The International Society of Precision Agriculture presents the
16th International Conference on Precision Agriculture
21–24 July 2024 | Manhattan, Kansas USA

Explainable Neural Network Alternatives for AI Predictions: Genetic Algorithm Quantitative Association Rule Mining

Everett, Mary[1]

[1]**University of Idaho, 875 Perimeter Drive, Moscow, Idaho, USA**

A paper from the Proceedings of the
16th International Conference on Precision Agriculture
21-24 July 2024
Manhattan, Kansas, United States

Abstract.

Neural networks in one form or another are common precision agriculture artificial intelligence techniques for making predictions based on data. However, neural networks are computationally intensive to train and to run, and are typically "black-box" models without explainable output. This paper investigates an alternative artificial intelligence prediction technique, genetic algorithm (GA) quantitative association rule mining (QARM), which creates explainable output with impacts directly quantified in the existing dataset. QARM takes one or more data features of a dataset and restricts the value of the feature between two bounds. These feature(s) are then associated with a particular outcome from the data (such as frost). The resulting rule's correlation can then be quantified in terms of the support (how often it is seen in the dataset), confidence (how often it co-occurs with the outcomes), and lift (how much more or less often we see this than expected). The genetic algorithm component finds the optimal features and value bounds to maximize the significance of the correlation. Generating quantitative association rules with genetic algorithms is not a new method, however, it is not commonly used and likely deserves more attention in the explainable AI realm. Additionally, this paper extends the technique by adding a sequence to each feature to analyze time data. Time steps were added to value bounds to determine what time range in the past was most significant to the correlation. This technique was compared with neural network predictors for multivariate time-sequence weather data in two scenarios: the open Jornada Basin LTR dataset for the purpose of predicting frost one day ahead, and a custom-collected dataset from Laurel Grove Wine Farm in Winchester, Virginia to predict frost in 5 minute intervals. The QARM GA technique had comparable performance to the neural network methods in the Jornada Basin dataset (0.803 F1 statistic score on the dataset compared to 0.847 for the neural network) while generating highly interpretable and computationally cheap-to-implement prediction rules. For the Laurel Grove Wine Farm study, both techniques were limited in overall results, but the genetic algorithm outperformed the neural network method (0.489 F1 score for the QARM GA method compared to 0.217 F1 score for the neural network). The results of these experiments indicate quantitative association rule mining is worth further investigation for artificial intelligence in precision agriculture.

Keywords.
frost prediction, quantitative association rule mining, genetic algorithms, microclimates

Introduction

Artificial intelligence has been identified as an emerging technology with high potential for improving precision agriculture forecasts. However, while many artificial techniques are performant on large amounts of data, they can be 'black box' models whose underlying operation is difficult to explain. Increased adoption of artificial intelligence techniques for precision agriculture will likely need a corresponding increase in trustworthy models.

Significance

This study aims to compare a more conventional AI technique (Long Short Term Memory Network, also called an LSTM) with a less-studied but more explainable Quantitative Association Rule Mining Genetic Algorithm (QARM GA) technique. Both models can be used as predictors, with LSTM models generally boasting impressive performance where other techniques fail. However, the LSTM models do not generally produce highly interpretable results. Additionally, when data is sub optimal, model performance is often less generalizable but is difficult to catch. QARM GA models, on the other hand, generate highly interpretable rules that directly relate the data to the prediction model. This study evaluates both techniques in terms of performance and interpretability for use in precision agriculture site-specific frost prediction.

Prior Work

The prior work section of this study will mostly focus on work in quantitative association rule mining with genetic algorithms, as a very large body of preexisting work exists for pure microclimate prediction tasks. In Alvarez & Jacinto (2012), this technique was used for large database rule mining in place of the preexisting apriori algorithm for rule mining, and genetic algorithms were used in Yan, Chengqi, & Shichao (2009) to avoid specifying a minimum support level. Alataş & Erhan (2006) mine both positive and negative association rules, where most algorithms only mine positive rules. Salleb-Aouissi et al. (2013) created QuantMiner, which is a tool for exploratory data analysis using these techniques. In Sinisterra-Sierra, Salvador, & Miriam (2023) genetic algorithms with quantitative association rule mining is used to find potential causal rules linking medical conditions with COVID-19. Martín et al. (2014) and Almasi & Mohammad (2015) both use multi-objective genetic algorithms in their mining. Li et al. (2022) uses this technique with many different types of data features to explore ocean dynamics. For time sequence data, Troncoso-García et al. (2023) uses previous time steps to predict future time steps for an energy consumption application. Most similar to this study, Martínez-Ballesteros et al. (2011) uses this technique in conjunction with a singular time sequence boundary for all rule features to predict ozone based on weather characteristics.

Methods

Quantitative Association Rules

Rule Format

Quantitative Association Rules are formed from an antecedent (also called a rule body) and a consequent (also called a rule head). They take the form outline in Equation 1.

feature_1[bound_i,bound_j],feature_2[bound_i,bound_j]...feature_n[bound_i,bound_j] =>consequent[bound_i, bound_j] (1)

Features 1 through n are features of the data and (in tabular data) would generally be represented in the columns. The consequent is also a feature of the data, typically the feature of interest that the stakeholder would like to predict. The bounds of each feature *i* and *j* are the lower and upper values respectively that a feature can take in the rule. Note that *i=j* is possible if the stakeholder is looking for features that take a discrete value (which in this paper is true for the consequent). Bounds *i* and *j* do not have to be the same across all the features in a rule, however.

The above rule indicates an association between the antecedent (features 1 through n) and the consequent feature. A rule with multiple features in the antecedent indicates all features take the associated values in whatever axis of interest is being operated in together. The best way to illustrate this is with an example. Let's say one has a dataset of weather values and a dataset of frost events for a farm. You want to determine associations between weather patterns the day before and frost events. You might come up with a rule of this form presented in Equation 2.

temperature[11, 30], humidity[30, 50] => frost[1, 1] (2)

This example rule states that an observed temperature value between 11 and 30 degrees and an observed humidity value between 30 and 50 percent has an association with a frost event (coded as "1" in this dataset) the next day.

Rules identify associations, but it is important to determine how "good" associations rules are with metrics. These metrics indicate the strength and relevance of a given rule which helps determine the applicability of the rule to real-world problem solving.

Rule metrics

There are many possible metrics for quantitative association rules, but this paper will focus on 3 of them in particular; support, confidence and lift. The support of the rule is given by Equation 3.

support = number of items with rule/total number of items (3)

Support looks at the number of items/occurrences in the dataset (typically represented by rows in tabular data) that have all items in the rule at their associated values present divided by the total number of items/occurrences in the dataset. This indicates how often the whole rule (antecedent and consequent) is seen in the dataset at all.

Using the previous example, if the rule had a 0.2 (20%) support value and there are 1000 items in the dataset, that would mean that there was an occurrence of temperature between 11 and 30 degrees and humidity between 30 and 50% and a frost event the next day in 200 of the 1000 cases. The maximum value for support is 1 (100% of the items in the dataset have the features at the given value and the associated consequent).

The confidence of the rule is given by Equation 4.

confidence = number of items with whole rule/number of items with just the antecedents (4)

Confidence differs from support in that while support looks at percentage of all dataset items with the whole rule, confidence looks at what percent of items with the antecedent features (with the associated bounds) also have the consequent. The maximum value for the confidence is also 1, meaning 100% of items with antecedent features also have the consequent feature.

For the example rule, if the rule had an 80% confidence, it means of all occurrences in the dataset with a temperature between 11 and 30 degrees and humidity between 30 and 50%, 80% of the time the occurrence also had a frost event the next day.

Lift is slightly more difficult to explain but is a measure of expectation of rule occurrence in the dataset. It is calculated in Equation 5.

*lift =(% of items with entire rule)/(% of items with rule antecedents) * (% of items with rule consequents)* (5)

Lift is essentially measuring how different the co-occurrence of the rule antecedents and consequent together is from the expected co-occurrence of these features based solely on chance (the denominator). A lift value very close to 1 indicates these rule values occurred about as often as expected, meaning they probably don't have a correlation significance outside of chance. Higher positive lift values indicate that the antecedents and consequent occur together more often than would be expected by chance. Lift values close to 0 indicate that the antecedents and consequent occur together less often than expected by chance.

For the example rule, if the rule had a lift of 3.1, it would indicate that a temperature between 11 and 30 degrees and a humidity value between 30 and 50% occurred with a frost event the next day 3.1 more times than would be expected by pure chance.

There are metrics used in quantitative association rule mining other than those listed above. Additionally, the number of occurrences with the antecedents and number of occurrences with the consequent can often be useful metrics to explore different facets of the data (for instance, the number of whole rule occurrences ratio of all consequent possibilities.)

Ideally, a "good" rule is a balance of support, confidence, and lift. A rule with high support indicates that there were enough occurrences of the rule in the dataset to ensure this rule is not a fluke. A rule with high confidence indicates a strong relationship between the antecedents and the consequent. A rule with high lift indicates this relationship between the antecedents and the consequent is statistically meaningful.

Sequence Extension of Quantitative Association Rules

Sequence data is common in determining weather patterns. To extend quantitative association rule mining for sequence data, it is useful to determine when a feature in the antecedent occurred in addition to the bounds on its values. Therefore, to each feature in the antecedent, we also add bounds on its occurrence in sequence, presented in Equation 6.

feature_1[bound_i, bound_j][seq_i, seq_j] feature_2[bound_i, bound_j][seq_i, seq_j]... feature_n[bound_i, bound_j][seq_i, seq_j] => consequent}[bound_i, bound_j] (6)

Sequence *i* and *j* markers represent the upper and lower bounds on where the antecedent features at the given values can occur in the sequence before the consequent occurrence.

Now if we have the rule if Equation 7:

temperature[11, 30][2,3] humidity[30, 50][5,6] => frost[1, 1] (7)

This would indicate that the temperature was between 11 and 30 degrees between 2 and 3 timesteps before and humidity was between 30 and 50% between 5 and 6 timesteps before the frost event in this association rule.

Genetic Algorithms

Genetic Algorithms involve sets of techniques that allow for smart searching of a solution space in order to optimize some problem. They are especially useful when brute force searching of a space is infeasible due to the high number of possible parameters and values.

Genetic algorithms involve the following concepts:

- Individual: This is a candidate solution of interest. The encoding of the solution is highly context-dependent. This is also sometimes referred to as a genome.
- Fitness: This is the 'score' of the performance of the candidate solution, generated by a fitness function. How the fitness function determines the suitability of an individual solution is also highly context dependent and often heuristic methods are used.
- Population: This is a collection of individuals. Populations are created initially and are changed via individuals through the algorithm run.
- Mutation: Mutation is the ability of the individual to change its encoding, often on one or more properties.
- Selection and crossover: This is the ability of a population to change by selecting individuals for the next population based on some criteria. Crossover involves mixing different individual encodings to generate new encodings.
- Tournament Selection: Tournament selection chooses a random subset of individuals from a population. Of those individuals, the highest fitness individual is selected to continue to the next generation.

- Diversity: Diversity is the degree of difference for individual encodings in the population from one another.

The genetic algorithm uses the following run sequence:

1. A population of individuals is initialized. The individual encodings in the population may be initialized randomly or based on some heuristic, but individuals should usually be different from each other.
2. Each individual in the population is scored on fitness based on the fitness function.
3. Selection of individuals based on a process (like tournament selection) is conducted.
4. A subset of the individuals have their encodings randomly mutated.
5. The process is repeated for a set number of generations or until a desired state is reached.

There are many hyperparameters in a genetic algorithm, similar to other artificial intelligence methods like neural networks. Genetic algorithms benefit from domain expertise in setting hyperparameters like population size, diversity needs, tournament selection size, mutation rate, mutation amount, and more.

Mining Quantitative Association Rules with Genetic Algorithms

For this study, genetic algorithms are used to mine high performance quantitative association rules. In this setup, individuals in the population are rules. They are initialized randomly with random features and random bounds on both feature values and sequence values within a set range. Mutation options can include adding or subtracting a feature, changing a feature's value bounds, or changing a feature's sequence bounds. Fitness is generated by a combination of the rule's associated support, confidence, and lift values.

Experiments and Results

In this section, two sets of experiments will be discussed. The first involves predicting next-day frost occurrences over the Jornada Basin dataset, reproduced from the researcher's dissertation. The second set of experiments involves frost prediction from weather data collected at Laurel Grove Wine Farm in Winchester, Virginia.

Prior work – Jornada Basin Frost Study

Research on genetic algorithms quantitative association rule mining for the Jornada Basin was conducted by the University of Idaho in summer 2023 and published in the investigating PhD student's dissertation (Everett, 2023). It is discussed in detail here and partially reproduced as the groundwork indicating the potential for technique.

Data

The Jornada Basin dataset involves weather data from the Jornada Basin Long Term Research Center in the Chihuahuan desert, New Mexico, USA (Yao et al., 2023). This dataset provides temperature/humidity, wind, and solar radiation data as daily summaries (min, max, average) for multiple years across 30 sites. Data was taken from 2013 to 2022 for the study, and since different sites sometimes had different reporting dates, the amount of data for each site varied. Large time frames with missing observations were eliminated and small time frames with missing observations were forward filled with data from the last valid observation. Frost events in this case were created based on whether or not the minimum temperature for the day had met or gone below 32 degrees F. Frost events were predicted on a reporting-site specific basis, and experiments were conducted using data only from the site or by using data from all sites together. Data was normalized for the LSTM models but not for the genetic algorithms, which used the standard deviation of the parameters to create mutations.

Comparison LSTM Model

Multiple LSTM models were run for the dissertation study, which in part tested various types of autoencoder pre-processing. However, for the most part, the models with and without autoencoder processing did not have statistically different differences in performance from one another. The best performing model achieved an F1 statistic of 0.847. It used 30 days of input with one day of output, and a custom autoencoder structure where first all datastreams for all sites were separately encoded, then all datastream latent space output was fed as input to another autoencoder by location, and the subsequent location latent spaces were fed to an LSTM model. The LSTM model had 64 nodes followed by a dropout layer of 20% and used the sigmoid function as its final activation. Early stopping was allowed after 10 epochs of the 100-epoch run. The mean squared error was used as the loss function. Other models that used only the 64 node LSTM layer and dropout performed similarly. The test set was split as 20% of the whole, with the validation set as 20% of the remainder. The batch size of the network was 32 samples, and Adam optimization was used.

QARM GA Model Setups

This section will discuss more in detail the QARM GA setups compared to the LSTM model setups.

For the genetic algorithms, there were two types of prediction models: one that tried to predict frost for one site by using only that site's data, and one that tried to predict frost for one site by using data from all sites concurrently.

Multiple variations of each model was run, broken up into slates that shared parameters by alphabetical letter. Letters A, C, E, G, and M used only specific site data as input, while letters B, D, F, H, and N used all site data as input. Additionally, these models were run with sequence and non-sequence models. For non-sequence models, the algorithm tried to generate rules using data only the day before a potential frost event, while in the sequence models, the algorithm could use any number of historical days up to one day before the frost event. The parameters for the sequence and non-sequence models are in Figures 1 and 2.

	Population Size	Generations	Diversify	Reseed	Fitness Function Index	Range Penalty Index
A/B						
1	150	150	T	T	1	0
2	150	150	T	T	2	0
3	150	150	T	F	1	0
4	150	150	T	F	2	0
C/D						
1	200	150	T	F	2	0
2	200	150	T	F	5	0
3	300	250	T	F	2	0
4	300	250	T	F	5	0
E/F						
1	100	100	T	F	2	0
2	150	150	T	T	2	0
3	100	100	T	F	5	0
4	150	150	T	T	5	0
G/H						
1	150	150	T	F	2	1
2	150	150	T	T	2	1
3	150	150	F	T	2	1
4	150	150	F	F	2	1
M/N						
1	200	150	T	F	2	1
2	200	150	T	F	5	1
3	300	250	T	F	2	1
4	300	250	T	F	5	1

Figure 1: The Parameters for non-sequence models for QARM GA Jornada Basin Models.

The population size and generations related directly to the genetic algorithm parameters discussed when explaining the algorithm. The diversify parameter is true when the rules kept across population runs have enforced diversity of different parameters. The reseed parameter is true when killed population members are replaced from the best performers list rather than randomly reinitialized. The range penalty reduces fitness if parameters in rules encompass too broad a range, and the sequence penalty works similarly for sequence rules with too broad a time frame. The calculations for the range and sequence penalty indexes as well as the fitness functions are presented in the Appendix.

The experiments and associated parameter sets above were run for each site 3 separate times.

Results

The results for the QARM GA models is presented in Figure 3 as the best F1 statistic across all sites for both the sequence and non-sequence models.

Overall the best F1 score across the models was 0.803, for a non-sequence QARM GA model predicting next-day frost from one day ahead with data from all sites. This is not quite as good as the LSTM model, with a score of 0.847, but is quite close and generated highly explainable rules. The sequence runs of the model also had slightly more stable performance across sub-optimal hyperparameter sets compared to the sequence models. A non-sequence sample rule for site frost prediction is presented in Figure 4 and for sequence in Figure 5. For the non-sequence rule, the C GRAV average air temperature measure was the best predictor of frost for the C Cali site, and in the sequence rule, the minimum air temperature for the P SMAL site was the best predictor of the frost in the M WELL site, 1-10 days beforehand.

	Population Size	Generations	Diversify	Reseed	Fitness Function Index	Range Penalty Index	Sequence Penalty Index
A/B							
1	150	150	T	T	1	0	2
2	150	150	T	T	1	0	3
3	150	150	T	T	2	0	2
4	150	150	T	T	2	0	3
C/D							
1	200	150	T	F	2	0	3
2	200	150	T	F	5	0	3
3	300	250	T	F	2	0	3
4	300	250	T	F	5	0	3
E/F							
1	100	100	T	F	2	0	3
2	150	150	T	T	2	0	3
3	100	100	T	F	5	0	3
4	150	150	T	T	5	0	3
G/H							
1	150	150	T	F	2	1	3
2	150	150	T	T	2	1	3
3	150	150	F	T	2	1	3
4	150	150	F	F	2	1	3
M/N							
1	200	150	T	F	2	1	3
2	200	150	T	F	5	1	3
3	300	250	T	F	2	1	3
4	300	250	T	F	5	1	3

Figure 2: The Parameters for sequence models for QARM GA Jornada Basin Models.

Run – Non Sequence	Param 1 Best F1	Param 2 Best F1	Param 3 Best F1	Param 4 Best F1
A	0.490	0.789	0.330	0.508
B	0.614	0.796	0.395	0.581
C	0.523	0.440	0.536	0.465
D	0.583	0.517	0.500	0.498
E	0.505	0.781	0.403	0.715
F	0.575	0.803	0.495	0.741
G	0.534	0.784	0.796	0.530
H	0.615	0.800	0.794	0.560
M	0.500	0.444	0.567	0.449
N	0.574	0.517	0.533	0.525
Run – Sequence				
A	0.503	0.563	0.730	0.751
B	0.529	0.631	0.752	0.767
C	0.693	0.585	0.694	0.597
D	0.720	0.622	0.674	0.618
E	0.667	0.746	0.591	0.699
F	0.701	0.764	0.624	0.691
G	0.671	0.716	0.699	0.687
H	0.707	0.771	0.774	0.688
M	0.693	0.575	0.693	0.594
N	0.696	0.606	0.716	0.606

Figure 3: Results of best F1 statistic across sequence and non-sequence models for Jornada data.

While not quite as performant as an LSTM model, the ability of the QARM GA to be directly understood and perhaps capture between-site interactions was considered useful for further study. For this reason, further testing took place for a custom collected dataset at Laurel Grove Wine Farm.

Laurel Grove Wine Farm Frost Study

Laurel Grove Wine Farm is a vineyard located in Winchester, Virginia, owned by research partners Dustin and Jaclyn Mommen. The planned vineyard encompasses 120 acres, but only one block was planned to be planted in April 2024. This block was divided into 4 zones based

```
"npp_c_cali": [
    {
        "parameters": {
            "npp_c_gravAir_TempC_Avg": {
                "lower_bound": -2.943737408868765,
                "upper_bound": 6.914454394126818
            }
        },
        "support": 0.06873879258816497,
        "confidence": 0.7516339869281046,
        "lift": 7.310951512387901,
        "fitness": 0.7134103174657996
    }
```

Figure 4: A non-sequence example rule from the Jornada study. Temperature in F.

```
"npp_m_well": [
    {
        "parameters": {
            "npp_p_smalAir_TempC_Min": {
                "lower_bound": -9.227515495969113,
                "upper_bound": -5.425585853230013,
                "seq_lower_bound": 0,
                "seq_upper_bound": 9
            }
        },
        "support": 0.1972504482964734,
        "confidence": 0.7051282051282052,
        "lift": 3.080103099685345,
        "fitness": 0.9585987087181179
    }
```

Figure 5: A sequence example rule from the Jornada study. Temperature in F.

temperature/humidity sensors placed along contours for each sub-block in the planting area, and frost events were analyzed for each zone (alpha, bravo, charlie, and delta). Based on the number and length of frost events, the alpha, bravo, and charlie zones were selected for actual planting in April 2024. However, for the study, the delta frost events were chosen for prediction as these generally precede frost events in most of the other zones and can provide additional time to react. Frost was considered any temperature at below 29 degrees F for this vineyard.

Data

Laurel Grove Wine Farm uses a high-density real-time wireless sensor network across its 120 acres of land. Data is taken from fully-featured weather rack stations and from temperature/humidity Bluetooth KKM K6P beacons in higher densities. However, this is a real-world setup for sensor network that is around 2 years old; there are often data outages or sensor issues. This creates a very noisy dataset where there were periods of high data amounts for sensors followed by no data. Some weather stations have issues reporting on some of the sensors, additionally.

Data was taken from 25 different weather station sensors and 17 KKM K6P temperature/humidity beacon sensors. All sensors are supposed to report their information every 5 minutes indefinitely, but due to sensor issues and/or network outages this is not always the case. The features reported by the weather stations include temperature, humidity, average wind speed, gust wind speed, wind direction, light, and uv. The features reported by the KKM K6P sensors in this study included temperature and humidity.

Figure 6 displays the layout of the KKM K6P sensors and weather rack sensors across Laurel Grove Wine Farm. The weather rack sensors were included over a wide area, while the KKM K6P sensors were included over a much more limited area due to time constraints of the study.

Data was normalized for the LSTM network but not for the QARM GA setup, which did not require it. For these experiments, April and March 2023 frost events were provided as the training set, with March 2024 frost events left as the test set.

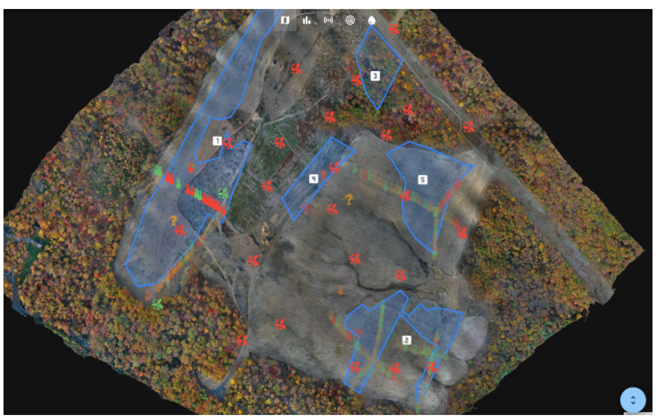

Figure 6: Icons for the KKM K6P sensors and weather racks sensor positions.

LSTM Neural Network Setup

Figure 7 lists the parameter sets for the LSTM experiments. There were 71 total, but only those with eventually non-zero f1 scores are reproduced here for brevity. The model number refers to the type of model setup which is outlined in the Appendix. The Dataset Backfilled parameter refers to data interpolation for the set - most missing data was forward filled by the last value in the data, however some sensors had large amounts of data missing at the onset. If this parameter is True, the dataset was backfilled with the last valid value; if False, the dataset was cut to take out the missing initial data. The Features Shortened parameter refers to the number of features used. For most experiments (Features Shortened set to False), all 114

Run Number	Model Number	Dataset Backfilled	Features Shortened	Batch Size	Epochs	Hrs input	Hrs offset	Monitoring
36	3	F	F	32	200	20	3	Recall
53	3	F	T	64	100	20	3	Recall
62	3	F	T	32	100	10	1	Recall
63	1	F	T	64	100	10	1	Recall
64	2	F	T	64	100	10	1	Recall
65	3	F	T	64	100	10	1	Recall
67	2	F	T	32	50	10	1	Recall
71	3	F	T	32	200	10	1	Recall

Figure 7: LSTM Model parameter setups with non-zero F1 score.

features were used, but for runs 48-71, a subset of 19 recurring features identified as related to frost events by the QARM GA experiments were used. The batch size parameters and epochs directly correspond to the neural network hyperparameters. The hours input field relates to the number of hours of input (in 5 minute intervals) which was given to the algorithm (20 hours = 240 samples) and the hours offset indicates how far ahead the network predict a single frost event. For a sample set starting at 8:00am one day, with a 20 hour input and 3 hour offset, the prediction would be relevant for 7:00am the next day. The monitoring parameter refers to the feature monitored to load in the best performing model, either the loss of the network (which used Binary Cross-Entropy as the loss function) or the recall of the network.

For the LSTM networks, both training and test sets used all frost event occurrences. This QARM GA algorithm only trained on the start of frost events, which is explained further in that section.

QARM GA Neural Network Setup

Experiment	Mutation Rate	Mutation Amount	Initial Rule Limit	Add /Subtract Percent	Change Percent	Max Mutation Tries	Fitness Function Index	Top Rules	Generations	Tournament Size	Sequence Limit	Sequence Offset
37	50	100	4	30	70	5	12	20	50	4	120	13
51	50	100	4	30	70	5	10	20	50	4	240	36
54	50	100	4	30	70	5	13	20	50	4	240	36
55	50	100	4	30	70	5	10	20	100	4	240	36
63	50	100	4	30	70	5	17	20	50	4	240	36
64	50	100	4	30	70	5	18	20	50	4	240	36
66	50	100	4	30	70	5	18	20	25	4	240	36
67	50	100	4	30	70	5	17	20	100	4	240	36
72	50	100	6	30	70	5	18	20	50	4	240	36
77	50	100	4	30	70	5	18	20	50	4	240	36
88	50	100	4	30	70	5	17	40	50	4	240	36
89	50	100	4	30	70	5	18	40	50	4	240	36

Figure 8: Parameter setups for 12 best QARM GA Model runs.

Figure 8 displays the experiment parameters for QARM GA Runs. There were 75 total runs, but only the 12 best are reproduced here for brevity. Unlike the LSTM models, data did not need to be normalized or filled, as the algorithm can handle different data intervals and missing data. The QARM GA models were trained with only the first frost event counted as a true frost event, to preclude the possibility of the algorithm correlating the reduction in temperature alone with a frost event, which would be possible for the sustained frost events. The test set included all frost events, however, to get a better reading on the model's predictive power.

The parameters for the QARM GA that were modulated include the mutation rate (percent of time the rule was chosen for mutation), the mutation amount (as a percent of the standard deviation of the chosen rule parameter), the initial rule limit (the maximum number of parameters that can be initialized for a rule), the Add Subtract and Change Percents, which must sum to 100 and indicate the percentage of the time a parameter is added/subtracted or the bounds are changed in a mutation event, the maximum mutation tries for validness before a mutation operation is abandoned, the index of the fitness function used (see Appendix for calculation), the number of top rules kept across generations, the number of generations to run, the tournament selection size, the limit of the sequence backward for each parameter, (in 5 minute intervals) and the sequence offset (same as corresponding LSTM offset, in 5 minute intervals).

All QARM GA models are modeled primarily on the F1 statistic, though precision and recall are also tracked. The number of ultimate rules tracked correspond to the number of top rules, so that parameter ultimately controls how many "sub-models" each run generates.

Results

The results for the best LSTM models are displayed in Figure 9 and for the best QARM GA models in Figure 10. For the QARM GA models, the best rule based on the F1 statistic has its metrics displayed.

Run Number	Model Number	Precision	Recall	F1 Score
36	3	0.313	0.120	0.173
53	3	0.024	0.013	0.017
62	3	0.056	0.771	0.105
63	1	0.142	0.452	0.217
64	2	0.031	0.061	0.041
65	3	0.036	0.074	0.049
67	2	0.017	0.168	0.031
71	3	0.044	0.082	0.057

Figure 9: Results for Laurel Grove LSTM models.

It can be seen first of all that neither model performed very well, with the highest F1 statistic across both models reaching only 0.489. There are likely several reasons for this. To begin, the model is predicting frost in 5 minute increments, which is very high fidelity, and lends itself to many false positives in the case of the QARM GA models. Of more importance, perhaps, is the ability of the model to give decent warning time before the start of a given frost event, but this is difficult to evaluate directly, and would be a good topic for future work. The length of the frost event was also important, and particularly difficult to predict for these models. These experiments also only had two months of training data (albeit high resolution data) and one month of test data. In addition, this was a custom collected, very noisy dataset with plenty of periodic data outages and uneven distributions of readings across sensors.

Experiment	Precision	Recall	F1	Index Best
37	0.345	0.843	0.489	5
51	0.321	0.481	0.386	5
54	0.354	0.258	0.298	17
55	0.366	0.258	0.303	13
63	0.213	0.420	0.283	9
64	0.251	0.420	0.314	13
66	0.472	0.226	0.306	9
67	0.530	0.258	0.347	10
72	0.257	0.415	0.318	3
77	0.508	0.258	0.342	8
88	0.196	0.508	0.283	14
89	0.564	0.258	is	17

Figure 10: Results for Laurel Grove QARM GA models.

Ultimately, the LSTM models performed very poorly on this task. The highest F1 statistic occurred was only 0.217, on a model using 10 hours of input with a 1 hour offset. This model was able to predict only 45% of the frost events. The overwhelming majority of the models did not predict any true frost events at all.

In contrast, most of the QARM GA models had at least one rule with some predictive power. The best F1 statistic occurred in run 51, at 0.489 (recall of 84.3%). Another 11 rules had an F1 score above 0.28.

While the F1 scores are somewhat unimpressive for both models, the explainability of the QARM GA rules points to some interesting potential insights about the way land features and weather might interact. The best-scoring rule for the QARM GA function indicated that in a particular moment in time, frost is predicted if the temperature on one particular KKM K6P sensor was between 35.4 and 28.4 degrees F at any point between 65 and 320 minute prior. While only a 34.5% precise rule, it predicted 84.3% of the frost events, and as can be seen in the graph in Figure 11, was able to predict the frost ahead of time. This could indicate that in most cases this sensor gives about an hour heads-up with temperature drops that lead to impactful freezes. Figure 12 displays the location of the temperature sensor, which is adjacent to the planted blocks.

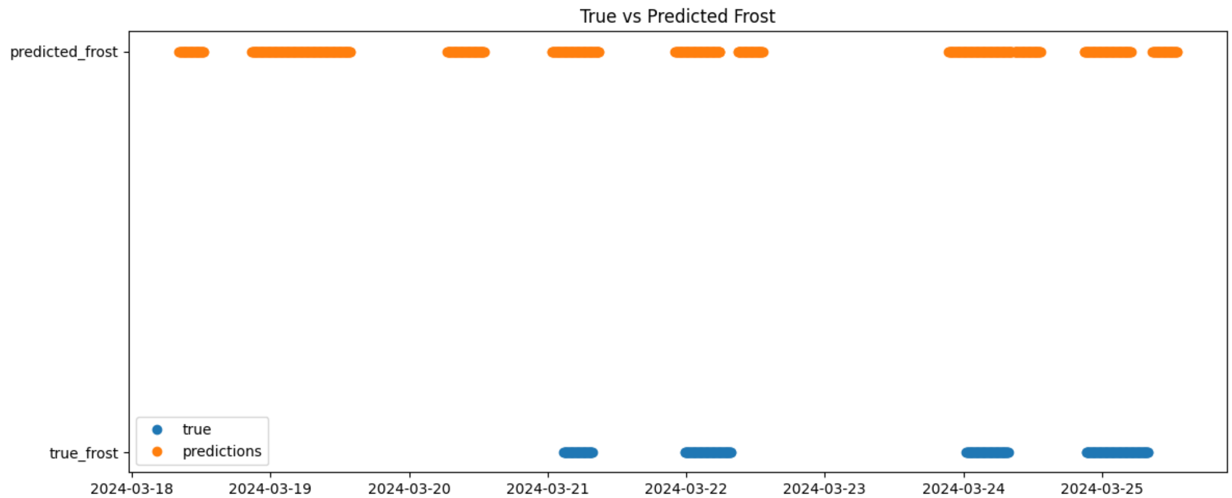

Figure 11: True vs. Predicted Frost evens for Laurel Grove Wine Farm based on rule predictor with best F1.

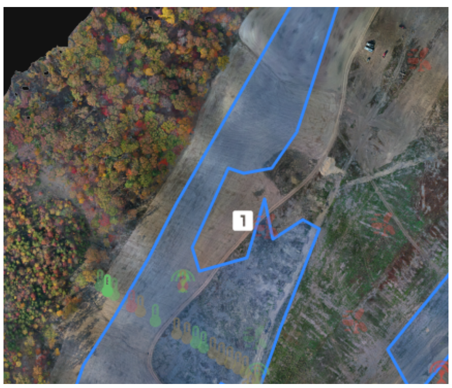

Figure 12: Location of sensor in best performing QARM GA rule (opaque).

Another rule stated that a frost event occurs if in the preceding 275 - 1205 minutes, the wind direction on one of the weather stations (weather rack 11) was either non-existent (0 degrees) or northeast (0.1-78.7 degrees). This rule only has a recall of 48.1% and an F1 of 38.6%, but is one of several rules that references wind properties of this particular sensor and their relation to frost events. This is an interesting result, as this sensor is considerably further away from the planted area Further work should explore whether there are environmentally sound reasons for this weather relation pattern between these micro-climates.

Limitations of the Study

This study was run with very limited data over a small area with a limited set of parameters. There might be considerably better model hyperparameters for both the LSTM models and the QARM GA models that would change the results of the study. Some of the sensors used in this study can have noisy readings and not all weather events may be accurately reflected in the dataset. An LSTM model also might not be the best neural network based time sequence model; but a transformer model (which may have performed better) was not selected due to data and time constraints.

Future Work

For future iterations of this study, data for more sensors will be incorporated for more months over a longer time span. Additionally, in the future it would be good to modify the genetic algorithm to make combinations of rules outside a logical "AND" operation, to include "OR" situations or possibly ensemble methods with rule voting weights. When the vineyard is more mature, it would be useful to understand exactly how these weather events impact the vines.

Conclusion

Early experiments suggest that the QARM GA method may be a good explainable alternative candidate to conventional neural network methods. In the Jornada dataset, these algorithms were able to provide simple rules with nearly the same F1 score as the much larger neural network models. In the custom collected dataset, while neither method was particularly performant, the QARM GA method produced better prediction results than the LSTM model. In both datasets, the QARM GA model had an additional advantage of providing indicators of land and weather interactions that may be useful to the farmer or grower in understanding land interactions aside from pure predictive performance.

Acknowledgments

The researchers would like to thank Dustin and Jaclyn Mommen of Laurel Grove Wine Farm for being part of the research team and allowing the University of Idaho to work with their vineyard data to explore frost interactions. The author would also like to thank Dr. John Shovic for research mentoring, and the SCARECRO team for data assistance, including Garrett Wells, Lacey Hunt, James Lasso, Walter Neils, Jordan Reed, and Zach Preston, for their work on the project over the last 2 and a half years.

References

Alataş, Bilal, and Erhan Akin. "An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules." Soft Computing 10 (2006): 230-237.

Almasi, Mehrdad, and Mohammad Saniee Abadeh. "Rare-PEARs: A new multi objective evolutionary algorithm to mine rare and non-redundant quantitative association rules." Knowledge-Based Systems 89 (2015): 366-384.

Alvarez, Victoria Pachon, and Jacinto Mata Vazquez. "An evolutionary algorithm to discover quantitative association rules from huge databases without the need for an a priori discretization." Expert Systems with Applications 39.1 (2012): 585-593.

Christou, Ioannis T., et al. "Predictive and explainable machine learning for industrial internet of things applications." 2020 16th international conference on distributed computing in sensor systems (DCOSS). IEEE, 2020.

Everett, Mary L. Exploration of Artificial Intelligence Techniques for Model Usability With Multivariate Time Sequence Data. Diss. University of Idaho, 2023.

Li, Lianwei, et al. "Posdms: A mining system for oceanic dynamics with time series of raster-formatted datasets." Remote Sensing 14.13 (2022): 2991.

Martín, D., et al. "QAR-CIP-NSGA-II: A new multi-objective evolutionary algorithm to mine quantitative association rules." Information Sciences 258 (2014): 1-28.

Martínez-Ballesteros, María, et al. "An evolutionary algorithm to discover quantitative association rules in multidimensional time series." Soft Computing 15 (2011): 2065-2084.

Salleb-Aouissi, Ansaf, et al. "Quantminer for mining quantitative association rules." Journal of Machine Learning Research 14 (2013): 3153-3157.

Sinisterra-Sierra, Santiago, Salvador Godoy-Calderón, and Miriam Pescador-Rojas. "COVID-19 data analysis with a multi-objective evolutionary algorithm for causal association rule mining." Mathematical and Computational Applications 28.1 (2023): 12.

Troncoso-García, A. R., et al. "A new approach based on association rules to add explainability to time series forecasting models." Information Fusion 94 (2023): 169-180.

Yan, Xiaowei, Chengqi Zhang, and Shichao Zhang. "Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support." Expert Systems with

Applications 36.2 (2009): 3066-3076.

Yao, Jin, et al. "Gap-filled daily precipitation at the 15 long-term NPP sites at Jornada Basin LTER, 1980-ongoing." (2023).

Appendix

Equations for Jornada QARM GA Models

The sequence penalty equations for the Jornada QARM GA models are below.

*Index 2: fitness = fitness – (1 – 0.5*amplitude_of_sequence) + 0.8*(fulfillment_early)* (8)

*Index 3: fitness = fitness – (1 – 0.5*amplitude_of_sequence) + 0.8*(fulfillment_late)* (9)

The *amplitude_of_sequence* parameter refers to the length of the sequence divided by the total possible length, averaged across the rule parameters. The *fulfillment_early* parameter is the % backward the earliest bound occurs in the total sequence limit, while the *fulfillment_late* is the % backward of the latest bound.

The range penalty equations for the Jornada QARM GA models are below.

Index 0: fitness = fitness – 1(0.1*amplitude_bound)* (10)

*Index 1: fitness = fitness – 0.2(*0.1*amplitude_bound)* (11)

The *amplitude_bound* parameter is the average bound range out of the total possible range across the rule features.

The fitness function equations for the Jornada QARM GA models are below.

*Index 1: fitness = (2*support)*(num_whole_rule/num_consequent)*confidence* (12)

*Index 2: fitness = 5*support +0.5*confidence* (13)

*Index 3: fitness = 5*support + 0.5*confidence +0.1*lift* (14)

Model Setups for Laurel Grove Wine Farm LSTM Models

Model 1: 128 node LSTM layer, 20% dropout layer, 96 node LSTM layer, 20% dropout layer, 64 node LSTM layer, 20% dropout layer, 64 node Dense layer with relu final activation.

Model 2: 64 node LSTM layer, 20% dropout layer, 48 node LSTM layer, 20% dropout layer, 32 node LSTM layer, 20% dropout layer, 32 node Dense layer with relu final activation.

Model 3: 16 node LSTM layer, 20% dropout layer, 12 node LSTM layer, 20% dropout layer, 8 node LSTM layer, 20% dropout layer, 8 node Dense layer with relu final activation.

Fitness Function Equations for Laurel Grove Wine Farm QARM GA Models

*Index 10: fitness = (2*support)+(num_whole_rule/num_consequent +(3*confidence)*(1-lift)* (15)

Index 12: support+5(num_whole_rule/num_consequent +(5*confidence)+(0.1*lift)* (16)

Index 13: support+3(num_whole_rule/num_consequent +(5*confidence)* (17)

*Index 17: support+(num_whole_rule/num_consequent +(8*confidence)*(0.1*lift)* (18)

Index 18: support+2(num_whole_rule/num_consequent +(10*confidence)*(0.1*lift)* (19)