Predicting Days in Hospital using Health Insurance Claims

**RUBINA JABEEN**¹, **S. NUSRATH FATIMA**²

¹PG Scholar, Dept of CSE, Shadan Women’s College of Engineering and Technology, Hyderabad, TS, India.
²Assistant Professor, Dept of CSE, Shadan Women’s College of Engineering and Technology, Hyderabad, TS, India.

**Abstract:** Health insurance costs across the world have increased alarmingly in recent years. A major cause of this increase is payment errors made by the insurance companies while processing claims. These errors often result in extra administrative effort to re-process (or rework) the claim which accounts for up to 30% of the administrative staff in a typical health insurer. We describe a system that helps reduce these errors using machine learning techniques by predicting claims that will need to be reworked, generating explanations to help the auditors correct these claims, and experiment with feature selection, concept drift, and active learning to collect feedback from the auditors to improve over time. We describe our framework, problem formulation, evaluation metrics, and experimental results on claims data from a large US health insurer. We show that our system results in an order of magnitude better precision (hit rate) over existing approaches which is accurate enough to potentially result in over $15-25 million in savings for a typical insurer. We also describe interesting research problems in this domain as well as design choices made to make the system easily deployable across health insurance companies.

**Keywords:** Health Care, Hospitalizations, Australia, Big Data, Predictive Modeling, Health Insurance Claims.

**I. INTRODUCTION**

Health insurance costs across the world have increased alarmingly in recent years. These costs have been passed down to consumers and employer-sponsored health insurance premiums have increased 131 percent over the last decade. A large proportion of these increases have been due to increase in the administrative costs of insurance providers. According to a study by McKinsey and Company, $186 billion of over-spending on healthcare in the US is related to high administrative costs. The typical process for insured healthcare in the US is that a patient goes to a service provider (medical facility) for the necessary care and the provider files a claim with the patient’s health insurance company for the services provided. The insurance company then pays the service provider based on multiple complex factors including eligibility of the patient at time of service, coverage of the procedures in the benefits, and contract status with the provider etc. Payment errors made by insurance companies while processing claims often result in re-processing of the claim. This extra administrative work to re-process claims is known as rework and accounts for a significant portion of the administrative costs and service issues of health plans. These errors have a direct monetary impact in terms of the insurance company paying more or less than what it should have estimates from a large insurance plan covering 6 million members had $400 million in identified overpayments.

In our discussions with major insurance companies, we have found that these errors result in loss of revenue of up to $1 billion each year. In addition to the direct monetary impact, there is also an indirect monetary impact since employees need to be hired to rework the claim and answer service calls regarding them. According to estimates by an Accenture study, 33% of the administrative workforce is directly or indirectly related to rework processing. These statistics make the problem of rework prevention extremely important and valuable to the healthcare industry and motivated the work described in this paper. There are two industry practices currently prevalent for identifying payment errors: random quality control audits and hypothesis (rule) based queries. Random audits are not very effective at identifying this rework since the majority of claims are correct and most of the effort spent on audits is wasted. In our discussions and research, we found that somewhere between 2% and 5% of the claims audited are reworks, making 95% to 98% of the effort spent in random audits a waste. Hypothesis based querying involves domain experts identifying several hypothesis about how rework occurs and instantiates itself in claim data. They create rules which are then turned into SQL queries to find matching claims. Typical systems contain a few thousand rules that are run every day identifying thousands of suspect claims.

This results in slightly better precision (or hit rate) than random audits but still require a lot of manual effort in discovering, building, updating, executing and maintaining the hypotheses and rules. These rules are also insurance plan specific and do not generalize well across companies thus making the deployment very time consuming and dependent on domain experts. In this paper, we describe our system to help reduce these claims errors using machine learning techniques by predicting claims that will need to be reworked, and experiment with generating explanations to help the auditors correct these claims, feature selection, concept drift, and active learning to collect feedback from the auditors to
improve over time. This Rework Prevention Tool has been developed in conjunction with industry experts from Accenture’s Claims Administration group who currently work with most of the large insurance companies in the US. We have applied this system to two large US health insurance companies. In the rest of this paper, we describe our system framework, problem formulation, evaluation metrics, and experimental results on claims data from a large US health insurer. We show that our system produces an order of magnitude better precision (hit rate) over existing approaches which is accurate enough to potentially result in over $15-25 million in savings each year for a typical insurer. This in turn would have a large effect on the healthcare costs as well as help make the healthcare process smoother. We also describe interesting research problems in this domain as well as design choices made to make the system easily deployable across health insurance companies.

II. PROBLEM FORMULATION
We formulate the problem of Rework prediction as a classification problem and generate a ranked list of claims that need to be manually reviewed. We give the details of the problem formulation in the sections below.

Fig.1. Claim Processing Pipeline

A. Claim Processing Overview

We start by giving an overview of the claims processing workflow (Fig.1) and describe how rework prediction fits in this workflow. Claims are created by service providers and submitted to the insurance company. They go through automatic validation checks followed by pricing using benefit rules and contracts. This takes place automatically in some cases and in other cases, manual intervention is required. Once the pricing is applied, claims get finalized and payment is sent to the service provider. The system we describe in this paper is the box placed after the pricing is applied to detect potential issues with the claim before it’s finalized so it can be corrected before payment is sent.

B. Requirements

We worked with domain experts to come up with requirements that would make a rework prediction solution practical and useful for insurance companies. These requirements are described below and motivate our solution.

- **Prepayment Prediction:** We need to identify rework claims before payment is sent with the information that is available at the time of claims submission. This seems obvious to data mining experts but the industry norm today is that most of this analysis happens after payment is made. The problem for insurance companies then becomes how to recover the payment (or pay penalties if there was an underpayment). Insurance companies have recovery departments tasked with recovering payments that have been made incorrectly. Since the motivation of the system is to reduce healthcare costs, the goal is to pay the claim correctly the first time and without the extra effort later to correct it.

- **Generalization:** We need to identify a wide variety of rework, not be limited to manually identified rules, and should be easily deployable across companies. Designing rules for identifying errors in payment requires deep domain knowledge. These rules are also very specific to individual companies making the process of transferring the system to new companies difficult and expensive. Thus our framework needs to be easily deployable across clients with minimal extra effort.

- **Accuracy:** We should flag suspect claims with high accuracy to make it financially feasible for a human auditor to examine the claim and correct it. Since examining a claim can take considerable time (ranging from 20 minutes to over an hour), the accuracy of our system has to be high enough to justify this extra cost.

- **Explanations:** We should provide a means of fixing the claim errors quickly and economically. It should be able to communicate to the auditors the reasons for which the system is predicting the claim as Rework.

- **Adaptability:** The framework should adapt to changes in environment and the healthcare ecosystem due to changes in legislations, insurance plans, contracts, and pricing. We need to make sure our system is able to adapt accordingly. The system is also expected to improve its accuracy over time as it observes more and more data. Thus the system should actively solicit and use the feedback from the auditors to keep adapting to the changes and improving its performance.

III. SYSTEM OVERVIEW

Our system consists of the following components: Data Collection, Feature Construction, Model Learning and Selection, item Scoring, Explanation Generation, User Feedback, and User Interface.

A. Data Collection

As shown in the Fig.1 we capture data from the claims processing pipeline when it is priced and ready for finalization and payment. The data we operate on is the entire claims data warehouse which contains all the claims that have been submitted and processed in the past several years. We also need labels for this data to train our models. There are two labels we need to assign: rework or correct. The claims assigned the label rework are those that were manually examined in the past and found to be correct. The process of getting these labeled claims is fairly difficult in insurance companies. The labeled
Predicting Days in Hospital using Health Insurance Claims

data exists but it’s distributed across several systems and is collected through different business processes. The labels we typically come from three primary sources.

The first source is the Quality Control Audit system which contains all the claims that have been manually audited by auditors for quality control. The class distribution in this data is typically 2-5% rework and 95-98% correct claims which is the overall distribution of the entire population of claims. The second source is the Provider Dispute system. This source contains claims that are discovered by the service providers as erroneously paid and sent back to the insurance company for re-processing. Although this happens after claim is paid, we can use these as labeled claims. Most of the claims that come through this system are re-work (typically underpayments). The third source is the financial recovery process. This process is also initiated after the claim is paid to recover overpayments the insurance company makes to providers. Since the number of claims in the provider dispute and financial recovery systems is much larger than that in the quality control system, the entire training set contains more rework examples (~60-80%) than correct examples (~20-40%). This raises interesting research issues since the training distribution is very different from the distribution that would occur in real data after deployment. The real data distribution is typically the opposite -95% rework, 5% correct since that is the distribution of the entire population of claims.

B. Feature Construction

Once the data is collected and labeled, feature construction is the next step that takes place. There are four classes of information in each claim: Member information, Provider information, Claim Header, and Claim Line Details. Our features are extracted from these classes of information. Member and Provider information span the entire claim and provide information about the Member (patient), and the provider (hospital, doctor, or medical facility). The claim header gives information about the entire claim as well. Contract information, Amount billed, Diagnosis codes, Dates of service are some examples of data fields in the claim header. The last source is the Claim Line Details. This gives the details of each line in the claim which is used to itemize the claim for each procedure that was conducted on the patient. For each procedure, the claim line includes the amount billed, the procedure code (CPT), the counter for the procedure (quantity). Since we are currently focusing on predicting the likelihood of the entire claim as being rework, we aggregate claim line details to the overall claim level. For example, we create features using Min, Max, and Average and standard deviation functions for each numeric data field in each line. We also create more aggregate features that are specific to procedure codes. We derive some more features that are specific to the claims processing domain. For example, we calculate the time (in days) between the date of service and the date of submission of the claim. The intuition for this feature is to figure out if the claim is valid as there is a time limit within which the claims should be filed with the plan and also to see if there is a correlation between Rework and late/early submission of claims. Overall, we end up with ~15,000 categorical and numerical features to build our models.

C. Model Learning & Selection

We experimented with SVMs as the main learning algorithms. SVMs have been shown to be robust for large data mining tasks with large feature sets. Since SVMs are not able to handle categorical data we had to create binary features from our categorical features which led to feature explosion with nearly 110,000 features. We used SVM perf for SVMs. We also used SVMs from other packages Weka, KNIME but neither of them was able to handle our dataset and run experiments efficiently. In our experience, SVM perf is the most efficient package which can handle large datasets quickly. We also chose SVM perf because of the extremely fast training time. One of our goals was to create a system that requires minimal machine learning expertise to deploy. We wanted to automate the selection of classifier parameters and the ideal feature set size empirically using hold-out sets. This required us to run thousands of experiments varying the training data, classifier parameters, feature set sizes, and evaluation metrics to optimize. SVM perf, because of its fast training time, was ideal for our needs since it allowed us to automate the model selection process. Another aspect of model selection is dealing with concept drift and change in the target function over time. The obvious approach is to use all the training data available at any given time to train the models. If there is a lot of concept drift over time, more recent training data is more useful and retaining all history may end up hurting the overall performance. Our model selection experiments take this into account and empirically estimate the best subset of the data that is useful to predict rework in the near future. We give more details in the experiments section later.

1. Feature Selection

Since we have more than 110K features after creating binary features from the categorical data, we wanted to see if feature selection would be useful. We wanted to see if we can reduce the storage requirements for the features as well as explore the effect of reducing the feature size on SVM accuracy. Although it has been shown that SVMs are able to compensate for feature noise and can handle large feature sets, we wanted to confirm that for our problem and data. We experimented with Information Gain measure but the runtime for our data set was impractically high so we used a frequency-based feature selection technique. Since converting categorical features into binary features creates a sparse feature space, frequency-based feature selection was useful in making our system more efficient both in terms of execution time and storage requirements.

IV. RESULTS

A. Performance on Different Sub-Populations

Table I summarizes the performance of the various models when trained and tested on all the four populations. The results for training data and predicting data, along with results

International Journal of Innovative Technologies
Volume.04, Issue No.11, August-2016, Pages: 2011-2016
of the two baseline methods are presented. Fig. 2 shows scatter-plots of the regression results for the group born before the year 1948. The subplot on the left is for training and while the right subplot is for prediction.

**Fig. 2. Scatter-Plots for Bagged Regression Tree Results for Customers Born Before Year 1948 (Those Aged Over 63 Years or Older When the Model Was Trained in 2011)**

### B. Evaluation of Features

The models predictive capability was investigated using the feature subsets mentioned independently and compared the predictions to the results when the algorithm used all features. These results are displayed in Table II for algorithms using demographic, medical and past cost and DIH features separately. Since the miscellaneous features comprise a small group of heterogeneous features only, this was not analyzed. The top 200 features for the four sub-populations mentioned were also examined. Table III lists some interesting top features extracted from ICD-10 primary diagnosis code derived on the 1+ days group. Age was used as a reference feature for comparison. Fig. 3 shows how the top 200 features are distributed among the four feature subsets. Subplot Fig. 3(a) to 3(d) display the distribution of the top 200 features across the whole population, subjects born on or after 1948, subjects born before 1948, and 1+ days group. Fig. 3(e) shows the proportion of all features among the four subsets.

### TABLE I. Performance Metrics for the Proposed Method, Evaluated on Different Populations. Shown are the Results for Testing with the Training Data on the Model after Training (with the Same Data) and Results When Validating the Trained Model with the Prediction Dataset. Performances of Baseline Models 1 and 2 are Also Displayed.

### TABLE II. Performance Metrics for Predictions Using Various Feature Category Subsets Only. The Three Feature Subsets Tested are: Demographic Features, Medical Features and Prior Cost/DIH Features. Subsets of Miscellaneous Features were Not Used.

### TABLE III. An Example of Interesting ICD-10 Primary Diagnosis Features Age was Used as a Reference Feature with an Importance Measure of 0.702. The Definition of This Importance Measure was Described.

---

**International Journal of Innovative Technologies**  
*Volume.04, Issue No.11, August-2016, Pages: 2011-2016*
Predicting Days in Hospital using Health Insurance Claims

With most of the large insurance companies in the US. We have applied this system to two large US health insurance companies. We described our system framework, problem formulation, evaluation metrics, and experimental results on claims data from a large US health insurer. We show that our system produces an order of magnitude better precision (hit rate) over existing approaches which is accurate enough to potentially result in over $15-25 million in savings each year for a typical insurer. This in turn would have a large effect on the healthcare costs as well as help make the healthcare process smoother. We also described interesting research problems in this domain as well as design choices made to make the system easily deployable across health insurance companies.

V. CONCLUSION

In this paper, we described our system to help reduce claims processing errors using machine learning techniques by predicting claims that will need to be reworked, and experiment with generating explanations to help the auditors correct these claims, feature selection, concept drift, and active learning to collect feedback from the auditors to improve over time. This Rework Prevention Tool has been developed in conjunction with industry experts from Accenture’s Claims Administration group who currently work with most of the large insurance companies in the US.

VI. REFERENCES


Author’s Profile:
Rubina Jabeen is currently pursuing M. Tech with specialization in Computer Science Engineering at Shadan Women’s College of Engineering and Technology, Telangana, India.

S. Nusrath Fatima is an Assistant Professor in Shadan Women’s College of Engineering and Technology. She has 5 years of teaching experience. She holds a post graduate degree of M.Tech (CSE) from JNTU. Her areas of Interest Networking.