Paper 3224-2019

The Anatomy of Artificial Intelligence Solution for Automation of Underwriting Decision-Making Process

Tanya Kolosova, Samuel Berestizhevsky

InProfix Inc.

ABSTRACT

Experience and good judgment are essential attributes of underwriters. Systematic analysis of the underwriting decision-making process was expected to increase the efficiency in developing these attributes in underwriters. However, this approach fell short of expectations. The industry still struggles with the pace at which knowledge and experience are delivered to the next generation of underwriters. The solution may lie in the development and deployment of artificial intelligence (AI) methods and algorithms in the automation of the underwriting decision-making process. This paper outlines the current state of the performance measurement of underwriting decision-making process through performance metrics (including a novel one). Further, this paper provides an in-depth description of artificial intelligence methods and algorithms that can be successfully used to automate underwriting decision-making process. The data from one of the leading insurance companies was used for the analysis and testing of proposed approaches.

INTRODUCTION

Traditionally, underwriting decisions were considered the result of the judgment, experience, and "gut feelings." Apprenticeship was a customary way for a new or young underwriter to gain required experience. However, it took a long time for senior underwriters to pass their expertise to a younger generation, who would then do the same for the next generation.

One approach to improve underwriting training was to analyze in detail the steps in the underwriting decision-making process and creating guidelines. Thinking of underwriting as an application of the specific steps in the decision-making process clarified its nature but formed a rigid structure that lost the flexibility of experienced underwriters. Thus, the approach fell short of expectations. The industry still struggles with the pace of knowledge and experience delivery to next generation of underwriters.

Also, insurers began to learn that experience is not necessarily the best or most efficient teacher—it is time-consuming and expensive, and subjective. Bad habits, as well as good habits, could be passed on through the apprentice system.

The improvement of underwriting decision-making process can be made through the following three steps:

1. Identification of the best underwriters using various relevant performance metrics
2. Training AI algorithms utilizing the knowledge of the best performing underwriters, and
3. Deployment of AI solution for automation of the underwriting decision-making process.

REVIEW OF UNDERWRITERS PERFORMANCE

In this section, we analyze the performance of underwriters using several existing metrics, and a novel one proposed here. We briefly analyze the characteristics of these metrics and then apply them to the real-life data provided by a workers' compensation insurer.

METRICS OF UNDERWRITERS PERFORMANCE

Different metrics can be used to evaluate underwriters' performance, such as hit ratio, conversion rate, and time-to-deal. We also propose a novel metric that can measure underwriters' performance dynamically.

Hit Ratio

Hit ratio ($HitR$) metric presents an underwriter performance that depends on the number of assigned applications. This metric is calculated over some period, that is defined depending on the workload of underwriters, and an expected time to close the deal (bind an insurance application), or the expiration period for a quote.

(1)

$$HitR_{ti} = \frac{BU_{ti}}{AU_{ti}}$$

where:

$HitR_{ti}$ – hit ratio of the underwriter i during the period t

BU_{ti} – number of bound applications (deals closed) by the underwriter i during the time period t

AU_{ti} – number of applications assigned to the underwriter i during the period t

Very often, underwriters reject applications out of hand, or after short evaluation against the insurer's rules and policies. Such applications are considered of low quality. Hit Ratio metric uses the number of all applications assigned to the underwriter regardless of the applications of different quality. Thus, instead of the hit ratio, the conversion rate metric would be an adequate way of measuring underwriters' performance.

Conversion Rate

Conversion rate metric (CR_{ti}) presents an underwriter's performance based on only those applications that were quoted by the underwriter. In other words, quoted applications are those that were not rejected by the insurer. This metric is calculated over a period defined by expected time to close the deal or the expiration time for a quote.

(2)

$$CR_{ti} = \frac{BU_{ti}}{QAU_{ti}}$$

where:

CR_{ti} – conversion rate of the underwriter i during the period t

BU_{ti} – number of bound applications for the underwriter i during the period t

QAU_{ti} – number of applications quoted by the underwriter i during the period t

This metric is usually calculated over an extended period because underwriters would want to wait for the outcome of all applications, including those that are still in the process of negotiation with customers/prospects. Thus, this metric, though providing an informative

backward-looking picture, can't be used as an operational metric, for example, monthly. Another problem with this metric is that it is not taking into consideration the workload of the specific underwriter. As an example, the underwriter who bound 1 out of 5 quoted applications will have the same $CR_{ti} = \frac{1}{5} = 0.2$ as a person who bound 10 out of 50 quoted applications $CR_{ti} = \frac{10}{50} = 0.2$.

We propose a new operational metric that addresses these issues.

Dynamic Conversion Rate

We propose a new operational metric called Dynamic Conversion Rate. Dynamic Conversion Rate measures conversion continuously, allowing for the monitoring of underwriters' performance as well as managing optimal workload.

Dynamic conversion rate metric (DCR_{mi}) presents a monthly underwriter's performance based on information available in the applicable month. This metric considers an underwriter as a service provider that has incoming requests (quoted applications) and outgoing results (bound applications). In other words, each month an underwriter has several quoted applications that are in the queue to be finalized: to be bound or rejected by a customer/prospect. Because DCR_{mi} measures performance every month, underwriters with low workload are penalized for inactive months. Dynamic conversion rate metric is calculated according to the following formulae:

(3)

$$DCR_{mi} = \frac{ln(BU_{mi} + 1)}{ln(QQAU_{mi} + 1)}$$

where:

DCR_{mi} – dynamic conversion rate of the underwriter i during the month m

BU_{ti} – number of bound applications for the underwriter i during the month m

$QQAU_{ti}$ – number of applications quoted by the underwriter i and awaiting resolution (in the queue) during the month m

Values of this metric lay between 0 and 1; DCR_{mi} equals 0 when there are no bound applications in a current month, and equals 1 when all quoted until now applications were bound in the current month. Thus, this metric presents an ongoing performance of underwriters, providing an operational measure that can be used for efficient management of the underwriting process. As well, DCR_{mi} metric takes into consideration the workload of an underwriter. Table 1 demonstrates the benefits of DCR_{mi} metric using the same example as we used above for CR_{ti} conversion rate metric.

Underwriter	**Quoted Applications ($QQAU_{ti}$)**	**Bound Applications (BU_{ti})**	CR_{ti}	DCR_{mi}
A	5	1	0.2	0.38685
B	50	10	0.2	0.60987

Table 1. Comparison of CR_{ti} and DCR_{mi} Metrics

As shown in Table 1, DCR_{mi} is higher for an underwriter B who has the same conversion rate CR_{ti} = 0.2 as underwriter A, but a larger number of quoted and bound applications.

Time-to-Deal

Time-to-deal (TTD) metric presents an underwriter's efficiency in closing the deals. Time-to-deal is calculated as a time interval in days between the date when the insurance application was assigned to an underwriter, and the date when the application was bound. TTD metric is censored by a quote expiration date. TTD can be averaged over an extended period like 6 months or a year, but in this case, it is not operational and can't be used for the successful management of underwriting process. We propose to use monthly time-to-deal metric TTD_{mi}, which is an average time-to-deal calculated for the underwriter i during the month m.

(4)

$$TTD_{mi} = \frac{\sum_{j=1}^{n} TTD_{mji}}{n}$$

where:

TTD_{mi} – an average time-to-deal of the underwriter i during the month m

TTD_{mji} – a time-to-deal of the bound application j out of n applications bound by the underwriter i during the month m .

ANALYSIS OF UNDERWRITERS PERFORMANCE

Using described above metrics, we analyze the performance of underwriters in Workers' Compensation line of insurance. In our study, we use data related to new workers' compensation applications for the period from May 01, 2013 until December 31, 2016 (44 months).

Data Description

Data for workers' compensation line of insurance include:

- dates of applications submission, dates of quotes, and dates of policy binding if any;
- parameters of insurance applications like industry, state, and number of employees;
- underwriters to whom applications were assigned.

Before the analysis, the data was "cleansed" and prepared in the following way:

- Applications that required more than three months to be bound were removed from the study (less than 5% of all bound applications). Usually, quotes provided by this insurer expire after 90 days.
- Underwriters with too small of sample size were excluded.

Applications Flow

Underwriters performance should be evaluated conditionally on insurance applications flow to the insurer, and on the process of assignment applications to underwriters. Thus, before estimating the dynamic conversion rate and time-to-deal per underwriter, we analyze applications flow and the process of applications assignment.

Distribution of an average number of applications per month demonstrates seasonality with a somewhat increased number of applications in March and October, and with a significant drop in December (Figure 1). Identification of this seasonality in the submission process helps create a balanced workload for underwriters.

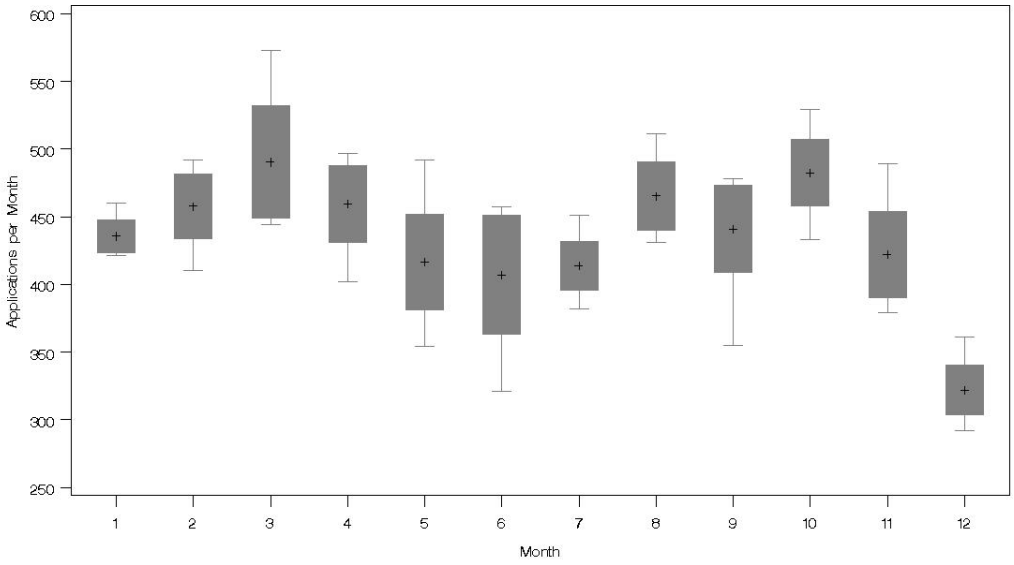

Figure 1. Average number of applications submitted monthly.

Distribution of an average number of applications per weekday demonstrates a slightly lower number of applications on Mondays (Figure 2).

The analysis indicates that the applications submission flow has some fluctuations among months and among weekdays which are easily manageable.

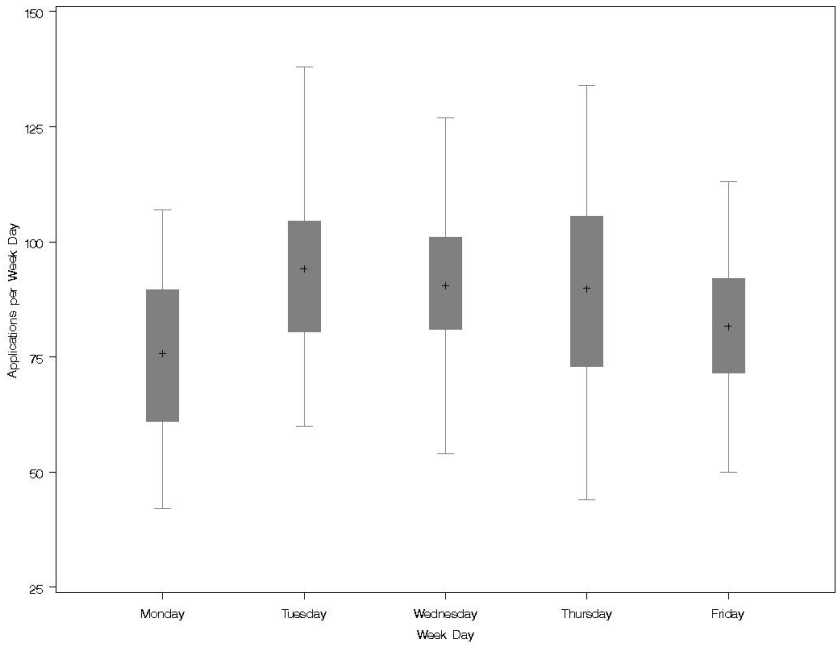

Figure 2. Average number of applications submitted per day of week

However, as we show in Figure 3, the workload of underwriters is far from being balanced. It makes an analysis of underwriters' performance more complicated.

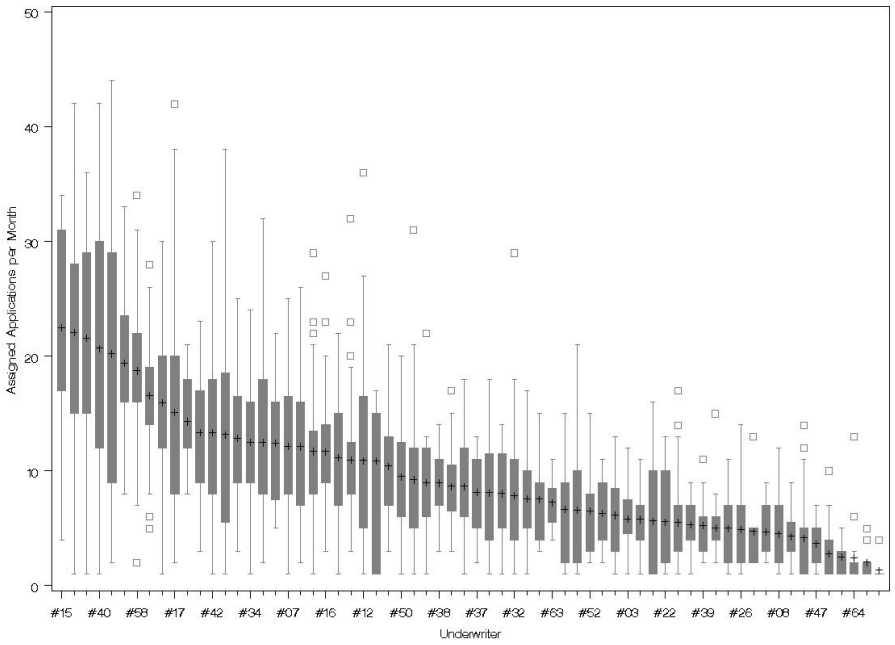

Figure 3. Average monthly applications assignment per underwriter.

On the one hand, it would be reasonable to assign more applications to underwriters with better performance. However, on the other hand, it is possible that fewer applications allow the underwriters to analyze the situation more comprehensively, to have a more in-depth negotiation with a prospect – and as a result to show better performance. We will discuss these hypotheses further.

Dynamic Conversion Rate per Underwriter

We calculated dynamic conversion rate DCR_{mi} per underwriter over the observation period of 44 months. For each underwriter, we calculated an average of her/his DCR_{mi} over 44 months period. Best underwriters, with an average dynamic conversion rate above 0.12, comprise about 8% of all underwriters. As shown in Figure 4, top performers have a significantly higher dynamic conversion rate than the rest of the underwriters.

We can now analyze the relations of the number of assigned applications to an underwriter with her/his performance measured by DCR_{mi}. For the top performers, the average number of monthly assigned applications per underwriter is 13 (median is 12). For the rest of the underwriters, the average number of monthly assigned applications per underwriter is 10 (median is 8). We performed a comparison of empirical distributions of numbers of monthly assigned applications per underwriter between the group of the top performers and the group of the rest of the underwriters using the Kolmogorov-Smirnov two-sample test. The asymptotic *p*-value for the Kolmogorov-Smirnov test is <.0001. This indicates that we should reject the null hypothesis that the distributions of numbers of monthly assigned applications per underwriter are the same for the 2 groups of underwriters. This demonstrates that, indeed, top underwriters (according to Dynamic Conversion Rate performance metric) assigned more applications than other underwriters, and regardless of increased workload, they still do substantially better than others.

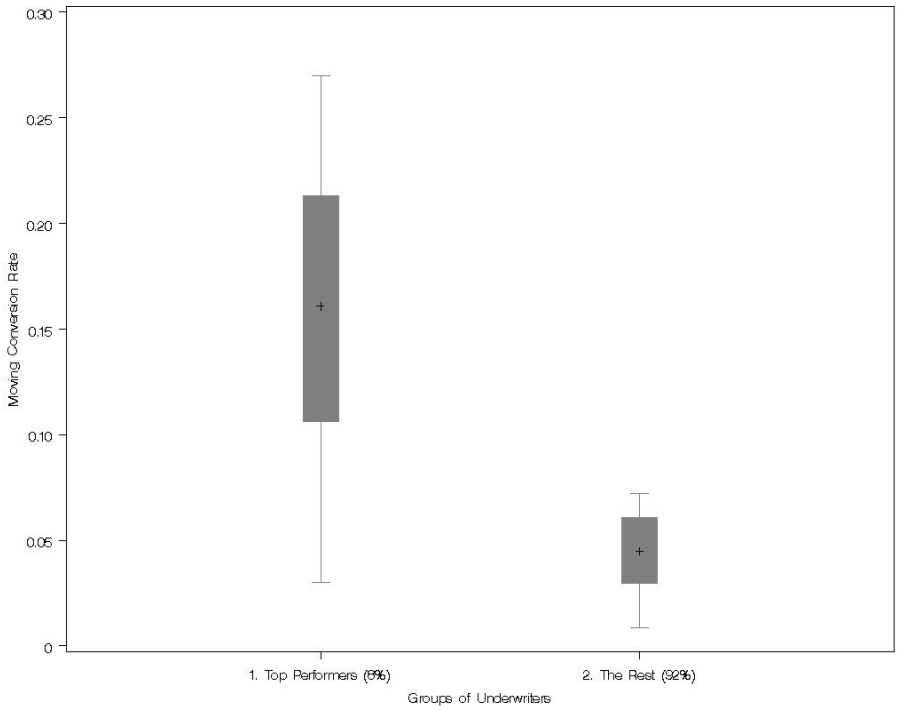

Figure 4. Comparison of Dynamic Conversion Rates between 2 Groups

Time-to-Deal per Underwriter

We calculated TTD_{mi} per underwriter. Time-to-deal varies significantly, from 0 to 90 days (where 90 days is an expiration term of a quote), with average 32 days and median 29 days. To compare time-to-deal of top performers vs. the rest of underwriters, we performed a comparison of empirical distributions of TTD_{mi} between these 2 groups using the Kolmogorov-Smirnov two-sample test. The asymptotic *p*-value for the Kolmogorov-Smirnov test is 0.1255. This indicates that we can't reject the null hypothesis that the distributions are the same for the 2 groups of underwriters. Although TTD_{mi} can't be used as a metric differentiating top performers from the rest of the underwriters, this metric is useful as an operational metric for the purposes of ongoing monitoring and management of underwriters' performance. This metric also can be used to build models to predict time-to deal.

TRADITIONAL APPROACH TO KNOWLEDGE DELIVERY

It is an ongoing effort for insurers to improve the quality and consistency of the underwriting process. For years, sound underwriting decisions were considered the result of the judgment, experience, and "gut feelings." The only way for a new or young underwriter to gain "judgment" was by working under the guidance of a successful senior underwriter and gradually gaining experience. (Randall, 1994).

Traditional ways to validate performance include underwriting file reviews, or audits, in trying to improve quality and consistency. These audits attempt to determine the effectiveness of underwriters. The audit process usually consists of six steps, validating each phase in the underwriting decision process:

1. Gather data on an applicant's loss exposures and associated hazards.
2. Evaluate the exposures and hazards.

3. Develop and test underwriting options.
4. Select an underwriting option, including selecting the appropriate premium.
5. Implement the underwriting decision.
6. Monitor the loss exposures.

This audit may find deficiencies in underwriting processes and provide a basis for additional coaching, improved documentation, and enhanced quality reviews. However, each step in this process is validated subjectively and qualitatively and is based on two assumptions. The first one is proper guidelines, which are usually updated infrequently and become obsolete with time. The second assumption is the existence of a matured management process, where the assignment of applications to underwriters is balanced, and identification of a problem and intervention to fix it are timely.

Unfortunately, in today's dynamic, large-scale and complex environment, these assumptions are often not valid. Also, the growing need for underwriters requires that training is streamlined, consistent and fast. Though thinking of underwriting as steps in the decision-making process clarified its nature, it only led to static guidelines that didn't include experience, sound judgment, and intuition of the top underwriters.

From our analysis of underwriters' performance, it is clear that a few underwriters perform significantly better than others, despite the guidelines and audits. More experienced and knowledgeable underwriters apply skills that can't be captured by guidelines, and their expert knowledge allows them to intuitively analyze multiple factors and their combinations to produce a better decision.

Artificial intelligence approaches based on machine learning algorithms can be used to model the decision-making process of the best underwriters, and then these models can be applied to ease and automate the underwriting decision-making process.

ANATOMY OF ARTIFICIAL INTELLIGENCE SOLUTION

The underwriting decision-making process includes the following components:

- Underwriters, who are assigned to work on different applications, in our example – for workers compensation insurance
- Insurance applications from various clients/prospects along with a description of different characteristics of the business, including business size, turnover, etc.
- Insurance products for workers compensation that have different features, including premium size, conditions of insurance, etc.
- Clients/prospects responses to underwriters quotes: accepting or rejecting of proposed insurance product

To formalize the underwriting decision-making process, we start with creating the proper data structure.

DATA STRUCTURE

Let $i = 1 \dots n$ denote underwriters, $j = 1 \dots m$ denote applications, and $k = 1 \dots p$ denote insurance products for workers compensation.

Each application is described by a set of features (parameters), such as the number of employees in the company, ages of the employees, the industry code, US state, number of claims in the previous year, etc. These features are standardized and presented by a vector

$A_{\mathrm{j}} = \{a_{\mathrm{j1}}, a_{\mathrm{j2}}, \dots a_{\mathrm{jM}}\}$,

where M is the number of insurance application features (parameters or characteristics).

Each insurance product is described by a set of features (parameters or characteristics), for example, the conditions covered by the policy, the deductible amount, premium amount, etc. Insurance product features presented by a vector

$$B_{\mathrm{k}} = \{b_{\mathrm{k1}}, b_{\mathrm{k2}}, \dots b_{\mathrm{kP}}\},$$

where P is the number of insurance product features. These features are standardized for all products, and if, for example, feature b_{k2} is not presented in the product k, then b_{k2} is assigned 0.

Let Y_{ijk} denote an outcome of each quoted insurance application – 1 means bound, and 0 means rejected. This structure allows capturing multiple quotes made by the same underwriter as a response to the same insurance application.

For example, the data structure will look like in Table 2.

Table 2. Example of Data Structure

Underwriter	Application	Features of Application					Product	Features of Product			Outcome
i	j	Number of employees a_{j1}	% of employees under 21 a_{j2}	% of employees over 50 a_{j3}	...	Claims in the last year $a_{j\mathrm{M}}$	k	Deductible amount b_{k1}	...	Premium amount b_{kP}	Y_{ijk}
...	...	...	...	...	...	...	...	...	...	...	...
10	100	30	5	3	...	0	1001	10,000	...	1,000	1
10	105	100	2	10	...	5	1001	40,000	...	1,500	0
10	105	100	2	10	...	5	1003	30,000	...	1,150	1
...	...	...	...	...	...	...	...	...	...	...	...
13	201	70	6	11	...	3	1002	22,000	...	1,500	0
13	202	47	4	9	...	1	1002	11,000	...	900	0
...	...	...	...	...	...	...	...	...	...	...	...

CLASSIFICATION APPROACH

In the broadest sense, we try to solve the problem of predicting whether or not the customer/prospect will accept the insurance product offered by an underwriter. We need to find a fit between features (parameters or characteristics) of the insurance application and the features of the insurance product and its price. The real challenge is that data is very unbalanced: bound insurance applications comprise about 9% of the quoted applications.

We approach the problem using the support-vector machine (SVM) -- a proven classification method that is also well-suited to our problem because it can directly minimize the classification error without requiring a statistical model. SVM can handle the massive feature space, and it is guaranteed to find a globally optimal separating hyperplane if it exists. Also, SVM models are built on the boundary cases, which allow them to handle missing data.

An SVM classifier can separate data that is not easily separable in the original space (for example, 2-dimensional space) by mapping data into a higher dimensional (transformed) space.

Simply saying, a non-linear classifier maps our data from the input space X to a feature space F using a non-linear function $\Phi: X \to F$. In the space *F* the discriminant function is:

(5)

$$f(x) = w^T \Phi(x) + b$$

Kernel

Kernel methods resolve the issue of mapping the data to a high dimensional feature-space. Suppose the weight vector can be expressed as a linear combination of the training examples:

(6)

$$w = \sum_{i=1}^{n} \alpha_i x_i$$

Then

(7)

$$f(x) = \sum_{i=1}^{n} \alpha_i x_i^T x + b$$

In the feature space, F this expression takes the form:

(8)

$$f(x) = \sum_{i=1}^{n} \alpha_i \Phi(x_i)^T \Phi(x) + b$$

The feature space F may be high dimensional, making this transformation difficult, unless the kernel function is defined in the form that can be computed efficiently:

(9)

$$K(x, x') = \Phi(x)^T \Phi(x')$$

In terms of the kernel function, the discriminant function can be written as:

(10)

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x, x') + b$$

There are different kernel functions. For example, the polynomial kernel of degree *d* is defined as:

(11)

$$K(x, x') = (x^T x' + 1)^d$$

The feature space for this kernel consists of all monomials up to degree *d*, for example:

(12)

$$x_1^{d_1} x_2^{d_2} \dots x_m^{d_m}, where \sum_{i=1}^{m} d_i \leq d$$

The increasing degree of the polynomial d leads to the increased flexibility of the classifier.

The other widely used kernel is the Gaussian (radial basis function) kernel defined by:

(13)

$$K(x, x') = \exp(-\gamma \|x - x'\|^2)$$

where $\gamma > 0$ is a parameter that controls the width of Gaussian. It plays a similar role as the degree of the polynomial kernel in controlling the flexibility of the resulting classifier.

Margin

The margin is the length of the normal vector to the hyperplane from the 2 closest cases called support vectors – in other words, it is the margin with which the two classes are separated.

For a given hyperplane we denote by $x_+(x_-)$ the closest point to the hyperplane among the positive and negative cases. The norm (length) of a vector *w* is denoted by $\|w\| = \sqrt{w^T w}$. From a geometric perspective, the margin of a hyperplane *f* with respect to a data set *D* can be defined as:

(14)

$$m_D(f) = \frac{1}{2}\hat{w}^T(x_+ - x_-) = \frac{1}{\|w\|}$$

where $\hat{w} = w/\|w\|$ is a unit vector in the direction of *w*, assuming that x_+ and x_- are equidistant from the decision boundary.

Optimization

SVM uses a kernel function to find a hyperplane that maximizes the distance (margin) between 2 classes (in our case, rejected quotes vs. accepted quotes) while minimizing training error.

The maximum margin classifier is the discriminant function that maximizes the geometric margin $\frac{1}{\|w\|}$ which is equivalent to minimizing $\|w\|^2$.

The soft-margin optimization problem (Cortes et al., 1995) is formulated as follows:

(15)

$$minimize\ on\ w, b: \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} e_i$$

$$subject\ to: y_i(w^T x_i + b) \geq 1 - e_i, \quad e_i \geq 0, \quad i = 1, \ldots, n$$

where:

w – margin,

b – the bias that translates the hyperplane away from the origin,

e_i – slack variables that allow an example to be in the margin ($0 \leq e_i \leq 1$, also called a margin error) or to be misclassified ($e_i > 1$).

C – the constant that sets the relative importance of maximizing the margin and minimizing the amount of slack.

Bias-variance tradeoff and SVM hyper-parameters

The bias is an error caused by wrong assumptions in the learning algorithm. High bias means an underfitting between features and outputs. The variance is an error produced

from variations in sensitivity metric in response to small fluctuations in the training sets. High variance means overfitting between features and outputs.

The bias-variance tradeoff is a central problem in supervised learning. In statistics and machine learning, the bias-variance tradeoff is the property of a set of predictive models, whereby models with lower bias in parameter estimation have a higher variance of the parameter estimates across samples and vice versa. The bias-variance problem is the conflict in trying to simultaneously minimize these two sources of error that prevent supervised learning algorithms like SVM from generalizing beyond their training sets.

The quality of results produced by SVM depends on the following hyper-parameters:

- The type of kernel (radial, polynomial, sigmoid, etc.)
- the cost of misclassification (penalty) that is denoted as C in (15),
- the parameters of the kernel, for example, γ in the cases of radial or sigmoid kernel transformations (13), or *d* in the case of the polynomial kernel (11)(12).

Large *C* leads to low bias and high variance, while small *C* leads to high bias and low variance. Large *d* or γ lead to high bias and small variance, while small *d* or γ lead to low bias and high variance. For example, if γ is large (variance is small), it means that the SVM results do not have a wide influence and can't be generalized beyond its training sets.

Because SVM doesn't model the data distribution, but instead directly minimizes the classification error, the output produced by SVM is a binary decision. Although a binary decision is sufficient for many problems, it is difficult to arrive at a meaningful bias-variance tradeoff. This concern can be alleviated by applying penalized logistic regression to SVM output. The benefit of using penalized logistic regression to SVM output is making results more generalizable (for example, decrease overfitting). In classification, SVM along with penalized logistic regression create output probabilities of class membership rather than point estimates produced by SVM alone. Re-phrasing a classification problem as a probabilistic classification creates conditions for application of bias-variance decomposition that was originally formulated for least-squares regression.

Building the classifier

The proposed AI solution attempts to "learn" from the best underwriters. Thus in our study, we used only data related to the top performers comprising 8% of all underwriters. The data we worked with was structured to express all problems of similar nature by broadly including all features (characteristics or parameters) of the insurance application and the offered product and price, along with binary values Y (equivalent to acceptance or rejection of a quote by a prospect) as presented in Table 2.

To avoid bias in our analysis, we created training and test data sets from the original data set using the "m-out-of-n" bootstrap method with replacement. This method replaced complicated and often inaccurate approximations of biases, variances, and other measures of uncertainty by computer simulations. In simple situations, the uncertainty of an estimate may be gauged by analytical calculation based on an assumed probability model for the available data. But in more complicated problems this approach can be tedious and difficult, and its results are potentially misleading if inappropriate assumptions or simplifications have been made. Applying the "m-out-of-n" bootstrap method with replacement, we select 2,000 training data sets that have the same distribution of outcome (reject/accept of insurance quote) as the original data set of the top performers' data. We also randomly select test data set.

Step 1

As a first step, we create statistically designed full factorial experiment to find out optimal hyper-parameters of SVM: kernel transformation function (radial, polynomial, and sigmoid kernel types), the kernel hyper-parameter (*d* or γ), and the algorithm hyper-parameter – *C*. For continues values of *C,* and *d* or γ hyper-parameters we aggregate their values in multiple groups. The SAS® FACTEX procedure can be used to create a full factorial experiment. For example, if we decide to create 6 groups of values for *C* and γ hyper-parameters, our full factorial design consists of 3^1 * 6^2=108 combinations (runs).

Step 2

For each of combinations presented in our full factorial design we execute the SAS HPSVM procedure as following:

```
%macro SVM (data=training_set, c=100, gamma=2, kernel = rbf,
            input_app = num_emp pct_under_21 pct_over_50 claims,
            input_prod = deduct premium output = output, target=outcome);

    PROC HPSVM DATA = &data
       C = &c /* the penalty value */
       MAXITER=10000 /* the maximum number of iterations */

    INPUT &input_app &input_prod / LEVEL=INTERVAL ;
    KERNEL &kernel
    %if %trim(%left(&kernel)) = rbf or
        %trim(%left(&kernel)) = sigmoid %then
        %do ;
          / k_par = &gamma ;
        %end ;
   OUTPUT &output ;
   TARGET &target ;
   run;
%mend SVM ;
```

As the final classifier, we select one that maximized sensitivity (the proportion of bound applications that were correctly classified) and specificity (the proportion of rejected quotes that were correctly classified) when applied to the training data sets.

Step 3

As a next step, to add a probabilistic element to the SVM results, we apply Firth's penalized logistic regression to the outcome of the SVM algorithm. SVM produces complete separation data. Firth's penalized likelihood approach is a method of addressing issues of separability. Firth's method is available in the LOGISTIC procedure and only for binary logistic models. It replaces the usual score equation:

(16)

$$g(\beta_j) = \sum_{i=1}^{n} (y_i - \pi_i) x_{ij} = 0 \ \ (j = 1, \ldots, p)$$

where p is the number of parameters in the model, with the modified score equation:

(17)

$$g(\beta_j) = \sum_{i=1}^{n} \left(y_i - \pi_i + h_i(0.5 - \pi_i)\right) x_{ij} = 0 \ \ (j = 1, \ldots, p)$$

where h_i is the *i*-th diagonal elements of the hat matrix:

(18)

$$W^{\frac{1}{2}}X(X'WX)^{-1}X'W^{\frac{1}{2}}$$

$$W = diag\{\pi_i(1-\pi_i)\}$$

The SVM_Decision data set presented in Output 1 below has 2 columns: Decision_Function column has SVM decision function values, and Outcome column has actual outcomes for each case (accepted offer is presented as 1 and rejected as 0).

Decision_Function	Outcome
0.1	1
0.5	1
0.9	0
-0.1	0
…	…

Output 1. SVM_Decision Data Set

The code below implements logistic regression with Firth's correction for bias:

```
PROC LOGISTIC data = SV_Decision ;
  CLASS Label (Param=Ref Ref='0') ;
  MODEL Label (event='1') = Decision_Function / CL FIRTH ;
RUN ;
```

Since the results of logistic regression are probabilities, we chose a cutoff or a threshold value to classify cases. The probabilities above this threshold value are classified as 1, and those below as 0. For the probabilities generated by the Firth penalized logistic regression, we search for such a threshold value that it further increases the sensitivity and the specificity of the results produced by SVM.

We perform the analysis for each of the 2,000 training data sets. As a final classifier, we select an SVM classifier with adjustment based on Firth penalized logistic regression that, on average, produces higher sensitivity and specificity metrics.

After applying different kernel types and various SVM hyper-parameters (according to the full factorial design of experiment described above), the best results occur with a radial kernel having a cost parameter $C \in [1000; 3000]$ and $\gamma \in [0.2; 0.6]$. The sensitivity of the classification results equals 0.987 on average, and the specificity equals 0.799 on average.

Then, we identify a threshold that improved classification based on the results of the Firth penalized logistic regression. The threshold value of 0.76 used to classify the results of the Firth regression, improved specificity to the level of 0.847 on average at the cost of a minor loss in sensitivity – which became equal 0.973 on average.

"Contamination" of training data sets

Now, we address the variance in sensitivity metric caused by small fluctuations in the training sets. To avoid high variance that causes overfitting between features and outputs, we perform "contamination" of training data sets. The "contamination" is performed by changing a small percentage of outcomes in the training data sets, planned according to the design of mixture experiments. In mixture experiments, we vary the proportions, as in our example – proportions of different outcomes. The details of the design of mixture experiments are out of the scope of this article; we only state that this approach allows performing controlled "contamination" of data. After the training data set is contaminated according to the design, we perform again Steps 1 - 3 where radial kernel having a cost parameter $C \in [1000; 3000]$ and $\gamma \in [0.2; 0.6]$ is considered as a starting point of the process. As a result, a robust classifier is built.

Applying the developed classifier to test data, we received the following results: specificity equals 0.859, and sensitivity equals 0.851.

Experimental Results

To test the efficiency of the solution, we used the following historical data:

1. properties/features of insurance products and prices that were offered in regards to insurance applications;
2. properties/features of insurance applications to which insurance products were offered ;
3. data about acceptance or rejection of insurance products by clients.

The data for the experiment was selected in such a way that all offers were rejected by clients. In other words, the conversion rate in this data sample was 0%.

The goal of the experiment was to apply the developed AI solution to find what insurance products and prices would fit the insurance applications in the way they would be accepted by clients. Each insurance application in the sample data received one quote for an insurance product, in rare cases two quotes. In our experiment, however, each application was combined with all possible insurance products and their prices. An example of such a data is presented in Table 3 where:

- gray-shadowed rows present the application and the product that was rejected by the customer, and
- white rows present all other possible products that could be offered.

This data was used as an input to our AI solution.

Application j	Features of Application					Product k	Features of Product			Classification outcome Y_{ijk}
	Number of employees a_{j1}	% of employees under 21 a_{j2}	% of employees over 50 a_{j3}	...	Claims in the last year a_{jM}		Deductible amount b_{k1}	...	Premium amount b_{kP}	
...	...	...	...	...	...	...	...	...	...	
201	70	6	11	...	3	1002	22,000	...	1,500	0
201	70	6	11	...	3	1001	20,000	...	1,250	0
201	70	6	11	...	3	1003	25,000	...	1,150	1
201	70	6	11	...	3	1004	28,000	...	1,000	0
202	47	4	9	...	1	1002	11,000	...	900	0
202	47	4	9	...	1	1001	10,000	...	900	0
202	47	4	9	...	1	1003	13,000	...	850	1
202	47	4	9	...	1	1004	15,000	...	800	1
...	...	...	...	...	...	...	...	...	...	

Table 3. Example of Historical Applications Data with Different Product Offers

The column entitled "Classification outcome Y_{ijk}" of Table 3 contains results of applied AI solution. Value 1 in this column identifies products that with high probability would be accepted by the customer. Relevant quotes for insurance products and prices identified by the AI solution expected to lead to the conversion rate of 14.7%. In other words, the experiment identified the applications and the products that would most likely lead to binding of the applications.

CONCLUSION

Insurance industry still struggles with the pace of knowledge delivery from experienced underwriters to young ones. Disadvantages of currently used methods, based on apprenticeship, are subjectivity of underwriters' decisions. Measurement of underwriters' performance also suffers from lack of efficiency, as existing metrics are delayed and not operational. We analyzed underwriters' performance metrics and proposed a novel one, Dynamic Conversion Rate, which allows measuring and monitoring the operational performance of underwriters (weekly, monthly, etc.)

We proposed the automation of the underwriting decision-making process based on the SVM machine-learning method and its improvements. Overall, with the reasonably efficient classifier built by SVM, and improved with a threshold of Firth penalized logistic regression, we achieved satisfactory results regarding sensitivity and specificity metrics.

We applied the proposed approach to real-life insurance applications and showed that if our AI solution was used in quoting of those insurance applications (in our experimental case -- workers compensation insurance), it would yield incremental improvement in application binding by 14.7%.

As the next step in our research, we are looking to apply our AI solution to different insurance lines like financial lines, small businesses, etc.

REFERENCES

Bickel, P. J., Ren, J. J. 1995. "The 'm Out of n' Bootstrap and Goodness of Fit Tests with Double Censored Data." *Robust Statistics, Data Analysis and Computer Intensive Methods*. 109:35-47.

Chidambaran, N. K., Pugel, T. A., Saunders, A. 1997. "An Investigation of the Performance of the U.S. Property-Liability Insurance Industry." *The Journal of Risk and Insurance.* 64:371-382.

Cortes, C., Vapnik V. 1995. "Support-Vector Networks." *Machine Learning*, 20:273-297.

Firth, D. 1993. "Bias Reduction of Maximum Likelihood Estimates." *Biometrics*, 80 No. 1: 27-38.

Kunreuther, H., Meszaros, J., Hogarth, R. M., Spranca, M. 1995. "Ambiguity and underwriter decision processes." *Journal of Economic Behavior & Organization,* 26:337-352.

Randall, E.D. 1994. "Introduction to Underwriting." Insurance Institute of America.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Tanya Kolosova	Samuel Berestizhevsky
425-241-6846	862-215-4611
tanyak@inprofix.com	samuelb@inprofix.com
http://www.inprofix.com	http://www.inprofix.com