Modeling Customer Attrition Based on Post-Event Insurance Data

Constanza Schubert[a,b]

[a] *11200 SW 8th Street ECS 354 Miami, FL 33199*
[b] *cschu031@fiu.edu*

Keywords: insurance, attrition, churn, binary classification, text classification.

Abstract: The insurance industry is a highly competitive space where customer experience and retention are key performance indicators for solidifying a competitive advantage. The objective of this project is to predict the likelihood of losing a certain customer and recommend actions to prevent attrition. Post-claim data containing claim details and survey responses were collected and assigned an overall attrition label for training various models. Organizations can use predictions to act on customers who are likely to churn and proactively incorporate recommendations into their standard procedures within claim operations. The implementation yields results better than the baseline for predicting the minority class (customers who did churn).

1. Introduction

Customer attrition, or the loss of customers, is used as a key performance indicator across several industries, including insurance. The insurance industry's customer acquisition costs are high, so it is important to retain existing customers rather than only focus on getting new ones [1]. Insurance companies possess transaction and customer level data which can be leveraged to model attrition. The distribution of resources for both customer retention and acquisition can be improved by being able to predict the likelihood of a customer to cancel their policy after filing a claim. A company can prioritize their efforts on people who have can probably be convinced to keep a policy through customer service strategies. Specifically, this project uses data provided by Assurant Inc., with a purpose to capitalize on such improvements.

Assurant is a global provider of housing and lifestyle solutions which protect, connect, and supports major consumer purchases [1]. Such purchases include homes, rental properties, vehicles, mobile devices, appliances electronics, and funerals. Assurant partners with brands in these spaces to offer insurance products, customer care claims, technical support, and B2B2C sales support. The organization is comprised of three global business units: Global Lifestyle, Global Housing, and Global Preneed. Global Lifestyle serves to connect and protect consumer devices, appliances, cars, and transactions. The Connected Living (CL) group within Global Lifestyle is responsible for the protection, repair, refurbishment, and replacement of mobile devices, smart home products, appliances, and electronics. Connected Living helps keep 48 million mobile devices connected and protected; along with 80 million appliances, 17 million tools, and 9 million pieces of jewelry and furniture. Given that Connected Living is a large business unit within the organization, this project will focus on predicting attrition specifically for customers of mobile insurance products. The aim of this project is to predict the probability of a customer canceling their policy after a certain period of time after a claim was created by using claim attributes and possibly text as features, depending on the model performance without text.

Assurant developed a proactive fraud prevention strategy designed to prevent, detect, and deter fraud. Dynamic Claims Management (DCM) is Assurant's global end-to-end claims process that includes processes that automate loss verification and maximize the fulfillment process. A risk assessment model determines the adjudication and fulfillment protocol applied to a claim with the goal of mitigating fraud risk while optimizing customer experience. The engine within DCM defines the course of action based on the circumstances presented by the claim but the system as a whole is flexible, allowing for human intervention and modification of the business rules. The risk assessment model assigns a score to a claim which falls within low, medium, or high risk. When there is a change in key data (customer information, address, payment method, velocity/fraud alerts, prior non-compliance, etc.), the claim gets scored again. Low risk claims are routed for authorization. Medium risk claims are routed to a request for proof; only until all required information has been received will a decision by made to reject or authorize the claim. High risk claims are routed to a rest for proof followed by an

internal review and then a decision is made. A claim issue is generated when there is any missing or additional information required prior to the claim being authorized, and the claims status changes to either Pending or In Review. The customer is advised that the claim is being reviewed and what the expectations are; this is known as the scrutiny process. A scrutiny level gets assigned to each issue associated with a claim, and the issue with the highest severity of scrutiny gets selected as the overall scrutiny grouping.

1.1. Motivation and Potential Applications

The prolongment of a customer's lifetime value (CLV) is important to remain competitive in the insurance industry. Plus, efforts to retain existing customers are less expensive than those to get new customers [2]. Customer attrition models which work well allow the development of strategies to retain and create loyalty in existing customers [2]. A predictive model can be integrated into a broader Customer Continuity Management (CCM) application which would contain customer data from various source applications in one view. A CCM application is supposed to show factors influencing attrition and a tool for managing the predicted and actual churned customers [2]..

2. Project Goals

The key goal of this project was to answer the following questions:

- Is it feasible to model customer attrition using claim data and achieve actionable results?
- Can specific attributes associated with high likelihood of attrition be derived?
- What recommendations can be made to improve attrition rates?

Model performance was evaluated against a baseline and feature importance analysis was conducted to determine the attributes that contribute to attrition.

3. Implementation

The approach was to build a model using only the structured fields from the data first and then incorporate language to improve performance. Initially, a binary feature indicating whether the open-ended survey question was answered was included to assess its impact. An SVM classifier among others were evaluated. A starting point for using language as features is a bag of words (BoW) model and subsequently TF-IDF. The work then progresses to use pre-trained word embeddings (Word2Vec, GLOVE) in a deep learning model. Training a word embedding using existing data was also considered to see if performance would improve. Lastly, a recurrent neural network (RNN) sequential classifier was used on a second dataset, to consider how claim events over a customer lifecycle affect attrition, rather than considering a single claim event.

3.1. Data Collection

All the data used for this project originates from Assurant's data warehouse. The primary dataset used is a customer-level table that is created using a SQL script which selects certain columns and filters the [AMSDOM_CLAIM] table within the [DW_FORECAST] database. The [AMSDOM_CLAIM] table consists of the set of customers who both filed a claim and completed a post-claim survey asking them questions about their experience. The answers to only three survey questions were included in the final table: Question 1, Question 2, and Question 3.

Question Number	Description	Response type
1	How likely is it that you would recommend the Equipment Protection program to a friend or colleague?	Numeric
2	In our continuous efforts to better serve you, please tell us why you chose these ratings.	Numeric
3	How satisfied are you with your replacement device?	Text

Table 1: Survey questions included in primary dataset.

Table 1 indicates the question descriptions. The answer to Question 1 is a number from 1 to 10, the net promoter score (NPS) rating. Question 2 asks the user to explain the reasoning behind the answer to Question

1 in the form of a free text response; this question is going to be used to extract features from text and predict attrition. The answers to this question vary in length. Question 3 asks the use to rate their overall satisfaction with the replacement device. Some of the attributes from the claim portion of the data include: claim status, device model claimed, device model shipped, whether the replacement device was new or not, loss type, scrutiny level, shipping method, deductible amount, and total claim time.

The number of days until cancellation can be calculated by subtracting the claim creation data from the policy end date. The size of the data without excluding individuals who did not answer the open-ended question in the survey is over 126,700 records for 1.5 years' worth of claims. The size is drastically reduced when considering only those who answered the open-ended question for building a model based on language and if we choose to limit the data range further. While exploring the data briefly, the cancel rate for a policy after a claim has been filed is approximately 20%. This percentage will vary based on the date range and other filters. Nevertheless, the classes will be imbalanced which is a challenge. And, the cancel rate will only get smaller if we limit the duration the client was active after the claim was created to 90 days, which tends to be a metric of interest to the business.

The secondary dataset was created to use as input to a recurrent neural network. The structure of the data differs from the main dataset in that we have multiple rows per enrollment, or customer. An enrollment is defined as a record of the wireless subscriber who elected the insurance product. A record is the set of individual mobile device(s) and the account the devices are attached to. A SQL script which generates the dataset uses two different databases from an Assurant server to capture the enrollments and claim details. It is comprised of 2 temporary tables (enrollment and claims) and a final select statement. The first temporary table selects the enrollments, filters, and indexes by creating a partition on the enrollment, equipment, and product ID's from the fact enrollment table. The index is a means to distinguishing device changes throughout a customer's subscription period, because in the enrollments table each row represents a change to the enrollment. The second temporary table selects all columns from the indexed enrollments table plus columns from the dimension enrollment, carrier, and location tables and joins them together. The last select statement aggregates the previous temporary table to get the number of device changes per enrollment and joins with columns from a claims table. Since a customer can file multiple claims, this join results in the output to have multiple rows per enrollment, one for each claim filed.

List of features in sequential dataset	
Account number	Claimed model
Subscriber coverage start and end date	Shipped model
Number of device changes	Deductible amount
Customer date of birth	Shipment count
Customer Zip code	Reshipped y/n
Claim creation date	Denied status date
Delivery type	Closed status date
Claim status	First shipped date
Claim source type	Calls per claim
Loss type	Issue count
Scrutiny severity level	Days to file claim after enrollment
Claimed model	Number of days since last claim

Table 2: List of features in dataset used for RNN.

The zip code was included to join with demographics data and the different status dates to calculate elapsed times since the claim was created and approved/authorized.

3.2. Preprocessing and Feature Engineering

A set of derived features were obtained from the data. Before extracting features from text responses, a binary feature was created based on whether the customer responded or not, to see if this is a significant predictor of attrition. The attrition period was calculated by subtracting the claim creation date from the subscriber end date and converting to months. Customers who are still enrolled have a subscriber end date of 12/31/9999'. A target attrition variable was defined in three ways: two-month, three-month, and overall attrition. Two-month attrition corresponds to the attrition period being 2 months or less and three-month attrition corresponds to 3 months or less. Overall attrition does not place an upper bound on the attrition period. Models were tested with each target variable.

The distribution of missing data across all fields was inspected. It was observed that the bulk of missing values came from claims which did not have a replacement device shipped to the customer. This is most often due to the claim being denied, and otherwise because the customer chose to pick up the device. These are still valid observations so they were not removed. The missing values in categorical features were treated as its own level when one-hot encoding to capture the meaning from the null data. The rest of missing values were from customers choosing not to answer some or all the survey questions included in the dataset. Observations in which ANSWER_1 or ANSWER_3 were blank were handled by using a simple imputer, replacing the blanks with the most frequent value. ANSWER_2 is the text response so missing values were not replaced. Categorical variables were one-hot encoded and any features which had a single constant value were discarded. The following were one-hot encoded: Delivery type, status description, claim source type, carrier, loss type, scrutiny severity level, and shipping method.

It has been previously validated statistically that certain differences between the mobile device claimed and the replacement device shipped to the customer are critical to their satisfaction and aggregated net promoter (NPS) score. These differences are the model, color, and capacity matching the claimed device. The data warehouse does not contain structured fields for both claimed and shipped devices on model, color, and capacity; the model claimed and model shipped fields contain the entire description of the device.

To determine whether the model name of the devices match, a function was created which calculates the standard Levenshtein distance similarity ratio between two strings. A threshold of 0.8 was set to distinguish a match from a non-match. For the color match feature, the color had to be extracted from the MODEL_CLAIMED and MODE_SHIPPED fields. Table 3 shows that the colors are inconsistently abbreviated, and there is no consistent location for the color to be in within the string. The solution was to conduct a fuzzy match using a regular expression (regex) pattern [3]. Each record gets searched on a regex pattern pertaining to each color on the list. The fuzzy regex search was set to permit at most 2 errors of any type (insertion, deletion, or substitution) when finding a match, since the colors within the strings are abbreviated). Once the standard colors are extracted for each field, they are compared and the Color Match field is created. A binary field for the capacity matching between the claimed and replacement device is obtained similarly, except that the regex match is exact instead of fuzzy. The pattern consists of looking for 2-3 numerical digits which are followed by gb.

MODEL_CLAIMED	MODEL_SHIPPED
IPHONE X 64GB GRY	IPHONE 11 PRO 512GB GRY
GALAXY S8+ 64GB BLK G955	GALAXY S8+ 64GB BLK G955
IPHONE 8+ 64GB GRY	IPHONE 8+ 64GB GRY
GALAXY S8 64GB BLK G950	GALAXY S8 64GB BLK G950
IPHONE XR 64GB BLU	IPHONE 11 PRO 64GB SLV

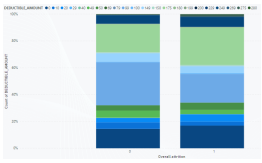
Table 3: Examples of "MODEL_CLAIMED" and "MODEL_SHIPPED" fields from which to extract the color and capacity.

Furthermore, the different types of scrutiny counts were combined into two categories, high and low scrutiny, by summing up certain scrutiny count fields. This was done because the individual fields did not have much variation across the target variables and were sparse. The scrutiny counts were aggregated by issue type, and each issue type is associated with a severity level, which is a number between 1 and 9. The issue types in descending order of severity are: Affidavit, SIU Review, Proof of Loss (POL) CRB Review, Original Invoice, Police Report, POL Only, Customer Authorization, ePrism Issue, and None. The High Scrutiny Count feature was created by summing up the counts for Affidavit, SIU Review, POL CRB Review, Original Invoice, and Police Report. Low Scrutiny Count is comprised of the remaining columns: POL Only, Customer Authorization, and ePrism Open Issue.

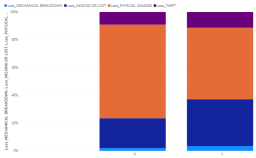
3.3. Data Characteristics

The characteristics of the data was explored through various visualizations, including plots from one-way tables, boxplots, and plots from two-way tables. Figure 1a shows that at a deductible of $175, which is on the higher end, there is a higher percentage of customers who choose to leave than to stay. A similar observation can be made for customers who lose their mobile devices.

Figure 2a shows that customers who get their claim denied tend to leave relatively more often than those who get their claim authorized. Figure 2b indicates the same observation for customers who receive a replacement device that does not match their claimed device. Figure 3a shows how the approximate attrition rate varies by different issue counts of high scrutiny on claims. One can see that for claims that have 6 or more issues of

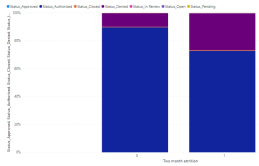


(a) Attrition variable counts across the deductible amount.

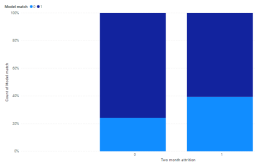


(b) Attrition variable counts across the claim loss type.

Figure 1: Visuals that describe how the deductible amount and loss type vary by the attrition variable.



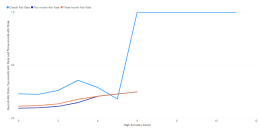
(a) Attrition variable counts across the claim status.



(b) Attrition variable counts across Model Match binary variable

Figure 2: Visuals that describe how the claim status and model match variable by the attrition variable.

high scrutiny, all customers end up leaving. Figure 3b displays the same concept but for low scrutiny issues on claims. For claims with under 10 issues of low scrunity, the attrition rate is under 0.2 which is lower than than the same under of issues but for high scrutiny.



(a) Attrition rate across number of High scrutiny issues on a claim.

(b) Attrition rate across number of Low scrutiny issues on a claim.

Figure 3: Visuals that describe how the attrition rate varies by the number of high and low scrutiny issues on a claim.

Figure 4 shows the attrition rates varying over the total time a claim stayed open. This excluded claims that were still under review when the data was extracted; only approved, authorized, denied, and closed claims were considered for this visual. There is a spike in the attrition rate for when the claim has been open for about 20 days.

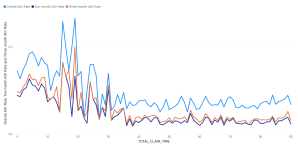
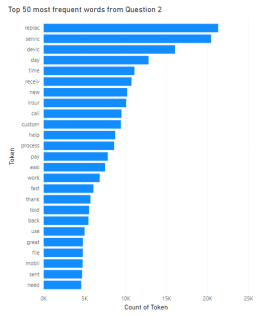


Figure 4: Attrition rate across the total time a claim was open.

After initial preprocessing of the survey responses (tokenization, stop word removal, and stemming), the most common tokens (words) were examined, as shown in figure 5a. The sentiment score of the responses showed that the majority are neutral and the tokens most associated with positive and negative sentiment is: service, replacement, device, easy, help, and day.

The emotion analysis shows that the most frequent emotion keywords are trust and anticipation. The Affect Intensity Lexicon has different emotion features and it yielded in sadness, closely followed by joy as the most frequent emotions. The predominant dimension from the VAD lexicon in the data is valence, closely followed by dominance.



(a) Top 25 most frequent tokens from text responses.

(b) Frequency count of sentiment categories from VADER.

Figure 5: Emotion frequencies from Affect Intensity and VAD lexicons.

Token	negative	neutral	positive	Total
replac	8255	2835	10212	**21302**
servic	5318	1409	13704	**20431**
devic	6873	1977	7179	**16029**
day	5134	1871	5805	**12810**
time	4961	1225	4894	**11080**
receiv	4169	1510	5046	**10725**
new	3838	1312	5046	**10196**
insur	6166	764	3148	**10078**
call	5437	724	3339	**9500**
custom	3363	380	5690	**9433**
help	1714	202	6820	**8736**
process	2260	920	5428	**8608**
pay	5173	416	2243	**7832**
easi	337	86	7077	**7500**
work	2700	856	3292	**6848**
fast	407	769	4897	**6073**
thank	497	58	5159	**5714**
told	3549	400	1602	**5551**
back	2998	579	1892	**5469**
use	2660	477	1860	**4997**
great	282	29	4479	**4790**
file	2079	378	2326	**4783**
mobil	2577	309	1873	**4759**
sent	2689	514	1493	**4696**
need	2142	473	1972	**4587**
Total	**85578**	**20473**	**116476**	**222527**

Figure 6: Sentiment categories for Top 25 tokens from Question 2.

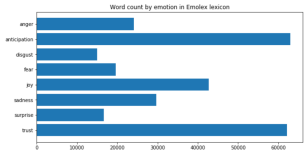
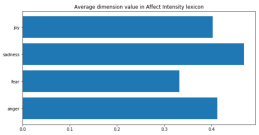
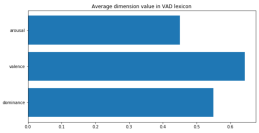


Figure 7: Emotion frequencies from Emolex in the data.



(a) Average emotion scores from Affect Intensity Lexicon in the data.



(b) Average emotion scores from VAD lexicon in the data.

Figure 8: Emotion frequencies from Affect Intensity and VAD lexicons.

3.4. Algorithms and Techniques

This section briefly describes the algorithms and techniques leveraged in this work: binary classifiers, text vectorization, embeddings, and deep learning approaches.

Logistic regression models the probability of the default class in a binary classification problem. It uses the

sigmoid (logistic) function to map the output of a linear equation to be between 0 and 1. The threshold to determine the predicted class is determined by the analyst. The logistic regression equation is shown below.

$$g(x) = \frac{1}{1 + \exp(-\theta^T x)}$$

A Support Vector Machine (SVM) is a classifier that finds a hyperplane which maximizes the margin between the two classes. The learning of the hyperplane in linear SVM is done by transforming the problem using linear algebra. SVM can efficiently perform a non-linear classification by implicitly mapping their inputs into high-dimensional feature spaces (also called kernel trick). Linear SVM can be rephrased using the inner product of any two given observations, rather than the observations themselves. The inner product between two vectors is the sum of the multiplication of each pair of input values. The reason for using SVM is its effectiveness in high dimensional spaces; it uses a subset of training points in the decision function making it memory efficient; and versatile in the sense that it can have different kernel functions specified.

XGBoost (eXtreme Gradient Boosting) is an implementation of gradient boosted decision trees that focuses on speed and performance [4]. Boosting is an ensemble technique to reduce bias by iteratively learning weak models and adding sequentially them to form a stronger model [5]. At each iteration, a new model is trained with respect to the error of the current ensemble. In gradient boosting, the gradient descent algorithm is used to minimize the loss function when adding new models to the ensemble [6]. The software library supports three variations of gradient boosting: gradient boosting, stochastic gradient boosting, and regularized gradient boosting. XGBoost is fast compared to other Random Forest implementations [7].

The survey responses' features were extracted by using different techniques. The simplest way to do this is to adopt the Bag-of-Words model (BoW). This framework disregards all order information in the words and only keeps the occurrence of words in a document [8]. By assigning each word a unique number, any document can be represented as a fixed-length vector with the length of the vocabulary of documents. The value in each vector position is a count of each word in the document. This method can be modified by changing the definition of a word or changing what to quantify about each word in the vector. In this research, a modified BoW model used word frequencies instead of using counts, because commonly used words will have large counts and misrepresent the data. The statistic used was TF-IDF which stands for term frequency-inverse document frequency, two components whose product result in the score assigned to each word. Term frequency indicates how often a word appears within a document. Inverse document frequency diminishes the weight of words that appear very frequently within a document set and increases the weight of terms that rarely occur [9]. A TF-IDF-based BoW model was expected to not be the most suitable approach to this problem, but it was still implemented to determine a benchmark for comparing performance of other solutions.

A more appropriate feature extraction technique is word embeddings, a type of word representation that allows words that have the same meaning to have a similar representation. Word2Vec provides a numeric representation for each word that captures different relations based on meaning, and is comprised of two algorithms: Continuous Bag-of-Words model (CBOW) and Skip-Gram model [10]. CBOW predicts a current word based on the previous surrounding words. Every word is mapped to a unique vector which is a column in a matrix W. The sum of the vectors is then used as features for prediction of the next word in a sentence. Conversely, the Skip-Gram algorithm uses one word to predict all surrounding words (context).

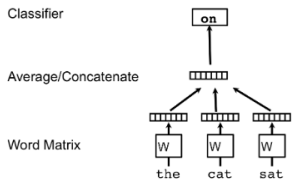


Figure 9: CBOW framework. The words the, cat, sat are used to predict the word on.

GloVe is a count-based method for learning distributional word representations. The main difference between this technique and Word2Vec is that the former considers the corpus' word occurrence statistics while also preserving semantic analogies through linear arithmetic [11]. The model is trained on the non-zero entries co-occurrence matrix from a corpus; the goal is to have the learned vectors' dot product equal the logarithm of the words' co-occurrence probability [11]. This also means that the ratios of co-occurrence probabilities are related to the corresponding vector differences in the word vector space because the logarithm of a ratio is the

same as a difference between logarithms. Pennington et al. showed that a ratio of co-occurrence probabilities possesses some meaning, which also gets encoded in the vector differences [11].

Since documents do not come in logical structures such as words and vary in length, a different method is needed for representing them numerically. Doc2Vec extends the Word2Vec model by mapping every paragraph to a unique vector which is represented by a column in matrix D and every word is also mapped to a unique vector, represented by a column in matrix W. The document-unique paragraph vector and word vectors are averaged to predict the next word in a context. Contexts are fixed length and sampling from a sliding window over a paragraph. Figure 8 simply describes this framework, called Distributed Memory Model of Paragraph Vectors (PV-DM).

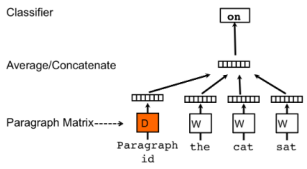


Figure 10: PV-DM framework. There is a paragraph token mapped to a vector via matrix D.

The paragraph token can be thought of as another word which acts as a memory that remembers what is missing from the current context (or the topic of the paragraph). Another algorithm, similar to skip-gram, may be used for representing documents: Distributed Bag of Words version of Paragraph Vector (PV-DBOW). In this method, the sliding window is eliminated, forcing the model to predict words randomly sampled from the paragraph in the output, getting rid of the sliding window.

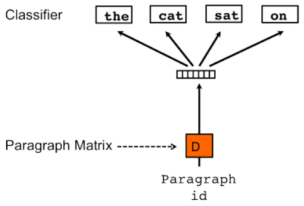


Figure 11: PV-DBOW framework. Paragraph vector is trained to predict the words in a small window.

PV-DBOW is faster because the word vectors do not have to be saved. Nevertheless, PV-DM should achieve better results. A Doc2Vec model based on PV-DM was trained on the text responses and the inferred vectors for the responses were used as features for a binary classifier. Text data is suitable for linear SVM classification because of the sparse, high-dimensional nature of text in which few features are irrelevant but tend to be correlated with other another [12]. Logistic regression, also a linear classifier, can produce results as effective as SVM, if tuned appropriately, for the feature space of text.

Pre-trained embeddings were incorporated as an embedding layer in neural networks for predicting attrition. Feed forward, convolutional, and recurrent neural networks were implemented.

The simplest neural network is to have a single neuron with an activation function which takes one input x and outputs a prediction. Complex neural networks take a neuron and stack them such that one neuron passes its output as in input into the next neuron. A set of input features (x_1, x_2, x_3) get connected to two neurons, called hidden units, because they produce intermediate features [13]. The word hidden in the term is due to the absence of ground truth values for the hidden units. The neural network determines three relevant features which can then produce a single output or prediction. The drawback is that usually these intermediate features that were discovered are difficult to for humans to interpret, as a common meaning is lost for complex networks [13]. The input features are called the input layer, which are connected to the hidden units, or hidden layer; and the output neuron is called the output layer.

Each hidden unit connected to the input layer outputs an activation value. The activation value of the first hidden unit in the first hidden layer would be denoted as $a_1^{[1]}$, as the input layer is layer 0. The general notation

for the input layer is as follows:

$$\begin{aligned} x_1 &= a_1^{[0]} \\ x_2 &= a_2^{[0]} \\ x_3 &= a_3^{[0]} \end{aligned} \tag{1}$$

Logistic regression can be represented as a single neuron. With an input of three features and the sigmoid activation function, the output layer is an estimated value of y. The logistic regression equation $g(x)$ is broken into two parts, $z = w^T x + b$ and $\sigma(z)$ where $\sigma(z) = \frac{1}{(1+e^{-z})}$.

In general, $a = g(z)$ where $g(z)$ is an activation function. Two other common activation functions are listed below.

$$\begin{aligned} g(z) &= \frac{1}{1+e^{-z}} \quad \text{sigmoid} \\ g(z) &= \max(z, 0) \quad \text{ReLU} \\ g(z) &= \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad \tanh \end{aligned} \tag{2}$$

A loss function L evaluates to what degree the actual outputs are correctly predicted by the model outputs. Cross-entropy loss is often used for binary classification. The weights in a neural network are updated on batches of data through backpropagation, which takes into accounts the actual and desired outputs [13]. The derivative of the loss with respect to each weight w is as follows:

$$\frac{\partial L(z,y)}{\partial w} = \frac{\partial L(z,y)}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w} \tag{3}$$

Forward propagation is performed with a batch of training examples to calculate the loss. The loss is then backpropagated to get the gradient (derivative) of the loss with respect to each weight, which are then used to update the weights of the neural network. The formula for updating weights is shown below.

$$w \leftarrow w - \alpha \frac{\partial L(z,y)}{\partial w} \tag{4}$$

Recurrent neural networks (RNNs) are a type neural networks that allow previous outputs to be used as inputs while having hidden states [14]. The loss function L is the summation of the loss at every time step.

$$L(\hat{y}, y) = \sum_{t=1}^{T_y} L(\hat{y}^{<t>}, y^{<t>}) \tag{5}$$

Backpropagation is performed at each time step T, where the derivative of the loss with respect to weight matrix W is the following:

$$\frac{\partial L^{(T)}}{\partial W} = \sum_{t=1}^{T} \frac{\partial L^{(T)}}{\partial W}\Big|_{(t)} \tag{6}$$

3.5. Feature extraction from text responses

A set of preliminary models using the primary dataset extracted from the data warehouse were created to get a sense of a minimum performance compared to the baseline. Since there is a significant class imbalance, several balancing techniques were tested: random under-sampling, random oversampling, and SMOTE. The next step was to continue with feature engineering to find additional features from the text responses and join the table with zip code-level demographics data.

First, the text responses had to be preprocessed. Contractions were expanded by matching on contractions using a regular expression pattern and conducting a replacement. The text was cleaned by removing punctuation, special characters, links, emoticons, and extra whitespace. Before vectorizing the text itself for features, it was used to derive a sentiment score and emotion features based on lexicons.

The clean text was fed to the VADER (Valence Aware Dictionary and sEntiment Reasoner) package, a lexicon-based sentiment analysis tool designed for social media data [15]. VADER works well with social media data and short text from other domains because it understands the sentiment of slang words, slang words as

modifiers, degree modifiers, emoticons, emojis, and acronyms. The output from VADER is a tuple of a positive, neutral, negative, and compound score. The compound score is calculated by adding the valence scores of each word in the lexicon, applying rule-based adjustments, and normalizing it to be between -1 and +1 [15]. Since the compound score is a weighted composite score, it was chosen as the single sentiment score.

In order to understand basic characteristics of the responses, the text had to be tokenized, which would also serve as a basis for text vectorization. The word_tokenizer method from the Natural Language Toolkit (NLTK) package was used, which tokenizes a string to split off punctuation other than periods [16]. NLTK's standard list of English stop words were used to remove them from the text, as well as a custom list based on the domain. For further standardization, NLTK's Porter stemmer was used for stemming the text, and this was kept as a separate field which would be used for preliminary text models.

Next, emotion analysis was conducted with the help of the following lexicons: NRC Emotion Lexicon; NRC Valence, Arousal, and Dominance lexicon; and NRC Emotion Intensity Lexicon. The NRC Emotion Lexicon, or Emolex, is a list of words and their associations with eight emotions: anger, fear, anticipation, trust, surprise, sadness, joy, and disgust [17]. Each word in the lexicon has an association flag (0 or 1) which indicates whether the target word has an association with one of the eight emotions, or affect category. The size of Emolex is 14,182 words, and Table 4 shows the frequency counts of each emotion within the lexicon. A function searches through Emolex to find word matches in the tokenized responses and counts how many times each emotion appears in each record.

Emotion	Frequency
fear	1476
anger	1247
trust	1231
sadness	1191
disgust	1058
anticipation	839
joy	689
surprise	534

Table 4: Frequency of each emotion in NRC Emotion Lexicon.

The NRC Valence, Arousal, and Dominance (VAD) Lexicon is a list of words and their valence, arousal, and dominance scores [18]. The scores range from 0 (lowest) to 1 (highest) for each dimension, and were obtained using best-worst scaling. The size of the lexicon is 20,007 words. Osgood et al. conducted a factor analysis to measure affective meanings of words [19]. This work laid the foundation for the three components of emotions (VAD). Valence is defined as the pleasantness of a stimulus; arousal is the intensity of emotion provoked; and dominance is the degree of control exerted by a stimulus [20]. The function searches through the NRC VAD lexicon in same manner as Emolex but takes the average of the score since it is a real number value. Table 5 shows words with the highest and lowest scores for each dimension that were present in the data.

Dimension	Word	Score	Word	Score
valence	love	1.000	corrupt	0.020
	happier	0.990	deceptive	0.027
	great	0.958	hate	0.031
arousal	furious	0.953	slow	0.073
	conflict	0.906	effortless	0.090
	awesome	0.897	smooth	0.127
dominance	success	0.981	poor	0.087
	knowledgable	0.971	mistake	0.102
	professional	0.943	idiot	0.113

Table 5: Words with highest and lowest VAD scores found in dataset.

The NRC Emotion Intensity Lexicon (Affect Intensity Lexicon) is a list of 6,000 words and their intensity scores for four emotions: anger, fear, sadness, and joy [21]. The intensity scores of association were obtained using best-worst scaling during annotation. The score ranges from 0 (lowest) to 1 (highest). The score for each emotion was aggregated by taking the average of each token found in the record.

The approach to vectorizing the text responses was to start with the Bag of Words (BoW) model and follow with TF-IDF weights. Unigrams, bigrams, trigrams, and skip-grams were tested with both BoW and TF-IDF models. Sci-kit learn does not have a skip-gram implementation, so a custom analyzer was created that generates k-skip n-grams. The resulting tokens can then be passed to the Count Vectorizer for BoW or TF-IDF Vectorizer. The model pipeline consists of the vectorizer (count or TF-IDF), dimensionality reduction, a balancing method, and a binary classifier.

3.6. Structured data model using additional features

The *dataframes* resulting from sentiment and emotion analyses were merged together so that each text record contained a sentiment score and emotion features.

After testing a preliminary model with minimal feature engineering, a next round of testing occurred with the additional sentiment, emotion, and zip code-level demographics features. The demographics dataset was extracted from the Assurant data warehouse and was merged with the primary data via a left join on the customer zip code. The models built were compared across the 3 different target variables (two month, three month, and overall attrition). The pipeline included feature scaling, imputation for missing values, feature selection, a balancing technique, and the classifier. SVM, XGB, Logistic Regression, and Random Forest classifiers were tested.

3.7. Deep learning approach with pre-trained embeddings

A pre-trained Word2Vec embedding from Google was used in an simple artificial neural network. The same procedure was compared to results from a GLOVE pre-trained embedding. The ANN was comprised of the following layers: input, embedding layer, dropout, 512-neuron hidden layer with a ReLu activation, and a single output with a sigmoid activation function. The loss function was binary cross entropy and Adam optimizer. These did not end up generating good results so the next logical step was to train a word embedding using the available domain data.

	Precision	Recall	Specificity	F1	Geometric Mean
Total	0.69	0.51	0.51	0.56	0.51

Table 6

3.8. Deep learning approach with training own word embedding

A Word2Vec model was trained on text responses using the Gensim software library [22]. The vector size was set to 300 (had also tried with 100) and the size of the vocabulary was 10,322 terms. The weight matrix from the model was fed to a neural network of a similar architecture described in the previous section. A Doc2Vec model was also experimented with, but it yielded very close results but with a much longer training time required. The terms which are specially salient between the two different classes having high TF-IDF scores were used as features in an SVM model along with structured features. A feature importance analysis was conducted for the combined model. Scikit-Learn's forests of trees was used to evaluate feature importance.

3.9. Sequential classifier for attrition

A recurrent neural network was used to model attrition from customer-level claim events. The data was a multiple input series where each row is a time step and containing multiple inputs and one output parallel series. For a LSTM model, the data was transformed into input/output samples, based on a fixed time step size. As a result some data had to be discarded which didn't fit the time step size. The transformed input data is in a three-dimensional structure (number of samples, number of time steps per sample, and number of variables). A simple LSTM architecture was used (input, LSTM layer, output) with a ReLu activation, Adam optimizer and binary cross-entropy loss.

4. Results

This section provides a summary of the results across the different techniques, as well as the results from the feature importance analysis.

4.1. Evaluating Success

The data is inherently imbalanced with a 82.3/17.7 split, with minority class being customers who ended their subscription. The metrics to consider need to be adequate for imbalanced data and the problem at hand. The following metrics were considered: recall, F1, and geometric mean. The baselines were derived from making predictions n at random; the F1 score becomes 0.7559 and recall is 0.6075. Other similar research was explored to see what their results were. Other attrition models for insurance data had F1 scores of 0.55 (AdaBoost) and 0.81(LSTM).

4.2. Models

Structured data model, no feature engineering:

	Precision	Recall	Specificity	F1	Geometric Mean	IBA	Support
0	0.84	0.69	0.56	0.76	0.62	0.39	19288
1	0.35	0.56	0.69	0.43	0.62	0.38	5829
avg / total	0.73	0.66	0.59	0.68	0.62	0.39	25117

Table 7

TF-IDF model:

	Precision	Recall	Specificity	F1	Geometric Mean	IBA	Support
0	0.79	0.75	0.32	0.77	0.49	0.25	19288
1	0.28	0.32	0.75	0.30	0.49	0.23	5829
avg / total	0.67	0.65	0.42	0.66	0.49	0.25	25117

Table 8

Deep learning, embedding trained on domain data:

	Precision	Recall	Specificity	F1	Geometric Mean
Total	0.72	0.71	0.40	0.72	0.48

Table 9: Deep learning model with embedding layer trained on domain data.

Combined model of structured data and features from text:

	Precision	Recall	Specificity	F1	Geometric Mean
Total	0.75	0.73	0.45	0.79	0.53

Table 10: Combined model of structured data and features from text.

5. Related Work

5.1. Brief Historical Outline

In 2005, Hur and Lim built a support vector machine (SVM) model to predict customer churn for an online car insurance service. At this time, the increase of people buying car insurance online served as motivation to study the online market, because online quotes enabled users to switch insurers with more ease. The balanced dataset contained 13,200 records with 25 attributes from an online car insurance quote service. The target variable is binary, with a value of 1 if a customer switches its insurance company next year, otherwise 0, if the customer did not switch. The SVM model using a Gaussian radial basis function was compared with a neural network and logistic regression; and their performance evaluated using accuracy. The SVM yielded in the highest accuracy on both the training and test sets, with 71% and 69%, respectively [23].

In 2008, Xia and Jin sought to create a model for customer churn in telecommunication carriers with greater predictive power. The authors argue that traditional classification methods cannot generalize well enough for large, high-dimensional, non-linear, or time series data. Although ANN and self-organizing maps (SOM) can be robust and enough for this kind of data, they are based on empirical risk minimization which results in low

generalization. To address this, the authors used a SVM with structural risk minimization and evaluated it using accuracy, hit rate, coverage, and lift [24]. Also around the same period, Coussement and Van den Poel conducted a study which used a support vector machine model to predict churn for a newspaper subscription service. The data originated from a Belgian newspaper publishing company which offers a subscription service whose price depends on subscription length and a promotional offer. Since a customer must stay enrolled for a certain amount of time before they can cancel (after the maturity date), the objective is to predict whether the subscription will be renewed within a period of four weeks after the maturity date. The independent variables include information about client-company interactions, renewal data, demographics, and subscription details; which were aggregated for a 30-month period at the subscriber and at the subscription level. The subscription level contains data from the current subscription. The subscriber level contains data from the customer; a customer may have multiple subscriptions. The balanced data used to train the model was cleaned to be at the subscription level where each observation represents a renewal point. They compared two techniques for parameter selection, which are based on grid search and cross-validation. The two SVMs were denoted SVM_acc and SVM_auc. The models are compared by considering the following metrics: AUC, percent correctly classified (PCC), and top-decile lift. The SVM_auc model performed significantly better than SVM_acc in terms of AUC for test sets which have less than or equal to 30% churners (including the real distribution of 11.14%). Also, SVM_auc performed better at the real distribution in terms of PCC. Plus, SVM_auc has a higher top-docile lift across the board. The study showed that SVMs can generalize well in the context of noisy marketing data [25]. When using the best parameter selection method, the SVM performed better than logistic regression. Yet, it did not perform better than a random forest in either case. Nevertheless, the most important variables that explain churn in this context are those related to interactions between the client and company as well as subscription features. It was also determined that monetary value and frequency are not significant in modeling churn.

In 2012, Yaxi Xu sought to address the class imbalance challenge present when creating churn models. Xu built an extended one-class SVM model to predict customer churn and compared this with a regular SVM, ANN, and a decision tree. One-class classification (OCC) is a way to deal with imbalanced datasets. In OCC, the model learns from a training set which contains only the object of a specific class, so that it can distinguish observations of this class amongst all observations [26]. So, in the case of churn, a model was trained only with observations of the majority, non-churners. Implementing OCC using a SVM involves finding the smallest hypersphere which contains most of the training data, also called Support Vector Data Description (SVDD). The Small Sphere and Large Margin (SSLM) method combines SVDD and the regular, binary SVM together. Its objective is to create a hypersphere which contains most of the examples belonging to the majority class (normal examples) whose volume is minimized and simultaneously maximize the margin between the sphere surface and minority class examples [26]. Both SVDD and SSLM involve creating a small hypersphere around the normal or positive data and could use the negative data to refine the classification boundary. The main difference is that SSLM maximizes the margin the margin between the sphere and the negative data while SVDD needs most of the negative data to be outside of the sphere region. Margin maximization allows SSLM to generalize better than SVDD [26]. In this experiment, customers who churned represent 14.3% of the data and were referred to as positive examples. The data was split into training and test set at a 60/40 split. The SSLM approach was chosen and its predictive power was evaluated by comparing several model techniques: ANN, decision tree, and SVM. The parameters for SVM were selected using a grid search and cross validation. The metrics for evaluating model performance were hit rate, coverage rate, and lift coefficient. When evaluating the models based on lift, coverage, and hit rate, the extended one-class SVM performed the best.

In 2014, Farquad et al. published a study which sought to address a disadvantage in SVM models: its inability to explain how it learns the training data in a way humans can understand. Rule extraction is a method to try to translate black box models into a comprehensible one.The authors established a hybrid approach to rule extraction from a SVM model for a customer relationship management (CRM) application. There are three components to this approach: feature set reduction through SVM-recursive feature elimination (SVM-RFE); training of SVM model and support vector extraction; and rule generation through a Nave Bayes Tree (NBTree) [27].

5.2. Challenges with attrition modeling

5.2.1. Characterizing an attrition event

The appropriate way to define an attrition/churn event or label a customer as churned depends on the type of data being considered. For instance, a churn event for a company selling subscriptions could be when the subscription is cancelled by the customer or by the company. A churn event for a non-subscription based business could be defined in several different ways. In the case of an e-commerce site, a customer might make

a purchase at any time. One approach is to place a time interval on the event, so someone churned if they did not make a purchase within the last set number of days.

The attrition (churn) rate represents the percentage of customers lost by a business out of its entire customer base. However, when considering customers who churned based on a time period, the customer base size is not simple to calculate as it is not static. This uncertainty affects how to use and interpret churn metrics/indicators. The smaller a company's customer base, the more volatile a monthly churn rate will appear. For a large customer base, quarterly and annual churn rates can be different based on how churn is calculated (time period) and the variance of the true churn rate across different time periods.

There are different types of attrition what could be modeled. Customers might abandon a business voluntarily or involuntarily. The latter might be encountered when a customer is not paying bills or is engaged in fraudulent activity. When it is inherent to a business policy, it could be more adequate to have a model specific to involuntary attrition.

5.2.2. Data challenges

It is expected for an organization to have a low attrition rate which results in the class imbalance issue for developing a model. In this case, the minority class is churned customers and the majority is non-churn customers. This causes poor performance as a classifier will try to maximize accuracy by predicting all observations to belong to the majority class. The accuracy might seem to look good but precision and recall are not. As previously mentioned, there are several techniques to handle imbalanced data which were performed in this work.

If the time period is increased enough or toward infinity, all customers will have churned and all the target labels will be 1. Hence, at the point in time that customers are still considered non-churned are still partially observed because they haven't left yet. In other words, their churn event is censored [28]. Churn event censorship is a challenge for machine learning methods that require dataset labels to be fully observable [28]. Another technique to employ is survival models.

When collecting data, one can use either aggregated customer-level metrics or transaction level, time series data. Aggregate features are usually easier to extract. However, using the average to aggregate data means treating each occurrence equally, which is usually inadequate, because more recent transactions closer to the churn event should have a larger weight than much older transactions. An alternative to taking the average is to apply an exponential moving average which assigns a higher weight to more recent transactions.

Once in production, one must constantly monitor the model's performance over time and re-train or complete additional development if necessary. This is due to the concept drift phenomenon. Concept drift is the idea that patterns in data for certain applications evolve over time, causing machine learning models to become obsolete [29]. An attrition model that works well today could become useless in the future due to changes in customer behavior which drive attrition.

6. Contributions

There is currently no predictive modeling being done with respect to attrition in the Connected Living group of Assurant. Attrition is a metric that is tracked and reported on, but there is no scoring of customers for attrition likelihood. Several models were tested and ultimately have a combined model that does not have a state of the art F1 score but is better than having nothing or choosing a random. The contributions of this research are a developed model and important features which can be used to make recommendations to the claims operations and customer experience teams on how to improve and prevent attrition.

References

[1] *Assurant - media resources [online]. https://www.assurant.com/news-insight/media-resources.*

[2] *D. L. Garcia, A. Nebot, A. Vellido, Intelligent data analysis approaches to churn as a business problem: a survey, Knowledge and Information Systems - Springer 51 (3) (2017) 719–774. doi:10.1007/s10115-016-0995-z.*

[3] *M. Barnett. mrab-regex [online]. https://pypi.org/project/regex/.*

[4] *T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, ACM, New York, NY, USA, 2016, pp. 785–794. doi:10.1145/2939672.2939785.*
URL `http://doi.acm.org/10.1145/2939672.2939785`

[5] *Y. Freund, R. E. Schapire, A short introduction to boosting, Journal of Japanese Society for Artificial Intelligence 14 (5) (1999) 771–780.*

[6] A. Natekin, A. Knoll, Gradient boosting machines, a tutorial, Frontiers in Neurorobotics 7 (2013) 21. doi:10.3389/fnbot.2013.00021.
URL https://www.frontiersin.org/article/10.3389/fnbot.2013.00021
[7] S. Pafka. Benchmarking random forest implementations [online]. http://datascience.la/benchmarking-random-forest-implementations/.
[8] Z. S. Harris, Distributional structure, Word 10 (2–3) (1954) 146–162.
[9] J. Ramos, et al., Using tf-idf to determine word relevance in document queries, in: Proceedings of the first instructional conference on machine learning, Vol. 242, Piscataway, NJ, 2003, pp. 133–142.
[10] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, 2014, pp. 1188–1196.
[11] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.
[12] C. C. Aggarwal, C. Zhai, Mining text data, Springer Science & Business Media, 2012.
[13] A. Ng, K. Katanforoosh, Cs229 lecture notes - deep learning (2018).
URL http://cs229.stanford.edu/notes/cs229-notes-deep_learning.pdf
[14] S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent convolutional neural networks for text classification, in: Twenty-ninth AAAI conference on artificial intelligence, 2015.
[15] C. J. Hutto, E. Gilbert, Vader: A parsimonious rule-based model for sentiment analysis of social media text, in: Eighth international AAAI conference on weblogs and social media, 2014.
[16] S. Bird, E. Klein, E. Loper, Natural language processing with Python: analyzing text with the natural language toolkit, " O'Reilly Media, Inc.", 2009.
[17] S. M. Mohammad, P. D. Turney, Crowdsourcing a word-emotion association lexicon 29 (3) (2013) 436–465.
[18] S. M. Mohammad, Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words, in: Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL), Melbourne, Australia, 2018.
[19] C. E. Osgood, G. J. Suci, P. H. Tannenbaum, The measurement of meaning, no. 47, University of Illinois press, 1957.
[20] A. B. Warriner, V. Kuperman, M. Brysbaert, Norms of valence, arousal, and dominance for 13,915 english lemmas, Behavior research methods 45 (4) (2013) 1191–1207.
[21] S. M. Mohammad, Word affect intensities, in: Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018), Miyazaki, Japan, 2018.
[22] R. Řehřek, P. Sojka, Gensimstatistical semantics in python, statistical semantics; gensim; Python; LDA; SVD.
[23] Y. Hur, S. Lim, Customer churning prediction using support vector machines in online auto insurance service, in: International Symposium on Neural Networks, Springer, 2005, pp. 928–933.
[24] G.-e. Xia, W.-d. Jin, Model of customer churn prediction on support vector machine, Systems Engineering-Theory & Practice 28 (1) (2008) 71–77.
[25] K. Coussement, D. Van den Poel, Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques, Expert systems with applications 34 (1) (2008) 313–327.
[26] Y. Xu, Predicting customer churn with extended one-class support vector machine, in: 2012 8th International Conference on Natural Computation, IEEE, 2012, pp. 97–100.
[27] M. A. H. Farquad, V. Ravi, S. B. Raju, Churn prediction using comprehensible support vector machine: An analytical crm application, Applied Soft Computing 19 (2014) 31–40.
[28] B. Khaleghi. What makes predicting customer churn a challenge? [online]. https://medium.com/@b.khaleghi/what-makes-predicting-customer-churn-a-challenge-be195f35366e.
[29] I. Žliobaitė, M. Pechenizkiy, J. Gama, An overview of concept drift applications, in: Big data analysis: new algorithms for a new society, Springer, 2016, pp. 91–114.