

UC Berkeley

UC Berkeley Electronic Theses and Dissertations

Title

Two Statistical Methods for Clustering Medicare Claims Data into Episodes of Care

Permalink

<https://escholarship.org/uc/item/22p3r8rx>

Author

Gibbons, Robert Lyon

Publication Date

2011

Peer reviewed|Thesis/dissertation

**Two Statistical Methods for Clustering Medicare Claims into Episodes of
Care**

by

Robert Lyon Gibbons

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Statistics

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Deborah Nolan, Chair

Professor Dan Klein

Professor Martin Wainwright

Spring 2011

Two Statistical Methods for Clustering Medicare Claims into Episodes of Care

Copyright 2011
by
Robert Lyon Gibbons

Abstract

Two Statistical Methods for Clustering Medicare Claims into Episodes of Care

by

Robert Lyon Gibbons

Doctor of Philosophy in Statistics

University of California, Berkeley

Professor Deborah Nolan, Chair

Methods for clustering health care claims into episodes of care are important tools for data analysis in evaluating health care outcomes and methods of payment. In this research, we implement two statistical methods (1) using an ensemble classifier, Random Forests, to estimate the strength of relationship in pairs of Medicare claims, and (2) using a sequence model and an Expectation Maximization (EM) algorithm.

Previous researchers into episode of care clustering have implemented other methods, some based on decision rules and others based on statistical methods. Other research in natural language processing, particularly conversation disentanglement, has developed methods that were inspirational for the methods that we implemented in this research.

We acquired two sets of Medicare claims data, one containing claims from 2006 and 2007 for 1.9 million patients (a 5 percent sample), and the other containing claims from 2007 and 2008 for 250 thousand patients. We tested our two statistical methods on claims for 50 randomly selected patients who were age 65 and older, using independent annotations by three licensed nurse practitioners. We found that the method using Random Forests outperformed the sequence model and achieved accuracy comparable to the nurse practitioner annotators. Future research into episode of care clustering might incorporate more extensive data on clinical relationships and create a more flexible representation of episode clusters, such as hierarchies and phases of care.

To Professors David Friedman and Leo Breiman,
U.C. Berkeley researchers and teachers whom I have greatly admired,
whose ideas have been both inspirational and enduring.

Contents

List of Figures	iv
List of Tables	v
1 Introduction	1
2 Previous work	2
2.1 Similar methods used in Natural Language Processing	3
3 Medicare claims data	5
3.1 Medicare data used in this research	5
4 First method for clustering Medicare claims: Pairs of line items, Random Forests	9
4.1 Method of clustering line items	10
4.2 Random Forests	12
4.2.1 Implementation of Random Forests in this research	12
4.2.2 Example from Random Forests fitted model	13
4.2.3 Number of variables at each Random Forests branching step	14
4.3 Clinical Classifications Software	14
4.4 Metathesaurus	16
4.5 Kullback-Leibler divergence terms	17
4.6 Modification of KL divergence terms	20
5 Second method for clustering Medicare claims: generative model, EM algorithm	26
5.1 Step 1: Randomly generate the starting days and diagnosis categories of new episodes	26
5.2 Step 2: For each new episode, randomly generate the episode as a Markov chain	27
5.3 Example of calculating probability of generating episodes	29
5.4 EM algorithm used to estimate parameters	31
5.4.1 Each EM iteration, episodes assigned using posterior probabilities	31
5.5 Final assignment of diagnostic events to episodes	34

6	Comparison of episodes against test set	36
6.1	Protocol for Nurse Practitioner review	36
6.2	Comparison of review results	37
6.3	Recall of complete episodes	37
6.4	One-to-one accuracy	38
6.5	Average pairwise agreement	39
6.6	Differences in results across EM iterations	41
6.7	Differences in episodes among reviewers	41
7	Conclusion	52
	Bibliography	54
A	SAS and R programs for first clustering method	57
A.1	SAS code for extracting line item pairs for same patient and provider	57
A.2	SAS code for improving estimated transition probabilities by multiplying transition matrix	60
A.3	SAS code for calculating SNOMED distances	63
A.4	SAS code for creating the independent and dependent variables on line item pairs	75
A.5	R code for fitting Random Forests model to line item pair data	84
A.6	SAS code for clustering line items into episodes of care	84
B	SAS programs for second clustering method	94
B.1	SAS code for producing initial parameter estimates and updating with EM algorithm	94
B.2	SAS code for clustering line items into episodes of care using estimated pa- rameters from EM algorithm	114

List of Figures

3.1	Among all 12,743 diagnoses (ordered from most to least frequent), empirical cumulative probability distribution, for patients age 65 and older	7
4.1	For a selected patient, example of method for clustering line items into episodes of care using weights from Random Forests	11
4.2	Example from fitted Random Forests model: Probability of same diagnosis category given number of days between line items	13
4.3	Example from fitted model that allowed Random Forests to select the best among all 8 variables at each branching step	15
4.4	Example from fitted model that limited Random Forests to the best of 3 randomly selected variables (out of 8 variables) at each branching step . . .	16
4.5	CCS “single-level” diagnosis groups that branch from “Diseases of the respiratory system”	22
4.6	Illustration of how we calculate SNOMED distance between diagnoses chronic fatigue syndrome and post-varicella encephalitis	23
4.7	Among distinct pairs of ICD-9 diagnoses, proportion in the same CCS diagnosis category given SNOMED distance between the diagnoses	24
4.8	Improvement in estimated transition probabilities among selected hypertension, vaccine codes, using matrix multiplication	25
5.1	For a selected patient, illustration of how the second step of our statistical model could generate three diagnostic events within an episode of care . . .	28
6.1	Example of how we calculated pairwise agreement for a selected line item .	46
6.2	Test set results across EM iterations	47
6.3	Test set results, EM clustering look-ahead	48
6.4	Test set results, look-ahead, number of episodes	49
6.5	NP one-to-one accuracy compared to entropy	50
6.6	RF one-to-one accuracy compared to entropy	51

List of Tables

3.1	Among Medicare patients, mean and standard deviation of number of line items, diagnoses, and providers during 2006-2007	6
3.2	For Medicare patients age 6t5 and older, the 10 most frequent ICD-9 diagnosis codes	8
4.1	Distribution of SNOMED distances between ICD-9 diagnosis code pairs . .	17
4.2	For the 6 diagnoses with the highest empirical probability conditional on ICD-9 code 493.22 (chronic obstructive asthma), comparison of marginal probability, conditional probability, and KL divergence term	19
4.3	For the 6 diagnoses with the highest empirical probability conditional on ICD-9 code 493.22 (chronic obstructive asthma), comparison of KL divergence terms using line items linked because of same provider, referring provider, and same calendar quarter	19
4.4	Initial estimated transition probabilities among selected hypertension and vaccine codes, based on same provider link	20
4.5	Among providers in 5 percent sample with at least 30 patients with diagnoses 4011 and/or 4019, percent billing only one of these diagnoses	21
5.1	Calculating the probability of generating two episodes	30
5.2	Example of calculating posterior probabilities	33
5.3	Parameter estimates during EM iterations	34
6.1	Recall of complete episodes	38
6.2	One-to-one accuracy	39
6.3	Example of date of service numbers	39
6.4	Average pairwise agreement within window k	41
6.5	Differences among reviewers in creating episodes	42

Acknowledgments

In the course of this research, I received much valuable advice and assistance. First, I would like to thank the members of my dissertation and exam committees, Professors Michael Jordan, Dan Klein, Deborah Nolan, and Martin Wainwright. Professor Nolan offered ideas and questions that led me to follow many creative avenues of thought that greatly improved the statistical methods that we developed. The classes that I took from Professors Jordan, Klein, and Wainwright gave me a broad understanding of statistical learning methods that allowed me to experiment with methods of my own. I also would like to thank Professor Russ Cucina from the University of California, San Francisco, for his advice on sources of data to reveal clinical relationships between diagnosis codes.

I also would like to thank the three nurse practitioners who worked with me on this research, Karen Hipkins, Eve Korshak, and Alice Sun. They took on the complex task of annotating episodes of care for a sample of patient records, and they also offered many valuable insights about the complex clinical issues that arose from those annotations.

I am truly grateful to my wife Sharon and my two sons, Ben and Peter, for their patience through the long and intense journey that this research project became.

Chapter 1

Introduction

There is much public policy interest in effective methods for grouping health insurance claims into episodes of care. These methods have important roles in paying for health care and evaluating treatment outcomes. For example, many payers of health insurance claims base payments on a fee-for-service approach, which can lead to overuse of services. An alternative, based on paying for performance, might involve identifying providers who play important roles in patients care and measuring the outcomes of that care based on episodes (Davis, 2007; Macurdy et al., 2008; Damberg et al., 2009).

In this paper, we present two methods for grouping Medicare health insurance claims into episodes of care. The first method uses a classification algorithm, Random Forests, to estimate the strength of relationship between pairs of diagnoses. This method then uses these results to cluster each patient's diagnoses and dates of service into episodes of care. The second method uses a statistical model to approximate the process by which physicians and other medical providers generate diagnosis codes in health insurance claims. The method uses the Expectation Maximization (EM) algorithm to estimate parameters of the model. In this work, we go on to describe tests that we performed using Medicare claims data to evaluate how effectively these methods work to group the claims into episodes.

For a random sample of 50 patients, we used several measures to compare episode of care clusters based on annotations by three nurse practitioners, our two statistical methods, and two more basic methods. We found that our first method (Random Forests) produced results that were comparable to the nurse practitioner annotations, while our second method (EM algorithm) did not perform quite as well.

Chapter 2

Previous work

A patient's set of medical conditions over a time period are typically referred to as "periods of illness", which are further divided into "episodes of care". An episode of care is usually defined as a sequence of diagnostic and therapeutic encounters from one or more providers for a specific medical condition (Solon, 1967; Damberg et al., 2009). Most computerized methods for grouping health insurance claims into episodes of care have used decision rules rather than statistical models. These decision-rule methods have primarily consisted of the following components:

- A classification table that groups diagnoses into disease categories
- For each disease category, a time threshold such that, within the same category, a new episode begins if the elapsed time without a health care encounter exceeds the time threshold
- For diagnoses that are not part of disease categories, a method for assigning these to episodes

For example, Cave (1995) classified diagnoses into 125 disease categories. Each disease category was associated with a time threshold (such as 60 days for chest pain and 365 days for hypertension). Health insurance claims with diagnoses that fell into the same disease categories were grouped into the same episodes of care, except that a new episode was created if the gap between health care events exceeded the time threshold for that disease category. Finally, each unassigned event that occurred during an existing episode was assigned to that episode. If the unassigned event occurred during more than one episode, the event was assigned to the episode with the highest risk, based on an "expected resource intensity" scale.

Another study (Brailer and Kroch, 1999) used a very similar approach, using 214 disease categories to group International Classification of Diseases (ICD-9) diagnoses. These 214 disease categories were developed through a sequence of academic physician panels. In this study, researchers examined the effects of adjusting the time thresholds for breaking disease category groupings into episodes based on gaps in time. These researchers found that when the time thresholds were extended, episodic care, in which treatment was provided intermittently, was grouped into longer episodes.

A quite recent study (Biermans et al., 2008) developed a method to group diagnoses into episodes of care based on decision rules applied to the time sequence of health care events. The first diagnosis in the time sequence was assigned to the first episode. Subsequent health care events were either linked to the first episode or began new episodes based on a decision table developed by physicians. If these decision tables did not assign all diagnoses to episodes, assignments could be made based on previously hand-labeled cases. An unassigned health care event would be grouped with an event that had been assigned to an episode based on the proportion (among the subset of hand-labeled cases with attributes similar to the unassigned event) that had been assigned to that type of episode. The following is an example from this study. Consider a patient with unassigned diagnosis “fainting” and previously assigned diagnosis “hypoglycemia”. The “fainting” event would be assigned to the same episode as the “hypoglycemia” event because in the hand-assigned data there were 15 patients with this pair of diagnoses, of which 8 patients (53 percent) had this pair hand-labeled into the same episode, which exceeded the cutoff (for this kind of pairwise decision rule) of 48 percent used by the researchers. These cutoff points were determined by optimizing the current grouping in the hand-assigned data (which was used as training data).

In contrast to the rule-based methods described above, a recent study (Son et al. 2008) uses statistical models to group health care events into episodes. This study uses a supervised method, requiring training data in which the episodes of care were correctly labeled. Instead of just one statistical model, this method used three models: (1) a “local model” that relates each health care encounter to the type of episode, (2) a “cohesion model” that captures the probability that a pair of health care encounters will appear in the same episode, and (3) an “evolutionary model”, which incorporates the probability within an episode of the next health care encounter given the encounters in the episode so far. Each of these models produces a cost function. The three cost functions are combined, and finally, simulated annealing is used, beginning from some starting configuration (and its associated cost function), new configurations are generated. If the new configuration lowers the cost, then the new configuration replaces the old. Otherwise, the new configuration replaces the old with some small probability, which decreases to zero according to a “cooling schedule”.

2.1 Similar methods used in Natural Language Processing

The process of clustering health care events into episodes of care is quite similar to the natural language processing (NLP) task of conversation disentanglement (also referred to as “conversation threading”), for which there has been some quite recent research. Natural language processing involves teaching computers to process language, such as translating text between languages, answering questions posed by internet users, recognizing and translating speech, analyzing sentence structure, and many other language tasks. In conversation disentanglement, researchers attempt to cluster text (such as chat room transcripts) or speech into separate conversations. The units to be clustered consist of sentences or groups of sentences (“utterances”) from the same speaker. There has been even more research into the broader task of topic segmentation, but the statistical models often used in topic segmentation do not have to account for temporal overlap among the

segments.

One research team (Elsner and Charniak, 2008) initially tried to develop a Bayesian generative statistical model, but abandoned this (because the choice of prior had too much influence) in favor of a “graph partitioning” approach. This approach first classified message pairs into “alike/different”, then found a corresponding graph partition into clusters. Another research group (Wang and Oard, 2009) developed a similarity measure based on several features (temporal distance, word similarity, and name references to other participants in the message stream) and then used an agglomerative clustering approach to decide iteratively whether to add messages to existing clusters (conversations) or start new clusters.

Chapter 3

Medicare claims data

In this research, our statistical model used Medicare health insurance claims. Medicare is a health insurance program in the United States for the elderly and severely disabled. As of calendar year 2008, there were 43 million beneficiaries enrolled in Medicare, of whom 38 million were age 65 and older. The Medicare insurance benefit is divided into four parts. Part A (hospital insurance) covers inpatient hospitalizations, hospice care, and short-term care in a nursing home following an inpatient hospitalization. Part B covers all other outpatient care, such as physician services, laboratory, and durable medical equipment. Medicare beneficiaries can enroll in one of the Medicare managed care plans, which is covered under Part C. Most prescription drugs are covered under Part D, a relatively new benefit that began in January 2006.

3.1 Medicare data used in this research

The Medicare Carrier files consist of claims for payment (under Medicare Part B) that are submitted to Medicare contractors by physicians and most other non-institutional providers. Each claim lists one or more line items, which in turn are comprised of one or more services (usually one service). The following is a description of the data that we relied on in our research.

For this research, we primarily used Medicare Part B claims submitted by physicians, laboratories, and other non-institutional providers. These included all Part B non-institutional claims except those submitted by suppliers of durable medical equipment, prosthetics, orthosis, and supplies (DMEPOS), which are processed separately. These claims data are usually referred to as the ‘Carrier files’, because (until recently) the Medicare contractors that processed these claims were known as ‘Carriers’.

These Carrier claims data contain many data elements that allow Medicare to identify the patient and provider, to determine the correct payment and co-payment amounts, and to capture information about the service that was provided. Each Medicare claim is a claim for payment from one provider for one or more services rendered to a single patient. These claims must be submitted by the end of the calendar year following the year in which the service was rendered. Each Medicare claim is comprised of one or more ‘line items’, which are distinct services provided by a provider to a patient, usually on a single day. For

Quantity (per patient)	Mean	Standard deviation
Line items	83.2	98.8
Line-item diagnoses	19.7	16.0
Providers	14.8	13.8

Table 3.1: Among Medicare patients, mean and standard deviation of number of line items, diagnoses, and providers during 2006-2007

example, if a physician sees a Medicare patient for an emergency department ‘evaluation and management’ service (such as HCPCS code 99282) and also performs a chest x-ray for the same patient on the same day (such as HCPCS code 71010), the Medicare claim for these two services will list these as two separate line items, each with a line-item diagnosis code that corresponds to the need for each of these services.

These claims data also contain other data elements that we found useful in our research. For example, these claims contained information that identified the provider, so that it was possible to determine, for each patient, all of the claims that had been provided by each provider. The claims also identified the referring provider (if any), the place of service, and the date on which the service was provided.

We obtained several sets of Medicare claims data for this research. First, we obtained limited datasets from the Carrier files (which include physician, lab, and other non-institutional claims billed to Medicare part B) for services rendered during calendar years 2006 and 2007. These datasets contain all Medicare Part B Carrier claims for 1,899,249 Medicare patients, a random sample of five percent of all Medicare patients (we will refer to this dataset hereinafter as the “5 percent sample”. Of these patients, 83 percent were age 65 or older at the time of their first Medicare claim in 2006 or 2007. These datasets contained 82,858,324 claims, which were further subdivided into 158,103,530 line items, which are the individual service component of claims. These datasets only included the calendar quarter of service, not the exact date of service. Table 3.1 displays several statistics regarding the distribution of line items, line-item diagnoses, and number of providers among these Medicare patients during 2006 and 2007.

For patients who were age 65 or older at the time of their first service in our 5 percent sample, there were 12,743 distinct values in the line-item diagnosis field. Figure 3.1 shows the empirical cumulative probability distribution in probability-rank order for these diagnoses. As shown in this figure, most of the probability mass is accounted for by a small portion of the diagnoses. For example, the top 100 diagnoses account for 55 percent of the probability mass.

Table 3.2 lists the top 10 line-item diagnosis codes among patients 65 or older. As can be seen from this table, some of these ICD-9 codes actually describe the procedure that was performed (such as vaccination) rather than providing a diagnosis.

To obtain the exact dates of service, we requested a second collection of identifiable Medicare data, which consisted of a simple random sample of 250,000 Medicare patients who were enrolled in Medicare during 2007 and/or 2008 (limited to the traditional fee-for-service program, not including Medicare managed care). Hereinafter, we will refer to this

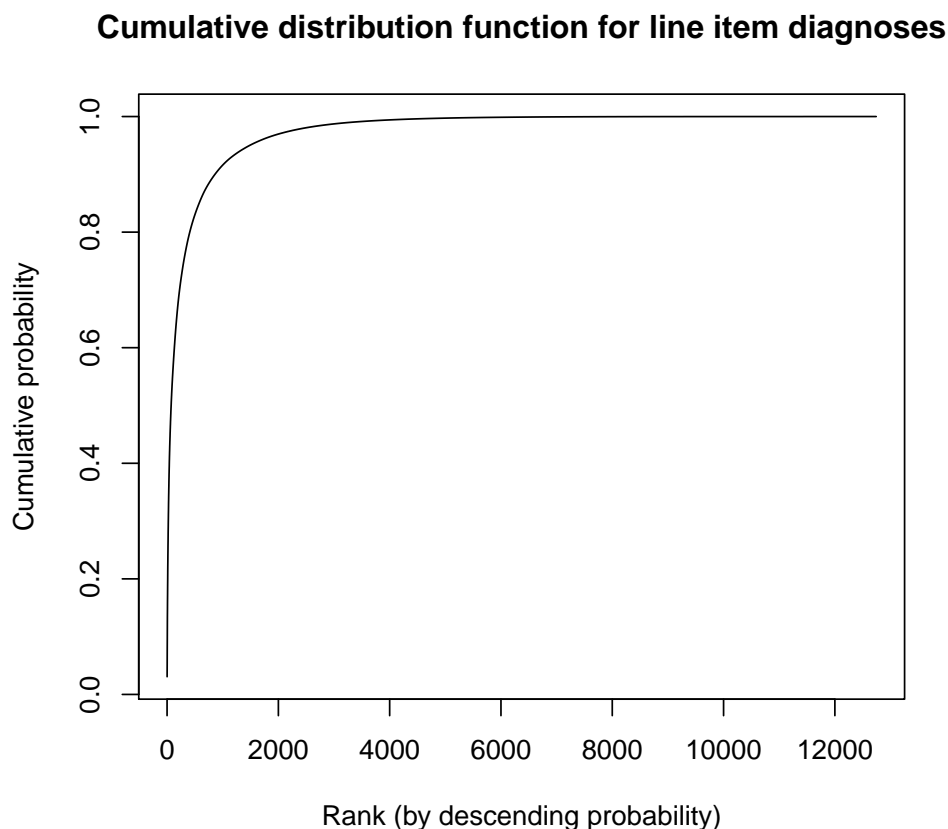


Figure 3.1: Among all 12,743 diagnoses (ordered from most to least frequent), empirical cumulative probability distribution, for patients age 65 and older

sample of claims for 250,000 patients as the “sample of 250,000 patients” to distinguish it from the larger “5 percent sample” that we described above. For this sample, we obtained both the Carrier file (the same file as the 5 percent sample) as well as the inpatient claims data for these patients. In addition, we obtained Medicare enrollment data, which contained demographic information about these Medicare patients. These patients were limited to Medicare enrollees where were age 65 or older as of January 1, 2007. Of these 250,000 patients, we determined from the enrollment data that 223,137 had been continuously enrolled in Medicare during all of 2007 and 2008. We then randomly partitioned these 223,137 patients into a “test set” consisting of 22,313 patients and a “training set” of 200,824 patients. For these 223,137 patients, there were 21,044,533 Carrier line items for service dates during 2007 and 2008.

Diagnosis	Empirical probability
250.00 Diabetes without mention of complication	0.031
401.1 Essential hypertension, benign	0.024
401.9 Essential hypertension, unspecified	0.023
427.31 Atrial fibrillation	0.021
272.4 Other and unspecified hyperlipidemia	0.018
V04.81 Need for vaccination: influenza	0.018
414.01 Coronary atherosclerosis of native coronary artery	0.013
786.50 Chest pain, unspecified	0.013
V58.61 Long-term (current) use of anticoagulants	0.012
285.9 Anemia, unspecified	0.012

Table 3.2: For Medicare patients age 65 and older, the 10 most frequent ICD-9 diagnosis codes

Chapter 4

First method for clustering Medicare claims: Pairs of line items, Random Forests

In this method, we clustered the line items for each patient into episodes of care based on weights associated with pairs of line items. These edge weights estimated the strength of relationship between the characteristics of the line item pairs. We estimated these weights using an ensemble classifier, Random Forests, with input features (derived from the Carrier data and other sources) that were defined on the line item pairs. This method is explained in detail below.

One ideal method for training Random Forests on line item pairs would have been to use training data in which, for each pair, we were given a variable that would indicate whether the line items were in the same episode of care. However, we did not have such an indicator variable. As a proxy, we used an indicator variable that showed whether the line items in the pair were in the same diagnostic category, as determined using software developed by the U.S. Agency for Health Care Research and Quality (AHRQ). In the Medicare Carrier data that we received (both our 5 percent sample and our sample of 250,000 patients), many of the diagnosis codes could be combined into categories using Clinical Classifications Software (CCS), which is maintained by AHRQ. We used these diagnosis categories to train Random Forests on pairs of line items, using “same diagnosis category” as the dependent variable. The CCS software and Random Forests are described in more detail in Section 4.3 and Section 4.2 respectively. The independent variables that we used in Random Forests are listed below.

- Number of days between line items (absolute value)
- Were both line items billed by the same provider?
- Did one of the line items list the provider of the other line item as the referring provider?
- Was either procedure code on a list (which we developed) of procedures that do not

contain information about the most likely diagnoses? For example, most office visits are on this list, but not a prostate-specific antigen test.

- SNOMED¹ distance between diagnoses, with values in $\{0, 1, 2, 3, 4\}$
- SNOMED distance between procedure codes, with values in $\{0, 1, 2, 3, 4\}$
- minimum of: SNOMED distance between diagnosis for first line item and procedure of the second, and SNOMED distance between diagnosis for second event and procedure for the first, with values in $\{0, 1, 2, 3, 4\}$
- Kullback-Leibler (KL) divergence term for the diagnoses for these line items

The SNOMED distance is defined in Section 4.4, which describes the Unified Medical Language System (UMLS) Metathesaurus. The KL divergence terms are defined in Section 4.5.

4.1 Method of clustering line items

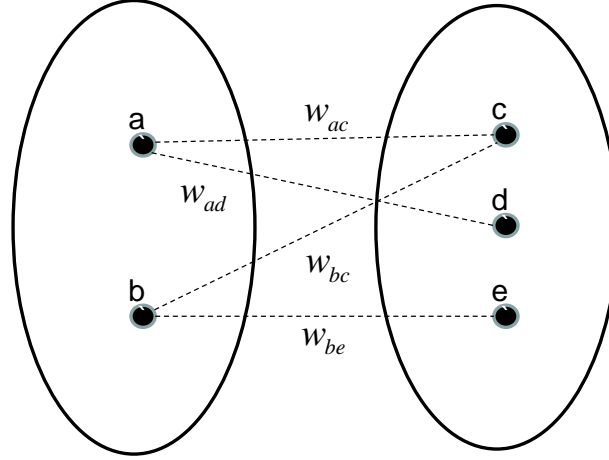
After we produced a fitted model using Random Forests algorithm, we clustered patients' line items into episodes of care. First, we created a database of all pairs of distinct line items. For each pair of line items, we assigned a weight in $[0,1]$ as follows. If the diagnoses were in the same CCS diagnosis category, we assigned a weight of 1. Otherwise, we assigned the probability from the fitted Random Forests model, based on the features for the pair of line items. We then clustered line items into episodes using the agglomerative clustering algorithm described below. We performed this clustering separately for each patient.

Notation used in this section	
$X_{i,j}$	line item j for patient i
$C_{i,j,t}$	cluster that line item $X_{i,j}$ is a member of at iteration t
L_i	number of line item pairs for patient i
$w_{a,b}$	edge weight between line items a and b
W	edge weight threshold used for allowing clusters to join
V	threshold for difference in days between two line items

Let $X_{i,j}$ denote line item j for patient i . Consider these line items as nodes in a graph. Let L_i be the number of all line item pairs (counting each of the two orderings only once) for patient i . Then for a pair of line items $X_{i,j}$ and $X_{i,k}$, let $w_{i,j,k}$ be the edge weight between the line items, as assigned based on Random Forests and the CCS diagnosis categories, as described above. Let $C_{i,j,t}$ denote the cluster that $X_{i,j}$ is a member of at iteration t , for $t \in \{0, 1, \dots\}$.

¹This research includes SNOMED Clinical Terms(R) (SNOMED CT (R)) which is used by permission of the International Health Terminology Standards Development Organisation (IHTSDO). All rights reserved. SNOMED CT (R) was originally created by The College of American Pathologists. "SNOMED" and "SNOMED CT" are registered trademarks of the IHTSDO.

Example of clustering method



The pair (a,c) is selected, and in each pair (a,e) and (b,d), line items are farther apart in days than threshold V . Therefore, the average edge weight for this join = $\frac{w_{ac} + w_{ad} + w_{bc} + w_{be}}{4}$

Figure 4.1: For a selected patient, example of method for clustering line items into episodes of care using weights from Random Forests

At $t = 0$, start with each line item in its own cluster. Choose a weight threshold W , which will constrain the clustering process to highly linked events. Let S_1, S_2, \dots, S_{L_i} be the set of line item pairs in descending order by edge weight, and for the same edge weight, sort in ascending order by the difference in days, with any further ties broken randomly. Do iterations $t = 1$ to L_i . Let $S_t = (X_{i,j}, X_{i,k})$. From the previous iteration $t - 1$, these line items are in clusters $C_{i,j,t-1}$ and $C_{i,k,t-1}$. Consider these clusters as fully connected graphs, with the line items as nodes. If these two line items are not already in the same cluster, that is, if $C_{i,j,t-1} \neq C_{i,k,t-1}$, then join these two clusters only if the average of weights on the newly created edges (the edges that were not in $C_{i,j,t-1}$ and $C_{i,k,t-1}$) is at least W . In calculating this average, we exclude edges for line item pairs that are more than V days apart (this seems to improve clustering for chronic conditions). This process of joining clusters is illustrated in Figure 4.1. The algorithm on which our final results are based used $V = -1$, which allowed a pair of clusters to join based on the largest of edge weights between pairs of items (with one item from each cluster).

4.2 Random Forests

The Random Forests algorithm, which was developed by Leo Breiman and Adele Cutler (Breiman 2001) is an ensemble classifier that expands on Breiman’s earlier work on Classification and Regression Trees (Breiman et. al. 1984). The original CART algorithm produced a binary classifier by iteratively subdividing the data, each time choosing an existing subdivision, a single variable from among the independent variables, and a splitting point from among the values of that variable. Each of these subdivisions is referred to as a “node”, and the variable and splitting point are chosen to minimize some measure of “node impurity” such as the Gini index of diversity, which for a node t is defined as $i(t) = \sum_{j \neq k} p(i|t)p(j|t)$, where the sum is taken across the categorical outcomes. For binary classification, with outcome in $\{0, 1\}$, this reduces to $i(t) = 2p(0|t)p(1|t)$. Therefore, if CART is using the Gini index to evaluate splitting node t into two nodes t_L and t_R , the reduction in the Gini index for binary classification will be $2(p(0|t)p(1|t) - (p(0|t_L)p(1|t_L) + p(0|t_R)p(1|t_R)))$. In developing the CART algorithm, the authors also introduced enhancements to the basic methodology such as “pruning” fully-grown CART trees (which performed better than introducing stopping rules), combining independent variables, and offering methods for dealing with missing values.

For each classification tree, the Random Forests algorithm introduces random selection into the original CART algorithm in two ways:

- The classification tree is based on a bootstrap sample of observations.
- At each branching step, a random subset of variables are selected as branching candidates.

These randomly generated classification trees are generated for a user-defined number of times, and then a final classification is produced for each observation based on majority of votes among trees. Alternatively an average among trees can produce class probabilities for each observation.

The Random Forests algorithm has been used in a wide range of research projects. For example, Xu and Jellick (2004) used Random Forests as an alternative to an n-gram model for predicting the next word in a document given n-1 previous words. Pang et. al. (2006) used Random Forests for evaluate which “pathways” (sets of genes) were most strongly correlated with phenotype and clinical outcomes. Guo et. al. (2004) used Random Forests to predict whether software modules were defective.

4.2.1 Implementation of Random Forests in this research

From our training subset of the sample of 250,000 patients, we generated all pairs of line items (where each of the two orderings was generated only once). This resulted in a dataset containing 1,843,232,546 line item pairs. Because this seemed larger than we could reasonably use with Random Forests, we used a collection of random samples, as described below.

We randomly partitioned this dataset of 1,843,232,546 pairs into 1,000 subsets. Next, from each of these subsets, we randomly selected a random sample of 50,000 pairs,

in such a way that exactly half of these pairs were for line items in the same CCS diagnosis category. For each of these 1,000 samples, we used Random Forests (with only 5 trees) to generate a fitted model. To produce a final fitted model, we averaged the probabilities across these 1,000 Random Forests fitted models.

4.2.2 Example from Random Forests fitted model

Figure 4.2 shows an example from the average fitted model produced by Random Forests (that is, the average across the 1,000 random samples as described in the previous subsection). In this example, we are holding constant all input features except the number of days between line items. These constant values were set to:

Same provider = No

Referring provider = Yes

SNOMED distance: diagnosis = 2, procedure = 4, diagnosis compared to procedure = 4

KL divergence term for diagnosis pair = 0

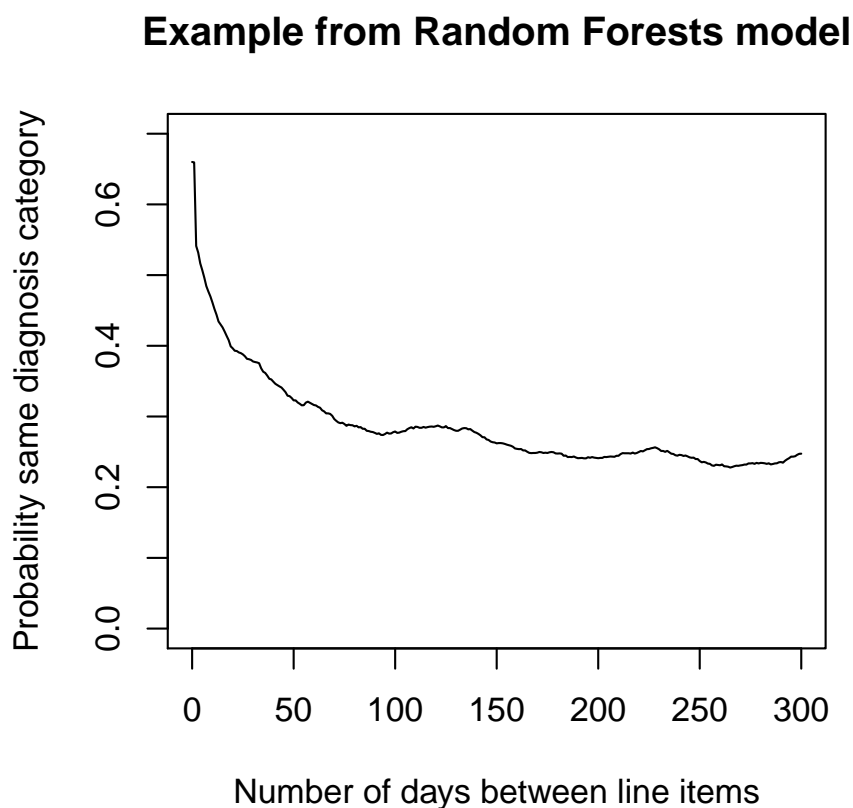


Figure 4.2: Example from fitted Random Forests model: Probability of same diagnosis category given number of days between line items

As shown in the figure, based on the fitted model from Random Forests, the probability that the diagnoses are in the same CCS category drops steeply when the difference in days is near zero, then levels off with increasing difference in days.

4.2.3 Number of variables at each Random Forests branching step

As described earlier in this section, the Random Forests algorithm allows the user to specify the number of randomly selected variables that will be branching candidates at each step of producing the classification tree. While we were fitting the Random Forests model, we noticed that this choice of number of variables has an important effect on the resulting model, as illustrated in Figure 4.3 and Figure 4.4. As in our previous Random Forests example, we are holding most of the variables constant at the values listed below, allowing the KL divergence terms to vary within each figure.

Same provider = Yes

Referring provider = No

SNOMED distance: diagnosis = 4, procedure = 4, diagnosis compared to procedure = 4

Number of days between line items = 0

In Figure 4.3 we allow Random Forests to select the best branching variable from among all 8 variables, while in Figure 4.4, Random Forests randomly selects at each step only 3 of the 8 variables as branching candidates. It appears that limiting the random selection to only 3 variables substantially improves the resulting model, because the probability function is much smoother with respect to increasing KL divergence term.

4.3 Clinical Classifications Software

We used the 2010 Clinical Classifications Software (CCS) as a preliminary indicator of clinical similarity between diagnosis codes. This software was developed over many years by the Agency for Healthcare Research and Quality (AHRQ), which is an agency of the United States government. This software maps more than 14,000 ICD-9 diagnosis codes and more than 3,900 procedure codes into clinically coherent groups. The software was developed as a tool for health care policy research.

The CCS software offers two distinct classification options for diagnoses. The first “single-level” option maps each ICD-9 diagnosis into only one of 283 diagnostic groups. For example, the following are three categories for respiratory diseases:

- Lung disease due to external agents (ICD-9 code example: Asbestosis)
- Other lower respiratory disease (ICD-9 code example: Postinflammatory pulmonary fibrosis)
- Other upper respiratory disease (ICD-9 code example: Chronic laryngitis)

We used this “single-level” option when we trained the Random Forests algorithm. In the later descriptions of our two algorithms and results, whenever we refer to a “CCS diagnosis category”, we always will be referring to this single-level option.

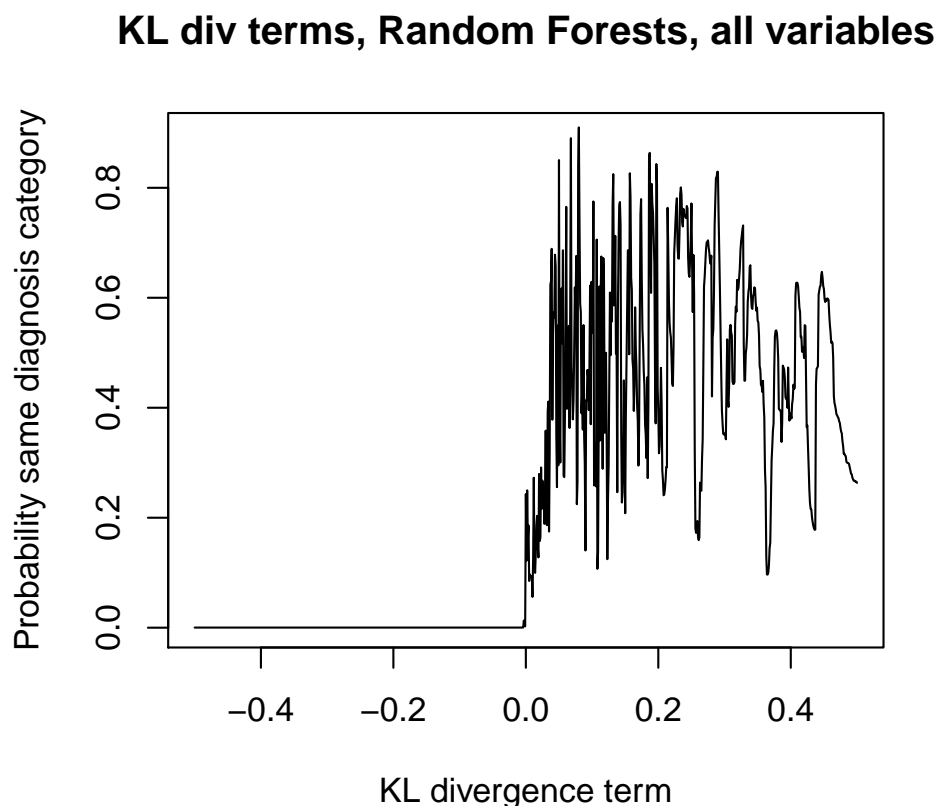


Figure 4.3: Example from fitted model that allowed Random Forests to select the best among all 8 variables at each branching step

The second “multiple-level” option maps ICD-9 codes into a branching hierarchy of diagnostic groups, with each ICD-9 code belonging to up to four diagnostic groups in the hierarchy. This hierarchy includes all of the single-level groups within the hierarchy, although not always at the same branching level of the hierarchy. For example, the three single-level respiratory disease groups listed above are part of an overall first branching-level group “Diseases of the respiratory system” (one of 18 broad diagnosis groups at this level), which includes 13 other groups from the single-level classification, such as “Asthma” and “Acute bronchitis”, as illustrated in Figure 4.5. In this figure, the 13 “single-level” diagnosis groups are contained in solid boxes, which branch off of the two diagnosis groups in the dashed boxes. The “multiple-level” CCS classification also includes 28 additional diagnosis groups that branch off of the 13 “single-level” groups in this figure.

KL div terms, Random Forests, 3 variables

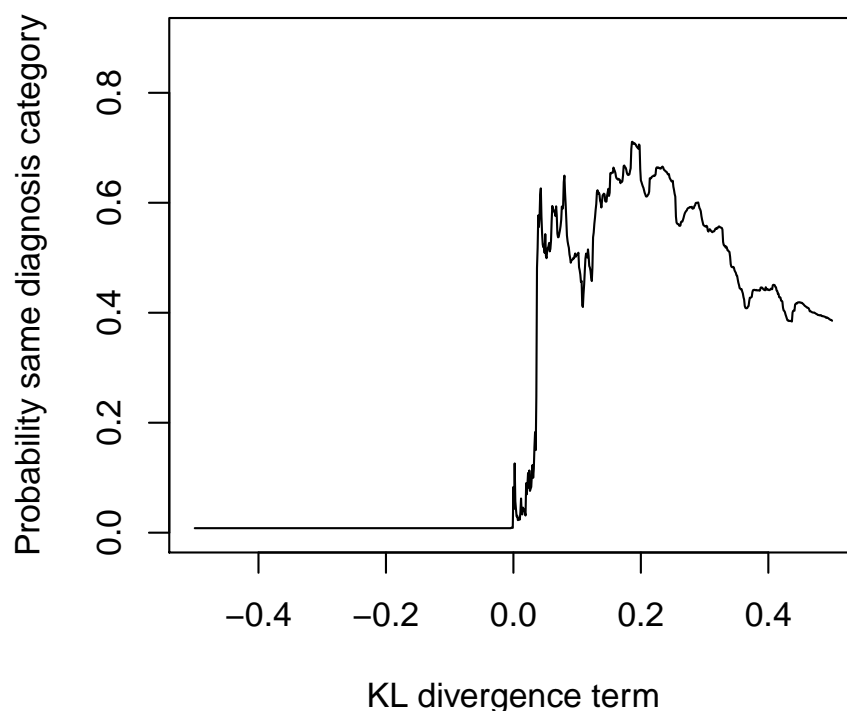


Figure 4.4: Example from fitted model that limited Random Forests to the best of 3 randomly selected variables (out of 8 variables) at each branching step

4.4 Metathesaurus

One of the features that we included in the Random Forests model was the “SNOMED distance” between two ICD-9 codes, where the distance was measured using the Unified Medical Language System (UMLS) Metathesaurus. This Metathesaurus, which is maintained by the U.S. National Library of Medicine, consists of more than 100 medical classification sets, code sets, thesauri, and lists of terms. Of these, we used only the following four sets:

- International Classification of Diseases, 9th Edition (ICD-9)
- Systematized Nomenclature of Medicine (SNOMED)
- Current Procedural Terminology (CPT)²

²CPT codes, descriptions and other data are copyright 1966, 1970, 1973, 1977, 1981, 1983-2009 Amer-

SNOMED distance	Number of diagnosis pairs
0	2,466
1	15,290
2	1,008,740
3	3,645,902
4	24,564,188

Table 4.1: Distribution of SNOMED distances between ICD-9 diagnosis code pairs

- Healthcare Common Procedure Coding System (HCPCS)

We used SNOMED to determine distances between pairs of ICD-9 diagnosis codes, and similarly between pairs of CPT/HCPCS codes (which we will refer to as “procedure codes”) and between diagnosis code - procedure code pairs. In the Metathesaurus, each diagnosis code was assigned to a concept, which also corresponded to a SNOMED code. For example, the ICD-9 code 78071 (Chronic fatigue syndrome) is assigned in the Metathesaurus to concept code C0015674, for which there is SNOMED element for “Chronic fatigue syndrome”. The SNOMED classification also defines relationships among elements. For example, chronic fatigue syndrome has a relation labeled “is a” to the SNOMED element “Post-viral disorder”, for which the SNOMED concept corresponding to ICD-9 code 0520 (Post-varicella encephalitis) also is linked via an “is a” relationship. Thus the ICD-9 codes 78071 and 0520 are linked through a path of length 2, as illustrated in Figure 4.6. Since there is not a shorter path, we assign a distance of 2 to the sibling relationship between these two ICD-9 codes. Some ICD-9 codes are assigned to the same concept in the Metathesaurus, so these are assigned a distance of 0, while parent-child relationships in SNOMED are assigned a distance of 1.

There are more than 14 thousand distinct diagnosis codes, so there are more than 196 million ordered pairs of diagnoses. Of these, Table 4.1 shows the distribution of SNOMED distances (calculated as above) up to a distance of 4.

As the SNOMED distance between pairs of diagnoses increases, the proportion of these pairs that are in the same CCS category decreases, as shown in Figure 4.7. Therefore, SNOMED distance is likely to perform well in determining the strength of relationship between pairs of health care encounters for the same patient, based on the diagnoses.

4.5 Kullback-Leibler divergence terms

Similar to collocation methods in natural language processing, we used statistics regarding pairs of line item diagnoses for the same patient and provider (where “same patient and provider” is the relationship that we substituted for NLP collocation). In NLP, these collocation methods use information about the frequency of words or phrases that are

in a specified relationship to one another (such as adjacent) to produce estimates about the underlying joint probability distributions of these words or phrases.

From our 5 percent sample data, we created a dataset containing all ordered pairs of line items that were billed for the same patient by the same provider. We excluded pairs where the line items were identical, but we included pairs where the line item was different (such as different dates of service) but the diagnosis was identical. For this analysis, we only included physician, nurse practitioner, and physician assistant claims.

Notation used in this section	
L	number of distinct diagnosis codes
M	number of ordered pairs of line items
(Y_i, Z_i)	a pair of diagnosis codes from the i th pair of line items
d_j	diagnosis code j from the set of distinct ICD-9 codes
N_j	Among the M ordered diagnosis pairs (Y_i, Z_i) , number of pairs with first diagnosis $Y_i = d_j$

Let M be the number of these ordered pairs of line items, and let (Y_i, Z_i) denote the line item diagnoses from the i th of these ordered pairs. Suppose that there are L distinct ICD-9 diagnosis codes, labeled $\{d_1, \dots, d_L\}$. Then if we let $N_j = \sum_{i=1}^M 1_{Y_i=d_j}$ be the number of pairs with the first element equal to diagnosis d_j , we then can calculate

$$p_{jk} = \frac{\sum_{i=1}^M I((Y_i, Z_i) = (d_j, d_k))}{N_j}$$

where $I()$ is the indicator function. That is, p_{jk} is the empirical conditional probability that the second diagnosis is d_k given that the first diagnosis is d_j . Similarly, we calculate the empirical marginal probability

$$q_k = \frac{\sum_{i=1}^M I(Z_i = d_k)}{M}$$

Finally, we calculate the KL divergence term $r_{jk} = p_{jk} \log \left(\frac{p_{jk}}{q_k} \right)$.

Since our goal was to estimate the strength of relationship between diagnosis pairs, we used these KL divergence terms r_{jk} rather than the conditional probabilities p_{jk} because they appeared to be superior in capturing these relationships. This is described below.

Table 4.2 shows the (empirical) conditional probability, marginal probability, and KL divergence term for selected “to-diagnoses”, conditional on ICD-9 code 493.22 (Chronic obstructive asthma, with acute exacerbation). ICD-9 code 493.22 was the “from-diagnosis” in 98,180 of our same-provider, same-patient pairs. The “to-diagnoses” in this table are the top six diagnosis codes based on the conditional probability, including statistics for ICD-9 code 493.22 paired with itself in the first row.

The last row in this table, for benign hypertension, illustrates why the KL divergence term appeared to be superior to the conditional probability in estimating the strength of relationship between diagnosis codes. Some diagnoses such as hypertension appeared frequently overall (as reflected in the marginal probability), and this frequent appearance

Diagnosis	Marginal probability	Conditional probability	KL divergence term
493.22 Chronic obstructive asthma, with acute exacerbation	0.00015	0.21929	1.592
496 Chronic airway obstruction	0.00977	0.05853	0.105
493.20 Chronic obstructive asthma, unspecified	0.00044	0.04676	0.217
491.21 Chronic obstructive bronchitis, with acute exacerbation	0.00261	0.03280	0.083
493.90 Asthma, unspecified	0.00251	0.02472	0.056
401.1 Essential hypertension, benign	0.03062	0.02072	-0.008

Table 4.2: For the 6 diagnoses with the highest empirical probability conditional on ICD-9 code 493.22 (chronic obstructive asthma), comparison of marginal probability, conditional probability, and KL divergence term

Diagnosis	Same provider	Referring provider	Same calendar quarter
493.22 Chronic obstructive asthma, with acute exacerbation	1.592	0.052	0.557
496 Chronic airway obstruction	0.105	0.156	0.124
493.20 Chronic obstructive asthma, unspecified	0.217	0.005	0.029
491.21 Chronic obstructive bronchitis, with acute exacerbation	0.083	0.070	0.111
493.90 Asthma, unspecified	0.056	0.017	0.039
401.1 Essential hypertension, benign	-0.008	-0.002	-0.004

Table 4.3: For the 6 diagnoses with the highest empirical probability conditional on ICD-9 code 493.22 (chronic obstructive asthma), comparison of KL divergence terms using line items linked because of same provider, referring provider, and same calendar quarter

resulted in high conditional probabilities even where the “from-diagnosis” apparently was clinically unrelated.

Table 4.2 only shows results based on same-provider links, but the Medicare claims data contained a number of other possible link choices. For example, we could have linked claims by referring provider (for the same patient, one line item lists a referring provider, which appears as the provider on another line item), and temporal links (for the same patient, two line items from different claims have dates of service in the same calendar quarter).³ As illustrated in Table 4.3, the resulting KL divergence terms resulting from these alternative linking methods were somewhat different from those resulting from same-provider links.

³In the 5 percent sample data, we did not have the date of service, only the calendar quarter of service.

Diagnosis i	Diagnosis j	Transition probability p_{ij}
4011	4019	0.016
4011	V0481	0.023
4019	4011	0.019
4019	V0481	0.020

Table 4.4: Initial estimated transition probabilities among selected hypertension and vaccine codes, based on same provider link

4.6 Modification of KL divergence terms

From the previous section on KL divergence terms, we let p_{ij} be the empirical transition probability from diagnosis d_j to d_k . Upon examination of this transition matrix, we noticed that it contained transition probabilities that apparently do not reflect the clinical relationships among diagnoses. Perhaps these arise (1) due to small sample sizes among pairs of diagnosis codes or (2) due to our choice of collocation data (same provider, same patient collocations). For example, the following charts illustrate the transition probabilities \tilde{g}_{ij} among three diagnosis codes. Two of these codes are for benign hypertension (ICD-9 code = 4011) and unspecified hypertension (4019), between which we might expect relatively high transition probabilities, and the third is for influenza vaccine (V0481), which we might expect to be only weakly related to the hypertension codes. However, as shown in Table 4.4, our initial transition matrix has a slightly higher transition probability from either of the hypertension codes to the vaccine code than in either direction between the two hypertension codes.

We then developed a method for adjusting this preliminary transition matrix P_0 based on our conjecture that the actual transition matrix P that we are estimating has an ordering of the diagnoses (an ordering identical for rows and columns in which the most related diagnoses are consecutive) such that P is almost the same as another matrix Q that only has nonzero elements along a block diagonal. In other words, let C_a and C_b , with $a \neq b$, be two distinct sets of diagnoses, where the diagnoses within C_a are clinically related to one another, and similarly for C_b , but the diagnoses for C_a and C_b are not clinically related, then the transition probabilities between C_a and C_b (and conversely between C_b and C_a) are zero in matrix Q and nearly zero in matrix P . If we assume that Q is aperiodic, then because Q is finite (there are finitely many diagnoses), then there exists a stationary distribution π for Q such that $\pi = (\pi_1, \pi_2, \dots, \pi_{N_c})$, where N_c is the number of these sets of related diagnoses, and for each set C_i , π_i is a stationary distribution (with at least one nonzero element) for that block of the transition matrix Q .

In this case, if the assumption that nonzero elements exist only along a block diagonal (given suitable ordering) is only approximately but not strictly true, we might still improve our original estimated transition matrix by matrix multiplication. That is, there might be some $s \in \{1, 2, \dots\}$ such that P_0^s is a better estimate for P than P_0 . On the other hand, since the conjecture is not strictly true (the nonzero elements of P are not restricted to a block diagonal), and therefore the stationary distribution τ for P is not simply the

Billing category	Number of providers	Percent of providers
Only diagnosis 4011	752	26.4%
Only diagnosis 4019	682	23.9%
Both 4011 and 4019	1416	49.7%
Total	2,850	100.0%

Table 4.5: Among providers in 5 percent sample with at least 30 patients with diagnoses 4011 and/or 4019, percent billing only one of these diagnoses

concatenation of the stationary distributions for diagnosis group blocks, $\{\tau_i\}$, then for s large, P_0^s might be a worse estimate for P than P_0 .

Upon review of matrices P_0^s for a number of values of s , it seems to us that a relatively small value of s , such as $s = 4$, improves the estimated transition matrix, but the estimate degrades for higher values of s . This is illustrated for the hypertension and vaccine codes in Figure 4.8. For the KL divergence feature that we used as one of the inputs to Random Forests, we used transition probabilities from matrix P_0^4 rather than P_0 .

It is not yet clear to us why P_0^4 is an improvement on P_0 . One possibility is that much of the inaccuracy in P_0 arises from small sample sizes used in producing the individual transition probabilities, and that matrix multiplication increases the sample sizes brought to bear on any estimated transition probability from diagnosis d_i to d_j (because in P_0^4 , the transition probabilities are calculated based on all paths of length 4 from d_i to d_j , and some of these paths might involve larger sample sizes). If this were true, that small sample sizes explain the improvement then some smoothing method might be a more suitable solution, as is often done in NLP applications. However, this does not explain the hypertension/vaccine results shown above, because these codes were among the most frequently occurring in the claims data, and so all of these estimated transition probabilities involved large sample sizes. Another possibility is that providers were more likely to bill using only one or the other of the two hypertension codes in this example, but seldom both. In this case, the matrix multiplication that produced P_0^4 would increase the estimated transition probabilities between the two hypertension codes via other diagnoses to which both hypertension codes were strongly related, and closer examination of transition probabilities in P_0 along short paths between the hypertension codes supports this idea.

Based on the results displayed in Table 4.5, it does appear that providers are more likely to bill using only one or the other of these two hypertension codes. We calculated these statistics from our 5 percent sample of Medicare data, limited to physicians, nurse practitioners, and physician assistant providers. In this table, we show that among providers with at least 30 patients with diagnoses 4011 and/or 4019 in the 5 percent sample, 50.3 percent only billed one of the two codes for all patients in the 5 percent sample.⁴

⁴This was a 5 percent sample of patients, not providers, so that for any patient we could analyze all line items for years 2006 and 2007, but for any provider we only could analyze a random sample of patients.

CCS diagnosis groups, respiratory

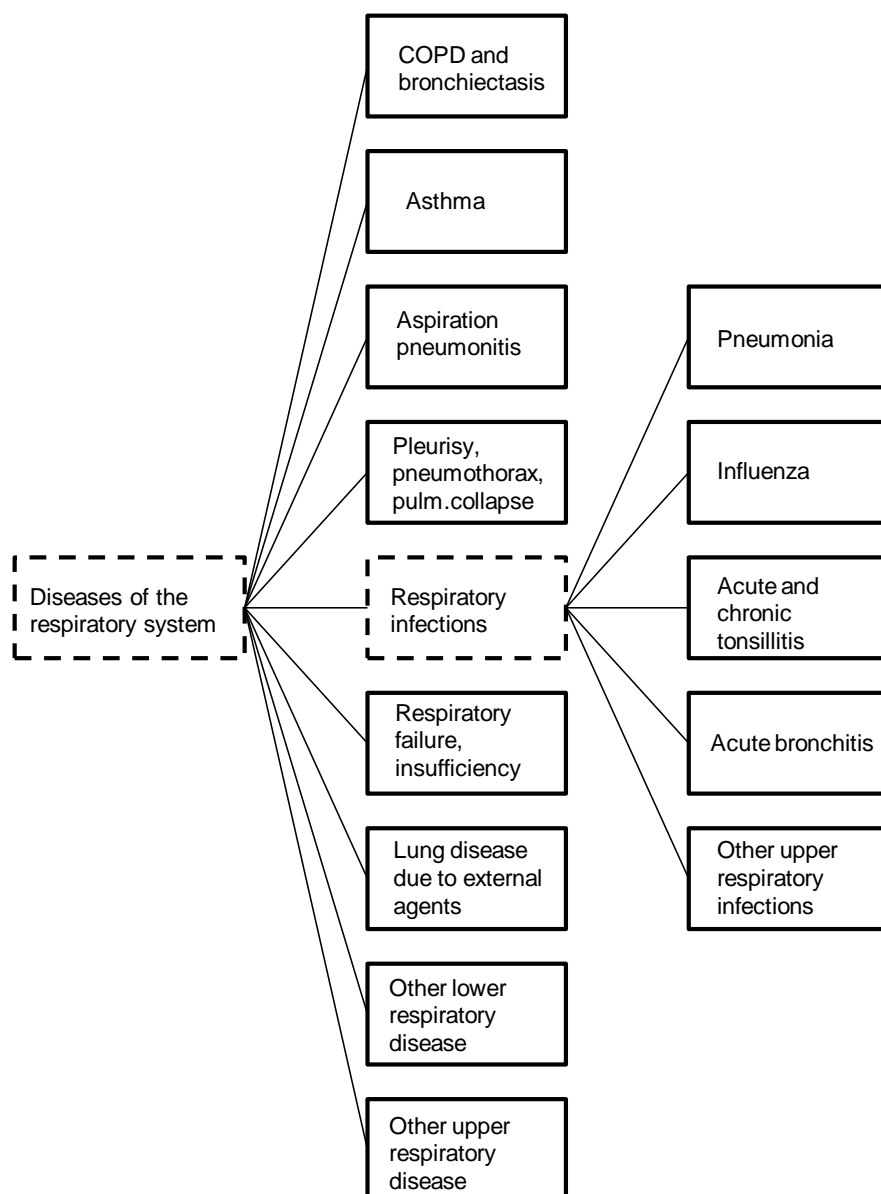


Figure 4.5: CCS “single-level” diagnosis groups that branch from “Diseases of the respiratory system”

Illustration of SNOMED distance

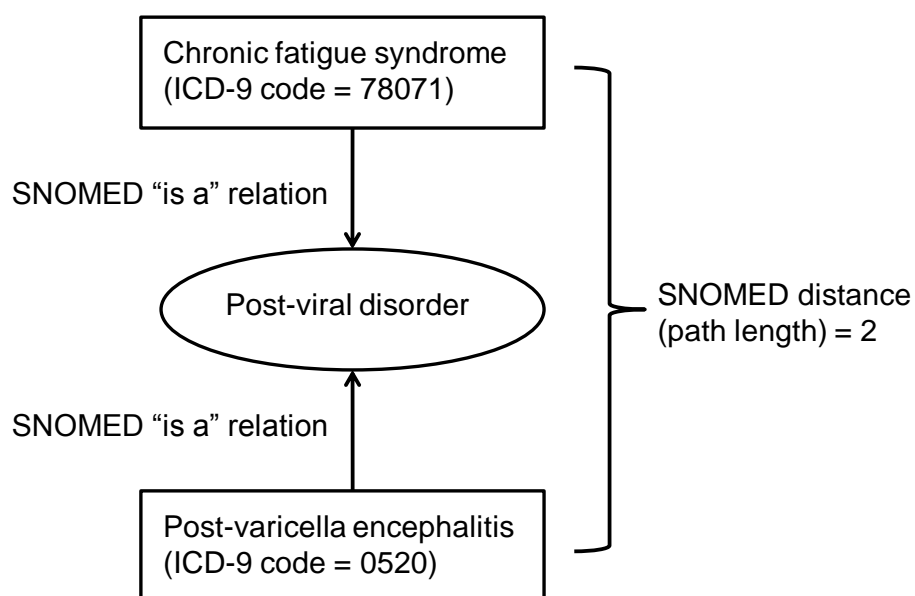


Figure 4.6: Illustration of how we calculate SNOMED distance between diagnoses chronic fatigue syndrome and post-varicella encephalitis

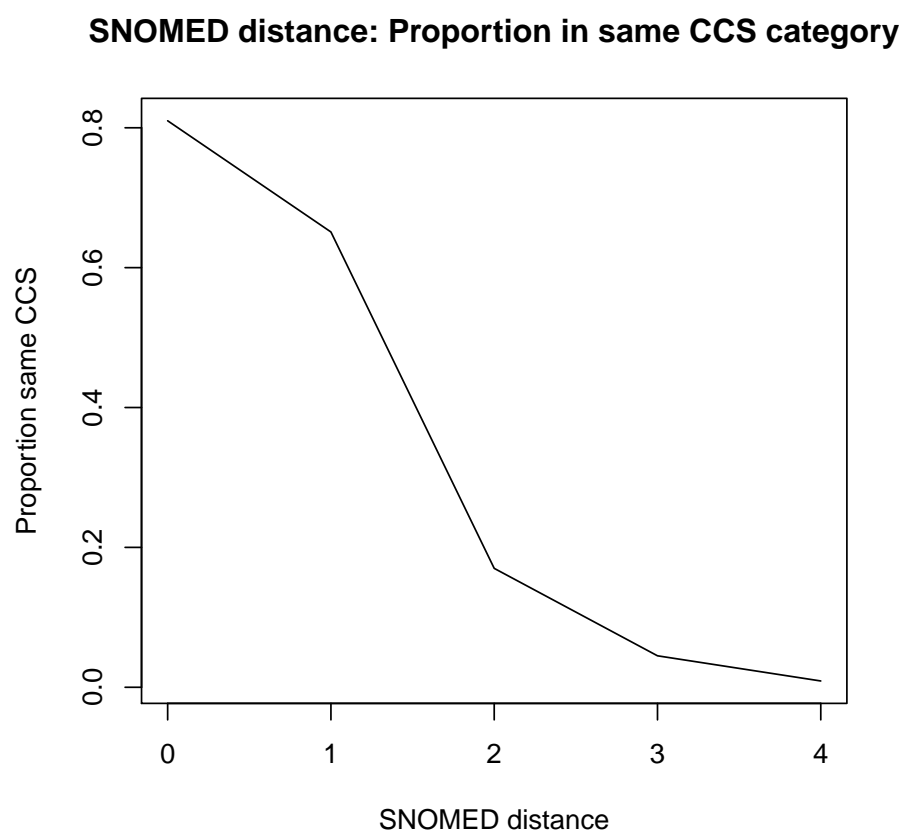


Figure 4.7: Among distinct pairs of ICD-9 diagnoses, proportion in the same CCS diagnosis category given SNOMED distance between the diagnoses

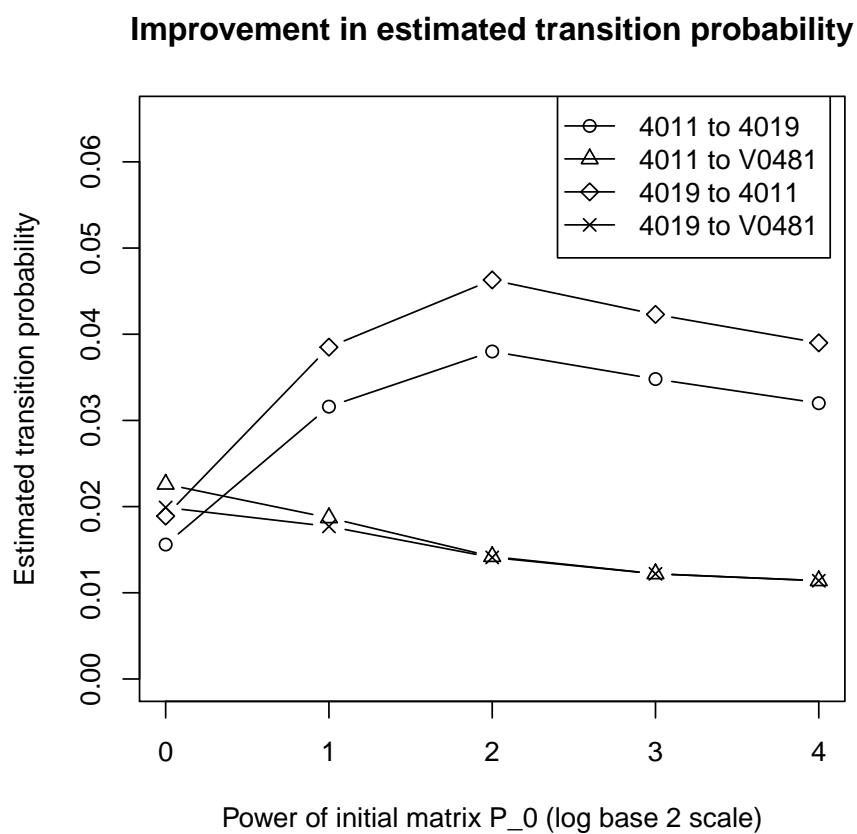


Figure 4.8: Improvement in estimated transition probabilities among selected hypertension, vaccine codes, using matrix multiplication

Chapter 5

Second method for clustering Medicare claims: generative model, EM algorithm

This second method used a statistical model for how Medicare line items are generated within episodes of care. In contrast, our first method did not use an explicit generative model, but instead modeled the relationship between (1) features of line item pairs and (2) the clinical similarity of the diagnoses within the pairs. In this second method, we used a Monte Carlo version of the EM algorithm to estimate parameters. After estimating parameters, we clustered line items into episodes of care so that we approximately maximized the probability (based on the generative model) of the resulting episode configuration.

We used the following statistical model to approximate the process by which episodes of care, line item diagnosis categories, and dates of service are generated in Medicare claims data. Let $t \in \{1, \dots, T\}$ be the set of days in some time interval. In this chapter, we will refer to a unique combination of patient, day (date of service), and diagnosis category as a “diagnostic event ” or simply an “event”.

For each patient (independently of other patients), this generative model has two steps: (1) randomly generate the diagnosis category and date of service combinations that will start new episodes, and (2) for each of these new episode starts, randomly generate the episode as a Markov chain. These two steps are described in more detail below.

5.1 Step 1: Randomly generate the starting days and diagnosis categories of new episodes

Let D be the number of single level CCS diagnosis categories. For each day $t \in \{1, \dots, T\}$ and each diagnosis category $i \in \{1, 2, \dots, D\}$, let $X_{i,t}$ be a Bernoulli random variable such that $X_{i,t} = 1$ if and only if an episode starts on day t with diagnosis i . For each i , we will assume that the set of random variables $\{X_{i,t} : t \in \{1, \dots, T\}\}$ are identically distributed. During this step we will thus generate the following $T \times D$ matrix:

$$\mathcal{X} = \begin{pmatrix} X_{1,1} & X_{1,2} & \dots & X_{1,D} \\ \vdots & \vdots & \ddots & \vdots \\ X_{T,1} & X_{T,2} & \dots & X_{T,D} \end{pmatrix}$$

This matrix of Bernoulli random variables requires TD parameters

$\{p_{it} : t \in \{1, \dots, T\}, i \in \{1, \dots, D\}\}$, but because for any $i \in \{1, \dots, D\}$ we have $p_{it} = p_{is}$ for any $t, s \in \{1, \dots, T\}$, we only require D parameters for which we suppress the day subscript and simply write $p_i = P(X_{i,t} = 1)$ for any t .

We also assume that the larger set of random variables $\{X_{i,t} : t \in \{1, \dots, T\}, i \in \{1, \dots, D\}\}$ are independent but not necessarily identically distributed. We realize that this assumption is not entirely realistic, but in fitting this model to Medicare claims data we found that the estimated values of the parameters $\{p_i : i \in \{1, \dots, D\}\}$ are so small that this approximation seems close enough, and it greatly simplifies parameter estimation.

5.2 Step 2: For each new episode, randomly generate the episode as a Markov chain

First, we number the episodes that started in step 1, with episode numbers in $\{1, 2, \dots, \mathcal{N}_i\}$ assigned in order by row of the matrix \mathcal{X} , where $\mathcal{N}_i = \sum_{i=1}^D \sum_{t=1}^T X_{i,t}$ is the number of episodes for this patient. For example, suppose that $T = 3$ and $D = 2$, and suppose during step 1 we randomly generate the following matrix:

$$\mathcal{X} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{pmatrix}$$

Then $\mathcal{N}_i = 2$, episode 1 starts on day 1 with diagnosis category 2, and episode 2 starts on day 3 with diagnosis category 1.

Next, for each episode $m \in \{1, \dots, \mathcal{N}_i\}$ and each event number $k \in \{1, 2, \dots\}$, we generate the following three random variables:

Random variable	Description
$S_{m,k}$	Indicates whether episode m is in the “stopped” state at event k
$Z_{m,k}$	diagnosis category for episode m , event k
$T_{m,k}$	date of service in $\{1, \dots, T\}$ for episode m , event k

This second step of the model is illustrated in Figure 5.1 for an episode that generates three events before stopping.

Second step of generative model for episodes


<u>Description</u>		<u>Random variables</u>	<u>Selected probabilities</u>
Given starting matrix \mathcal{X} , the initial diagnosis category for episode m is i and the day is t_1 . The episode does not stop at this initial event.		$Z_{m,1} = i$ $T_{m,1} = t_1$ $S_{m,1} = 0$	$P(S_{m,1} = 0 Z_{m,1} = i) = 1 - h_i$
Event 2 has diagnosis category j and takes place $t_2 - t_1$ days after event 1. The episode does not stop at event 2.		$Z_{m,2} = j$ $T_{m,2} = t_2$ $S_{m,2} = 0$	$P(Z_{m,2} = j Z_{m,1} = i) = g_{ij}$ $P(T_{m,2} - T_{m,1} = t_2 - t_1 Z_{m,1} = i) = q_{i,t_2-t_1}$
Event 3 has diagnosis category k and takes place $t_3 - t_2$ days after event 2. The episode stops at event 3.		$Z_{m,3} = k$ $T_{m,3} = t_3$ $S_{m,3} = 1$	$P(S_{m,3} = 1 Z_{m,3} = k) = h_k$

Figure 5.1: For a selected patient, illustration of how the second step of our statistical model could generate three diagnostic events within an episode of care

For the first event in the episode, that is when $k = 1$, $Z_{m,1}$ and $T_{m,1}$ are assigned the starting diagnosis category and date of service given by the starting matrix \mathcal{X} . Then for all $k \in \{1, 2, \dots\}$, let $h_i = P(S_{m,k} = 1 | Z_{m,k} = i, S_{m,k-1} \neq 1)$ be the probability that episode m stops at event k given that the episode is not already in the “stopped” state. Also for $k \in \{1, 2, \dots\}$, $P(S_{m,k} = 1 | S_{m,k-1} = 1) = 1$, which means that once an episode reaches the “stopped” state it remains in that state. For $k \in \{2, 3, \dots\}$, let $g_{ij} = P(Z_{m,k} = j | Z_{m,k-1} = i)$ be the probability that event k of episode m has diagnosis category j given the previous event had diagnosis category i . The collection of these diagnosis category transition probabilities forms a transition matrix G . Finally for $k \in \{2, 3, \dots\}$ and $t \in \{0, 1, \dots\}$, let $q_{it} = P(T_{m,k} - T_{m,k-1} = t | Z_{m,k-1} = i)$ be the probability that the number of days between event $k - 1$ and event k is t given that event $k - 1$ had diagnosis category i . Combining these transition probabilities, we denote the transition probability from one event with diagnosis category i to the next with diagnosis category j at a distance

Notation used in this chapter	
D	number of diagnosis categories
M	number of dates of service in time window ($M = 730$ in our implementation)
K	number of patients in data used to train EM algorithm
d_ν	diagnosis category for event ν
t_ν	date of service number for event ν
e_ν	episode number for event ν
s_ν	indicator for whether the episode containing event ν stopped with event ν
p_j	episode-generating probability for an episode that starts with diagnosis category j
h_j	probability that episode with latest diagnosis category j will stop at that event
g_{ij}	transition probability from diagnosis category i to j
G	matrix of transition probabilities g_{ij}
q_{ik}	for an episode with latest diagnosis category i that will not stop at the latest event, probability from Geometric distribution on $\{0, 1, \dots\}$ (with a parameter that depends on i) that the next event in the episode will be k days later
θ	set of all parameters for this model
$\hat{\theta}_k$	in EM algorithm, estimate for θ at iteration k
X	For a set of one or more patients, the observed information (diagnosis category, date of service, patient identifiers)
Y	For a set of one or more patients, the (unobserved) information about assignments of events to episodes

t days later by $f_{ijt} = q_{it}g_{ij}$.

5.3 Example of calculating probability of generating episodes

The following is an example of the calculation of the probability that three events will be generated for one patient, in the specific configuration described below. Let t_1, t_2 , and t_3 be days with $1 \leq t_1 < t_2 < t_3 \leq M$, where M is the number of days in the time window. Suppose that the episode labels are $e_1 = e_3 = 1$ and $e_2 = 2$. That is, the events on days t_1 and t_3 are in the same episode, and the event on day t_2 starts a different episode. We will label the diagnostic categories for these events d_1, d_2 , and d_3 respectively. Suppose that episode 1 stops on day t_3 but episode 2 continues past the last day M in our observed time period. We denote this by letting $s_1 = 0$, $s_2 = 0$, and $s_3 = 1$, where these s_i are indicators for whether the corresponding episodes stopped on these days.

Then for this patient, the probability of producing this configuration,

$$P((d_1, t_1, e_1 = 1, s_1 = 0), (d_2, t_2, e_2 = 2, s_2 = 0), (d_3, t_3, e_3 = 1, s_3 = 1))$$

is the product of the factors in Table 5.1.

In this table, $G_{2, M-t_2}$ is the cumulative distribution function (cdf) corresponding to the Geometric probability function g for diagnosis category 2, which determines the

Factor	Description
$\prod_{i=1}^D (1 - p_i)^{t_1-1}$	probability that no new episodes begin during days 1 through $t_1 - 1$
$p_1 \prod_{i=2}^D (1 - p_i)$	probability that an episode begins with diagnosis category 1 (and no other episodes begin) on day t_1
$\prod_{i=1}^D (1 - p_i)^{t_2-t_1-1}$	probability that no new episodes begin during days $t_1 + 1$ through $t_2 - 1$
$p_2 (1 - p_1) \prod_{i=3}^D (1 - p_i)$	probability that an episode with diagnosis category 2 (and no other episodes) begins on day t_2
$\prod_{i=1}^D (1 - p_i)^{M-t_2}$	probability that no new episodes begin during days $t_2 + 1$ through day M
$(1 - h_1) f_{1,2,t_3-t_1} h_3$	probability that episode 1 generates a second event on day t_3 and then stops
$(1 - h_2) (1 - G_{2,M-t_2})$	probability that episode 2 generates a second event on some day after day M

Table 5.1: Calculating the probability of generating two episodes

number of days from an event with diagnosis category 2 to the next event in the same episode, so that $G_{2,M-t_2} = \sum_{k=0}^{M-t_2} g_{2,k}$.

5.4 EM algorithm used to estimate parameters

To estimate the parameters of this model, we used a Monte Carlo version of the Expectation Maximization (EM) algorithm. EM algorithms often are used to estimate parameters from models that are being fit to data in which the random variables are not all observed. In our case, we observed variables for the patients' diagnoses and dates of service but not the random variables corresponding to the episode labels.

The EM algorithm is used as an approximation for estimating parameters through the method of Maximum Likelihood. The following gives details regarding the EM algorithm as applied to our generative model. Let $X = (X_1, X_2, \dots, X_N)$ be the vector of observed random variables for a collection of N patients, and let $Y = (Y_1, Y_2, \dots, Y_N)$ denote the unobserved random variables. If we had observed both X and Y , we could estimate the parameters by maximum likelihood: $\hat{\theta} \in \sup_{\theta \in \Theta} \log P_{\theta}(X, Y)$. However, since we do not observe Y , the EM algorithm uses an iterative substitute. Let $\hat{\theta}_0$ be some initial estimate for θ . Then we update this parameter estimate by

$$\hat{\theta}_{k+1} \in \sup_{\theta} E_{\hat{\theta}_k} \log P_{\theta}(X, Y)$$

where the expectation uses the probability distribution of Y with respect to the parameter estimate $\hat{\theta}_k$ from the previous iteration. In the version of the EM algorithm that we implemented, the expectation in each EM iteration is approximated by a Monte Carlo simulation. We used a Monte Carlo simulation because of the computational complexity of directly computing this expectation.

5.4.1 Each EM iteration, episodes assigned using posterior probabilities

As part of our Monte Carlo EM algorithm, we randomly assigned diagnostic events to episodes based on posterior probability distributions. Specifically, for a specific patient, suppose we have assigned the first k events in time-sequence to episodes (e_1, \dots, e_k) . Then we randomly assigned event $k+1$ to an episode based on the probability $P(e_{k+1}|e_1, \dots, e_k)$, where here e_{k+1} is a random variable.

The following is a detailed example of how we calculated these posterior probabilities. To show this calculation carefully, however, we will need to use a slightly more complex alternative notation for the information that we can observe regarding diagnostic events. We will only use this alternative notation for the calculation in this example.

Let X be a random variable representing a diagnostic event, where X contains the following random elements:

seq(X) = time-sequence order (among all events for the same patient)
 day(X) = date-of-service
 diag(X) = diagnosis category
 epi(X) = episode label

$\text{stop}(X)$ = indicator whether the episode ended with this event

To condense this notation, we will let

$$I(X) = (\text{seq}(X), \text{day}(X), \text{diag}(X), \text{epi}(X))$$

denote this set of information about X , and we will let

$$J(X) = (\text{seq}(X), \text{day}(X), \text{diag}(X), \text{epi}(X))$$

be the same information about X as $I(X)$ but without the episode label. Because we will only be using this notation temporarily, and we will switch back to the previous notation partway through the example, we list below the correspondence between this temporary notation and our usual notation about events.

If $\text{seq}(X) = i$, which means that for this patient, the diagnostic event is the i th in time-sequence order, then:

$$\begin{aligned} \text{day}(X) &= t_i \\ \text{diag}(X) &= d_i \\ \text{epi}(X) &= e_i \end{aligned}$$

Now suppose that for our selected patient we have observed the sequence order, diagnosis category, and dates of service for three diagnostic events X , Y , and Z , that $(\text{seq}(X), \text{seq}(Y), \text{seq}(Z)) = (1, 2, 3)$, and that in our current iteration of the Monte Carlo EM algorithm, we have randomly assigned episode labels to X and Y . We then want to compute $P(\text{epi}(Z) = k | I(X), I(Y), J(Z))$ for any possible episode label k . For example, suppose $(\text{epi}(X), \text{epi}(Y)) = (1, 2)$. Then by Bayes' rule:

$$\begin{aligned} P(\text{epi}(Z) = k | I(X), I(Y), J(Z)) &= \\ \frac{P(J(Z) | I(X), I(Y), \text{epi}(Z) = k)}{\sum_{m=1}^3 P(J(Z) | I(X), I(Y), \text{epi}(Z) = m)} \end{aligned}$$

For $m = 1$, the conditional probability in the denominator is the product of the terms in Table 5.2, which we can more clearly express in our original notation for events.

Once we have randomly assigned diagnostic events to episodes, we estimate parameters in preparation for the next EM iteration. If we were not using a Monte Carlo version of the EM algorithm, our parameter estimates at iteration $i + 1$ would be

$$\hat{\theta}_{i+1} = \arg \max_{\theta} E_{\hat{\theta}_i} \log P_{\theta}(V, W)$$

where V denotes the observed random variables (diagnoses, dates of service, patient identifiers) and W denote the unobserved variables (episode labels, episode-stopping indicators). However, calculating this expectation is computationally infeasible. The probability $P_{\theta}(V, W)$ factors because of independence across patients, so that

$$P_{\theta}(V, W) = \prod_{k=1}^K P_{\theta}(V_k, W_k)$$

Factor	Description
$\prod_{i=1}^D (1 - p_i)^{M-t_2}$	probability that no new episodes begin during days $t_2 + 1$ through M
$\frac{h_2 + ((1 - h_2) (1 - G_{2,M-t_2}))}{h_2 + ((1 - h_2) (1 - G_{2,t_3-t_2}))}$	probability that episode 2 (which contains event Y) either stopped on day t_2 or continues after day M , conditional on the observed information that episode 2 either stopped on day t_2 or continued up to day t_3
$\frac{(1 - h_1) f_{1,3,t_3-t_1}}{h_1 + ((1 - h_1) (1 - g_{1,t_2-t_1-1}))}$	probability that episode 1 (which contains event X) produces its next event with diagnosis category 3 on day t_3 , conditional on the observed information that episode 1 had either stopped on day t_1 or continued up to day t_3

Table 5.2: Example of calculating posterior probabilities

Parameter	Estimate
p_i	number of episodes beginning with diagnosis category i divided by $M \times K$.
mean parameter for $q_{i,\cdot}$	mean distance to next event in episode among events with diagnosis category i
g_{ij}	transition probabilities estimated using KL positive-valued divergence terms
h_i	of events with diagnosis category i , proportion that were the last event in the episode

Table 5.3: Parameter estimates during EM iterations

where K is the number of patients in the data we are using to estimate parameters using the EM algorithm. However, for each patient, the assignment of events to episodes depends on the previous assignments in time sequence.

Because of this computational complexity, we used a Monte Carlo simulation to approximately calculate the parameter updates $\hat{\theta}_{i+1}$. Specifically, at the $i + 1$ iteration, we randomly assigned the diagnostic events for patients in the training set to episodes using parameters $\hat{\theta}_i$ from the previous iteration (as shown earlier in this subsection). We then estimated parameters $\hat{\theta}_{i+1}$ using statistics based on these randomly assigned episodes, as shown in Table 5.3.

5.5 Final assignment of diagnostic events to episodes

Using the estimated parameters $\hat{\theta}_i$ that were produced by running the EM algorithm for i iterations, we produced a final assignment of diagnostic events to episodes.¹ Ideally, we would have assigned events to episodes \hat{W} as

$$\hat{W}_{\text{EM},i} = \arg \max_{\tilde{W}} \log P_{\hat{\theta}_i}(V, \tilde{W})$$

where V denotes the observed information (diagnoses, dates of service, patient identifiers, and W denotes the information about episode assignments). However, this is too computationally complex, so we approximated this optimum assignment by assigning events to

¹For a comparison of our results across the EM iterations, see Figure 6.2

episodes one by one in time sequence for each patient, looking ahead some $k \in \{0, 1, \dots\}$ to evaluate the configuration with the maximum probability across the next k events, conditional at each step on the episode assignments already made for the patient.

More formally, consider the diagnostic events for a single patient. We assign the first event in the time sequence to episode 1, so that $e_1 = 1$. Then for $j > 1$, we let τ_j be the number of distinct episodes constructed for this patient so far, and then we assign e_j such that $e_j \in \{1, 2, \dots, \tau_j + 1\}$ and

$$(e_j, w_{j+1}, \dots, w_{j+k}) \in \arg \max_{\tilde{W}_{[j, j+k]}} P_{\hat{\theta}_i} \left(\left(V_{[j, j+k]}, \tilde{W}_{[j, j+k]} \mid \left(V_{[1, j-1]}, W_{[1, j-1]} \right) \right) \right)$$

where $W_{[\alpha, \beta]} = (w_\alpha, \dots, w_\beta)$.

Chapter 6

Comparison of episodes against test set

To evaluate the performance of the two methods described in the previous chapters, we compared the resulting clusters for a test set consisting of 50 patients, who were selected using simple random sampling from the population of all patients in our identifiable file who had at least one Carrier claim in each year of 2007 and 2008. For each patient in this test set, the line items were manually clustered into episodes of care by each of three Nurse Practitioners, who were working entirely independently of one another.

6.1 Protocol for Nurse Practitioner review

For each patient, reviewers were given a list of all line items for service dates in 2007-2008, with the following information:

- Patient age category (in 5-year increments)
- Patient gender
- From- and to-dates of service
- ICD-9 line item diagnosis code and description
- CPT/HCPCS procedure code and description
- Place of service
- Provider identifier and specialty
- Referring provider identifier

For inpatient claims with service dates in 2007-2008, reviewers also were given the from- and to-dates of the inpatient stays. For each of these patients, reviewers clustered the line items into episodes of care, which we defined as sequences of care that were coherent,

both in terms of the patient’s disease, injury, or reason for care and also in terms of the process of resolving the care given to these patients by providers.

Regarding the coherence of the disease, injury, or reason for care, reviewers could determine that a sequence of care for the same provider, even on the same date of service, actually consisted of two or more distinct episodes, if reviewers determined that the provider was providing care for two or more clinically distinct diseases, injuries, or reasons for care. Regarding the coherence of the care given by providers, reviewers could determine that a sequence of care for the same or similar diseases or injuries actually consisted of two distinct episodes, if reviewers determined that the earlier of these sequences of care had been completely resolved (even if the underlying condition might still be present, such as for a chronic condition). For example, two annual flu vaccines would be contained in two distinct episodes of care, while a series of kidney dialysis treatment visits would be clustered into one episode.

A more difficult case involved screenings that might or might not be routine. For example, a patient who received two glaucoma screenings as part of annual eye exams would be considered to have two distinct episodes of care, while another patient with much more frequent glaucoma screenings and other care related to glaucoma would have only one episode of care for all of the care related to glaucoma.

6.2 Comparison of review results

To compare results among Nurse Practitioner reviews and our two algorithms, we used three approaches: (1) recall of complete episodes, (2) one-to-one accuracy, and (3) average pairwise agreement within a distance of k service dates.

We also considered whether a relatively naive model might achieve good results, and so we included two additional models. The first model, which we will refer to as the ‘daily episode’ model, simply assigned all line items for a given patient to the same episode if and only if they had the same date of service. The second naive model, which we will call the ‘diagnosis category’ model, assigned all line items to the same episode if and only if they belonged to the same AHRQ diagnosis category.

6.3 Recall of complete episodes

For a patient $i \in \{1, \dots, 50\}$, let $\mathcal{L}_i = \{1, 2, \dots, n_i\}$ be the set of all n_i line items for that patient. Then for $j \in \mathcal{L}_i$, let $e_{i,j}^T$ and $e_{i,j}^C$ be the episode numbers for the j th line item for the i th patient, as reviewed by reviewers T and C respectively (which we will consider the ‘truth’ and ‘comparison’ reviewers). Then we will determine, of the episodes assigned by reviewer T to the line items among the 50 patients, what proportion of the episodes assigned by T contained exactly the same line items (and no more) according to reviewer C .

Formally, we calculate:

$$S^{T,C} = \frac{\sum_{i=1}^{50} \sum_{j=1}^{m_i^T} g(i,j)}{\sum_{i=1}^{50} m_i^T}$$

NP	RF	Seq.	DiagCat	Daily
0.413	0.466	0.398	0.455	0.303

Table 6.1: Recall of complete episodes

where m_i^T is the number of episodes for patient i according to reviewer T , and $g(i, j)$ is an indicator function that takes the value 1 if and only if there exists an episode among the episodes created by reviewer C that contains exactly the same line items as episode $e_{i,j}^T$.

To calculate average “recall of complete episodes” among the Nurse Practitioner reviewers, we averaged the above result across the 6 possible assignments to T and C among the 3 Nurse Practitioners, labeled NP1, NP2, and NP3 below.

$$A_{NP}^{\text{recall}} = \frac{S^{NP1,NP2} + S^{NP2,NP1} + S^{NP1,NP3} + S^{NP3,NP1} + S^{NP2,NP3} + S^{NP3,NP2}}{6}$$

Similarly, for each model M (where M is one of Random Forests model, sequence model, daily episode model, or diagnosis category model), we calculated the average “recall of complete episodes” by comparing the model to each of the three Nurse Practitioners, assuming in turn that each Nurse Practitioner’s review constituted the ‘truth’.

$$A_M^{\text{recall}} = \frac{S^{NP1,M} + S^{NP2,M} + S^{NP3,M}}{3}$$

As shown in the Table 6.1, the Random Forests (RF) model performed better than the Nurse Practitioner (NP) annotations according to this measure, while the daily episode model performed dismally. The AHRQ diagnosis categories performed nearly as well as the Random Forests model on this measure.

6.4 One-to-one accuracy

We define this method as given in Elsner and Charniak (2010). For each patient i and each pair of reviewers T and C , we pair episodes to maximize the number of line items in the overlapping episodes. More formally, let \mathcal{P}_i^T and \mathcal{P}_i^C be partitions (that is, episode annotations) of the line items \mathcal{L}_i from patient i according to reviewers T and C respectively. Specifically, $\mathcal{P}_i^T = \{P_{i,1}^T, \dots, P_{i,m_i^T}^T\}$, where each $P_{i,j}^T \subseteq \mathcal{L}_i$ is a subset of the line items for patient i , $\bigcup_{j=1}^{m_i^T} P_{i,j}^T = \mathcal{L}_i$, and $P_{i,j}^T \cap P_{i,k}^T = \emptyset$ for $j \neq k$. Similarly, \mathcal{P}_i^C is a partition of line items \mathcal{L}_i according to reviewer C . Let $F_i = \{f : \mathcal{P}_i^T \rightarrow \mathcal{P}_i^C : f \text{ is one-to-one}\}$ be the set of all one-to-one maps from the partition \mathcal{P}_i^T to partition \mathcal{P}_i^C . Note that the domain of f need not be all of \mathcal{P}_i^T (that is, $f(P_{i,j}^T)$ might not be defined for some j). Then to maximize the number of line items in the overlapping subsets (episodes), we let

$$V_i^{T,C} = \max_{f \in F} \sum_{S \in \text{dom}(f)} |S \cap f(S)|$$

NP	RF	Seq.	DiagCat	Daily
0.781	0.760	0.705	0.708	0.325

Table 6.2: One-to-one accuracy

Line item number	Date of svc.	Date of svc. number
1	11	$r_{i,1} = 1$
2	26	$r_{i,1} = 2$
3	26	$r_{i,1} = 2$
4	63	$r_{i,1} = 3$

Table 6.3: Example of date of service numbers

where $|\cdot|$ is the cardinality of the set. Note that $V_i^{T,C} = V_i^{C,T}$ because we can simply invert the one to one maps in F , resulting in the same overlapping subsets of line items.

To complete the calculation using this method, we determine the proportion of all line items in these overlapping subsets across all patients:

$$W^{T,C} = \frac{\sum_{i=1}^{50} V_i^{T,C}}{\sum_{i=1}^{50} n_i}$$

Then to compare among the Nurse Practitioner reviews, we average among the three pairs (since $W^{T,C} = W^{C,T}$ for any reviewers T and C).

$$A_{\text{NP}}^{\text{accuracy}} = \frac{W^{\text{NP1,NP2}} + W^{\text{NP1,NP3}} + W^{\text{NP2,NP3}}}{3}$$

and

$$A_{\text{M}}^{\text{accuracy}} = \frac{W^{\text{NP1,M}} + W^{\text{NP2,M}} + W^{\text{NP3,M}}}{3}$$

The results of this comparison are presented in Table 6.2.

6.5 Average pairwise agreement

In calculating this pairwise agreement, we averaged, for each patient, the pairwise agreement of episode assignments across a moving window of width k of dates of service. This is explained in detail below.

For a selected patient i and line item j , let R_i be the number of distinct dates of service, to which we assign date-of-service numbers in date order as $\{r_{i,1}, \dots, r_{i,R_i}\}$. For example, if a patient had four line items on the three dates of service shown in Table 6.3, then $R_i = 3$ and the date-of-service numbers would be as in the far right column.

Next, we choose $k \in \{1, 2, \dots\}$, and we identify the set $\mathcal{B}_{i,j,k}$ of line items other than j that are no more than k dates of service distant from the date-of-service of our selected line item j . $\mathcal{B}_{i,j,k}$ can be partitioned into subsets $B_{i,j,-k} \dots B_{i,j,k}$ such that for any $v \in \{-k, \dots, k\}$, $w \in B_{i,j,v} \Rightarrow (r_{i,w} - r_{i,j}) = v$. That is, we have partitioned $\mathcal{B}_{i,j,k}$ into subsets of line items with the same dates of service.

Next we average the agreement between a reviewer T and another reviewer C in terms of pairwise episode assignments between the selected line item and all of the line items on the date of service that is exactly v dates of service distant from the selected line item.

$$\beta_{i,j,v}^{T,C} = \frac{\sum_{w:w \in B_{i,j,v}} I\left(I\left(e_{i,j}^T = e_{i,w}^T\right) = I\left(e_{i,j}^C = e_{i,w}^C\right)\right)}{|B_{i,j,v}|} \quad (6.1)$$

where $I()$ is the indicator function.

This calculation of pairwise agreement is illustrated in Figure 6.1. In this figure, we have selected line item $j = 8$ for patient i . If we let $k = 2$, then the corresponding set $\mathcal{B}_{i,8,2}$ consists of the line items on days 10, 14, 15, 19, and 21, except for the selected line item j . The subset $B_{i,8,-2}$ consists of all line items on day 10 (that is, line items 2, 3, and 4). In calculating $\beta_{i,8,-2}^{T,C}$, we will sum over $w \in \{2, 3, 4\}$. When $w = 3$, we have $e_{i,8}^T = 4$ because reviewer T assigned line item 8 to episode 4. We also have $e_{i,3}^T = 2$, so the indicator function resolves to $I\left(e_{i,8}^T = e_{i,3}^T\right) = 0$. Performing the same calculation for reviewer C leads to $I\left(e_{i,8}^C = e_{i,3}^C\right) = 0$, so we conclude that $I\left(I\left(e_{i,8}^T = e_{i,3}^T\right) = I\left(e_{i,8}^C = e_{i,3}^C\right)\right) = 1$. When we sum these results for $w \in \{2, 3, 4\}$, we conclude that $\beta_{i,8,-2}^{T,C} = \frac{2}{3}$. Similarly, $\beta_{i,8,-1}^{T,C} = 1$, $\beta_{i,8,0}^{T,C} = \frac{1}{2}$, $\beta_{i,8,1}^{T,C} = 1$, and $\beta_{i,8,2}^{T,C} = \frac{1}{2}$.

Next we average this pairwise agreement across all dates of service that are within k dates of service of the selected line item $X_{i,j}$

$$\alpha_{i,j,k}^{T,C} = \frac{1}{2k+1} \sum_{v=-k}^k \beta_{i,j,v}^{T,C} \quad (6.2)$$

Continuing the example from above, $\alpha_{i,8,2}^{T,C} = \frac{1}{5} \left(\frac{2}{3} + 1 + \frac{1}{2} + 1 + \frac{1}{2} \right) = \frac{11}{15}$

We then average this pairwise agreement across all line items j for all patients i in the sample.

$$\tau_k^{T,C} = \frac{1}{50} \sum_{i=1}^{50} \frac{1}{n_i} \sum_{j=1}^{n_i} \alpha_{i,j,k}^{T,C} \quad (6.3)$$

Finally, to calculate these pairwise agreement among Nurse Practitioners, and also between a selected model M and the Nurse Practitioners, we proceeded as we did for the one-to-one episode accuracy.

$$A_{NP}^{\text{k-pairs}} = \frac{\tau^{NP1,NP2} + \tau^{NP1,NP3} + \tau^{NP2,NP3}}{3}$$

and

k	NP	RF	Seq.	DiagCat	Daily
1	0.900	0.882	0.852	0.834	0.665
2	0.900	0.882	0.852	0.834	0.663
3	0.899	0.882	0.851	0.834	0.663

Table 6.4: Average pairwise agreement within window k

$$A_M^{k\text{-pairs}} = \frac{\tau^{\text{NP1,M}} + \tau^{\text{NP2,M}} + \tau^{\text{NP3,M}}}{3}$$

Using this pairwise agreement measurement for a range of values of k , we produced Table 6.4.

6.6 Differences in results across EM iterations

The second method, the generative statistical model, used a Monte Carlo EM algorithm to estimate parameters. As shown in Figure 6.2, the 50 patient annotations using this method seem to perform best by about 5 or 6 EM iterations, although the initial parameter estimates (at iteration 0) were almost as good. Although it is quite possible for this EM algorithm to reach a sub-optimal local maximum, it also is possible that our initial parameter estimates were quite good for this model, because they were based on transitions within CCS diagnosis categories using our database of Medicare claims for 250,000 patients.

In this second method, once we estimated parameters using the EM algorithm, we clustered line items into episodes for the 50 patients in our test set to approximately maximize the probability of the episode configuration, as described in Section 5.5. We did this separately for each patient, evaluating line items in time sequence and looking ahead some $k \in \{0, 1, \dots\}$ events to calculate the maximum configuration probability including these k events. Although we expected the test results to improve for k larger, we actually did not see much change (in fact a small decline) in the test results as we increased k from 0 to 3. This is shown in Figure 6.3.

Increasing values of k also corresponded to decreasing number of episodes created using this method, as shown in Figure 6.4.

6.7 Differences in episodes among reviewers

The nurse practitioner reviewers differed somewhat in the number of episodes that they created among the 50 patients in the test set, as shown in Table 6.5. In this table, the episode entropy measures the dispersion of line items among episodes according to a selected reviewer T : $-\sum_{j=1}^{E^T} q_j^T \log q_j^T$ where q_j^T is the proportion of line items (for all 50 patients) that are contained in episode j as assigned by reviewer T , and E^T is the number of episodes created by reviewer T .

Based on the one-to-one accuracy metric, reviewers differed in their assignments of line items to episodes primarily because of differences in how they dealt with patients

Reviewer	Number of episodes	Episode entropy
A	510	5.09
B	552	5.33
C	708	5.48

Table 6.5: Differences among reviewers in creating episodes

with more complex diagnosis combinations. As shown in Figure 6.5 and Figure 6.6, both reviewer and the Random Forests sequence model were more accurate for patients with lower entropy, where entropy was measured as the extent of dispersion of line items among the CCS diagnosis categories: $-\sum_{j=1}^{m_i} p_{i,j} \log p_{i,j}$, where m_i is the number of diagnosis categories for patient i and $p_{i,j}$ is the proportion of line items from patient i that are in CCS diagnosis category j .

The following are several examples of patients for whom the nurse practitioners substantially differed in assigning line items to episodes.

Example 1

For this patient, approximately half of the line items were for diagnoses for lower leg conditions, such as:

1. pain in joint, lower leg (ICD-9 = 71946)
2. osteoarthritis of lower leg (71516)
3. displacement of the big toe (Hallux Valgus, 7350)
4. effusion of joint, ankle and foot (71907)

The dates of service for these conditions were in general scattered throughout the two-year period. All three reviewers grouped $\{1, 2\}$ together, as well as 3,4, but only one of the reviewers created a single episode $\{1, 2, 3, 4\}$, while the other two reviewers created two episodes.

Example 2

For this patient, the reviewers diverged in assigning episodes for two conditions that comprised a substantial percentage of the patient’s line items. First, the patient had atherosclerosis and ischemic heart disease, which appeared in the data with line-item diagnoses such as “coronary atherosclerosis” (ICD-9 = 41401) and “intermediate coronary syndrome” (4111), as well as (on electrocardiogram procedure line items) several symptomatic ICD-9 codes “chest pain” (786), “malaise and fatigue” (78079), “shortness of breath” (78605) and “hyperventilation” (78601). This heart condition resulted in multiple inpatient hospitalizations over the two-year period, and one of the reviewers separated the latest (and most intensive) hospitalization from the other hospitalizations, while the

other two reviewers combined all of the hospitalizations and outpatient services for this heart condition into one episode.

For this patient, the reviewers also differed in how they created episodes from a sequence of line items with a diagnosis of urinary tract infection (5990). For this diagnosis, there were several service dates consisting of urinalysis and similar services, but they were clustered in time into two groups separated by approximately four months. Two of the reviewers grouped all of these into one episode, reasoning that in the later events, the medical providers might have been following up to make sure that the urinary tract infection had cleared up, while the third reviewer separated these line items into two episodes.

Example 3

This patient had two metabolic conditions: diabetes (ICD-9 = 25000 and 25070) and hyperlipidemia (2722). In addition, the patient had a cardiovascular condition (7852). To further complicate the sequence, the diabetes diagnoses were on line items for several different types of care: ophthalmology, podiatry, and laboratory tests. From this collection of line items, each of the reviewers developed a rather distinct episode of care configuration for this patient.

Reviewer A

Episode 1: diabetes - all procedures, plus some with diagnosis of hyperlipidemia that seemed (based on the procedure) most likely for diabetes

Episode 2: all other hyperlipidemia line items

Episode 3: cardiovascular

Reviewer B

Episode 1: diabetes - podiatry

Episode 2: diabetes - ophthalmology

Episode 3: diabetes - lab tests, cardiovascular, hyperlipidemia

Reviewer C

Episode 1: diabetes - all procedures

Episode 2: hyperlipidemia

Episode 3: cardiovascular

The episode configuration selected by reviewer A in this example illustrates a problem that the reviewers often encountered for such services as laboratory tests, as well as other types of services such as ambulance trips – the diagnosis sometimes did not appear to match the condition for which the service was performed. For example, a patient might receive a prostate-specific antigen (PSA) test with a line item diagnosis of hyperlipidemia.

Example 4

The Nurse Practitioner reviewers found it quite difficult to determine how to allocate line items that were part of routine office visit dates of service (such as annual physical exams) to episodes of care. On the one hand, perhaps these line items should be grouped into a large “routine exam” episode of care, or on the other hand, perhaps they should be broken into smaller components based on the distinct diseases being screened for. This patient example highlights some of these difficulties.

This patient received two annual physical exams along with related services on the same date of services, some of which were billed by different providers than the provider who billed for the routine exam. The first of these annual exams consisted of the following 9 components:

1. general medical exam
2. evaluation and management (E/M) for “disorders of soft tissues” (ICD-9 = 7299)
3. influenza vaccine
4. pneumonia vaccine
5. PSA test
6. other lab tests (blood/urine)
7. chest x-ray
8. electrocardiogram (ECG)
9. computed tomography (CT) scan, thorax

The following displays the episodes created from this collection by each of the three reviewers.

Reviewer A

Episode 1: exam, vaccines, PSA test, other lab tests, ECG

Episode 2: E/M service

Episode 3: chest x-ray, CT scan

Reviewer B

Episode 1: exam, PSA test, other lab tests, ECG, chest x-ray, CT scan

Episode 2: E/M service

Episode 3: vaccines

Reviewer C

Episode 1: exam, PSA test, other lab tests
Episode 2: E/M service
Episode 3: chest x-ray, CT scan
Episode 4: influenza vaccine
Episode 5: pneumonia vaccine
Episode 6: ECG

For this patient, our two episode clustering algorithms separated these services into more episodes than any of the nurse practitioner annotators. Of the 9 procedures listed earlier for this date of service, the Random Forests algorithm only clustered (1) the pneumonia and influenza vaccines, and (2) the CT and chest x-ray. The sequence model clustered these as well, plus that model clustered the PSA test, ECG, and lab tests into one episode.

This example highlights the difficulty of determining whether to group elements of services that take place on the date of a routine exam with services that occur on other dates. This patient received another CT scan 7 months after the one in the example above, for which reviewer B created a separate episode, but which reviewers A and C grouped with the episode consisting of a chest x-ray and CT scan (labeled “Episode 3” above).

Calculation of pairwise agreement

Date of service	Line item	Reviewer / episode number	
		T	C
6	1	1	1
10	2	1	1
	3	2	1
	4	4	1
14	5	3	2
	6	3	3
15	7	4	3
	8	4	4
	9	4	4
19	10	2	5
21	11	2	5
	12	4	6
23	13	6	6
	14	6	3

Line items 3 and 8 are in different episodes according to both T and C

Selected line item 8

Line items 8 and 12 are in the same episode according to T but not C

Figure 6.1: Example of how we calculated pairwise agreement for a selected line item

Test set results across EM iterations

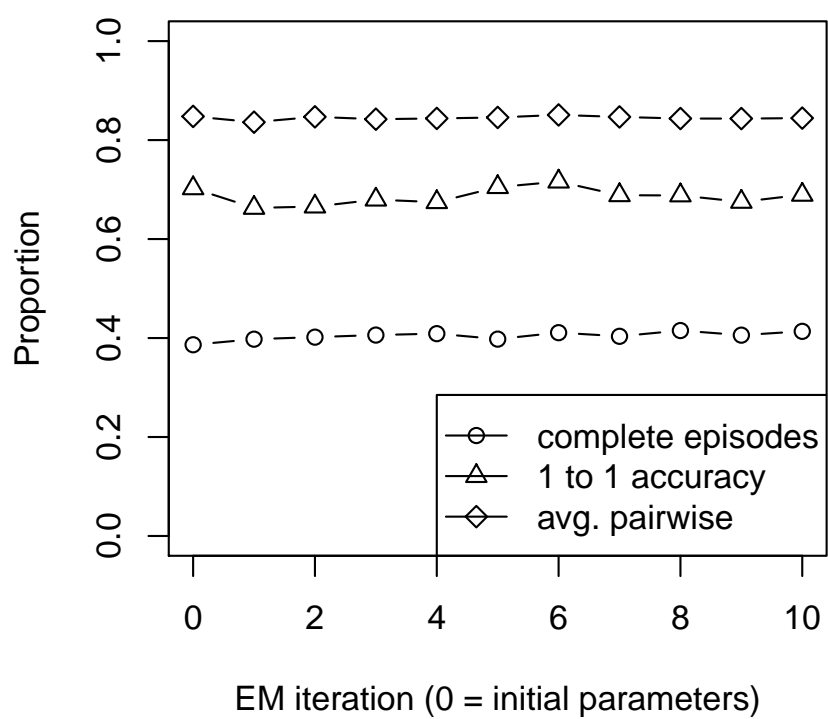


Figure 6.2: Test set results across EM iterations

Test set results, episode clustering look-ahead

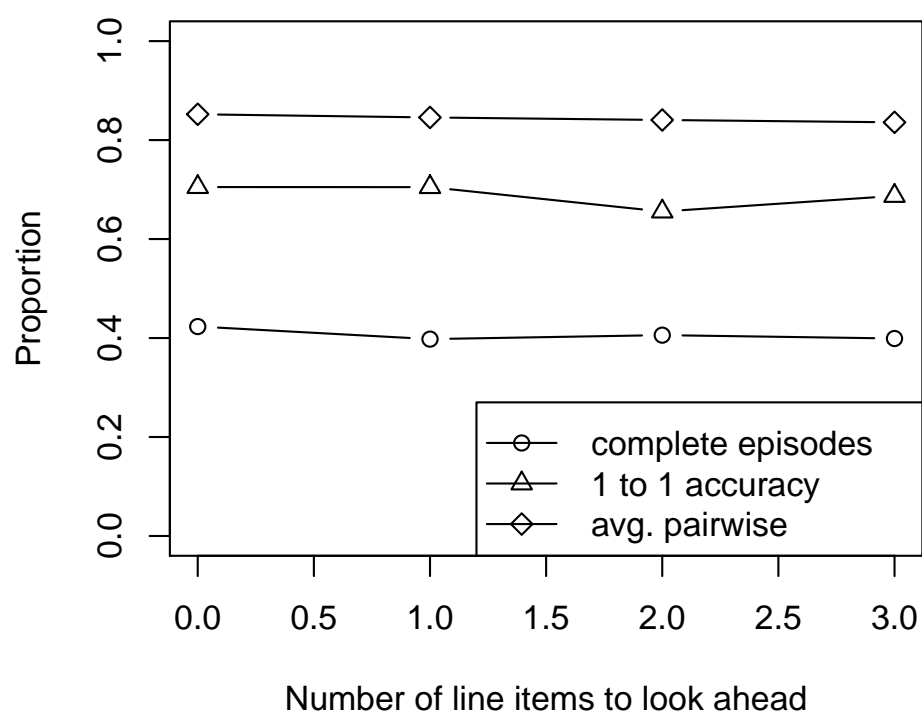


Figure 6.3: Test set results, EM clustering look-ahead

Test set results, look-ahead, number of episodes

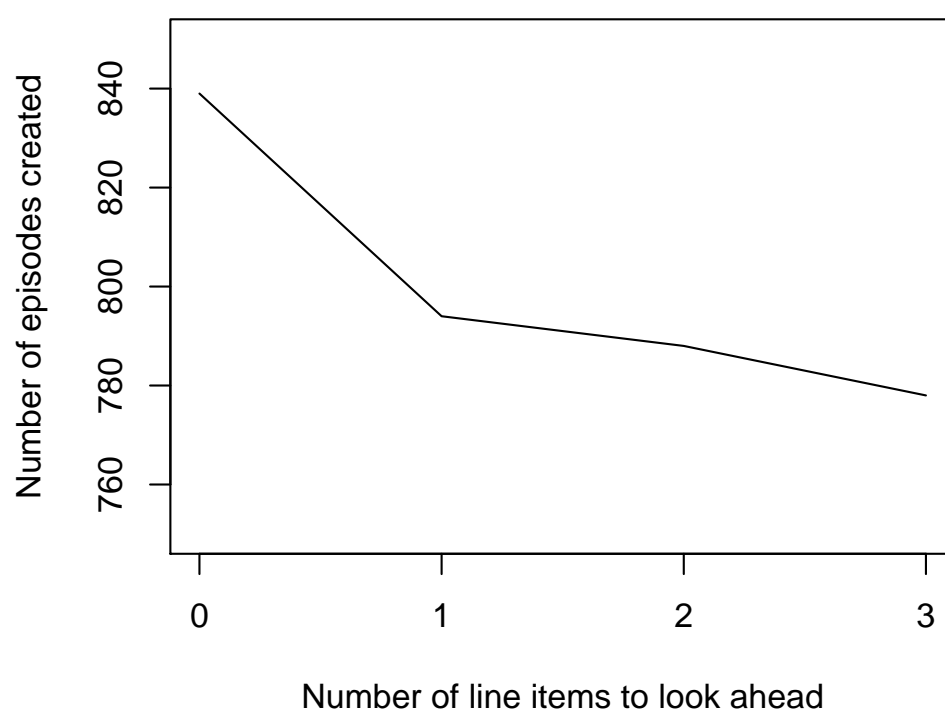


Figure 6.4: Test set results, look-ahead, number of episodes

NP one-to-one accuracy compared to entropy

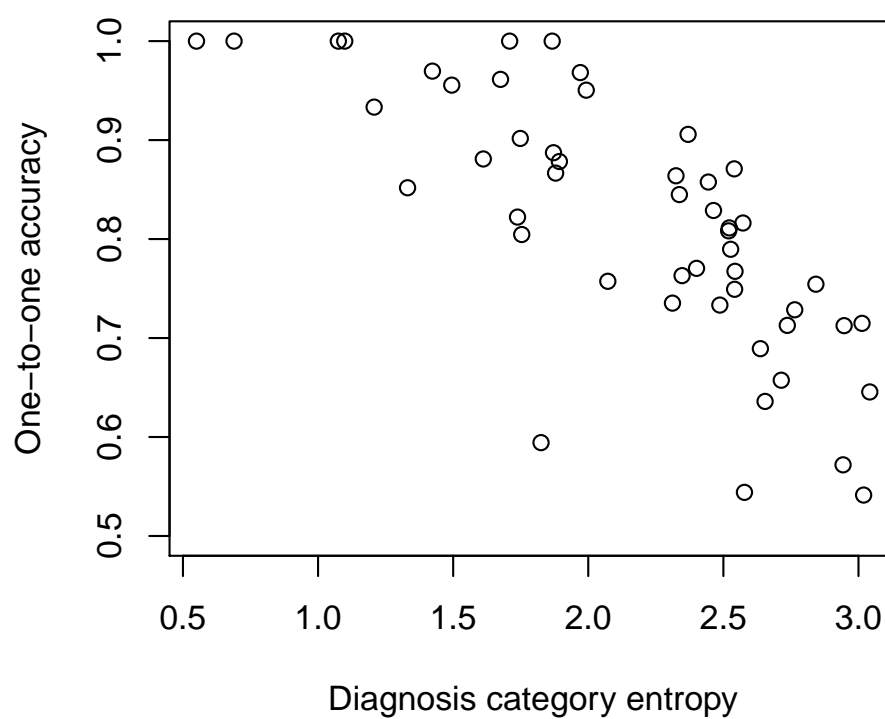


Figure 6.5: NP one-to-one accuracy compared to entropy

RF one-to-one accuracy compared to entropy

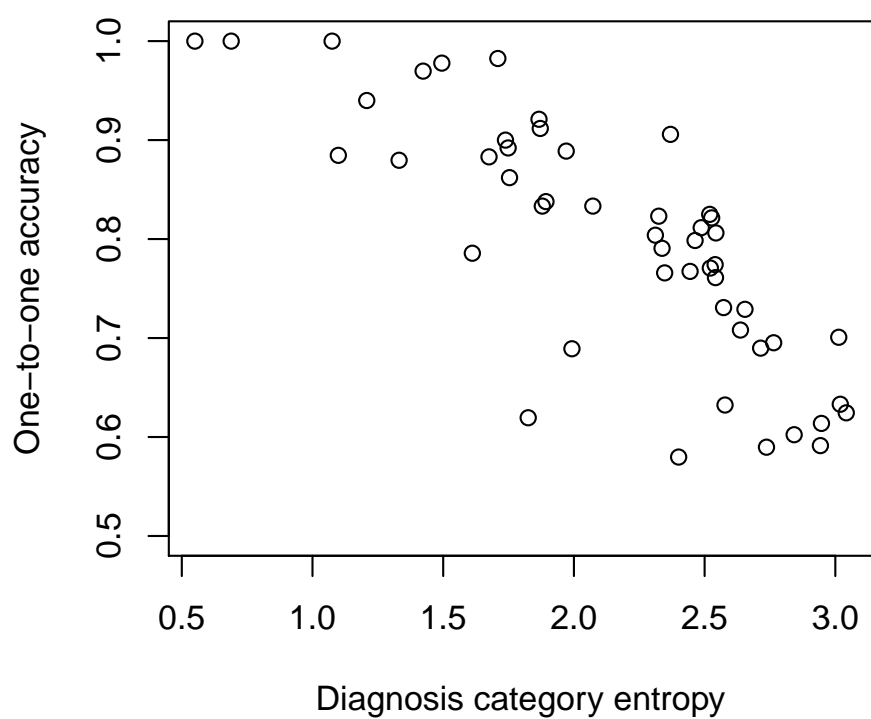


Figure 6.6: RF one-to-one accuracy compared to entropy

Chapter 7

Conclusion

In this research, we implemented two statistical methods for clustering Medicare claims into episodes of care. Both of these methods were improvements on simpler models, such as the one using only the AHRQ diagnosis categories to construct episodes, and one model – the one using Random Forests and the UMLS Metathesaurus – achieved results that were quite close to the nurse practitioner annotations.

All of the annotations in this research were less accurate for patients with more complex conditions, which reflected the difficulty of constructing episodes when the patients' multiple conditions were related clinically. Because the annotation process (both for the nurse practitioners and for the statistical methods) was limited to information in the Medicare claims data, there were many instances in which the annotation involved a close decision between two or more apparently reasonable episode configurations. For the reviews by the nurse practitioners, the decision process could have been greatly enhanced if medical records for these patients had been available, such as using a two-stage annotation process in which the second stage would have updated the annotation based on additional detail available from the medical records.

These statistical methods could have been substantially improved if we had access to additional claims data. Because of the cost of acquiring Medicare claims data for research, we were limited to a 5 percent sample of patients, and for the more detailed data that included dates of service (which revealed the sequence of care), we only had data for 250,000 patients with dates of service over only two years. Although this might seem like a relatively large set of data, we were using it to estimate statistical relationships among more than 14,000 diagnosis codes, and also between diagnosis codes and procedure codes. Because of the limited amount of data in the sample of 250,000 patients, we relied more heavily on the 5 percent sample, which contained claims data for approximately 1.9 million patients, but which did not allow us to determine the sequence of care within each calendar quarter.

These two statistical methods might also have been improved if we had been able to include claims data from all of the Medicare claims types. As it was, we only acquired Part B “Carrier” and Part A inpatient claims, but the other Medicare claims types, particularly outpatient hospital and Part D medications, could have clarified the sequence of care so that better annotations might have been possible, both by the nurse practitioners and by our statistical methods.

Our research made progress in developing methods for quantifying the strength of relationships among diagnosis codes and to a lesser extent between diagnosis codes and procedure codes, but much more could be done. We relied on two existing sources of information about these relationships, (1) the AHRQ diagnosis categories, and (2) the UMLS Metathesaurus, particularly the clinical relationships in SNOMED data. However, there are many other sources of data concerning these clinical relationships that could greatly improve these statistical methods for clustering claims into episodes.

Future research also might develop methods for resolving some of the difficult decisions that arose during the annotation process. For example, should each routine exam (such as an annual physical exam) be a single episode or should the services that accompany these exams (such as specific blood tests) be clustered with care on different dates of service for similar medical conditions? For chronic conditions with periodic crises (such as inpatient hospitalizations), should the entire sequence of care be clustered into one episode, or should it be divided into phases, perhaps even labeled according to the severity of illness or intensity of intervention? What would be the best representation of conditions that branch off (such as following inpatient hospitalizations) or in which the care apparently converges after being treated separately by different practitioners? Possibly a hierarchical clustering approach would be superior to the single-cluster approach that we took in this research, particularly for more complex patients.

These approaches – using a wider range of clinical information and allowing more flexible clustering – could greatly enhance the methods that we have presented and that have previously been implemented by other researchers.

Bibliography

- [1] Unified Medical Language System (UMLS). <https://uts.nlm.nih.gov/home.html>, 2010.
- [2] American Medical Association. Current Procedural Terminology 2007, 2006.
- [3] M.C.J. Biermans, D.H. de Bakker, R.A. Verheij, J.V. Gravestijn, M.W. van der Linden, and P.F. de Vries Robbé. Development of a case-based system for grouping diagnoses in general practice. *International Journal of Medical Informatics*, 77(7):431–439, 2008.
- [4] M.C.J. Biermans, G.H. Elbers, R.A. Verheij, W. Jan van der Veen, G.A. Zielhuis, and P.F. de Vries Robbé. External validation of EPICON: a grouping system for estimating morbidity rates using electronic medical records. *Journal of the American Medical Informatics Association*, 15(6):770, 2008.
- [5] D.J. Brailer and E.A. Kroch. Member risk adjustment for ambulatory episodes of care. *Health Care Management Science*, 2(3):125–136, 1999.
- [6] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [7] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.
- [8] C. Buck. 2007 HCPCS Level II, 2007.
- [9] D.G. Cave. Profiling physician practice patterns using diagnostic episode clusters. *Medical care*, 33(5):463, 1995.
- [10] W.W. Chapman, L.M. Christensen, M.M. Wagner, P.J. Haug, O. Ivanov, J.N. Dowling, and R.T. Olszewski. Classifying free-text triage chief complaints into syndromic categories with natural language processing. *Artificial Intelligence in Medicine*, 33(1):31–40, 2005.
- [11] C.L. Damberg, M.E. Sorbero, P.S. Hussey, S. Lovejoy, L. Hangsheng, and A. Mehrotra. Exploring Episode-based Approaches for Medicare Performance Measurement, Accountability and Payment. <http://aspe.hhs.gov/health/reports/09/mcperform/report.pdf>, 2009.
- [12] K. Davis. Paying for care episodes and care coordination. *New England Journal of Medicine*, 356(11):1166, 2007.

- [13] S. Doddi, A. Marathe, S.S. Ravi, and D.C. Torney. Discovery of association rules in medical data. *Informatics for Health and Social Care*, 26(1):25–33, 2001.
- [14] A. Elixhauser, C. Steiner, and L. Palmer. Clinical Classifications Software (CCS). <http://www.hcup-us-ahrq.gov/toolssoftware/ccs/ccs.jsp>, 2010.
- [15] M. Elsner and E. Charniak. You talking to me? a corpus and algorithm for conversation disentanglement. *Proceedings of ACL-08: HLT*, pages 834–842, 2008.
- [16] M. Elsner and E. Charniak. Disentangling chat. *Computational Linguistics*, 36(3):389–409, 2010.
- [17] D. Feng, E. Shaw, J. Kim, and E. Hovy. Learning to detect conversation focus of threaded discussions. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 208–215. Association for Computational Linguistics, 2006.
- [18] Centers for Medicare and Medicaid Services. Medicare Enrollment: National Trends 1966-2008. <http://www.cms.gov/MedicareEnRpts/Downloads/HISMI08.pdf>.
- [19] G. Grimmett and D. Stirzaker. *Probability and Random Processes, Third Edition*. Oxford University Press, 2001.
- [20] L. Guo, Y. Ma, B. Cukic, and H. Singh. Robust prediction of fault-proneness by random forests. 2004.
- [21] A. Hart and B. Ford. ICD-9-CM Professional for Physicians Volumes 1–2, 2007.
- [22] R.L. Houchens, S. McCracken, W. Marder, and R. Kelley. The use of an episode grouper for physician profiling in Medicare: A preliminary investigation. http://www.medpac.gov/documents/Jun09_episodegroupersstability.contractor.jp.pdf, 2009.
- [23] D. Jurafsky, J.H. Martin, and A. Kehler. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, 2009.
- [24] T. MaCurdy, J. Kerwin, J. Gibbs, E. Lin, C. Cotterman, M. O’Brien-Strain, and N. Theobald. Evaluating the functionality of the Symmetry ETG and Medstat MEG software in forming episodes of care using Medicare data. <http://www.cms.hhs.gov/Reports/downloads/MaCurdy.pdf>, 2009.
- [25] C.D. Manning, H. Schütze, and MITCogNet. *Foundations of statistical natural language processing*, volume 59. MIT Press, 1999.
- [26] G.J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. John Wiley Sons, 2008.

- [27] Medicare Payment Advisory Commission (MedPAC). Payment basics: Physician services payment system. http://www.medpac.gov/documents/MedPAC_Payment_Basics_10_Physician.pdf, 2010.
- [28] H. Pang, A. Lin, M. Holford, B.E. Enerson, B. Lu, M.P. Lawton, E. Floyd, and H. Zhao. Pathway analysis using random forests classification and regression. *Bioinformatics*, 22(16):2028, 2006.
- [29] H.H. Pham, D. Schrag, A.S. O’Malley, B. Wu, and P.B. Bach. Care patterns in Medicare and their implications for pay for performance. *New England Journal of Medicine*, 356(11):1130, 2007.
- [30] J.A. Solon, J.J. Feeney, S.H. Jones, R.D. Rigg, and C.G. Sheps. Delineating episodes of medical care. *American Journal of Public Health*, 57(3):401, 1967.
- [31] R.Y. Son, R.K. Taira, H. Kangarloo, and A.F. Cárdenas. Context-sensitive correlation of implicitly related data: an episode creation methodology. *Information Technology in Biomedicine, IEEE Transactions on*, 12(5):549–560, 2008.
- [32] R.K. Taira, V. Bashyam, and H. Kangarloo. A field theoretical approach to medical natural language processing. *Information Technology in Biomedicine, IEEE Transactions on*, 11(4):364–375, 2007.
- [33] G.D. Venolia and C. Neustaedter. Understanding sequence and reply relationships within email conversations: a mixed-model visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 361–368. ACM, 2003.
- [34] L. Wang and D.W. Oard. Context-based message expansion for disentanglement of interleaved text conversations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 200–208. Association for Computational Linguistics, 2009.
- [35] Y.C. Wang, M. Joshi, W. Cohen, and C. Rosé. Recovering implicit thread structure in newsgroup style conversations. In *Proceedings of the 2nd International Conference on Weblogs and Social Media (ICWSM II)*, 2008.
- [36] P. Xu and F. Jelinek. Random forests in language modeling. In *Proc. EMNLP*, 2004.

Appendix A

SAS and R programs for first clustering method

To implement our two statistical methods, we used SAS software for Windows version 9.2 and R version 2.12.0. The following are the most important of the SAS and R programs that we used.

A.1 SAS code for extracting line item pairs for same patient and provider

From our 5 percent sample of Medicare claims with service dates in calendar years 2006 and 2007, we used the following SAS code to extract all pairs of line items for the same patient and provider, limited to physician, nurse practitioner, and physician assistant claims. We had previously partitioned the 5 percent sample data into 1,000 parts in such a way that all claims from a given patient were in the same part. In the SAS code below, we loop through these 1,000 parts to perform the analysis more efficiently.

```
%macro specmac;
  and spec in(
    '01','02','03','04','05','06','07','08','09','10',
    '11','12','13','14','16','18','20',
    '22','24','25','26','28','29','30',
    '33','34','36','37','38','39','40',
    '44','46','48','50','66',
    '72','76','77','78','79',
    '81','82','83','84','85','86','90',
    '91','92','93','94','97','98','99'
  )
%mend specmac;

%macro sameprov_ldiag_mac(dlev);
```

```

options nonotes nosource;
%do thous_num=0 %to &max_thous;
  %put Running SAMEPROV_LDIAG_MAC iteration &thous_num of &max_thous,
  DLEV = &dlev;
  data temp1(keep=desy_sort_key upin provid claim_no qtr diag&dlev sex);
  set g_out.line0607_partition_&thous_num(
    keep=spec ldiag clm_thru_dt desy_sort_key provid upin claim_no
    bene_sex_ident_cd
    rename=(bene_sex_ident_cd=sex));
  where desy_sort_key^=' ' and (provid^=' ' or upin^=' ') and sex in('1','2')
  %if &specopt=1 %then %do;
    %specmac
  %end;
  ;
  attrib qtr length=3;
  if substr(left(clm_thru_dt),1,4)='2006' then do;
    qtr=put(substr(left(clm_thru_dt),5,1),1.);
  end;
  else qtr=4 + put(substr(left(clm_thru_dt),5,1),1.);
  attrib diag&dlev length=$&dlev;
  diag&dlev=substr(ldiag,1,&dlev);
run;

proc sql;
  create table temp2 as
  select desy_sort_key, upin, provid, count(*) as num_diag&dlev
  from temp1
  group by desy_sort_key, upin, provid
  having calculated num_diag&dlev>1;
quit;

proc sql;
  create table temp3 as
  select a.*, b.num_diag&dlev
  from temp1 as a, temp2 as b
  where a.desy_sort_key=b.desy_sort_key and a.provid=b.provid
  and a.upin=b.upin;
quit;

proc sql;
  create table temp4 as
  select a.sex, a.diag&dlev as from_diag&dlev,
  b.diag&dlev as to_diag&dlev,
  (1/a.num_diag&dlev) as wt

```

```

from temp3 as a, temp3 as b
where a.desy_sort_key=b.desy_sort_key
      and a.provid=b.provid and a.upin=b.upin
      and (a.qtr=b.qtr or a.claim_no=b.claim_no
            or a.diag&dlev=b.diag&dlev);
quit;

proc sql;
  create table sameprovtrans_&thous_num as
  select from_diag&dlev, to_diag&dlev, sum(wt) as thous_wtsum,
         count(*) as thous_numtrans,
         sum(wt*(sex='1')) as thous_male_wtsum,
         sum(sex='1') as thous_male_numtrans,
         sum(wt*(sex='2')) as thous_female_wtsum,
         sum(sex='2') as thous_female_numtrans
  from temp4
  group by from_diag&dlev, to_diag&dlev;
quit;

proc datasets library=work nolist;
delete temp1 temp2 temp3 temp4;
quit;

%end;

options notes source;

data temp_sameprovtrans;
set
%do thous_num=0 %to &max_thous;
  sameprovtrans_&thous_num
%end;
;
run;

proc sql;
  create table cjunk.ldiag&dlev._sameprovtrans as
  select from_diag&dlev, to_diag&dlev, sum(thous_wtsum) as wtsum,
         sum(thous_numtrans) as numtrans,
         sum(thous_male_wtsum) as male_wtsum,
         sum(thous_male_numtrans) as male_numtrans,
         sum(thous_female_wtsum) as female_wtsum,
         sum(thous_female_numtrans) as female_numtrans
  from temp_sameprovtrans

```

```

group by from_diag&dlev, to_diag&dlev;
quit;

proc datasets library=work nolist;
delete temp_sameprovtrans
%do thous_num=0 %to &max_thous;
    sameprovtrans_&thous_num
%end;
;
quit;

%mend sameprov_ldiag_mac;

%let max_thous=999;
%let specopt=1;
%sameprov_ldiag_mac(5)

```

A.2 SAS code for improving estimated transition probabilities by multiplying transition matrix

Starting with a transition matrix developed from the line item pairs produced above (same patient, same provider), we used the following SAS code to repeatedly multiply this transition matrix by itself, which improved the transition probabilities as described in Section 4.6.

```

%let selrel=prov;
%let selgend=;
%let minprobmac=0.0001;

proc sql;
    create table cjunk.&selrel._&selgend.trans0 as
    select from_diag5, to_diag5, sum(&selgend.numtrans) as from_coun,
        &selgend.numtrans/sum(&selgend.numtrans) as probtrans
    from cjunk.ldiag5_same&selrel.trans
    where &selgend.numtrans>0
    group by from_diag5;
quit;

%macro beliefmac;

    options nonotes nosource;
    %put Running BELIEFMAC iter &seliter,

```

```

breaking input database into 1,000 parts;
%let iterminus=%eval(&seliter-1);

data
%do thous_num=0 %to 999;
  trans&iterminus._&thous_num(keep=from_diag5
  to_diag5 probtrans diag123)
%end;
;
set cjunk.&selrel._&selgend.trans&iterminus;
attrib diag3 length=$3;
diag3=substr(from_diag5,1,3);
if anydigit(substr(diag3,1,1))=0 then do;
  if substr(diag3,1,1)='E' then substr(diag3,1,1)='0';
  else if substr(diag3,1,1)='V' then substr(diag3,1,1)='1';
  else substr(diag3,1,1)='2';
end;
if anydigit(substr(diag3,2,1))=0 then substr(diag3,2,1)='0';
if anydigit(substr(diag3,3,1))=0 then substr(diag3,3,1)='0';
attrib diag123 length=3;
diag123=diag3;
if diag123=1 then output trans&iterminus._1;
%do thous_num=2 %to 999;
  else if diag123=&thous_num
  then output trans&iterminus._&thous_num;
%end;
else output trans&iterminus._0;

run;

%do thous_num=0 %to 999;
  %put Running BELIEFMAC iter &seliter on part &thous_num of 999;

  proc sql;
    create table temp1 as
  select a.from_diag5, b.to_diag5,
    sum(a.probtrans*b.probtrans) as temp_probtrans
  from trans&iterminus._&thous_num as a,
  cjunk.&selrel._&selgend.trans&iterminus as b
  where a.to_diag5=b.from_diag5
  group by a.from_diag5, b.to_diag5
  having calculated temp_probtrans>=&minprobmac;
quit;

```

```

proc sql;
    create table trans&seliter._&thous_num as
    select from_diag5, to_diag5,
    temp_probtrans/sum(temp_probtrans) as probtrans
    from temp1
    group by from_diag5;
quit;

proc datasets library=work nolist;
delete temp1;
quit;

%end;

options notes source;

data cjunk.&selrel._&selgend.trans&seliter;
set
%do thous_num=0 %to 999;
    trans&seliter._&thous_num
%end;
;
run;

proc datasets library=work nolist;
delete
%do thous_num=0 %to 999;
    trans&iterminus._&thous_num
%end;
;
quit;

%mend beliefmac;

%let seliter=1; *Select iteration number;
%beliefmac

%let seliter=2;
%beliefmac

%let seliter=3;
%beliefmac

%let seliter=4;

```



```
%beliefmac
```

A.3 SAS code for calculating SNOMED distances

Using the UMLS Metathesaurus as well as ICD-9 and CPT/HCPCS code data, we used the following SAS code to calculate “SNOMED distance” values as described in Section 4.4.

```
%macro cui_removmac;

  %*Delete CUIs that will create inappropriate links, such
  as Moved elsewhere;

  if substr(str,1,5)='Moved'
  or substr(str,1,11)='Non-current'
  or substr(str,1,9)='Duplicate'
  or str in(
    'Self perception, self concept pattern',
    'Linkage concept',
    'Bracken test of basic concept scale',
    'Extinct cross type concept',
    'Inactive concept',
    'Ambiguous concept',
    'Outdated concept',
    'Erroneous concept',
    'Reason not stated concept',
    'Navigational concept',
    'Special concept',
    'Namespace concept'
  )
  then delete;

%mend cui_removmac;

data temp_snomed_uniq(keep=cui str qual_str);
set gradstat.snomed(keep=cui str ts lat);
where lat='ENG';
if ts='P' then qual_str=1;
else qual_str=2;
run;

proc sort data=temp_snomed_uniq;
```

```

by cui qual_str;
run;

data snomed_uniq;
set temp_snomed_uniq;
by cui qual_str;
if first.cui;

%cui_removmac

run;

data diag_linecoun2;
set cjunk.diag_linecoun;
attrib ldiag3 length=$3 ldiag4 length=$4;
ldiag3=substr(ldiag,1,3);
ldiag4=substr(ldiag,1,4);
claims_ldiag_len=length(trim(ldiag));
run;

proc sql;
    create table icd9_2 as
    select distinct code, cui
    from gradstat.icd9;
quit;

data icd9_3;
set icd9_2;
where index(code,'-')=0;
attrib ldiag length=$5 ldiag4 length=$4 ldiag3 length=$3;
ldiag=compress(code,'.');
ldiag4=substr(ldiag,1,4);
ldiag3=substr(ldiag,1,3);
meta_ldiag_len=length(trim(ldiag));
run;

proc sql;
    create table diag_linecoun3 as
    select a.*, b.cui, b.meta_ldiag_len,
    b.ldiag as meta_ldiag, b.ldiag4 as meta_ldiag4,
    b.ldiag3 as meta_ldiag3,
    (a.ldiag=b.ldiag) as ldiag_matc,
    (a.ldiag4=b.ldiag4) as ldiag4_matc,
    (a.ldiag3=b.ldiag3) as ldiag3_matc

```

```

    from diag_linecoun2 as a, icd9_3 as b
    where a.ldiag=b.ldiag or a.ldiag4=b.ldiag4 or a.ldiag3=b.ldiag3;
quit;

data diag_linecoun4;
set diag_linecoun3;
if ldiag_matc=1 then qual_matc=1;
else if claims_ldiag_len-meta_ldiag_len=1 and
    ((claims_ldiag_len=5 and ldiag4_matc=1)
    or (claims_ldiag_len=4 and ldiag3_matc=1)) then qual_matc=2;
else qual_matc=3;

run;

proc sort data=diag_linecoun4;
by ldiag cui qual_matc;
run;

data diag_linecoun5;
set diag_linecoun4;
by ldiag cui qual_matc;
if first.cui;
run;

proc sort data=diag_linecoun5;
by ldiag qual_matc;
run;

data pretemp_diag_linecoun6(keep=ldiag qual_matc meta_ldiag cui);
set diag_linecoun5;
by ldiag qual_matc;
if first.ldiag then obscoun=0;
if qual_matc in(1,2) then do;
    obscoun+1;
    output;
end;
if qual_matc=3 and obscoun=0 then output;
run;

proc freq data=pretemp_diag_linecoun6;
tables qual_matc;
run;

*Only keep QUAL_MATC=1 or 2;

```

```
data temp_diag_linecoun6;
set pretemp_diag_linecoun6;
where qual_matc in(1,2);
run;
```

*For each LDIAG, only output CUIs from the best QUAL_MATC value
(although there may be more than one CUI);

```
proc sort data=temp_diag_linecoun6;
by ldiag qual_matc;
run;
```

```
data temp2_diag_linecoun6(drop=otflag);
set temp_diag_linecoun6;
by ldiag qual_matc;
if first.ldiag then otflag=1;
if otflag=1 then output;
if last.qual_matc then otflag=0;
```

```
retain otflag;
run;
```

*Extract CPT/HCPCS codes and corresponding CUIs;

```
data cpthcpcs;
set gradstat.cpt gradstat.hcpcs;
where length(trim(code))=5;
run;
```

```
proc sql;
  create table cpthcpcs2 as
  select distinct code, cui
  from cpthcpcs;
quit;
```

```
proc sql;
  create table cpthcpcs3 as
  select a.hcpcs, b.cui
  from cjunk.hcpcs_linecoun as a, cpthcpcs2 as b
  where a.hcpcs=b.code;
quit;
```

*Combine the diagnosis and CPT/HCPCS databases;

```

data diag_linecoun6;
set temp2_diag_linecoun6(in=ss1)
  cpthcpcs3(rename=(hcpcs=ldiag));
is_diag=ss1;
run;

```

*From the database of SNOMED relationships,
extract relationships
likely to be meaningful in linking ICD-9 codes;

```

data mrrel_snomed_sub;
set gradstat.mrrel_snomed(keep=cui1 cui2 rela);
where rela in(
  'associated_finding_of',
  'associated_morphology_of',
  'associated_procedure_of',
  'associated_with',
  'causative_agent_of',
  'cause_of',
  'definitional_manifestation_of',
  'direct_morphology_of',
  'due_to',
  'finding_site_of',
  'focus_of',
  'has_associated_finding',
  'has_associated_morphology',
  'has_associated_procedure',
  'has_causative_agent',
  'has_definitional_manifestation',
  'has_direct_morphology',
  'has_direct_procedure_site',
  'has_finding_site',
  'has_focus',
  'has_indirect_morphology',
  'has_indirect_procedure_site',
  'has_part',
  'has_procedure_morphology',
  'indirect_morphology_of',
  'inverse_isa',
  'inverse_may_be_a',
  'isa',
  'is_alternative_use',
  'may_be_a',

```

```

    'occurs_after',
    'part_of',
    'same_as'
);
run;

```

```

proc sql;
    create table mrrel_snomed_sub2 as
    select a.*
    from mrrel_snomed_sub as a, snomed_uniq as b
    where a.cui1=b.cui;
quit;

```

```

proc sql;
    create table mrrel_snomed_sub3 as
    select a.*
    from mrrel_snomed_sub2 as a, snomed_uniq as b
    where a.cui2=b.cui;
quit;

```

*Extract the distinct CUIs that appear in the ICD-9 section of the Metathesaurus, and find relationships with other CUIs as they are defined in the SNOMED portion of the metathesaurus;

```

proc sql;
    create table icd9_cui as
    select distinct cui as ldiag_cui
    from diag_linecoun6;
quit;

```

*STEP: Prepare for evaluating degrees of separation between pairs of diagnosis codes, by splintering databases;

```

%macro thouscuimac;

```

```

    data
    %do thous_num=0 %to 999;
        &otname._&thous_num
    %end;
    ;
    set &inpname;
    attrib cuipart length=3;
    cuipart=substr(&cuivar,6,3);

```

```

    if cuipart=0 then output &otname._0;
    %do thous_num=1 %to 999;
        else if cuipart=&thous_num then output &otname._&thous_num;
    %end;
    else output &otname._0;
    run;

%mend thouscuimac;

%let inpname=diag_linecoun6;
%let otname=diag_linecoun6;
%let cuivar=cui;
%thouscuimac

%let inpname=mrrel_snomed_sub3;
%let otname=mrrel_snomed_cui1;
%let cuivar=cui1;
%thouscuimac

%let inpname=mrrel_snomed_sub3;
%let otname=mrrel_snomed_cui2;
%let cuivar=cui2;
%thouscuimac

%let inpname=icd9_cui;
%let otname=icd9_cui;
%let cuivar=ldiag_cui;
%thouscuimac

*Add a SNOMED relationship link to the specified database,
and output with the specified name;

%macro prelinkmac;

    %do thous_num=0 %to 999;

        proc sql;
            create table temp1 as
            select a.ldiag_cui, b.cui2 as rel_cui
            from &inpdat._&thous_num as a,
            mrrel_snomed_cui1_&thous_num as b
            where a.&cuivar=b.cui1 and a.ldiag_cui^=cui2;
        quit;

```

```

proc sql;
  create table temp2 as
  select a.ldiag_cui, b.cui1 as rel_cui
  from &inpdatt._&thous_num as a,
  mrrel_snomed_cui2_&thous_num as b
  where a.&cuiivar=b.cui2 and a.ldiag_cui^=cui1;
quit;

data prelink_&thous_num;
set temp1 temp2;
run;

proc datasets library=work nolist;
delete temp1 temp2;
quit;

%end;

data prelink;
set
%do thous_num=0 %to 999;
  prelink_&thous_num
%end;
;
run;

proc sql;
  create table &otdat as
select distinct ldiag_cui, rel_cui
from prelink;
quit;

proc datasets library=work nolist;
delete prelink
%do thous_num=0 %to 999;
  prelink_&thous_num
%end;
;
quit;

%*Partition the resulting database into 1,000 parts;

%let inpname=&otdat;
%let otname=&otdat;

```



```

%let cuivar=rel_cui;
%thouscuimac

proc datasets library=work nolist;
delete &otdat;
quit;

%*Match the resulting database with the CUIs that
are associated with diagnosis codes;

%do thous_num=0 %to 999;
  proc sql;
    create table &otdat._reldiagcui_&thous_num as
    select a.ldiag_cui as ldiag_cui1, b.ldiag_cui as ldiag_cui2
    from &otdat._&thous_num as a, icd9_cui_&thous_num as b
    where a.rel_cui=b.ldiag_cui;
  quit;
%end;

%mend prelinkmac;

*For a selected number of distance levels,
build relationship chains;

%macro chainmac;

  options nonotes nosource;

  %put Running CHAINMAC macro, linking diagnoses with same CUI;

  %do thous_num=0 %to 999;

    proc sql;
    create table diagrela_&thous_num as
    select a.ldiag as ldiag1, a.is_diag as is_diag1,
      b.ldiag as ldiag2, b.is_diag as is_diag2, 0 as dist
    from diag_linecoun6_&thous_num as a,
    diag_linecoun6_&thous_num as b
    where a.cui=b.cui and a.ldiag^=b.ldiag;
  quit;

  %end;

  data diagrela;

```

```

set
%do thous_num=0 %to 999;
    diagrela_&thous_num
%end;
;
run;

%do linknum=1 %to &maxlink;

    %put Running CHAINMAC iteration &linknum of &maxlink;

    %if &linknum=1 %then %do;

        %let inpdat=icd9_cui;
        %let otdat=icd9path1;
        %let cuivar=ldiag_cui;
        %prelinkmac

    %end;

%else %do;

    %let linkminus=%eval(&linknum-1);

    %let inpdat=icd9path&linkminus;
    %let otdat=icd9path&linknum;
    %let cuivar=rel_cui;
    %prelinkmac

    proc datasets library=work nolist;
    delete
        %do thous_num=0 %to 999;
            icd9path&linkminus._&thous_num
        %end;
    ;
    quit;

%end;

%end; /*End of LINKNUM loop;

/*For each diagnosis CUI link level, find the smallest
distance;

```

```

%put For pairs of diagnosis CUIs,
finding smallest distances among link levels;

%do thous_num=0 %to 999;
  data temp_comblink;
set
%do linknum=1 %to &maxlink;
  icd9path&linknum._reldiagcui_&thous_num(in=ss&linknum)
%end;
  ;
  if ss1 then dist=1;
%if &maxlink>1 %then %do;
  %do linknum=2 %to &maxlink;
    else if ss&linknum then dist=&linknum;
  %end;
%end;
run;

proc sql;
  create table temp2_comblink as
  select ldiag_cui1, ldiag_cui2, min(dist) as dist
  from temp_comblink
  group by ldiag_cui1, ldiag_cui2;
quit;

proc sql;
  create table comblink&thous_num as
  select a.ldiag_cui1, a.dist, b.ldiag as ldiag2,
  b.is_diag as is_diag2
  from temp2_comblink as a, diag_linecoun6_&thous_num as b
  where a.ldiag_cui2=b.cui;
quit;

proc datasets library=work nolist;
delete temp_comblink temp2_comblink;
quit;

%end;

%put Assembling databases;

data comblink;
set

```

```

%do thous_num=0 %to 999;
    comblink&thous_num
%end;
;
run;

proc datasets library=work nolist;
delete
%do thous_num=0 %to 999;
    comblink&thous_num
%end;
;
quit;

%let inpname=comblink;
%let otname=seccomblink;
%let cuivar=ldiag_cui1;
%thouscuimac

%do thous_num=0 %to 999;

proc sql;
    create table sec2comblink_&thous_num as
    select a.ldiag2, a.is_diag2, a.dist, b.ldiag as ldiag1,
    b.is_diag as is_diag1
    from seccomblink_&thous_num as a, diag_linecoun6_&thous_num as b
    where a.ldiag_cui1=b.cui;
quit;

%end;

options notes source;

data sec2comblink;
set
%do thous_num=0 %to 999;
    sec2comblink_&thous_num
%end;
;
run;

data gradstat.diaglinks_maxlink&maxlink;
set sec2comblink diagrela;
run;

```

```

proc sort data=gradstat.diaglinks_maxlink&maxlink;
  by ldiag1 dist ldiag2;
run;

%mend chainmac;

%let maxlink=3;
%chainmac

```

A.4 SAS code for creating the independent and dependent variables on line item pairs

For pairs of line items, we used the following SAS code to create the 8 independent variables and the dependent variable (same CCS diagnosis category), in preparation for using the Random Forests algorithm in R.

```

*Select the subset of AHRQ single classes for training;

%let ahrq_trainmac=255<=ldiagkat1<=259 or ldiagkat1>=2616
or 255<=ldiagkat2<=259 or ldiagkat2>=2616;

*The EMAC macro is a utility macro that will create
format labels, dealing with the case where the number
is in scientific notation;

%macro emac(selvar);
  attrib labelnum length=$11;
  attrib label length=$11;
  eloc=index(left(&selvar),'E');
  if eloc>0 then do;
    label_length=length(left(&selvar));
  const_exp=.;
  const_exp=substr(left(&selvar),eloc+2,label_length-(eloc+1));
  numzero=const_exp-1;
  is_neg=(substr(left(&selvar),1,1)='-');
  labelnum=compress(substr(left(&selvar),(1+is_neg),eloc-1),'.');
  label='0.';
  if numzero>0 then do;
    do z=1 to numzero;
      label=trim(label)||'0';
    end;
  end;

```

```

end;
label=trim(label)||labelnum;
if is_neg=1 then label='- '||trim(label);
  end;
  else do;
    label=substr(left(&selvar),1,11);
  end;
%mend emacs;

*STEP: Create a SAS format that assigns a SNOMED distance
to diagnosis and procedure code pairs;

data
  links_diag(keep=ldiag1 ldiag2 dist)
  links_hcpcs(keep=ldiag1 ldiag2 dist)
  links_mixed(keep=ldiag1 ldiag2 dist)
;
set gradstat.diaglinks_maxlink&maxlink;
if is_diag1=is_diag2 then do;
  if ldiag1<ldiag2 then do;
    if is_diag1=1 then output links_diag;
  else output links_hcpcs;
  end;
end;
else if is_diag1=1 then output links_mixed;
run;

%macro linkfmtmac;

  proc sort data=links_&seltype nodupkey;
  by ldiag1 ldiag2;
  run;

  data temp1;
  set links_&seltype end=last;
  attrib codepair length=$11;
  codepair=
  trim(input(left(ldiag1),$5.))||", "||trim(input(left(ldiag2),$5.));
  fmtname="sno&seltype.fmt";
  type='c';
  attrib label length=$11;
  start=codepair;
  end=codepair;
  attrib label length=$11;

```

```

label=left(dist);
output;
if last then do;
    start=' ';
    end=' ';
    hlo='0';
    label=left(&maxlink+1);
    output;
end;
run;

proc format library=work cntlin=temp1;
run;

proc datasets library=work nolist;
delete temp1;
quit;

%Mend linkfmtmac;

%let seltype=diag;
%linkfmtmac

%let seltype=hcpcs;
%linkfmtmac

%let seltype=mixed;
%linkfmtmac

%macro diagpairmac(selrel,selprobvar,seliter);

proc sql;
    create table temp_pairwise0a as
    select from_diag5, sum(numtrans) as from_coun
    from cjunk.ldiag5_same&selrel.trans
    where numtrans>0
    group by from_diag5;
quit;

proc sql;
    create table pairwise0a as
    select a.from_diag5, a.to_diag5,
    a.probtrans as p1, max(b.from_coun,0) as from_coun
    from cjunk.prov_trans&seliter as a left join

```

```

        temp_pairwise0a as b
    on a.from_diag5=b.from_diag5;
quit;

proc sql;
    create table pairwise0_marginal as
    select to_diag5, sum(numtrans) as num_to_diag5
    from cjunk.ldiag5_same&selrel.trans
    where numtrans>0
    group by to_diag5;
quit;

proc sql;
    create table pairwise0_marginal2 as
    select *, num_to_diag5/sum(num_to_diag5) as p
    from pairwise0_marginal;
quit;

proc sql;
    create table pairwise1_&selrel as
    select a.*, a.from_diag5 as diagkat1, a.to_diag5 as diagkat2,
        b.p, b.num_to_diag5, a.p1/b.p as temp_avgprob,
        2*a.p1*log(a.p1/b.p) as temp_kldivterm
    from pairwise0a as a, pairwise0_marginal2 as b
    where a.to_diag5=b.to_diag5;
quit;

%*Use the average of the KL divergence terms;

data temp_pairwise2_&selrel;
set pairwise1_&selrel;
if diagkat1<=diagkat2 then output;
else do;
    diagkat1=to_diag5;
diagkat2=from_diag5;
output;
end;
run;

proc sql;
    create table temp2_pairwise2_&selrel as
    select diagkat1, diagkat2, count(*) as pair_numobs,
        mean(temp_&selprobvar) as pair_avg_&selprobvar
    from temp_pairwise2_&selrel

```



```

    group by diagkat1, diagkat2;
quit;

data temp3_pairwise2_&selrel;
set temp2_pairwise2_&selrel;
if pair_numobs=1 and diagkat1^=diagkat2
then &selprobvar=pair_avg_&selprobvar/2;
else &selprobvar=pair_avg_&selprobvar;
run;

proc sort data=temp3_pairwise2_&selrel;
by diagkat1 diagkat2;
run;

data pairwise2_&selrel;
set temp3_pairwise2_&selrel end=last;
attrib diagpair length=$11;
diagpair=
trim(input(left(diagkat1),$5.))||", "||trim(input(left(diagkat2),$5.));
fmtname="diag_&selrel._pairfmt";
type='c';
attrib label length=$11;
start=diagpair;
end=diagpair;
%emac(&selprobvar)
output;
if last then do;
    start=' ';
    end=' ';
    hlo='0';
    label='0';
    output;
end;
run;

proc format library=work cntlin=pairwise2_&selrel;
run;

%mend diagpairmac;

%diagpairmac(prov,kldivterm,2)

%macro forestvarmac;

```

```

tdtdiff=abs(tdt1-tdt2);
sameprov=(provider_id1=provider_id2);
refprov=(
  (provider_id1^=' ' and provider_id1=rfr_provider_id2)
  or (provider_id2^=' ' and provider_id2=rfr_provider_id1)
);
  attrib diagpair length=$11;
  if ldiag1<=ldiag2 then do;
    diagpair=
      trim(input(left(ldiag1),$5.))||", "||trim(input(left(ldiag2),$5.));
  end;
else do;
  diagpair=
    trim(input(left(ldiag2),$5.))||", "||trim(input(left(ldiag1),$5.));
end;
ldiag_link=input(put(diagpair,$diag_prov_pairfmt.),8.3);
if ldiag1=ldiag2 then ldiag_dist=0;
  else ldiag_dist=input(put(diagpair,$snodiagfmt.),3.0);

%*ANYOFFICE attempts to capture several procedures that can be associated
with a wide range of diagnoses;

anyoffice=("36400"<=hcpcs1<="36425" or "36400"<=hcpcs2<="36425"
  or "99201"<=hcpcs1<="99499" or "99201"<=hcpcs2<="99499");
attrib hcpcspair length=$11;
if hcpcs1<hcpcs2 then hcpcspair=
  trim(input(left(hcpcs1),$5.))||", "||trim(input(left(hcpcs2),$5.));
else hcpcspair=
  trim(input(left(hcpcs2),$5.))||", "||trim(input(left(hcpcs1),$5.));
  if hcpcs1^=hcpcs2
    then hcpcs_dist=input(put(hcpcspair,$snohcpcsfmt.),3.0);
else hcpcs_dist=0;
attrib mixedpair1 mixedpair2 length=$11;
  mixedpair1=
    trim(input(left(ldiag1),$5.))||", "||trim(input(left(hcpcs2),$5.));
  mixedpair2=
    trim(input(left(ldiag2),$5.))||", "||trim(input(left(hcpcs1),$5.));
mixed_dist=min(
  input(put(mixedpair1,$snomixedfmt.),3.0),
  input(put(mixedpair2,$snomixedfmt.),3.0)
);

%mend forestvarmac;

```

```

%macro forest_expmac;

    data _null_;
    set &inpnamemac;
    file "c:\junk\&otnamemac..txt";
    put @1 sameddiagkat
        @5 sameprov
        @10 refprov
        @15 ldiag_link
        @25 ldiag_dist
        @30 hcpcs_dist
        @35 mixed_dist
    @40 tdttdiff
    @50 anyoffice
    ;
    run;

%mend forest_expmac;

*The FORESTMAC macro will produce one of two sets
(depending on the value of argument ISTRAIN). If ISTRAIN=1,
then the set does not include diagnoses that match the
more heterogeneous AHRQ categories, such as for E- and V-codes);

%macro forestmac(istrain,selpart,wholebenemac);

    data temp1;
    set
    %if &istrain=1 %then %do;
        ccw.partition_carrier_randpairs_&selpart;
        if &ahrq_trainmac then delete;
    %end;
    %else %do;
        ccw.partition_carrier_pairs_&selpart;
    %end;
    run;

    proc sql noprint;
        select count(*) into :dkatmac0 - :dkatmac1
        from temp1
            group by sameddiagkat;
    quit;

    data forest(keep=bene_id ldiag1 ldiag2 ldiagkat1 ldiagkat2

```

```

    lineid1 lineid2 hcpcs1 hcpcs2
    sameprov refprov ldiag_link ldiag_dist
    hcpcs_dist mixed_dist sameddiagkat
    tdttdiff anyoffice);
set temp1;
if sameddiagkat=0 then do;
    targprob=&targsamp/(2*&dkatmac0);
end;
else targprob=&targsamp/(2*&dkatmac1);
%if &wholebenemac=0 %then %do;
    if ranuni(0)<=targprob;
%end;

%forestvarmac

run;

%let inpnamemac=forest;

%if &istrain=1 %then %do;
    %let otnamemac=forest_&selpart;
data cjunk.forest_&selpart;
set forest;
forestorder=_n_;
run;
%end;
%else %do;
    %let otnamemac=testforest_&selpart;
data cjunk.testforest_&selpart;
set forest;
forestorder=_n_;
run;
%end;
%forest_expmac

proc datasets library=work nolist;
delete temp1;
quit;

%mend forestmac;

%macro all_trainforestmac(minpartmac,maxpartmac);

    options nonotes nosource;

```

```

%do i=&minpartmac %to &maxpartmac;
%put Running ALL_TRAINFORESTMAC
    iteration &i of &minpartmac to &maxpartmac;
    %forestmac(1,&i,0)
%end;

options notes source;

%mend all_trainforestmac;

%all_trainforestmac(1,1000)

*For a selected test set created by running FORESTMAC on a specified
partition of the data, we can select a range of PARTITION_RANDORDER
(patient identifiers) to create a subset;

%macro forest_testsubmac(selpart,min_partition_randorder,
max_partition_randorder);

proc sql;
    create table temp1 as
select a.*, b.partition_randorder
from cjunk.testforest_&selpart as a left join
ccw.rand_partition as b
on a.bene_id=b.bene_id;
quit;

data temp2(drop=forestorder);
set temp1;
where &min_partition_randorder<=partition_randorder
<=&max_partition_randorder;
run;

data
cjunk.testforest_&selpart._&min_partition_randorder._&max_partition_randorder;
set temp2;
forestorder=_n_;
run;

proc sort data=
cjunk.testforest_&selpart._&min_partition_randorder._&max_partition_randorder;
by forestorder;
run;

```

```

%let inpnamemac
=cjunk.testforest_&selpart._&min_partition_randorder._&max_partition_randorder;
%let otnamemac
=testforest_&selpart._&min_partition_randorder._&max_partition_randorder;
%forest_expmac

%mend forest_testsubmac;

%forest_testsubmac(1,1,55)

```

A.5 R code for fitting Random Forests model to line item pair data

Using the Random Forests algorithm in R, we produced 1,000 fitted models based on randomly selected line item pairs, as described in Section 4.2.1. We used these fitted models to evaluate the test set of 50 randomly selected patients, as shown in the R code below.

```

testforest <- read.table("c:\\junk\\testforest_1_1_55.txt")

for (i in 1:1000){
  forest <- read.table(paste("c:\\junk\\forest_",
    as.character(i),".txt",sep=''))
  thisforest <- randomForest(as.factor(V1) ~ . ,
    data=forest,mtry=3,ntree=5)
  thispred <- predict(thisforest,testforest,type="prob")[,2]
  write(thispred,file=paste("c:\\junk\\testforestpred_",
    as.character(i),".txt",sep=''),ncolumns=1)
}

```

A.6 SAS code for clustering line items into episodes of care

Using the results from the Random Forests algorithm, we used the following SAS code to cluster line items into episodes of care for the 50 test set patients, using agglomerative clustering as described in Section 4.1.

```

*Having run the above database (the one created by FOREST_TESTSUBMAC)
through RANDFOREST.R, we can merge it back in to TESTDAT;

```

*Input arguments to the ALL_TESTFORESTMAC macro:

```
TESTDATMAC = input database (the SAS database that
created input to Random Forests)
MINPARTMAC = minimum and maximum iterations from R
MAXPARTMAC
FAKEWRDMAC = the word FAKE if Random Forests run on data
from RANDFOREST_TESTSET, otherwise blank
```

```
;
```

```
%macro all_testforestmac(testdatmac,minpartmac,maxpartmac,fakewrdmac);
```

```
options nonotes nosource;
```

```
%do i=&minpartmac %to &maxpartmac;
```

```
  %put Running ALL_TESTFORESTMAC iteration &i of &minpartmac to &maxpartmac;
```

```
  data testforestpred_&i;
  infile "c:\junk\&fakewrdmac.testforestpred_&i..txt";
  input prob_&i;
  forestorder=_n_;
  run;
```

```
proc sort data=testforestpred_&i;
by forestorder;
run;
```

```
%end;
```

```
options notes source;
```

```
data testforestpred;
merge &testdatmac
%do i=&minpartmac %to &maxpartmac;
  testforestpred_&i
%end;
;
by forestorder;
prob=mean(of prob_&minpartmac-prob_&maxpartmac);
probvote=(prob>0.5);
correctdiagkat=(probvote=sameddiagkat);
```

```
run;
```

```

data testforestpred2;
set testforestpred(drop=prob_&minpartmac-prob_&maxpartmac);
run;

proc sort data=testforestpred2;
by sameddiagkat;
run;

proc freq data=testforestpred2;
tables correctdiagkat;
by sameddiagkat;
title "Trained on patient parts &minpartmac through
&maxpartmac, tested on &testdatmac";
run;

title;
run;

%if &testdatmac=cjunk.testforest_1_1_55 %then %do;
  data ccw.testforestpred_part1;
  set testforestpred2;
  run;
%end;

%mend all_testforestmac;

*For the test set, run ALL_TESTFORESTMAC for patients 1 through 55
(this contains the randomly selected 50 test set patients,
plus 5 more that did not have claims in both years 2007 and
2008);

%all_testforestmac(cjunk.testforest_1_1_55,1,1000,)

*STEP: For an input database of a set of patients (all claims
for each patient), with edge weights for pairs,
we can cluster these patients using agglomerative clustering.;

%macro forestclustmac(inpforest,selpart,delmactdtthreshmac,tdtlinkmac);

options nonotes nosource;

proc sql;
  create table desylist as

```



```

select bene_id, count(*) as katcount, min(lineid1) as min_lineid,
max(lineid2) as max_lineid,
(calculated max_lineid-calculated min_lineid)+1 as totlines
from &inpforest
group by bene_id
order by bene_id;
quit;

data _null_;
set desylist end=last;
if last then call symput("clustbene_mac",left(_n_));
run;

%do thisbene=1 %to &clustbene_mac;

    %put Running FORESTCLUSTMAC on patient &thisbene of &clustbene_mac;

    data _null_;
    set desylist;
    by bene_id;
    if _n_=&thisbene then do;
        call symput("desynamemac",left(bene_id));
        call symput("min_lineid_mac",left(min_lineid));
        call symput("maxkat_mac",left(totlines));
    stop;
    end;
run;

data thisdesypairs;
set &inpforest;
where bene_id="&desynamemac";
ccs_noclass1=(&ahrq_trainmac);
if sameddiagkat=1 and ccs_noclass1=0
and tdttdiff<=&tdtlinkmac then final_link=1;
else if ldiag1=ldiag2 and tdttdiff<=&tdtlinkmac then final_link=1;
else final_link=prob;
rev_lineid1=(lineid1-&min_lineid_mac)+1;
rev_lineid2=(lineid2-&min_lineid_mac)+1;
randnum=ranuni(0);
run;

data thisdesypairs2;
set thisdesypairs;
fmtname='forestpairfmt';

```

```

type='i';
attrib label length=$11;
%emac(final_link)
pairkat=((rev_lineid1-1)*&maxkat_mac)+rev_lineid2;
start=pairkat;
end=pairkat;
output;
pairkat=((rev_lineid2-1)*&maxkat_mac)+rev_lineid1;
start=pairkat;
end=pairkat;
output;
run;

proc format cntlin=thisdesypairs2;
run;

data thisdesypairs3;
set thisdesypairs;
fmtname='tdtpairfmt';
type='i';
attrib label length=$11;
label=substr(left(tdttdiff),1,11);
pairkat=((rev_lineid1-1)*&maxkat_mac)+rev_lineid2;
start=pairkat;
end=pairkat;
output;
pairkat=((rev_lineid2-1)*&maxkat_mac)+rev_lineid1;
start=pairkat;
end=pairkat;
output;
run;

proc format cntlin=thisdesypairs3;
run;

proc sort data=thisdesypairs;
by descending final_link sameddiagkat ldiagkat1 tdttdiff randnum;
run;

data tempclust1(keep=bene_id clustcoun carr1-carr&maxkat_mac);
set thisdesypairs end=last;

%*New arrays:

```

CARR keeps track of the cluster number
for each line item (the index corresponds
to the LINEID variable)

ONEPREVARR and TWOPREVARR use two different
indices to keep track of line item IDs
corresponding to existing clusters (if any) ;

```
array carr{&maxkat_mac} carr1-carr&maxkat_mac;
array oneprevarr{&maxkat_mac} oneprev1-oneprev&maxkat_mac;
array twoprevarr{&maxkat_mac} twoprev1-twoprev&maxkat_mac;
```

```
if _n_=1 then do;
  carr{rev_lineid1}=1;
  carr{rev_lineid2}=1;
  clustcoun=1;
end;
else do;
```

```
  %*Determine whether either of these line items
  already is in a cluster;
```

```
  clustkat1=carr{rev_lineid1};
  clustkat2=carr{rev_lineid2};
```

```
  %*If not, and the link value for this pair is at
  least the specified threshold, then start new cluster;
```

```
  if clustkat1=. and clustkat2=. then do;
  if final_link>=&final_threshmac then do;
    clustcoun=clustcoun+1;
    carr{rev_lineid1}=clustcoun;
    carr{rev_lineid2}=clustcoun;
  end;
end;
```

```
  %*If so, and these line items are not already in the same cluster,
  then evaluate the possibility of linking these two line items;
```

```
  else if carr{rev_lineid1}^=carr{rev_lineid2} then do;
```

```
    oneprevcoun=0;
    if clustkat1^=. then do;
      do i=1 to &maxkat_mac;
```

```

        if carr{i}=clustkat1 then do;
            oneprevcoun=oneprevcoun+1;
            oneprevarr{oneprevcoun}=i;
        end;
    end;
end;
else do;
    oneprevcoun=1;
    oneprevarr{1}=rev_lineid1;
end;

    twoprevcoun=0;
    if clustkat2^=. then do;
        do i=1 to &maxkat_mac;
            if carr{i}=clustkat2 then do;
                twoprevcoun=twoprevcoun+1;
                twoprevarr{twoprevcoun}=i;
            end;
        end;
    end;
end;
else do;
    twoprevcoun=1;
    twoprevarr{1}=rev_lineid2;
end;

    prevpaircoun=0;
    prevpair_finlinksum=0;

    do i=1 to oneprevcoun;
        do j=1 to twoprevcoun;
            pairkat=((oneprevarr{i}-1)*&maxkat_mac)+twoprevarr{j};
            if input(pairkat,tdtpairfmt.)<=&tdtthreshmac
                or (oneprevarr{i}=rev_lineid1
                    and twoprevarr{j}=rev_lineid2) then do;
                prevpaircoun=prevpaircoun+1;
                prevpair_finlinksum=
                    prevpair_finlinksum+input(pairkat,forestpairfmt.);
            end;
        end;
    end;

    prevavg=prevpair_finlinksum/prevpaircoun;

    if prevavg>=&final_threshmac then do;

```

```

        thisclust=min(carr{rev_lineid1},carr{rev_lineid2});
do i=1 to &maxkat_mac;
    if i=rev_lineid1 or i=rev_lineid2 then carr{i}=thisclust;
if clustkat1^=. then do;
    if carr{i}=clustkat1 then carr{i}=thisclust;
end;
    if clustkat2^=. then do;
        if carr{i}=clustkat2 then carr{i}=thisclust;
    end;
end;
end;
end;

end;  /*End of ELSE do (to clustkat1=. and clustkat2=.);
end;  /*End of ELSE DO (to if _n_=1);

if last then output;
retain clustcoun carr1-carr&maxkat_mac;
run;

/*From this cluster-array database, for this patient, create
a database with each feature set and the corresponding cluster number;

data tempclust2(keep=bene_id rev_lineid lineid tempclustid);
set tempclust1;
attrib bene_id length=$15;
bene_id("&desynamemac");
array carr{&maxkat_mac} carr1-carr&maxkat_mac;
noclustid=clustcoun;
do i=1 to &maxkat_mac;
    rev_lineid=i;
lineid=(rev_lineid-1)+&min_lineid_mac;
    if carr{i}^=. then tempclustid=carr{i};
    else do;
        noclustid=noclustid+1;
        tempclustid=noclustid;
    end;
    output;
end;
run;

proc sort data=tempclust2;
by tempclustid;
run;

```

```

data patclust_&thisbene(drop=tempclustid);
set tempclust2;
by tempclustid;
if _n_=1 then clustid=0;
if first.tempclustid then clustid=clustid+1;
retain clustid;
run;

%if &delmac=1 %then %do;

    proc datasets library=work nolist;
    delete thisdesypairs thisdesypairs2
    thisdesypairs3 tempclust1 tempclust2;
    quit;

%end;

%end;  %*End of THISBENE loop;

options notes source;

data patclust_all;
set
%do thisdesy=1 %to &clustbene_mac;
    patclust_&thisdesy
%end;
;
run;

proc sort data=patclust_all;
by bene_id lineid;
run;

proc sort data=ccw.partition_carrier_&selpart;
by bene_id lineid;
run;

data forestpred_cluster;
merge patclust_all(in=ss1) ccw.partition_carrier_&selpart;
by bene_id lineid;
if ss1;
run;

%mend forestclustmac;

```

*For a range of patient IDs (MIN_PATIENTMAC and MAX_PATIENTMAC) and a given part (SELPART) and a probability threshold (FINAL_THRESHMAC), and a difference in dates threshold to limit clustering penalties for chronic conditions (TDTTHRESHMAC), and a difference in dates threshold (TDTLINKMAC) beyond which to remove the assumption of a perfect link for same diagnosis category (or same diagnosis for vague diagnosis categories), cluster using random forests results based on multiple Random Forests models (averaged) in groups of 50,000 pairs;

```
%macro mult_forestclustmac(min_patientmac,max_patientmac,
selpart,final_threshmac,tdtthreshmac,tdtlinkmac);
```

```
  data this_randorder(keep=bene_id);
  set ccw.rand_partition;
  where &min_patientmac<=partition_randorder<=&max_patientmac;
  run;
```

```
  proc sql;
    create table sel_testforestpred as
    select a.*
    from ccw.testforestpred_part&selpart as a, this_randorder as b
    where a.bene_id=b.bene_id;
  quit;
```

```
  %forestclustmac(sel_testforestpred,&selpart,1,&tdtthreshmac,&tdtlinkmac)
```

```
%mend mult_forestclustmac;
```

```
%mult_forestclustmac(1,55,1,0.8,-1,300)
```

Appendix B

SAS programs for second clustering method

B.1 SAS code for producing initial parameter estimates and updating with EM algorithm

We used the following SAS code to produce initial parameter estimates and update them using a Monte Carlo version of the EM algorithm. This corresponds to the methodology described in Section 5.4.

*Macro variable settings:

DGMAC = number of diagnosis categories (based on format created below)

MINPROBEXP = exp(-minprobexp)=minimum probability for any quantity (starting, stopping, transition)

;

%let dgmacc=319;

%let minprobexp=20;

*STEP: Create format assigning, for each diagnosis code, an integer-valued diagnosis category (requires having first run the preliminary step below);

*Produce formats for CCS categories;

%include 'c:\ccs\ccsimp.sas';

data final_diagfreq2;

set ccw.final_diagfreq;


```

ldiagkat=input(ldiag,$sng2dccc.);
if ldiagkat=. or 255<=ldiagkat<=259 or ldiagkat>=2616 then do;
  revldiagkat=0;
  if substr(ldiag,1,4) in('V586','V761','V723')
  or ldiag in('V7283','V7284')
  then do;
    isnot_fin_ldiagkat=2;
    if substr(ldiag,1,4) in('V586') then sec_ldiagkat=1;
    else if substr(ldiag,1,4) in('V761') then sec_ldiagkat=2;
    else if substr(ldiag,1,4) in('V723') then sec_ldiagkat=3;
    else if ldiag in('V7283','V7284') then sec_ldiagkat=4;
  end;
  else if ldiag_numpatients>=1000 then isnot_fin_ldiagkat=3;
  else isnot_fin_ldiagkat=4;
end;
else do;
  revldiagkat=ldiagkat;
  isnot_fin_ldiagkat=1;
end;

run;

proc sort data=final_diagfreq2;
by isnot_fin_ldiagkat revldiagkat sec_ldiagkat ldiag;
run;

data ccw.final_diagfreq3;
set final_diagfreq2;
by isnot_fin_ldiagkat revldiagkat sec_ldiagkat ldiag;
if _n_=1 then findiagkat=0;
if isnot_fin_ldiagkat=1 then do;
  if first.revldiagkat then findiagkat+1;
end;
else if isnot_fin_ldiagkat=2 then do;
  if first.sec_ldiagkat then findiagkat+1;
end;
else if isnot_fin_ldiagkat=3 then do;
  findiagkat+1;
end;
else if first.isnot_fin_ldiagkat then findiagkat+1;
run;

data final_diagfreq4;
set ccw.final_diagfreq3;

```

```

fmtname='findiagkatfmt';
type='i';
attrib label length=$11;
attrib start stop length=$5;
start=ldiag;
stop=ldiag;
label=substr(left(findiagkat),1,11);
run;

proc format cntlin=final_diagfreq4;
run;

*PRELIMINARY STEP: From the combination of training and test
data, determine all of the diagnostic codes that
we will need to account for;

proc sql;
  create table ccw.final_diagfreq as
  select ldiag, count(*) as ldiag_freq, count(distinct(bene_id))
  as ldiag_numpatients
  from ccw.bcarrier)
  group by ldiag;
quit;

*PRELIMINARY STEP: Using the training data, produce geometric
distance transition parameters and parameters
for starting and stopping probabilities;

%macro distancemac;

  options nonotes nosource;

  %do i=113 %to &max_thous;
    %put Running macro DISTANCEMAC
    iteration &i of 113 to &max_thous;

  proc sql;
    create table temp1 as
    select distinct fdt, partition_randorder, ldiag
    from ccw.partition_carrier_&i;
  quit;

  data temp2;
  set temp1;

```

```

finldiagkat=input(ldiag,findiagkatfmt.);
randnum=ranuni(0);
run;

proc sort data=temp2;
by partition_randorder fdt randnum;
run;

data temp3_trans(keep=prevldiagkat finldiagkat);
prevldiagkat=finldiagkat;
set temp2;
by partition_randorder fdt randnum;
if first.partition_randorder=0 then output;
run;

proc sql;
    create table transdiagkat_&i as
    select finldiagkat, prevldiagkat, count(*) as thous_numtrans
    from temp3_trans
    group by finldiagkat, prevldiagkat;
quit;

proc sort data=temp2;
by partition_randorder finldiagkat fdt;
run;

data temp3(keep=finldiagkat partition_randorder
    fdt prevfdt strtobs stpobs tdtdiff ldiagkat_id
    epilength);
prevfdt=fdt;
set temp2;
by partition_randorder finldiagkat fdt;
if first.finldiagkat then do;
    ldiagkat_id=0;
end;
strt_fdt=fdt;
ldiagkat_id+1;
format prevfdt strt_fdt date9.;
    strtobs=first.finldiagkat;
    stpobs=last.finldiagkat;
if first.finldiagkat then prevfdt=.;
    else tdtdiff=fdt-prevfdt;
    epilength=(fdt-strt_fdt)+1;
retain strt_fdt;

```

```

run;

proc sql;
  create table strtdiagkat_&i as
  select finldiagkat, count(*) as thous_numstrt
  from temp3
  where strtobs=1
  group by finldiagkat;
quit;

proc sql;
  create table stpddiagkat_&i as
  select finldiagkat, sum(ldiagkat_id) as thous_numtrans,
  sum(epilength) as thous_sum_epilength
  from temp3
  where stpobs=1
  group by finldiagkat;
quit;

proc sql;
  create table geomdiagkat_&i as
  select finldiagkat, sum(tdtdiff) as thous_sum_tdtdiff,
  count(*) as thous_numtrans
  from temp3
  where strtobs ^=1
  group by finldiagkat;
quit;

proc datasets library=work nolist;
  delete temp1 temp2 temp3 temp3_trans;
quit;

%end;

options notes source;

data temp_strtdiagkat;
  set
  %do i=113 %to &max_thous;
    strtdiagkat_&i
  %end;
;
run;

```

```

proc sql;
    create table ccw.strtdiagkat as
    select finldiagkat, sum(thous_numstrt) as numstrt
    from temp_strtdiagkat
    group by finldiagkat;
quit;

data temp_stpdiagkat;
set
%do i=113 %to &max_thous;
    stpdiagkat_&i
%end;
;
run;

proc sql;
    create table ccw.stpdiagkat as
    select finldiagkat, sum(thous_numtrans) as numtrans,
    sum(thous_sum_epilength) as sum_epilength,
    count(*) as numstp,
    calculated numstp/calculated numtrans as dayprobstp
    from temp_stpdiagkat
    group by finldiagkat;
quit;

data temp_geomdiagkat;
set
%do i=113 %to &max_thous;
    geomdiagkat_&i
%end;
;
run;

proc sql;
    create table ccw.geomdiagkat as
    select finldiagkat, sum(thous_sum_tdttdiff) as sum_tdttdiff,
    sum(thous_numtrans) as numtrans,
    calculated numtrans/calculated sum_tdttdiff as geomparam
    from temp_geomdiagkat
    group by finldiagkat;
quit;

data temp_transdiagkat;
set

```

```

%do i=113 %to &max_thous;
    transdiagkat_&i
%end;
;
run;

proc sql;
    create table temp2_transdiagkat as
    select finldiagkat, prevldiagkat,
    sum(thous_numtrans) as numtrans
    from temp_transdiagkat
    group by finldiagkat, prevldiagkat;
    quit;

proc sql;
    create table ccw.transdiagkat as
    select *, sum(numtrans) as numprev, count(*) as numtranskat
    from temp2_transdiagkat
    group by prevldiagkat;
    quit;

%mend distancemac;

%let max_thous=1117;
%distancemac

*STEP: Create formats for starting, stopping,
and transition probabilities;

*Produce parameters for the
daily probability of generating an episode
with a particular diagnosis category
and produce the corresponding format;

%macro genparam_mac(inpdat);

    data genparams;
    set &inpdat end=last;
    fmtname='epigenfmt';
    type='i';
    numpatients=(223136-22314)+1;
    numdays=730*numpatients;
    genp=numstrt/numdays;
    attrib label length=$11;

```

```

%emac(genp)
start=finldiagkat;
end=finldiagkat;
output;
if last then do;
    hlo='0';
    start=.;
    end=.;
    genp=exp(-&minprobexp);
    %emac(genp)
    output;
end;
run;

proc format cntlin=genparams;
run;

%mend genparam_mac;

*Produce stopping parameters;

%macro stpparams_mac(inpdat);

    data stpparams;
    set &inpdat end=last;
    fmtname='epistopfmt';
    type='i';
    attrib label length=$11;
    %emac(dayprobstp)
    start=finldiagkat;
    end=finldiagkat;
    output;
    if last then do;
        hlo='0';
        start=.;
        end=.;
        dayprobstp=exp(-&minprobexp);
        %emac(dayprobstp)
        output;
    end;
run;

proc format cntlin=stpparams;
run;

```

```

%mend stpparams_mac;

*Produce geometric time transition parameters;

%macro gendist_mac(inpdat);

    data gendist;
    set &inpdat end=last;
    fmtname='distfmt';

    type='i';
    attrib label length=$11;
    %emac(geomparam)
    start=finldiagkat;
    end=finldiagkat;
    output;
    if last then do;
        hlo='0';
        start=.;
        end=.;
        geomparam=exp(-&minprobexp);
        %emac(geomparam)
        output;
    end;
    run;

    proc format cntlin=gendist;
    run;

%mend gendist_mac;

*Produce transition matrix format based on KL divergence
statistics;

%macro transdiagkat_mac(inpdat);

    proc sql;
        create table transdiagkat_marg as
        select finldiagkat, sum(numtrans) as to_numtrans
        from &inpdat
        group by finldiagkat;
    quit;

```



```

proc sql;
  create table transdiagkat_marg2 as
  select *, to_numtrans/sum(to_numtrans) as marg_probtrans
  from transdiagkat_marg;
quit;

proc sql;
  create table transdiagkat2 as
  select a.*, b.marg_probtrans, b.to_numtrans,
         a.numtrans/a.numprev as condit_probtrans,
  calculated condit_probtrans*log(calculated condit_probtrans/
b.marg_probtrans) as kldivterm
  from &inpdatt as a, transdiagkat_marg2 as b
  where a.finldiagkat=b.finldiagkat;
quit;

proc sql;
  create table transdiagkat3 as
  select *, count(*) as kldiv_numtranskat,
  kldivterm/sum(kldivterm) as kldiv_probtrans
  from transdiagkat2
  where kldivterm>0
  group by prevldiagkat;
quit;

data transdiagkat4;
set transdiagkat3 end=last;
fmtname='diagtransfmt';
type='i';
transprob=kldiv_probtrans*(1-((&dgmac-kldiv_numtranskat)*
(exp(-&minprobexp))));
attrib label length=$11;
%emac(transprob)
pairkat=((prevldiagkat-1)*&dgmac)+finldiagkat;
start=pairkat;
end=pairkat;
output;
if last then do;
  hlo='0';
  start=.;
  end=.;
  transprob=exp(-&minprobexp);
  %emac(transprob)
  output;

```

```

end;
run;

proc format cntlin=transdiagkat4;
run;

%mend transdiagkat_macro;

*STEP: Initialize parameters;

%genparam_macro(ccw.strtdiagkat)
%stpparams_macro(ccw.stpdiagkat)
%gendist_macro(ccw.geomdiagkat)
%transdiagkat_macro(ccw.transdiagkat)

*STEP: Update parameter estimates based on EM algorithm;

*Given an input database with episode labels, the MLEMAC macro will
produce approximate maximum likelihood estimates of all parameters;

%macro mlemac(inpdat);

  %*New variables (sum across all patients):

  NUMSTRT = for the given diagnosis category, number of days on
            which new episode started with that diagnosis at start

  ;

  proc sql;
    create table mle_genstats1 as
  select finldiagkat, count(*) as numstrt
  from &inpdat
  where newepi=1
  group by finldiagkat;
  quit;

  %*Next we estimate the episode-stopping parameters;

  %*New variables:

  TOT_EVENTS = for each episode category, number of events in data
  TOT_STOPEVENTS = of TOT_EVENTS, number that stopped an episode
  DAYPROBSTP = estimated stopping probability for

```

```

    each episode category

;

proc sql;
    create table mle_stopstats2 as
select finldiagkat, count(*) as tot_events, sum(stopthis)
as tot_stopevents,
calculated tot_stopevents
/calculated tot_events as dayprobstp
from &inpdatt
group by finldiagkat
    having calculated tot_stopevents>0;
quit;

%*Next, we estimate the diagnosis
group-to-diagnosis transition parameters;

proc sql;
    create table mle_epistats1 as
select prevdiagkat as prevldiagkat,
finldiagkat, count(*) as numtrans
from &inpdatt
where newepi=0
group by prevdiagkat, finldiagkat;
quit;

proc sql;
    create table mle_epistats2 as
select *, sum(numtrans) as numprev
from mle_epistats1
group by prevldiagkat;
quit;

%*Next we estimate the distance-to-next event parameters;

proc sql;
    create table mle_diststats2 as
select prevdiagkat as finldiagkat,
count(*) as numgaps, sum(gap+1) as sumgap,
calculated numgaps/calculated sumgap as geomparam
from &inpdatt
where newepi=0
group by prevdiagkat;

```

```

quit;

%mend mlemac;

*The EMMAC macro will perform one iteration of the EM algorithm;

%macro emmac(thous_num);

  proc sql;
    create table temp_ranepisodes as
    select distinct partition_randorder, fdt, ldiag
    from ccw.partition_carrier_&thous_num;
  quit;

  data ranepisodes(drop=ldiag);
  set temp_ranepisodes;
  randorder=ranuni(0);
  finldiagkat=input(ldiag,findiagkatfmt.);
  run;

  proc sort data=ranepisodes;
  by partition_randorder fdt randorder;
  run;

  data em1(keep=partition_randorder finldiagkat episode_id fdt newepi
  epistop gap randorder);
  prevday=fdt;
  set ranepisodes;
  by partition_randorder;
  format prevday date9.;

  %*Selected variable definitions:

  PREVDAY = Day from previous observation
  (used when previous day for same patient)

  EPICOUN = keeps track of how many
  episodes are active for this patient

  EPISODE_ID = identifier for each episode

  MAX_EPI_ID = current maximum of this episode identifier

;

```

```

%*Temporary array definitions:

DEX = index of the array locations
      (diagnosis categories) of the currently
      active episodes. That is, if the Ith element of DEX
      equals K, then we look for the Kth element (for vectors)
      and the Kth row (the first element for two-dimensional arrays)
      to find the corresponding episode information in the arrays
      defined below.

REVDEX = for any given diagnosis, gives the DEX location of the
          episode (if any) with the selected
          diagnosis as its latest diagnosis

IDARR = for each active episode, the identifier associated with it
        (will be output as EPISODE_ID)

DATEARR = array of corresponding dates
          on which the episode started

PREVDATEARR = array of the latest
             event date for this episode

CURREPIPROBS = for an existing episode, probability that the episode
               continued to the current day with the current diagnosis,
               given that it had continued up to PREVDAY
               (the previous day on which there was any event for
               this patient) The last element of CURREPIPROBS will
               contain the probability of generating a new episode
               rather than continuing existing ones

;

array dex{&dgmac} _temporary_;
array revdex{&dgmac} _temporary_;
array idarr{&dgmac} _temporary_;
array datearr{&dgmac} _temporary_;
array prevdatearr{&dgmac} _temporary_;

if first.partition_randorder then do;

    epicoun=0;
    max_epi_id=0;

```

```

    %*For all episodes, blank out the information;

    do i=1 to &dgmac;
    dex{i}=.;
    revdex{i}=.;
        idarr{i}=.;
    datearr{i}=.;
    prevdatearr{i}=.;
    end;

    end; %*End of first.partition_randorder sub-routine;

    retain _all_;

    array currepiprobs{&dgplusmac};

    %*For each existing episode(if any),
    compute the CURREPIPROBS probabilities;

    if epicoun>0 then do;
        do dexi=1 to epicoun;
            i=dex{dexi};
            currepiprobs{i}=(
                (1-input(i,epistopfmt.))
                *pdf('GEOMETRIC',fdt-prevdatearr{i},input(i,distfmt.))
                *input(((i-1)*&dgmac)+finldiagkat,diagtransfmt.)
            )
        /(
            input(i,epistopfmt.)+((1-(input(i,epistopfmt.)))
                *(1-cdf('GEOMETRIC',(prevday-prevdatearr{i}),
                    input(i,distfmt.))))
        );
    end; %*End of DEXI loop;
    end; %*End of EPICOUN>0 sub-routine;

    %*The last element of the CURREPIPROBS array
    contains the probability of generating a new
    episode with the current event diagnosis;

    currepiprobs{&dgplusmac}=
    input(finldiagkat,epigenfmt.)/(1-input(finldiagkat,epigenfmt.));

    %*Next we select an episode to assign the

```

```

current event to, based on the posterior probability
of episode given current event;

postnorm=sum(of currepiprobs{*});
randpost=ranuni(0);
cumpost=0;
postdone=0;
seldexi=epicoun+1;
do dexi=1 to epicoun;
    i=dex{dexi};
    if postdone=0 then do;
        cumpost=cumpost+(currepiprobs{i}/postnorm);
        if randpost<=cumpost then do;
            seldexi=dexi;
        end;
    end;
end;
postdone=1;

%*The NEWEPI variable is output with this observation, and it
indicates whether this event started a new episode;

if seldexi=epicoun+1 then do;
    newepi=1;
if datearr{finldiagkat}=fdt then do;
    epicoun=epicoun+1;
    dex{epicoun}=finldiagkat;
    revdex{finldiagkat}=epicoun;
end;
max_epi_id=max_epi_id+1;
idarr{finldiagkat}=max_epi_id;
datearr{finldiagkat}=fdt;
prevdatearr{finldiagkat}=fdt;
end;

else do;
    newepi=0;

%*The SELI variable contains the selected episode
index (the index of the episode ID, date, probabilities, etc.);

seli=dex{seldexi};

%*If this event is not the start of a new episode,

```

```

    then the GAP variable will show how many days
    since the previous event in this same episode;

    gap=fdt-prevdatearr{seli};

    %*Now for this continued episode, we update the arrays;

if seli~=finldiagkat then do;
    prevdatearr{seli}=.;
    datearr{finldiagkat}=datearr{seli};
    datearr{seli}=.;
    idarr{finldiagkat}=idarr{seli};
    idarr{seli}=.;
    revdex{finldiagkat}=revdex{seli};
    revdex{seli}=.;
    dex{seldexi}=finldiagkat;
end;
prevdatearr{finldiagkat}=fdt;

end; %*End of ELSE DO corresponding to IF SELDEXI=0;

%*It is possible that we have had an existing episode
that ended with the current diagnosis, but the current
diagnosis either is starting a new episode or continued
a different episode. In this case, we need to remove
the DEX entry and compress the DEX array;

do dexi=1 to epicoun;
    if dexi~=revdex{finldiagkat} and dex{dexi}=finldiagkat then do;
        if dexi<epicoun then do;
            do i=dexi to (epicoun-1);
                dex{i}=dex{i+1};
            end;
        end;
        dex{epicoun}=.;
        epicoun=epicoun-1;
    end;
end;

%*If it turns out that this episode has a last event at this
point, we would like to determine randomly whether it stopped;

randstop=ranuni(0);

```



```

    contpast=1-cdf('GEOMETRIC',&daymac-fdt,
    input(finldiagkat,distfmt.));
    probstop=input(finldiagkat,epistopfmt.)
    /(((1-input(finldiagkat,epistopfmt.))*contpast)
    +input(finldiagkat,epistopfmt.));
    if randstop<=probstop then epistop=1;
    else epistop=0;

    %*Finally, we create variables that will be
    output with this event;

    episode_id=idarr{finldiagkat};

    run;

    proc sort data=em1;
    by partition_randorder episode_id descending fdt
    descending randorder;
    run;

    data em2(drop=epistop epi_lastday);
    set em1;
    by partition_randorder episode_id;
    if first.episode_id then do;
    stopthis=epistop;
    epi_finevent=1;
    if epistop=1 then epi_lastday=fdt;
    else epi_lastday=&daymac;
    end;
    else do;
    stopthis=0;
    epi_finevent=0;
    end;
    output em2;
    retain epi_lastday;
    run;

    proc sort data=em2;
    by partition_randorder episode_id fdt randorder;
    run;

    data em3_&thous_num;
    prevdiagkat=finldiagkat;
    set em2;

```

```

if newepi=1 then prevdiagkat=.;
run;

proc datasets library=work nolist;
delete em1 em2 temp_randepisodes randepisodes;
quit;

%mend emmac;

%macro all_emmac(prinitermac);

    %do i=113 %to &max_thous;

        %if &prinitermac=1 %then %do;
            %put EM iteration on part &i of 113 to &max_thous;
        %end;

    %emmac(&i)

    %end;

    data em3;
    set
    %do i=113 %to &max_thous;
        em3_&i
    %end;
    ;
    run;

    proc datasets library=work nolist;
    delete
    %do i=113 %to &max_thous;
        em3_&i
    %end;
    ;
    quit;

    %mlemac(em3)

    %genparam_mac(mle_genstats1)
    %stpparams_mac(mle_stopstats2)
    %gendist_mac(mle_diststats2)
    %transdiagkat_mac(mle_epistats2)

```

```

%mend all_emmac;

%macro em_finmle_mac(selfinvar);

    data ccw.fin_&selfinvar;
    set ccw.&selfinvar._&num_emiter;
    run;

%mend em_finmle_mac;

%macro em_itermle_mac(selfinvar);

    data ccw.&selfinvar._&emiter;
    set &selfinvar;
    run;

%mend em_itermle_mac;

%macro em_itermac(num_emiter);
    options nonotes nosource;
    %do emiter=1 %to &num_emiter;
        %put Running EM_ITERMAC iteration &emiter of &num_emiter;
        %if &num_emiter=1 %then %do;
            %all_emmac(1)
        %end;
    %else %do;
        %all_emmac(0)
    %end;
    %em_itermle_mac(mle_genstats1)
    %em_itermle_mac(mle_stopstats2)
    %em_itermle_mac(mle_diststats2)
    %em_itermle_mac(mle_epistats2)

    proc datasets library=work nolist;
    delete
        mle_genstats1
        mle_stopstats2
        mle_diststats2
    mle_epistats2
    ;
    quit;

%end;
options notes source;

```

```

%em_finmle_mac(mle_genstats1)
%em_finmle_mac(mle_stopstats2)
%em_finmle_mac(mle_diststats2)
%em_finmle_mac(mle_epistats2)

%Mend em_itermac;

%let dgplusmac=%eval(&dgmac+1);
%let daymac=730;
%let max_thous=1117;
%em_itermac(30)

```

B.2 SAS code for clustering line items into episodes of care using estimated parameters from EM algorithm

Using the estimated parameters produced by the EM algorithm, we used the following SAS code to cluster line items into episodes of care, as described in Section 5.5.

*STEP: For a selected patient, use these final parameters to cluster the patient events into episodes;

*The LABELMAC macro will assign episode labels to the input database based on an approximate maximum likelihood configuration, using the following arguments:

```

LDAT = selected input database

LOOKMAC = lookahead number of events, including
the current event

;

%macro look_calcmac;

%*Calculate continuation/stopping probabilities
for the existing episodes, for those that still
are the latest of their respective episodes;

do i=1 to epicoun;
  if lstarr{dex{i}}=1 then contprobarr{i}=log(
    input(dex{i},epistopfmt.)+((1-(input(dex{i},epistopfmt.)))
    *(1-cdf('GEOMETRIC',(fdt-prevdatearr{dex{i}})),

```

```

        input(dex{i},distfmt.))))
    );
else contprobarr{i}=.;
end;

%*Calculate generative, transition, and
continuation/stopping probabilities (as applicable)
for the look-ahead events;

do i=1 to &lookmac;
    if i<=numlook then do;
        if nxtprevdiagarr{i}=. then do;
            nxtprobarr{i}=log(input(nxtdiagarr{i},epigenfmt.)/
            (1-input(nxtdiagarr{i},epigenfmt.)));
        if i=1 then nxtprob1_nocont=nxtprobarr{i};
        end;
        else if input(((nxtprevdiagarr{i}-1)*&dgmact)
        +nxtdiagarr{i},diagtransfmt.)>0 then do;

            thisgeom=pdf('GEOMETRIC',nxtdayarr{i}
            -nxtprevdatearr{i},input(nxtprevdiagarr{i},distfmt.));
            if thisgeom>0 then do;
                nxtprobarr{i}=
                    log(pdf('GEOMETRIC',nxtdayarr{i}
                    -nxtprevdatearr{i},input(nxtprevdiagarr{i},distfmt.)))
                    +log(input(((nxtprevdiagarr{i}-1)
                    *&dgmact)+nxtdiagarr{i},diagtransfmt.)));
            if i=1 then nxtprob1_nocont=nxtprobarr{i};
            end;
        else nxtprobarr{i}=(-999999999999999999);

%*For those lookahead events that are the
latest in their episodes, add the log of the
continuation/stopping probabilities;

if nextlstart{i}=1 then nxtprobarr{i}=nxtprobarr{i}+log(
    input(nxtdiagarr{i},epistopfmt.)
    +((1-input(nxtdiagarr{i},epistopfmt.))
        *(1-cdf('GEOMETRIC',(fdt-nxtprevdatearr{i})
        ,input(nxtdiagarr{i},distfmt.)))))
);

end;
%*Otherwise, the transition probability

```

```

    is zero, so we set the log probability to
    an extremely small number;
    else nxtprobarr{i}=(-999999999999999999);
    end;
else nxtprobarr{i}=.;
end;

logprob=prevlogprob+sum(of nxtprobarr{*},of contprobarr{*});

if best_logprob=. or logprob>best_logprob then do;
    best_logprob=logprob;
    best_prevlogprob=prevlogprob+nxtprob1_nocont;
do i=1 to numlook;
    best_nxtdexarr{i}=nxtdexarr{i};
end;
end;

%mend look_calcmac;

%macro labelmac(ldat,lookmac);

    %*Prepare and then label the data;

    data label_randepisodes(drop=ldiag);
    set &ldat(keep=partition_randorder fdt ldiag);
    randorder=ranuni(0);
    finldiagkat=input(ldiag,findiagkatfmt.);
    run;

    proc sort data=label_randepisodes nodupkey;
    by partition_randorder fdt finldiagkat;
    run;

    proc sort data=label_randepisodes;
    by partition_randorder descending fdt descending randorder;
    run;

    %*For each event, produce an array of the next LOOKMAC events,
    including the current event;

    data label_randepisodes2(drop=i);
    set label_randepisodes;
    by partition_randorder;

```

```

%*The NUMLOOK variable contains the actual number of
items in the future lookup queue;

array nxtdayarr{&lookmac} nxtday1-nxtday&lookmac;
array nxtdiagarr{&lookmac} nxtdiag1-nxtdiag&lookmac;

if first.partition_randorder then do;
  do i=1 to &lookmac;
    nxtdayarr{i}=.;
    nxtdiagarr{i}=.;
  numlook=0;
end;
end;
if &lookmac>1 then do;
  do i=&lookmac-1 to 1 by -1;
    nxtdayarr{i+1}=nxtdayarr{i};
    nxtdiagarr{i+1}=nxtdiagarr{i};
  end;
end;

if numlook+1<=&lookmac then numlook=numlook+1;

nxtdayarr{1}=fdt;
nxtdiagarr{1}=finldiagkat;

retain numlook nxtday1-nxtday&lookmac nxtdiag1-nxtdiag&lookmac;

run;

proc sort data=label_randepisodes2;
by partition_randorder fdt randorder;
run;

data label_randepisodes3(keep=partition_randorder
finldiagkat episode_id fdt);
set label_randepisodes2;
by partition_randorder;

%*Selected variable definitions:

LOGPROB = log probability of configuration
          through this event plus look-ahead events
          for the selected configuration of the
          current plus lookahead events

```

```

PREVLOGPROB = log probability of configuration before this event

BEST_LOGPROB = among configurations of the current
plus lookahead events, best value of LOGPROG

EPICOUN = keeps track of how many episodes
are active for this patient

EPISODE_ID = identifier for each episode

MAX_EPI_ID = current maximum of this episode identifier

;

%*Temporary array definitions:

DEX = index of the array locations (diagnosis categories)
      of the currently active episodes. That is,
      if the Ith element of DEX equals K, then we
      look for the Kth element (for vectors) and the
      Kth row (the first element for two-dimensional arrays)
      to find the corresponding episode information
      in the arrays defined below.

REVDEX = for any given diagnosis, gives the DEX location of the
          episode (if any) with the selected
          diagnosis as its latest diagnosis

IDARR = for each active episode, the identifier associated with it
        (will be output as EPISODE_ID)

PREVDATEARR = array of the latest event date for this episode

LSTARR = including the current assignment of look-ahead events,
is this the last event of this particular episode

1 = yes
0 = no
blank = this diagnosis category is not part of a current episode

CONTPROBARR = log probabilities associated with continuing
(up to the current date) or stopping the existing events;

```



```

;

if _n_=1 then patientcoun=0;

array dex{&dgmac} _temporary_;
array revdex{&dgmac} _temporary_;
array idarr{&dgmac} _temporary_;
array prevdatearr{&dgmac} _temporary_;

if first.partition_randorder then do;

    patientcoun+1;
    put "Partitioning patient " patientcoun " ,
    LOOKMAC = &lookmac, EM iteration &this_emiter";

    %*For all episodes, blank out the information;

    do i=1 to &dgmac;
        dex{i}=.;
        revdex{i}=.;
        idarr{i}=.;
        prevdatearr{i}=.;
    end;

    %*The first event will be assigned to episode 1;

    dex{1}=finldiagkat;
    revdex{finldiagkat}=1;
    idarr{finldiagkat}=1;
    prevdatearr{finldiagkat}=fdt;
    max_epi_id=1;
    episode_id=1;
    epicoun=1;
    prevlogprob=log(input(finldiagkat,epigenfmt.)
    /(1-input(finldiagkat,epigenfmt.)));

    end; %*End of first.patientid sub-routine;

    retain _all_;

    array lstarr{&dgmac} lst1-lst&dgmac;
    array contprobarr{&dgmac} contprob1-contprob&dgmac;

    %*Initialize the above array to indicate that the events in all current

```

```

episodes are the latest for their respective episodes;

do i=1 to epicoun;
    lstarr{dex{i}}=1;
end;

%*The NXTLSTARR array indicates, for the look-ahead events,
the same as the LSTARR:

    1 = this event is the latest for the episode
    0 = not the latest

;

array nxlstarr{&lookmac} nxlst1-nxlst&lookmac;

%*These arrays also were defined in the previous data step;

array nxtdayarr{&lookmac} nxtday1-nxtday&lookmac;
array nxtdiagarr{&lookmac} nxtdiag1-nxtdiag&lookmac;

%*The NXTDEXARR array keeps track of the index values of
episode assignments for the look-ahead variables, and the
BEST_NXTDEXARR array keeps track of these assignments for
the best (maximum likelihood) episode assignment.
The TEMP_NXTDEXARR array is used for making a copy
of the NXTDEXARR array but blanking out a selected element
(to then compute a maximum);

array nxtdexarr{&lookmac} nxtdex1-nxtdex&lookmac;
array temp_nxtdexarr{&lookmac} temp_nxtdex1-temp_nxtdex&lookmac;
array best_nxtdexarr{&lookmac} best_nxtdex1-best_nxtdex&lookmac;

array nxtprevdiagarr{&lookmac} nxtprevdiag1-nxtprevdiag&lookmac;
array nxtprevdatearr{&lookmac} nxtprevdate1-nxtprevdate&lookmac;

%*The NXTPROBARR array keeps track of the log probability associated
with generative or transition/distance probabilities
for each look-ahead event;

array nxtprobarr{&lookmac} nxtprob1-nxtprob&lookmac;

%*Now we loop through all possible look-ahead configurations,
using the LOOK_CALCMAC macro to compute the log probability of each.

```

```

We start this loop with all look-ahead events set to the first
existing episode (or to 1 if there are not any existing episodes),
and we finish with each look-ahead event starting a new episode;

if first.partition_randorder=0 then do;

    do i=1 to numlook;
        nxdexarr{i}=1;
    best_nxdexarr{i}=1;
    if i<numlook then nxlstarr{i}=0;
    else nxlstarr{i}=1;
    if i>1 then do;
        nxtprevdiagarr{i}=nxtdiagarr{i-1};
nxdprevdatearr{i}=nxtdayarr{i-1};
    end;
    else do;
        nxtprevdiagarr{i}=dex{1};
nxdprevdatearr{i}=prevdatearr{dex{1}};
    end;
    end;

lstarr{dex{1}}=0;
%look_calcmac

    do until (nxdexarr{numlook}=epicoun+numlook);

%*Reset the LSTARR array to indicate that all of the events
carried over from previous observations are the latest
in their respective episodes;

do i=1 to epicoun;
    lstarr{dex{i}}=1;
end;

    %*Advance the last look-ahead event one position,
starting with continuations of existing episodes, then
generating a new episode. If the last look-ahead event
already generates a new episode, then advance other
look-ahead events that are not already generating new episodes;

lookiterdone=0;
    do i=numlook to 1 by -1;
if lookiterdone=0 then do;
    do j=1 to numlook;

```

```

        if i~=j then temp_nxtdexarr{j}=nxtdexarr{j};
else temp_nxtdexarr{j}=.;
end;
        if nxtdexarr{i}<min(epicoun+i,
            max(epicoun,of temp_nxtdexarr{*}))+1)
            then do;
                lookiterdone=1;
                nxtdexarr{i}=nxtdexarr{i}+1;
            end;
        else nxtdexarr{i}=1;
        end;
    end;

    %*Next, determine the previous diagnoses and dates for episodes that
    are continued;

    do i=1 to numlook;
        nxtprevdiagarr{i}=.;
    nxtprevdatearr{i}=.;
    nextlstarr{i}=1;
        if nxtdexarr{i}<=epicoun then do;
            nxtprevdiagarr{i}=dex{nxtdexarr{i}};
            nxtprevdatearr{i}=prevdatearr{dex{nxtdexarr{i}}};
            lstarr{dex{nxtdexarr{i}}}=0;
        end;
    if i>1 then do;
        do j=1 to (i-1);
            if nxtdexarr{i}=nxtdexarr{j} then do;
                nxtprevdiagarr{i}=nxtdiagarr{j};
                nxtprevdatearr{i}=nxtdayarr{j};
                nextlstarr{j}=0;
            end;
        end;
    end;
    end;

    %look_calcmac

end; %*End of NXTEPISODE_ID do until loop;

%*Finally, we output the approximate
maximum likelihood label for this event;

%*Assign the episode ID, and also increase EPICOUN and MAX_EPI_ID

```

```

    if a new episode was generated;

    if best_nxtdex1<=epicoun then do;
        episode_id=idarr{dex{best_nxtdex1}};
        revdex{dex{best_nxtdex1}}=.;
        prevdatearr{dex{best_nxtdex1}}=.;
        idarr{dex{best_nxtdex1}}=.;
    end;
    else do;
max_epi_id=max_epi_id+1;
episode_id=max_epi_id;
        epicoun=best_nxtdex1;
    end;

dex{best_nxtdex1}=finldiagkat;

%*It is possible that this event continues or begins one episode,
but there already is another episode with latest event the same
diagnosis as this event. In this case, we need to subtract one
from EPICOUN and compress the DEX array (that is, we terminate
the existing episode with the same diagnosis);

if revdex{finldiagkat}~=. then do;
    if revdex{finldiagkat}<epicoun then do;
        do i=revdex{finldiagkat} to (epicoun-1);
            dex{i}=dex{i+1};
        revdex{dex{i}}=revdex{dex{i}}-1;
    end;
    end;
    dex{epicoun}=.;
    epicoun=epicoun-1;
end;

%*Create new values for this episode;

revdex{finldiagkat}=best_nxtdex1;
prevdatearr{finldiagkat}=fdt;
idarr{finldiagkat}=episode_id;

%*Add the log probability of generating/transitioning to the
current event based on this episode label to the log
probability of events up to this one;

prevlogprob=best_prevlogprob;

```

```

end; %*End of FIRST.PATIENTID=0 sub-routine;

run;

%mend labelmac;

*The EM_SVMAC macro: From the database
that results from this EM algorithm clustering,
save permanent databases separately for each in a range of patient
ID numbers, with a specified EM iteration number (for the parameters
used) appended to the output name;

%macro em_svmac(minpatient_mac, maxpatient_mac, emitermac);

  %do thispt=&minpatient_mac %to &maxpatient_mac;

    data ccw.emclust_p&thispt._&emitermac._&lookmac;
    set label_randepisodes3;
    where partition_randorder=&thispt;
    run;

  %end;

%mend em_svmac;

*For the final test cases, label episodes using the EM parameters from
each of a range of iterations;

%macro emfintest_mac(min_emiter_mac,max_emiter_mac,
min_partition_randorder,max_partition_randorder,lookmac);

  data emfintest;
  set ccw.partition_carrier_1;
  where &min_partition_randorder
  <=partition_randorder<=&max_partition_randorder;
  run;

  options nonotes nosource;

  %if &min_emiter_mac=0 %then %do;

    %put Running EMFINTEST_MAC iteration 0 (using before-EM parameters);

```

```

%genparam_mac(ccw.strtdiagkat)
%stpparams_mac(ccw.stpdiagkat)
%gendist_mac(ccw.geomdiagkat)
%transdiagkat_mac(ccw.transdiagkat)

%let this_emiter=0;

%labelmac(emfintest,&lookmac)
%em_svmac(&min_partition_randorder,&max_partition_randorder,0)

%end;

%if &min_emiter_mac=0 %then %let beg_emiter_mac=1;
%else %let beg_emiter_mac=&min_emiter_mac;

%do this_emiter=&beg_emiter_mac %to &max_emiter_mac;

    %put Running EMFINTEST_MAC iteration
    &this_emiter of &min_emiter_mac to &max_emiter_mac;

    %genparam_mac(ccw.mle_genstats1_&this_emiter)
    %stpparams_mac(ccw.mle_stopstats2_&this_emiter)
    %gendist_mac(ccw.mle_diststats2_&this_emiter)
    %transdiagkat_mac(ccw.mle_epistats2_&this_emiter)

%labelmac(emfintest,&lookmac)
%em_svmac(&min_partition_randorder,
&max_partition_randorder,&this_emiter)

%end;

options notes source;

%mend emfintest_mac;

*Arguments required for EMFINTEST_MAC macro:

MIN_EMITER_MAC = min and max of the EM iterations
MAX_EMITER_MAC   (parameter update iterations) to use for clustering

MIN_PARTITION_RANDORDER = range of patient IDs
MAX_PARTITION_RANDORDER   to include in clustering database

LOOKMAC = see earlier definition

```

```
;  
  
%emfintest_mac(0,10,1,55,2)  
  
%emfintest_mac(5,5,1,55,1)  
%emfintest_mac(5,5,1,55,3)  
%emfintest_mac(5,5,1,55,4)
```