CASE STUDY: LEARNING TO ROUTE EMAIL TO IMPROVE EFFICIENCY IN A CONTACT CENTRE

Philip McCann

MSc. in Computer Science (Data Analytics)

College of Engineering and Informatics

National University of Ireland, Galway

Supervisor: Dr. Lukasz Porwol

This dissertation is for the degree of Master of Science

September 2018

Declaration

I hereby certify that this material, which I now submit for assessment on the program of study leading to the award of Masters in Science is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: ____________________ Date: ____________________

Acknowledgements

I would like to thank my project supervisor, Lukasz Porwol for his positive attitude, expert guidance and motivation when it was most needed.

I would also like to thank my colleagues at Avaya for allowing me the time to study for this course and to complete this research project. Thanks to Richard Simpson, Dave O'Connell, Tom O'Rourke and Joe Flaherty. I am so very grateful to Jim and Tracy for their cooperation and generous offers of support.

A special thanks to my family; Fiona, Lauren, Gabriel and Hugh for their constant support and encouragement.

Abstract

The modern contact centre is a hub of communications between business and customers, connecting over multiple channels to facilitate sales and support to deliver the best customer service. An efficient contact centre will maximize the performance of the agents to deliver a prompt and informed service and that depends on software applications to connect the customer to the best available agent in a timely manner.

The Avaya contact centre collects and stores a lot of data that has not been fully utilized to improve the features and performance of the applications. With access to historical data from a customer site, this research takes the opportunity to introduce data mining and machine learning techniques to email routing to improve the performance against some key metrics; agent occupancy rate, cost per call and service level response times.

Contents

Figures

Tables

List of Abbreviations

Abbreviation	**Description**
AAAD	Avaya Aura® Agent Desktop
AACC	Avaya Aura® Contact Centre
ACD	Automatic Call Distribution
API	Application Programming Interface
ARFF	Attribute-Relation File Format used by WEKA
AUROC	Area under ROC curve
CC	Carbon Copy
CCMA	Contact Centre Manager Administration component of AACC
CCMM	Contact Centre Multimedia component of AACC
CCMS	Contact Centre Manager Server component of AACC
CCT	Communications Control Toolkit component of AACC
CMF	Contact Management Framework component of AACC
EP&T	Emerging Products and Technology
FN	False Negative
FP	False Positive
IBk	Instance-Based learning with parameter k (WEKA kNN)
IDE	Integrated Development Environment
IDF	Inverse Document Frequency
IIS	Internet Information Services
IP	Internet Protocol
IVR	Interactive Voice Response
kNN	K Nearest Neighbours
ML	Machine Learning
NB	Naïve Bayes
ROC	Receiver Operating Characteristic
SMO	Sequential Minimal Optimization (WEKA SVM)
SQL	Structured Query Language
SVM	Support Vector Machine
TF	Term Frequency
TN	True Negative
TP	True Positive
URL	Uniform Resource Locator
WEKA	Waikato Environment for Knowledge Analysis

1 Introduction

The modern contact centre is a hub of communications between business and customers, connecting over multiple channels to facilitate sales and support to deliver the best customer service. An efficient contact centre will maximize the performance of the agents to deliver a prompt and informed service and that depends on software applications to connect the customer to the best available agent in a timely manner.

Avaya in Galway is a research and development centre specializing in software for contact centre, with the Avaya Aura® Contact Centre (AACC) and Avaya Oceana™ products designed and developed mainly from the Galway hub. AACC is a mature product with a large installed customer base. To serve those customers there is a continuous cycle delivering feature and quality improvements in major software and feature pack releases. Each improvement in the feature set adds competitive value to Avaya to differentiate from other similar products in the marketplace. For AACC customers, they get the benefit of the new features to be able to run an efficient contact centre and deliver a high standard of service to keep their customers satisfied in turn.

There is a lot of data collected in the contact centre; call recordings, transcripts, customer histories and agent reports [1], and that constitutes a valuable resource that can enable the next generation of features and improvements in the software. These features will help to better the customer experience and improve contact centre efficiency. Research and development is being undertaken in areas of service automation, sentiment analysis, speech analytics, customer journey developments and more [2] [3].

This work looks to make improvements in the contact centre against key performance metrics by changing how incoming emails are routed. Some emails should be automatically closed with no agent interaction, others need to be routed to the person in the organization best equipped to handle it. Data mining and machine learning tools and techniques will discover properties in the data history that can inform a better initial routing for new contacts arriving into the contact centre. The goals therefore are to use the contact history from an AACC system to train a series of ensemble classifiers to categorize incoming emails in two ways:

- Identify contacts that should not enter the contact centre. When emails become contacts, they consume server resources throughout the lifecycle of queuing, assignment and storage. These can often be spam that have escaped external filters or other automated replies or delivery failure messages.
- Identify valid contacts that do not need agent interaction. A customer will often reply in an email conversation with an acknowledgment or follow up that should go on the record but does not need any interaction with an agent. These contacts should be queued with a lower priority so that agents can attend to the more important, service level impacting work items in a timely fashion.

Either one of these will contribute an improvement in the contact centre efficiency and service level performance and both combined will make an improvement in staff and resource utilization from the better initial assignment.

1.1 Motivation

This project was conceived in the AACC research and development team with the cooperation of a customer and partner that participate regularly in our software trials and provide excellent feedback on features and roadmap. With an understanding of their business processes as used in a busy contact centre, and with access to anonymized historical data from the site, the opportunity was to use the site as a case study to demonstrate how data mining and machine learning techniques can be applied to a real-world business case.

The site is a multi-channel contact centre handling voice calls, email, web chat, fax and scanned document contact types, but mostly voice and email. Spam, defined as unsolicited bulk email [4], is a big problem in any email system, and commercial spam filters do a very good job in preventing unwanted content from getting to the contact centre mailboxes. However, this is an email intensive business, and a small percentage getting through still presents a significant problem to be managed. As well as spam there are other classes of unwanted email, such as out-of-office replies, delivery failure loop backs and invitations from social media sites such as LinkedIn. When these contacts are handled at the site they are closed off as “Spam” although that is not the strict definition which should be both unsolicited and bulk. The chart in Figure 1.1 compares the incoming email contact rate with those identified as spam. The peaks and troughs show a drop off in activity at the weekend, but the rate of spam is consistent at 5% of the total.

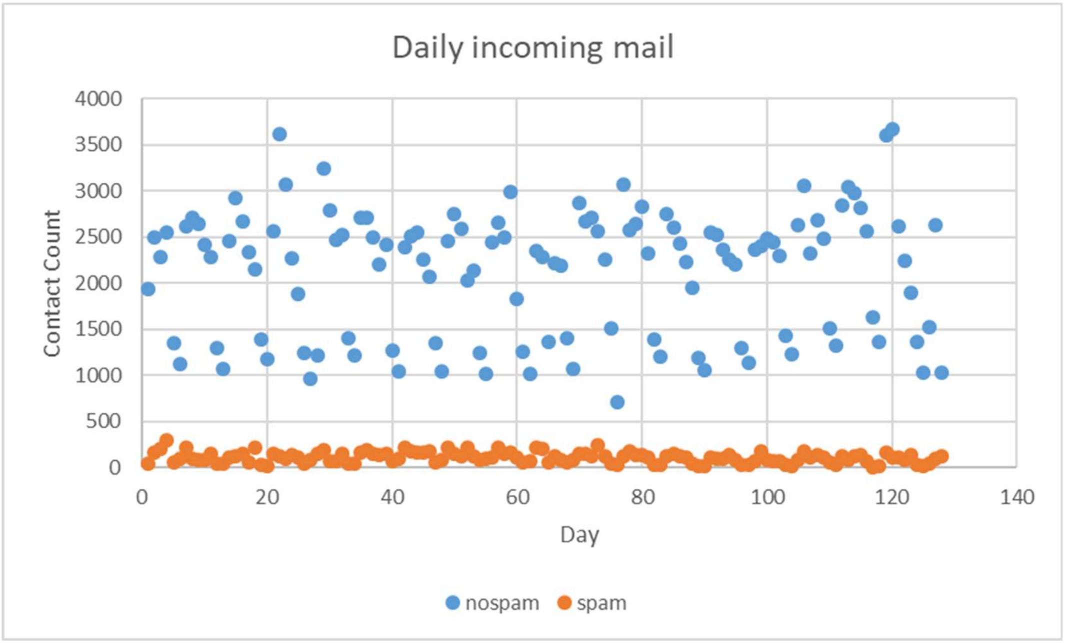


Figure 1.1. Incoming daily rates of email classed as spam

Over a 4-month period, the site received 14125 such emails which accounted for 160 hours just to identify that they were unwanted. Preventing these emails from getting routed to agents is a supervised learning task to use the features from the historical data tagged by the agents to classify future contacts and close them automatically.

Another change that our customers could benefit from, is to improve the handling of emails that come in to the contact centre but do not need any further action from the agents. These are typically the acknowledgement of a previous reply, the "ok, thanks" type of response, and while they should still be routed to the contact centre and added to the customer record, it should be with a lower priority to make sure that agents are giving focus to more urgent activity. Again, this is a classification task based on the historical evidence from how earlier contacts were handled. Figure 1.2 charts the daily reply rates for email contacts and shows that about two thirds of contacts are replied to.

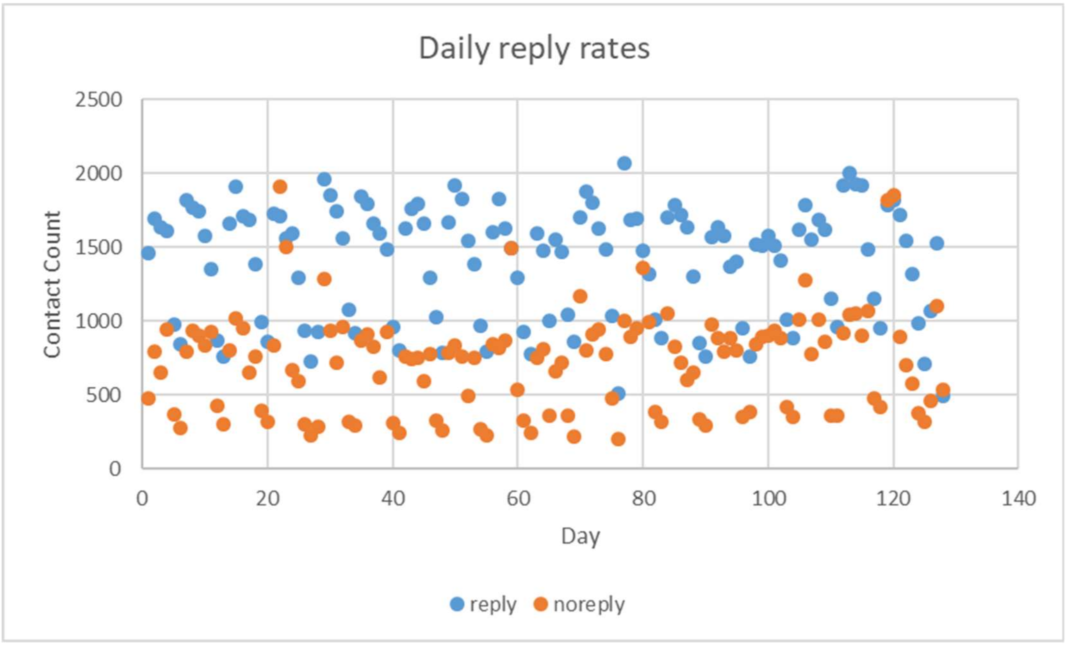


Figure 1.2. Daily reply rates (excluding spam)

1.2 Objectives

The objectives therefore are:

- To demonstrate how the email routing in AACC can be improved with an innovative, machine learning based approach.

- To analyze the historical data from a customer site, identify and extract the features from the email history, and determine the best algorithms, and combinations of algorithms to use for the selected classification tasks; spam detection and reply prediction.
- To implement and deploy the trained models in a lab environment using the AACC open architecture that will route new email based on the model predictions. This deployment template will show how the benefits of the machine learning can be delivered easily to the AACC installed base without a need to reengineer the solution.

1.3 Layout

This thesis is structured in 6 chapters. Chapter 1 is an introduction to the research area and sets out the objectives. Chapter 2 is a review of the current understanding of the relevant areas of text classification, machine learning and email routing. Chapter 3 describes the design of the classifier followed in chapter 4 by the description of the implementation and experimentation. The results of the experiments are discussed in chapter 5. Finally, in chapter 6 are the conclusions and suggestions for future work.

2 Literature Review

2.1 Overview

This chapter describes the state of art for handling email in the contact centre, and the data mining and machine learning techniques that can improve it. It will describe the commonly used text classification algorithms to determine how they might be useful in classifying email in the contact centre domain.

2.2 Contact Centre

In the past, sales and support activities were voice based and conducted by making and receiving large numbers of phone calls. The development of automatic call distribution (ACD) systems to manage large volumes of phone calls made it possible to establish an environment where these calls could be handled efficiently; the call center. Further advances in voice technology led to interactive voice response (IVR) systems to collect information from the caller where they could then be served with a recorded message or connected to an agent with the knowledge to answer the query. Fast-forward to today and the rise of digital communication is much more than just phone calls. When we pick up a phone today, it is more likely to send an email, text or instant message than to make a voice call. The call center has moved on too, to handle interactions on all these digital channels. The "call" is now a "contact" on any voice or multimedia channel, and it is the contacts that must be routed a suitable agent in the contact centre.

2.3 Email Routing

In the modern contact centre, and specifically in AACC, the process to route contacts to agents is based on matching the contents of the incoming email with the skills of the agent. Agents that have been trained and have expertise in certain areas are assigned to groups, called skillsets, that align with their expertise. When an email arrives into the contact centre it is analyzed for words or phrases and assigned to a skillset and routed onward to an agent. In AACC, the "MailTo" address of the incoming email might represent a mailbox or an alias for a mailbox. The mailboxes are polled regularly to retrieve the messages and then a set of system and custom routing rules are applied to the incoming email. First, a system level rule to detect if the mail is a delivery failure message related to previously sent outgoing mail, then a system level rule to detect for prohibited words or phrases, and finally customized rules based on the address the mail was sent to. The custom rules can be actioned on the sender address or on keyword and partial keyword matching. The custom group of rules can be applied on a best match or first match basis to prioritize based on the score or the rule. When the routing rule has been determined then the skillset and priority is known, and the email is progressed through the contact centre to be delivered to an agent with the appropriate skills [5].

When an agent is dealing with a contact they have the option to resolve it with a templated or freeform text reply to the customer. When complete, they must wrap-up the contact and set a closed reason. If they are not able to complete the task, then they can transfer to another skillset where an agent with the correct skills can pick up and complete the task. Analysis of these transfer patterns could form the basis of further improvements discussed later as future work.

The closed reason is an indicator of the resolution status; for example, Sale or No Sale to capture useful information to report on later. It is often used to capture if the email contact was spam that escaped an external spam filter or custom spam rule in the system. For this research, it provides a supervised sample of data identified as spam for this specific contact centre domain.

Email routing is comparable from each of the companies leading the way in contact centre. Avaya, Genesys, Cisco and Aspect offer the same screening rules and keyword matching albeit with some differences in terminology [6] [7] [8]. Genesys also has an additional and separately licensed Content Analyzer that allows email categorization based on the word vectors from the email text [9]. It does not allow additional features to be derived so models will lack accuracy compared to training with the full scope to define extra features. This platform uses quite technical language in the applications; training, testing, cross-validation, and although it is a good introduction to bringing machine learning to the product, it cannot fully match a dedicated machine learning platform like WEKA, Scikit-learn or R. For specific learning problems, a better approach is to conduct the training in an expert and fully featured machine learning environment and then use the integration points in the applications to benefit from the results. To learn from the features of the historic email corpus and make predictions about future email contacts is a text classification task.

2.4 Text Classification

The goal of text classification is to group similar documents into categories by analyzing the word features in the text. In the book Mining Text Content [10], Aggarwal devotes a chapter to survey text classification algorithms and identifies Decision Trees, Support Vector Machines (SVM), Nearest Neighbour and Bayesian classifiers as methods commonly used in text classification. These are the easier classifiers to understand and produce hypotheses that are easiest to interpret. To apply machine learning to any problem, it is best to start simple. Ideally, the classifier will produce a hypothesis that has a high success rate and is easy to understand. The best results can come from an ensemble of independent, better than random classifiers. Different classifiers can be prone to overfitting and errors, so combining the results of a selection of different classifiers in an ensemble, will improve the performance.

The text classification process is consistent across the literature and follows a well-defined process. The training data corpus is a collection of documents and the words in the corpus form the vocabulary. Each document is read and prepared by tokenizing the document into an array of words. The words can be stemmed to common roots and stopwords can be removed. Stopwords (such as “a”, “the”, "of") that appear in all the documents do not contribute to learning and are best to be removed to reduce the feature dimensionality and focus on the important topic related words. The remaining text is then represented in a vector space model mapping the normalized word counts to the documents in which they occur. When the documents in the vector space have already been categorized, then the bag-of-words features represented in the vector space can be used in supervised learning [11] [12] [13] [14].

2.4.1 Naive Bayes

Naive Bayes is a simple supervised classifier that assigns a classification to a sample based on the occurrences of the sample words in the training data. It is based on Bayes' rule that can predict the probability of a class given an observation based on previous, prior observed probabilities of the features occurring in the data.

$$P(B \mid A) = P(A \mid B)\, P(B) / P(A)$$

A Naïve Bayes classifier calculates the probability for each possible class given the feature evidence, and the class that maximizes that probability is the most likely and is the predicted outcome. The predicted class *c* with possible values in *N*, can be expressed formally therefore in terms of the feature word vector ***w*,** as follows [15]:

$$c = \text{argmax}_{c \in N}\, P(c)\, P(\mathbf{w} \mid c)$$

2.4.2 Decision Tree

A decision tree is a readable, easy to understand hypothesis used to make predictions for unseen samples. J48 is the WEKA implementation of Ross Quinlan's C4.5 decision tree algorithm. It is a supervised learning algorithm that generates a decision tree from a set of categorized examples. The information gain is calculated for each of the features in the training set and the feature that yields the highest information gain, the most informative, is added as a decision node on the tree. A branch is added from the decision node to represent each possible outcome. The information gain calculation is repeated recursively for the subset of data partitioned by the branch to create more decision nodes based on the most informative features. If all the samples in a subset return the same classification, then that is the decision. If all attributes have been tested, then the majority class of the remaining samples is the decision [16].

2.4.3 K-Nearest Neighbours

K-Nearest Neighbours (kNN) is a simple instance-based learning algorithm that assigns a class to a sample by comparing it to the *k* most similar training samples and assigning the majority class. It is a lazy algorithm and represents each training sample as a feature vector. The sample case is compared to all the training samples by measuring the Euclidean or other distance and the majority vote from the *k* closest is assigned to the sample [17]. The WEKA implementation of the kNN algorithm is called IBk (instance-based learning with parameter *k*). The example below shows how the yellow star sample would be classed as a blue circle using 3 neighbours on a two-to-one majority vote.

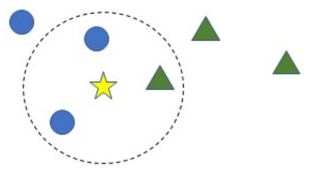


Figure 2.1. Example of classification with kNN

2.4.4 Support Vector Machine

The support vector machine, SVM, is a classifier that tries to find a decision plane to best separate the training samples into two classes. When the data is not linearly separable, it is transformed to a new feature space using a kernel function where it does become linearly separable. SVM works well with high dimensional data so performs well for text classification tasks. The WEKA implementation of the SVM algorithm is called SMO, Sequential Minimal Optimization. This is the name of the algorithm used to solve the quadratic programming problem to find the hyperplane to maximize the separation [18] [19].

2.5 Spam Filtering

Spam makes up almost half of all email traffic, and it used to be much more (in 2008 it was 90%) [20]. Every company should have a spam filter in place and especially in the contact centre scenario that must handle large volumes of email. Spam filters prevent harmful emails getting to the contact centre and to end user inboxes and preserve the company resources for authentic content. A product like SpamAssassin will perform hundreds of tests on the header and body of the emails, and will combine scores from whitelists, blacklists and collaborative databases to identify and block the delivery of spam email [21]. Spam filters can be 99.9% effective now when deployed and configured properly.

In the customer data that is the focus of this research, the definition of spam is modified to include unwanted emails such as spam filter escapes, out-of-office replies, invitations and delivery failures. Similar features from these labelled instances in the contact centre history will be used to filter out these emails in future so that they are not routed onwards to agents.

2.6 Reply Prediction

Much of the literature on email reply prediction seems to deal with end user email inbox management [22] [23]. In the contact centre, the same learning will be applied to modify the routing to prioritize those emails where the customer needs a reply. All email contacts will get routed to agents but the queuing in AACC will ensure the higher priority items route first, so that replies are sent out in a timely manner to achieve better service levels.

2.7 Feature Selection

Each supervised training instance contains the text properties of the email; sender, recipients, subject and email body. The words contained in these fields are added to a vector space model with the associated classification. From this bag-of-words, the term frequency and inverse document frequency, TF-IDF, will be a measure of the importance of the words in the corpus. The models are not only trained on the bag-of-words, but also using meaningful features extracted from the context of the email.

On the question of identifying spam email, these additional features are properties like the time of day the mail arrives, or the number of recipients, or whether this email a follow-up in a conversation. These properties need to be extracted from the context and added as extra features in the training instances [24].

On the question of predicting if an email needs a reply, the most recent message is the email chain is important, so the earlier embedded conversation should be removed from the training data. In the paper "Email Reply Prediction: A Machine Learning Approach" the authors identified the use of interrogative words, previous communication from the same sender as well as the bag-of-words as features that contribute to a prediction score [22].

2.8 Technical Environment

The tools and applications used to conduct this research were in part driven by the existing technology in the AACC solution. The Intersystems Caché database and Microsoft Visual Studio .NET development environment allowed for easier integration with the database and the open interfaces. Additional modelling and analysis was done in WEKA due to the extensive range of machine learning algorithms and user-friendly interface. Further drilldown analysis and preparation of charts for visualization were compiled in Microsoft Excel.

2.8.1 Intersystems Caché

The contact centre application has, at its core, an Intersystems Caché object database [25]. This is a database and programming environment that allows the objects modelled in the database to be manipulated in the Caché objectscript language or using more conventional, relational database SQL commands. AACC is currently using Caché version 2017.2. It offers a fully featured programming IDE in the Caché Studio application, and in the Atelier plug-in for Eclipse. Much of the data wrangling and feature extraction work was done in Caché Studio, Caché Terminal and in the SQL immediate execution in the System Management Portal. Objectscript functions were written to extract the Caché data to in ARFF format for processing in WEKA. The same methods are exposed with Caché web methods so that the new ML Open Interface component can use them to convert the real-time data into a form to pass to the trained WEKA models for predictions.

2.8.2 Microsoft Visual C#

The deployment of the trained model must conform to the AACC Email Open Interfaces specification [26]. This is most easily achieved by writing a C# web service for hosting in the IIS instance on the AACC server. The IKVM software is used to convert the weka.jar to a .NET compatible library, weka.dll. The C# webservice method loads the trained WEKA models and implement the classification ensembles to make predictions on the new emails passing through the AACC email manager.

2.8.3 WEKA

The Waikato Environment for Knowledge Analysis, WEKA, is an open source application that provides implementations of a large range of machine learning algorithms [27]. The WEKA Explorer application is used to analyze the data features from the ARFF files extracted from the contact centre data. A variety of algorithms and properties can be compared quickly and iteratively to find the best the algorithms and important features. The models once trained and evaluated can then be exported from WEKA and the model files loaded and used again later.

The knowledge flow application in WEKA allows a workflow of tasks to be sequenced together to load the input training data in ARFF format, select the class column, split the training data for 10-fold cross validation and then feed in to selection of classifiers. The results piped through to a TextViewer and to a ModelPerformanceChart and ImageViewer where ROC curves could compare the relative performance of each.

At runtime in the ML Open Interface, the WEKA models are loaded and the distributionForInstance method used to get a probabilistic prediction for each possible outcome. The WEKA features used in the research are collated in Table 2.1.

WEKA Feature	**Description**
MultiFilter	Create a chain of filters for a FilteredClassifier
FilteredClassifier	Apply a filter before training
StringToWordVector	Create a vector space model from a string of text
AttributeSelection	Sort and keep the most important features by information gain
Discretize	Convert continuous numeric data to discrete ranges of bins
ClassBalancer	Distribute weights of training sample in an unbalanced data set
J48	WEKA implementation of the C4.5 decision tree classifier
SMO	WEKA implementation of the Support Vector Machine classifier
IBk	WEKA implementation of K-Nearest Neighbours classifier
NaiveBayes	WEKA implementation of Naive Bayes classifier
SimpleKMeans	WEKA implementation of K-Means clusterer
ArffLoader	Open an ARFF format file
ClassAssigner	Pick the label from the list of attributes in the ARFF file
ClassValuePicker	Pick the positive value from the class label
CrossValidationFoldMaker	Split the training data into folds for training and testing
ModelPerformanceChart	Visualize classifiers' performance
ImageViewer	View images from the ModelPerformanceChart
TextViewer	View classifier performance in a text viewer
DistributionForInstance	Called on the C# interface to make a prediction on a sample

Table 2.1. WEKA functionality used in this project

2.8.4 Microsoft Excel

Microsoft Excel is a spreadsheet application that is part of the Microsoft Office suite. It was used in the analysis of the model performance data from the WEKA output to sort, filter and aggregate the data to gain insight and produce charts and tables to help visualize the results.

3 Design

This chapter describes the design of the solution to address the objectives set out in chapter 1, that is to introduce machine learning techniques to AACC by implementing prediction models to assist with email routing.

3.1 Problem

The email routing in AACC was identified as an area that could be enhanced to offer improved feature performance to benefit our customers. The areas selected for further investigation and research in this project are in the areas of spam detection and reply prediction with each classification having a bearing on how the contact is routed onward though the contact centre. Figure 3.1 from the AACC training material shows the lifecycle of an email contact through the AACC components. The rules engine applies the predefined rules maintained by the application administrator to match keywords to skills. The contact is saved to the database and triggers the flow through the queuing components, CCMS and CCT until the email is presented to an agent on the Agent Desktop. The agent handles the email to answer the query and wrap-up with a closed reason. There is no feedback to automatically update the rules if the routing is incorrect. In that case, the contact would present to an agent and will have to be transferred to another agent or skill at that time.

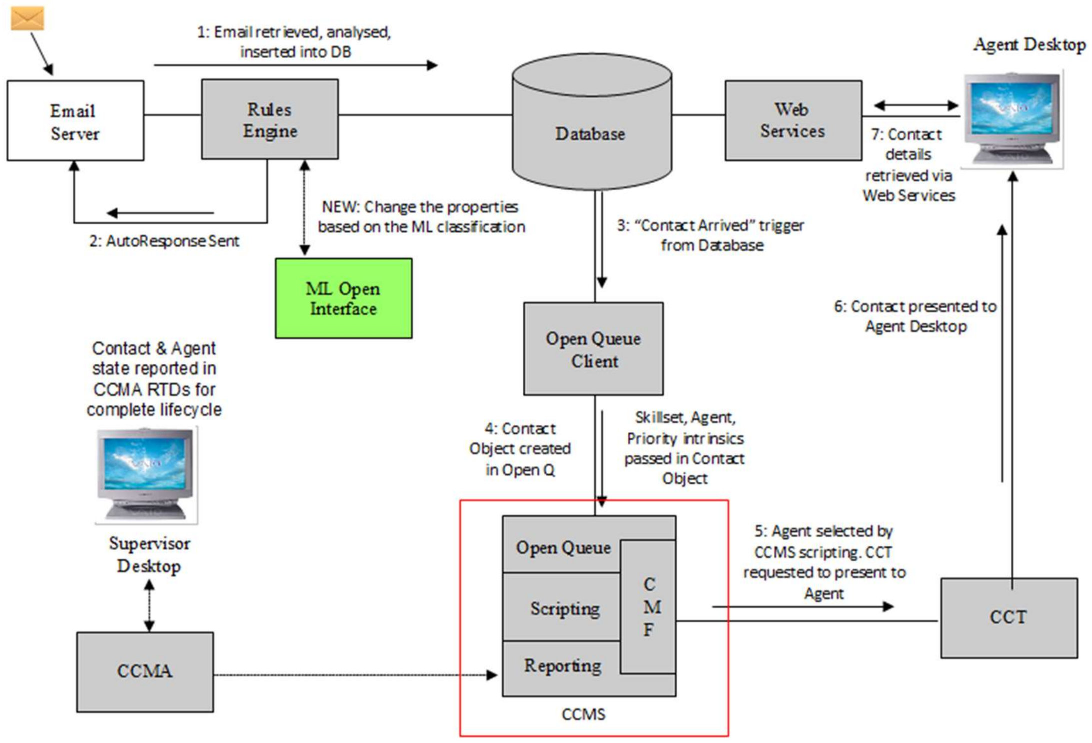

1: Email retrieved, analysed, inserted into DB
Email Server
Rules Engine
Database
Web Services
Agent Desktop
7: Contact details retrieved via Web Services
2: AutoResponse Sent
NEW: Change the properties based on the ML classification
ML Open Interface
3: "Contact Arrived" trigger from Database
6: Contact presented to Agent Desktop
Open Queue Client
Contact & Agent state reported in CCMA RTDs for complete lifecycle
4: Contact Object created in Open Q
Skillset, Agent, Priority intrinsics passed in Contact Object
Supervisor Desktop
Open Queue
Scripting
Reporting
CMF
CCMS
5: Agent selected by CCMS scripting. CCT requested to present to Agent
CCT
CCMA

Figure 3.1. AACC Email Lifecycle with proposed Machine Learning Open Interface

To improve this, a set of prediction models will be trained using the historical contact centre data. These models will be deployed into the lifecycle as a new component built according to the AACC Email Open Interface API [26]. Each email coming through the email manager will be processed through the ML Open Interface and the predictions from the models applied to augment the basic email routing with improvements in agent efficiency as an outcome.

3.2 Approach

The approach is to follow the Cross Industry Standard Process for Data Mining. This is the iterative process to prepare the data for modelling in WEKA and use the model outcomes to feed back to modify the data feature selection and training model parameters until a satisfactory performance is achieved and the models can be deployed. The modelling in WEKA will develop an understanding of the data and the features and how they correlate with the class predictions.

The data used in the research is from a live customer site from a period over 4 months, with activity from about 100 agents handling in the region of 2000 email contacts per day, a corpus of almost 300,000 emails. This data will be analyzed to identify the features that can contribute to the classification problems of detecting spam and predicting if a reply is needed. The features will be extracted from the core AACC data into a sandbox table for further manipulation. The site classifies 4.8% of the email contacts as spam and 37.6% of emails do not receive a reply from an agent.

Figure 3.2 shows the iterative approach to the training and evaluation of the classification models. When the performance is good enough to be deployed then the WEKA model files will be exported from the tool and saved to be used in the ML Open Interface web service project.

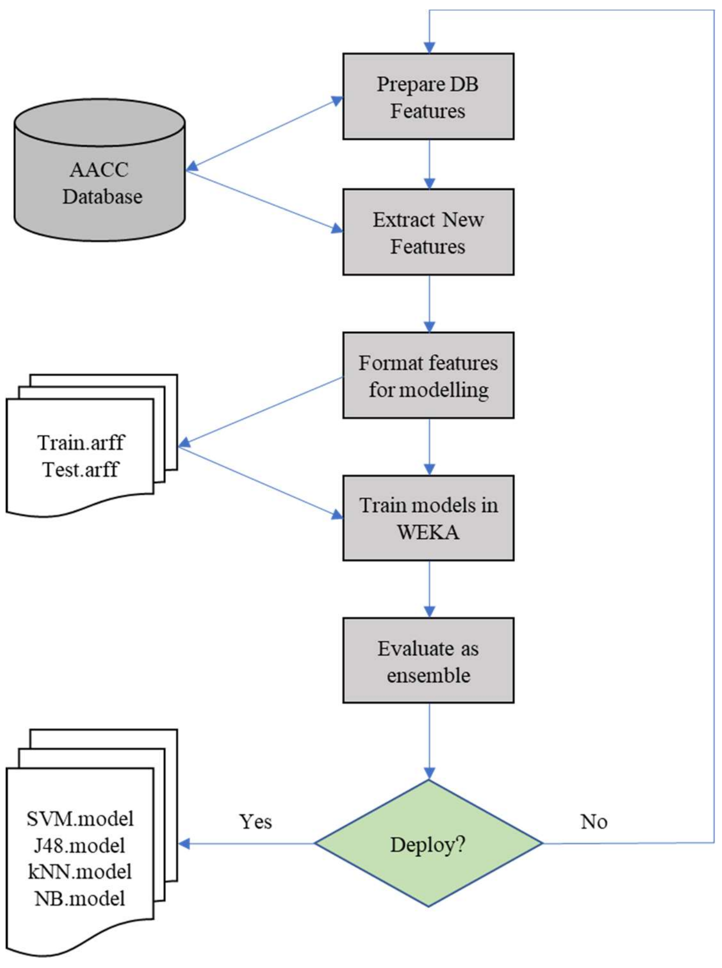


Figure 3.2. Iterative experimentation to train the WEKA models

The email open interfaces follow a simple pattern where up to five email properties can be passed in, and five others returned to add or change the properties that then get persisted to the contact centre database. The AACC administration screens to configure the rules is shown in the appendix 8.4.

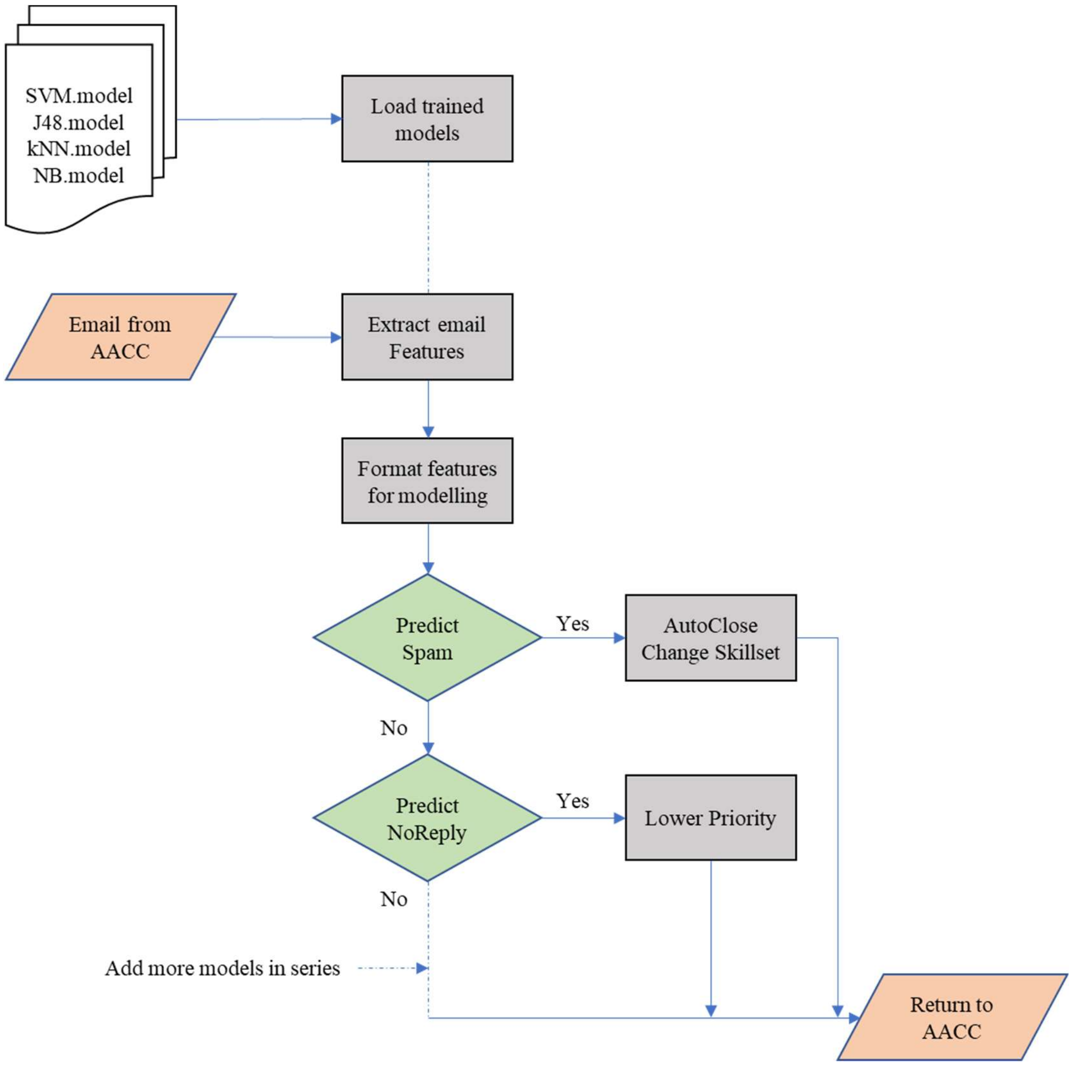


Figure 3.3. Machine Learning Open Interface deployment of trained WEKA models

The design of the ML Open Interface component is shown in Figure 3.3. When the service starts up, the trained models are loaded and then each time the email manager is processing a contact it will pass the contact properties to the web method. The properties will be extracted in the form expected by the models, and each model will be asked to make a prediction. The code in the ML Open Interface will implement an ensemble vote with a configurable threshold to determine the final prediction. If the first decision determines the contact to be spam, then the properties will be set to assign the contact to the EM_Spam skillset and to be automatically closed. If the first decision was not Spam, then it continues to make the next prediction; Reply or NoReply. Once again, the prediction will be formed by an ensemble vote and if NoReply is the outcome the priority of the contact will be lowered and returned to the AACC flow. Further decision blocks could be added to the series in future if other improvements are identified and modelled.

3.3 Experiment Design

The experiments were designed to iteratively discover more about the dataset and modify the features in the training to improve the accuracy of the predictions in each run. These iterations looked to follow a similar pattern: to prepare the feature data in the database table for easy and repeatable extraction to the ARFF file format. The extraction routines were modified to sample the data for training (60%) and testing (40%).

In each run, the training data was used to train various models which were first validated using 10-fold cross validation to get a measure of the accuracy. The trained models were then tested with the hold-out data and the accuracy verified to be in the same range at the cross-validated data to ensure that the models did not overfit the training data. After some experiments, the WEKA output was set to include the detailed information about the output predictions so that it was possible to drill down into the detail of the test instances; where the models agreed, where there were differences, where the probabilistic classifiers were certain of the prediction. The detailed information from WEKA allowed deeper analysis of the results in Excel, to build up the ensemble prediction and investigate the threshold selection.

4 Implementation

This chapter will discuss the implementation from the initial data collection, data preparation and wrangling to extract the features in a way that can be used in the WEKA modelling. It will also describe how the models were trained, saved and used by an open interface plug-in consumed directly by AACC to alter the email properties.

4.1 Data preparation

The AACC system uses an Intersystems Caché database to store the contact and customer history from all the previous interactions, the relevant subset of the entity relationship model is shown in Figure 4.1 in a diagram taken from the AACC Performance Manager and Data Dictionary [28]. When an email contact arrives in to the contact centre the MIME message is deconstructed and stored in its constituent parts; from address, to addresses, CC addresses, date, subject, plain text part, html text part, and attachments.

The hierarchy from customer to contact to action represents the continuing dialog between a customer and the contact centre. The customer table stores information about the end user and groups all correspondence with that individual. The contact level stores information about separate interactions over a variety of channels; Email, Chat, Fax, SMS and Scanned Documents (voice calls are stored in a different system). The action table stores activities related to each interaction; so, for a typical email contact there would be an email incoming from the customer, followed by an immediate auto-response email back to the customer, possibly followed by a reply from an agent to the customer and finally a wrap-up action to capture the outcome.

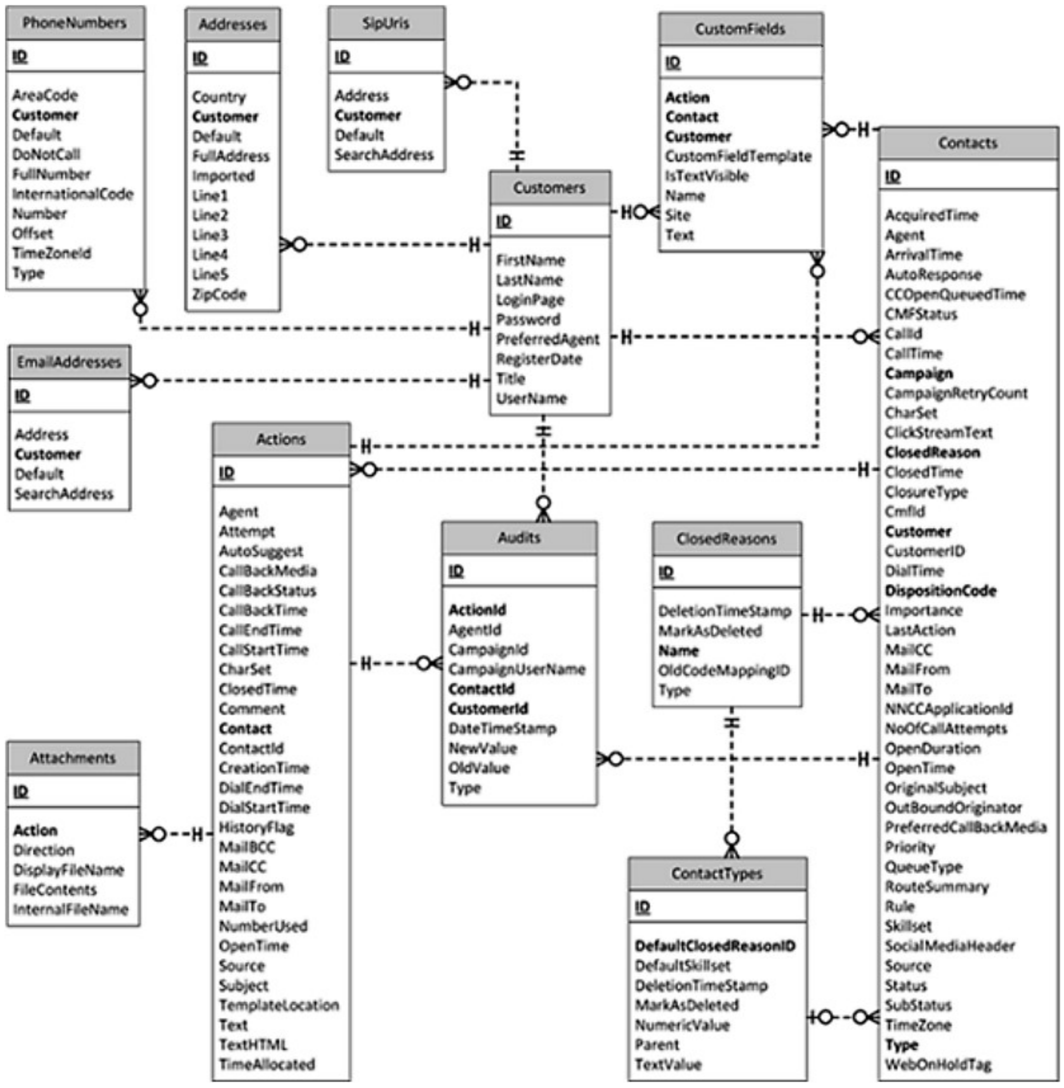


Figure 4.1. ER Diagram showing AACC tables related to email contact history [28]

For this research the primary data of interest is from the actions that represent an incoming email from a customer. This data was consolidated in a new table called "Features" for further analysis and manipulation. SQL statements and Caché objectscript functions were written to insert the instance records into the Features table and then modified to clean the data and derive new aggregated properties [29]. Some of the SQL commands are captured in the appendix 8.1 and these were written to perform the following filtering and aggregation functions:

- Focus only on email, remove contacts from other channels; Chat, Fax, SMS, Scanned Documents.

- Remove unresolved (New and Open) contacts from the data snapshot. These will not be properly labelled samples as they have no closed reason and it is uncertain if there would have been a reply.
- Identify the recipient mailbox from the lists of address in the MailTo or MailCC fields. The mailbox is the most significant recipient as it is the reason the email was ingested into the contact centre.
- Set the Followup property by analyzing the subject field for a marker indicating a dialogue with the contact centre. An email is unlikely to be spam if it is a continuation of a conversation.
- Set Spam / NoSpam category based on the ClosedReason set by the agent. Store the spam label explicitly to simplify the extraction to the ARFF format.
- Set Reply / NoReply category by looking for "Email from Agent to Customer" in the contact action log. Store the NoReply label explicitly to simplify the extraction to the ARFF format
- Keep the most recent part from the email chain only, remove the conversation history. The conversation in the earlier emails will add content to the bag-of-words that could dilute the meaning of the incoming email, to make it harder to identify the focus of the most recent reply.
- Count the recipients from the MailTo and MailCC lists. Long lists of recipients could indicate that the message was not intended for the contact centre which might expect one recipient; e.g. info@avaya.com.
- Count the interrogation words and symbols (Who, What, When, Why, Where, How, "?", etc.). These words and symbols indicate a question that suggests the need for a reply [22].
- Perform entity tagging by using regular expression matches to replace IP addresses, email addresses, phone numbers, links and image URLs. The specific content of these fields will have a low term frequency but become useful when replaced with a placeholder tag; e.g. |url|. A bag-of-words vector containing the presence of one or more URLs, links and images is more informative as a training feature.

The following tables describe first, the feature columns taken directly from the AACC tables and second, those derived using SQL or objectscript code.

Feature Column	**Description**
ContactID	Primary key field and link to the core AACC data
CustomerID	Foreign key field link to the AACC data
MailFrom	Source email address of the incoming contact
MailTo	The csv list of recipients the email was sent to (one of these will be the contact centre mail box)
Subject	The subject property of the email
Body	The plain text body of the email (stripped of html tags)
Rule	The name of the rule that was used to route the contact
Skillset	The name of the skill where the contact was routed (of interest for future work)
ClosedReason	The wrap-up decision assigned by the agent on completion. This field is used by the agents to label some contacts as spam
OpenDuration	The amount of time agents spent handling the contact. This is an accumulation of all the time agents spent on the contact before getting resolved.
Importance	Importance property from the incoming MIME email

Table 4.1. Features copied from the AACC tables

Feature Column	**Description**
Followup	Flag indicating if the customer mail was part of a conversation
ReplyCount	Flag indicating if the customer mail was replied to by an agent
MailToCount	Number of mail to recipients
MailCCCount	Number of CC’d recipients
MailBox	Contact centre mailbox extracted from MailTo or MailCC field
NewCustomer	Flag indicating if the incoming contact was the first from this customer
MessageClean	Interim step to remove the conversation from the mail to leave

	the most recent reply
MessageCleanTagged	Replace entities (IP address, links, images, phone numbers) with tags
ArrivalTime	The time of day the contact arrived in the contact centre
TextLength	The length of the email message
SubjectLength	The length of the email subject
Spam	Flag if the closed reason was “Spam”
W5	Presence of interrogation words (e.g. Who, What, When, Why, Where, ?)
AttachmentCount	Number of attachments on the incoming contact

Table 4.2. Features derived from the AACC data

The features were identified from an understanding of domain and from the literature. To train the models for spam detection the features identified were Followup, RecipientCount, ArrivalTime, TextLength, SubjectLength, and the bag-of-words from the Subject and message Body. Caché objectscript code was used to populate training and validation test ARFF files from the Features table to be used in the experiments. A sample of the objectscript code is listed is in the appendix 8.2 and a redacted sample of the ARFF file format is listed in appendix at 8.3. Early experiments in WEKA meant that correlation between these attributes could be quantified using the Information Gain evaluator of the AttributeSelection filter.

The following charts help visualize some of the relationships between the features and class labels. The examples show the cases when the body text contains zero or more URLs, when the body text contains the word “address”, and if the email is a follow-up to an earlier contact. In each chart those labelled Spam are coloured blue and NoSpam is in red. URLs are a good indicator of spam in this contact centre context, while the word “address”, or follow-up emails that are part of an ongoing dialogue are indicative that an email is not spam. There were many other examples that could be shown, but these few had a high relative information gain and are shown here to represent the analysis.

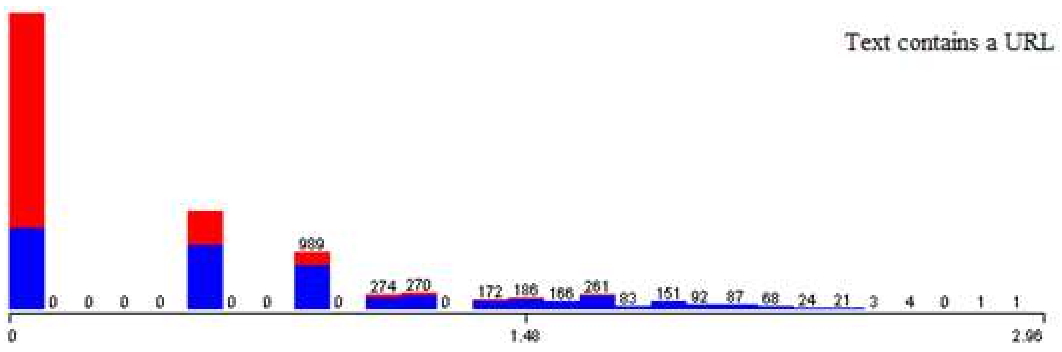


Figure 4.2. Correlation between Spam class and text containing URLs

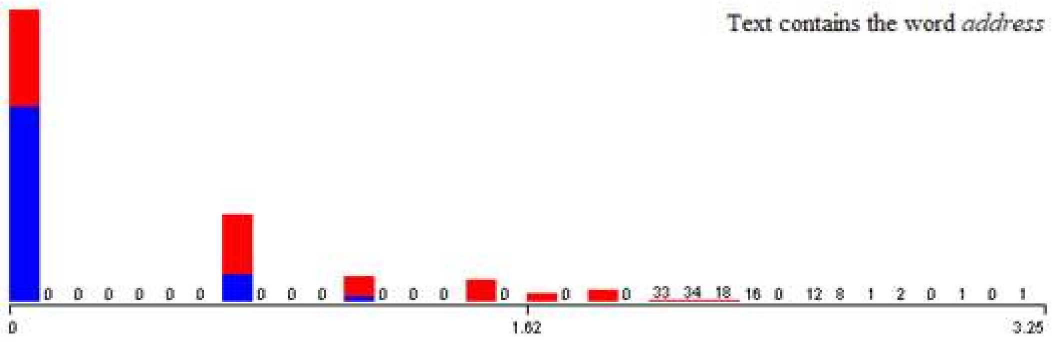


Figure 4.3. Correlation between Spam class and text containing the word *address*

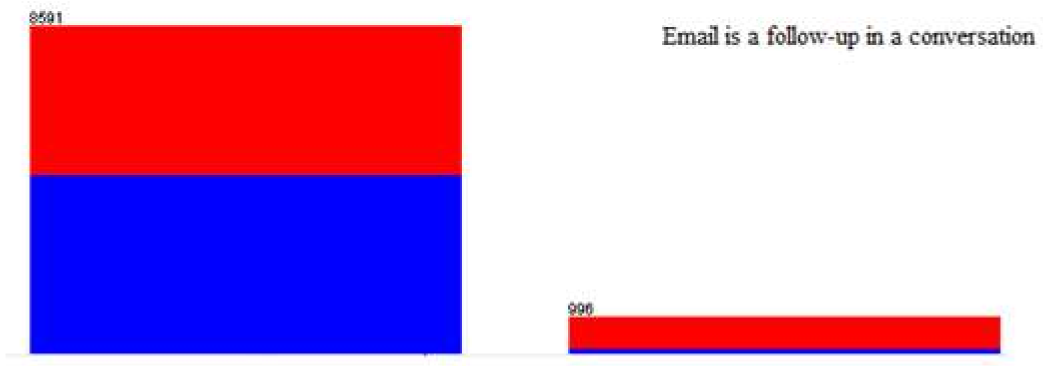


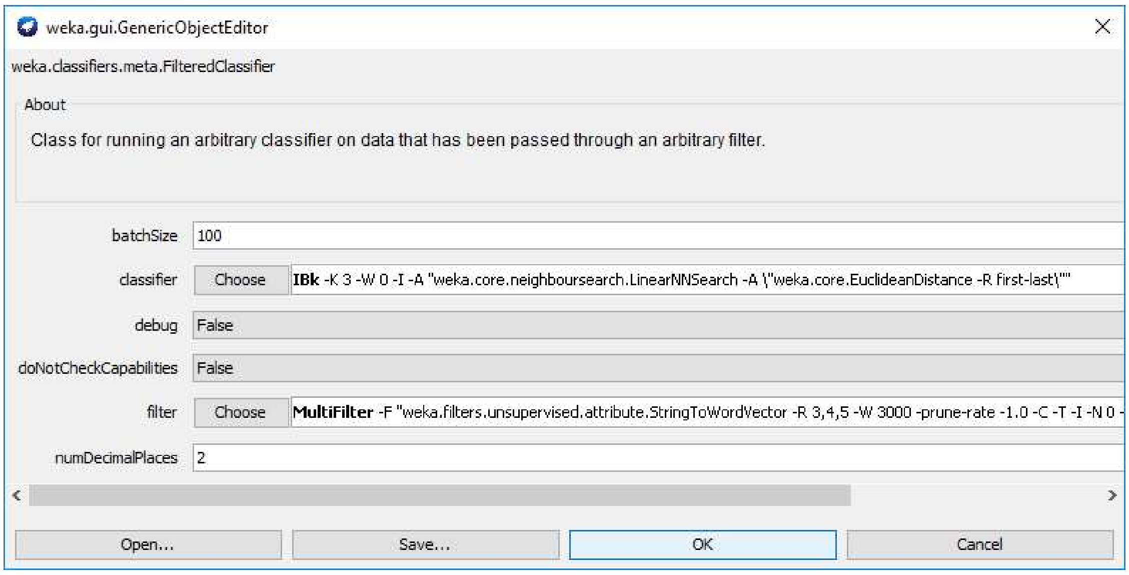
Figure 4.4. Correlation between Spam class and derived feature Followup

The classification of the training samples comes from the historical records in the contact centre and may contain inconsistencies. The day-to-day agent behaviour in how they deal with certain types of email should be consistent but is subject to human error. The action to close a contact and set the closed reason to “Spam” is more likely to be omitted, than for a contact that is not spam to be misclassified as such. For this use case, that the class label noise is biased toward NoSpam may be beneficial as it is better to route the spam contact instead of automatically closing a legitimate email.

4.2 Experimentation

The WEKA workbench provides implementations of many machine learning algorithms. Modelling experiments were conducted using the WEKA explorer application to find the best combination of features, filters and classifier properties.

The training data from the AACC system contains a combination of numeric features; Followup, ArrivalTime, TextLength, SubjectLength and text features; MailTo, Subject and Body. The FilteredClassifier combines a preprocessing filter with a classifier. The StringToWordVector filter was used to convert the text features to a numeric word count vector, a bag-of-words, where each unique word becomes a feature. Adjusting the StringToWordVector filter properties formed part of the experimentation to vary the use of TF-IDF weighting, stemming, stop word handling, tokenization and words to keep properties to find the configuration that produced the most accurate results. The AttributeSelection filter prioritizes the features that yield the highest information gain to feed onward to the classifier training and reduces the dimensionality by discarding the ineffective features.


weka.gui.GenericObjectEditor

weka.classifiers.meta.FilteredClassifier

About

Class for running an arbitrary classifier on data that has been passed through an arbitrary filter.

```
batchSize              100
classifier             IBk -K 3 -W 0 -I -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
debug                  False
doNotCheckCapabilities False
filter                 MultiFilter -F "weka.filters.unsupervised.attribute.StringToWordVector -R 3,4,5 -W 3000 -prune-rate -1.0 -C -T -I -N 0 -
numDecimalPlaces       2
```

Figure 4.5. FilteredClassifier to apply MultiFilter and kNN

The FilteredClassifier was configured to run the experiments using the commonly used text classification algorithms; Support Vector Machine, C45 Decision Tree, k-Nearest Neighbours and Naïve Bayes. In WEKA, these are implemented respectively as SMO, J48, IBk and Naïve Bayes classifiers. In each iteration the same number from each class of samples were selected for balance in the training. A set of 16,000 samples was split 60% for training and 10-fold cross-validation with the remaining 40% held back for validation testing. The similarity in the accuracy from the cross-validation test and the hold-out test set ensures that the model has not been trained to overfit the training data.

Spam prevention			
Model	Cross-validation	Hold-out	Diff
NB	85.89	84.84	1.05
kNN	86.38	85.89	0.49
J48	88.46	88.08	0.38
SMO	90.18	89.11	1.07

Table 4.3. Compare cross-validation to hold-out accuracy for spam detection

Reply prediction			
Model	Cross-validation	Hold-out	Diff
NB	73.65	73.96	-0.31
kNN	73.70	72.31	1.39
J48	79.00	78.49	0.51
SMO	81.70	81.33	0.37

Table 4.4. Compare cross-validation to hold-out accuracy for reply prediction

Using balanced training data split close to 50-50 the accuracy of the trained models consistently suggested that for each of these classification tasks, the Support Vector Machine performed best, followed by the Decision Tree, then k-Nearest Neighbours and the worst performing algorithm was Naïve Bayes. Further experimentation to improve the performance of the Bayes classifier by discretizing the numeric attributes did not help.

Spam prevention							
Model	Accuracy	TP	FP	Precision	Recall	F-Score	AUROC
NB	84.84	0.848	0.151	0.849	0.848	0.848	0.889
kNN	85.89	0.859	0.140	0.861	0.859	0.859	0.913
J48	88.08	0.881	0.119	0.881	0.881	0.881	0.922
SMO	89.11	0.891	0.109	0.891	0.891	0.891	0.948

Table 4.5. Accuracy comparison for different model algorithms for spam detection

Reply prediction							
Model	Accuracy	TP	FP	Precision	Recall	F-Score	AUROC
NB	73.96	0.740	0.260	0.757	0.740	0.735	0.802
kNN	72.31	0.723	0.277	0.735	0.723	0.719	0.822
J48	78.49	0.785	0.215	0.785	0.785	0.785	0.823
SMO	81.33	0.813	0.187	0.814	0.813	0.813	0.875

Table 4.6. Accuracy comparison for different model algorithms for reply prediction

When the trained spam prevention models were tested using an unbalanced data set representative of the real-world with 5% spam in the samples, then the accuracy dropped, and the performance became unacceptable, no better than random. This seemed to be because many of the samples in the training and testing data set were incorrectly labelled due to the manual and inconsistent behaviour of the agents in the contact centre during the wrap-up phase when handling email contacts.

Changes were implemented to try and improve the classification results, with additional experiments focused on SVM, the most successful classifier from the earlier experiments. First, the training data was selected at random with no attempt to balance the spread of each class, then the WEKA ClassBalancer filter was applied to increase the weight of the minority class in the training. Neither of these changes made any difference to the performance. An attempt was made to use the prediction and probability from the probabilistic classifiers to assign a score to each sample and apply through a threshold to make the final prediction, but a clear threshold could not be identified to make this effective. The cause of the poor performance was due to the poor class labelling so changes to properly label the source data were needed.

This problem was discussed by Nicholson *et al.* where they proposed a cluster-based correction method to automatically address and fix label noise [30]. A similar methodology was used to discover the distinct categories of spam, which provided enough information that much of the label noise could be identified and updated manually.

To get an understanding of the extent of the poor labelling, all the subject text from samples labelled Spam were extracted to an ARFF format file and processed through a WEKA k-Means clustering algorithm to detect the different types of mail that were being tagged [31]. This process yielded seven clusters that could be extrapolated back to different known categories of email. When emails matching the keywords from the clusters were pulled from the entire corpus it was possible to quantify the problem. Many of these were invitations and reminders from LinkedIn, some were related to spam filter detection warnings. Some were from testing of a custom solution to send emails from a web page as a route to the contact centre agents.

k-Means Cluster Categories	Spam	NoSpam	Total
LinkedIn	616	614	1230
Reminder about your invitation	990	946	1936
Web portal test	587	8643	9230
Spam quarantine report	1428	710	2138
User quarantine release notification	71	3	74
Pet Expo	104	73	177
Response from customer care	80	714	794
	3876	**11703**	**15579**

Table 4.7. Categories found by running k-Means on the subject field of spam email

Each of the emails in these clusters are so similar, that if one is spam then all must be spam. Table 4.7 shows the main keywords that were identified in each cluster with a breakdown of how consistently the samples were labelled. Table 4.8 shows in more detail emails from the LinkedIn cluster, where some emails with the identical subject (and body) were labelled Spam and others were not. The 11703 incorrectly labelled data items were updated in the database in preparation for another data extraction and training run.

Subject	Label
40,000 leads from...LinkedIn???	NoSpam
Faça parte da minha rede no LinkedIn	Spam
Invitación a conectarnos en LinkedIn	NoSpam
Invitación a conectarnos en LinkedIn	Spam
Invitation à se connecter sur LinkedIn	NoSpam
Invitation à se connecter sur LinkedIn	Spam
Invito a collegarsi su LinkedIn	NoSpam
Invito a collegarsi su LinkedIn	Spam
Join my network on LinkedIn	NoSpam
Kontaktförfrågan på LinkedIn	Spam
Millions of LinkedIn passwords stolen	Spam
Reset Your LinkedIn Password	NoSpam

Reset Your LinkedIn Password	Spam

Table 4.8. Samples of mis-labelled data with subject containing LinkedIn.

In the previous tests the SVM classifier got 89% accuracy on a real-world unbalanced data set (where only 5% were labelled Spam), which was poor. Fixing the labels revealed that there is closer to 10% unwanted mail in the system. This investigation also showed that there was other data in the corpus that had been handled using a delivery failure rule, did not route to agents and so did not get a closed reason. These 397 contacts were also updated to assign the proper label, Spam.

SVM	Accuracy	TP	FP	Precision	Recall	F-Score	AUROC
Before	89.11	0.891	0.109	0.891	0.891	0.891	0.948
After	95.32	0.953	0.233	0.953	0.953	0.953	0.947

Table 4.9. SVM improvement after fixing class label noise

With this significant improvement observed in the SVM trained model, the other models were trained again, and the same order of improvement observed with those. Also, the decision tree produced from the properly labelled data had a much simpler form, reduced in complexity from a tree with 226 leaf nodes to a tree with just 62 leaf nodes.

To visualize the performance on a test set of 4016 samples the predictions from each of the four models was added together to get an ensemble score shown in Figure 4.6. Each of the four models will predict Spam or NoSpam with a probability. The Spam probabilities are added and the NoSpam probabilities subtracted to give a total ensemble score for each sample (in the range -4 to +4). The scores were sorted in ascending order and plotted on the chart alongside the instance label (where NoSpam = -1 and Spam = 1). The clusters of NoSpam samples between 3700 and 3820 on the chart represent 46 samples in the data. On closer inspection, 30 of these were labelled incorrectly, but no further work was done to resolve this remaining noise. This performance represents a successful way to identify unwanted emails arriving in to the contact centre. It remains to select a suitable threshold for the ensemble score to maximize the detection and minimize the false classification of NoSpam. The threshold could be configurable in a deployed solution. Using the ensemble prediction with no threshold in this test scenario would detect 286 of the 388 labelled Spam with 77 false positives (NoSpam classed as Spam). Setting a threshold ensemble score greater than 3 will detect 228 labelled Spam with 7 just false positives. This partitioning is shown by the red vertical line on the chart.

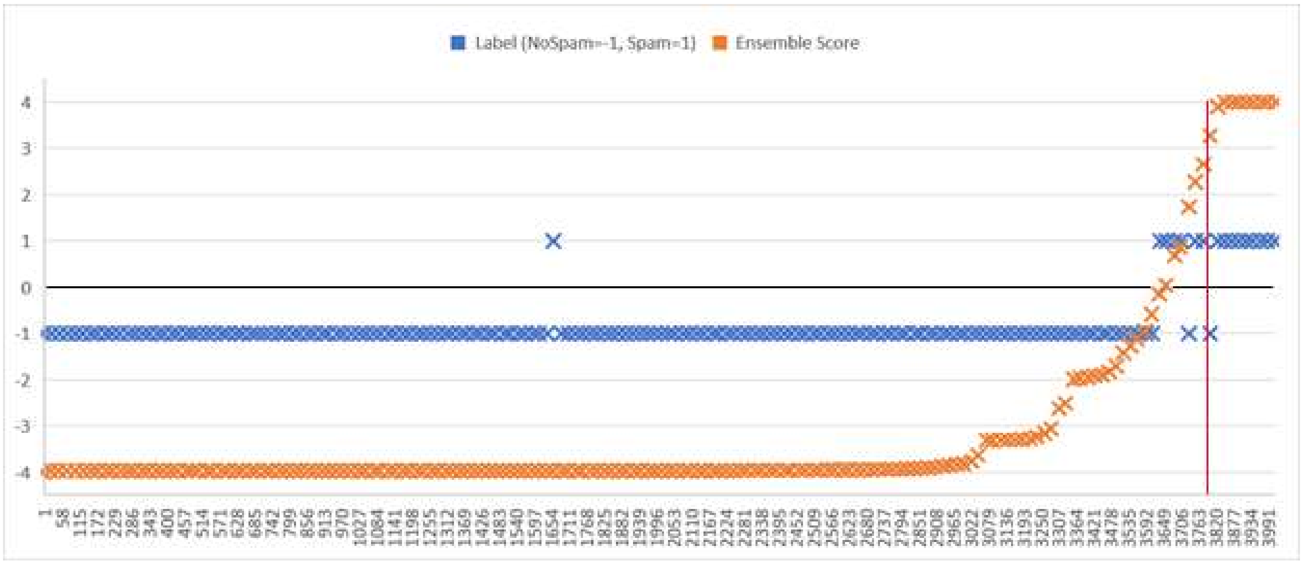


Figure 4.6. Chart showing the ensemble spam score and the assigned class label

When the same experiments were performed on the data extracted to predict if a reply would be sent, there was less certainty in the predictions with more contradictions between the individual models, with relatively few scoring -4 or 4. However, when these separate scores were aggregated together in an ensemble there was a sharp improvement in the predictive performance. This is noted by Dietterich, that “a key to successful ensemble methods is to construct individual classifiers with error rates below 0.5 whose errors are at least somehow uncorrelated” [32]. Figure 4.7 shows the aggregate ensemble score for the NoReply prediction plotted alongside the assigned label from the test data set (where Reply = -1, NoReply = 1). When no threshold is applied (i.e. taking a positive ensemble score to predict that no reply will be sent and a negative score predicting that a reply will be sent), 28 instances incorrectly classified that no reply would be needed where one was sent from the contact centre. Setting the threshold to 1 was able to fully separate the data to recall all the emails that need a reply while identifying 793 from 1099 that would not. This partitioning with threshold of 1 is shown by the red vertical line on the chart. If the model predicts that no reply will be sent, then the priority can be lowered, and higher priority tasks delivered to agents more quickly.

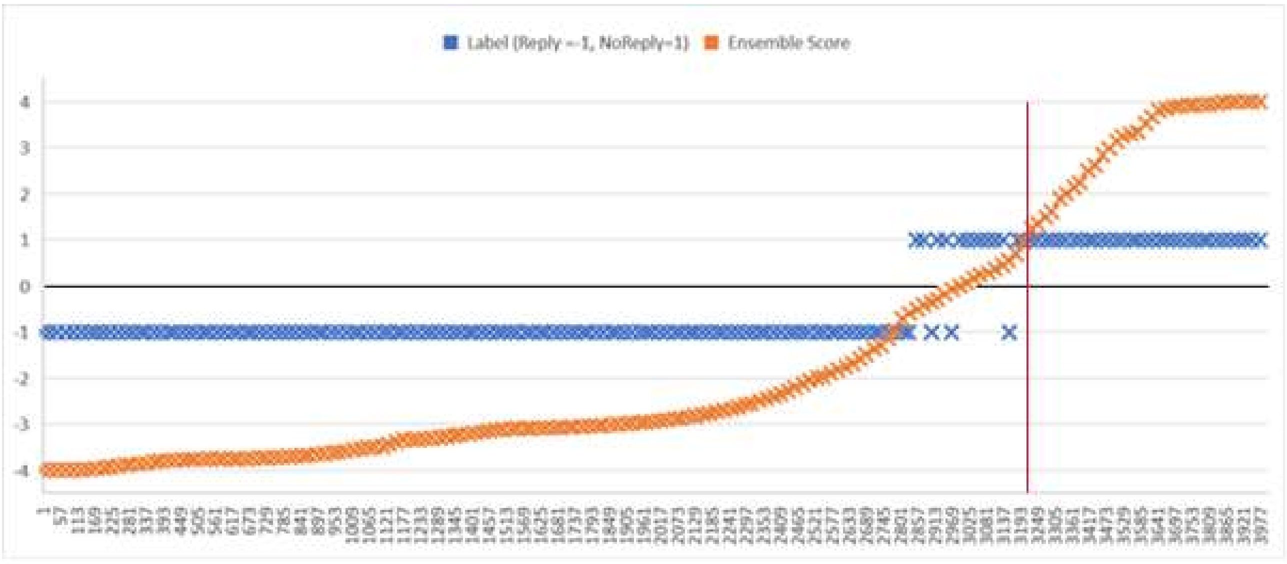


Figure 4.7. Chart showing the ensemble NoReply score and the assigned class label.

5 Results

In the objectives set out at the start, the goals were to improve the email routing in AACC with a machine learning approach. This learning was focused on two separate problems; to prevent the routing of unwanted spam email and to identify emails that will not need an agent to send a reply and route with a low priority. This chapter will present the results from the individual performance of each of the models; SVM, J48, kNN and Naïve Bayes, the ensemble performance considering the aggregate score from all the models and the ensemble performance after applying a threshold.

Each of the models is summarized in a confusion matrix, with the counts of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) predictions displayed. Each of these tests were conducted using the hold-out data set after the model had been trained on a separate training data set.

Model		Predicted	
		A	B
Label	A	TP	FN
	B	FP	TN

Accuracy Measure		
Precision	Recall	Accuracy
P(A)	R(A)	Accuracy
P(B)	R(B)	

The accuracy scores in the next sections are shown for Precision, Recall and Accuracy defined as follows. The precision and recall are calculated in terms of both the majority and minority classes as follows:

Precision (A) = TP / (TP + FP)

Recall (A) = TP / (TP + FN)

Accuracy = (TP + FN) / (TP + FN + TN + FP)

5.1 Spam Detection result

Precision and recall are the important measures of accuracy in this unbalanced data, that is, to find the spam without the risk of losing anything important. In terms of the accuracy measures, it is to maximize the recall of the majority class NoSpam while also preserving a high precision for the minority class, Spam. The overall accuracy score is shown but is less meaningful as the effectiveness of the system is no good if there is a too many false spam predictions [33].

SVM		Predicted	
		Spam	NoSpam
Label	Spam	289	99
	NoSpam	89	3539

Accuracy Measure		
Precision	Recall	Accuracy
0.765	0.745	0.953
0.973	0.975	

J48		Predicted	
		Spam	NoSpam
Label	Spam	270	118
	NoSpam	57	3571

Accuracy Measure		
Precision	Recall	Accuracy
0.826	0.696	0.956
0.968	0.984	

IBk		Predicted	
		Spam	NoSpam
Label	Spam	257	131
	NoSpam	88	3540

Accuracy Measure		
Precision	Recall	Accuracy
0.745	0.662	0.945
0.964	0.976	

Naïve Bayes		Predicted	
		Spam	NoSpam
Label	Spam	351	37
	NoSpam	291	3337

Accuracy Measure		
Precision	Recall	Accuracy
0.547	0.905	0.918
0.989	0.920	

Ensemble >0		Predicted	
		Spam	NoSpam
Label	Spam	286	102
	NoSpam	77	3551

Accuracy Measure		
Precision	Recall	Accuracy
0.788	0.737	0.955
0.972	0.979	

Ensemble >3		Predicted	
		Spam	NoSpam
Label	Spam	218	170
	NoSpam	7	3621

Accuracy Measure		
Precision	Recall	Accuracy
0.969	0.562	0.956
0.955	0.998	

Table 5.1. Confusion matrices for spam detection

The ensemble prediction, passed through a threshold where the score must be greater than 3 to be classed as Spam produces a useful balance for spam detection. Of the 7 false positive predictions, 5 should really have been labelled as Spam in the first place and the other 2 were foreign language messages where there were not enough similar samples in the training data.

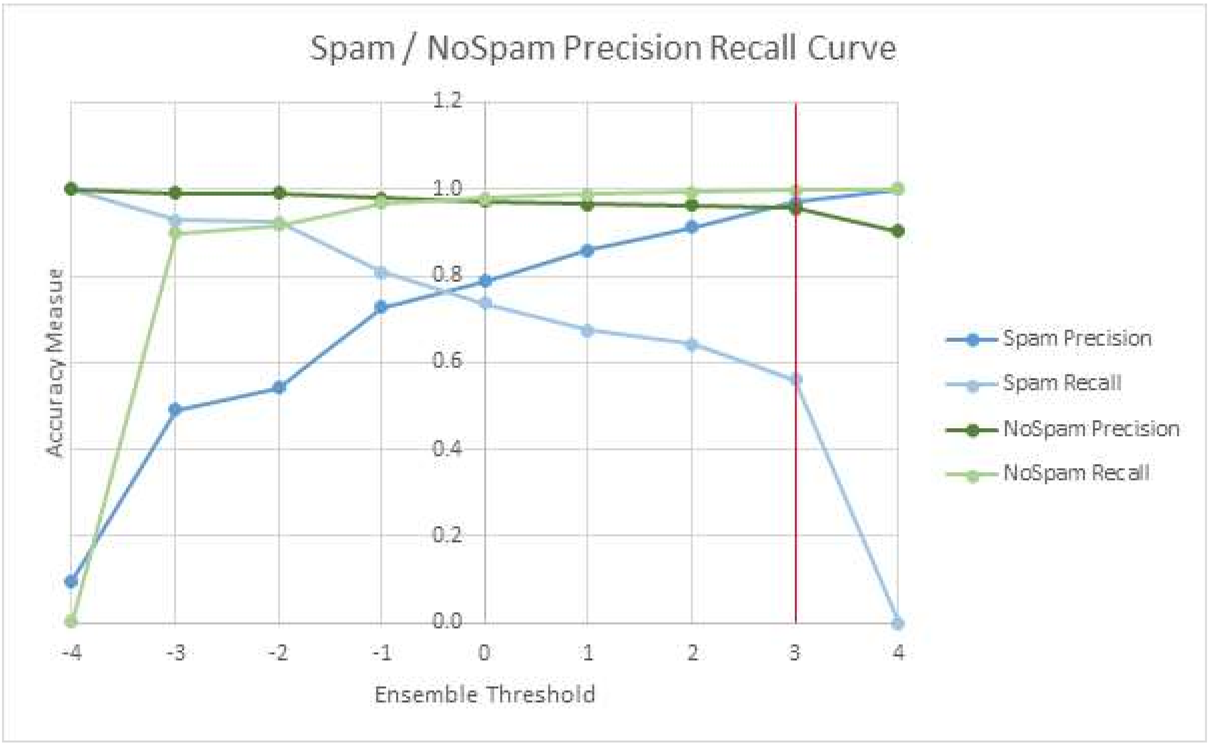


Figure 5.1. Plot of precision and recall against thresholds for spam detection

The graph in Figure 5.1 shows the comparison for precision and recall for Spam and NoSpam at selected threshold values for the ensemble to help select the best threshold for deployment. To maximize the number of spam emails detected will demand a low threshold, but this must be traded off against the number of false positives. This shows that with the threshold at 3, the model will identify almost 60% of the unwanted content with the minimum impact on the important emails flowing through the contact centre [34].

This represents a saving of 30 hours per month which will improve over time with additional training of the models on the edge cases that continue to escape and get closed as Spam. This continued improvement acting to boost the detection of the edge cases.

5.2 Reply Prediction result

In a similar way, the analysis on the results of the reply prediction problem show the comparison between the individual classifiers, and the ensemble of the four combined together.

SVM		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2600	301	0.877	0.896	0.834
	NoReply	365	734	0.709	0.668	

J48		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2631	270	0.859	0.907	0.824
	NoReply	433	666	0.712	0.606	

IBk		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2174	727	0.877	0.749	0.742
	NoReply	306	793	0.522	0.722	

Naïve Bayes		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2417	484	0.856	0.833	0.777
	NoReply	407	692	0.588	0.630	

Ensemble >0		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2873	28	0.960	0.990	0.963
	NoReply	121	978	0.972	0.890	

Ensemble >1		Predicted		Accuracy Measure		
		Reply	NoReply	Precision	Recall	Accuracy
Label	Reply	2901	0	0.905	1.000	0.924
	NoReply	306	793	1.000	0.722	

Table 5.2. Confusion matrices for reply prediction

None of the individual models demonstrate enough accuracy in their predictions to be useful as a real-world deployment. When they are aggregated together in the ensemble, even with a low threshold applied, the performance is excellent and represents a model that could easily be delivered on the customer site to improve the prioritization of work.

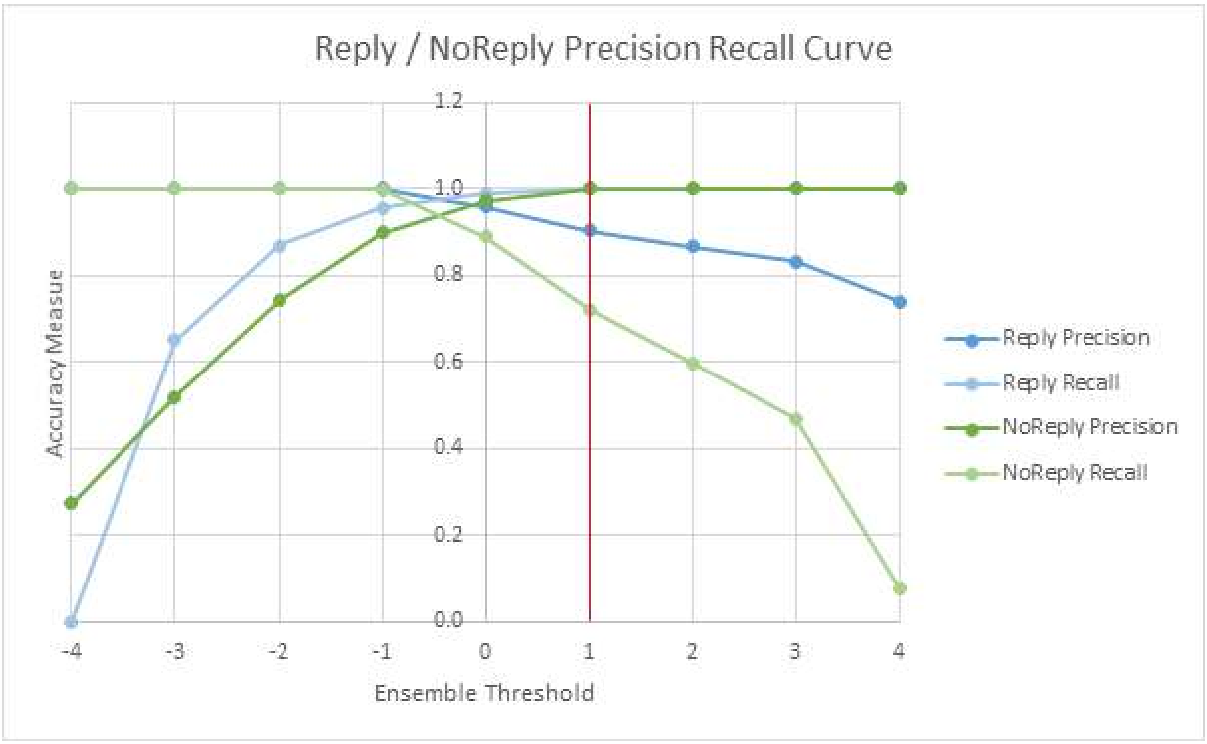
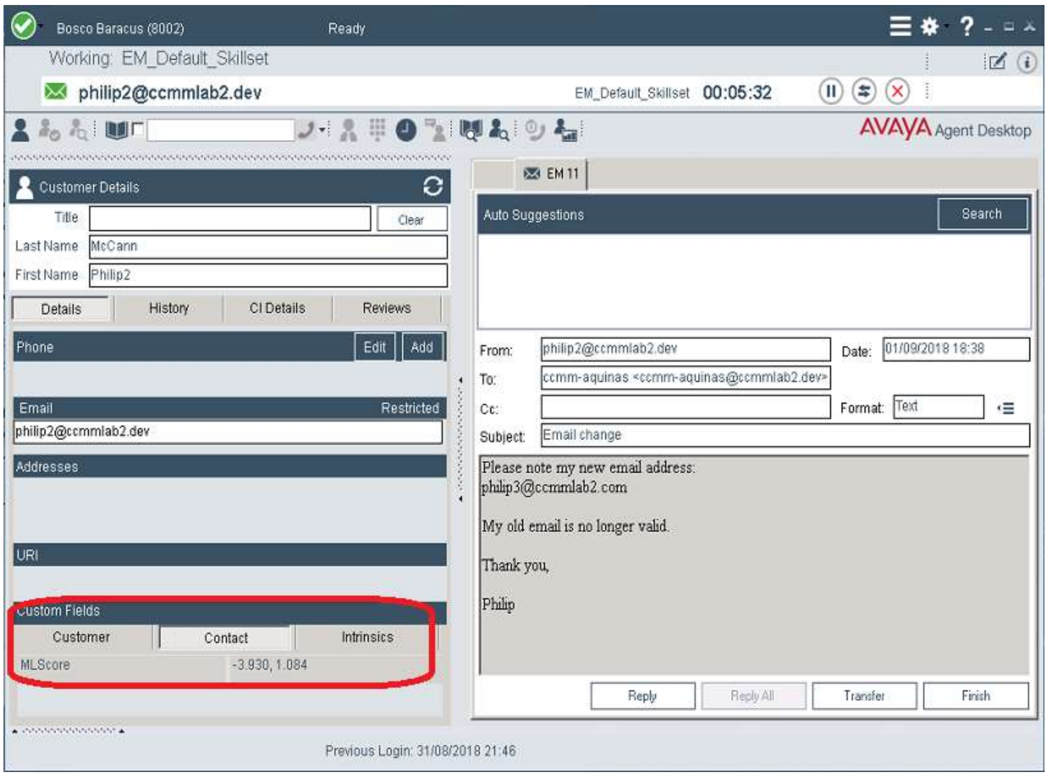


Figure 5.2. Plot of precision and recall against ensemble for reply prediction

5.3 Deployment

The ML Open Interface web application developed to integrate with the AACC email manager program was deployed in the IIS instance on a CCMM server in the lab. Known samples from the test data were used to test the runtime ensembles in Visual Studio to make sure the expected results were returned, and then the CCMM system was configured to include the new web method as part of the workflow for email routing. A variety of emails modelled on the known samples from the customer data were routed through the lab and handled in the open interface to change the routing outcomes. For the purposes of visibility and verification, a new custom-field property, “MLScore” was used to return a string representing the predictions. This field is persisted in the database and can be queried to assess the effectiveness of the routing changes.

An example of the AAAD screen is shown in Figure 5.3 after handling an email consistent with those from the customer site that do not get a reply. The Spam ensemble assigned a score of -3.930, classifying the email as NoSpam, and the NoReply ensemble assigned a score of 1.084 which exceeds the configured ensemble threshold, so this contact classified as NoReply and was created with a low priority (with priority in the range 1 to 10, where 1 is High and 10 is Low).



Title:
Last Name: McCann
First Name: Philip2
Email: philip2@ccmmlab2.dev
MLScore: -3.930, 1.084

From: philip2@ccmmlab2.dev
Date: 01/09/2018 18:38
To: ccmm-aquinas <ccmm-aquinas@ccmmlab2.dev>
Cc:
Format: Text
Subject: Email change

Please note my new email address:
philip3@ccmmlab2.com

My old email is no longer valid.

Thank you,

Philip

Figure 5.3. AAAD after routing an email through the ML Open Interface

6 Conclusions

6.1 Summary

This project sought to investigate how email routing in the contact centre could be improved with the application of machine learning techniques. An effective plug-in for the established AACC product was developed to utilize the historical data resource in a positive feature enhancement to benefit Avaya's end customers.

It has been shown that the framework developed to add modular learning model ensembles in series using the email open interface API can eliminate unwanted contacts from routing through the contact centre unnecessarily, and that higher priority items can be escalated to agents by analyzing the features of the incoming email. The enhancements when combined will improve the contact centre metrics for agent occupancy rate, cost per call and service level response times to end customers [35].

At the outset, one motivation was to eliminate the 14125 emails routed through the contact centre at a cost of 160 hours, in fact there were almost 27000 and a cost of 200 hours. The quality of the labelled data was a limiting factor in the success of the predictions from the models. A change in the operating process at the site to enforce more consistent agent behaviour and closed reason labelling would provide a better data source for training. Improved data quality, coupled with incremental re-training of the models will cause a positive feedback loop of better predictions and fewer unwanted emails entering the contact centre.

6.2 Future Work

These improvements, spam detection and reply prediction, were selected as they could add specific value to the customer dataset that was available for this research and implementation. The software developed sits outside the core application and could be customized by Avaya EP&T for the bespoke demands of other sites. It could also be developed into a core feature of the email routing paradigm for AACC or Avaya Oceana™.

The customer data used in this research is based on a model where there are a lot of inbound mailboxes that map easily to the agent skills. The routing rules do not have to parse the email contents to distribute the work. For other AACC implementations, fewer mailboxes are used with more complex sets of configured rules to match emails to skillsets using sender and keyword matching administered in the application. The primary goal of the email routing to assign the contact to the best skillset so that it is presented to an agent with the correct training and with the correct priority.

With that in mind, a further learning model could be added to the series to identify and learn from the transfer patterns in the contact history. A contact transferred from one skillset to another is indicative of an incorrect initial assignment, and therefore a problem with the routing. An additional learning model could be used to learn from the history and update the assignment to the correct skill. With this improvement, a contact would be more likely to be resolved by the first agent with fewer transfers.

7 Bibliography

[1] Avaya Trend Advisor, "Big Data in the Contact Center," 2013. [Online]. Available: http://www.club-cmmc.it/lettura/BigData.pdf. [Accessed 05 09 2018].

[2] R. Rowan, "Analytics in the Contact Center; The Road to a Better and More Profitable Customer Experience," 2011. [Online]. Available: https://www.avaya.com/en/documents/analytics-in-the-cc-gcc4844-%284%29.pdf. [Accessed 05 09 2018].

[3] K. Kirkpatrick, "AI in contact centers," *Communications of the ACM,* vol. 60, pp. 18-19, 2017.

[4] The Spamhaus Project, "The Definition of Spam," The Spamhaus Project, [Online]. Available: https://www.spamhaus.org/consumer/definition/. [Accessed 15 08 2018].

[5] Avaya, "Avaya Aura Contact Center Server Administration," 2018. [Online]. Available: https://downloads.avaya.com/css/P8/documents/101017434. [Accessed 05 09 2018].

[6] D. Krauss, S. Blood and S. Harrison, "Magic Quadrant for Contact Center Infrastructure, Worldwide," Gartner, 17 05 2018. [Online]. Available: https://www.gartner.com/doc/3875565/magic-quadrant-contact-center-infrastructure. [Accessed 05 09 2018].

[7] Cisco, "Cisco Unified E-Mail Interaction Manager 9.0 for Cisco Unified Contact Center Enterprise and Hosted and Cisco Unified Intelligent Contact Management Data Sheet," Cisco, 01 2016. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/customer-collaboration/unified-email-interaction-manager/data_sheet_c78-451504.html. [Accessed 05 09 2018].

[8] Aspect, "Aspect® Unified IP® Multi-choice Engagement," Aspect, 2013. [Online]. Available: https://www.aspect.com/globalassets/aspect-unified-ip-multichoice-engagement-ds.pdf. [Accessed 05 09 2018].

[9] Genesys, "Genesys Knowledge Management User Guide," 2017. [Online]. Available: https://docs.genesys.com/Documentation/ES/8.5.1/KMUser/FAQ?action=pdfbook&title=Documentation:ES:KMUser:FAQ:8.5.1. [Accessed 05 09 2018].

[10] C. C. Aggarwal and C. Zhai, "A Survey of Text Classification Algorithms," in *Mining Text Data*, Boston, MA, Springer, 2012, pp. 163-222.

[11] D. Jurafsky and J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (1st ed.), PTR, Upper Saddle River, NJ, USA: Prentice Hall, 2000.

[12] M. Ikonomakis, S. Kotsiantis and V. Tampakas, "Text Classification Using Machine Learning Techniques," *WSEAS transactions on computers,* vol. 4, pp. 966-974, 2005.

[13] W. Zhang, T. Yoshida and X. Tang, "Text classification based on multi-word with support vector machine," *Knowledge-Based Systems,* vol. 21, pp. 879-886, 2008.

[14] C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, New York, NY, USA: Cambridge University Press, 2008.

[15] S. Xu, "Bayesian Naïve Bayes classifiers to text classification," *Journal of Information Science,* vol. 44, no. 1, pp. 48 - 59, 2016.

[16] R. J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1993.

[17] A. Kataria and M. D. Singh, "A Review of Data Classification Using K-Nearest Neighbour," *International Journal of Emerging Technology and Advanced Engineering,* vol. 3, no. 6, pp. 354-360, 2013.

[18] V. N. Vapnik, The Nature of Statistical Learning Theory, Springer: New York, 1995.

[19] T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features," in *Machine Learning: ECML-98*, Berlin, Heidlberg, 1998.

[20] Statistica, "Global spam volume as percentage of total e-mail traffic from January 2014 to March 2018, by month," Statistica, [Online]. Available: https://www.statista.com/statistics/420391/spam-email-traffic-share/. [Accessed 15 08 2018].

[21] SpamAssassin, "Apache SpamAssassin," Apache, [Online]. Available: https://spamassassin.apache.org/index.html. [Accessed 15 08 2018].

[22] T. Ayodele, S. Zhou and R. Khusainov, "Email Reply Prediction: A Machine Learning Approach," in *Human Interface and the Management of Information. Information and Interaction*, San Diego, CA, 2009.

[23] L. Yang, S. T. Dumais, P. N. Bennett and A. H. Awadallah, "Characterizing and Predicting Enterprise Email Reply Behavior," *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval,* pp. 235-244, 2017.

[24] P. Liu and T.-S. Moh, "Content Based Spam E-mail Filtering," in *International Conference on Collaboration Technologies and Systems (CTS)*, Orlando, FL, 2016.

[25] Intersystems, "Caché Technology Guide," 2013. [Online]. Available: https://www.intersystems.com/wp-content/uploads/sites/8/CacheTechnologyGuide.pdf. [Accessed 05 09 2018].

[26] T. Murphy, "Email Open Interfaces - Developing Compliant Web Services," 2014. [Online]. Available: https://www.devconnectprogram.com/fileMedia/download/62af286e-3588-484e-94a1-635f6a0cfd3c. [Accessed 05 09 2018].

[27] I. H. Witten, E. Frank and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques (3rd ed.), San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 2011.

[28] Avaya, "Contact Center Performance Manager Data Dictionary," 2018. [Online]. Available: https://downloads.avaya.com/css/P8/documents/101020527. [Accessed 05 09 2018].

[29] J. Tang, H. Li, Y. Cao and Z. Tang, "Email Data Cleaning," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, New York, NY, ACM, 2005, pp. 489-498.

[30] B. Nicholson, J. Zhang, S. V. Sheng and Z. Wang, "Label noise correction methods," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Paris, France, 2015.

[31] B. Frénay and A. Kabán, "A comprehensive introduction to label noise," in *Proceedings of the 2014 European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2014)*, Bruges, 2014.

[32] T. G. Dietterich, "Ensemble Methods in Machine Learning," in *Proceedings of the First International Workshop on Multiple Classifier Systems*, London, UK, 2000.

[33] K. P. Shung, "Accuracy, Precision, Recall or F1?," Towards Data Science, [Online]. Available: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9. [Accessed 28 08 2018].

[34] N. Donges, "Evaluation Metrics for Classification," machinelearning-blog.com, 3 04 2018. [Online]. Available: https://machinelearning-blog.com/2018/04/03/evaluation-metrics-for-classification/. [Accessed 29 08 2018].

[35] ContactBabel, "The 2017-18 UK Contact Centre Decision-Makers Guide

(15th Edition)," 2017. [Online]. Available: https://www.cirrusresponse.com/wp-content/uploads/2017/11/UK-CC-DMG-2017-Cirrus-Platinum.pdf. [Accessed 05 09 2018].

8 Appendices

8.1 SQL Queries for data preparation

```
-- Bulk load the Features table from the AACC actions
insert into csd.Features
(
    ContactID,
    CustomerID,
    mailFrom,
    MailTo,
    Subject,
    QueueType,
    Body,
    Skillset,
    SkillsetID,
    RuleID,
    ClosedReason,
    OpenDuration
)
select
    contact,
    contact->Customer,
    contact->MailFrom,
    contact->MailTo,
    contact->OriginalSubject,
    contact->Queuetype->TextValue,
    substring(text, 0, 10000),
    contact->Skillset->Name,
    contact->Skillset,
    contact->Rule,
    contact->ClosedReason->Name,
    contact->OpenDuration
from cls.Actions
where Source->TextValue = 'EMail from Customer'

-- Set the recipient mailbox from the 'mailto' list
update csd.Features As feat
set mailbox = mailboxes.mailbox
from csd.features
    inner join
    (
    select "name" _ '@' _ "domainname" as mailbox
    from cls.Inboxes
    ) mailboxes
    on csd.features.mailto like '%' _ mailboxes.mailbox _ '%'
where feat.ContactID = csd.features.ContactID

-- Set the Followup flag to identify ongoing conversation
update csd.Features
set Followup = 1
where Subject like '%[<A_' _ CustomerID _ '>]%'


-- Set the AgentReply flag to identify is a contact was replied to by an
-- agent. Convoluted way to update a column due to performance issues
with
-- tempdb expansion
insert into csd.temp(contactID, Value)
select contact, count(ID) counter
from cls.Actions
where source->TextValue = 'EMail from Agent to Customer'
group by contact
```

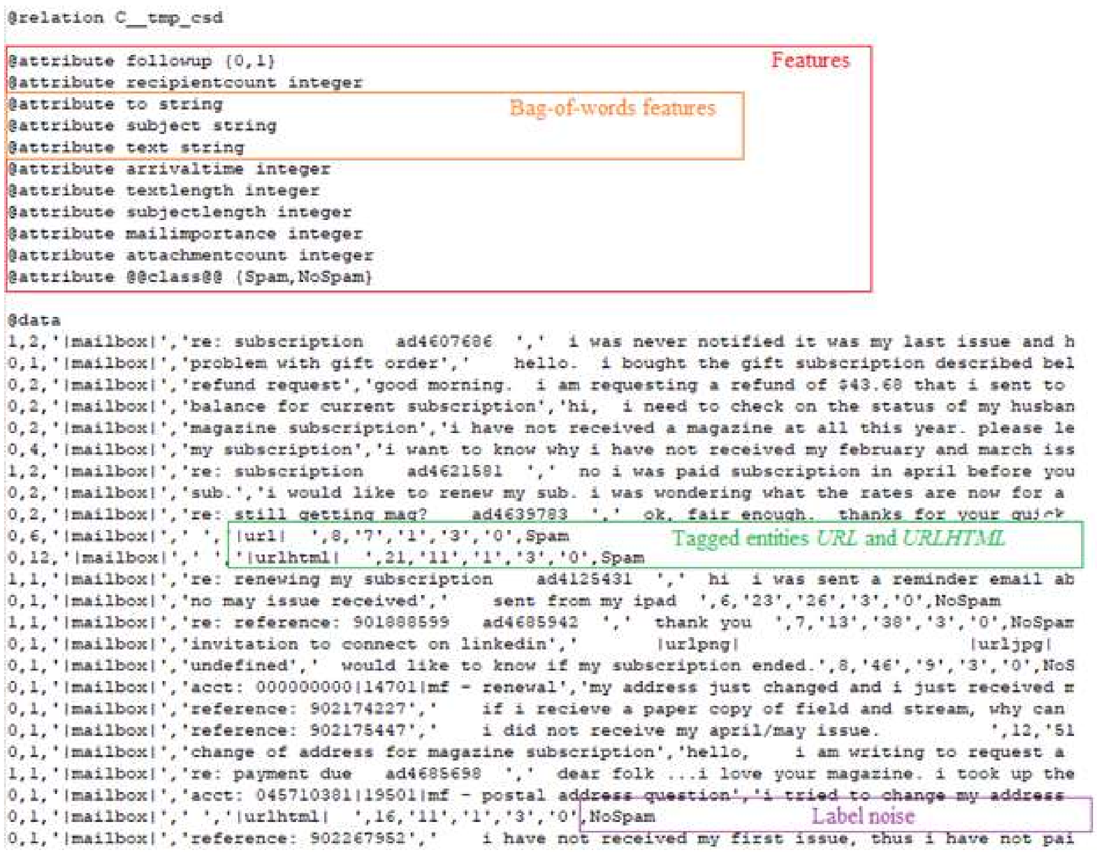
```
update csd.Features As feat
set agentReply = Value
from csd.features
    inner join csd.temp on csd.features.contactID = csd.temp.contactID

-- Set the MailToCount and MailCCCount to identify the number of
recipients
update csd.Features
set MailCCCount = (len(mailCC) - len(replace(mailCC, '@', '')))
from csd.Features f
    inner join cls.Contacts c on f.contactid = c.id

update csd.Features
set MailToCount = (len(MailTo) - len(replace(MailTo, '@', '')))
from csd.Features f

-- Set the Spam property
--(used as the @@class when generating ARFF format files for WEKA)
update csd.Features
set Spam = case
        when ClosedReason = 'Spam' then 'Spam'
        else 'NoSpam'
    end


-- Convoluted way to update a column due to performance issues with
-- tempdb expansion
update csd.Features As feat
set NewCustomer = 0

insert into csd.temp(contactID, value)
select cls.Contacts.ID,
    datediff('second', cls.Contacts.Arrivaltime,
cls.Customers.registerdate)
from cls.Contacts
    inner join cls.Customers on cls.Contacts.Customer = cls.Customers.ID

-- handle lag in email manager creating the customer and contact
delete from csd.temp where value < -900

update csd.Features As feat
set NewCustomer = 1
where contactid in (select contactid from csd.temp)
```

8.2 ObjectScript to extract ARFF files

```
ClassMethod SaveSpamToArff(split As %Integer = 60,
datasetSize As %Integer = 10000, name As %String)
{
  set dir = ##class(%File).NormalizeDirectory("c:\tmp\csd\" _
$zdate($horolog, 3) _ " - Spam - " _ name)
  do ##class(%File).CreateDirectoryChain(dir)

  set train = ##class(%FileBinaryStream).%New()
  set train.Filename = dir _ "\train_spam.arff"

  set test = ##class(%FileBinaryStream).%New()
  set test.Filename = dir _ "\test_spam.arff"

  set holdout = ##class(%FileBinaryStream).%New()
  set holdout.Filename = dir _ "\holdout_spam.arff"

  set head = "@relation C__tmp_csd" _ $c(13) _ $c(10) _
    "" _ $c(13) _ $c(10) _
    "@attribute followup {0,1}" _ $c(13) _ $c(10) _
    "@attribute recipientcount integer" _ $c(13) _ $c(10) _
    "@attribute to string" _ $c(13) _ $c(10) _
    "@attribute subject string" _ $c(13) _ $c(10) _
    "@attribute text string" _ $c(13) _ $c(10) _
    "@attribute arrivaltime integer" _ $c(13) _ $c(10) _
    "@attribute textlength integer" _ $c(13) _ $c(10) _
    "@attribute subjectlength integer" _ $c(13) _ $c(10) _
    "@attribute mailimportance integer" _ $c(13) _ $c(10) _
    "@attribute attachmentcount integer" _ $c(13) _ $c(10) _
    "@attribute @@class@@ {Spam,NoSpam}" _ $c(13) _ $c(10) _
    "" _ $c(13) _ $c(10) _
    "@data"

  do train.WriteLine(head)
  do test.WriteLine(head)
  do holdout.WriteLine(head)

  set sql = "select ID from csd.Features where id < 200000 "
  set rs = ##class(%Library.ResultSet).%New()
  set st = rs.Prepare(sql)
  set st = rs.Execute()

  set count = 0
  while (rs.Next() && (count < datasetSize))
  {
    set id = rs.Get("ID")

    set keep = $random(100)
    if (keep < 25)
    {
      set r = $random(100)
      if (r < split)
      {
        set file = train
      }
      else
      {
        set file = test
      }

      set feature = ##class(csd.Features).%OpenId(id)
      set cat = feature.Spam
      set follow = feature.FollowUp

      // Identify SPAM from first 1000 words
      set words = $piece(feature.MessageCleanTagged, " ", 1, 1000)
      set len = $length(feature.MessageCleanTagged)
      set len2 = $length(feature.Subject)
      set arrivaltime = feature.ArrivalTime

      // How many recipients (To + CC)
      set recipientCount = feature.MailToCount + feature.MailCCCount

      set arff = follow _ "," _ recipientCount _ "," _
..toArff("|mailbox|") _ ","
        _ ..toArff(feature.Subject) _ "," _ ..toArff(words) _ "," _
arrivaltime _ ","
        _ ..toArff(len) _ "," _ ..toArff(len2) _ "," _
..toArff(feature.Importance) _ ","
        _ ..toArff(feature.AttachmentCount) _ "," _ cat

      do file.WriteLine(arff)
      set count = count + 1
    }
  }

  set hold = datasetSize * (100 - split) / 100

  set sql = "select ID from csd.Features where len(messagecleantagged) >
4 and id > 200000 "
  set rs = ##class(%Library.ResultSet).%New()
  set st = rs.Prepare(sql)
  set st = rs.Execute()

  set count = 0
  while (rs.Next()&& (count < hold))
  {
    set id = rs.Get("ID")

    set keep = $random(100)
    if (keep < 25)
    {
      set feature = ##class(csd.Features).%OpenId(id)
      set cat = feature.Spam
      set follow = feature.FollowUp

      // Identify SPAM from first 1000 words
      set words = $piece(feature.MessageCleanTagged, " ", 1, 1000)
      set len = $length(feature.MessageCleanTagged)
      set len2 = $length(feature.Subject)
      set arrivaltime = feature.ArrivalTime

      // How many recipients (To + CC)
      set recipientCount = feature.MailToCount + feature.MailCCCount

      set arff = follow _ "," _ recipientCount _ "," _
..toArff("|mailbox|") _ ","
        _ ..toArff(feature.Subject) _ "," _ ..toArff(words) _ "," _
arrivaltime _ ","
        _ ..toArff(len) _ "," _ ..toArff(len2) _ "," _
..toArff(feature.Importance) _ ","
        _ ..toArff(feature.AttachmentCount) _ "," _ cat

      do holdout.WriteLine(arff)
      set count = count + 1
    }
  }

  set st = train.SaveStream()
  set rc = train.%Close()

  set st = test.SaveStream()
  set rc = test.%Close()

  set st = holdout.SaveStream()
  set rc = holdout.%Close()
}

ClassMethod toArff(str As %String)
{
  set str = $replace(str, "(", " ")
  set str = $replace(str, ")", " ")
  set str = $replace(str, "{", " ")
  set str = $replace(str, "}", " ")
  set str = $replace(str, "[", " ")
  set str = $replace(str, "]", " ")
  set str = $replace(str, "<", " ")
  set str = $replace(str, ">", " ")
  set str = $replace(str, """", " ")
  set str = $replace(str, "'", " ")
  set str = $replace(str, $C(0), " ")
  set str = $replace(str, $C(9), " ")
  set str = $replace(str, $C(13), " ")
  set str = $replace(str, $C(10), " ")
  set str = $zconvert(str, "L")
  set str = "'" _ str _ "'"
  quit str
}
```

8.3 Sample ARFF file

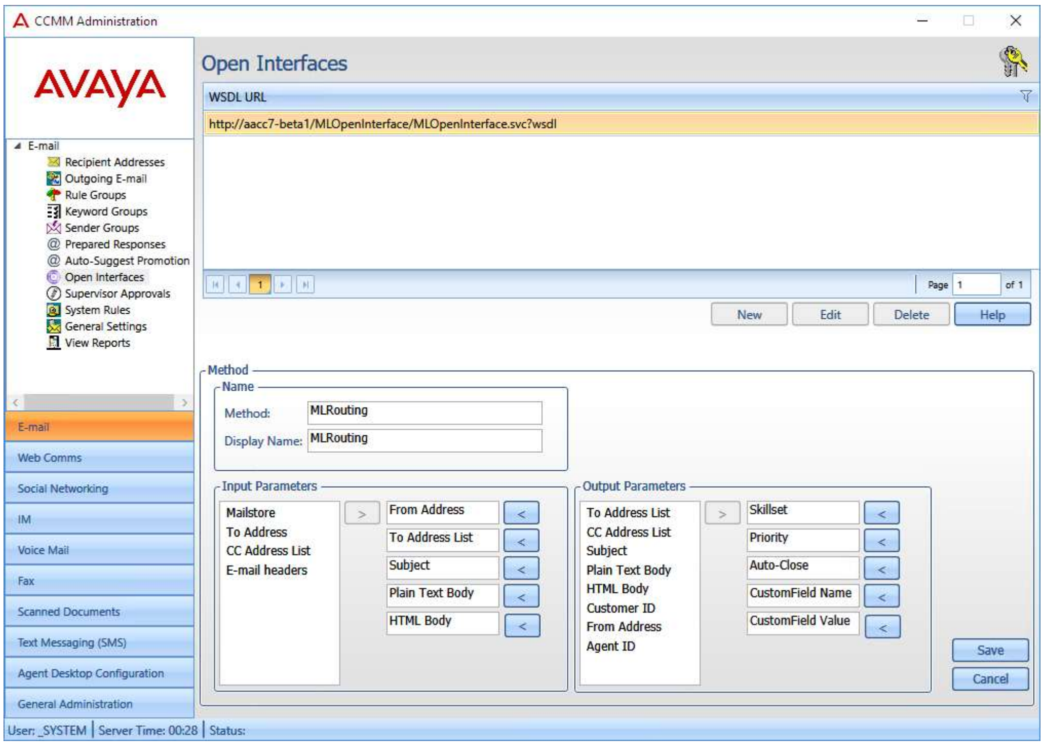
Taken from a screenshot showing a training ARFF file with customer information redacted. It shows the structure of the file. The first block identifies the fields and types of data in the csv @data section. Some Spam and NoSpam records are shown with samples of entity tagging and class label noise.



```
@relation C__tmp_csd

@attribute followup {0,1}
@attribute recipientcount integer
@attribute to string
@attribute subject string
@attribute text string
@attribute arrivaltime integer
@attribute textlength integer
@attribute subjectlength integer
@attribute mailimportance integer
@attribute attachmentcount integer
@attribute @@class@@ {Spam,NoSpam}

@data
1,2,'|mailbox|','re: subscription   ad4607686  ','  i was never notified it was my last issue and h
0,1,'|mailbox|','problem with gift order','    hello.  i bought the gift subscription described bel
0,2,'|mailbox|','refund request','good morning.  i am requesting a refund of $43.68 that i sent to
0,2,'|mailbox|','balance for current subscription','hi,  i need to check on the status of my husban
0,2,'|mailbox|','magazine subscription','i have not received a magazine at all this year. please le
0,4,'|mailbox|','my subscription','i want to know why i have not received my february and march iss
1,2,'|mailbox|','re: subscription    ad4621581  ','  no i was paid subscription in april before you
0,2,'|mailbox|','sub.','i would like to renew my sub. i was wondering what the rates are now for a
0,2,'|mailbox|','re: still getting mag?    ad4639783  ','  ok, fair enough.  thanks for your quick
0,6,'|mailbox|',' ',' |url|  ',8,'7','1','3','0',Spam
0,12,'|mailbox|',' ',' '|urlhtml|  ',21,'11','1','3','0',Spam
1,1,'|mailbox|','re: renewing my subscription    ad4125431  ','  hi  i was sent a reminder email ab
0,1,'|mailbox|','no may issue received','    sent from my ipad  ',6,'23','26','3','0',NoSpam
1,1,'|mailbox|','re: reference: 901888599   ad4685942  ','  thank you  ',7,'13','38','3','0',NoSpam
0,1,'|mailbox|','invitation to connect on linkedin','       |urlpng|                    |urljpg|
0,1,'|mailbox|','undefined','  would like to know if my subscription ended.',8,'46','9','3','0',NoS
0,1,'|mailbox|','acct: 000000000|14701|mf - renewal','my address just changed and i just received m
0,1,'|mailbox|','reference: 902174227','    if i recieve a paper copy of field and stream, why can
0,1,'|mailbox|','reference: 902175447','    i did not receive my april/may issue.          ',12,'51
0,1,'|mailbox|','change of address for magazine subscription','hello,    i am writing to request a
1,1,'|mailbox|','re: payment due   ad4685698  ','  dear folk ...i love your magazine. i took up the
0,1,'|mailbox|','acct: 045710381|19501|mf - postal address question','i tried to change my address
0,1,'|mailbox|',' ','|urlhtml|  ',16,'11','1','3','0',NoSpam
0,1,'|mailbox|','reference: 902267952','    i have not received my first issue, thus i have not pai
```

8.4 Configuring an Email Open Interface in AACC

The following images show how the email open interface is configured in AACC and added to the workflow process for the rules. A sample email manager log with the MLScore for Spam detection and Reply prediction returned from the Open Interface.



CCMM Administration

Open Interfaces

WSDL URL
http://aacc7-beta1/MLOpenInterface/MLOpenInterface.svc?wsdl

Page 1 of 1

New | Edit | Delete | Help

Method
Name
Method: MLRouting
Display Name: MLRouting

Input Parameters
Mailstore
To Address
CC Address List
E-mail headers

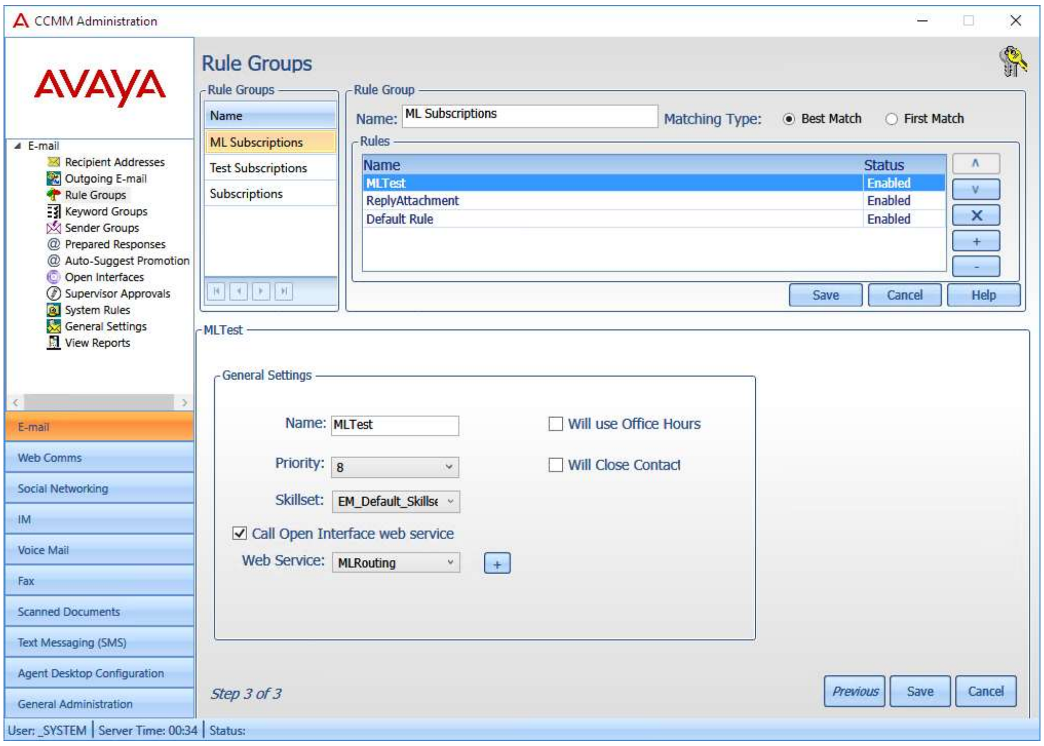
From Address
To Address List
Subject
Plain Text Body
HTML Body

Output Parameters
To Address List
CC Address List
Subject
Plain Text Body
HTML Body
Customer ID
From Address
Agent ID

Skillset
Priority
Auto-Close
CustomField Name
CustomField Value

Save | Cancel

User: _SYSTEM | Server Time: 00:28 | Status:



CCMM Administration

Rule Groups

Rule Groups
Name
ML Subscriptions
Test Subscriptions
Subscriptions

Rule Group
Name: ML Subscriptions
Matching Type: [✓ Best Match] First Match

Rules

Name	Status
MLTest	Enabled
ReplyAttachment	Enabled
Default Rule	Enabled

Save | Cancel | Help

MLTest

General Settings
Name: MLTest
Priority: 8
Skillset: EM_Default_Skillse
☐ Will use Office Hours
☐ Will Close Contact
☑ Call Open Interface web service
Web Service: MLRouting

Step 3 of 3

Previous | Save | Cancel

User: _SYSTEM | Server Time: 00:34 | Status:

Extract from the Email Manager log calling out to the ML Open Interface:

```
2018-09-01 18:38:41.690 EmailManager:MH      4912:13 14217   Debug   None

Web service invoker: process reply

2018-09-01 18:38:41.690 EmailManager:MH      4912:13 14186   Debug   None

Create open interfaces result set

2018-09-01 18:38:41.691 EmailManager:MH      4912:13 14221   Debug   None

Result   Set:   {SKILLSET=null,   PRIORITY=10,   CUSTOM_FIELD_VALUE=-3.930,   1.084,
AUTO_CLOSE=0, CUSTOM_FIELD_NAME=MLScore}
```