Treating Stochasticity of Olive-Fruit Fly's Outbreaks via Machine Learning Algorithms[☆]

Romanos Kalamatianos[1], Katia Kermanidis[1], Ioannis Karydis[1,2], Markos Avlonitis[1]

[1] *Department of Informatics, Ionian University, Corfu, Greece 49132*
{c14kala, kerman, karydis, avlon}@ionio.gr
[2] *Creative Web Applications P.C., Corfu, Greece 49131*
jonjon@cwa.gr

Abstract

Olive fruit fly trap measurements are used as one of the indicators for olive grove infestation, and therefore, as a consultation tool on spraying parameters. In this paper, machine learning techniques are used to predict the next olive fruit fly trap measurement, given input knowledge of previous trap measurements as well as an attribute that acts as a correlation model between the temperature and the development of a pest's population, known as the Degree Day model. This is the first time the Degree Day model is utilized as input in classification algorithms for the prediction of olive fruit fly trap measurements. Various classification algorithms are employed and applied to different environmental settings, in extensive comparative experiments, in order to detect the impact of the latter on olive fruit fly population prediction.

Keywords: olive fruit fly, machine learning, population prediction, classification, Naïve Bayes, Nearest Neighbors, Decision trees, Random forests, Support Vector Machines, Neural Networks

1. Introduction

The olive fruit fly is a pest that has been recorded to infest solely the olive fruit since at least the third century BC [2]. Such infestation causes great

[☆] This work is an extension of [1]

damage to the production of both olive oil and table olives [3] in many olive oil producing countries, including Greece. The olive fruit fly is active during the summer and reaches succesive population peaks during autumn, while during the winter and in the first months of spring it hibernates, until environmental conditions are favorable for it to re-emerge [2].

The population growth of the olive fruit fly and, by extension, the level of infestation of an olive grove are affected by various environmental factors. A short non-exhausting list of these factors can be summarized to: temperature, humidity, fruit bearing, olive grove orientation, olive grove variety, spatial diffusion, interaction between neighboring micro-climates [4, 5, 6]. The net result of all these factors is to introduce an spatiotemporal stochasticity in the evolution of olive fruit fly population. While an earlier model proposed by Avlonitis et al. [7] indeed addressed some of the aforementioned complex factors, such as spatial evolution of olive fruit fly, the robust modeling of spatiotemporal evolution can only be achieved by means of the stochastic generalization of the well known logistic equation for the olive fruit fly population, as shown in Equation 1

$$\frac{\partial p}{\partial t} = \beta p(1-p) + c\frac{\partial^2 p}{\partial x^2} + g(p)\delta p \tag{1}$$

where p is the population density, β is the rate of increase, $c\frac{\partial^2 \cdot p}{\partial x^2}$ is the diffusion term in space and $g(p) \cdot \delta p$ models the spatiotemporal stochasticity, $g(p)$ being the corresponding noise amplitude. Within this context, the induced randomness in the time and space of the olive fruit fly outbreak emerges as one of the most crucial product parameters[1].

Population control of the olive fruit fly can be achieved through spraying of the olive trees, either with localized bait or universally [2, 9] at an olive grove. However, in order for the spraying to have effect, it has to be applied when conditions are appropriate. Two factors indicate when spraying should commence [9]: (a) the ripeness level of the olive fruit, as the fruit needs to be

[1] For a recent stochastic model predicting population outbreaks the interested reader is referred to [8]

ripe in order for it to be susceptible to the olive fruit fly and (b) the population of the fly, i.e. when a certain population threshold (recorded via sampling) is exceeded. Sampling is achieved through McPhail traps or yellow sticky traps [2, 9]. The infestation threshold is usually set to seven olive fruit flies per trap per week during the summer while decreased to five olive fruit flies per trap per week during autumn [9]. In both cases each trap covers an area of $77.000m^2$ or approximately 1.000 olive trees.

The aim of this paper is to predict future olive fruit fly trap measurements, and by extension olive fruit fly infestations/outbreaks, using machine learning algorithms.

1.1. Motivation & Contribution

The effect of insects that produce harm to humans' concerns and especially on crops, are of great significance. Pests infect and feed from the fruits and grains of agricultural goods, thus greatly reducing their value. In turn this leads to loss of both alimentary raw material as well as invested funds [10, 11]. Moreover, existing research on olive fruit fly has focused mainly on aspects such as the biology, ecology, management, and impact on olive production. On the other hand, infestation prediction has yet to receive significant attention, despite the wide availability of pests' and contextual data from olive groves as well as the positive effects such a prediction could bring on treating olive fruit fly infestations and thus ameliorating the olive fruit production.

To address these requirements, our previous work [1] presented extensive experimentation with various classification algorithms, on different environmental settings, in order to detect the impact of environmental parameters on olive fruit fly population prediction. Therein, the proposed feature vector consisted of environmental parameters, specifically temperature and information about previous trap measurements. Nevertheless, the feature vector utilized in [1] was somewhat generic: it did not take into account the specifics of the olive fruit fly's bio-cycle, but rather incorporated parameters that affect it. Moreover, while the experimental evaluation was indeed promising, the distribution of the target class in the data presented a high imbalance, leading to poor performance in the prediction of the minority class.

This work significantly extends [1] by

1. Proposing a new feature that closely addresses requirements of the olive fruit fly. We incorporated the Degree Day model as feature, which correlates the environmental temperature with development of an organism.
2. Conducting and presenting promising experimental results with the aforementioned new feature. We experimented on three feature sets, where each feature set varied in the number of attributes, having as a constant attribute the aforementioned feature. Finally, the algorithms utilized therein were applied on four different experimentation methods.
3. Employing various means to address the class imbalance problem. Specifically, we utilized synthetic oversampling to the dataset via the SMOTE (Synthetic Minority Oversampling Technique) technique. Furthermore, we experimented with meta-learners. Finally, the three-fold bin of the trap measurement related attributes was reduced to two by merging two bins that ultimately served the same purpose.

The rest of the paper is organised as follows: Section 2 presents background information and related work, while Section 3 discuses the methodology utilized for the collection of data from environmental sensors and olive fruit fly traps, the features selected and extracted from the raw data as well as the class imbalance problem identified. Next, Section 4 details with the experimental setup and the experimental results obtained. Finally, the paper is concluded in Section 5.

2. Background & Related research

This Section details necessary background information on machine learning methods as well as related existing research on olive fruit fly infestation prediction.

2.1. Machine Learning Algorithms

A number of classification algorithms exist that are suitable for the purposes of experimentation on the theme of this work. As classification approaches to olive fruit fly infestation prediction are extremely limited in number in the literature, the choice of classification algorithms has to a large extent been based on exploratory criteria, with the aim to cover varying learner families, including meta-learning. The following machine learning algorithms were used:

- J48 [12], a decision tree induction algorithm and it is a version of C4.5, an earlier algorithm developed by J. Ross Quinlan [13].
- Sequential Minimal Optimization or SMO [14], an ameliorated algorithm 100 for training support vector machines.
- Naïve Bayes [12] , a probabilistic classifier based on the assumption of conditional independence [15].
- Random Forest [16], a meta-learning classification algorithm that runs iteratively.
- AdaBoost [17], another meta-learning algorithm.
- Ibk [12], an implementation of the k-nearest neighbor algorithm.
- Multilayer Perceptron, an artificial neural network [18].

Very often machine learning algorithms face the problem of over-fitting, i.e. they cope optimally when evaluated on the training data, but poorly on new unseen test data. Usually, the more complicated the model generated by a learning algorithm, the more prone it is to over-fitting. In our experiments, we dealt with the problem of over-fitting by

- applying post-pruning, i.e. sub-tree raising, to the trees generated by the tree-based classifiers (J48 and RandomForest),
- employing a low-complexity polynomial function in the Support Vector Machine kernel, i.e. a first-degree polynomial,

- choosing a low-complexity perceptron structure, that includes only one hidden layer, and investigating its performance with varying numbers of nodes, starting from very low-complexity of only one node, to 15 nodes,
- investigating other neural network structures of low complexity with varying number of inputs and hidden layers, thus of varying complexity.

As far as the problem of outliers is concerned, among all the attributes in our proposed feature vector, only one attribute had numeric values. After analysis of the numeric attribute no instances were found with an unusual/strongly deviating value. Therefore our feature vector space included no significant cases of outliers' instances.

2.2. State-of-the-art Research on Fruit Fly Infestation Prediction

Related research indicates numerous attempts to simulate the population dynamics of the olive fruit fly as well as the prediction of outbreaks. Comins & Fletcher [19] developed a simulation model which predicted the phenology and dynamics of the olive fruit fly using field data. Pommois et al. [20] and Bruno et al. [21] used a cellular automata model to simulate the spatiotemporal infestation of olive groves by the olive fruit fly. Gilioli & Pasquali [22] used an individual-based model to model the development of the olive fruit fly numerically. Avlonitis et al. [7] proposed an evolution equation based on the dispersion of the olive fruit fly to express population outbreaks. In [23], Gutierrez et al. developed a weather-driven physiologically-based demographic model to simulate how the population dynamics and phenology of the olive fruit and olive fruit fly are affected by climate change. Garcia Adeva et al. [24, 25] developed a web-based simulation model to simulate the spatial and temporal development of Bactrocera fruit flies outbreaks, with the use of Finite State Machines. Finally, Voulgaris et al. [26] developed an information system capable of simulating the population dynamics of the olive fruit fly as well as its spatial dispersion in a real field, resulting in predicting population outbreaks. The system proposed in [26] has since been further modified and optimized [27, 28].

In [29], del Sagrado et al. proposed the prediction of olive fruit fly infestation using information about olive tree's health as well as trap measurements. The notion of crop's health, as described therein, includes measurements that address all three stages of the tasks associated with crops: inauguration, monitoring and conclusion of the crop. Their research utilized classification trees and Bayesian networks for the identification of predictors indicating plant health treatment requirements in relation to olive fruit fly infestation. The results obtain therein indicated that the classification tree approach performed favorably to Bayesian networks in terms of simplicity, success rate and sensitivity, though with poor performance on specificity. Moreover, del Sagrado et al. attempted complexity reduction by use of a subset of the available variables to uncertain results. The approach proposed herein differentiates from [29] by proposing a feature vector that is based on both trap measurements as well as environmental factors, such as temperature, instead of the olive tree health.

In somewhat partially related research directions, machine learning techniques have been used to detect oil spills on the surface of the sea by scanning radar images [30], to automatically identify species by sound [31] and to monitor flood protection systems [32]. Machine learning techniques have also been applied in numerous agriculture processes such as the prediction of when a cow should be culled in a dairy herd [33], the estimation of soil moisture [34] and the estimation of a cow's oestrus [35].

Our proposal has the following advantages and disadvantages in regard to the work done in [29].

Advantages:

- The use of the Degree Day (DD) model as a feature attribute. DD (also known as Growing Degree Days) is a correlation model between temperature and the development, and more generally the activity, of a pest population [36]. This model calculates the heat accumulated by an organism during the day, between lower and upper thresholds. By using this model, one can determine, for instance, the amount of heat and by extension the total time required for an organism to transform from one development stage to the other.

- The proposed feature vector could be applied for the prediction of infestation of other pests other than the olive fruit fly. Since, as described above, the DD attribute correlates the development of an organism with the temperature of the environment. Therefore, DD is used in agriculture to predict certain events, such as to predict plant stages [37] or to predict insect development and by extension pest activity [38].

- Computing the DD units is a standard methodology and various methods can be employed [36, 39]. Furthermore, it is a quantifying characteristic and the process of computation can be automated with the use of a computer algorithm. In contrast, in [29] the use of experts is required to assess the crops health based on the sample being collected. In the case that many experts are employed, there is a chance that a difference of opinion may occur.

Disadvantage: Determining the starting date of DD accumulation can be a cumbersome task. In the case of the olive fruit fly, the starting period is when the first eggs are deposed inside the olive fruits. As such, intensive monitoring of the olive grove is required to determine when the first eggs were laid. For instance, in Corfu, Greece under favorable environmental conditions the olive fruit fly begins laying eggs between end of June and start of July. However, there have been cases where due to hot weather oviposition began at late August. Therefore, determining the starting date of accumulation can span from a few days to months.

Concluding, our proposed feature vector incorporates an attribute that closely addresses the biological cycle of the olive fruit fly, which can be easily calculated either manually or automatically. Furthermore, the proposed feature vector can be applied on other pests and insects, in order to predict possible infestation outbreaks.

The main contribution of our proposal in regard to related proposals [30] is the utilization of the Degree Day model, as input in classification algorithms for the prediction of olive fruit fly trap measurements. The use of the aforementioned model with machine learning techniques for this specific problem, to our knowledge, is attempted for the first time.

3. Data Collection

3.1. Environmental Data

The data used in the experiments presented in this work were collected from environmental sensors and olive fruit fly traps that were installed at 16 locations on the north-western side of the island of Corfu, Greece. Readings from the olive fruit fly traps showed the total number of olive fruit flies caught by in the trap. Each reading was conducted, at all traps' locations, every five days for the period from June 10th, 2015 to September 29th, 2015 and from July 8th, 2016 to October 1st, 2016. The sensors at each location logged temperature values at a 15 minutes interval, while a few of these also logged relative humidity values.

3.2. Feature Selection

In order to perform classification experiments, the aforementioned environmental data were transformed into the following set of numeric attributes (in order to represent readings as feature-value learning vectors):

- Mean temperature of the last five days before next trap reading
- Average maximum temperature of the last five days before next trap reading
- Average minimum temperature of the last five days before next trap reading
- Day 1 Mean Temperature
- Day 1 Maximum Temperature

- Day 1 Minimum Temperature
- Day 2 Mean Temperature
- Day 2 Maximum Temperature
- Day 2 Minimum Temperature
- Day 3 Mean Temperature
- Day 3 Maximum Temperature
- Day 3 Minimum Temperature
- Day 4 Mean Temperature
- Day 4 Maximum Temperature
- Day 4 Minimum Temperature
- Day 5 Mean Temperature
- Day 5 Maximum Temperature
- Day 5 Minimum Temperature

Apart from the environmental attributes the trap's measurement of the last reading (number of flies caught) attribute, was also used as input.

Finally, another attribute to be included to the feature vector was the Degree Days (DD). Various methods can be employed to calculate the DD such as the max-min, "saw-tooth" and double sine curve methods, as described by Wilson & Barnett [36]. However, the most common are the single sine curve and mean temperature methods [39]. For the calculation of DD in this work, we employed the following calculation method originally presented in [27], as shown in Equation 2

$$DD = (t - T_L) * (1 - \frac{1}{1 + e^{-10*(t-T_U)}}) \tag{2}$$

where t is the mean day's temperature while T_L and T_U are the lower and upper, respectively, temperature thresholds for the development of the olive fruit fly.

According to the authors of [40, 41], for the olive's fruit fly eggs to hatch 49.77 DD are required while to reach the adult age, the olive fruit fly requires 379.02 DD counting from the day the egg was deposited on the olive fruit. The intuition for the use of this composite attribute is that, having DD into the feature vector, provided an assumption as to the date of the eggs' deposit, prior to the trap measurement. This, as is also claimed in the literature, is expected to support the prediction of an outbreak.

3.3. Feature Vector Extraction

The process of extracting the attributes for the feature vectors from the sensor data was automated by use of a script. The script was written in Python and automatically computed the mean, mean maximum and mean minimum temperature for the five day period before the next trap reading, as well as the mean, maximum and minimum temperatures for each day in the aforementioned five day period. The script exported all vectors in a CSV (Comma Separated Values) file for further processing. Finally trap readings were added manually at each corresponding vector instance.

The temperature-related attributes, initially numeric, were discretized into the following three bins:

- < 15, temperature is lower than 15 ^oC,
- 15 to 32, temperature is between 15 ^oC and 32 ^oC,
- > 32, temperature is greater than 32 ^oC.

The discretization of the temperature values was based on the temperature range (between 15 ^oC and 32 ^oC [42]), in which the olive fruit fly is active. In cases that the temperature of the environment is below the lower or exceeds the upper threshold, then the olive fruit fly is motionless due to cold or heat.

Accordingly, herein we assume that outside the optimal temperature range of the olive fruit fly, the traps will not capture any olive fruit flies.

Trap reading related attributes have also been discretized into the following bins:

- 0 to 4, none or up to 4 olive fruit flies caught in the trap,
- ≥ 5, greater than or equal to five olive fruit flies caught in the trap.

In [1], the quantization of traps' measurements was ternary ("0 to 4", "5 to 6" and "≥ 7") based on theoretical analysis, i.e. the infestation threshold depending on the season the measurements are made. Specifically, in the summer months the infestation threshold was set to seven olive fruit flies per trap per week. On the other hand, from September onwards the infestation threshold was decreased to five olive fruit flies per trap per week, due to cooler weather [9]. Therefore, although the last bin value would always indicate infestation, the second bin value would be depended on the season.

3.4. The Class Imbalance Problem

Field trap measurements analysis indicated low frequency of occurrence for the "5 to 6" bin on accumulated two years' of collected data.

This leads to an overrepresentation of the first and last bins in the data, compared to the second bin. Prediction of instances of the minority class suffers, due to their sparseness.

The problem of class imbalance has been dealt with in previous work in different ways. Random oversampling (random replication of minority examples) and random undersampling (random elimination of majority examples) have been utilized to balance the class distribution [43]. According to several researchers, random oversampling may lead to ovefitting, while random undersampling entails the risk of removing potentially useful data, i.e. data that is crucial for the induction process [44]. Therefore focused resampling methods have been proposed. Focused oversampling techniques vary from replicating only minority examples that appear in the borderline between the two classes [43], to creating new minority class examples by interpolating between existing minority examples [45] and approaches that combine oversampling with data cleaning (removing examples of both classes) [46]. Focused undersampling techniques vary from removing negative examples that participate in Tomek links [47], to finding a consistent subset of examples using the condensed nearest neighbor rule [48], and to the combination of these two approaches called one-sided sampling [49]. Another way to address the class imbalance issue is cost-sensitive learning, i.e. the implementation of classifiers that do not treat all misclassification costs as equal [50, 51]. Finally, metalearning [52, 53] has also been proposed to deal with imbalanced datasets, as it forces the learner to focus on hard examples.

In the present work varying means have been employed to address the imbalance issue:

- Synthetic oversampling is applied to the dataset using the SMOTE technique [45]. Oversampling is chosen to undersampling due to the limited number of available examples. SMOTE (Synthetic Minority Oversampling TEchnique) creates new synthetic examples from a minority class sample, by considering its nearest neighbors and linearly combining the sample with each neighbor. SMOTE has been used extensively for detecting network intrusion [54], for detecting sentence boundaries in speech [55], for species distribution prediction [56], for detecting breast cancer [57], as well as in bioinformatics applications [58, 59, 60, 61, 62].

- Metalearning is applied to 'force' the learner to mind the erroneously classified instances. Boosting as well as the RandomForest metalearning schemata have been experimented with.

- The initial three-fold threshold of the trap measurements is relaxed, given the aforementioned dual threshold on the infestation based on the ambient temperature, both bins "5 to 6" and "$\geq$ 7" served qualitatively the same purpose. Thus, bins “5 to 6” and “$\geq$ 7” were aggregated to the “$\geq$ 5” super-class bin, while infestation indication was also aggregated to the same bin.

4. Performance Evaluation

In support of the efficiency of the proposed feature vector and the examined machine learning algorithms, this section presents a number of experiments that have been performed. A concise description of the experimentation platform and data sets is also given followed by a performance analysis.

4.1. Experimental Setup

Since the proposed machine learning approach, for the specific problem described herein, is done for the first time, the selection of the architecture of the proposed models was purely exploratory. Therefore, an optimal architecture cannot be known beforehand. The proposed models were selected based on their ability to satisfactorily tackle the over-fitting problem, the type of feature attributes and the small number of training instances that we had in our disposal.

201 training instances were supplied. Due to the small size of the training data, no test set could be supplied for the validation of the results. Therefore the 10-fold cross-validation method was used. The original sample was randomly partitioned into ten sub-samples. One out of ten sub-samples is kept as validation data for testing the model, and the remaining nine sub-samples are used as training data. The cross-validation process was then repeated ten times, with each of the ten sub-samples being used only once as validation data while results were averaged across the ten experiments. The feature vector was augmented to include another attribute, denoting the next trap reading, that was used as the classification class.

The WEKA machine learning workbench[2] was used for running the classifi-

[2] `http://www.cs.waikato.ac.nz/ml/weka/`

cation experiments. In the sequel, Tables 1,2,3,4,5,6 and 7 present the parameter values selected for each of the classification algorithms selected for experimentation. In the k-nearest neighbor algorithm, the number of nearest neighbors ranged from 1 to 33 neighbors, while only odd values were selected, to ensure that ties are avoided, and majority voting is feasible. In the Multilayer Perceptron, experiments were conducted for one hidden layer with the number of nodes ranging from 1 to 10.

In some cases, the SMOTE technique [45] was utilized in order to resample the dataset whenever the training set included a minority class. Accordingly, four methods are examined in the sequel:

Method 1 The results of our previous work [1].

Method 2 The feature vector of our previous work [1] was utilized with the SMOTE technique on the data of 2015 for training while the original data of 2016 were used as a test set.

Method 3 The proposed feature vector was used, with the SMOTE technique utilized on the data of 2015 for training while the data of 2016 were used as evaluation set.

Method 4 The feature vector utilized in Method 3, with the aggregation of the infestation bins from tertiary in our previous work to binary herein, from both the data from 2015 and 2016

For the classification problem using Neural Networks, we utilized a two-layered feed-forward network with sigmoid hidden and output neurons with varying neurons at the hidden layer in order to test the effect of the neuron and size. The experimentation also included the division of the dataset into training, validation of generality, and testing subsets in different sizes. In all experiments with the NN presented herein evaluation of the performance was only based on the testing subset. The learning function used was the scaled conjugate gradient back-propagation function while the performance function was

Binary Splits	No
Confidence Factor	0.25
Minimum Instances per Leaf	2
Reduced Error Pruning	No
Subtree raising	Yes
Pruned	Yes
Laplace smoothing	No

Table 1: J48 parameter values.

Complexity parameter	1.0
Round-off error	1.0E-12
Filter Type	Normalize training data
Kernel	PolyKernel
Random seed for cross validation	1
Tolerance parameter	0.001

Table 2: SMO parameter values.

Use kernel estimator	No
Use supervised discretization	No

Table 3: Naïve Bayes parameter values.

Maximum Depth	Unlimited
Number of Attributes	0
Number of trees to be generated	100
Seed	1

Table 4: RandomForest parameter values.

the Mean Squared Error (MSE), between the outputs and targets, performance function as well as the absolute percentage of erroneous classifications. Each testing of the network was repeated 10 times and both MSE and the percentage of erroneous classifications were averaged in order to generalize results. More-

Classifier	SMO
Number of Iterations	10
Seed	1
Use resampling	No
Weight Threshold	100

Table 5: AdaBoost parameter values.

Cross Validate	No
Distance Weighting	No
Use of Mean Squared Error	No
Nearest Neighbor Search Algorithm	LinearNNSearch
Window Size	0

Table 6: IBk parameter values.

Decrease learning rate	No
Hidden layers	1
Learning rate	0.3
Momentum	0.2
Nominal to binary filter	Yes
Normalize attributes	Yes
Normalize numeric class	Yes
Reset	Yes
Seed	0
Training time	500
Validation set size	0
Validation threshold	20

Table 7: Multilayer Perceptron parameter values.

over, in order to present a composite metric of efficiency including both MSE and the percentage of erroneous classifications, the equally weighted product of both these metrics was also utilized. In this experimentation set, the following

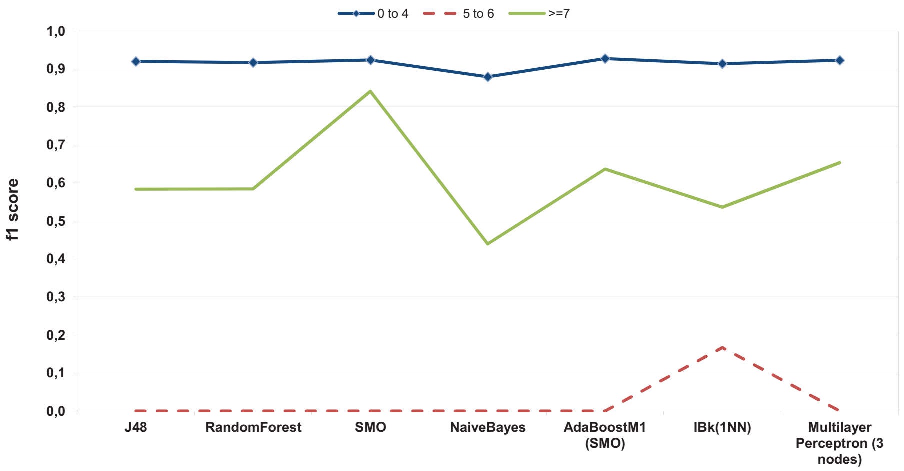

Figure 1: f1 score for all classification algorithms for each classification class.

3 feature-sets were examined:

2 features Solely the values of DD as well as the previous trap measurements.

20 features All discretized features, DD and the previous trap measurements.

19 features All discretized features and the previous trap measurements (i.e. similar to "20 features" but without the DD).

4.2. Experimental Results

Figure 1 displays the classification results of all the aforementioned machine learning algorithms (Method 1) for each classification class. In this case, the feature vector consisted of 20 attributes, 18 of which were temperature related while the last two were trap related. The class "5 to 6" included too little data to receive useful results. The results of J48, SMO, AdaBoostM1, IBk and multilayer perceptron are comparable for the class "0 to 4", while for the class "$\geq$ 7" SMO produced the best results. On the other hand, Naïve Bayes produces the worst results in comparison with the other algorithms, with a significant decrease in all classes.

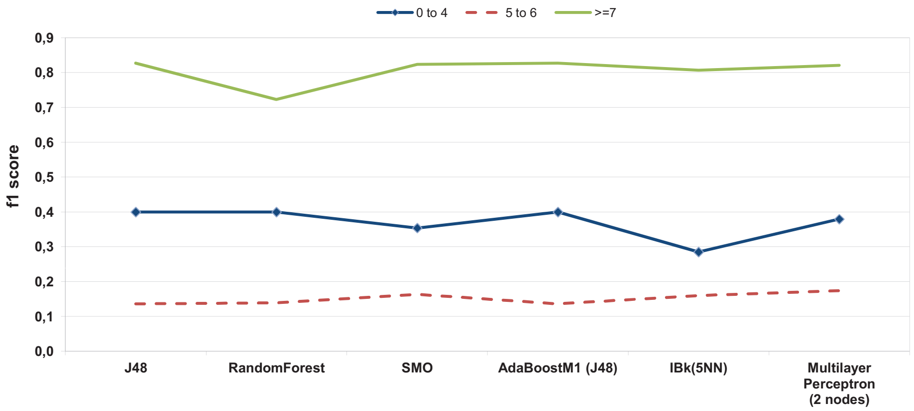

Figure 2: f1 score for all classification algorithms for each classification class (SMOTE on train data, train:2015, test:2016).

In the next set of experiments, the feature vector of our previous work [1] was utilized with the SMOTE technique on the data of 2015 for training while the original data of 2016 were used as a test set (Method 2). Our aim in this experiment was to examine if balancing the training set of 2015 and evaluating them against the new data collected from 2016 would produce better results. Figure 2 presents the f1 score that all classification algorithms achieved for each classification class. J48, SMO, AdaBoostM1, IBk and multilayer perceptron produce comparable performance for the class "≥ 7" while RandomForest indicates a significantly lower performance. In class "0 to 4", J48, RandomForest and AdaBoostM1 produce comparable performance with SMO little lower and IBk and multilayer perceptron significantly lower performance. "5 to 6" class' results are distinctively lower than the other two classes with SMO, IBk and the multilayer perceptron as the top results. Similarly, Figure 3 presents the precision, recall and f1 score that the IBk algorithm achieved. The best f1 score was achieved for 5 nearest neighbors. Finally, 4 presents the precision, recall and f1 score that the multilayer perceptron, with 1 hidden layer, algorithm achieved. The best f1 score was achieved for 2 nodes.

Following the paradigm of the previous experimentation set, in the next experimentation set the proposed feature vector was used, while the SMOTE

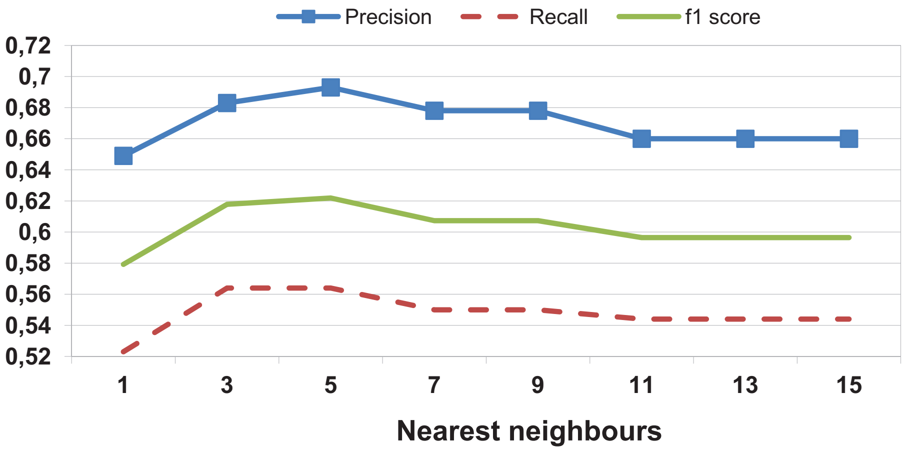

Figure 3: Precision, Recall and f1 score for Ibk classification algorithm (SMOTE on train data, train:2015, test:2016).

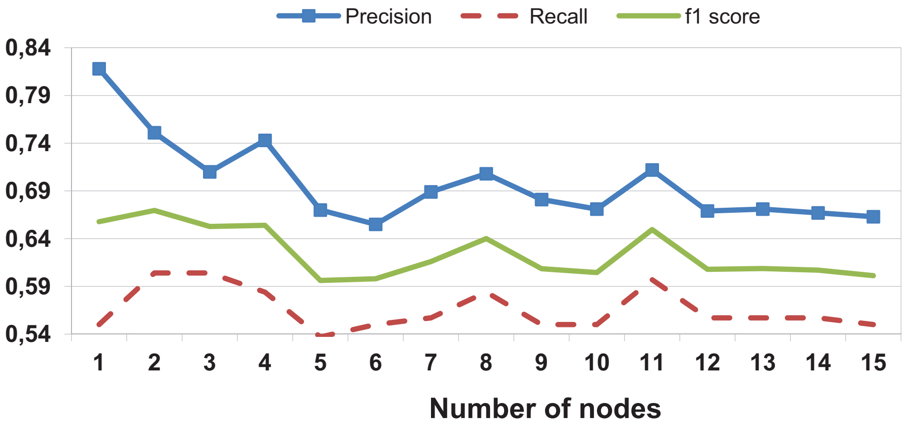

Figure 4: Precision, Recall and f1 score for multilayer perceptron classification algorithm (SMOTE on train data, train:2015, test:2016).

technique was applied on both the data from 2015 and 2016 in order to balance the classification class values (Method 3). The feature vector of this experiment set was reduced to three attributes, namely DD, Previous Reading and Reading, since DD is derived from temperature, we decided to remove all temperature related attributes. Thus, in this experiment, we examined if applying the same procedure as in Method 2 on the new feature vector would yield better results.

Figure 5 presents the f1 score the J48, RandomForest, SMO, Naïve Bayes, AdaBoostM1(SMO), Ibk and multilayer perceptron algorithms achieved. For class "$\geq$ 7" SMO and the multilayer perceptron present the best performance with J48, AdaBoostM1(SMO) and IBk following suite at lower but comparable performances while RandomForest and Naïve Bayes presenting the lowest. In class "0 to 4" the best performance is achieved by AdaBoostM1(SMO), with J48, SMO, Naïve Bayes, IBk and multilayer perceptron following, while RandomForest showing the worst performance. Again, class "5 to 6" included too little data to receive useful results, despite the slightly increased performance of the Naïve Bayes algorithm.

Figure 6 presents the precision, recall and f1 score that the Ibk algorithm achieved. The best f1 score was achieved for 3 nearest neighbors. Finally, 7 presents the precision, recall and f1 score that the multilayer perceptron, with 1 hidden layer, algorithm achieved. The best f1 score was achieved for 1 node.

The next experimentation set utilized the proposed feature vector, with the aggregation of the infestation bins from tertiary in our previous work to binary herein, from both the data from 2015 and 2016 (Method 4). This experiment is another attempt to balance the classification class by merging two class values into one. Figure 8 presents the f1 score that the J48, RandomForest, SMO, Naïve Bayes, IBk and multilayer perceptron algorithms achieved for each classification class. Although not significantly better, the IBk performed better than the rest at the f1 score with multilayer perceptron being very close in both "0 to 4" and "$\geq$ 5" classes. Similarly, J48, SMO and Naïve Bayes presented slightly lower but closely matching performances, while RandomForest achieved the worst, in both classes. Figure 9 presents the precision, recall and f1 score that the Ibk algorithm

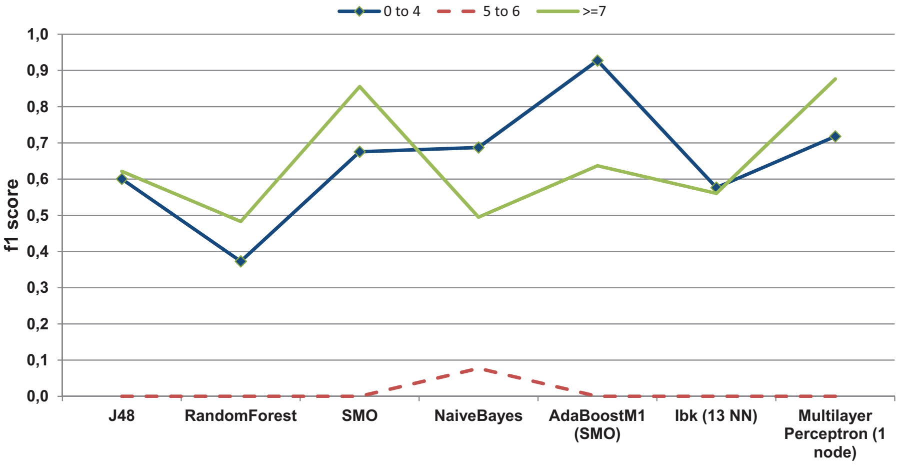

Figure 5: f1 score for J48, RandomForest, SMO, Naïve Bayes, AdaBoostM1, IBk and multi-layer perceptron classification algorithms for each classification class (SMOTE on train & test data).

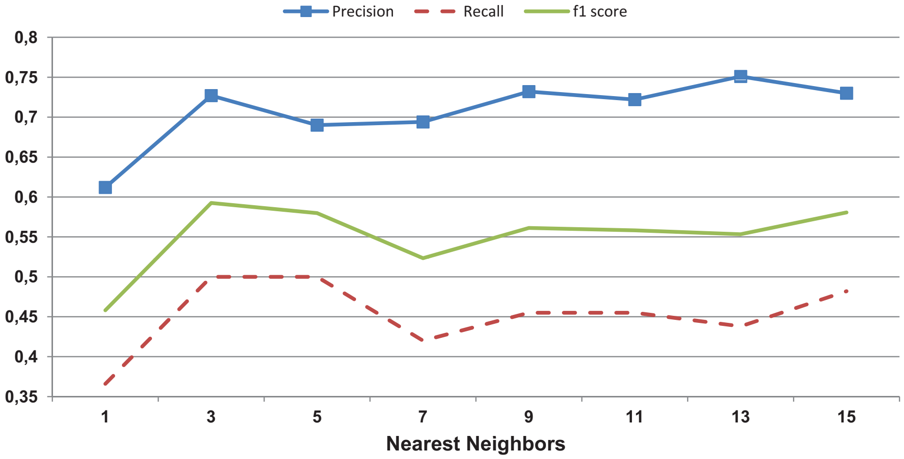

Figure 6: Precision, Recall and f1 score for Ibk classification algorithm (SMOTE on train & test data).

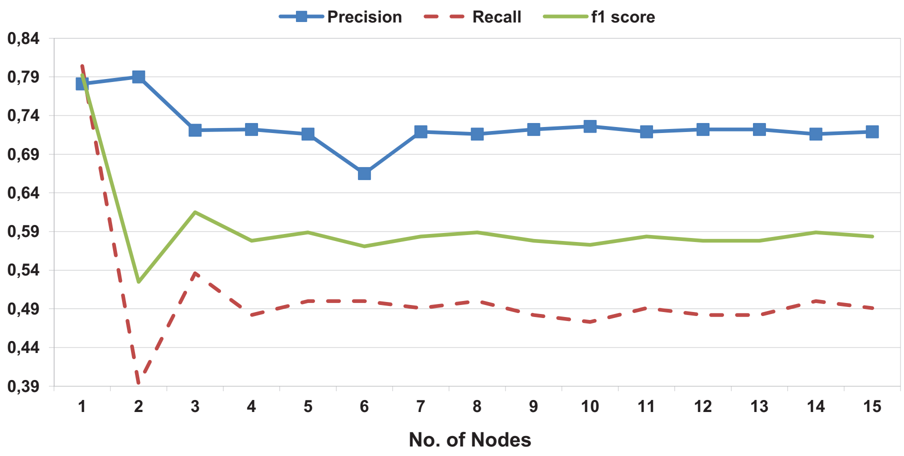

Figure 7: Precision, Recall and f1 score for multilayer perceptron classification algorithm (SMOTE on train & test data).

achieved. The best f1 score was achieved for 13 nearest neighbors. Finally, 7 presents the precision, recall and f1 score that the multilayer perceptron, with 1 hidden layer, algorithm achieved. The best f1 score was achieved for 10 nodes.

The next experiment focuses at the comparison the feature vector of our previous work [1] with the one proposed herein provide for each Method when the best performing algorithm is selected. Figure 11 depicts the collective results. The higher performance of Method 1 is misleading, as this Method dealt with a highly imbalanced dataset, and the reported metrics constitute the average values over all class labels. In truth, performance on the minority class label alone (which is basically the class of interest) is much lower. Methods 2, 3 and 4 address a balanced dataset, and though the average performance values over all class labels are lower, performance on the class of interest is significantly higher.

The final experimentation set focuses on the use of Neural Networks for the classification of the data collected (including environmental, previous class as well as composite, such as DD) for the prediction of the class of olive fruit flies within traps. The intuition of this experimentation set is twofold: initially (1) to identify the capability of the utilized Neural Network to accurately predict

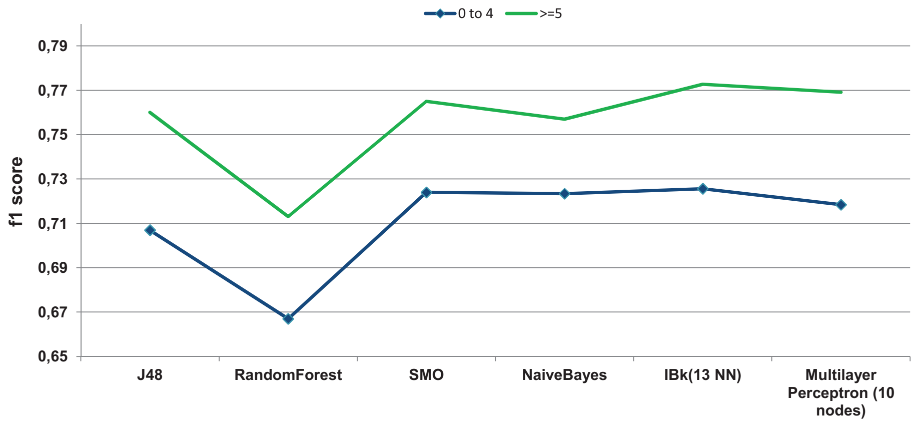

Figure 8: f1 score for J48, RandomForest, SMO, Naïve Bayes, IBk and multilayer perceptron classification algorithms for each classification class (aggregated infestation bins).

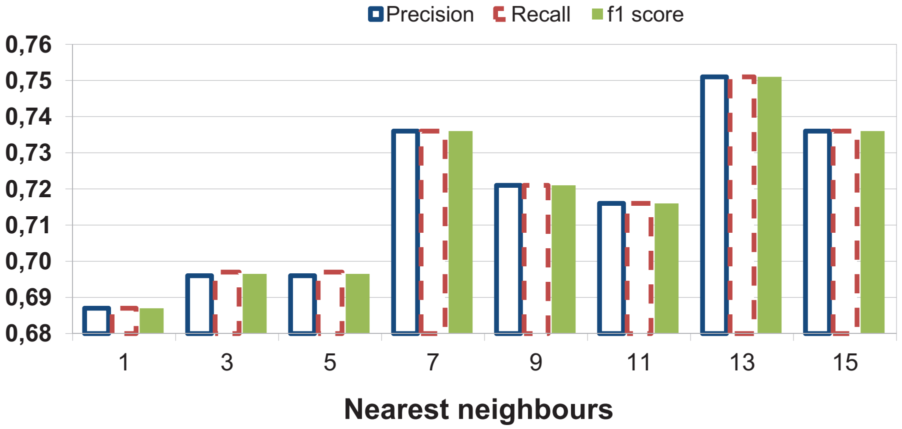

Figure 9: Precision, Recall and f1 score for Ibk classification algorithm (aggregated infestation bins).

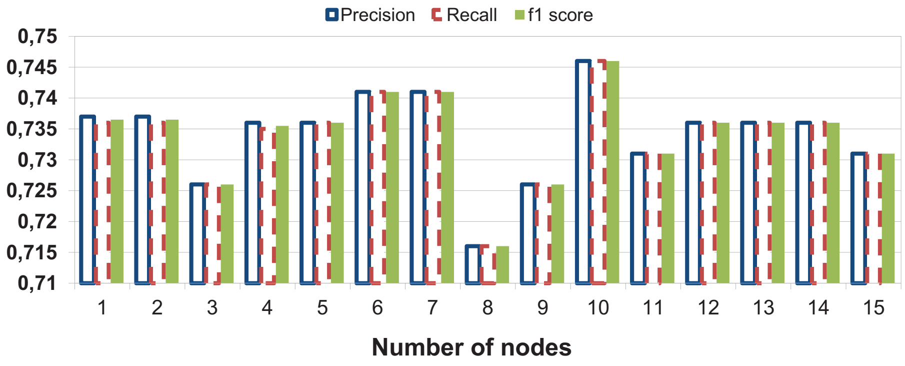

Figure 10: Precision, Recall and f1 score for multilayer perceptron classification algorithm (aggregated infestation bins).

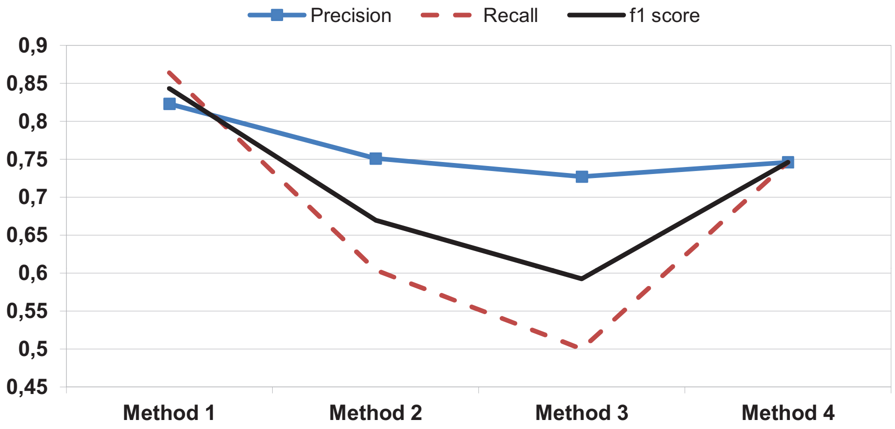

Figure 11: Precision, Recall and f1 score comparison between best results of each Method.

the number of collected pests given the available input data in comparison to the other Machine Learning techniques utilized herein, as well as (2) to verify the contribution of various feature vector sets to the discriminating capability of the NN classification.

In order to present a cumulative result of the NN technique for all feature-sets, Figure 12 presents solely the composite metric of the equally weighted product of both MSE and the percentage of erroneous classifications. The x-axis is not continuous and designates the parameters used for each value of the methods' experimentation. These parameteres are only shown in detail in Table 8 for the best result of each feature-set.

Feature-set	hidden layers	Ratios	Composite metric	% of erroneous classifications	MSE
2 features	1	50-40-10	0.12198	24.98%	0.16258
20 features	5	40-50-10	0.12463	23.48%	0.16288
19 features	1	10-80-10	0.12859	28.48%	0.17981

Table 8: Detailed parameters for the best composite metric result for all feature-sets.

The best result is achieved by the "2 feature"-set with "20 feature"-set and "19 feature"-set following. Despite the very small variation of the composite metric, the information of Table 8 indicates the clear superiority of the inclusion of DD within the feature set as lack increases significantly both elements of the composite metric, while when available, both MSE and the percentage of erroneous classifications are highly comparable.

Moreover, the percentage of correct classifications, that represents the precision of the NN methodology, is highly comparable to the precision achieved by the other Machine Learning techniques utilized herein, as shown in Figure 11.

5. Conclusions

In this work, supervised machine learning is used to predict future olive fruit fly population outbreaks. The proposed feature vector consists of environmental parameters, specifically temperature, information about previous population

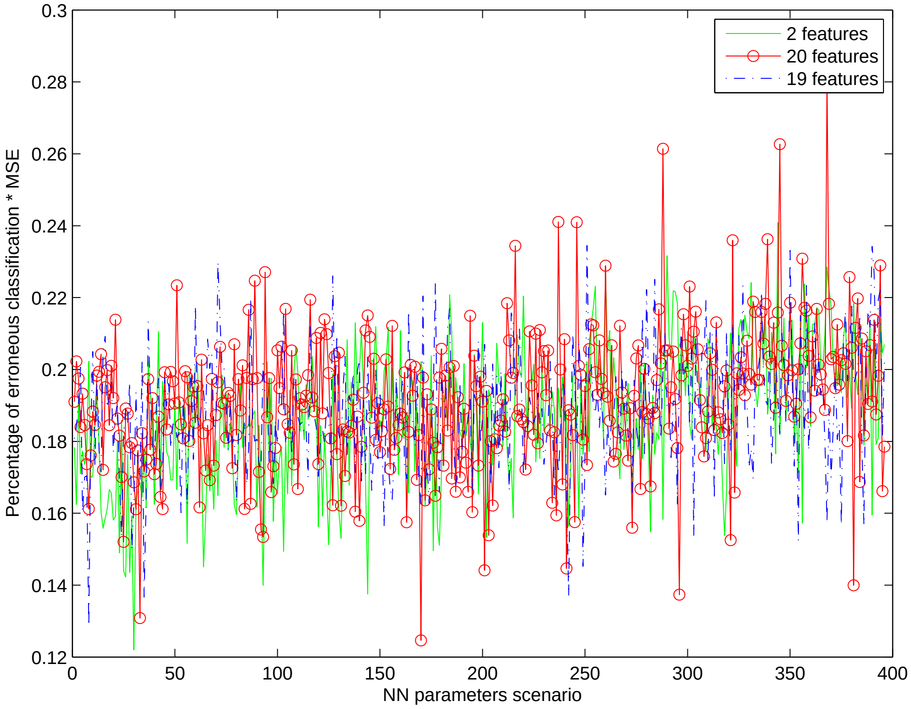

Figure 12: Product of percentage of erroneous classification and MSE for all feature-sets.

data and more importantly the development stage according to the corresponding DD model. Results produced by the conducted experiments are promising while indicating the superiority of the performance given the proposed additional attribute of the development stage to the feature vector.

This can be understood in the basis of Equation 2: it can be seen that the presence of the temperature thresholds values, T_L, T_U may result to a dramatic change on the development stages of the olive fruit fly population even if only small differences of environmental temperatures are present. As a result, a machine learning algorithm based only on the absolute values of the environmental temperatures will not be able to correctly predict olive fruit fly outbreaks.

On the other hand, future research should take into account more environmental parameters such as relative humidity, the amount of light the olive fruit flies are exposed to and diffusion characteristics. This can be understood in the basis of Equation 1 where the presence of the diffusion term is crucial for the robust modeling of olive fruit fly population in time and space. Indeed, humidity, luminance and fruit bearing percentage may drastically affect diffusion of the olive fruit fly population within the olive grove.

Finally, the experiments described are planned to be conducted again on more training instances as measurement data accumulate. Providing more training instances to the machine learning algorithms should produce better results.

Acknowledgment

Financial support from the European Union and National Funds of Greece and Albania under the IPA Cross-Border Programme “Greece - Albania 2007-2013”, of the project titled “Enhancing Olive Oil Production with the use of Innovative ICT” with the acronym “e-Olive”, is gratefully acknowledged.

References

[1] R. Kalamatianos, K. Kermanidis, M. Avlonitis, I. Karydis, Environmental impact on predicting olive fruit fly population using trap measurements,

in: IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, 2016, pp. 180–190.

[2] P. Vossen, L. G. Varel, D. Alexandra, Olive fruit fly, Tech. rep., University of California Cooperative Extension - Sonoma County, `http://cenapa.ucanr.edu/files/52578.pdf` (2004).

[3] R. Rice, Bionomics of the olive fruit fly bactrocera (dacus) oleae, University of California Plant Protection Quarterly 10 (2000) 1–5.

[4] B. Fletcher, Temperature development rate rela-tionships of the immature stages and adults of tephritid fruit flies, in: H. G. Robinson, A. S. (Ed.), Fruit Flies: Their Biology, Natural Enemies and Control, Vol. 3A, Elsevier, 1989, p. 273289.

[5] V. Y. Yokoyama, P. Rendon, J. Sivinski, Biological control of olive fruit fly (diptera: Tephritidae) by reseases of psyttalia cf. concolor (hymenoptera: Braconidae) in california, parasitoid longevity in presence of the host, and host status of walnut husk fly, in: International Symposium on Fruit Flies of Economic Importance, 2006, pp. 157–164.

[6] G. D. Broufas, M. L. Pappas, D. S. Koveos, Effect of relative humidity on longevity, ovarian maturation, and egg production in the olive fruit fly (diptera: Tephritidae), Annals of the Entomological Society of America 102 (1) (2009) 70–75. `doi:10.1603/008.102.0107`.

[7] M. Avlonitis, D. Tragoudaras, M. Stefanidakis, Stochastic processes and insect outbreak systems: Application to olive fruit fly, in: Proceedings of the 3rd IASME/WSEAS International Conference on energy, environment, ecosystems and sustainable development, 2007, pp. 98–103.

[8] M. Avlonitis, On the problem of early detection of users interaction outbreaks via stochastic differential models, Engineering Applications of Artificial Intelligence 51 (2016) 92 – 96, mining the Humanities: Technologies

and Applications. doi:http://dx.doi.org/10.1016/j.engappai.2016.01.008.

[9] A. Patsias, Fighting the olive fruit fly (from greek "*η καταλέμηση του δάκου της ελιάς*"), Tech. rep., Publicity Department of Agricultural Sector Applications and Publicity, Nicosia, Cyprus, http://www.moa.gov.cy/moa/da/da.nsf/All/59FDB03C747214A6C2257A22003F48D2/$file/KatapolemisiDakouElias.pdf?OpenElement (2005).

[10] D. A. Murray, M. B. Clarke, D. A. Ronning, Estimating invertebrate pest losses in six major australian grain crops, Australian Journal of Entomology 52 (3) (2013) 227–241.

[11] G. E. Haniotakis, Olive pest control: present status and prospects, IOBC wprs Bulletin 28 (9) (2005) 1.

[12] I. H. Witten, E. Frank, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2005.

[13] J. R. Quinlan, Bagging, boosting, and c4.5, in: AAAI/IAAI, Vol. 1, 1996, pp. 725–730.

[14] J. Platt, Sequential minimal optimization: A fast algorithm for training support vector machines, Tech. rep., Microsoft Research, technical report msr-tr-98-14 (1998).

[15] S. Russell, P. Norvig, Artificial intelligence: a modern approach, Prentice Hall Upper Saddle River, 2009.

[16] L. Breiman, Random forests, Machine learning 45 (1) (2001) 5–32.

[17] Y. Freund, R. E. Schapire, Experiments with a new boosting algorithm, in: International Conference on Machine Learning, Vol. 96, 1996, pp. 148–156.

[18] F. Rosenblatt, Principles of neurodynamics: Perceptrons and the theory of brain mechanisms, Spartan Books, 1961.

[19] H. N. Comins, B. S. Fletcher, Simulation of fruit fly population dynamics, with particular reference to the olive fruit fly, dacus oleae, Ecological modelling 40 (3) (1988) 213–231.

[20] P. Pommois, P. Brunetti, V. Bruno, A. Mazzei, V. Baldacchini, S. Di Gregorio, Flysim: a cellular automata model of bactrocera oleae (olive fruit fly) infestation and first simulations, in: International Conference on Cellular Automata, Springer, 2006, pp. 311–320.

[21] V. Bruno, V. Baldacchini, S. Di Gregorio, Temperature effects on olive fruit fly infestation in the flysim cellular automata model, in: Natural Computing, Springer, 2010, pp. 125–132.

[22] G. Gilioli, S. Pasquali, Use of individual-based models for population parameters estimation, ecological modelling 200 (1) (2007) 109–118.

[23] A. P. Gutierrez, L. Ponti, Q. Cossu, Effects of climate warming on olive and olive fly (bactrocera oleae (gmelin)) in california and italy, Climatic Change 95 (1-2) (2009) 195–217.

[24] J. G. Adeva, J. Botha, M. Reynolds, A simulation modelling approach to forecast establishment and spread of bactrocera fruit flies, Ecological Modelling 227 (2012) 93–108.

[25] J. G. Adeva, M. Reynolds, Web-based simulation of fruit fly to support biosecurity decision-making, Ecological informatics 9 (2012) 19–36.

[26] S. Voulgaris, M. Stefanidakis, A. Floros, M. Avlonitis, Stochastic modeling and simulation of olive fruit fly outbreaks, Procedia Technology 8 (2013) 580–586.

[27] R. Kalamatianos, M. Avlonitis, S. Stravoravdis, Complex networks and simulation strategies: An application to olive fruit fly dispersion, in: Information, Intelligence, Systems and Applications (IISA), 2015 6th International Conference on, IEEE, 2015, pp. 1–6.

[28] R. Kalamatianos, M. Avlonitis, The role of tree distribution and olive fruit bearing in olive fruit fly infestation, in: Proceedings of 7th International Conference on Information and Communication Technologies in Agriculture, Food and Environment, 2015.

[29] J. del Sagrado, I. M. del Águila, Olive fly infestation prediction using machine learning techniques, in: Conference of the Spanish Association for Artificial Intelligence, Springer, 2007, pp. 229–238.

[30] M. Kubat, R. C. Holte, S. Matwin, Machine learning for the detection of oil spills in satellite radar images, Machine Learning 30 (2) (1998) 195–215. doi:10.1023/A:1007452223027.

[31] M. A. Acevedo, C. J. Corrada-Bravo, H. Corrada-Bravo, L. J. Villanueva-Rivera, T. M. Aide, Automated classification of bird and amphibian calls using machine learning: A comparison of methods, Ecological Informatics 4 (4) (2009) 206–214, http://dx.doi.org/10.1016/j.ecoinf.2009.06.005.

[32] A. L. Pyayt, I. I. Mokhov, B. Lang, V. V. Krzhizhanovskaya, R. J. Meijer, Machine learning methods for environmental monitoring and flood protection, World Academy of Science, Engineering and Technology 5 (2011) 82–85.

[33] R. J. McQueen, S. R. Garner, C. G. Nevill-Manning, I. H. Witten, Applying machine learning to agricultural data, Computers and electronics in agriculture 12 (4) (1995) 275–293.

[34] S. Ahmad, A. Kalra, H. Stephen, Estimating soil moisture using remote sensing data: A machine learning approach, Advances in Water Resources 33 (1) (2010) 69–80.

[35] R. S. Mitchell, R. A. Sherlock, L. A. Smith, An investigation into the use of machine learning for determining oestrus in cows, Computers and Electronics in Agriculture 15 (3) (1996) 195–213.

[36] L. Wilson, W. Barnett, Degree-days: an aid in crop and pest management, California Agriculture 37 (1) (1983) 4–7.

[37] P. Miller, W. Lanier, S. Brandt, Using growing degree days to predict plant stages, Ag/Extension Communications Coordinator, Communications Services, Montana State University-Bozeman, Bozeman, MO.

[38] D. A. Herms, Using degree-days and plant phenology to predict pest activity, IPM (integrated pest management) of midwest landscapes (2004) 49–59.

[39] P. W. Brown, Heat units, Bull 8915.

[40] A. Crovetti, F. Quaglia, G. Loi, E. Rossi, P. Malfatti, F. Chesi, B. Conti, A. Belcari, A. Raspi, B. Paparatti, Influenza di temperatura e umidità sullo sviluppo degli stadi preimaginali di dacus oleae (gmelin), Frust Entom 18 (1982) 133–166.

[41] M. F. Gonçalves, L. M. Torres, The use of the cumulative degree-days to predict olive fly, bactrocera oleae (rossi), activity in traditional olive groves from the northeast of portugal, Journal of Pest Science 84 (2) (2011) 187–197.

[42] P. Vossen, Monitoring and control of olive fruit fly (olf) for oil production in california, Tech. rep., University of California Cooperative Extension, `http://cesonoma.ucanr.edu/files/203835.pdf` (2014).

[43] N. Japkowicz, The class imbalance problem: Significance and strategies, in: Proc. of the Intl Conf. on Artificial Intelligence, Citeseer, 2000.

[44] G. Batista, R. C. Prati, M. C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM Sigkdd Explorations Newsletter 6 (1) (2004) 20–29.

[45] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of artificial intelligence research 16 (2002) 321–357.

[46] G. Batista, A. L. Bazzan, M. C. Monard, Balancing training data for automated annotation of keywords: a case study., in: Proceedings of the Second Brazilian Workshop on Bioinformatics, 2003, pp. 20–28.

[47] I. Tomek, Two modifications of cnn, IEEE Transactions on Systems, Man and Communications 6 (1976) 769–772.

[48] P. Hart, The condensed nearest neighbor rule (corresp.), IEEE Transactions on Information Theory 14 (3) (1968) 515–516.

[49] M. Kubat, S. Matwin, et al., Addressing the curse of imbalanced training sets: one-sided selection, in: ICML, Vol. 97, Nashville, USA, 1997, pp. 179–186.

[50] P. Domingos, Metacost: A general method for making classifiers cost-sensitive, in: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 1999, pp. 155–164.

[51] F. Provost, T. Fawcett, Robust classification for imprecise environments, Machine learning 42 (3) (2001) 203–231.

[52] R. E. Schapire, The boosting approach to machine learning: An overview, in: Nonlinear estimation and classification, Springer, 2003, pp. 149–171.

[53] H. Guo, H. L. Viktor, Learning from imbalanced data sets with boosting and data generation: the databoost-im approach, ACM SIGKDD Explorations Newsletter 6 (1) (2004) 30–39.

[54] D. A. Cieslak, N. V. Chawla, A. Striegel, Combating imbalance in network intrusion datasets., in: GrC, 2006, pp. 732–737.

[55] Y. Liu, N. V. Chawla, M. P. Harper, E. Shriberg, A. Stolcke, A study in machine learning from imbalanced data for sentence boundary detection in speech, Computer Speech & Language 20 (4) (2006) 468–494.

[56] R. A. Johnson, N. V. Chawla, J. J. Hellmann, Species distribution modeling and prediction: A class imbalance problem, in: Intelligent Data Understanding (CIDU), 2012 Conference on, IEEE, 2012, pp. 9–16.

[57] A. Fallahi, S. Jafari, An expert system for detection of breast cancer using data preprocessing and bayesian network, Int J Adv Sci Technol 34 (2011) 65–70.

[58] R. Batuwita, V. Palade, micropred: effective classification of pre-mirnas for human mirna gene prediction, Bioinformatics 25 (8) (2009) 989–995.

[59] J. Xiao, X. Tang, Y. Li, Z. Fang, D. Ma, Y. He, M. Li, Identification of microrna precursors based on random forest with network-level representation method of stem-loop structure, BMC bioinformatics 12 (1) (2011) 1.

[60] K. D. MacIsaac, D. B. Gordon, L. Nekludova, D. T. Odom, J. Schreiber, D. K. Gifford, R. A. Young, E. Fraenkel, A hypothesis-based approach for identifying the binding specificity of regulatory proteins from chromatin immunoprecipitation data, Bioinformatics 22 (4) (2006) 423–429.

[61] J. Wang, M. Xu, H. Wang, J. Zhang, Classification of imbalanced data by using the smote algorithm and locally linear embedding, in: 2006 8th international Conference on Signal Processing, Vol. 3, IEEE, 2006.

[62] S. Doyle, J. Monaco, M. Feldman, J. Tomaszewski, A. Madabhushi, An active learning based classification strategy for the minority class problem: application to histopathology annotation, BMC bioinformatics 12 (1) (2011) 424.