Ludwig-Maximilians-Universität München

Institut für Statistik

Automating Survey Coding for Occupation

Master's Thesis

Author:
Malte Schierholz

Advisor:
Prof. Dr. Frauke Kreuter

April 2014

Abstract

Currently, most surveys ask for occupation with open-ended questions. The verbatim responses are coded afterwards into a classification with hundreds of categories and thousands of jobs, which is an error-prone, time-consuming, and costly task. Research related to the coding of occupations is summarized with an international literature review. Special attention is paid to our main topic, the automation of coding.

A prominent approach for automated coding is to consult a dictionary on the correct code. In contrast, we focus on data-based methods where codes for new answers are predicted from those answers that are already coded. Four different coding methods are tested on two data sets: (1) Rule-based Coding that consults a dictionary, (2) data-based Naive Bayes that allows coding for text answers with multiple words, (3) data-based Bayesian Categorical is used to improve performance when relatively few answers were coded before, and (4) Combined Methods (Boosting) combining predictions from the first three methods.

The proposed Bayesian Categorical model is able to code 38% of all answers at 3% error rate without human interaction. In all remaining cases or for higher quality human intellect is needed to decide on the correct code and computer software can only assist by suggesting possible job codes. With the prototype software we developed for this task, we expect that for 74% of all answers the correct category is provided within the top five code suggestions. The training data used for prediction consists of only 32882 coded answers which is small compared to other systems with similar purpose. The proportions given above are expected to improve with additional training data.

Contents

1 Introduction **1**

2 Background **3**

2.1 Coding Examples 3

2.2 Code Structures and Classifications 4

2.2.1 German Classification of Occupations 2010 5

2.3 Coding Options: Manual or Automatic 10

2.4 Techniques for Automated Coding 11

2.4.1 Rule-Based Coding 12

2.4.2 Data-Based Coding with Supervised Learning Techniques 13

2.5 Coding Evaluation 14

2.5.1 Quality of Occupation Coding 17

3 Data Analysis **20**

3.1 Description of Survey Data 20

3.1.1 Job Codes 23

3.2 Methods for Automated Coding 25

3.2.1 Rule-based Coding 30

3.2.2 Naive Bayes 31

3.2.3 Bayesian Categorical 37

3.2.4 Combined Methods (Boosting) 42

4 A Prototype for Computer-Assisted Coding **50**

5 Conclusion and Perspectives **53**

Bibliography **55**

A Diagrams **63**

B Exemplary Job Category Suggestions **65**

Chapter 1

Introduction

Surveys are a well-established instrument to collect information about society, living conditions, people's background, or their environment. Most survey questions are written in a closed format and respondents mark the best-fitting category. An example for this is the question about sex with two standard categories "male" and "female", and a third category "indeterminate", which is most adequate for intersex people, often missing. The definition of answer categories prior to field measurements may be problematic or the sheer number of possible categories prohibits the use of a closed question. An alternative is then to ask open-ended questions and record the exact verbatim answer given from the respondent. For statistical analysis it is necessary to assign these answers to categories. While this is done by the respondents themselves for closed questions, this task is laborious for open-ended questions. Traditionally, interviewers or clerks, sometimes called "coders", have been employed to do this time-consuming coding job.

The application of open-ended questions is tempting for social scientists. There is no need to define answer categories, and respondents are not influenced from predefined categories. Nevertheless, closed questions are often preferred to circumvent high coding costs (cf. Reja et al. (2003)). Closely related to survey coding is content analysis which "has been defined as a systematic, replicable technique for compressing many words of text into fewer content categories based on explicit rules of coding" (Stemler, 2001). Its aim is more general than survey coding in the sense that whole documents instead of respondent's answers need to be classified into categories. The problem is still the same. For the large number of documents to be categorized it would be helpful to cut costs with automated coding methods. Scharkow (2012) applies machine learning methods for automated content analysis.

Our goal is to facilitate survey coding using machine learning methods. The idea is to use answers that were coded before to predict correct codes for new answers. These methods are continuously and successful applied for survey coding in other countries whereas the German coding praxis is lagging behind.

In our study we focus on the coding of employments. The same problem was addressed internationally multiple times but as we are concerned with German employments, we need to account for some special characteristics. The current official German employment classification consists of 1286 well-defined job categories, more than in most other countries. For a detailed ascertainment and coding into the correct category, German surveys often ask not one but two or three open-ended questions on the employment. The adaption of automatic prediction methods to the German environment is further complicated by the fact that our training data consists of only 32882 job records, far less than what is available in comparable systems. With the proposed methods for automated occupation coding we try to address this problem of limited training data. Though, better performance will require additional training data.

This thesis is organized as follows. In section 2 we provide a literature review and theoretical considerations about automated coding. Special attention is paid to employment coding and quality control. The main part of our work is in section 3. Four different techniques for automated coding are described. The performance from these prediction methods is tested on two data sets that have employments already coded. We believe most helpful for German employment coding will be a computer system that suggests possible job categories to human coders who decide which category is correct. A prototype for this is described in section 4.

Chapter 2

Background

2.1 Coding Examples

A multiplicity of different response types requires coding. Hacking and Willenborg (2012), for example, list multiple variables that are coded at Statistics Netherlands: Education, occupation, articles, shops, industrial sector, job vacancy, important political problems, and causes of death. Groves et al. (2009) names some frequently used classification systems where coding is necessary: The *Standard Occupational Classification*, the *North American Industry Classification System*, the *International Classification of Diseases*, and the *Diagnostic and Statistical Manual for Mental Disorders*. For most of the examples above, coding is necessary, because the high number of target categories makes it impractical to present all of them to the respondent. For other variables, including the following, the researcher wants to avoid that respondents are influenced from predefined answer categories.

DeBell (2013) describes open-ended question coding on the American National Election Studies (ANES) asking "'what job or political office' is now held by various prominent officials". The answers have been coded into four categories to distinguish misinformed, uninformed, partially informed, and fully informed respondents. The author laments on problematic coding practices that "support only the simplest and grossest inferences" (quote from Gibson and Caldeira (2009)) and develops new coding rules to counter this "data analysis crisis".

Esuli and Sebastiani (2010) apply coding to multiple market research problems. They give an exemplary question "What is your favourite soft drink?" that respondents answer typically with a product or brand name.

Groves et al. (2009) point out that coding is not only done for textual responses. There is more nonnumeric data collected in surveys that needs a numeric value assigned, e.g., visual images, sounds, soil or blood samples, or geographical data (respondent's position) to be coded into some geographic unit.

2.2 Code Structures and Classifications

Coding is the process of transforming nonnumeric material to a numeric code. Groves et al. (2009) call it both "an act of translation and an act of summarization". When there exists an one-to-one mapping between the original material and the target code, the mapping can be carried out without problems. When, however, "frameworks are mismatched, the translation task can be complex and subject to error". For the summarization part, someone has to decide "whether two verbal representations are equivalent" and what the "level of summarization" should be. Taken together, the code structure is central to the problem of coding and shall be described in more detail here.

For the construction of new code structures, Groves et al. (2009) give some general rules for codes to be useful:

1. "A unique number, used later for statistical computing
2. A text label, designed to describe all the answers assigned to the category
3. Total exhaustive treatment of answers (all responses should be able to be assigned to a category)
4. Mutual exclusivity (no single response should be assignable to more than one category)
5. A number of unique categories that fit the purposes of the analyst". It is suggested that "each of the code categories should link to different parts of key hypothesis". For example, employment could be coded by "supervisory status" to "separate supervisors from nonsupervisors", or by educational background required for a job.

Regarding the points 3) and 4) it is almost always the case that some answers do not fall into predefined codes. Groves et al. (2009) therefore suggest to test and refine the code structure on the basis of previously collected responses. Further, "coding structures must be designed to handle all responses, even those judged as uninformative". It is therefore recommended to include further categories for respondents that did not give an answer or for cases when it is not possible to ascertain the correct code.

As described, in a perfect research world one would setup and test a code structure according to the research hypothesis. This is in contrast to coding when the code structure is an official classification. Hacking and Willenborg (2012) point out that classifications often have been constructed from a theoretical perspective and without actual usage in mind. Typically, a classification cannot be changed easily as (possibly international) committees are responsible for their maintenance. Furthermore, when changes are made, this is often done "with regards to the subject matter itself and not with observation/measurement in mind".

Because a given classification cannot be changed, the coding practice needs to cope with arising problems. Hacking and Willenborg (2012) point out the following difficulties:

- "The categories cannot clearly be distinguished;
- The categories are rare in the population;
- There is not very much empirical material available to describe the categories, or the empirical information is not sufficiently diverse;
- There are categories that are close together, and therefore it is difficult to distinguish between them;
- The categories are very clearly defined and also occur in practice, but they are not actually used in practice because nobody uses the associated distinction."

Hacking and Willenborg (2012) describe further principles common for classifications that can be useful for coding: The structure of a classification is a tree. In this mathematical concept, the leaves are the most specific categories having more general parent categories. So-called "classifying principles" are used to distinguish more specialized categories from each other. We will exemplify the tree structure together with classifying principles in the next section related to the German Classification of Occupations.

2.2.1 German Classification of Occupations 2010

To classify occupations, the *International Standard Classification of Occupations 2008 (ISCO-08)* is widely used. Additionally, many countries have their own national classifications. As this study is concerned with German occupations, we will follow the work from Hartmann and Schütz (2002), TNS Infratest Sozialforschung (2012) and Paulus and Matthes (2013) and use the German national classification for coding. This section describes this classification and how it is connected to other classifications in more detail.

Until 2010, two German national classifications have been used, one published by the Federal Employment Agency ("BA") in 1988 (*Klassifikation der Berufe 1988*, KldB 1988) and the other one published by the Federal Statistical Office in 1992 (*Klassifikation der Berufe 1992*, KldB 1992). As both classifications have a common origin in theoretical work from the 1960s, they were outdated and replaced by the German *Klassifikation der Berufe 2010* (KldB 2010). This classification was developed with two main goals: Special characteristics of the German labor market were taken into account while at the same time a high degree of compatibility with the international ISCO-08 was obtained (Bundesagentur für Arbeit, 2011).

The KldB 2010 is a hierarchical classification (graph theory would call it a tree) with five levels where the ten top-level categories (*Berufsbereiche*) are the most general and the

fifth level with 1286 categories (*Berufsgattungen*) is the most specific. Figure 2.1 is a small extract from the classification to be read as follows. The *Berufsgattung* "Berufe in der Landwirtschaft (ohne Spezialisierung) - Helfer-/Anlerntätigkeiten" contains multiple jobs. This *Berufsgattung* itself is included in the *Berufsuntergruppe* "Berufe in der Landwirtschaft (ohne Spezialisierung)" which itself is part of the *Berufsgruppe* "Landwirtschaft" and its parent categories. The code numbers reflect these relations in the sense that the first digit specifies the most general *Berufsbereich*, the first two digits give the more specific *Berufshauptgruppe* and so on. Figure 2.2 provides the number of categories at each level in the classification.

The KldB 2010 is structured by two dimensions ("classifying principles"): professional specialisation ("Berufsfachlichkeit") and the skill level ("Anforderungsniveau"). The first four digits are used to group occupations by professional specialisation. Based on the capabilities, skills, and knowledge required for a job, a cluster analysis was performed to group jobs with a higher degree of similarity into the same category. The clustered results were reviewed multiple times by specialists and so the 2-, 3-, and 4-digit categories may be used for comparisons. The last digit allows for different degrees of complexity within occupations, i.e. the skill level. Each *Berufsuntergruppe* (4-digits) combines up to four *Berufsgattungen* (5-digits): (1) Auxiliary and semiskilled occupations, (2) specialized occupations, (3) complex occupations for specialists, and (4) highly complex occupations. These *Berufsgattungen* are mainly defined by the duration of formal vocational education. Figure 2.3 illustrates similar occupations with different skill levels (Paulus and Matthes, 2013).

There exist, however, some exceptions to these general classifying principles (Paulus and Matthes (2013), Bundesagentur für Arbeit (2011)):

- The *Berufsuntergruppe* (4-digits) has an indicator function: If the fourth digit is "0", the corresponding employments cover various duties without further specialization. Typically, this applies to auxiliary occupations. An "8" at the fourth digit is used for employments with a specific focus that do not suit into the other defined *Berufsuntergruppen*.

- To identify all supervisors and managers uniquely within a specific *Berufsgruppe* (3-digits), these are grouped together in a *Berufsuntergruppe* labeled with a "9" at the fourth digit. Managers are assumed to have highly complex occupations and are hence given a "4" in the fifth digit. Supervisors, in particular the German "Meister", typically work in less complex occupations and therefore get a "3" in the last digit.

- For occupations in the one-digit Berufsbereich for military, the KldB 2010 groups occupations only into four Berufsgattungen: 01104 for officers, 01203 for high-ranked sergeants, 01302 for low-ranked sergeants, and 01402 for privates.

Extract from the Classification of Occupations 2010 (KldB 2010)
1 Land-, Forst- und Tierwirtschaft und Gartenbau
11 Land- Tier- und Forstwirtschaftsberufe
111 Landwirtschaft
1110 Berufe in der Landwirtschaft (ohne Spezialisierung)
11101 Berufe in der Landwirtschaft (ohne Spezialisierung) - Helfer-/Anlerntätigkeiten
11102 Berufe in der Landwirtschaft (ohne Spezialisierung) - fachlich ausgerichtete Tätigkeiten
11103 Berufe in der Landwirtschaft (ohne Spezialisierung) - komplexe Spezialistentätigkeiten
11104 Berufe in der Landwirtschaft (ohne Spezialisierung) - hoch komplexe Tätigkeiten
1111 Berufe in der Landtechnik (contains 2 Berufsgattungen)
1112 Landwirtschaftliche Sachverständige (contains 2 Berufsgattungen)
1113 Berufe im landwirtschaftlich-technischen Laboratorium (contains 2 Berufsgattungen)
1118 Berufe in der Landwirtschaft (sonstige spezifische Tätigkeitsangabe) (contains 3 Berufsgattungen)
1119 Aufsichts- und Führungskräfte - Landwirtschaft (contains 2 Berufsgattungen)
112 Tierwirtschaft (contains 5 Berufsuntergruppen)
113 Pferdewirtschaft (contains 6 Berufsuntergruppen)
114 Fischwirtschaft (contains 4 Berufsuntergruppen)
115 Tierpflege (contains 5 Berufsuntergruppen)
116 Weinbau (contains 2 Berufsuntergruppen)
117 Forst- und Jagdwirtschaft, Landschaftspflege (contains 5 Berufsuntergruppen)
12 Gartenbauberufe und Floristik
121 Gartenbau (contains 6 Berufsuntergruppen)
122 Floristik (contains 2 Berufsuntergruppen)
2 Rohstoffgewinnung, Produktion und Fertigung (contains 8 Berufshauptgruppen)
⋮
0 Militär
01 Angehörige der regulären Streitkräfte
011 Offiziere
0110 Offiziere
01104 Offiziere - Hoch komplexe Tätigkeiten
012 Unteroffiziere mit Portepee
0120 Unteroffiziere mit Portepee
01203 Unteroffiziere mit Portepee - Komplexe Spezialistentätigkeiten
013 Unteroffiziere ohne Portepee
0130 Unteroffiziere ohne Portepee
01302 Unteroffiziere ohne Portepee - Fachlich ausgerichtete Tätigkeiten
014 Angehörige der regulären Streitkräfte in sonstigen Rängen
0140 Angehörige der regulären Streitkräfte in sonstigen Rängen
01402 Angehörige der regulären Streitkräfte in sonstigen Rängen - Fachlich ausgerichtete Tätigkeiten

Figure 2.1: Extract from the Classification of Occupations 2010 (KldB 2010)

10 Berufsbereiche (one-digit) 37 Berufshauptgruppen (two-digits) 144 Berufsgruppen (three-digits) 700 Berufsuntergruppen (four-digits) 1286 Berufsgattungen (five-digits)

Figure 2.2: Number of Categories in the KldB 2010

Skill Level	Assigned Occupations → 5-digits from KldB 2010
1: Helfer-/Anlerntätigkeiten	Gesundheits- und Krankenpflegehelfer/in → 8130**1**
2: fachlich ausgerichtete Tätigkeiten	Gesundheits- und Krankenpfleger/in → 8130**2**
3: komplexe Spezialistentätigkeiten	Fachkrankenschwester-/pfleger → 8131**3**
4: hoch komplexe Tätigkeiten	Allgemeinarzt/-ärztin → 8140**4**

Figure 2.3: Berufsgattungen (5-digits) in Health and Patient Care (taken from Paulus and Matthes (2013))

For its job placement activities, the Federal Employment Agency uses the so-called *Dokumentationskennziffer* (DKZ) which is derived from the KldB 2010. The DKZ-database is continuously updated and contains all occupation and vocational training names used currently in Germany together with further occupation-specific information. The DKZ is an eight-digit number where the first five digits are identical to the KldB 2010. The last three digits specify one particular occupation (as opposed to occupation categories in the KldB). The sixth digit is used to distinguish between occupations (digit equals "1" or "2") and vocational trainings (digit equals "8" or "9").

With ISCO-08, KldB 2010, and the DKZ, three different classifications are available for the coding of occupation. The DKZ is the most detailed and the other classifications can be derived from it. When the last three digits are truncated, one obtains the KldB 2010. For international studies, the ISCO-08 classification is often used. As the KldB 2010 was developed to be compatible with ISCO-08, the transition from KldB 2010 to ISCO-08 can be done using a transition table. For 90% of the KldB-categories, there exists exactly one corresponding category in ISCO-08, otherwise more than one. Other studies are concerned with the social position, socio-economic status, or job prestige. Nearly all common measures for it (e.g., class scheme of Erikson, Goldthorpe and Portocarero (EGP), European Socio-economic Classification (ESeC), Magnitude Prestige Scale (MPS), Standard International Occupational Prestige Scale (SIOPS), or International Socio-Economic Index (ISEI)) are based on ISCO-coded occupations (Paulus and Matthes, 2013).

Not only is the DKZ the most detailed classification for German occupations but a large part of the DKZ-database is also available online. Paulus and Matthes (2013) therefore recommend using the published resources from the DKZ for automatic and computer-assisted coding. We will discuss and use the different resources in section 3.2. Despite all the advantages from the DKZ, it is only a supplementary tool for coding into official classifications.

Who	Advantages	Disadvantages
Respondent	• direct feedback	• no knowledge of the classification
Interviewer	• direct feedback	• superficial knowledge of the classification
Professional Coder	• expert in the classification • can also use extra information that was included • in general, can interpret answers better than a computer program	• direct feedback not always possible • feedback is very time-consuming • coding may be inconsistent
Computer Program	• fast, consistent coding • coding knowledge is specified in a system and is therefore transferrable • can operate day and night	• no direct feedback • only the relatively simple cases are coded (but that is often the bulk)

Figure 2.4: Possible Places for coding (table taken from Hacking and Willenborg (2012))

For a number of reasons it is not a suitable target classification in itself. Because it is updated daily, it may happen that the correct category for an answer changes over night. Also, the DKZ is not just the 6-th level in the KldB but a full hierarchy with multiple levels and thus it can happen that a specific and a more general DKZ code both are correct. A file with 3920 DKZ codes is available for download and another, overlapping set with 3098 DKZ codes is used for the BERUFENET[1] online. Taken together this means that the DKZ is not a stable classification where all categories are well defined. We will therefore use the 5-digit KldB 2010-*Berufsgattungen* for the coding of occupations in this work.

2.3 Coding Options: Manual or Automatic

In principle, different kinds of coding systems exist: *manual coding*, *computer-assisted coding*, and *automatic coding*. It depends on the complexity of the coding task which option is best and combinations of these systems are typically used in practice. This section and figure 2.4 explore the different options in more detail.

For most survey questions, coding is done implicitly by the respondent. That is, after a closed question was asked, the respondent indicates the most adequate category. Sometimes, - and this is the case for occupations - the coding scheme contains too many categories or is too complex for the respondent. In these cases, an open-ended question is asked and the textual answer is categorized by a professional coder. With this method, relevant details may be missing when the coder does his work. As a resort, it has been suggested to give the coding task to the interviewer who can inquire all necessary information during the interview (cf. Conrad (1997), Hacking and Willenborg (2012)).

Computer-assisted coding is used to facilitate the coding task with specially designed computer programs. While the decision which category is correct remains with the human coder, the coding program offers help and often suggests a small number of adequate categories. For occupations, Bushnell (1998) has shown that a computer program may accelerate the coding process and increase coding quality at the same time. A specialized software for this task is the Cascot-program[2] for occupation coding in the United Kingdom. More generally, the integration of open-ended questions into surveys is a longstanding methodological concern and both Fielding et al. (2013) and Esuli and Sebastiani (2010) describe various software solutions available for the analysis and coding of verbatim answers.

In contrast to computer-assisted coding, in automatic coding the computer automatically assigns one target category. According to Lyberg and Kasprzyk (1997), proportions as high as 70%-80% may be coded this way while maintaining low error rates. When automatically assigned codes are expected to be incorrect, these residual cases are conferred to an expert for the final classification. Around the world, statistical agencies have developed programs for automatic coding and reported satisfying results as well as cost-savings (cf. United Nations Statistical Commission and Economic Commission for Europe (1997)).

When it is not relevant to distinguish between computer-assisted coding and automatic coding, we will use the term *automated coding* that can mean both. The next section will give an overview over the techniques used for automated coding.

Technique	**Description or Example (Original String → Parsed String)**
Replacement of Symbols	'+' → 'plus'
Non-standard Character Replacement	'#' → ' ' (space)
Replacement of Abbreviations	'Prof.' → 'professor'
Substitution of Letters	'à' → 'a'
Removing Stop Words	'the' → ' ' (space)
Splitting Composite Words	'machinefabrieksopzichter' → 'machine fabriek s opzichter'
Spell Checking	Spell checkers or fuzzy matching methods (e.g., trigrams, Levenshtein distance, Soundex)
Phrasing	Split text into different phrases that will be coded separately
Tokenizing	Split phrases into single words or n-grams (e.g. 'machine' → {' 'ma, mac, ach, chi, hin, ine, ne' '})
Lemmatization / Stemming	'mine' → 'his' / 'fished' → 'fish'
Replace Synonyms, Loan Words and Hypernyms	'account manager' → *Dutch equivalent*, 'tomato' → 'greenhouse vegetables'
Word-Sense disambiguation	'bank' → 'banking institution' (based on context)

Figure 2.5: Preprocessing for Texts (summary from Hacking and Willenborg (2012))

2.4 Techniques for Automated Coding

In order to automatically code a textual answer, most methods rely on a dictionary containing this answer or other answers with similar meaning together with the corresponding code. In section 2.4.1, we will give an overview over systems with rule-based coding. All described systems have in common that expert knowledge is required to set up these systems. In contrast, the data-based coding techniques summarized in section 2.4.2 use material that was coded previously by human coders. If a sufficient number of equal or similar answers (this is quite similar to a dictionary) is already coded into only one category, chances are good that the current answer may also fall into the same category.

Human language exhibits a high degree of variety, e.g., spelling errors, grammatical forms, slang language, and synonyms. Both the expert and the data-based methods perform better when textual answers and entries from the dictionary can be matched to each other. Therefore, a number of functions may be used to bring these texts into a standardized form that simplifies textual comparison. Figure 2.5 contains a number of textual preprocessing techniques that have been suggested for this task in the context of automated coding.

[1] http://berufenet.arbeitsagentur.de/berufe/

[2] Online available at http://www2.warwick.ac.uk/fac/soc/ier/software/cascot/

2.4.1 Rule-Based Coding

The simplest approach to automated coding uses logical rules: Under exactly specified conditions a code is assigned. For example, when a (preprocessed) answer is identical to a given string, the corresponding code is assigned. For occupation, various authors have described this technique (e.g., Geis (2011), Drasch et al. (2012), Jung et al. (2008), Conrad (1997)) and use it as a first step. Although a few 1000 rules exist in these systems, it is rare to code more than 50% of the occupation codes accurately. Hartmann and Schütz (2002) have generated additional rules for higher production rates and describe the arising problems.

Closely related is an approach based on dictionaries that associate each entry with exactly one code. Some expressions in the dictionary may appear multiple times with different codes. Now, the coding task is to match a given answer in standardized form (i.e. preprocessed with methods described in figure 2.5) to one or more entries in the dictionary. Exact matches are not needed but the match must be close enough to exclude any ambiguity. When only one match is found, the associated code is assigned. Multiple matches may be resolved manually or automatically with the help of weighting algorithms that make use of how specific associations between expressions and particular codes are. Conrad (1997) gives the main concepts in greater detail and describes the historical development at the US Census Bureau. Different statistical agencies around the world have used this approach (see United Nations Statistical Commission and Economic Commission for Europe (1997)).

One of these systems is *G-Code* (old name *ACTR*) which has been under development by Statistics Canada for more than 20 years. It is a generalized coding software in the sense that it can be used for different languages and coding tasks. Its particular strength are sophisticated text processing functions that transform natural language answers with equivalent meaning into a standardized form that can be looked up in a dictionary. Good performance results have been reported for Canada (Tourigny and Moloney, 1997) and Italy (Ferrillo et al., 2008). Research related to this software has been published by Gillman and Appel (1994) and Macchia et al. (2010).

Another idea is to exploit the linguistic relation between textual answers and the target category description. Textual answers and target categories may be represented in the same vector space. Then, one assigns the category which is most similar (cosine similarity) to the textual response. This technique from Information Retrieval is described in Manning et al. (2008). Jung et al. (2008) and Viechnicki (1998) find that this similarity-based approach is outperformed by dictionary-based and multinomial regression methods.

A recent approach to utilize the linguistic relation between textual response and category description has been described by Sangameshwar and Palshikar (2013). A promising feature in their prototype is that it searches for synonyms and related words from a public database. The use of such semantic relationships has shown to be useful for coding

(e.g., Jung et al. (2008), Hacking and Willenborg (2012)). Willenborg (2012) describes the underlying concepts in detail.

Most methods described so far suffer from one drawback: Substantial background knowledge or human supervision is needed to set up the software. Lyberg and Kasprzyk (1997) observe that coding rules are suboptimal when they are only based on expert descriptions. The software is much more efficient when the "empirical pattern generated by respondents themselves" is used to create the dictionary. A similar view is expressed by Giorgetti and Sebastiani (2003) who come to the conclusion that supervised learning methods may outperform traditional methods. The next section will give an overview over these approaches.

2.4.2 Data-Based Coding with Supervised Learning Techniques

The automated classification of texts into predefined categories is well-studied in the field of machine learning (e.g., Aggarwal and Zhai (2012), Sebastiani (2002)). The task is to learn from training data, i.e. existing text documents are already grouped into categories, and use this data to predict the correct category for additional texts. Some algorithms allow classification into hierarchically structured target categories (e.g., Esuli et al. (2008)), which appears useful for automated coding. Typically, text classification techniques are designed to classify whole documents with multiple words into a small number of categories.

The survey coding task is more challenging. Although it is theoretically equivalent to the classification of text, practical aspects differ. In surveys, the respondents typically answer with only a few words and the number of possible categories may be very large. Text classification has nonetheless been applied to the field of survey coding. Esuli and Sebastiani (2010) describe automatic coding software designed to classify short survey answers into classifications with only two categories.

In the following, we will give some examples from working systems that code occupations automatically. Compared to other text classification algorithms, ideas are simple and training data should be large:

- The US Census Bureau has been experimented with "nearest neighbor and fuzzy search techniques" (Gillman and Appel, 1994) and neural networks (Conrad, 1997) for the coding of occupations. Current practice is still dictionary-based (see above). Multiple dictionaries are created automatically from training data, one dictionary for single word entries, another dictionary for two-word long entries, and a third dictionary for whole answer texts. To include an entry in the dictionary, it needs to appear multiple times in the training data and a strong association to a specific code is required (Thompson et al., 2012).

- Hacking and Willenborg (2012) use how close words W correspond with particular categories C_i. If a particular word falls mostly in one or a few categories ("lawyer" in

contrast to "employee"), a higher specificity score

$$F(W) = \frac{\sqrt{\sum_{i=1}^{n} P(C_i|W)^2}}{n}$$

is calculated, where n is the number of categories which had the word W assigned. If more than one word from the current verbatim match with some answer from the training set, specificity scores for these words are added up. If the similarity is higher than a certain threshold, the corresponding code is assigned.

- Jung et al. (2008) use training data to learn a maximum entropy model that estimates the conditional probability $p(Code_i|Textualanswer)$. To this end, it was necessary to build a large domain specific thesaurus that reduces the number of possible textual answers.

2.5 Coding Evaluation

Finding a single correct category for a given answer is not always possible. Textual answers may be very general (e.g., "Angestellter" / "clerk") and allow coding into multiple similar categories. For example, "call-center telephonist" may be coded into both categories "call-center agent" and "telephonist" (examples come from Hartmann and Schütz (2002) resp. Drasch et al. (2012)). Campanelli et al. (1997) state that coding quality "can be seen to depend on a number of factors, such as the type of question, the nature of the answers, the length and adequacy of the coding frame, and the training and supervision of coders." The same authors summarize the following definitions to measure coding quality:

- *Reliability* is the proportion of agreement between two different coders. It ranges from the worst case 0, if coders always assign different codes to the same answer, to 1, if coders completely agree for all answers. For supervision of individual coders, it may be useful to calculate the reliability for each coder separately. It is also possible to study the reliability of individual codes to find inherent weaknesses in a given code frame.

- Two coders might assign the same code not by a shared understanding but by chance. An estimator, *Cohen's Kappa* is proposed for adjustment. For the large KldB-coding frame at hand, however, it is highly improbable to assign a correct code by chance and therefore we will not use Kappa.

- When coding reliability is low, derived estimators, such as the population share with a specific characteristic, have increased variance. Given a measurement model, the

variance is increased by the *variance inflation factor*

$$Eff = (1 + \rho_c(M - 1)(1 - \kappa_i))$$

where ρ_c measures systematic biases in the coding process, M gives the average coding workload, and κ_i is the reliability of the individual code. Based on this formula it is argued that more coders with less workload for each might reduce the variance of estimators.

- It is desired to see if coders assign the "right" code to textual answers, i.e. the code that corresponds best to the described occupation. This concept of *validity* is, however, hard to operationalize. A possibly ideal criterion would be to send an expert team to observe and consult the respondent and have the occupation coded afterwards. As this is not achievable, some classification experts might be asked for a "correct" coding given the textual answers. The coder's work can then be compared to this expert work.

To measure the performance in automatic coding, further measures for quality and efficiency are common in the literature:

- The *agreement rate* (or its inverse, the *error rate*) is the proportion of automatically generated codes that agree with a manual-assigned code. Hereby it is assumed that the manual-assigned code is the correct code. Some systems also account for erroneous codes from manual coding (e.g., Tourigny and Moloney (1997), Thompson et al. (2012), Svensson (2012)). Often it is required that the automatic coding system performs as good as professional coders.

- The *coding* or *production rate* is the proportion of codes that can be generated automatically. With a higher production rate, fewer text answers are presented to professionals for manual coding making the coding process less expensive.

- *Speed* considerations are sometimes made. As some automatic coding techniques are computationally intensive, one might observe the time needed for model training or for prediction.

It is relevant to note that there is a trade-off between agreement rate and production rate: There are always some answers that are hard to code automatically and should be left to specialist coders. This will decrease the production rate but increase the agreement rate. Predicting which codes will be correct is therefore an important task that was studied by Chen et al. (1993) and Kaptein (2005). The following is an example from Thompson et al. (2012) that describes actual usage at the U.S. Census Bureau. A logit model with 79 independent variables is used to calculate the probability PHAT for an automatically

assigned code to be correct. Only when PHAT exceeds a fixed score cutoff, the answer is coded automatically. The score cutoffs were set such that automatic coding with PHAT = score cutoff is expected to perform as accurate as 100% manual coding. With this score cutoff, a 43% production rate together with an agreement rate around 94,14% was calculated on verification data.

Other topics on coding quality have gained less attention in the literature and we will only touch those as well: DeBell (2013) comments on optimal practices for manual coding into small-size coding schemes that are rarely ever fulfilled. Hacking and Willenborg (2012) emphasize that not a single code but multiple ones may be considered correct. Esuli and Sebastiani (2010) describe an accuracy measure useful when it is not of relevance to assign each answer to the correct code but only the population estimate is of interest.

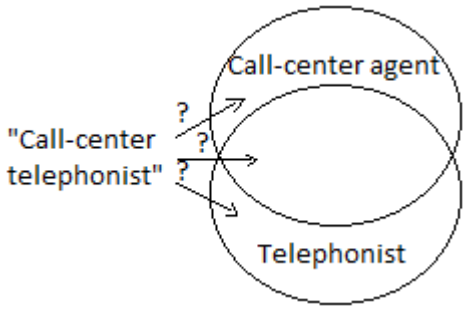

Figure 2.6: Coding Ambiguity

In order to comply with the international standard ISO 20252 for market, opinion and social research, Statistics Sweden has implemented several measures for quality control in the coding process. Erroneous codes can be corrected and the coding process can be improved by identification of problematic categories. For human coders with noticeable high error rates adequate training is given. An IT-tool was developed for computer-assisted coding that supports independent verification coding (Svensson, 2012).

We shall conclude this section with a thought experiment demonstrating that reliability is not to be optimized at all costs and the ideal automatic coding software may need to make random decisions: Imagine a verbatim answer that cannot clearly be assigned into one category A, but fits into two categories A and B equally well (see Figure 2.6: contrary to the assumption described here, both categories are not exactly equal). Furthermore, we assume that no other verbatim can be coded into these categories. At this point, a general rule can be included in the coding manual that assigns the verbatim to category A. When coders know this rule, inter-coder reliability increases, but this comes at the cost of interpretability of category A and B. Contrary to the category definitions, category B is empty and category A has doubled its size! Therefore, both proportions may only be interpreted with the knowledge of coding rules, or, in other words, coding rules have been added to the original category definitions. This needs to be made transparent to all data users.

While it may be acceptable to have such coding rules in a coding manual published (e.g., TNS Infratest Sozialforschung (2012), conventions from Geis (2011)) but generally not known to the end user, matters become even worse with deterministic automatic coding software that documents such rules only implicitly in the database. From a theoretic point

of view, we therefore suggest the following solution: Do not create a rule but let coders decide which category comes closer to the verbatim's meaning. When many coders do this task, the law of large numbers ensures that both categories are assigned with the same probability which is in accordance with our original assumption. The ideal computer program should also allow for variations in coder decisions which can be done using the Bernoulli distribution. Because it is clearly not desirable to have two categories with equal meaning in the coding scheme, one may want to merge both categories afterwards. Note that this argument is related to the survey literature, where it is generally feared that specific coders are biased in their decisions by a preference for particular categories and therefore variance is inflated (e.g. Groves et al. (2009)). In other words, the argument is that unknown coding rules related to specific categories will create a systematic coding bias as well.

2.5.1 Quality of Occupation Coding

In this section we will give an international overview on the quality of occupation coding. Empirical results for Germany will be discussed below. As a reference it shall suffice here to say that the inter-coder reliability for coding into the old 7-digit DKZ is below 70%.

In the United States, the US Census Bureau used 1.5 million responses from the American Community Survey (ACS) to learn a model for industry and occupation coding (4-digit). Coding of industry and occupation is carried out in parallel to use the code from one variable for prediction of the other. Clerical coders as well as the coding software "are required to maintain an error rate of 5% or lower as determined by a quality assurance process run". Although the training data set is huge, a production rate of only 43% is achieved (Thompson et al., 2012). This number may be compared to the production rate in computer-assisted clerical coding where only "[a]pproximately 18 percent of all industry and occupation responses are sent to coding referralists" (U.S. Census Bureau, 2009).

The Automated Industry and Occupation Coding System for the Koreans uses training data from the 2005 Census with about two million records. *Company name*, *business Category*, *department*, *position*, and *job description* is used to predict 1 of 450 categories from the South Korean standard code book. If the agreement rate is fixed at 98% a 73% production rate is reached (Jung et al., 2008).

The French automated coding system SICORE is reported to have a 66% production rate and a 96% agreement rate for occupations (Riviere, 1997). Although this result seems excellent, it should be taken with caution because the quality controls are not well described in the report.

In the Labour Force Survey conducted by Statistics Sweden, more than 80% of occupations were coded during the interview, a small percentage by automatic coding methods, and the remaining, most difficult cases (15-20%) by computer-assisted manual coding. Error

rates are only reported on the highly aggregated one-digit level and are at 9% for manual coding in ISCO. Even smaller error rates are achieved for cases from interviewer or automated coding (Svensson, 2012).

In the United Kingdom, multiple studies have examined coding into the *Standard Occupational Classification* (SOC) with 371 categories. Campanelli et al. (1997) find an inter-coder reliability of 78% for intermediate level coders. Other cited studies vary between 70% when office coders are compared to interviewer field coding and 84% for expert coders.

For Germany, three different classifications are available for occupation. A manual for coding into the international ISCO-08 is given by Geis (2011), coding into the national KldB has been described by Hartmann and Schütz (2002) and TNS Infratest Sozialforschung (2012), and Paulus and Matthes (2013) give a manual for coding into the DKZ which is derived from the KldB. To obtain good coding results, it is generally recommended to ask 2-3 questions about the employment and a further question about the professional status ("Berufliche Stellung").[3] If available, further variables like industry, size of enterprise, school and vocational education, or employment history have been useful as well. Geis and Hoffmeyer-Zlotnik (2000) provide additional background information.

Though some attempts have been made to improve automated coding for ISCO (see Hoffmeyer-Zlotnik et al. (2004), Hoffmeyer-Zlotnik and Warner (2012)), the rule-based method currently employed by Geis (2011) has a production rate lower than 50% and manual checking is intended. The quality for coding according to the ISCO-88 classification (390 categories) has been investigated by Maaz et al. (2009). In their study, two professional institutes and two research assistants without prior coding experiences have coded occupations from the parents of 300 high school graduates. For the 12 resulting combinations from four individual coders, inter-coder reliability varies between 41.6% and 53%. After aggregating coding decisions into the ten one-digit major groups, reliability increases to 67.5% to 74.7%. When coded occupations are transformed into the *International Socio-Economic Index of Occupational Status* (ISEI), measures of validity are more promising. The authors conclude that, while the ISCO scale only has low reliability, other derived scales may still be valid.

Quality checks for coding into the KldB 2010 have been presented by Prigge et al. (2013): The reliability for 5-digit KldB is above 80%, for 2-digit KldB above 90%, Cohens Kappa has been calculated for supervisors and managers (4th digit = 9) to 82.6%, and for the skill level (only 5th digit) to 88.0%.

[3] Statistisches Bundesamt (2010) give the following standard formulations:

- Welche berufliche Tätigkeit üben Sie derzeit hauptsächlich aus?
- Bitte beschreiben Sie mir diese berufliche Tätigkeit genau.
- Hat dieser Beruf noch einen besonderen Namen?
- Nun sagen Sie mir bitte nach dieser Liste hier, zu welcher Gruppe dieser Beruf gehört.

Main results for semi-automated coding into the DKZ are summarized as follows: 61% of the textual answers needed manual coding and 9% of these had to be revised by a supervisor. The remaining 39% were coded automatically (the larger part) or semi-automatically with a human decision. Inter-Coder Reliability was only calculated for answers that were manually coded with the following results: 50% for the 7-digit DKZ, 65% for the 4-digit KldB 1988, 79% for the 2-digit KldB and 70% for the 4-digit ISCO-88. Under the strong assumption that automatic coding was correct in all cases, there is 70% overall reliability for the DKZ as mentioned above. Drasch et al. (2012) argue that automatically coded answers will have lower error rates than manual coding, and for some answers multiple categories may be considered correct. They further hope that coding into the newly developed KldB 2010 will increase inter-coder reliabilities. The study from Prigge et al. (2013) described above supports this hypothesis.

Summarizing, this short international survey reveals some interesting points. First of all, quality measures are not consistent and often describe only one aspect from the whole coding process. In some studies cited above, reliability is calculated, others report the production rate and the proportion of "correct" codes. Though these concepts are not directly comparable, the wide variety of reported quality is eye-catching. This can be best illustrated with the following numbers. With the DKZ approach, reliability for the 4-digit ISCO is above 70%. When ISCO was coded directly, it was below 53% (4-digit) and below 75% for 1-digit codes. The Swedish system reaches error rates below 9% for 1-digit ISCO. This high variability is no surprise, but arises from the fact that ambiguity in verbatim answers, coding procedure and coder's expertise determine the quality of coding.

Despite all the differences, the difficulty to code occupations is obvious in all studies. This underlines the need for quality control and systematic improvement. We shall further note that quality measures are often - if at all - documented in some technical manual and not used for further analysis. It may be more relevant to look at the quality of derived indexes like the approach from Maaz et al. (2009) described above. An even more ambitious task is to find ways to incorporate into statistical analysis the uncertainty from measurement inherent to the occupation variable and see how results change.

Regarding automatic coding, we shall point out that, even though training data used by Thompson et al. (2012) and Jung et al. (2008) is huge, production rates are between 43% and 73% and thus not neccessarily higher than systems with carefully designed rules. With the exception of Jung et al. (2008), all automatic systems envisage manual or computer-assisted coding for doubtful cases. We believe such a tool can prove useful for Germany as well.

Chapter 3

Data Analysis

To use computers for automated coding, one needs to supply the machine with relevant background information. As described in section 2.4, hand-crafted rules and dictionaries are often used but laborious to construct. The other option is to use training data, where verbatim answers are already coded. Our work focuses on the latter, and different methods to predict new codes using training data will be discussed in section 3.2. Data from the ALWA survey is used to train and test the algorithms. To see if automated coding procedures can be generalized to new data sets, we use another test set from the lidA survey. Both data sources will be described in detail in the next section.

3.1 Description of Survey Data

The ALWA survey (short for 'Arbeiten und Lernen im Wandel', translated 'Working and Learning in a Changing World') described by Antoni et al. (2010) has been conducted to study how informal competencies and knowledge, aside from formal educational attainments, support professional careers. To this end, a clustered sample from all persons born between 1956 and 1988 and living in Germany was drawn and questioned about their educational and professional development. 10404 telephone interviews (CATI) were conducted. In this sample the following groups are underrepresented: the young, the low-educated and persons with a migration background are less frequent compared to the total population.

We are only interested into the employment biography, i.e. all the jobs that each person was holding during her lifetime. In the dataset we used, 32882 job records from 9227 different persons are present. When people find a new job, they often keep working in the same occupational area and thus the job reports from a single person are not statistically independent and often even identical. Many dependent answers lead to a dataset with less diversity compared to independent answers and thus the effective sample size is smaller than 32882 job records. We are interested how well our prediction methods generalize for new, independent job descriptions. Special provisions are taken and will be described below to

provide performance measures that hold also for independent answers.

To allow for comparisons over different data sets, codings from another study are used as well: The lidA survey (short for 'leben in der Arbeit. Kohortenstudie zu Gesundheit und Älterwerden in der Arbeit') is a cohort study to examine the relationship between work and health among aging employees. The total population consists of all employees with social insurance and born either in 1959 or 1965 excluding public officials ("Beamte") and self-employed workers. A sample of 6585 persons was interviewed face-to-face (CAPI). This sample is nearly representative of the population with only small deviations similar to those described for the ALWA study above (Schröder et al., 2013). Each person gives information on her current job, or, for the unemployed, the last job before unemployment.

For occupation coding, professional coders use a number of different variables from the dataset. 2-3 questions on employment activities and a further question on professional status are most helpful for coding and asked in most German surveys to classify the occupation. We will use the same variables for automated coding as well. Before we can consider generalizations over different datasets, we must look if these input variables have a similar format. As we will describe in the following, some of these variables differ in relevant aspects.

Prior to all analysis, we make the following standardizations with all verbatim answers: All letters are capitalized, special German characters replaced (e.g., 'Ä' to 'AE', 'ß' to 'SS', etc.), punctuation and short abbreviations (i.e., at most 3 characters followed by a '.') removed, and white spaces at the start and

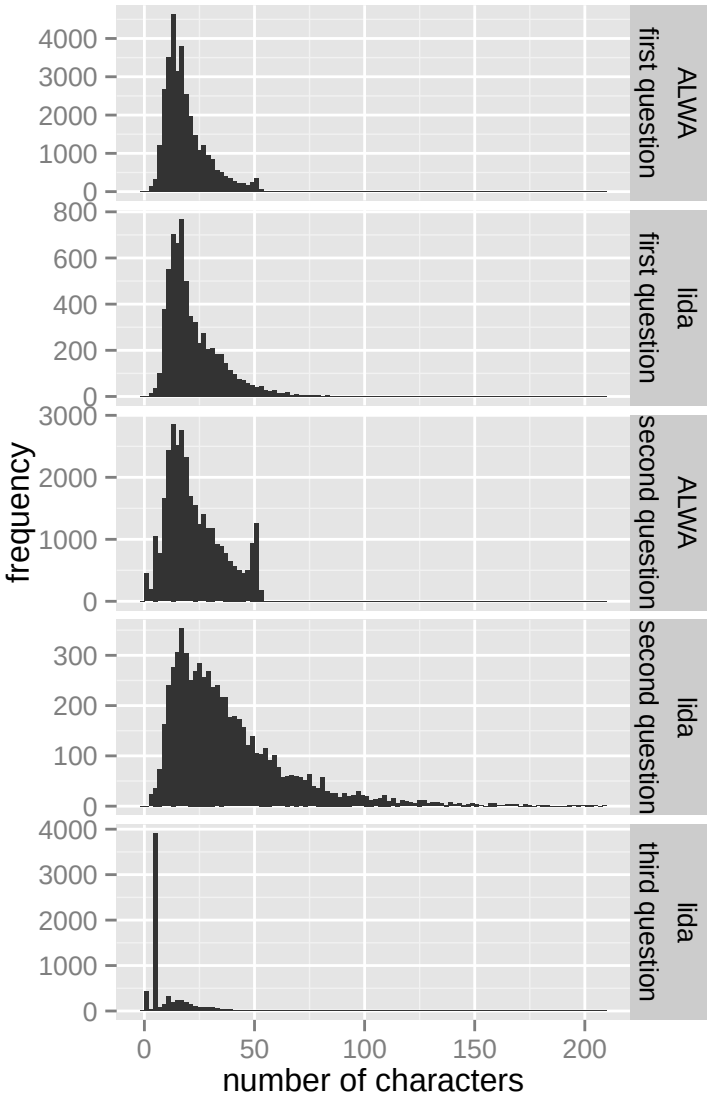

Figure 3.1: Number of Characters to Verbatim Answers

	ALWA	lidA
Second answer refused	0.3%	0.1%
Second answer is not informative	32.1%	8.0%
Second answer equals first answer	9.1%	6.2%
Second answer contains additional information	58.4%	85.6%

Table 3.1: Information Content from Second Answer

end of each string are trimmed.

Figure 3.1 shows different answer lengths to the open-ended questions "Welche berufliche Tätigkeit üben Sie derzeit hauptsächlich aus?" (first question), "Bitte beschreiben Sie mir diese berufliche Tätigkeit genau." (second question), and "Hat dieser Beruf noch einen besonderen Namen?" (third question). When ALWA answers exceeded a limit of 50 characters, the last characters were clipped and the full answer is not saved. While answer length to the first question does not differ much, answers for the second question in lidA are in general longer than for ALWA. The third question was not asked in ALWA. In lidA 57% of the respondents answered this question for another job name with a simple 4-digit "nein" (no).

Two possible explanations for the longer answers in lidA are that, firstly, respondents are less willing to give detailed answers after they have answered the same question for multiple prior jobs before and, secondly, respondents may want to give more details on themselves in personal interviews (lidA) compared to telephone interviews (ALWA). A closer view into the second answer provides additional evidence that respondents in the lidA study were more motivated to give informative answers. Table 3.1 summarizes common answers to the second question. For a small proportion of respondents, interviewers recorded refused answers ('-7' or 'verweigert') that we replaced with the word 'VERWEIGERT' for further processing. A proportion of 32.1% from the ALWA respondents did not specify their job with the second answer. Frequent records for this are 'keine näheren Angabe', 'nein', 'dto.', 'dito', '-8', 'weiß nicht', and the empty string. We replace such statements with the answer given to the first question in order to treat them the same way as those answers where identical words are given for the first and second question. Only 58.4% from the ALWA study and 85.6% from the lidA study give additional details about their job in the second question that can be used for coding.

Careful inspection of the verbatim answers reveales additional patterns that a perfect automated coding algorithm should recognize automatically: This includes misspelled words, answers with a hyphen (i.e. for the answer 'Küchen- und Möbelmonteur' the two words 'Küchenmonteur' and 'Möbelmonteur' would be better suited as algorithm input), and the detection of multiple jobs (i.e. 'Schlosser und Kraftfahrer' cannot be coded into one category). We will not provide solutions for these problems but use only the simple algorithms described above for string preprocessing.

Aside from verbatim answers about employment activities, the professional status is used for coding. ALWA and lidA both asked for it with a closed question but different answer categories were used. We therefore aggregated categories from both studies into a less detailed variable such that an exact mapping from both studies into the new variable exists. Figure A.2 shows the resulting category scheme and relative frequencies how often each category is found in each study. Large differences between ALWA and lidA are probably caused by different total populations in both studies.

3.1.1 Job Codes

The coding procedure and quality checks for ALWA have been documented by Drasch et al. (2012). Automatic coding was complemented by manual coding with special provisions for dificult cases. Because the original answers were coded into the out-dated 7-digit DKZ, a transition table was used to convert the codes into the current 8-digit DKZ where the first five digits represent the KldB 2010. For lidA, answers were coded directly into the current DKZ/KldB2010.

For the coding in both studies, additonal categories were necessary. When it was not possible to find the correct code for a verbatim answer, it was coded as an imprecise answer. In ALWA, a proportion of 0.46% of all answers was coded as imprecise compared to a proportion of 1.05% in lidA. Because answers from lidA are in general longer and therefore should be more precise, this significant difference comes as a surprise and we recommend further investigation. For the coding in ALWA, further categories were introduced for student research assistants, helpers not included in other codes, and persons with multiple jobs. Together with 1286 categories defined in the KldB 2010, this gives us in total 1290 categories for coding.

Although different populations were interviewed for ALWA and lidA, one might hypothize that each job category has the same probability of occurence in both studies. This assumption may be tested with the χ^2-Test for homogenity. The test statistic is

$$\chi^2 = \sum_{i=1}^{k} \sum_{j=1}^{m} \frac{(f_{ij} - \hat{f}_{ij})^2}{\hat{f}_{ij}}$$

with $k = 2$ studies, m is the number of categories, f_{ij} the frequency of category j in study i and $\hat{f}_{ij}$ is the expected frequency under the null hypothesis. Applied to the two-digit Berufshauptgruppen, the null is significantly recected with $\chi^2 = 615.4$. Particular high deviations between both studies can be found for the Berufshauptgruppen 'Medizinische Gesundheitsberufe' (more frequent in lidA than expected, $\frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}} = 85$) and 'Mechatronik-, Energie- und Elektroberufe' (less frequent in lidA than expected, $\frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}} = 58$). Further research would be required to find out if the differences are caused by distinct total populations or by

disparate coding practices in both studies. For the different proportions of answers coded as imprecise (see above, $\frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}} = 82$) the latter explanation is more plausible to us.

While the test for equal distributions of Berufshauptgruppen reveals relevant differences in both studies, the same test is also helpful to check if frequent answers in both studies have been coded into the same categories. For each first answer, e.g. 'Sachbearbeiterin' ('clerk'), that was coded multiple times in ALWA as well as in lidA one may expect that both studies code the same word typically into the same category. To test this, we calculate the χ^2-statistic for each first word. Due to the small number of observations for each word, assumptions for formal tests are in general not fulfilled. High χ^2-statistics are, however, still a good indicator to find first answers that were coded systematically different in both studies. We therefore recommend this statistic to find erroneous code assignments for manual inspection.

Two examples may illustrate the use: After calculating the χ^2-statistic for all first answers, we find that the Sachbearbeiterin ("clerk") has the highest score $\chi^2 = 315$ of all first answers. Closer inspections shows that different standard categories were used in lidA (97% coded into category 71302) and ALWA (66% coded into category 71402). Another example is the Informatiker ('computer scientist', $\chi^2 = 14$). In lidA, four persons gave this first answer and all were coded into the general computer science category 43104. ALWA, in contrast, coded ten persons with the very same first answer in three different, more specific categories (mostly in categories 43414/43423 for software development). With a closer inspection of the second answer, more precise code assignment would have been possible for lidA, too.

A further indicator that lidA codings may often be correct but overly general is the following. The KldB 2010 includes an alphabetic dictionary with 24000 occupation titles that assigns a 5-digit code to each occupation. 45% (lidA) respectively 49% (ALWA) of all first answers have an exact match to one of those dictionary entries. For lidA, 95.6% of all dictionary codes with exact matches agree with the assigned code whereas the same number is only 76.4% for ALWA. This difference can possibly be explained with the Informatiker-example described above. The lidA codes are in accordance with the dictionary entry while the ALWA codes are not. Because lidA did not use additional information from the second answer to find the most specific job, the codes are in better alignment with dictionary codes from the KldB 2010. We are skeptical that this implies better job codes as well. In section 3.2.1 the dictionary coding method is described in detail.

To summarize, we have shown that lidA and ALWA data differ in many aspects. Two different populations were surveyed and relevant variables do not follow the same distribution. In lidA, the average answer length for the second answer is longer but it has possibly been used less for coding. There is evidence that people with similar jobs in both studies have been coded systematically into different categories. Moreover, many categories were

not used once for coding. Out of 1290 existing categories, 437 categories (ALWA) respectively 646 categories (lidA) have not been used a single time. No prediction algorithm that is based on this training data will therefore predict these categories. This is a first sign - and others will follow - that additional training data will improve all the methods proposed in the next section for automated coding.

3.2 Methods for Automated Coding

Our aim is to develop new automated techniques to reduce the amount of work required for coding. At the same time, the quality is of high relevance and needs to be closely monitored. All automated coding systems we have described in section 2.5.1 require therefore human efforts to code difficult cases. But even if the human makes the final decision, computer-assisted coding has proven useful. Hereby, the computer program provides a list with possible categories to reduce the time needed to search for the correct code. When the number of suggested categories is large, the coding clerk may find ordered results helpful with best fitting categories first. All probabilistic methods described below provide a score that can be used for ordering.

Computer-assisted coding is one automated coding method, automatic coding the other. When human supervision is not required for quality control, the top-ranked category is a natural candidate for automatic coding. Then, it becomes essential to estimate the probability that this top-ranked category is also the correct one. Typically, only those answers with highest correctness probabilities are coded automatically, the rest is referred to a human coder. Thompson et al. (2012) and Jung et al. (2008) both fix this probability at a point such that more than 94% of automatically generated codes agree with human coding decisions.

To test our methods we use the ALWA and lidA data described above. Only the ALWA data is large enough to be used for training. We therefore split the ALWA data into training data with 7436 persons having 26297 jobs recorded and test data with 1791 persons having 6585 jobs recorded. The split is done at random, but under the condition that no person has her different jobs she was holding during her lifetime scattered over both the training and the test data. This condition avoids unrealistic good results that may happen when a person gives multiple times the same answer to describe the same job. If these answers were scattered over training and test data the algorithm would find useful training data more often than what would be the case for a different data set. To see how good our automated coding methods using the same ALWA training data generalize to new coding situations, test data from the lidA survey is used. With 6585 respondents in lidA, both test data sets are of equal size. As we have seen above, ALWA and lidA codes differ systematically and thus one must expect test performance to be worse in lidA when the same ALWA training

data is used for prediction in both test data sets.

We believe that employment coding should not be done in the back office from computers and coding clerks but at the time of the interview when the interviewer can ask for further details from the respondent. For this, the interviewer shall be provided with the ordered list of suggested job categories, alike to computer-assisted coding. A difference between the coding methods arises in the fact, that back office coding should use as much information as available to find the correct code. This is not the case for our desired general tool for interviewer coding, where it is prohibitive to assume that questions found useful for back office coding about industry, vocational education, or employer's size are always asked beforehand and can be used for interviewer coding. Also, a second and third question about the respondent's employment is relevant for back office coding, but the tool for interviewer coding should work without because the interviewer is expected to ask more precise answers. Unless otherwise noted, we have therefore tested our prediction methods using only the respondent's first answer and the shortened differentiated professional status as depicted in Figure A.2. In any case, the Naive Bayes method and the Combined Method are designed to allow usage of additional covariates. Improvements over the following reported results should therefore easily be possible for back office coding.

The following sections describe different methods we have tested for automated coding. Each algorithm except the rule-based coding makes predictions using training data from $n = 26297$ answers that are already coded. To measure performance, we have test data from $m = 6585$ respondents available. The common output from all algorithms is a score θ_{lj} for each respondent in the test data, $l = 1, ..., m$, and all possible categories $j = 1, ..., J$ where $J = 1290$ is the number of job categories. The construction of these scores differs for each method but with one property holding for all: The score θ_{lj} is expected to correlate with the true probability $P(c_j|l)$ that job category c_j is correct for respondent l. In fact, with the exception of the rule-based coding method, the idea for all the following methods is to estimate this probability, setting $\theta_{lj} = \hat{P}(c_j|l)$. We therefore call θ_{lj} the *estimated correctness probability*. To obtain these probabilities, statistical models are built from training data with respondents $i = 1, ..., n$. Estimations obtained from the training data are then extrapolated to the test data.

The different prediction methods may be considered as black-box algorithms where one is not interested into the internal mode of operation. From this point of view our main results are presented next. The best method we developed is a combination of the other algorithms as described in section 3.2.4. Figure 3.2 shows the usefulness of this method for computer-assisted coding. For the diagram, the estimated correctness probability θ_{lj} have been sorted for each respondent l with highest scores first. The associated codes can then be presented to the coding clerk who should hopefully find the correct job category within the first few suggestions that have highest scores. The graphic shows that top-ranked

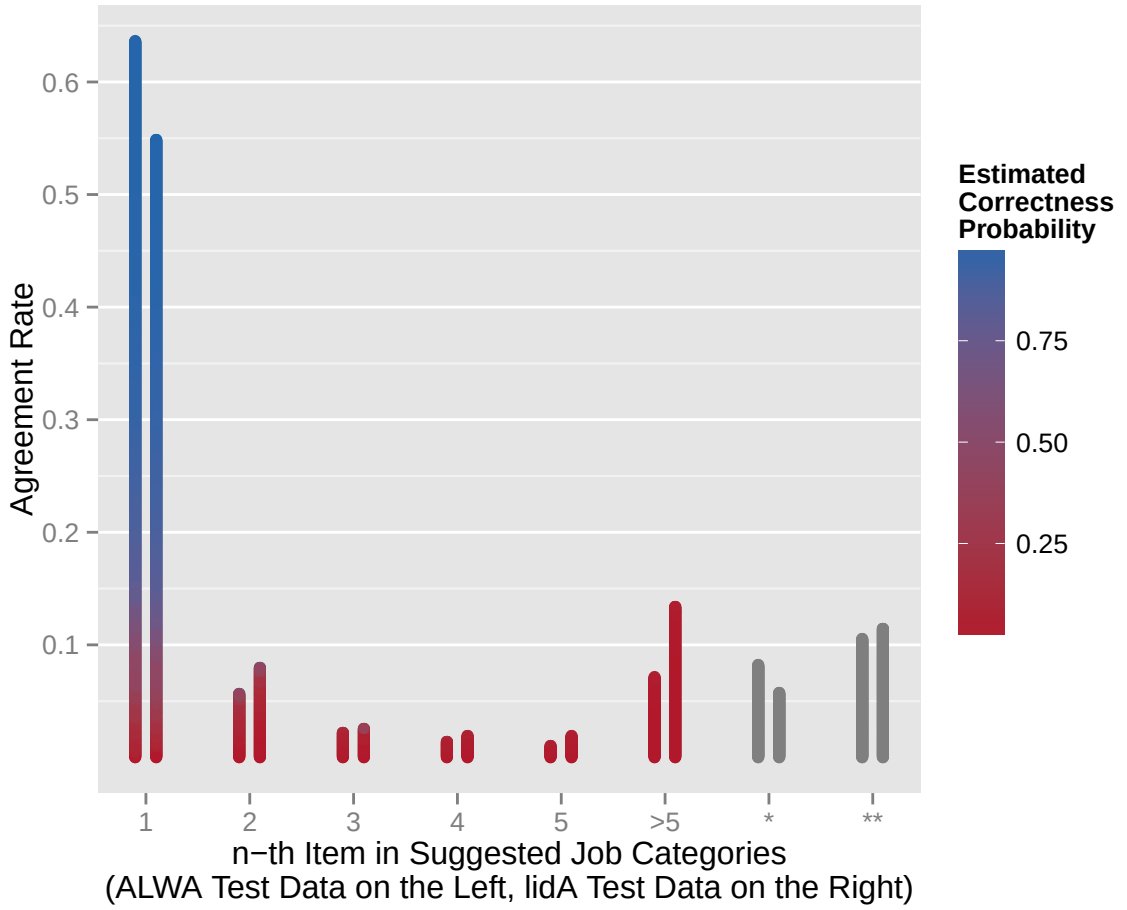

Figure 3.2: Agreement rates for comuter-assisted coding. Shown are relative frequencies how often the n-th ranked category is correct. Error rates are given for grey bars.
'*' = Suggestions do not include the correct category
'**' = No suggestions available

job categories (1st item) are in agreement with the assigned code 63.64% of the time for ALWA (left) and 54.88% for lidA (right). The worse performance in the lidA test data is as expected because of the described systematic differences in ALWA and lidA codes. One can further see that suggested categories ranked second to fifth contain a substantial proportion of correct codes. Thus, it would be possible for a human coder to find for 74.08% of all answers (ALWA, lidA: 69.35%) the correct code within the top five suggestions. For the residual cases the the system is less useful due to different reasons: A proportion of 7.15% for ALWA (lidA: 13.41%) has the correct job category only suggested after the five top-ranked categories. For these cases the available training data and dictionaries still find the correct job category which can be suggested to a professional coder. This is not the case for other answers, marked grey in the diagram. For 8.23% (ALWA, lidA: 5.76%) we find only wrong code suggestions in training data and dictionaries and for further 10.54% (ALWA, lidA: 11.48%) these sources do not provide any hint about a possible job category at all. Taken

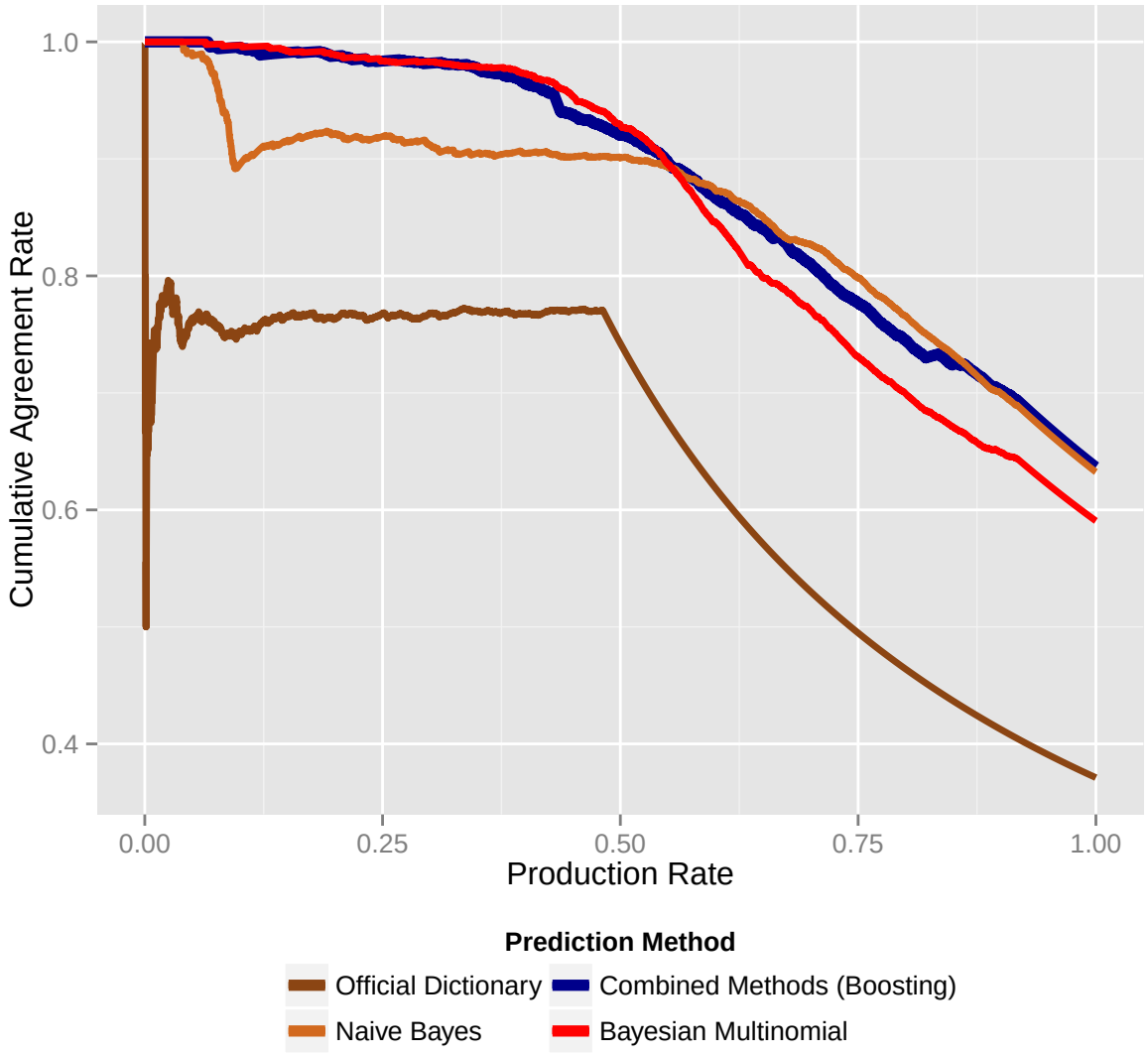

Figure 3.3: Agreement and Production Rates for ALWA-test data

together, this sums up to nearly 20% of all answers where neither dictionary nor data-based statistical learning methods are able to give any suggestion about the correct job category. Only additional training data, more dictionary rules, or better string preprocessing may be useful to process these answers with automated coding methods. Colors are used to depict the algorithm's certainty that the correct category has been found. Answers with high estimated correctness probability θ_{lj} are marked blue. These are candidates for automatic coding without human interaction.

If one desires automatic coding, the algorithm's performance is better described with figure 3.3. Applying ideas from Chen et al. (1993), the chart compares quality from automatic coding for ALWA test data using different prediction methods. As described before, the top-ranked category suggestion is the only candidate for automatic coding. This category suggestion is, however, not always correct and it is therefore relevant to decide automatically

if the suggested code shall be assigned by the computer or if the answer is referred to manual coding. The estimated correctness probability for the top-ranked category, $\theta_{l(1)} = \max_j \theta_{lj}$, can be used for this decision. Answers are only coded without human supervision when this probability is above a certain threshold and otherwise not. When this threshold is higher, fewer codes are assigned automatically and thus the production rate is smaller. At the same time, the cumulative agreement rate, i.e., the proportion of automatic code assignments that agree with the human-coded 'true' job categories rises. The diagram shows that, if one were to fix the desired agreement rate at 95%, it would be possible to code 43.25% of all answers with the 'Combined Methods (Boosting)'- algorithm, 7.96% with Naive Bayes, and 45.36% with the Bayesian Multinomial model. Although these numbers show that the last model performs best when high agreement rates are necessary, this is not the case if one were to code at 100% production rate all top-ranking answers. In this case, only 59.06% would agree with human code decisions for the Bayesian Multinomial method, 63.23% for the Naive Bayes method, and, as we have seen in figure 3.2, 63.64% for the 'Combined Methods (Boosting)'. The 'Official Dictionary'-method cannot be compared directly to the other methods, because it does not provide estimated correctness probabilities necessary for ordering. There is still one possible point of comparison, namely that 48.77% of all answer are found in a dictionary (production rate) and 76.44% of these dictionary entries provide the correct code (agreement rate). Variation at smaller production rates is due to random ordering of dictionary matches. For a production rate above 48.77%, the agreement rate decreases towards an overall accuracy of $0.4877 * 0.7644 + (1 - 0.4877) * 0 = 37.28\%$ at 100% production rate, because all answers not found in the dictionary cannot be coded accurately. Production and agreement rates for the lidA test data are provided in the appendix (A.1) and are in accordance but without new insights for our discussion here.

While the black-box approach above is good for an overview over methods used and results obtained, it is necessary to go into detail to understand why the different prediction methods perform as they do. Within the next few sections we provide the required background and further evaluation. In section 3.2.1 we describe how we use an existing job catalogue for automatic coding. Our study focuses, however, on the other possible source for background information: we use previous code assignments from the ALWA study to predict new codes. Two different methods, Naive Bayes and Bayesian Multinomial, applied for this task are described in sections 3.2.2 and 3.2.3. As we will see, a particular strength from the Naive Bayes model is that it uses the full answer string. This is in general not possible with the Bayesian Multinomial model, which comes with another advantage. Because prior information is used, one can account for small training frequencies when certain answers are not used often. To combine the strengths from all three methods, section 3.2.4 provides the details on the last method which is based on boosting.

3.2.1 Rule-based Coding

The definition of rules to assign verbatim answers into predefined categories is often done for survey coding. The idea is to match answers with entries from a dictionary and assign the corresponding code. The coding manual given by Geis (2011) is based on a dictionary and Drasch et al. (2012) have used dictionaries from the DKZ with 42000 job names and additional 101000 search words for semi-automatic coding of ALWA data. This method is also internationally the prevalent procedure and different coding programs developed for dictionary-based coding are described in sections 2.3 and 2.4.1.

For German occupations a number of different dictionaries exist. The official KldB 2010 documentation (Bundesagentur für Arbeit, 2011) includes an alphabetic list with 24000 job and occupation names together with corresponding 5-digit codes[1]. Other dictionaries are available as part of the 8-digit DKZ. The Federal Employment Agency uses the DKZ for various services and updates the database on a regular basis[2]. The file B_SY.txt contains 3920 8-digit DKZ codes, each with short and long job names in male, female, and neutral format (6 names in total). Additionally, the file B_SW.txt provides more than 150000 search words that link to one or more DKZ job codes and in the BERUFENET we find similar jobs ("Beschäftigungs-/Besetzungsalternativen") that may be helpful for computer-assisted coding.

Here, we use only the static and well documented alphabetic dictionary from the KldB 2010 and come back to the DKZ dictionaries only in section 3.2.4. The reason is that we want the dictionary to be a stable point of reference. If the rules from a dictionary are followed, identical verbatim answers are always coded into the same category, which explains the popularity of this method. At the same time predefined rules may be problematic. If a verbatim answer fits into multiple categories, a coding rule defines which single category is to be used and thus the underlying ambiguity is concealed. When the dictionary assigns answers to incorrect codes, systematic errors happen. No analysis of errors in coding dictionaries is known to us and thus it is unknown how frequent these dictionary misassignments are. Interpretation of job codes is therefore only possible in the light of those dictionary rules that were used for coding. With the regular updates in DKZ dictionaries this would be impossible. Also, all research should be reproducible but for coding this is not possible when updated dictionary versions are used.

The alphabetic list of occupations comes with some challenges for dictionary-based coding. The problems involved are best described with the following example: Two entries from this dictionary are "Betriebsschlosser/in" (Code 25102) and "Betriebsschlosser/in

[1] Online at http://statistik.arbeitsagentur.de/Navigation/Statistik/Grundlagen/Klassifikation-der-Berufe/KldB2010/Systematik-Verzeichnisse/Systematik-Verzeichnisse-Nav.html

[2] Relevant online services can be found at http://berufenet.arbeitsagentur.de/berufe/ and http://download-portal.arbeitsagentur.de/ (most relevant are the files B_SY.txt and B_SW.txt)

(Landtechnik)" (Code 25222). The first problem consists of different male and female names for many occupations. We therefore searched for frequent word endings to extract the corresponding male and female names (here "Betriebsschlosser" and "Betriebsschlosserin"). This procedure is obvious for the ending "/in", more difficult for endings like in "Leitende/r kaufmännische/r Angestellte/r", and automatic recognition for names like "Absteckdirektrice/-modelleur" was not possible for us. For 2091 out of 24000 job names from the dictionary we do not find male and female forms automatically and these entries are therefore discarded. The second problem arises from the fact that people do not use parentheses in verbatim answers. For simplicity we delete parentheses and the text within.

Results from dictionary based coding have been reported earlier in this thesis. 45% (lidA) respectively 49% (ALWA) of all first answers have an exact match to one of these (preprocessed) dictionary entries, either in male or female form. Only if there is exactly one match it is counted as a match. This means in particular for the "Betriebsschlosser" where two possible codes (25102 and 25222) are found that this word is not coded automatically using this preprocessed dictionary. For lidA, 95.6% of all dictionary codes with exact matches agree with the assigned code whereas the same number is only 76.4% for ALWA.

Another problem is that many jobs have general names (e.g., "Agrarwirt/in" or "Tischler/in") that code in one category and more specific names (e.g., "Agrarwirt/in Baumpflege und Baumsanierung", "Agrarwirt/in Besamungswesen", ... or "Bautischler/in", "Billardtischler/in", ...) that code into different categories. We assume that people often only answer with the general name "Agrarwirt" or "Tischler" and the text is therefore miscoded when in fact the more specific name would be correct. This means, when rule-based coding is based only on the first answer, automatic code assignments are often incorrect. Computer-assisted coding and coding during the interview may lead to better results and a prototype for it is presented in chapter 4. With this method, job codes are suggested to the human coder not only when the dictionary match is exact but also if the given answer is part but not identical to the dictionary entry (partial match).

3.2.2 Naive Bayes

The Naive Bayes algorithm is well-known and often used as a benchmark for new algorithms (e.g Lewis (1998)). We apply it, because it provides a simple technique to handle answers with multiple words and any number of covariates can be included in the model.

Theory

Let $c_j, j = 1, ..., J$ specify the J job categories, q_i is a verbatim answer and x_i are further covariates for respondent $i, i = 1, ..., N$.

Using Bayes rule, one may calculate the probability that respondent i works in job category c_j,

$$P(c_j|q_i, x_i) = \frac{P(q_i, x_i|c_j) \times P(c_j)}{P(q_i, x_i)} \tag{3.1}$$

It is natural to predict that category c_j with the highest probability given the covariates. Tutz (2000) (p. 344) shows that this prediction rule minimizes the probability for false classification. This minimal error probability can in theory be calculated as

$$\epsilon_{opt} = \sum_{q_i, x_i} \min_{j=1,...,J} (1 - P(c_j|q_i, x_i)) \times P(q_i, x_i) \tag{3.2}$$

Problems arise because the right hand side in formula 3.1 is in general not known. While the denominator $P(q_i, x_i)$ is constant for all c_j and can be neglected, the numerator needs to be estimated. This is difficult in particular for $P(q_i, x_i|c_j)$ because the number of possible combinations between arbitrary verbatim answers q_i and all possible values for covariates x_i and job categories c_j is far larger than the size of our training data. Instead of estimating all the probabilities in this three way contingency table, we reduce dimensions with the Naive Bayes assumption of condtional independence between answers and other covariates given the job category. With this assumption we may write

$$P(c_j|q_i, x_i) \propto P(q_i, x_i|c_j) \times P(c_j) \tag{3.3}$$

$$\propto P(q_i|c_j) \times P(x_i|c_j) \times P(c_j) \tag{3.4}$$

The Naive Bayes assumption gives us therefore a way to handle a high number of covariates by multiplying conditional probabilities together. For $P(x_i|c_j)$ and $P(c_j)$, the relative frequencies are obvious estimators.

More difficult is the handling of language. How should we estimate $P(q_i|c_j)$, the probability that the respondent gives the observed answer q_i given that she works in job category c_j? This problem has been studied extensively in the field of Information Retrieval (e.g. Manning et al. (2008)) and text categorization (e.g., Aggarwal and Zhai (2012) and McCallum and Nigam (1998)). We follow a common approach that is also based on the Naive Bayes assumption. The basic trick is to neglect, again, dependencies between how often single words $w_{i1}, ..., w_{iV}$ appear in q_i (the so-called bag of words assumption) and model this with a multinomial distribution:

$$P(q_i|c_j) = P(W_1 = w_{i1}, ..., W_V = w_{iV}|c_j) \tag{3.5}$$

$$= K_{q_i} \prod_{v=1}^{V} P(T_v|c_j)^{w_{iv}} \tag{3.6}$$

Hereby, $v = 1, ..., V$ is an index for the V possible words that may be used by respondents, $W_1, ..., W_V|c_j$ is the distribution of word frequencies given c_j which is assumed to follow the multinomial distribution with parameters $P(T_1|c_j), ..., P(T_V|c_j)$, interpretable as probabilities that a word T_v is used by a respondent given she is in category c_j. The constant $K_{q_i} = \frac{(\sum_{v=1}^{V} w_{iv})!}{\prod_{v=1}^{V}(w_{iv}!)}$ can be ignored because it does not depend on the job category c_j. We simplify this model further by setting the word frequency for a particular word w_{iv} to one when it appears at least once in answer q_i.

Estimation of usage probabilities for particular words $P(T_v|c_j)$ by a respondent is now the key to achieve good model performance. Relative frequencies are not satisfactory because many words are not used often and the contingency table for words and job categories is very sparse. When a respondent uses a new word T_v not answered before, $\hat{P}_{ML}(T_v|c_j) = 0$, and inserting this estimators into the formulas above yields $P(c_j|q_i, x_i) = 0$ for all job categories, which is obviously not desirable. Even worse is the case when the respondent's answer contains a word that was only used a single time before. $\hat{P}_{ML}(T_v|c_j)$ will be zero for all but one category c_j and as a result this c_j is strongly suggested by the algorithm to be the correct category although only one little used word has indicated it.

Smoothing is therefore necessary and we use Jelinek-Mercer smoothing, which is a weighted average from a category specific frequency estimate and a global estimate,

$$\hat{P}(T_v|c_j) = \lambda \hat{P}_{ML}(T_v|c_j) + (1 - \lambda)\hat{P}_{ML}(T_v) \tag{3.7}$$

Although Manning et al. (2008) stress the importance to choose λ well, we tested it with $\lambda = 0.7$ and $\lambda = 0.95$ and did not find large performance differences. Because predictions were slightly better with $\lambda = 0.95$, we set λ accordingly in the following analysis.

To summarize the formulas above, we attain an estimation for $P(c_j|q_i, x_i)$ by plugging in the different relative frequencies (ML-estimators) as

$$\hat{P}(c_j|q_i, x_i) \propto \hat{P}(c_j) \times \hat{P}(x_i|c_j) \times \hat{P}(q_i|c_j) \tag{3.8}$$

$$\propto \hat{P}_{ML}(c_j) \times \hat{P}_{ML}(x_i|c_j) \times \prod_{v=1}^{V} P(T_v|c_j)^{w_{iv}} \tag{3.9}$$

$$\propto \hat{P}_{ML}(c_j) \times \hat{P}_{ML}(x_i|c_j) \times \prod_{v=1}^{V} (\lambda \hat{P}_{ML}(T_v|c_j) + (1 - \lambda)\hat{P}_{ML}(T_v))^{w_{iv}} \tag{3.10}$$

$$\propto \frac{\#\{c_j\}}{N} \times \frac{\#\{x_i|c_j\}}{\#\{c_j\}} \times \prod_{v=1}^{V} (\lambda \frac{\#\{T_v|c_j\}}{\#\{c_j\}} + (1 - \lambda)\frac{\#\{T_v\}}{\sum_{u=1}^{V} \#\{T_u\}})^{w_{iv}} \tag{3.11}$$

with $w_{iv} = 1$ if word T_v was used by respondent i and $w_{iv} = 0$ otherwise. $\#$ is the counting operator and thus $\frac{\#\{x_i|c_j\}}{\#\{c_j\}}$ is the proportion of respondents with covariate x_i from all respondents in job category c_j. While this proportion is counted on the basis of respondents,

Name	Used Variables	λ	AUC
NB 1-answer lambda = 0.7	First Answer	0.7	0.877
NB 1-answer	First Answer	0.95	0.884
NB 1-answer W Prof. Status	First Answer & Professional Status	0.95	0.886
NB 1-answer W Full Training	First Answer (& Second Answer pasted)	0.95	0.864
NB 2-answers	First & Second Answer pasted	0.95	0.832

Figure 3.4: Properties from various Naive Bayes Models

$\frac{\#\{T_v|c_j\}}{\#\{c_j\}}$ is the proportion of the number of word T_v over all words used to describe category c_j.

Though this is the basic formula used, our calculations deviate in some technical aspects. First, in the next section we do not estimate $\hat{P}_{ML}(c_j)$ with relative answer frequencies $\frac{\#\{c_j\}}{N}$ but with relative freqencies how often single words are coded into category c_j. Second, the ML-estimator $\hat{P}_{ML}(x_i|c_j) = \frac{\#\{x_i|c_j\}}{\#\{c_j\}}$ is not defined if $\#\{c_j\} = 0$ and not desireable for small $\#\{c_j\}$, because one would estimate $\hat{P}(c_j|q_i, x_i) = 0$ if $\#\{x_i|c_j\} = 0$. As a workaround we set $\hat{P}(x_i|c_j) = \min_k \frac{\#\{x_i|c_k\}}{\#\{c_k\}}$ if it would be zero otherwise for the Naive Bayes model and for the Combined Methods algorithm we fix all $\hat{P}_{ML}(x_i|c_j)$ smaller than 0.05 at 0.03. Third, we find in section 3.2.4 numerical instabilities when we multiply $\hat{P}(c_j) \times \hat{P}(x_i|c_j) \times \hat{P}(q_i|c_j)$ and solve these using logarithms $\exp(\log \hat{P}(c_j) + \log \hat{P}(x_i|c_j) + \log \hat{P}(q_i|c_j))$. While we do not expect these technicalities to change our interpretations, they do explain different numerical result shown in Figures 3.3 and 3.5.

The final estimated correctness probability θ_{lj} is then calculated as

$$\theta_{lj} = \hat{P}(c_j|q_l, x_l) = \frac{\hat{P}(c_j) \times \hat{P}(x_l|c_j) \times \hat{P}(q_l|c_j)}{\sum_{k=1}^{J} \hat{P}(c_k) \times \hat{P}(x_l|c_k) \times \hat{P}(q_l|c_k)} \tag{3.12}$$

Evaluation

Naive Bayes predictions can be obtained from a number of different settings. Figure 3.4 provides an overview over the five different methods we tested for prediction. 'NB 1-answer lambda = 0.7' and 'NB 1-answer' are used to compare different choices for λ. Both methods use only the first verbatim answer and no further variables. Because performance is slightly better for $\lambda = 0.95$, this value is used for the other methods with additional covariates. 'NB 1-answer W Prof. Status' includes information on the professional status and the last two methods make use of the first and second verbatim answers by connecting both answers to a single text in the training data. The difference between 'NB 1-answer W Full Training' and 'NB 2-answers' consists in the fact that the former uses only the first answer to predict job codes and the latter connects first and second answers in the test data like it is done in the training data.

Although we do not recommend evaluating performance from different prediction meth-

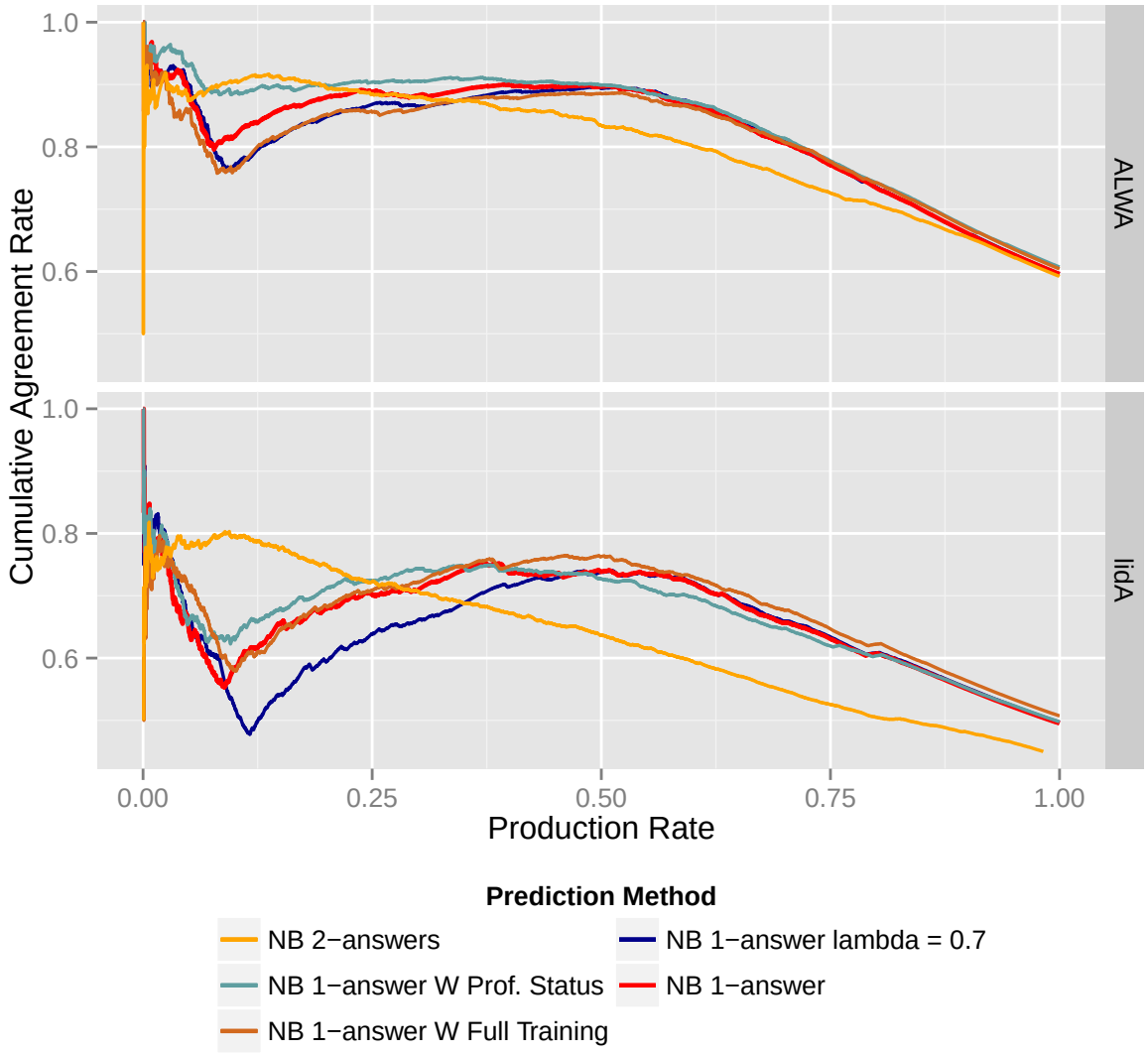

Figure 3.5: Agreement and Production Rates for Different Naive Bayes Procedures

ods using a single number, the AUC is such a number and we provide it for reference. It ranges from its (practical) minimum 0.5 if assignments were made at random to the perfect maximum score 1 signifying that the prediction method can perfectly discriminate between correct and wrong top-ranked category suggestions. Loosely speaking, it measures how good the choices are to find a cutoff point on the scale of estimated correctness probabilities to distinguish between top code suggestions in agreement with human coders and those suggestions that disagree. A detailed discussion about this prediction performance measure is given by Fawcett (2003).

Performance comparison from the different prediction methods is better done with diagrams as depicted in figure 3.5. With the exception of the 'NB 2-answers' method, all curves follow a similar pattern. For very low production rates agreement rates are around 0.9, then they decrease at high gradients before the curves slowly increase back to a 0.9 agreement

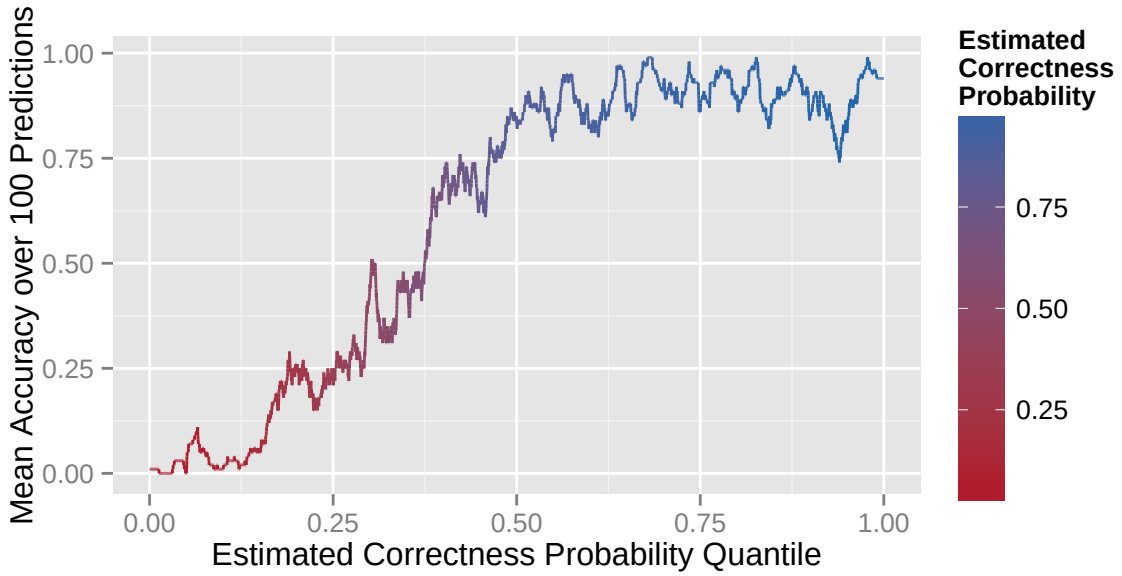

Figure 3.6: Calibration for Naive Bayes First Answer with Professional Status

rate at production rates around 0.5. Close inspection of responses shows that this dent at 0.1 production rate is due to long verbatim answers with multiple words. The algorithm often calculates high estimated correctness probabilities for these answers although the agreement of suggested categories with human-coded categories is often not given. The 'NB 1-answer W Prof. Status' and 'NB 2-answers' methods show that this effect can be avoided when additional covariates are used for prediction. We further observe that agreement rates for most methods are around 0.9 for a 0.5 production rate but rarely above. If one is not willing to accept 10% erroneous codes, no Naive Bayes method is therefore useful for automatic coding. Still, these numbers show that any Naive Bayes method may prove useful for computer-assisted coding.

Lower agreement rates for lidA compared to ALWA suggestions are, again, due to systematic differences in both codes. The comparison shows other peculiarities that we are not able to explain. In particular, the lines for the 'NB 1-answer lambda = 0.7' and 'NB 1-answer W Prof. Status' methods appear to have different characteristics in both data sets. It is also relevant that agreement rates at 100% production rate are nearly identical with one exception: The 'NB 2-answers' method performs worse for lidA predictions. This means that, although with the second verbatim answer more information is entered, the proportion of codes correctly predicted decreases.

Additional insights about strengths and weaknesses of Naive Bayes predictions are provided in figure 3.6. The diagram shows how well estimated correctness probabilities from the 'NB 1-answer W Prof. Status' model align with underlying true probabilities for a code to be correct. Around 10% of the test data has very low estimated correctness probabilities (red) and the suggested codes are - as expected - typically incorrect. Further 40%

have medium estimated correctness probabilities (violet) and as these probabilities rise the suggested codes are also more often the correct ones. For the other half of the data, the prediction method provides estimated correctness probabilities that are all above 0.85, for 1/3 of the data even above 0.95. Still, accuracy for this top-valued third is only 91% and the algorithm systematically overestimates its confidence. Even worse, for this top-half the estimated correctness probabilities do not seem to correlate with true probabilities. Naive Bayes methods are therefore inapplicable for automatic coding in high quality. The Bayesian Categorical method described next will overcome these restrictions.

3.2.3 Bayesian Categorical

Many first answers are short with only one or two words. In the small training data we have available, some of these answers do not appear at all or only a few times. This rareness is problematic, because if answer A_l was coded into the job category c_j only once, then the estimator for $\theta_{lj} = \hat{P}(c_j|A_l)$ will be very imprecise. With the Bayesian Categorical model we tackle this problem. The theory is based on well-known conjugate Bayesian analysis (e.g., Wagner (2010/2011)) with a simple extension described below.

Theory

The approach taken above is frequentist in nature, that is, we try to estimate some underlying "true" value $\hat{\theta}_j = \hat{P}(c_j|q_i, x_i)$ that we wish to be identical with the relative frequency that category c_j occurs. In this section we follow a different path, Bayesian in nature. Probabilities are used to quantify the degree of belief about the parameter θ. The basic Bayesian idea is given with the formula

$$p(\theta|y_1, ..., y_n) = \frac{p(y_1, ..., y_n|\theta)p(\theta)}{p(y_1, ..., y_n)}$$

The posteriori distribution $p(\theta|y_1, ..., y_n)$ is obtained when the likelihood for the observed data $p(y_1, ..., y_n|\theta)$ is multiplied with the prior distribution $p(\theta)$. Using this formula, one updates his current belief about the parameter θ. When no prior information is available, a uniform, "non-informative" distribution is often used for θ. This degree of belief is improved with the new posteriori distribution that reflects new knowledge from the data.

For the coding of occupation, the values $y_1, ..., y_n$ denote the assigned codes for n respondents. All codes are realizations from a categorical distribution $Y = (Y^{(1)}, ..., Y^{(j)}, ..., Y^{(J)})$ with $Y^{(j)} = 1$ if code c_j was assigned and 0 otherwise. A categorical distribution has density

$$p(y^{(1)}, ..., y^{(J)}) = \prod_{j=1}^{J} \theta_j^{y^{(j)}} \tag{3.13}$$

When the likelihood is categorical, a widely used prior is the Dirichlet distribution $(\theta_1, ..., \theta_J) \sim Dir(\alpha_1, ..., \alpha_J)$. Its density is

$$p(\theta_1, ..., \theta_J) = \frac{1}{B(\alpha)} \prod_{j=1}^{J} \theta_j^{\alpha_j} \tag{3.14}$$

where the normalization constant $B(\alpha)$ is the multinomial Beta function. The expected value from the Dirichlet distribution is $\mathbb{E}(\theta_j) = \frac{\alpha_i}{\sum_{k=1}^{J} \alpha_k}$. This choice of a prior allows for a conjugate Bayesian analysis where the posteriori distribution is again a Dirichlet distribution. This is shown by multiplying formulas 3.13 and 3.14,

$$\begin{aligned}
p(\theta_1, ..., \theta_J | y_1, ..., y_n) &\propto p(y_1, ..., y_n | \theta_1, ..., \theta_J) p(\theta_1, ..., \theta_J) && (3.15)\\
&\propto (\prod_{i=1}^{n} p(y_i | \theta_1, ..., \theta_J)) p(\theta_1, ..., \theta_J) && (3.16)\\
&\propto (\prod_{i=1}^{n} \prod_{j=1}^{J} \theta_j^{y_i^{(j)}}) \prod_{j=1}^{J} \theta_j^{\alpha_j} && (3.17)\\
&\propto \prod_{j=1}^{J} \theta_j^{\sum_{i=1}^{n} y_i^{(j)} + \alpha_j} && (3.18)
\end{aligned}$$

which is the kernel from the Dirichlet distribution, $(\theta_1, ..., \theta_J) | y_1, ..., y_n \sim Dir(\sum_{i=1}^{n} y_i^{(1)} + \alpha_1, ..., \sum_{i=1}^{n} y_i^{(J)} + \alpha_J) = Dir(\#\{c_1\} + \alpha_1, ..., \#\{c_J\} + \alpha_J)$. As above, $\#\{c_j\}$ denotes here the number of anwers coded into category c_j. When not the full posteriori distribution is required but only an estimator for c_j, a good choice is often the posteriori expectation $\hat{\theta}_j = \hat{P}(c_j) = \mathbb{E}(\theta_j | x_1, ..., x_n) = \frac{\#\{c_j\} + \alpha_j}{\sum_{k=1}^{J} \#\{c_k\} + \alpha_k}$.

Our choice to use the Dirichlet prior is favorable for a number of reasons: To calculate the posteriori we only need to count and add values, which makes computation simple. When no prior information is available, one may set $\alpha_1 = ... = \alpha_J$ and thus no category is expected to be more probable beforehand. The parameters α_j also have a simple interpretation: Because they are added to the number of observed categories $\#\{c_j\}$, α_j may be regarded as the number of categories that we have observed in prior (imaginary) studies. This interpretation is also supported by the equation

$$\begin{aligned}
\mathbb{E}(\theta_j | x_1, ..., x_n) &= \frac{\#\{c_j\} + \alpha_j}{\sum_{k=1}^{J} \#\{c_k\} + \alpha_k} && (3.19)\\
&= \omega \frac{\alpha_j}{\sum_{k=1}^{J} \alpha_k} + (1 - \omega) \frac{\#\{c_j\}}{n} && (3.20)
\end{aligned}$$

with $\omega = \frac{\sum_{k=1}^{J} \alpha_j}{n + \sum_{k=1}^{J} \alpha_j}$. It shows that the posteriori expectation is a weighted mean from the prior expectation and the relative frequencies in the observed data. The prior has

therefore a shrinkage effect, i.e. relative frequencies in the data are drawn towards the prior expectation. Also note that, when larger numbers are chosen for the parameters α_i, the prior information has stronger effect on the posteriori expectation and, in contrast, larger sample sizes n strengthen the importance of observed data.

The discussion above shows that a Bayesian categorical model is adequate to estimate category probabilities $\hat{\theta_j} = \hat{P}(c_j)$. Verbatim answers, however, have not been used so far for prediction. Next, we will extend the method to use covariate information and calculate estimated correctness probabilities $\hat{\theta_j} = \hat{P}(c_j|q_i)$. The idea is to train the model using only a subset from all the coded persons, namely we choose those codes y_i where the verbatim answer given is exactly identical to the answer we try to predict, named q_i above. In other words, instead of using code frequencies from all observations, $\#\{c_j\}$, the likelihood is formed from code frequencies $\#\{c_j|q_i\}$ where the given answers are identical.

Another question is how to choose the prior parameters $\alpha_1, ..., \alpha_J$. While identical α are reasonable to express no prior knowledge, we prefer to use relative frequencies for the different categories, $\#\{c_1\}/n, ..., \#\{c_J\}/n$. Due to the high number of categories, we expect that relative frequencies are all very low and thus nearly identical. Because relative frequencies sum up to 1, prior knowledge has an impact on the final result as if exactly one additional person was asked about their job code. For answers that were coded many times, this is negligible, but when an answer was only coded a single time into category c_j, it is relevant. In this case, the posteriori expectation evaluates to, with slight abuse of notation,

$$\mathbb{E}(\theta_j|\#\{c_j|q_i\} = 1, \#\{c_j\} = 1) = \frac{\#\{c_j|q_i\} + \#\{c_j\}/n}{\sum_{k=1}^{J}(\#\{c_k|q_i\} + \#\{c_k\}/n)} \tag{3.21}$$

$$= \frac{1 + 1/n}{1 + 1} \tag{3.22}$$

$$\approx 0.5 \tag{3.23}$$

and setting $\sum_{k=1}^{J} \#\{c_k\}/n = 1$ has clearly a huge impact on the final result. To allow for more flexibility in prior assumptions, we multiply the prior relative frequencies suggested above with a constant α. This number describes, on how many imaginary persons we build our prior beliefs. The best choice for α will be discussed in the following section.

To summarize, the Bayesian categorical approach provides us with a Dirichlet distribution over the probabilities $\theta_1, ..., \theta_J$ that associated categories $c_1, ..., c_J$ are correct, given an answer q_i from the respondent. The distribution parameters are given in the equation

$$(\theta_1, ..., \theta_J)|(y_1, q_1 = q_i), ..., (y_n, q_n = q_i) \sim \tag{3.24}$$

$$Dir(\#\{c_1|q_i\} + \alpha\#\{c_1\}/n, ..., \#\{c_J|q_i\} + \alpha\#\{c_J\}/n) \tag{3.25}$$

Hereby, the terms $(y_1, q_1 = q_i), ..., (y_n, q_n = q_i)$ denote that only respondents that gave

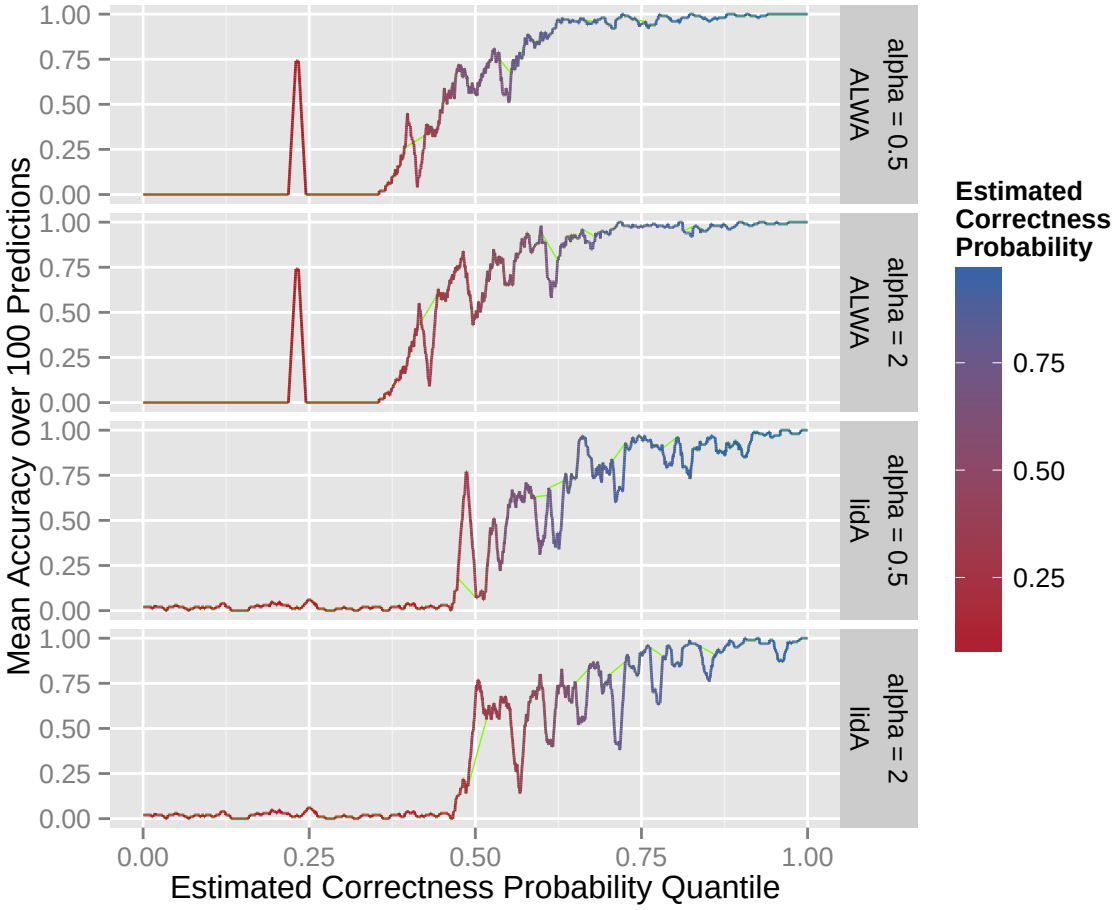

Figure 3.7: Calibration for Bayesian Categorical models

the desired answer q_i are used for estimation in the likelihood. Relevant estimators like the posteriori expectation can easily be calculated from this distribution. Noteworthy is also the aggregation property from the Dirichlet distribution. When only the distribution over a single parameter θ_j is of interest, this parameter follows a Beta distribution with parameters

$$\theta_j|(y_1, q_1 = q_i), ..., (y_n, q_n = q_i) \sim Beta(\#\{c_j|q_i\} + \alpha\#\{c_j\}/n, \tag{3.26}$$

$$(\sum_{k=1}^{J} \#\{c_k|q_i\} + \alpha\#\{c_k\}/n) - (\#\{c_j|q_i\} + \alpha\#\{c_j\}/n)) \tag{3.27}$$

Evaluation

The good performance of the Bayesian Categorical model has been shown in figure 3.3 and it is a method well suited to find answers that shall be coded automatically without human supervision at high agreement rates. Our suggestion is to set the prior parameter $\alpha = 0.5$. With this choice the AUC equals 0.963 for the ALWA test data which is considerably higher than the AUC from any Naive Bayes model. This section will provide some insights how this good performance is reached and why we choose to set $\alpha = 0.5$.

Relevant properties how good the Bayesian Categorical model with $\alpha = 0.5$ can predict job categories from the ALWA test data can be seen in the top panel from figure 3.7. For 35% of all answers (the quantile on the x-axis) an identical answer is not available in the training data. Without any information, the most frequent job category 71402 is predicted with an estimated correctness probability at 0.054. In general, these predictions are false and accuracies are zero. A substantial amount of answers is human-coded into this category by chance, which explains the peek at 0.23. Further 27% of the test answers have already been coded in the training data but into numerous different categories. The estimated correctness probabilities for these answers are between 0.07 and 0.85 which depends on relative frequencies how often specific answers were coded into a single job category. Answers like "Wissenschaftlicher Mitarbeiter" or "Technischer Angestellter", for example, have been coded into numerous different job categories and thus the algorithm expresses its uncertainty about the correct code with low estimated correctness probabilities. When the algorithm estimates correctness probabilities above 0.85, few suggested categories are wrong and one could assign the predicted code automatically. With this cutoff point, automatic coding would be possible at a production rate of 38% and an agreement rate (y-axis) around 97%. This result is promising at first sight but there are two drawbacks. First, the comparison with the lidA test data shows that human coders find other job codes for lidA even if code assignments for a unique first answer in ALWA are nearly definite. Second, these numbers are not better than the dictionary-based automatic coding that was done for original ALWA coding and it is well possible that our method simply reconstructs the results from ALWA automatic coding.

It must be noted that a few very frequent answers determine what the graph looks like. This is best seen in the diagram for lidA test data with $\alpha = 2$ where five green lines are best visible. Each line shows how the graph would look different if certain responses were not given. The high amplitude at the 0.5 quantile is due to 82 answers "Verkäuferin" (estimated correctness probability = 0.30) where the suggested category is in fact accurate at 85%. Other large deviations in the graph result from the answers "Sachbearbeiterin" (estimated correctness probability = 0.67, 33 answers with 0% accuracy), "Kaufmännische Angestellte" (estimated correctness probability = 0.79, 52 answers with 2% accuracy), "Lagerist" (estimated correctness probability = 0.87, 27 answers with 0% accuracy), and "Verwaltungsangestellte" (estimated correctness probability = 0.91, 62 answers with 71% accuracy).

A prediction method is better if the algorithm clearly distinguishes between answers that can be coded automatically and those that can not. High and many amplitudes in the diagram represent unpredictibility as opposed to those parts in the diagram that are stable and correctness is therefore simple to predict. The diagram shows therefore that the algorithm's performance with prior parameter $\alpha = 0.5$ is clearly better compared with

$\alpha = 2$. Our choice for $\alpha = 0.5$ has the following motivation. For answers that appear only a single time in the training data and are given again in the test data, the relative frequencies that assigned codes agree in training and test data are 78% (ALWA) resp. 63% (lidA). When such answers are predicted with the Bayesian Categorical model one would want estimated correctness probabilities to have similar values. These estimated correctness probabilities are posteriori expectations that evaluate to

$$\mathbb{E}(\theta_j|\#\{c_j|q_i\} = 1, \#\{c_j\} = 1) = \frac{\#\{c_j|q_i\} + \alpha\#\{c_j\}/n}{\sum_{k=1}^{J}(\#\{c_k|q_i\} + \alpha\#\{c_k\}/n)} \approx \frac{1}{1+\alpha} \tag{3.28}$$

when $\#\{c_j|q_i\} = 1$ and $\#\{c_j\} = 1$ are given. Setting $\mathbb{E}(\theta_j|\#\{c_j|q_i\} = 1, \#\{c_j\} = 1) = 0.78$ and solving for α yields $\alpha = \frac{1-0.78}{0.78} = 0.28$ as the optimal value to predict new codes in the ALWA test data. For lidA one calculates $\alpha = \frac{1-0.63}{0.63} = 0.59$. Our choice $\alpha = 0.5$ is a conservative center point from both calculations that can be interpreted as a prior belief that $\frac{2}{3}$ of all answers that were coded once in the training data will get the same code in the test data.

3.2.4 Combined Methods (Boosting)

Over the last sections we explored a number of different methods that can be used to find adequate job categories. Different dictionaries exist and can be consulted to find the correct code for a given answer. With the Naive Bayes and Bayesian Categorical models, we suggested two probabilistic algorithms that use ALWA training data for automatic coding. Figure 3.3 and the discussion above have shown that these algorithms have different strengths. The Bayesian Categorical model is useful for short answers where identical answers were already coded in the training data. The Naive Bayes method reaches only lower agreement rates but gives reasonable category suggestions for a larger proportion of answers. Not identical answers but identical words in non-identical answers are the engine for this. We have further seen that there is a relevant proportion in the test data where no adequate code suggestions are found due to the limited size of the training data. Especially for these cases, we expect that rule-based coding from a dictionary will provide additional category suggestions for automated coding. In this section we suggest a method to combine the different algorithms described before. This allows better performance when the different strengths from all procedures are combined. It is easily possible to construct other methods that may be useful for coding. The question is then, if this new method complements existing methods in a helpful way or if it is useless. Within this section we will describe a possible way to evaluate predictive performance from different coding methods.

Central for this section is the following idea. All different coding methods m calculate scores (we called them estimated correctness probabilities before) $\theta_{lj}^{(m)}$ for each response l and all possible job categories j. As already noted in section 3.2, these scores are expected

c_j	c_j correct	$Score_{lj}^{(1)}$	$Score_{lj}^{(2)}$	$\cdots$
$c_1 = 01104$	FALSE	$\theta_{l,1}^{(1)}$	$\theta_{l,1}^{(2)}$	$\cdots$
$c_2 = 01203$	TRUE	$\theta_{l,2}^{(1)}$	$\theta_{l,2}^{(2)}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$c_{1290} = 99999$	FALSE	$\theta_{l,1290}^{(1)}$	$\theta_{l,1290}^{(2)}$	$\cdots$

Figure 3.8: Exemplarious data frame for person l with correct job category $c_j = 01203$

to correlate with the true probability $P(c_j|l)$ that category c_j is correct for respondent l. We now build for each respondent a data frame with $J = 1290$ rows for the different job categories. With each row j it is suggested that job category c_j is correct. When for a person l the correct category is known this is inserted into the data frame. This variable, "c_j correct", is the target variable for the following models. All scores form different models, $\theta_{lj}^{(m)}$, are also included into the data frame and will serve as covariates. In this section we will then try to predict the binary variable "c_j correct" given all the scores from all different methods, $\theta_{lj}^{(m)}$. Powerful algorithms for binary classification are available. An exemplary representation of this data frame for one person is given in figure 3.8.

To train the model it is not sufficient to use only one person with exactly one correct category, but the same training data as before is used. For each person we calculate a data frame as described above and bind all the different data frames together. The training data used before consists of $n = 26297$ answers and thus the new training data has $26297 * 1290 = 33923130$ rows with $\sum c_j\text{correct} = 26297$. A problem arises in the fact that one might use training data twice: a first time to find scores $\theta_{lj}^{(m)}$ and a second time to fit a model that predicts "c_j correct". Associations between scores $\theta_{lj}^{(m)}$ and the target variable "c_j correct" would therefore be different for the training and the test data which is clearly not desired. Instead, we predict all scores $\theta_{lj}^{(m)}$ without usage of verbatim answers from person l. This means that, when we used before the frequency for job category c_j given a verbatim answer q_i, $\#\{c_j|q_i\}$, we now subtract 1 from these frequencies, $\#\{c_j|q_i\} - 1$, to calculate scores for the training set.

In prior sections we had to calculate a single score $\theta_{lj}^{(1)}$ that we hoped to be correlated as close as possible with the true probability $P(c_j|l)$. With the new method described in this section, a multitude of different scores $\theta_{lj}^{(1)}, ..., \theta_{lj}^{(M)}$ may be used for prediction and many scores that reflect different structures in the data should improve the final prediction. We construct a number of additional scores. An overview over all scores is given in figure 3.9.

Most scores require that the exact first answer matches perfectly with previous answers from the training data or with dictionary entries. The exact word sequence is therefore the input for these score construction methods. When first answers are more complex and consist of multiple words, identical word sequences are often not found in the dictionary. In this case, it may be helpful to find a useful substring to feed into the algorithm. When

Name	Description
numVerzeichnisBerufsben	Number of dictionary entries from the alphabetic dictionary that suggest category c_j (exact and partial matches)
phraseNumVerzeichnisBerufsben	(phrase) Number of dictionary entries from the alphabetic dictionary that suggest category c_j (exact and partial matches)
numExactSuchwort	Number of dictionary entries from the search word dictionary that suggest category c_j (only exact matches)
phraseNumExactSuchwort	(phrase) Number of dictionary entries from the search word dictionary that suggest category c_j (only exact matches)
numPartialSuchwort	Number of dictionary entries from the search word dictionary that suggest category c_j (exact and partial matches)
phraseNumPartialSuchwort	(phrase) Number of dictionary entries from the search word dictionary that suggest category c_j (exact and partial matches)
ALWAfrequencies	Number of identical answers in ALWA training data that were coded into category c_j (only exact matches)
phraseALWAfrequencies	(phrase) Number of identical answers in ALWA training data that were coded into category c_j (only exact matches)
posterioriExpectation	Posteriori expectation (= estimated correctness probability) from Bayesian Categorical model for category c_j
phrasePosterioriExpectation	(phrase) Posteriori expectation (= estimated correctness probability) from Bayesian Categorical model for category c_j
postProb0point5	Posteriori probability $P(\theta_j > 0.05)$ from Bayesian Categorical model for category c_j
phrasePostProb0point5	(phrase) Posteriori probability $P(\theta_j > 0.05)$ from Bayesian Categorical model for category c_j
NBprob	Probability for category c_j (= estimated correctness probability) from Naive Bayes model using the first answer and the professional status
beruflicheStellung	Professional status x_l from person l
freqBeruflicheStellung	Number of identical professional status in ALWA training data that were coded into category c_j
numSuggestedCategories	Number of suggested categories for person l

Figure 3.9: Variables Used for the Combined Methods model

Possible Phrases	Frequency in ALWA	Relative Frequency in best Category	Frequency in best Category (product)
GARTEN UND LANDSCHAFTSBAU BETRIEBSLEITER	5	1	5
GARTEN UND LANDSCHAFTSBAU	8	0.875	7
GARTEN UND	0	NA	0
GARTEN	0	NA	0
UND LANDSCHAFTSBAU BETRIEBSLEITER	0	NA	0
UND LANDSCHAFTSBAU	0	NA	0
UND	0	NA	0
LANDSCHAFTSBAU BETRIEBSLEITER	0	NA	0
LANDSCHAFTSBAU	2	1	2
BETRIEBSLEITER	0	NA	0

Figure 3.10: Exemplarious Phrase Identification for Verbatim Answer "Garten- und Landschaftsbau Betriebsleiter". "GARTEN UND LANDSCHAFTSBAU" becomes Input Phrase

methods use only the substring instead of the full answer it is indicated with the keyword "phrase" in figure 3.9. To find a useful phrase, we calculate for all single words in the answer and all combinations of successive words how frequent these possible phrases are in the ALWA data and how good they align to a specific code. Input for the algorithm is then the phrase where the product of frequency and code alignment is maximal. Figure 3.10 provides an example how the input phrase is calculated.

When we build for each person in the training data a new data frame with $J = 1290$ rows and bind all these data frames together, the resulting data frame is with 33923130 rows quite large. In fact, computer performance restrictions make it necessary to reduce its size. At the same time, many of the J category suggestions are not helpful at all because not a single score $\theta_{lj}^{(m)}$ indicates that category c_j could be correct for person l. For example, if the answer from person l is "nurse", many health job categories are meaningful code suggestions but job categories from gardening and floristry are not helpful. We then keep only those rows in the data frame, where at least one entry from one dictionary or the ALWA training data suggests that this categtory could be correct (i.e., for at least one $m = 1, ..., 8$ is $\theta_{lj}^{(m)} > 0$ for the top eight variables in figure 3.9) and drop all other rows. Category suggestions are also kept if $\hat{P}(q_l|c_j) > median_j(\hat{P}(q_l|c_j))$ in the notation from equation 3.5. After dropping all irrelevant category suggestions, the remaining category suggestions are counted and the number is saved in the variable *numSuggestedCategories*. This number may be helpful to predict the correct category because the probability for an entry from a dictionary to be

correct increases when only few or no other categories are suggested.

The task is now to estimate category correctness, named "c_j correct" in figure 3.8 which is a binary response. We described numerous covariates that are correlated by construction. Many different algorithms have been implemented into standard software and may be used for this task. First, we tried the Breiman's random forest algorithm which was implemented by Liaw and Wiener (2002) into an R-package with the same name. Variable importance was calculated and suggested that the covariates `beruflicheStellung`, `freqBeruflicheStellung`, `posterioriExpectation`, `postProb0point5`, `phrasePosterioriExpectation`, `phrasePostProb0point5`, `NBprob`, and `numSuggestedCategories` have higher relevance for prediction than the other variables. In the end, we were not satisfied with random forests for the following reasons: The `random forest`-package only returns frequencies how many trees vote for or against a specific outcome but results for probabilistic interpretation are not provided. Another R-package is `randomForestSRC` from Ishwaran and Kogalur (2013) which gives the required output but long calculations on large training data to make very few new predictions prohibit its usage for interactive occupation coding. Because we did not find a random forest implementation that fits our purpose we resort to boosting which will be described in the rest of the section.

Theory

Our choice is to use gradient boosted trees as implemented in the R-package `mboost` because trees allow for high degrees of interaction between different covariates and because it is possible to estimate probabilities that category c_j is correct. Here we give only a very brief review on the most relevant properties and suggest for a more thorough introduction the chapters nine and ten from Hastie et al. (2009). Our presentation follows the style of Hofner et al. (2012) who give a tutorial for the `mboost`-package.

Let y be the response variable ("c_j correct" here) and x a vector of covariates. Boosting aims to estimate the optimal prediction function

$$f^* := \arg\min_f \mathbb{E}_{y,x}(\rho(y, f(x^T))) \tag{3.29}$$

that minimizes the expected loss ρ between the true values y and predicted values $f(x^T)$. Because our response variable is binary, we choose the negative binomial log-likelihood as a loss function which is also used for logistic regression models. In practice it is necessary to approximate the expectation above with the observed mean $f^* :\approx \arg\min_f \sum_{i=1}^n \rho(y_i, f(x_i^T))$.

The final boosting estimation is calculated as $\hat{f} = f^{[0]} + \sum_{m=1}^{m_{stop}} \nu \hat{u}^{[m]}$. Starting with some initial value for $\hat{f}$, $f^{[0]}$, the algorithm iteratively estimates base learners $\hat{u}^{[m]}$ that reduce the loss between true values y and current predictions $f^{[0]} + \sum_{k=1}^{m-1} \nu \hat{u}^{[k]}$. These base learners can be any type of model and we use conditional inference trees (see Hothorn et al.

(2006)) for this. The hyperparameter $\nu \in (0, 1)$ controls the step size what impact single base learners have and thus the speed how fast the predictions improve. While it has been shown that the exact value for ν is of minor relevance, it is generally recommended to use small values for ν such that the algorithm does not overshoot the optimal solution. With smaller values for ν the number of iterations m_{stop} grows until the algorithm reaches a good solution. m_{stop} is a major tuning parameter that needs careful evaluation. If it is chosen too small, the model is not yet well fitted to the data. On the other hand, with large m_{stop} overfitting is likely to occur.

We use trees as base learners to allow for complex interactions within the covariates. When fitting trees, the desired tree size needs to be set in advance which is another hyperparameter. Hastie et al. (2009) argue in the context of boosting that trees chosen too large will yield less accurate predictions. They further state that the maximal number of interacting covariates is directly given by (tree size - 1). Because we suspect many interactions between different covariates in our data the tree size must not be too small. We applied bootstrapping with 3 folds to find optimal hyperparameters m_{stop}, ν, and tree size.

Evaluation

The large size of the training data with 820340 rows gives rise to practical problems when gradient boosted trees are fitted. Due to limited memory space it proved impossible to run as many iterations until the perfect stopping point m_{stop} is found. Therefore, we used only a random sample of 600000 observations for training and stopped after $m_{stop} = 31$ iterations. Beforehand we tested different hyperparameters to find an optimal combination using a small random sample of only 30000 observations. Our final parameters are tree size $= 7$ and $\nu = 0.6$ where the large number for ν reflects that large step sizes are necessary to reach a good fit after only 31 iterations. The resulting model is then used to calculate correctness probabilities for all suggested categories. This allows ordering of job categories with most probable categories first, which is useful for computer-assisted coding.

For automatic coding and for a better presentation of result we select only the top-suggested category for each answer. These are merged with those answers that were not included in the training data because the verbatim answers provided no code suggestions. Now, we work again with the original training data consisting of $n = 26297$ answers. For each answer the top-ranked category suggestion is included as well as the corresponding covariates from figure 3.9. We further add one covariate that gives the estimated correctness probability from boosted trees and a further binary covariate to indicate if an answer was found in the large training data described above (equivalent to *numSuggestedCategories* > 0). If the answer was not included before and only merged into the small training data, it will be impossible to find the correct job category for it.

Again we use gradient boosted trees to decide if the suggested category is correct (meta-

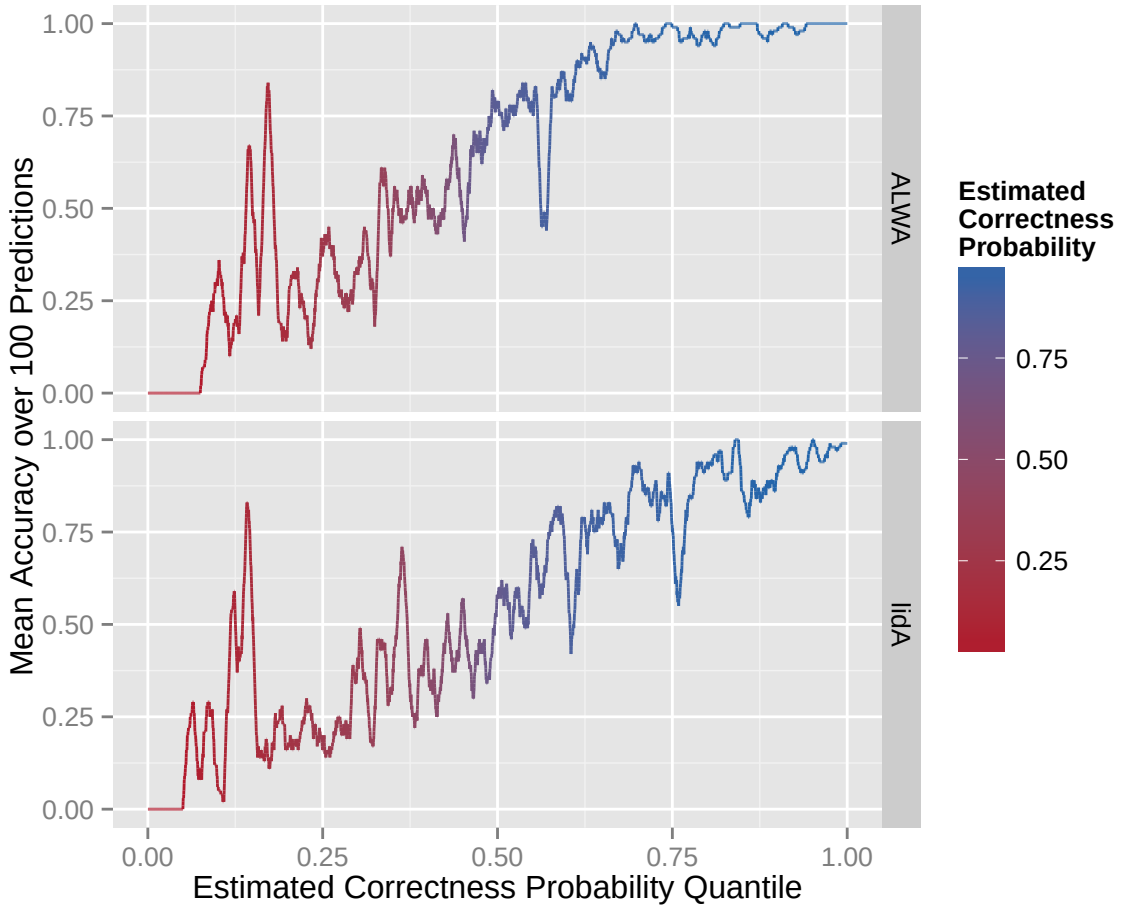

Figure 3.11: Calibration for Boosting Method

classification). With the smaller training size we use the full training data and experiment with different parameter values until we find optimal hyperparameters $m_{stop} = 241$, $\nu = 0.07$ and maxdepth $= 11$ which yields the minimum Cross-validated Negative Binomial Likelihood equal to 0.2542. Our final estimated correctness probabilities are predictions from this model. The good performance from the boosting method already have been described in the context of figure 3.3 and further details are provided next.

Figure 3.11 is best compared to figures 3.6 and 3.7 to see similarities and differences for all proposed methods. On the left side of the graph we find those answers with lowest estimated correctness probabilities where no categories were suggested and automatic coding is not useful. This is the case for 8.23% of all answers (ALWA, lidA: 5.76%), a proportion a few percent points lower than what we observed for the Naive Bayes method. In the middle part there is a large proportion of answers that may be correct or may not and must be referred to a human coder. As has been described in section 3.2.3, only a few very frequent answers are probably responsible for what this center part of the graph looks like. Answers where the correct code can be determined mostly automatic are on the right side. For ALWA, a third of all answers with highest estimated correctness probability (all above

0.9465) reaches 98.04% accuracy which is competitive with the Bayesian Categorical model for automatic coding (lidA: top 26% are above this threshold and have overall accuracy of 90.11%). Systematic coding differences in ALWA and lidA are once again to be blamed for worse predictions on the lidA test data. The AUC equals 0.888 for the ALWA test data, which is slightly better than the AUCs from Naive Bayes models. The AUC from the Bayesian categorical model at 0.963 is superior because that model finds a clear decision boundary between those answers practical for automatic coding and those that are not. In fact, the AUC performance measure punishes the boosting method because it suggests more, but probably incorrect answers. The AUC metric is therefore not helpful here.

The starting point for this section was the idea to combine strengths from different methods into a single better model. How did we succeed? Figure 3.3 shows that the Combined Method has not highest agreement rates for all production rates but is always close to it. The combination of methods is therefore a success. Still, we cannot recommend it for all purposes and the computational requirements are such that the simpler techniques may be preferred. For automatic coding with the desired high agreement rates there is no clear evidence, if the Bayesian categorical model or the boosting model is preferable. This is similar for computer-assisted coding. When all answers are considered, agreement rates from the boosting model and Naive Bayes are nearly identical and thus it is unknown which model is more useful for this task. Only the use of multiple dictionaries leads to fewer answers that have no job category suggested. This makes us recommend the Combined Model for computer-assisted coding.

Chapter 4

A Prototype for Computer-Assisted Coding

A few decades ago, before every office was equipped with a computer, coding clerks had to thumb through printed classifications and alphabetic dictionaries to find the desired job code. Today, computers are omnipresent and in sections 2.3 and 2.4.1 we mention various programs available for computer-assisted coding. Statistical agencies have developed own software to meet their special requirements for large-scale classifications like employment (e.g., Tourigny and Moloney (1997), U.S. Census Bureau (2009), Svensson (2012)). In Germany, the Federal Employment Agency provides two online tools with similar functionality but not tailored to code thousands of answers. Both tools list all possible jobs from the DKZ after a search string is entered.[1]

The result list from both tools is often quite large and not perspiciuous at first glance. The algorithms we developed produce relief. We can order the results with the most relevant job codes first. Figure 4.1 gives the output from our system for the exemplarious verbatim answer "Fleischer" ("butcher") and further examples are provided in the appendix. This list can then be used for computer-assisted coding. For full comprehension of this figure it is necessary to point out a number of details:

- On the top is the verbatim answer for the first employment question. Right next to it we see the "phrase" which is the most meaningful substring from the original answer and was calculated automatically. Both the answer and the phrase are used independently to find possible job categories. The last entry from the first line is the job code that professional coders have assigned to the answer. Of course, for computer-assisted coding this code will not be available.

- The categories shown are selected with the Combined Method. This means that ALWA training data and multiple dictionaries are searched for full and partial matches with

[1] Online at http://bns-ts.arbeitsagentur.de/ and http://berufenet.arbeitsagentur.de/dkz/

```
Eingegebener Beruf: FLEISCHER | Phrase: FLEISCHER | Coded: 29232
--------------------------------------------------------------------------------
    Lebensmittel- & Genussmittelherstellung : ( 35 Antworten in ALWA;
    Corr. Prob. =  1.002298 )
--------------------------------------------------------------------------------
~~~~~~~~~
Berufe Lebensmittelherstellung (o.Spez.)
29201 ..... z.B.: Helfer/in - Lebensmittelherstellung
~~~~~~~~~
Berufe Fleischverarbeitung
29232 ..... z.B.: Fleischer/in || Fleischer/in
29233 ..... z.B.: Techniker/in - Lebensmitteltechnik (Fleischereitechnik)
~~~~~~~~~
Aufsichts- & Führungskr.-Lebensmittel- & Genussmittelherst.
29293 ..... z.B.: Fleischermeister/in
--------------------------------------------------------------------------------
    Verkauf von Lebensmitteln : ( 0 Antworten in ALWA; Corr. Prob. =  0.009152703 )
--------------------------------------------------------------------------------
~~~~~~~~~
Berufe Verkauf von Fleischwaren
62322 ..... z.B.: Gewerbegehilfe/-gehilfin - Fleischerhandwerk ||
                  Fachverkäufer/in - Nahrungsmittelhandw.(Fleischerei)
--------------------------------------------------------------------------------
    Verkauf (ohne Produktspezialisierung) : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.007885854 )
--------------------------------------------------------------------------------
~~~~~~~~~
Aufsichts- & Führungskr.-Verkauf
62194 ..... z.B.: Verkaufsleiter/in im Nahrungsmittelhandwerk
--------------------------------------------------------------------------------
    Unternehmensorganisation & -strategie : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.006899456 )
--------------------------------------------------------------------------------
~~~~~~~~~
Berufe Unternehmensorg.&-strat.(s.spez.Tät.)
71383 ..... z.B.: Betriebswirt/in (Fachschule) - Vieh und Fleisch
```

Figure 4.1: Algorithm output for answer 'Fleischer'

the given answer. If the algorithm finds any indication that a category might be correct it is presented to the human coder.

- For a fast overview over the possible codes, we have sorted the output with most probable job codes first. Estimated correctness probabilities from the Combined Method are used for sorting. For each job code, this is a number between 0 and 1 but it is not enforced that they sum up to 1 for all job codes. We also report how often an identical answer was coded into each category in the ALWA training data.

- A complete job code according to the German employment classification is always a five-digit number. In the example, the "Helfer/in - Lebensmittelherstellung" is one job within the job category "29201". Each job category pools multiple jobs. For intuitive understanding which jobs belong into a category, we generate automatically for each category 1-3 exemplarious job names from DKZ dictionaries (see code 62322). The official category name is not written down explicitly but can be derived implicitly with background knowledge.

- All job categories have similarities to each other and our result presentation arranges them accordingly. The first three digits from a code specify the so-called "Berufsgruppe". For the "Helfer/in - Lebensmittelherstellung" this is the code "292" named "Lebensmittel-& Genussmittelherstellung" (digits are not explicitly written down). Similarly, the first four digits define the "Berufsuntergruppe", which is "Berufe Lebensmittelherstellung (ohne Spezialisierung)" in our example. Another job, the "Fleischer/in" has been classified into the same Berufsgruppe but a different Berufsuntergruppe named "Berufe Fleischverarbeitung". Within this Berufsuntergruppe, we see the meaning of the fifth digit that reflects the skill level for a job. Auxiliary and semiskilled occupations have been assigned to categories where the last digit is a "1", specialized occupations have a "2" in their last digit, complex occupations for specialists a "3", and highly complex occupations a "4". The official name for a five-digit "Berufsgattung" can then be derived from the name of the Berufsuntergruppe and the last digit. For example, the "Helfer/in - Lebensmittelherstellung" is one job in the category named "Berufe Lebensmittelherstellung (ohne Spezialisierung) - Helfer-/Anlerntätigkeiten" and the "Fleischer/in" is in the Berufsgattung "Berufe in der Fleischverarbeitung - fachlich ausgerichtete Tätigkeiten".

Chapter 5

Conclusion and Perspectives

Although most surveys avoid asking open-ended questions when possible, closed questions are not always feasible. Occupation is one example that is typically asked with open-ended questions and statistical agencies around the world struggle to code the verbatim answers into large coding schemes with 100s of categories at low costs and high quality. In this thesis, we have summarized the international literature on coding with a focus on the coding of German occupations. The use of technology is widespread for computer-assisted and automatic coding but the algorithms behind differ in many aspects. Some agencies continuously monitor the coding quality from professional coders and computer systems. Though, reported quality measures from the coding of occupation vary strongly within Germany and worldwide and research into the causes and possible ways for improvement is only at the beginning.

A central objective for this thesis was to develop supervised learning algorithms that use coded answers from prior studies to predict new codes. Data from two surveys were used to test the different methods. We have shown that systematic differences in ALWA and lidA codes explain why our algorithms perform worse if lidA codes are predicted from ALWA training data. Moreover, the limited size of the ALWA data forms an obstacle that we can elude only partly. When answers have not been coded before it is impossible to find the correct code from training data. With large training data, supervised learning methods have been applied before for automated coding. We suggest the Bayesian Categorical model to account for higher uncertainty about the correct code when answers were only given a few times before. Promising coding results are obtained from our small training data and even better performance is reached when different dictionary and two data-based coding approaches are combined. If more training data were available, we expect additional improvements. Before our methods facilitate coding in practice, we recommend finding larger training data where good coding quality is known.

Training data from one survey can be useful for another survey in a number of ways. When both data sets have answers already coded, the χ^2-statistic may be used to find

systematic differences in both data sets. The cheapest way to code new answers is with automatic coding. The proposed Bayesian Categorical model is able to code 38% of all answers at 3% error rate without human interaction. This is competitive with the dictionary-based method from Drasch et al. (2012) who report a production rate around 39%. Even more useful might be our prototype for computer-assisted coding where the computer suggests the codes most probable and a human coder decides which one is correct. When information from dictionaries and training data is combined, for 74% of all answers the correct category is provided within the top five code suggestions. This allows the coding clerk to decide within seconds on the correct code and only the residual 26% are more laborious. These performance measures are calculated for the situation when only the first verbatim answer and the professional status are used to predict new codes. Typically, additional helpful covariates are available like a second verbatim answer or the employer's industry. The Naive Bayes method and the Combined Method we proposed provide intuitive ways to include such information and thus better performance can be expected.

A good system for computer-assisted coding is not only helpful for the omnipresent back office coding but offers also new opportunities. A sample of the coded answers could be forwarded automatically to a second human coder for control. This would allow one to monitor coding quality automatically and to take action for continuous improvement (c.f. Svensson (2012)). Another strand for future research is the development of better classification algorithms. Literature on text classification is abundant and we expect special problems within this area to be relevant for automated coding. Related keywords are deep learning (e.g. Bengio et al. (2012)), the classification of short (e.g. Romero et al. (2013)) and noisy (e.g. Agarwal et al. (2007)) text, language models (e.g. Liu and Croft (2004)), and hierarchical classification (e.g. Silla and Freitas (2011)). One may also try to soften the Naive Bayes (e.g. Peng et al. (2004)) assumption or to combine ideas from the Naive Bayes and the Bayesian Categorical model to obtain a single model. The next step in our research is less ambitious. We plan to use computer-assisted coding techniques during the interview and evaluate its quality.

Bibliography

Agarwal, S., Godbole, S., Punjani, D. and Roy, S. (2007). How Much Noise Is Too Much: A Study in Automatic Text Classification, *2007 Seventh IEEE International Conference on Data Mining*, 2007 Seventh IEEE International Conference on Data Mining, icdm, pp. 3–12.
URL: *http://doi.ieeecomputersociety.org/10.1109/ICDM.2007.21*

Aggarwal, C. and Zhai, C. (2012). A survey of text classification algorithms, *in* C. C. Aggarwal and C. Zhai (eds), *Mining Text Data*, Springer US, pp. 163–222.
URL: *http://dx.doi.org/10.1007/978-1-4614-3223-4_6*

Antoni, M., Drasch, K., Kleinert, C., Matthes, B., Ruland, M. and Trahms, A. (2010). Arbeiten und Lernen im Wandel * Teil 1: Überblick über die Studie, *FDZ-Methodenreport 05/2010*, Forschungsdatenzentrum der Bundesagentur für Arbeit im Institut für Arbeitsmarkt- und Berufsforschung, Nuremberg.
URL: *http://fdz.iab.de/de/FDZ_Individual_Data/ALWA.aspx*

Bengio, Y., Courville, A. C. and Vincent, P. (2012). Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives, *CoRR* **abs/1206.5538**.
URL: *http://arxiv.org/abs/1206.5538*

Bundesagentur für Arbeit (2011). *Klassifikation der Berufe 2010. Band 1: Systematischer und alphabetischer Teil mit Erläuterungen*, Bundesagentur für Arbeit, Nuremberg.

Bushnell, D. (1998). An Evaluation of Computer-assisted Occupation Coding, *in* A. Westlake, J. Martin, M. Rigg and C. Skinner (eds), *New Methods for Survey Research, Proceedings of the International Conference*, Association for Survey Computing, Chilworth Manor, Southampton, pp. 23–36.
URL: *http://www.asc.org.uk/publications/proceedings/ASC1998Proceedings.pdf*

Campanelli, P., Thomson, K., Moon, N. and Staples, T. (1997). The Quality of Occupational Coding in the United Kingdom, *in* L. Lyberg, P. Biemer, M. Collins, E. DeLeeuw, C. Dippo, N. Schwarz and D. Trewin (eds), *Survey Measurement and Process Quality*,

John Wiley & Sons, Inc., New York, pp. 437–453.
URL: http://dx.doi.org/10.1002/9781118490013.ch19

Chen, B.-C., Creecy, R. H. and Appel, M. V. (1993). Error Control of Automated Industry and Occupation Coding, *Journal of Official Statistics* **9**(4): 729–745.

Conrad, F. G. (1997). Using Expert Systems To Model And Improve Survey Classification Processes, *in* L. Lyberg, P. Biemer, M. Collins, E. DeLeeuw, C. Dippo, N. Schwarz and D. Trewin (eds), *Survey Measurement and Process Quality*, John Wiley & Sons, New York, pp. 393–414.

DeBell, M. (2013). Harder Than It Looks: Coding Political Knowledge on the ANES, *Political Analysis* **21**(4): 393–406.
URL: http://pan.oxfordjournals.org/content/21/4/393.abstract

Dowle, M., Short, T. and Lianoglou, S. (2012). *data.table: Extension of data.frame for fast indexing, fast ordered joins, fast assignment, fast grouping and list columns.* R package version 1.8.6.
URL: http://CRAN.R-project.org/package=data.table

Drasch, K., Matthes, B., Munz, M., Paulus, W. and Valentin, M.-A. (2012). Arbeiten und Lernen im Wandel * Teil V: Die Codierung der offenen Angaben zur beruflichen Tätigkeit, Ausbildung und Branche, *FDZ-Methodenreport 04/2012*, Forschungsdatenzentrum der Bundesagentur für Arbeit im Institut für Arbeitsmarkt- und Berufsforschung, Nuremberg.
URL: http://fdz.iab.de/187/section.aspx/Publikation/k120504304

Esuli, A., Fagni, T. and Sebastiani, F. (2008). Boosting multi-label hierarchical text categorization, *Information Retrieval* **11**(4): 287–313.
URL: http://dx.doi.org/10.1007/s10791-008-9047-y

Esuli, A. and Sebastiani, F. (2010). Machines that learn how to code open-ended survey data, *International Journal of Market Research* **52**.
URL: https://www.mrs.org.uk/ijmr_article/article/92864

Fawcett, T. (2003). ROC Graphs: Notes and Practical Considerations for Data Mining Researchers, *Technical report*, HP Laboratories, Palo Alto.

Feinerer, I. and Hornik, K. (2012). *tm: Text Mining Package.* R package version 0.5-8.1.
URL: http://CRAN.R-project.org/package=tm

Ferrillo, A., Macchia, S. and Vicari, P. (2008). Different quality tests on the automatic coding procedure for the Economic Activities descriptions, *Proceedings of the European*

Conference on Quality in Official Statistics - Q2008.
URL: http://q2008.istat.it/sessions/paper/15Ferrillo.pdf

Fielding, J., Fielding, N. and Hughes, G. (2013). Opening up open-ended survey data using qualitative software, *Quality & Quantity* **47**(6): 3261–3276.
URL: http://dx.doi.org/10.1007/s11135-012-9716-1

Geis, A. (2011). *Handbuch der Berufsvercodung*, GESIS. Survey Design and Methodology, Mannheim.
URL: http://www.gesis.org/unser-angebot/daten-erheben/berufscodierung/

Geis, A. J. and Hoffmeyer-Zlotnik, J. H. (2000). Stand der Berufsvercodung, *ZUMA-Nachrichten* **24**(47): 103–128.
URL: http://www.gesis.org/fileadmin/upload/forschung/publikationen/zeitschriften/zuma_nachrichten/zn_47.pdf

Gibson, J. L. and Caldeira, G. A. (2009). Knowing the Supreme Court? A Reconsideration of Public Ignorance of the High Court, *The Journal of Politics* **71**: 429–441.
URL: http://journals.cambridge.org/article_S0022381609090379

Gillman, D. W. and Appel, M. V. (1994). Automated Coding Research at the Census Bureau, *Statistical research report series*, U.S. Census Bureau, Washington, DC.
URL: https://www.census.gov/srd/papers/pdf/rr94-4.pdf

Giorgetti, D. and Sebastiani, F. (2003). Automating survey coding by multiclass text categorization techniques, *Journal of the American Society for Information Science and Technology* **54**(14): 1269–1277.
URL: http://dx.doi.org/10.1002/asi.10335

Groves, R. M., Fowler, F. J., Couper, M. P., Lepkowski, J. M., Singer, E. and Tourangeau, R. (2009). *Survey Methodology (Wiley Series in Survey Methodology)*, Wiley.

Hacking, W. and Willenborg, L. (2012). Theme: Coding; interpreting short descriptions using a classification, *Statistics methods*, Statistics Netherlands, The Hague/Heerlen.
URL: http://www.cbs.nl/en-GB/menu/methoden/gevalideerde-methoden/bewerken/default.htm

Hartmann, J. and Schütz, G. (2002). Die Klassifizierung der Berufe und der Wirtschaftszweige im Sozio-oekonomischen Panel * Neuvercodung der Daten 1984 - 2001, *Technical report*, Infratest Sozialforschung, Munich.
URL: http://www.diw.de/documents/dokumentenarchiv/17/diw_01.c.40132.de/vercodung.pdf

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Series in Statistics, 2 edn, Springer.
URL: *http://statweb.stanford.edu/~tibs/ElemStatLearn/index.html*

Hoffmeyer-Zlotnik, J. H., Hess, D. and Geis, A. J. (2004). Computerunterstützte Vercodung der International Standard Classification of Occupations (ISCO-88), *ZUMA-Nachrichten* **28**(55): 29–52.

Hoffmeyer-Zlotnik, J. H. and Warner, U. (2012). *Harmonisierung demographischer und sozio-ökonomischer Variablen: Instrumente für die international vergleichende Surveyforschung*, VS Verlag für Sozialwissenschaften, Wiesbaden.

Hofner, B., Mayr, A., Robinzonov, N. and Schmid, M. (2012). Model-based Boosting in R * A Hands-on Tutorial Using the R Package mboost, *R Package Vignette* .
URL: *http://cran.r-project.org/web/packages/mboost/vignettes/mboost_tutorial.pdf*

Hothorn, T., Hornik, K. and Zeileis, A. (2006). Unbiased Recursive Partitioning: A Conditional Inference Framework, *Journal of Computational and Graphical Statistics* **15**(3): 651–674.

Ishwaran, H. and Kogalur, U. B. (2013). *Random Forests for Survival, Regression and Classification (RF-SRC)*. R package version 1.4.
URL: *http://cran.r-project.org/web/packages/randomForestSRC/*

Jung, Y., Yoo, J., Myaeng, S.-H. and Han, D.-C. (2008). A Web-Based Automated System for Industry and Occupation Coding, *in* J. Bailey, D. Maier, K.-D. Schewe, B. Thalheim and X. Wang (eds), *Web Information Systems Engineering - WISE 2008*, Vol. 5175 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 443–457.
URL: *http://dx.doi.org/10.1007/978-3-540-85481-4_33*

Kaptein, R. (2005). *Meta-Classifier Approaches to Reliable Text Classification*, Master's thesis, Universiteit Maastricht, Maastricht.

Lewis, D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval, *in* C. Nédellec and C. Rouveirol (eds), *Machine Learning: ECML-98*, Vol. 1398 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 4–15.
URL: *http://dx.doi.org/10.1007/BFb0026666*

Liaw, A. and Wiener, M. (2002). Classification and Regression by randomForest, *R News* **2**(3): 18–22.
URL: *http://CRAN.R-project.org/doc/Rnews/*

Liu, X. and Croft, W. B. (2004). Cluster-based Retrieval Using Language Models, *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, ACM, New York, NY, USA, pp. 186–193.
URL: *http://doi.acm.org/10.1145/1008992.1009026*

Lyberg, L. and Kasprzyk, D. (1997). Some Aspects of Post-Survey Processing, *in* L. Lyberg, P. Biemer, M. Collins, E. DeLeeuw, C. Dippo, N. Schwarz and D. Trewin (eds), *Survey Measurement and Process Quality*, John Wiley & Sons, New York, pp. 393–414.

Maaz, K., Trautwein, U., Gresch, C., Lüdtke, O. and Watermann, R. (2009). Intercoder-Reliabilität bei der Berufscodierung nach der ISCO-88 und Validität des sozioökonomischen Status, *Zeitschrift für Erziehungswissenschaft* **12**(2): 281–301.
URL: *http://dx.doi.org/10.1007/s11618-009-0068-0*

Macchia, S., Murgia, M. and Vicari, P. (2010). Integration between automatic coding and statistical analysis of textual data systems, *in* S. Bolasco, I. Chiari and L. Giuliano (eds), *Proceedings of the 10th International Conference on Statistical Analysis of Textual Data * Jadt 2010*, Sapienza University of Rome and Istat - Instituto Nazionale di Statistica and Enel - Ente Nazionale di Energia Elettrica and Percodsi srl and SAS Institute.

Manning, C. D., Raghavan, P. and Schütze, H. (2008). *Introduction to Information Retrieval*, Cambridge University Press, Cambridge.
URL: *http://nlp.stanford.edu/IR-book/*

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive bayes text classification, *AAAI-98 workshop on learning for text categorization*, Vol. 752, pp. 41–48.

Paulus, W. and Matthes, B. (2013). Klassifikation der Berufe * Struktur, Codierung und Umsteigeschlüssel, *FDZ-Methodenreport 08/2013*, Forschungsdatenzentrum der Bundesagentur für Arbeit im Institut für Arbeitsmarkt- und Berufsforschung, Nuremberg.
URL: *http://fdz.iab.de/187/section.aspx/Publikation/k131014a03*

Peng, F., Schuurmans, D. and Wang, S. (2004). Augmenting Naive Bayes Classifiers with Statistical Language Models, *Information Retrieval* **7**(3-4): 317–345.

Prigge, M., Liebers, F. and Latza, U. (2013). Kodierung von Berufsangaben nach KldB2010 im Rahmen der Gutenberg-Gesundheitsstudie (GHS) - Methodisches Vorgehen, Qualität und Auswertemöglichkeiten. Präsentation auf der 8. Jahrestagung der DGEpi und 1. Internationales LIFE-Symposium.

R Core Team (2012a). *foreign: Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ...* R package version 0.8-51.
URL: http://CRAN.R-project.org/package=foreign

R Core Team (2012b). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
URL: http://www.R-project.org/

Reja, U., Manfreda, K. L., Hlebec, V. and Vehovar, V. (2003). Open-ended vs. close-ended questions in web questionnaires, *Developments in Applied Statistics (Metodološki zvezki)* **19**: 159–77.

Riviere, P. (1997). SICORE - General Automatic Coding System, *in* United Nations Statistical Commission and Economic Commission for Europe (ed.), *Statistical Data Editing Volume No. 2*, United Nations, New York.
URL: http://www.unece.org/stats/publications/editing/SDE2.html

Romero, F. P., Julián-Iranzo, P., Soto, A., Ferreira-Satler, M. and Gallardo-Casero, J. (2013). Classifying unlabeled short texts using a fuzzy declarative approach, *Language Resources and Evaluation* **47**(1): 151–178.
URL: http://dx.doi.org/10.1007/s10579-012-9203-2

Sangameshwar, P. and Palshikar, G. (2013). SurveyCoder: A System for Classification of Survey Responses, *in* E. Métais, F. Meziane, M. Saraee, V. Sugumaran and S. Vadera (eds), *Natural Language Processing and Information Systems, 18th International Conference on Applications of Natural Language to Information Systems*, Vol. 7934 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 417–420.
URL: http://dx.doi.org/10.1007/978-3-642-38824-8_52

Scharkow, M. (2012). *Automatische Inhaltsanalyse und maschinelles Lernen*, PhD thesis, Universität der Künste Berlin, Berlin.
URL: http://opus.kobv.de/udk/volltexte/2012/40/pdf/dissertation_scharkow_final_udk.pdf

Schröder, H., Kersting, A., Gilberg, R. and Steinwede, J. (2013). Methodenbericht zur Haupterhebung lidA - leben in der Arbeit, *FDZ-Methodenreport 01/2013*, Forschungsdatenzentrum der Bundesagentur für Arbeit im Institut für Arbeitsmarkt- und Berufsforschung, Nuremberg.
URL: http://fdz.iab.de/187/section.aspx/Publikation/k130307302

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization, *ACM Comput. Surv.* **34**(1): 1–47.
URL: http://doi.acm.org/10.1145/505282.505283

Silla, C. N. J. and Freitas, A. x. (2011). A survey of hierarchical classification across different application domains, *Data Mining and Knowledge Discovery* **22**(1-2): 31–72.
URL: http://dx.doi.org/10.1007/s10618-010-0175-9

Sing, T., Sander, O., Beerenwinkel, N. and Lengauer, T. (2012). *ROCR: Visualizing the performance of scoring classifiers.* R package version 1.0-4.
URL: http://CRAN.R-project.org/package=ROCR

Statistisches Bundesamt (2010). *Demographische Standards*, Statistisches Bundesamt, Wiesbaden.

Stemler, S. (2001). An overview of content analysis, *Practical Assessment, Research & Evaluation* **7**.
URL: http://pareonline.net/getvn.asp?v=7&n=17

Svensson, J. (2012). Quality control of coding of survey responses at Statistics Sweden, *Proceedings of the European Conference on Quality in Official Statistics - Q2012*.
URL: http://www.q2012.gr/default.asp?p=14

Thompson, M., Kornbau, M. E. and Vesely, J. (2012). Creating an Automated Industry and Occupation Coding Process for the American Community Survey, unpublished.

TNS Infratest Sozialforschung (2012). *Die Vercodung der offenen Angaben zur beruflichen Tätigkeit nach der Klassifikation der Berufe 2010 (KldB2010) und nach der International Standard Classification of Occupations (ISCO08) * Entscheidungsregeln bei nicht eindeutigen Angaben*, TNS Infratest Sozialforschung, Munich.
URL: http://www.bibb.de/dokumente/pdf/a22_etb_Berufsvercodung_KldB2010_ISCO08.pdf

Tourigny, J. Y. and Moloney, J. (1997). The 1991 Canadian Census of Population Experience with Automated Coding, *in* United Nations Statistical Commission and Economic Commission for Europe (ed.), *Statistical Data Editing Volume No. 2*, United Nations, New York.
URL: http://www.unece.org/stats/publications/editing/SDE2.html

Tutz, G. (2000). *Die Analyse kategorialer Daten*, Oldenbourg.

United Nations Statistical Commission and Economic Commission for Europe (ed.) (1997). *Statistical Data Editing Volume No. 2*, United Nations, New York, chapter 6.
URL: http://www.unece.org/stats/publications/editing/SDE2.html

U.S. Census Bureau (2009). *Design and Methodology: American Community Survey*, U.S. Census Bureau, Washington, DC.
URL: http://www.census.gov/acs/www/methodology/methodology_main/

Viechnicki, P. (1998). A performance evaluation of automatic survey classifiers, *in* V. Honavar and G. Slutzki (eds), *Grammatical Inference*, Vol. 1433 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 244–256.
URL: *http://dx.doi.org/10.1007/BFb0054080*

Wagner, H. (2010/2011). Einführung in die Bayes-Statistik WiSe 2010/11, Lecture at LMU Munich.
URL: *http://thomas.userweb.mwn.de/Lehre/wise1011/Bayes_1011/*

Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*, Springer New York.
URL: *http://had.co.nz/ggplot2/book*

Wickham, H. (2012). *stringr: Make it easier to work with strings.* R package version 0.6.2.
URL: *http://CRAN.R-project.org/package=stringr*

Willenborg, L. (2012). Semantic Networks for Automatic Coding, *Discussion paper*, Statistics Netherlands, The Hague/Heerlen.
URL: *http://www.cbs.nl/en-GB/menu/methoden/onderzoek-methoden/discussionpapers/archief/2012/semantic-networks-for-automatic-coding.htm*

Appendix A

Diagrams

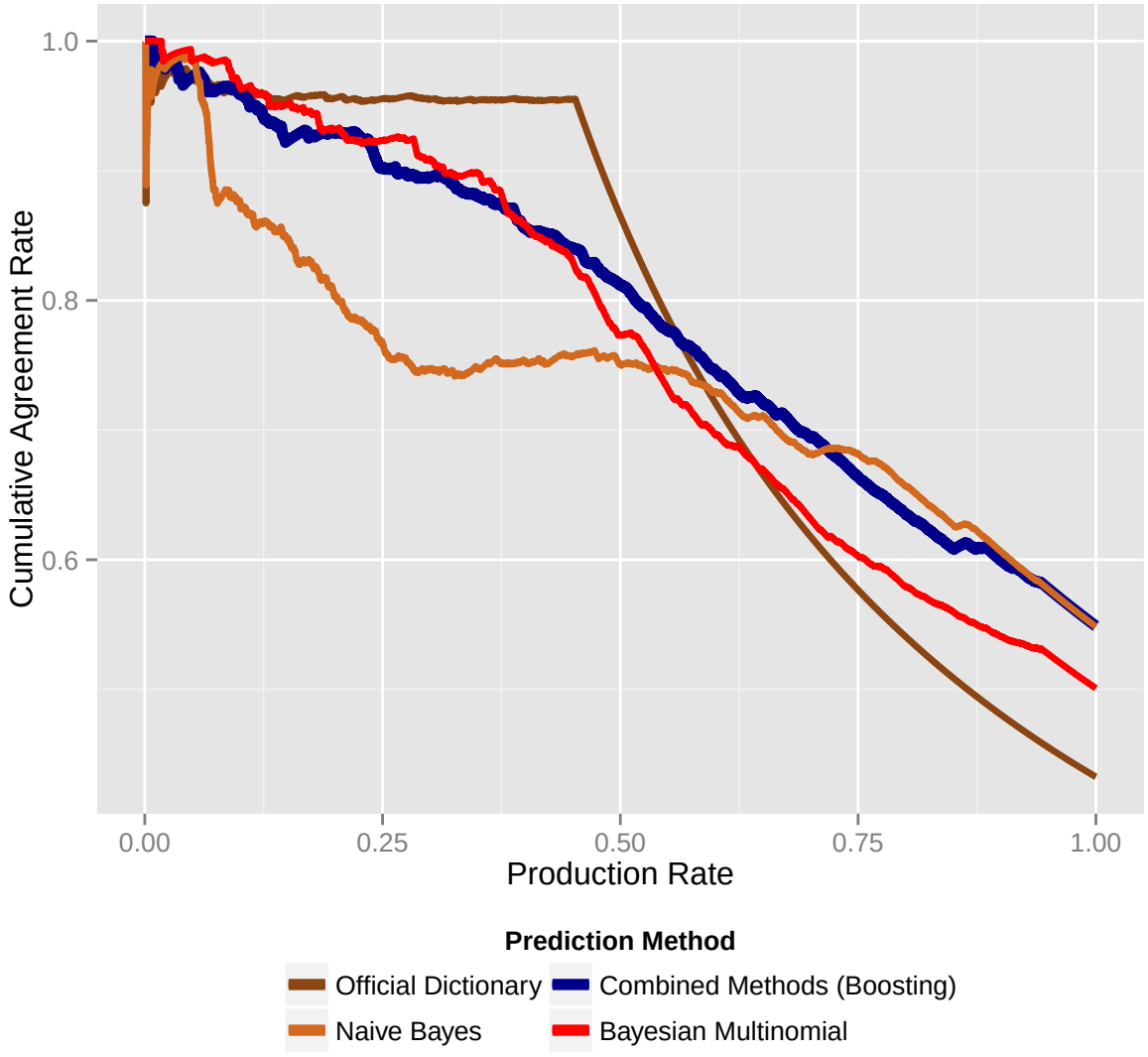

Figure A.1: Agreement and Production Rates for lidA-test data

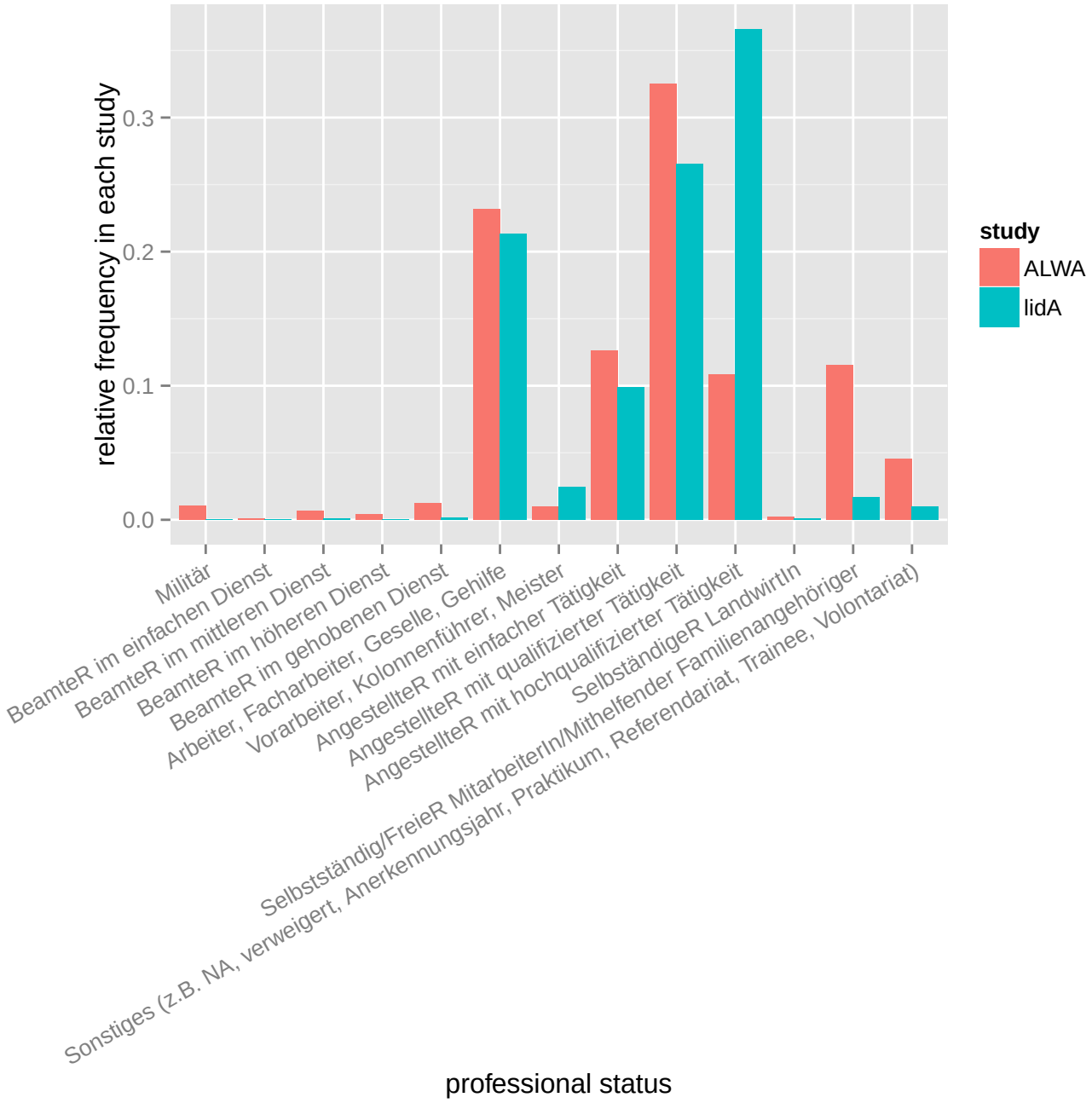

Figure A.2: Professional Status in both datasets

Appendix B

Exemplary Job Category Suggestions

```
Eingegebener Beruf: BAUMASCHINIST | Phrase: BAUMASCHINIST | Coded: 52522
-------------------------------------------------------------------------------
    Bau- & Transportgeräteführung : ( 5 Antworten in ALWA; Corr. Prob. =  0.9529602 )
-------------------------------------------------------------------------------
~~~~~~~~~
Führer/innen Erdbewegungs- & verw. Maschinen
52522 ..... z.B.: Baugeräteführer/in || Baumaschinist/in
-------------------------------------------------------------------------------
    Berg-, Tagebau & Sprengtechnik : ( 0 Antworten in ALWA; Corr. Prob. =  0.008892748 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Berg- & Tagebau
21112 ..... z.B.: Bergbaumaschinist/in || Bergmechaniker/in
-------------------------------------------------------------------------------
    Tiefbau : ( 0 Antworten in ALWA; Corr. Prob. =  0.007855415 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Gleisbau
32232 ..... z.B.: Gleisbaumaschinist/in || Gleisbauer/in
```

Figure B.1: Algorithm output for answer 'Baumaschinist'

```
Eingegebener Beruf: KUECHENHELFERIN | Phrase: KUECHENHELFERIN | Coded: 29301
--------------------------------------------------------------------------------
    Speisenzubereitung : ( 1 Antworten in ALWA; Corr. Prob. =  0.9399518 )
--------------------------------------------------------------------------------
~~~~~~~~~
Köche/Köchinnen (o.Spez.)
29301 ..... z.B.: Küchenhelfer/in || Küchenhelfer/in
```

Figure B.2: Algorithm output for answer 'Küchenhelferin'

```
Eingegebener Beruf: ALTENPFLEGEHELFERIN IM SENIORENHEIM |
Phrase: ALTENPFLEGEHELFERIN | Coded: 82101
--------------------------------------------------------------------------------
    Altenpflege : ( 0 Antworten in ALWA; Corr. Prob. =  0.9301586 )
--------------------------------------------------------------------------------
~~~~~~~~~
Berufe Altenpflege (o.Spez.)
82101 ..... z.B.: Altenpflegehelfer/in || Hilfskraft - Altenpflege
--------------------------------------------------------------------------------
    Gesundheits- & Krankenpflege, Rettungsdienst & Geburtshilfe :
    ( 0 Antworten in ALWA; Corr. Prob. =  0.01307547 )
--------------------------------------------------------------------------------
~~~~~~~~~
Berufe Gesundh.- & Krankenpflege (o.Spez.)
81301 ..... z.B.: Kranken- und Altenpflegehelfer/in ||
                  Gesundheits- und Krankenpflegehelfer/in
```

Figure B.3: Algorithm output for answer 'Altenpflegehelferin im Seniorenheim'

```
Eingegebener Beruf: BUCHHALTERIN | Phrase: BUCHHALTERIN | Coded: 72213
-------------------------------------------------------------------------------
    Rechnungswesen, Controlling & Revision : ( 36 Antworten in ALWA;
    Corr. Prob. =  0.202529 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Buchhaltung
72212 ..... z.B.: Kfm. Ass./Wirtschaftsassistent/in - DV/Rechnungswesen
72213 ..... z.B.: Finanzbuchhalter/in || Kontokorrentbuchhalter/in ||
                  Lohn- und Gehaltsbuchhalter/in
~~~~~~~~~
Berufe Kostenrechnung & Kalkulation
72223 ..... z.B.: Kostenrechner/in
-------------------------------------------------------------------------------
    Informatik : ( 52 Antworten in ALWA; Corr. Prob. =  0.1426831 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Wirtschaftsinformatik
43112 ..... z.B.: Wirtschaftsassistent/in - DV/Rechnungswesen
-------------------------------------------------------------------------------
    Versicherungs- & Finanzdienstleistungen : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.01019844 )
-------------------------------------------------------------------------------
~~~~~~~~~
Anlageberater/innen & sonst.Finanzdienstl.
72123 ..... z.B.: Wertpapiersachbearbeiter/in
-------------------------------------------------------------------------------
    Hotellerie : ( 0 Antworten in ALWA; Corr. Prob. =  0.007068873 )
-------------------------------------------------------------------------------
~~~~~~~~~
Hotelkaufleute
63212 ..... z.B.: Hotelkaufmann/-frau
-------------------------------------------------------------------------------
    Tourismus & Sport : ( 0 Antworten in ALWA; Corr. Prob. =  0.006908971 )
-------------------------------------------------------------------------------
~~~~~~~~~
Tourismuskaufleute
63113 ..... z.B.: Betriebswirt/in (Fachschule) - Touristik/Reiseverkehr
```

Figure B.4: Algorithm output for answer 'Buchhalterin'

```
Eingegebener Beruf: ENTWICKLUNGSINGENIEUR FUER MASCHINENBAU |
Phrase: ENTWICKLUNGSINGENIEUR | Coded: 27104
-------------------------------------------------------------------------------
    Technische Forschung & Entwicklung : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.4761688 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe techn. Forsch. & Entwickl. (o.Spez.)
27104 ..... z.B.: Forschungs- und Entwicklungsingenieur/in || Entwicklungsingenieur/in
-------------------------------------------------------------------------------
    Maschinenbau- & Betriebstechnik : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.3666988 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Maschinenbau-&Betriebstech.(o.Spez.)
25104 ..... z.B.: Ingenieur/in - Maschinenbau (allgemeiner Maschinenbau)
-------------------------------------------------------------------------------
    Elektrotechnik : ( 0 Antworten in ALWA; Corr. Prob. =  0.01888952 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Elektrotechnik (o.Spez.)
26304 ..... z.B.: Ingenieur/in - Elektrotechnik (allgemeine Elektrotechnik)
-------------------------------------------------------------------------------
    Softwareentwicklung & Programmierung : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.01817224 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Softwareentwicklung
43414 ..... z.B.: Softwareentwickler/in || Softwareentwickler/in
-------------------------------------------------------------------------------
    Informatik : ( 0 Antworten in ALWA; Corr. Prob. =  0.01432823 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Informatik (o.Spez.)
43104 ..... z.B.: Dipl.-Informatiker/in (FH)
-------------------------------------------------------------------------------
    IT-Netzwerktechnik, IT-Koord., IT-Admin. & IT-Organisation :
    ( 0 Antworten in ALWA; Corr. Prob. =  0.009487363 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe IT-Koordination
43323 ..... z.B.: IT-Entwickler/in
```

Figure B.5: Algorithm output for answer 'Entwicklungsingeneur für Maschinenbau'

```
Eingegebener Beruf: SYSTEMANALYTIKERIN | Phrase: SYSTEMANALYTIKERIN | Coded: 43214
-------------------------------------------------------------------------------
    Softwareentwicklung & Programmierung : ( 1 Antworten in ALWA;
    Corr. Prob. =  0.4589744 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Softwareentwicklung
43414 ..... z.B.: Softwareentwickler/in
-------------------------------------------------------------------------------
    Vermessung & Kartografie : ( 1 Antworten in ALWA; Corr. Prob. =  0.4086493 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Vermessungstechnik
31214 ..... z.B.: Ingenieur/in - Vermessungswesen
-------------------------------------------------------------------------------
    IT-Systemanalyse, IT-Anwendungsberatung & IT-Vertrieb : ( 0 Antworten in ALWA;
    Corr. Prob. =  0.09561851 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe IT-Systemanalyse
43214 ..... z.B.: Systemanalytiker/in || IT-Systemanalytiker/in
-------------------------------------------------------------------------------
    Informatik : ( 0 Antworten in ALWA; Corr. Prob. =  0.0385127 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe Wirtschaftsinformatik
43112 ..... z.B.: Assistent/in - Informatik (Wirtschaftsinformatik)
43114 ..... z.B.: Verwaltungsinformatiker/in (Hochschule)
-------------------------------------------------------------------------------
    Redaktion & Journalismus : ( 0 Antworten in ALWA; Corr. Prob. =  0.01367254 )
-------------------------------------------------------------------------------
~~~~~~~~~
Redakteure/-innen & Journalisten/-innen
92413 ..... z.B.: Lernsystemanalytiker/in
-------------------------------------------------------------------------------
    IT-Netzwerktechnik, IT-Koord., IT-Admin. & IT-Organisation :
    ( 0 Antworten in ALWA; Corr. Prob. =  0.00698711 )
-------------------------------------------------------------------------------
~~~~~~~~~
Berufe IT-Netzw.-Adm&-Orga.(sonst.spez.Tät.)
43384 ..... z.B.: Fraud-Analyst/in
```

Figure B.6: Algorithm output for answer 'Systemanalytikerin'

Erklärung zur Urheberschaft

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe.

München, den 25. April 2014

(Malte Schierholz)