ministerie van Justitie en Veiligheid.

Van Delden, A., Scholtus, S. and J. Burger (2016), Accuracy of Mixed-Source Statistics as Affected by Classification Errors. Journal of Official Statistics 32, 619–642.

Valliant, R. (2020). Comparing alternative for estimation from nonprobability samples. Journal of Survey Statistics and Methodology 8, 231–263.

Wang, J., Wang, Z., Zhang, D. and J. Yan (2017). Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17) 2915- 2921

Ziegler, A and P. Czyż (2019). Unsupervised Recalibration. Avaliable at Archiv:1908.09157v2.

Appendix 1: Deviance

The deviance is defined as follows. Let $\widehat{\boldsymbol{\theta}}$ be a vector of estimated parameters of a model, let $\widehat{\boldsymbol{\theta}}_s$ be the parameters of a saturated model and $\widehat{\boldsymbol{\theta}}_0$ be the parameters of a model 0 that one likes to test. Let $P(\boldsymbol{v}|\widehat{\boldsymbol{\theta}})$ be the probability to observe the vector of values $\boldsymbol{v}$ given the estimated model parameters $\widehat{\boldsymbol{\theta}}$. The deviance of some predictions $\widehat{\boldsymbol{v}}$ for $\boldsymbol{v}$ are now given by:

$$D(\widehat{\boldsymbol{v}},\ \boldsymbol{v}) = 2\log\{P(\boldsymbol{v}|\widehat{\boldsymbol{\theta}}_s)\} - 2\log\{P(\boldsymbol{v}|\widehat{\boldsymbol{\theta}}_0)\}.$$

The saturate model will have a perfect fit which leads to a log likelihood of 1. Large values of the deviance indicate a poor model fit, thus that there is a small probability model 0 is equally likely as the saturated model.

More generally one can also compare the deviance of a small model 0 with that of a larger model 1:

$$D(\widehat{\boldsymbol{v}}_1, \widehat{\boldsymbol{v}}_0) = 2\log\{P(\boldsymbol{v}|\widehat{\boldsymbol{\theta}}_1)\} - 2\log\{P(\boldsymbol{v}|\widehat{\boldsymbol{\theta}}_0)\}$$

Again, large values indicate that there is a small probability that model smaller model 0 fits a good as the larger model 1.

The deviance has a chi-squared distribution with as degrees of freedom the difference in the residuals of model 1 minus that of model 0. We now apply a one-sided test of the probability that the deviance is at most as large as $D(\widehat{\boldsymbol{v}}_1, \widehat{\boldsymbol{v}}_0) =$

The p-value is then given as

$$P(D(\widehat{\boldsymbol{v}}_1, \widehat{\boldsymbol{v}}_0) = 1 - \text{chisq.cdf}\{D(\widehat{\boldsymbol{v}}_1, \widehat{\boldsymbol{v}}_0), df\}$$

Appendix 2: 3-digit SCM codes

Table 18. Completeness of declaration and clarification fields in 2016. $^{(a)}$

SCM	Description	# records	Decl. field (%)	Clar. field (%)
101	Theft, misappropriation and burglary	506027	94	95
102	Swindle	46420	98	97
103	Forgery crimes: coins, stamps, writing	26605	83	84
105	Healing	2085	67	75
104	Extortion	9595	9.2	57
106	Bankruptcy	177	36	83
107	Money laundering	1068	3.5	54
108	Property crime	0	0	0
201	Destruction and damage	99407	94	93
202	Public order and safety offence	13604	65	78
203	Arson	5058	83	88
204	Public authority offence	7832	8.3	48
301	Abuse	51102	78	85
302	Threat and stalking	32862	76	79
303	Sexual offence	8800	28	86
304	Life crime	3171	74	70
305	Deprivation of freedom, hostage-taking	688	41	61
306	Human trafficking	680	11	78
307	Violent crime, other	1249	85	75
400	Other crimes WvSr, such as insult	14256	73	71
501	Leave the scene of an accident	79880	97	97
502	Drunk driving	27343	0.58	93
503	Driving when licence has been suspended, unauthorized driving	5601	0.70	89
504	Driving during driving ban	214	0.47	72
505	Having a false license plate	903	37	74
506	Joyriding	340	63	69
507	Refusal breath analyser, blood test	687	0.15	64
508	Traffic offence	4757	1.6	88
601	Hard drugs	8232	0.91	66
602	Soft drugs	9436	3.3	81
603	Import and export of drugs, other drug offenses	287	9.8	75
700	Gun crimes	7143	1.6	66
901	Military crimes	0	0	0
902	Crimes other: waste, transport substances, flying, navigating	4239	29	74
909	Unknown (NA = not available, not counted)	NA		

(a) The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

Explanation of symbols

Empty cell	Figure not applicable
.	Figure is unknown, insufficiently reliable or confidential
*	Provisional figure
**	Revised provisional figure
2017–2018	2017 to 2018 inclusive
2017/2018	Average for 2017 to 2018 inclusive
2017/’18	Crop year, financial year, school year, etc., beginning in 2017 and ending in 2018
2013/’14–2017/’18	Crop year, financial year, etc., 2015/’16 to 2017/’18 inclusive

Due to rounding, some totals may not correspond to the sum of the separate figures. Selections of the BVH data are also used by CBS to make (yearly) output on crime frequencies by SCM codes, resulting in official CBS figures. However, the selections made by CBS for those official figures differ on some points from the selections made in the current report, therefore differences may occur. The current report is not intended as a publication with official numbers on cybercrime, but concerns a research report to test methods to derive cybercrime from police records.

Colophon

Publisher
Centraal Bureau voor de Statistiek
Henri Faasdreef 312, 2492 JP Den Haag
www.cbs.nl

Prepress
Statistics Netherlands, CCN Creation and visualisation

Design
Edenspiekermann

Information
Telephone +31 88 570 70 70, fax +31 70 337 59 94
Via contactform: www.cbs.nl/information