Computers and Electronics in Agriculture 104 (2014) 93–104

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Machine learning assessments of soil drying for agricultural planning

Evan J. Coopersmith [a,*], Barbara S. Minsker [a], Craig E. Wenzel [b], Brian J. Gilmore [b]

[a] Department of Civil and Environmental Engineering, University of Illinois, Urbana-Champaign, Urbana, IL 61801, United States
[b] John Deere Technological Innovation Center, Champaign, IL 61820, United States

ARTICLE INFO

Article history:
Received 22 March 2013
Received in revised form 25 February 2014
Accepted 9 April 2014

Keywords:
Soil drying
Field readiness
Machine learning
Nearest neighbors
Soil moisture
Decision support

ABSTRACT

The hydrologic processes of wetting and drying play a crucial role in agricultural activities involving heavy equipment on unpaved terrain. When soil conditions moisten, equipment can become mired, causing expensive delays. While experienced users may assess soil conditions before entering off-road areas, novice users or those who must remotely assess sites before traveling may have difficulty assessing conditions reliably. One means of assessing dryness is remotely-monitored *in situ* sensors. Unfortunately, land owners hesitate to place sensors due to monetary costs, complexity, and sometimes infeasibility of physical visits to remote locations. This work addresses these limitations by modeling the wetting/drying process through machine learning algorithms fed by hydrologic data – remotely assessing soil conditions using only publicly-accessible information. Classification trees, k-nearest-neighbors, and boosted perceptrons deliver statistical soil dryness estimates at a site located in Urbana, IL. The k-nearest-neighbor and boosted perceptron algorithms both performed with 91–94% accuracy, with most misclassifications falling within calculated margins of error. These analyses demonstrate that reasonably accurate predictions of current soil conditions are possible with only precipitation and potential evaporation data. These two values are measured throughout the continental United States and are likely to be available globally from satellite sensors in the near future. Through this type of approach, agricultural management decisions can be enabled remotely, without the time and expense of on-site visitations or extensive ground-based sensory grids.

1. Introduction

Previous work has made forays into soil drying assessments over a diverse set of geographic locations, climate conditions, and functional objectives. The primary dynamic process affecting soil drying is precipitation (Entekhabi and Rodriguez-Iturbe, 1994). For this reason, models of wetting and drying have often focused upon an "antecedent precipitation index" (API), using a pre-set window of previous rainfall to estimate current levels of soil moisture (Saxton and Lenz, 1967). This particular concept of calculating an API has been applied in a variety of contexts: in conjunction with microwave sensing for soil moisture estimation (Blanchard et al., 1981), soil water recession modeling for agriculture (Choudhury and Blanchard, 1983), and for weather prediction (Wetzel and Chang, 1988). Another approach is the development of a stochastic model to estimate soil moisture distributions using daily rainfall and an initialization of soil moisture values (Farago, 1985). However, both the API and stochastic approach require an initial condition for soil moisture at the location where estimates are desired. This hampers applicability at many locations that do not have soil moisture sensors.

Other models have taken a hydrologic approach, employing precipitation and surface radiation to estimate soil moisture (Capehart and Carlson, 1994), but these models require boundary conditions, initial conditions, and parameters of a thermal and/or hydraulic nature that can be difficult to obtain broadly. Pan et al. (2003) and Pan (2012) addressed this concern by deriving a "diagnostic soil moisture equation" from a stochastic, linear partial differential equation. Soil moisture then becomes a function of a temporally-decaying sum of previous rainfall. Their approach no longer requires an initial condition, nor recalibration, but does require a soil moisture sensor at the location in question to calibrate the equation initially. Measuring soil moisture directly is plausible, but soil heterogeneity may necessitate numerous sensors to address spatial variation of soil moisture adequately (Pan and Peters-Lidard, 2008). The alternative approach of a soil water balance can be applied, but must be recalibrated frequently, since errors are cumulative (Jones, 2004).

In the agricultural arena, Gamache et al. (2009) developed a soil drying model, but its predictions require data from cone penetrometer and soil moisture sensors, two data sources that are not

* Corresponding author. Tel.: +1 610 639 2087.
E-mail address: ecooper2@gmail.com (E.J. Coopersmith).

http://dx.doi.org/10.1016/j.compag.2014.04.004
0168-1699/

currently available at most remote sites. Another inquiry along similar lines uses knowledge of soil types, which is theoretically public, but then continues to require soil moisture levels from proximal sensors (Chico-Santamarta et al., 2009).

Other approaches eschew the strategic placement of soil sensors in favor of modeling tire slip as a function of tractor properties (Sahu and Raheman, 2008) or other details such as vehicle type, speed, load distribution, number of passes, etc. (Pytka, 2009 and Lamande and Schjonning, 2008). These vehicle-specific properties are often unavailable outside of research studies. Another early work attempts to assess the suitability of site conditions, but uses very specific information that is not likely to be available to most applications, such as "stress–strain rate relations" or the results of triaxial tests (Sharifat and Kushwaha, 2000).

Lee and Wang (2009) focus solely on radar and other remote sensing data, but consider only the properties of snow coverage along with the hardness and density of mixtures of snow and ice. These traits are not appropriate for warmer weather conditions of interest to agriculture. Sliva et al. (2009) modeled drying properties of sugarcane fields in Brazil, but their model requires that the prediction occur in a well-specified location within a pre-determined time frame.

Alternatively, considerable prior research seeks techniques for improving agricultural conditions (Tullberg et al., 2007; Shoop et al., 2002 and Lebert et al., 2006) or minimizing the effects of traffic (Raper, 2005) rather than delivering a dryness assessment. For instance, one paper recommends a protocol for improving drying via the application of manure (Mosaddeghi et al., 2000).

This paper addresses these gaps by developing and testing a machine learning model of soil drying that requires only precipitation and potential evaporatranspiration estimates. Precipitation is widely available at high temporal resolution on a 1 km by 1 km grid from NEXRAD[1] throughout the continental United States. Potential evapotranspiration is available publically in Illinois from the Illinois State Water Survey, and can be estimated in other locations using three approaches (Jensen et al., 1990). The first method requires only air temperature and day length (Thornthwaite, 1948; Hamon, 1963). The second method requires air temperature and net radiation (Priestley and Taylor, 1972). The third, and most detailed approach, requires the information from the second as well as wind speed and relative humidity (Monteith, 1965). The Illinois Climate Network (ICN) data used in this analysis employs the third approach, but one of these three approaches should be applicable anywhere throughout the United States.

For the purposes of this analysis, the notion of dryness represents a user-defined assessment with qualitatively consistent designations for a particular application. For example, diverse agricultural activities (crops, livestock, etc.) may possess different notions of acceptable soil conditions, but provided the algorithm is given training data consistent to one particular context, it will adapt appropriately. This current analysis focuses upon a general test case for agricultural soil drying, where "dry" implies that a given tract of farmland is viable for a particular type of work (e.g., planting, crop treatment, or harvesting) on a given day.

Section 2 presents the case study, a brief overview of the geography and relevant features of the South Farms test site that is the focus of the data analysis presented in this work. Next, Section 3 describes the methodology used in remotely estimating dryness from public data sources. Section 4 then presents the results of the case study application and compares the relative performance of the algorithms. Finally, the paper concludes in Section 5 with an assessment of which machine learning techniques have performed most successfully, a discussion of these results in the context of previous work, and a brief discussion of potential future enhancements and other applications of this research.

[1] http://nmq.ou.edu/beta/q2-tools.html, provided through the University of Oklahoma.

2. Case study – South Farms, Urbana, IL

The methods developed in this study are tested at the University of Illinois South Farms located in Urbana IL, which is classified as a continental or microthermal climate, Dfa by the Koppen–Geiger classification system (Koppen, 1936; updated by Peel et al., 2007). The specific climate zone is characterized by a warm, humid summer and colder, drier winters. Annual rainfall levels, gathered from 1990 to 2011 at the ICN sensor located near the test site (Fig. 2.1), average approximately 1013 mm per year. The potential evapotranspiration estimate over the same time period is 1046 mm per year. The warmest month is July and the coolest is January, with average daily temperatures of 75.1 and 26.9°F (24.0 and −2.8 Celsius) respectively. As precipitation levels and potential evapotranspiration are both highest during the summer (the middle of growing season), the flat landscape yields a test site that will be characterized by multiple periods of wetting and drying during any growing season. Agricultural sites in this region are often tile-drained, which results in a shorter soil drying system memory than similar locations without the tile drains.

Fig. 2.1 presents the location of the test site, located within the Energy and Biosciences Institute (EBI) energy farm. Also pictured is the ICN sensor platform used in this analysis. The ICN sensors provide readings of potential evaporation (which incorporates solar radiation, humidity, wind, temperature, etc.) and precipitation. To the southeast are the largest plots maintained by EBI, upon which soil condition assessments were gathered.

A John Deere intern, Jordan Pitcher, provided assessments of soil conditions throughout the growing season within the green square. Mr. Pitcher has extensive agricultural experience and his assessments served as the soil dryness training and validation data for the machine learning algorithms. The sensors labeled "EBI" provide precipitation information.

3. Methodology

This section describes the framework developed for assessing dryness based on soil drying. An overview of the approach is first provided, followed by a discussion of the various input data sources, a description of the algorithms used to assess soil drying and their outputs, and concluding with the computational tools and requirements for implementation.

The first approach, the k-nearest-neighbor (KNN) algorithm, which was introduced by Fix and Hodges (1951) and deployed in many water resources and hydroinformatics applications (e.g., Kumar et al., 2006, Meliker et al., 2008, McRoberts et al., 2007; Nemes et al., 2008 and Coopersmith et al., 2011), is an intuitively satisfying approach for classification, analysis, and forecasting. The algorithm simply uses current precipitation and potential evapotranspiration measurements to locate the most similar examples from historical data (whose field conditions are known) and, in turn, leverages those similar examples to estimate the current field readiness.

The second algorithm, decision trees (also referred to as classification or regression trees), are non-parametric classification tools that recursively split datasets by values of the independent variables to minimize entropy in each subset and, thus, maximize information gain (Breiman et al., 1984). These algorithms are available in most statistical programming packages (Breiman et al., 1993) and have been deployed in a variety of environmental contexts, such as sustainable forest resource management (Aertsen

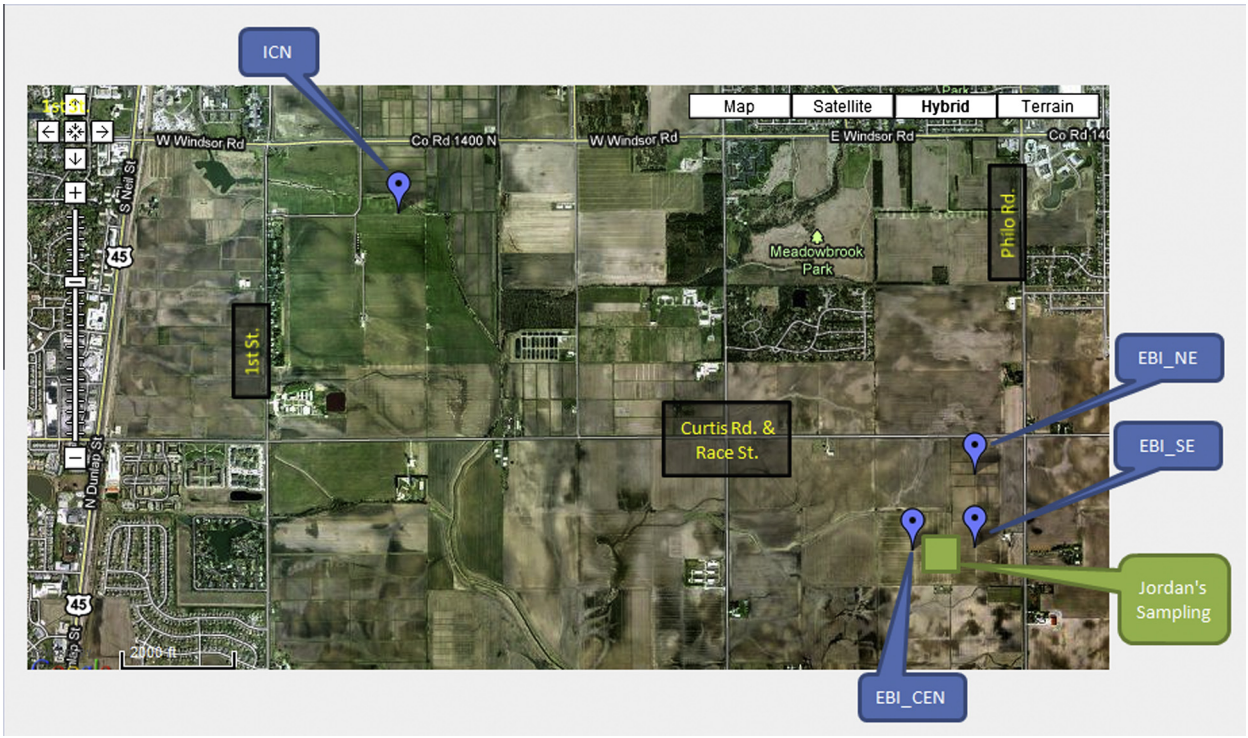

Fig. 2.1. The South Farms, Urbana-Champaign, IL. Sampling location shown in green, Illinois Climate Network (ICN) sensor in northwest in blue (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

et al., 2011), crop identification for soil management (Pena-Barragan et al., 2011), and image classification for mapping the vegetation across arid rangelands (Lailiberte et al., 2007) – an enhancement of earlier decision tree work on land cover (Brodley and Freidl, 1997). Tree-based models have even been constructed in the field of finance to predict the failures of business ventures (Li et al., 2010).

The final algorithm, boosted perceptron, builds from the perceptron – the simplest, single-layer, feed-forward form of an artificial neural network (Russell and Norvig, 2010, p. 729). Perceptrons, like other neural networks and more traditional statistical methods such as nonlinear regression, fit coefficients to predict the values of independent variables by iterating over a dataset and adjusting those coefficients to improve predictions. The boosting process, using the *adaboost* algorithm (Freund and Schapire, 1997, described by Russell and Norvig, 2010, p. 751), has been utilized to solve a variety of environmental problems, from predicting fishery catches (Li et al., 2011), to aiding forest managers map those locations at greatest risk to forest pests (Haywood and Stone, 2011) and, like classification trees, for classification of land cover from remote imagery (Stavrakoudis et al., in press). Boosting allows multiple predictors to be developed, with each having a 'vote' in the ultimate prediction; the relative impact of each 'vote' is a function of that predictor's overall accuracy – implementation details are provided in Russell and Norvig (2010).

The soil drying assessment methodology is summarized in Fig. 3.1. On the top row, four data sources are presented, including training and validation data from volunteers and the precipitation and potential evaporation inputs which define climatic conditions. Next, these data are fetched, stored, and ultimately assimilated and formatted as input streams for the three machine learning algorithms shown in the rounded rectangles at the bottom of Fig. 3.1. All three algorithms generate outputs which estimate dryness either as a binary classification ("dry" or "wet") or as a probability of a given classification.

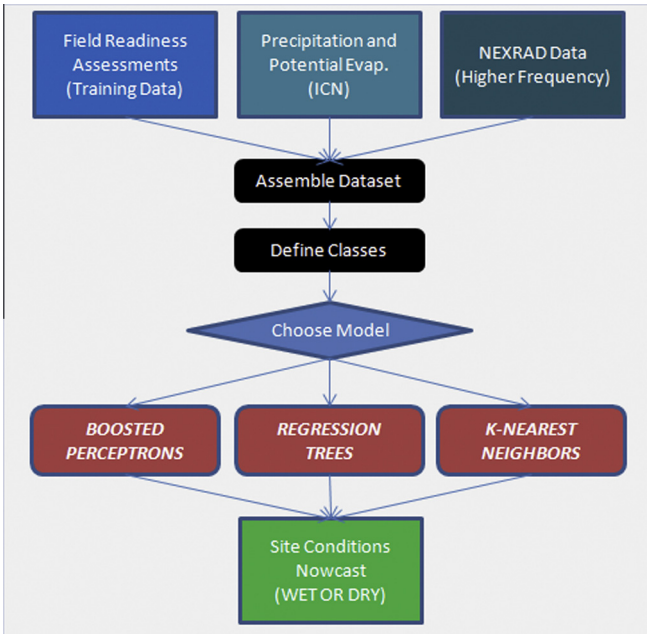

Fig. 3.1. Flow chart of the approach data sources.

The first source of input data is Nexrad radar data (the third box from the left, top row, Fig. 3.1. This high-frequency radar allows for precipitation readings hourly with nearly complete national coverage at approximately 1 km × 1 km granularity. The second data source, from the ICN, provides potential evaporation at each of their 20 sensors throughout Illinois.

The ICN sensor provides hourly readings of potential evapotranspiration rates using the modified Penman–Monteith equation (Monteith, 1965). This calculation combines latent heat of evaporation, net irradiance, and constant values for specific heat of air,

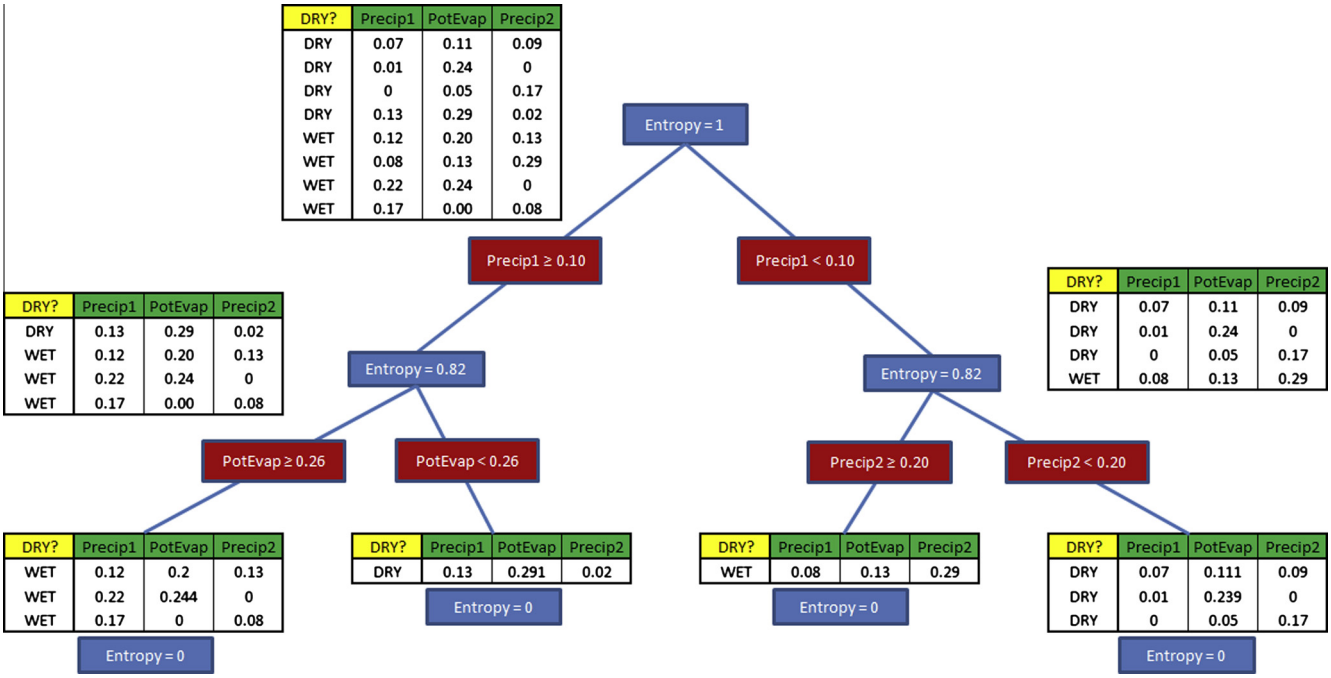

Fig. 3.2. Sample classification tree.

specific humidity, dry air density, conductivity of air, conductivity of stoma, and a psychrometric constant. The ICN sensor (see Fig. 2.1) provides an opportunity for assessing the drying process using public data. Finally, the values of soil dryness assessments, illustrated by the upper-left box of Fig. 3.1, are the soil condition data for model training (fitting the model parameters) and testing (validation on data that were not used for training) given the current meteorological conditions and those from the most recent days. This qualitative metric is defined on an integral scale from 1 to 5, with the following description:

1 – The site is impassable; the equipment used might literally become stuck in the mud.
3 – The site is usable, but the equipment would leave deep ruts.
5 – The site is dry, the ground is hard, the equipment will leave only shallow track.

The values of 2 and 4 allow for descriptions which fill the gray areas between 1 and 3 or 3 and 5 respectively. As a means of developing a binary classification, a rating of 1 or 2 will be considered "wet," while a rating of 3, 4, or 5 will be considered "dry." Before settling upon this human-based, and therefore inherently subjective measurement, an alternative was evaluated. A cone penetrometer, a tool which measures the quantity of pressure in pounds per square inch needed to compress the soil a given distance, was used in conjunction with the qualitative assessments. While its readings do bear some non-trivial correlation to the qualitative dryness metric, it is not reliably consistent with the expert assessment of whether or not the site conditions were appropriate for use at any given time.

Though naturally, a site with zero moisture that is nearly incompressible is "dry" and an extremely soft, wet soil is "wet," this distinction was found to be insufficient for assessing drying. When objective cutoffs for penetrometer readings were determined such that the proportion of "dry" days as chosen by the expert was equal to the proportion chosen by the penetrometer, disagreement occurred on over 30% of the days measured. Simply put, soil drying is a phenomenon that is not readily assessed by a single physical measurement.

3.1. Machine learning algorithms

This section will introduce the three machine learning algorithms deployed for assessing soil dryness conditions. The first subsection will address classification trees, the second will discuss k-nearest-neighbor algorithms, the third will illustrate boosted perceptrons, and the final subsection will present the techniques used to apply these algorithms to soil drying assessments.

3.1.1. Classification trees

Classification trees, specifically the implementation most commonly available (Breiman et al., 1993), function via the iterative dichotimiser algorithm, ID3 (Quinlan, 1986). Predicated upon Occam's razor, which stipules that a simple theory is generally preferable to a more complex one, a classification tree aims to minimize the error associated with a prediction by splitting at each node based upon maximum information gain. A simplified example is presented in Fig. 3.3 where eight hypothetical drying scenarios are classified either as dry or not (labeled "YES" or "NO"). The values of each independent variable are presented alongside the dryness classifications in the boxes. "Precip1" and

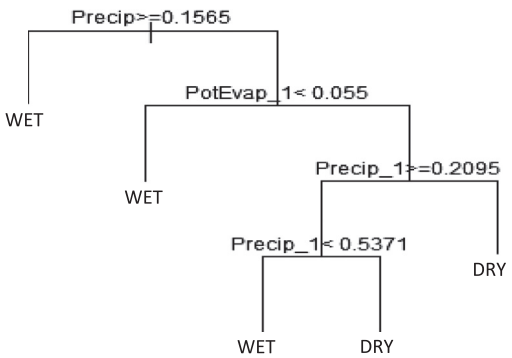

Fig. 3.3. Example of an efficient classification tree, considering only relevant features. Note that if precipitation exceeds 0.1565, it is the only variable considered.

"Precip2" represent rainfall over the most recent 24 h and from 24 to 48 h, respectively.

Fig. 3.2 illustrates that initially, there are four days classified as dry and four that are not (thus a blind guess at any given day's classification would have a 50% chance of being correct). By dividing the dataset into days with Precip1 $\geqslant 0.10$ and days with Precip1 < 0.10, two groups of four emerge, each of which have three days of one classification and a fourth of the other. This is "less disordered." A prediction based only on this single statistic, precipitation from the last 24 h, would classify six of the eight days (75%) correctly. Finally, the two groups of four are split on potential evapotranspiration and Precip2 to yield four fully-organized clusters.

In the specific case of soil drying, consider a dataset of size n, in which x examples are "dry" and $n-x$ examples, therefore, are "wet." Let us define information gain in terms of decreased entropy, where entropy is defined as follows:

$$E(S) = -\sum_{i=1}^{n} p_i * log_2(p_i) \tag{3.1}$$

where $E(S)$ represents the information entropy of set S, n indicates the total number of examples in S, and p_i signifies the proportion of examples which meet criteria i. Inspection reveals that for perfect classification, in which all elements in S are of a single class, entropy is equal to zero. Conversely, for maximum entropy, in which each class i is equally represented within S, entropy is equal to unity. Thus, for the specific case of soil drying:

$$E(S) = -\left[\frac{x}{n} log_2\left(\frac{x}{n}\right) + \frac{n-x}{n} log_2\left(\frac{n-x}{n}\right)\right] \tag{3.2}$$

Once entropy is computed for the entire dataset, an attribute A is chosen, such that information gain in terms of entropy is maximized:

$$G(S,A) = E(S) - \sum_{j=1}^{m} p_j(A_j) E(S_{A_j}) \tag{3.3}$$

where $G(S,A)$ is the information gain in set S (initially the entire data set) after splitting the dataset using attribute A, m represents the number of different possible segments of the continuous values of attribute A present in set S (for all trees discussed in this analysis, $m = 2$, as each tree is split into exactly two branches at each node), A_j refers to all values of attribute A that fall within segment j, $p_j(A_j)$ indicates the proportion of examples with attribute A_j in S, and S_{A_j} is simply the subset of S containing only examples characterized by attribute A_j.

After the first split of the dataset, further splits (nodes) are added iteratively, maximizing information gain at each step (greedy) and stopping when the number of constituent members of each leaf falls below a user-defined value deemed too low for further splitting. Alternatively, splitting can also terminate if no attribute exists such that a positive information gain can be achieved.

The advantage of classification trees, especially in circumstances where the number of examples within the training set is limited, is their ability to ignore certain attributes in situations where their information is irrelevant. Where many machine learning algorithms must utilize every feature at their disposal for each example presented, a classification tree can determine that, given one feature, the others need not be considered. Consider the case in which an intense rain event has occurred over the last six hours. Clearly, the field will be soaking wet, and unfit for use. The precipitation data from two days previously and any notion of yesterday's potential evapotranspiration rate becomes extraneous. Thus, one branch of the classification tree may require only one split from the root node, yielding a classification instantly under certain conditions, as shown in Fig. 3.3.

In the tree above, if recent precipitation exceeds a certain value, the field is classified as wet without even considering potential evapotranspiration or earlier days' rainfall. This allows the power of the algorithm and the information from the available training data to be focused upon the conditions in which soil conditions are most uncertain. With limited data, adding unnecessary features into the decision process may introduce noise without helping discern true signals. Classification trees avoid this concern, allowing for a larger number of decision variables to enter the model, while only those that conditionally prove most influential are employed for prediction.

3.1.2. k-nearest-neighbors (KNN)

The k-nearest-neighbor algorithm (KNN) searches a database for historical examples that are most similar to current conditions and then determines the proportion of those historical examples that possess the property under examination for prediction (in this case, a particular dryness rating). Generally, similarity is determined using a simple Euclidian distance function in attribute space. Consider an input vector as follows:

$$X = (x_1, x_2, x_3, \ldots, x_n) \tag{3.4}$$

Aligning the scales of each independent variable's distribution can be achieved by simply applying the following transformation of each non-normalized variable, z_i:

$$x_i = \frac{z_i - \mu_i}{\sigma_i} \tag{3.5}$$

where μ_i represents the mean of the distribution of variable i, and σ_i denotes the standard deviation with respect to variable i. Next, let us consider a training example Y, replete with the same features as X, also normalized by the same distributions for all variables i:

$$Y = (y_1, y_2, y_3, \ldots, y_n), y_i \sim N(0,1), \quad \forall_i \tag{3.6}$$

The n-dimensional distance function (Kumar et al., 2006, p.394) can be applied between vectors X and Y:

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \quad \forall_i \tag{3.7}$$

Thus, for a given set of current conditions X, a distance value can be determined between X and every element of the historical database. Next, a simple sort algorithm is performed, and the k historical examples with the lowest distance values d are selected as a 'similar set'. At this point, in the case of soil conditions, the proportion of examples contained in the similar set which were classified as "dry" becomes the best estimate of the probability of that classification for the current situation. Essentially, KNN asks the question "what happened historically when conditions looked similar to the way they do today?"

While elegant and satisfying, the curse of dimensionality becomes unavoidable with an algorithm of this nature as more features are added, especially when the number of historical training and testing examples is low (on the order of 10^2). With each additional feature of comparison, another independent variable dimension appears, and it becomes exponentially more difficult to locate similar matches. This requires that the value chosen for k (user-selected) must decrease exponentially with each added dimension. That is, with each added dimension, either many fewer matches are considered "similar" and/or the standard of similarity becomes decidedly more lax.

To illustrate this point, consider field conditions in which no rain has fallen in the past 48 h. If the algorithm searches for, say, ten similar days historically, it is likely that there will be ten days

for which little or no rain has fallen for 48 h. However, when we add another dimension, such as a high level of potential evaporation, finding "similar" examples becomes more challenging. Some of the previously located ten similar examples will have higher rates of potential evaporation while others will be lower. Adding the dimension of potential evaporation results in fewer similar matches (choose from the previous ten similar matches the examples with high rates of potential evaporation) or less similar matches (insist on ten matches and accept that a few will contain non-trivial amounts of rain or lower rates of potential evaporation).

The previous issue notwithstanding, KNN produces a natural binomial distribution in its outcomes, as each element of the similar set becomes, essentially, a "voter" in a classifying election. Consider the simple case where $k = 9$. The most straightforward classification emerges by selecting whichever of the binary classes characterizes five or more members of the similar set. Alternatively, a confidence interval can easily be constructed:

$$P(A) = \frac{a}{k} \pm \frac{\left(\frac{a}{k}\right)\left(1-\frac{a}{k}\right)}{\sqrt{k}} z \tag{3.8}$$

where $P(A)$ is the probability that the current event falls within class A, a is the number of elements in the k-most similar set from class A. The Gaussian variable z specifies the confidence interval's scope, i.e. $z = 1.96$ for a 95% confidence interval and $z = 2.58$ for a 99% confidence interval.

To avoid a scenario in which all k elements of the similar set possess the same classification (i.e., a degenerate confidence interval case), two dummy examples are added to each similar set, one of which always reads "dry" and one of which is perpetually labeled "wet." In this manner, no classification will possess an unrealistic zero margin of uncertainty.

3.1.3. Boosted perceptrons

The final algorithm to be considered, also generally effective given limited information, is adaboost. Adaboost is an example of ensemble learning, which is used to generate multiple hypotheses (perceptrons) regarding the underlying function being modeled and to combine each resulting prediction and relative likelihood of accuracy (Russell and Norvig, 2010). In fact, adaboost is a mechanism to improve the performance of any weak learning algorithm, but for the purposes of this discussion, this section will discuss the "boosting" of a simple perceptron.

Perceptrons represent perhaps the most basic classifier, the simple linear separator. Consider, again, a series of examples, each characterized by a vector of features as given by Eq. (3.4). Next, a weight vector, is constructed as follows:

$$W = (\omega_1, \omega_2, \omega_3, \ldots, \omega_n) \tag{3.9}$$

Such that:

$$\sum_{i=1}^{n} x_i \omega_i > \theta \tag{3.10}$$

θ serves as a threshold with which to delineate the two classifications. That is, the dot product of the weight vector and the input variables yields a scalar value. Values greater than θ output one classification while those below θ output another.

In the case of a problem involving modeling of the physical environment, especially one in which assessments are likely to be made with imperfect or incomplete information, it will likely be impossible to construct a linear separator (such as the one in Eq. (3.10)) with every training example classified correctly. For this reason, adaboost improves model error by increasing the significance of those examples classified incorrectly, and decreasing the significance of those points for which the training algorithm was correct. In each iteration, a new linear separator is developed to minimize the weighted sum of squared errors. If certain examples are repeatedly misclassified, their errors will magnify until eventually, the best separator becomes one which will classify those examples appropriately.

The general theory of boosting stipulates that for any weak learner (one that classifies at least as well as a random guess), with a sufficient number of iterations, perfect classification will occur over all training examples, although performance in validation will plateau considerably sooner (Russell and Norvig, 2010).

Though there are numerous numerical patterns through which misclassified examples can be magnified and correctly labeled examples can be diminished. Adaboost's technique, used in this work, is given below.

Consider a series of m examples:

$$(X_1, C_1), (X_2, C_2) \ldots, (X_m, C_m) \tag{3.11}$$

where each X_i is a vector of features, akin to Eq. (3.4), and each C_i represents a binary classification. Initialize a weight for each example:

$$D_t(i) = \frac{1}{m}, \quad \forall_i, t = 1 \tag{3.12}$$

Determine a classifier, h_t, such that any vector of independent variables maps to -1 or 1. This is achieved via a threshold function as shown in Eq. (3.10). Stated mathematically:

$$h_t(X_i) \rightarrow, \{-1, 1\}, \forall_{i,t}$$

$$h_t = \underset{h_t \in H^{\epsilon_t}}{argmin} \quad \text{where } \epsilon_t = \sum_{i=1}^{m} D_t(i) * \left\{ \begin{matrix} 0 \text{ if } C_i = h_t(X_i) \\ 1 \text{ if } C_i = h_t(X_i) \end{matrix} \right\} \tag{3.13}$$

Stop if $\epsilon_t \geq 0.5$, as this violates the principle of superiority to a random guess.

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{3.14}$$

At this point, it becomes evident why the error rate ϵ_t of classifier h_t must be strictly less than 0.5, otherwise Eq. (3.14) produces an undefined result. Next, $D_t(i)$ is updated:

$$D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t C_i h_t(x_i)}}{\sum_i D_t(i) e^{-\alpha_t C_i h_t(x_i)}} \tag{3.15}$$

The denominator is simply a normalization term to ensure that the weights of each example sum to unity. The final classifier behaves as a weighted aggregate of each iteration:

$$H(X) = sign\left[\sum_{t=1}^{T} \alpha_t h_t(X)\right] \tag{3.16}$$

Once again, like the KNN algorithm, this algorithm can be manipulated slightly such that each classification falls between 0 and 1. First, rather than classifying in a binary manner of -1 or 1, simply choose 0 or 1. Second, in Eq. (3.16), a removal of the sign() operator will yield a weighted average of 0 s and 1 s. Thus, once again, a binomial distribution can be modeled and a confidence interval can be developed as shown in Eq. (3.8) for KNN, simply replacing $\frac{a}{k}$ with $H(X)$.

3.1.4. Application of the algorithms

For each of the three algorithms discussed, certain problem-specific adjustments are required to ensure the production of meaningful results. First, it is important to recognize that there are certain summer days for which obtaining a correct classification becomes trivial, simply because no precipitation has been seen for extended periods of time. No computational algorithm is required to inform the farmer of soil conditions after a sufficient

duration of dry conditions. Consequently, so as to avoid artificially enhancing the algorithm's accuracy, all examples for this case study in which no rain has fallen within the three previous days are eliminated from the testing set. Three days exceeds the longest drying period observed in these tile-drained agricultural fields, although a longer period would likely be necessary in less well-drained areas. This approach ensures that the algorithm's accuracy is measured only on examples for which the nowcast is non-trivial.

Second, as alluded to in the previous section, a substantial validation set is held aside from the collected dataset that is not used to calibrate the machine learning algorithms. By testing the best model only on training examples from previously unused data, external validity is verified and the probability of that model functioning effectively for future assessments of soil conditions increases. It is worth noting that, as the growing season of 2010 represents the entirety of the available data for this particular analysis, the training set contains 34 of the 109 days for which data have been gathered, a figure which could be perceived as fairly low considering that many machine learning algorithms train on much larger proportions of the data. This occurred because the beginning of the growing season in the spring contains considerably more wetting and drying events than the summer and fall. Should the training/testing ratio approach the more common 80%/20%, the validation set might not contain a single substantial period of storms. Moreover, random assignments to training and testing sets cannot be made in this case, as subsequent training examples are, naturally, dependent on previous examples, as drying involves a continuously varying, physical system. Thus, the training data consists of data taken on or before June 24, 2010, and testing data consists of the growing season thereafter (ending September 24, 2010).

Finally, precipitation data older than 48 h are not included in the models as independent variables. While there are certainly numerous locations in which rainfall's influence on the height of the water table, or even the dryness of the topsoil, might endure for weeks or even months, the fields at the South Farms test site are all tile-drained and empirical examination of the auto-correlation in soil moisture reveals that only the first two days affect dryness assessments.

3.2. Implementation

The machine learning algorithms constructed for this analysis were implemented in R, version 2.11.1, for Windows Vista. The implementation of the classification tree algorithm and the visualization thereof was aided, in part, by the *rpart* library. The remaining two algorithms (KNN and boosted perceptrons) were developed from scratch in R. The input variables are the precipitation data over the past 24 h, the precipitation data from 48 h previously, and the potential evapotranspiration estimates from the previous 24 h. Daily precipitation rates are obtained by weighted summing of hourly values from NEXRAD, as shown in Eq. (3.17 (and the latter from a simple sum of the past 48 hourly rainfall values).

$$Precip\_1 = \frac{\sum_{i=1}^{24} P_i * \left(\frac{i}{24}\right)^3}{\sum_{i}^{24} \left(\frac{i}{24}\right)^3} \tag{3.17}$$

where P_i represents the precipitation falling in hour i. In this weighting, $i = 1$ refers to rainfall from 23 to 24 h before the time in question and $i = 24$ refers to the most recent hour's rain. The weighting is used because many models (e.g., the diagnostic soil moisture equation by Pan et al., 2003; Pan, 2012), recognize that more recent precipitation plays a more meaningful role in predicting current precipitation. The 48-h series is un-weighted to provide a contrast between a recent storm, amidst otherwise dry weather and a two-day period that has been consistently wet. Other weighting schemes (or none at all) were also examined but they did not perform as well as this approach. Precipitation readings, from NEXRAD data, are computed using inverse-distance-weighted interpolations (Shepard, 1968) from a 5×5 grid of 1 km by 1 km precipitation radar values that surround the test site.

Potential evaporation estimates from the ICN's hourly sensor readings are summed from the most recent 24 hourly ICN values using a sliding 24-h window.

In terms of computational demands, an office laptop (these algorithms were run on an intel i5, 2.53 MHz) was sufficient for timely execution. For classification trees, the tree could be placed in memory in a matter of a few seconds with individual conditions evaluated almost instantaneously. For the KNN algorithm, a historical database could be entered into memory and back-tested in under one minute and individual queries occurred in under one second. Finally, for the boosted perceptron algorithm (which is the most computationally expensive), the multiple iterations required to develop the various linear separators could generally be implemented in under ten minutes, with individual conditions tested almost instantly.

It is worth noting, however, that these three algorithms respond differently to increasing scales. While a classification tree will take longer to construct with more examples and more features to model, once a tree is entered into memory, any new example can be classified very quickly, even with a very large tree. Classification time for a single example is approximately $O(\log n)$ where n is the size of the historical database, and thus, grows very slowly. For KNN, entering the database into memory is simply $O(n)$, but back-testing any example requires a sort of the data, which is, at minimum, $O(n \log n)$. Thus, for very large datasets, the computational expense could become substantial. Boosted perceptrons can become time consuming to construct as datasets become very large, especially as the number of linear separators becomes large. Fitting the weights of the linear separators (the coefficients associated with each independent variable) through stochastic gradient descent runs in $O(nm)$ time where m is the number of features and n represents the number of training examples. However, once a boosted separator is placed in memory, new examples can be classified almost instantly, as that computational time grows as $O(m)$ where m is the number of features. Thus, so long as our variable space is unchanged, more data will not slow classifications.

4. Results

In this section, results will bfe presented for each of the algorithms investigated in this research, beginning with classification trees, proceeding to k-nearest-neighbors, and then finishing with the boosted perceptron results. The section will conclude with a comparison of the relative accuracies of each algorithm.

4.1. Classification trees

For each algorithm, calibration and testing began as data arrived throughout the growing season. In the case of classification trees, during the very first attempts at constructing predictive models, the ability of classification trees to ignore irrelevant information in specific situations allowed for simple trees which outperformed other algorithms. However, as the data set became increasingly rich, the performance of the trees was rapidly surpassed by the remaining two algorithms. The classification tree was trained on the 34 dates on or before 6/24/10 and tested on 75 days thereafter. The date was selected to ensure that two storms (wetting/drying events) occurred during testing data. Employing a more traditional 80%/20% calibration/validation ratio

(or even 50/50) would place all of the storms in the calibration set, then asking the algorithm to predict only drier dates. This tree, which appears in Fig. 4.1, includes three features: the last 24 h of precipitation, with more recent rain weighted more heavily (see Eq. (3.17)); precipitation from the last 48 h; and the most recent 24 h of potential evapotranspiration.

It is worth noting that this tree contains only five terminal leaf nodes, as the training data consist of only thirty-four examples. To avoid over-fitting the data and subsequent drop-offs in performance during validation, splitting was prohibited if a given node contained fewer than eight elements and no split was considered acceptable if fewer than three elements remained in any terminal leaf.

Note that in Fig. 4.1, if the inequality is *true*, the algorithm proceeds to the left branch of each bifurcation. For example, if "Precip_2," which refers to all precipitation falling within the previous 48 h (mm), is too low (not much rain recently), the algorithm chooses "DRY" and does not even consider any other features.

This tree's performance (see Fig. 4.2), at roughly 88% on the validation data falls below that of other algorithms. It is important to recognize that the actual dryness is scaled 1–5, while the algorithm attempting to classify soil conditions only returns two values ("dry" or "wet"). A "correct" classification occurs if and only if both the blue and red lines fall within the same shaded region. Thus, for example, after one error in early September, the algorithm is correct on every subsequent day, despite the distance between the red and blue lines. The one day on which the blue and red lines overlap (9/22) is no more "correct" than the day before or after.

Despite the 88% accuracy of the tree in Fig. 4.1, careful observation reveals that potential evaporation data never becomes a splitting criterion. Evidently, there is always a more effective means of increasing information gain, considering that classification trees must consider variables one at a time. With this in mind, a "complete" tree can be constructed, splitting until either: (a) nodes contain a single example, or (b) no improvement is possible on training data. Given consistently labeled examples, a decision tree ought to correctly classify all training data. However, over-fitting a tree in this manner can lead to deteriorating performance in validation.

Once again, decision trees possess the inherent advantage of focusing upon the variable which produces the maximum quantity of variance at any given stage of the decision-process, yet, unfortunately, they fail to utilize all relevant features without over-fitting the data. While the aforementioned strength outweighed the weakness during initial testing (when data limitations were extreme), the following algorithm improved upon them.

4.2. k-nearest-neighbors

Next, the best binary classifier is constructed using the k-nearest neighbor algorithm with, in this case, three input variables. As described for the classification tree algorithm, the first variable represents an aggregation of all precipitation within the most recent 24-h period, weighting more recent rainfall more heavily. The second represents a simple aggregate of all rainfall within the most recent 48-h period. The third is the potential evapotranspiration over the most recent 24 h. For the sake of visual clarity, in Fig. 4.3 the red line for KNN prediction is scaled closer to five when increasing numbers of the similar set classify the example in question as "dry" and closer to one when more members of the similar set classify to "wet." The scale system used ensures that 5.5/11 votes would classify to 2.5 – the boundary between the green and red shaded regions. In other words, a 50/50 split of the similar set would yield a forecast on the threshold between dry and wet. In this case, only five errors in classification occur, bringing our accuracy up to 93% within the validation set. Moreover, the cases in which errors occur are those for which the eleven nearest neighbors have split votes.

In Fig. 4.4, a binomial, 99% confidence interval is computed using the KNN votes and dummy votes as shown in Eq. (3.8). Note that the five errors (cases in which the human's assessment is green and the computer's prediction is red or vice versa) all occur where the boundary (dotted-line) between dry and wet falls within the confidence interval. That is, the algorithm never errs outside of its designed margin for error in seventy-four classifications. Real-time use of such an algorithm could occasionally cause uncertainty in cases where a field is truly "dry" by predicting a probability of that classification near 50%. However, if the algorithm predicts that a field is viable at a given point in time with a high probability, a farmer could act with confidence that his/her site is, in fact, dry.

4.3. Boosted perceptrons

Lastly, binary classifications are developed using the boosted perceptron algorithm with the same three inputs as deployed for the k-nearest-neighbor algorithm. Like KNN, the algorithm's ultimate prediction is the aggregate of multiple "votes," in this case, individual perceptrons with weights as shown and calculated in

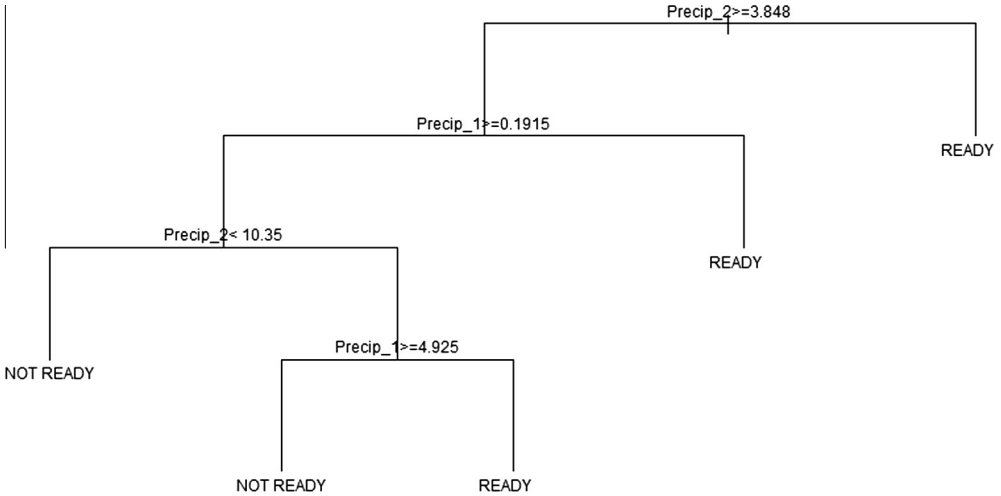

Fig. 4.1. The "best" classification tree – highest rate of agreement with manual assessment.

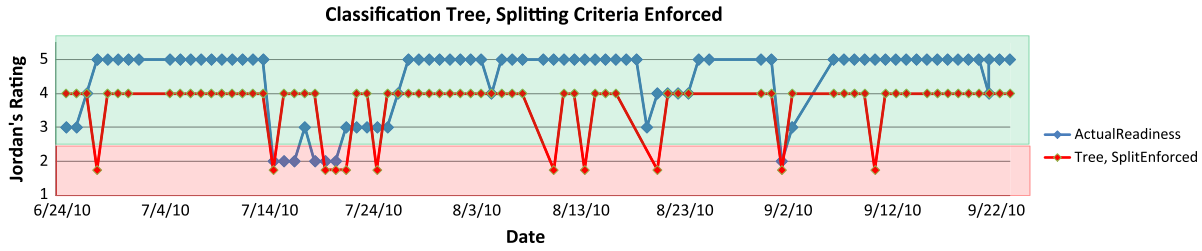

Fig. 4.2. (top) – The "best" classification tree, with splitting criteria enforced, actual (blue) vs. predicted (red) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

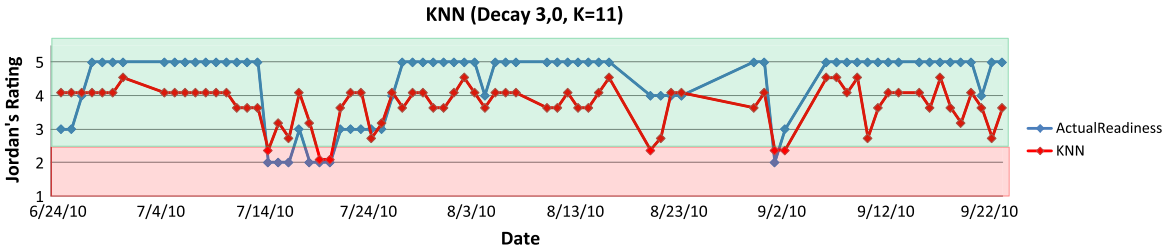

Fig. 4.3. (middle) – KNN, actual (blue) vs. predicted (red) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

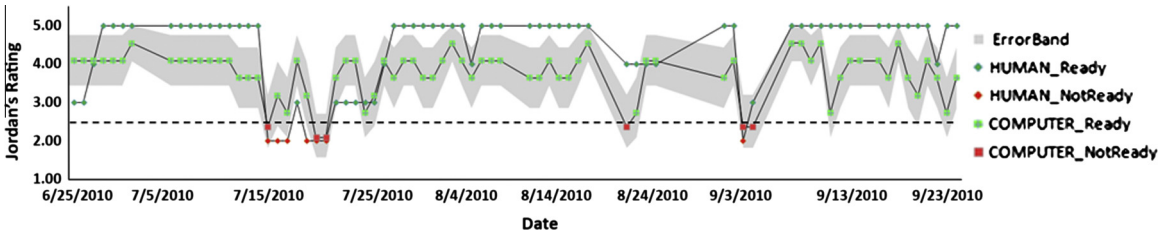

Fig. 4.4. (bottom) – KNN, actual (blue) vs. predicted (red), with binomial confidence interval (gray band) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

Eq. (3.14). Thus, any classification is scaled between 0 and 1 and can be visualized on the 1–5 scale such that any classification greater than 0.5 (dry) appears above 2.5 on the *y*-axis, as shown in Fig. 4.5. The performance is quite similar to the KNN algorithm, as 92% of all points are classified correctly, and several dates with errors overlap between KNN and boosted perceptrons. However, the KNN forecasts generally yield estimates closer to 50% while the boosted perceptron algorithm outputs more extreme predictions (closer to 0 or 100%) – this is a mathematical relic of the fact that the most similar examples in KNN often contain both "dry" and "wet" days, as well as off-setting dummy votes. The boosted perceptrons, without the advantage of a "dummy" vote, often will classify 100% toward one designation or the other. Encouragingly, in 34/35 (97%) of the cases in which the boosted perceptron algorithm is "certain," that is to say that each individual classifier agrees on the classification, the ultimate conclusion is correct. The erroneous case will be discussed in a later section.

As was the case with KNN, a confidence interval can be constructed around the 0–1 classification found with boosted perceptrons, then scaled from 1 to 5 for visualization purposes. The results are shown in Fig. 4.6. As is evident from the above chart, on several dates the confidence interval is degenerate, that is to say, of nil width. Despite this, there are only two cases in which a misclassification occurs without the confidence interval crossing the boundary between dry (green shaded region) and wet (red shaded region).

4.4. Algorithmic comparison

Finally, in Fig. 4.7 the accuracy of the three algorithms is compared. The three bars compare the accuracy of each of the algorithms tested on validation data that was not used for training. The classification trees have the worst performance of the three algorithms, misclassifying several examples that the other two algorithms classify correctly, while KNN performs the best – with even stronger performance after a confidence interval is introduced.

To explore these differences further, Fig. 4.8 compares daily predictions from the three algorithms with the farmer's assessments. It appears that there are a handful of "aberrant" examples within the training set (8/21, 9/2, and 9/11), in which the actual assessment is "dry" despite higher levels of precipitation and lower levels of potential evaporation – perhaps the qualitative assessments are inconsistent in a few cases. Given the limited number of training examples, these handful of unusual examples can play a much more substantial role in the ultimate classification of a validation example.

KNN narrowly outperforms the boosted perceptron algorithm, failing to err even once outside of the confidence intervals constructed (Fig. 4.4). However, the boosted perception is much more likely to classify strongly, i.e. return a probability closer to 0% or 100%. This is because with KNN, where $k = 11$, given the limited training data, it is a virtual certainty that at least a couple of those

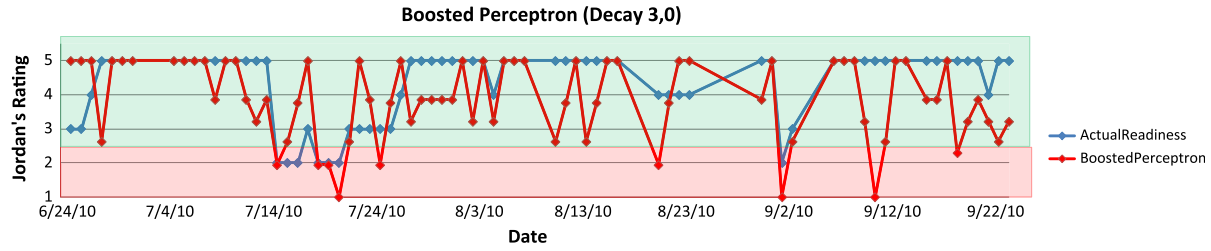

Fig. 4.5. (top) – Boosted Perceptron, actual (blue) vs. predicted (red) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

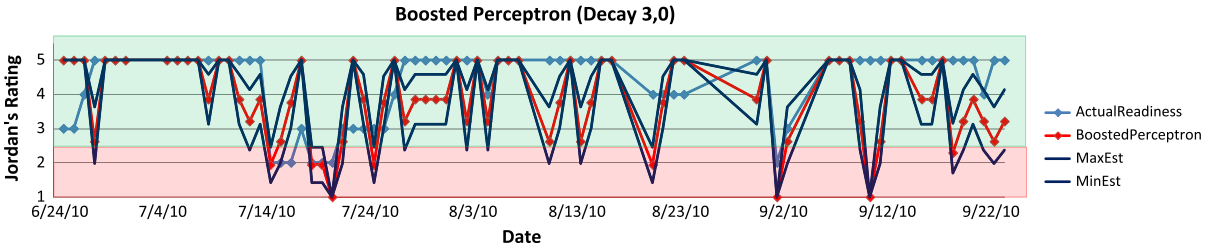

Fig. 4.6. (middle) – Boosted perceptron, actual (blue) vs. predicted (red), with binomial confidence interval (black) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

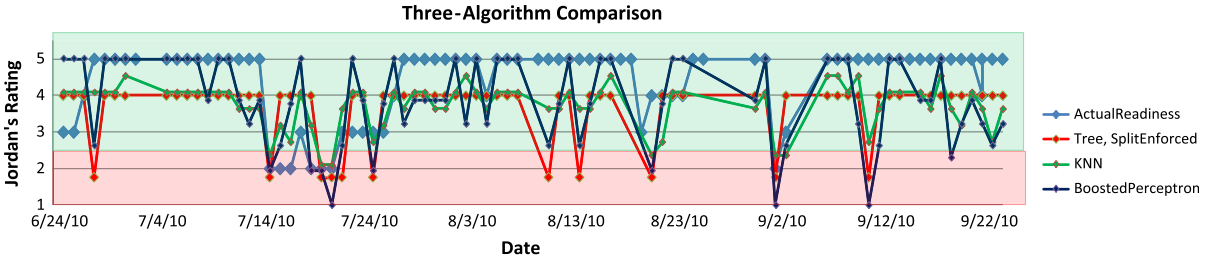

Fig. 4.7. (bottom) – Accuracy of the three algorithms (Tree: red, KNN: green, Perceptron: navy) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

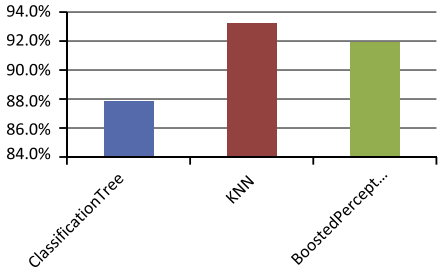

Fig. 4.8. Comparison of prediction agreement among the three algorithms.

similar examples deviate from the rest. This ensures that at most 9 or 10 examples will classify the same way, even before adding dummy votes. However, the boosted perceptron, by construction, develops hypotheses which, at times, return classifications at the extremes (and is 97% correct when it does). Future growing seasons with additional data at more sites will determine whether KNN or boosted perceptrons emerges as the best practice for predictions of this nature.

Ultimately, of the three algorithms examined, two represent viable and effective means of modeling the dryness of a specific site given user-defined assessments – KNN and boosted perceptrons. Both fall between 91% and 94% accuracy, even after removing those days for which classification would be easiest due to a lack of recent precipitation. Furthermore, the errors made by these two algorithms tend to align, illustrating that other factors may explain the erroneous results beyond simple model inadequacy. On three of the six days (7/17/10, 7/25/10, 9/19/10) in which the boosted perceptron algorithm erred, no time stamp was given for the qualitative measurement. One of these days (7/17/10) represented one of the five misclassifications for KNN as well. As a result, the algorithm selected a default time (9:00 AM, used whenever an actual time of day was not provided by the data gatherer) upon which to base predictions. As these measurements were varied in terms of their time of day from early morning to late afternoon, it is wholly plausible that the algorithm was correct at the time of day assumed, but ultimately incorrect given the unknown time at which the measurement was actually obtained.

Furthermore, several errors also occurred when the field conditions themselves were borderline cases between two classifications. In this case, minor qualitative inconsistencies in human assessment will lead to incorrect classifications.

Finally, the qualitative assessments lack any internal means of verification – as a result, it is possible that a data point is erroneously reported. In the case of 9/11/2010, substantial precipitation fell within the most recent 24-h period as well as from 24 to 48 h. Additionally, rates of potential evapotranspiration were below half of the growing season average over the 24 h preceding measurement. The classification tree and boosted perceptron classified the field as "wet" while KNN chose "dry." The qualitative conditions were reported as a "5" – i.e. a fully dried soil. This value is dubious, yet unverifiable retroactively, and resulted in errors for

two algorithms (as well as the one case in which the boosted perceptron predicted a 0% or 100% probability of dryness and was incorrect).

While boosted perceptrons currently do not match the performance of KNN, the level of accuracy is only marginally inferior. By using stochastic gradient descent (Russell and Norvig, 2010) to fit the parameters of each individual hypothesis, dimensionality issues create less of a concern. Of course, with each additional variable, the opportunity to over fit increases as well.

5. Discussion and conclusions

Despite occasional errors, which are uncommon and seem well explained by issues with data gathering, qualitative assessments, and missing information, two algorithms perform at 92% and 93% accuracy on validation data, and exceed even those figures when a confidence interval is introduced. It is possible that these statistics would improve as each successive growing season provides an increasingly rich data set, incorporating a greater variety of conditions and locations. Each algorithm needs further validation, but these results suggest that nowcasts of field readiness with high levels of accuracy are likely attainable.

Follow-on research suggests that the wetting/drying process can be modeled at one location with insights applied at other locations that are similar in terms of hydroclimate and soil types (Coopersmith et al., 2014). Additional research is needed to extend the approach to field readiness taken in this work to areas with less available data through such insights, which focused on soil moisture prediction.

Based on our findings to date, we expect that field readiness assessments will be more accurate in sandier soils where drainage occurs quickly and predictably with minimal overland flow and ponding, but perhaps more challenging in poorly-drained locations where longer precipitation histories would be required and mechanisms are more complex. The latter case, which does not occur at the field site used in this work, could require coupling the machine learning approach with more mechanistic models, as explored by Coopersmith et al. (2014) for soil moisture prediction. Furthermore, as alluded to previously, it is likely that different input variables might be required under different site conditions – not only longer precipitation histories, but potentially also parameters describing soil texture and vegetation coverage.

What is truly encouraging about these results is not only their accuracy, but the limited information requirements. Many agricultural plots in the Midwest and elsewhere are characterized by relatively efficient drainage from tile drains. In sites such as the test location in Urbana IL, USA, knowledge of rainfall over the most recent 48 h is wholly sufficient to model growing season wetting conditions. The API approach requires precipitation histories in excess of two weeks. The diagnostic soil moisture equation requires an indeterminate precipitation history that can be two to three months long. While this may ultimately provide a more precise estimate of soil moisture, in terms of tangible decision-support, the method presented in this paper offers sufficient performance at lower data costs. Moreover, both the diagnostic soil moisture equation and API approaches (and hydrologic models) pose difficulties in addressing precipitation that arrives as snow, settles in snow packs (not wetting the soil), then melts at a later date. When the precipitation history demanded is quite lengthy, a prediction during the planting season (e.g., early April at our test site) is likely to contain days of precipitation that have arrived as snow/ice and will disrupt results.

Furthermore, lengthier predictive windows may cause increasingly persistent errors. Consider a model that employs a 50-day precipitation history. If, on day X, the model is incorrect, on day $X+1$, the model is likely to be incorrect as well, as after all, 49 of the 50 days of precipitation data used for day X are still employed for day $X+1$. For this reason, an erroneous data point, a phenomenon whose impact is poorly addressed by the model, or simply a slight miscalibration, can endure for extended periods. If only two days of history are needed, a error on day X is likely to be resolved on day $X+1$, and is entirely irrelevant by day $X+2$.

Given the construction of these algorithms using free software (R) and computational requirements that are feasible on a conventional laptop, subsequent seasons and locations of data could be integrated without the need for structural changes to the software. Many hydrologic models for such wetting/drying analysis require frequent recalibration to avoid cumulative errors (Jones, 2004). However, by using individuals' personal standards of "wet" and "dry," these machine learning tools are naturally, and constantly recalibrating, adjusting to the judgment of the decision-maker. By asking decision-makers to provide feedback on the accuracy of the predictions (e.g. through a mobile application), each input augments the set of training data from which the algorithm can base its decisions. Errors do not propagate, but rather, are smoothed by increased knowledge.

Additionally, since these models are developed through machine learning protocols rather than physical soil mechanics, provided that a repository of consistently labeled training examples exist, such algorithms can be constructed for soil drying in many agricultural contexts. As the results indicate, public data yield relatively accurate predictions, implying that these algorithms can be constructed without the cost and logistical complexity of adding remotely-accessed sensory grids. Both the diagnostic soil moisture equation, API, and hydrologic approaches require soil moisture sensors to initially calibrate the model. In the case of machine learning models of qualitative assessments, the individual decision-maker is sufficient as a calibrating tool – and if their opinion of what "dry enough" entails changes, so too can the model.

The next step in this research involves creating a dashboard of "nowcasts" for soil conditions, viewable as a geospatial browser layer, allowing meaningful visualization of the spatial results. Additionally, future work is needed to investigate the inclusion of the components of potential evapotranspiration (which came from the Illinois State Water Survey) from national sources, in order to expand assessments beyond the state of Illinois. Where ICN data are unavailable, future work will be required to integrate public sources of temperature, wind, humidity, and solar radiation data to produce the potential evapotranspiration estimates that this model requires. It is worth noting that while precipitation data is becoming increasingly ubiquitous globally (especially in light of NASA's Global Precipitation Measurement Mission), potential evaporation estimates could be problematic outside the United States. However, remotely sensed solar radiation from satellites and other sources may ultimately resolve this issue.

Machine learning algorithms, being sufficiently flexible, should be able to adapt their parameters to any established, consistent, dryness metric. Such predictions could be worth investigating in a wide variety of circumstances (e.g., agricultural land use, confined animal feeding operations, and natural resource management). If constructed solely with public data, the number of site users who could conceivably benefit is substantial. Subsequent work has developed a U.S. hydro-climatic classification system (Coopersmith et al., 2012) that can be used to generalize our approach by allowing calibration at a single site to be reapplied anywhere of the same class. Preliminary results are encouraging and will be published in a future paper.

References

Aertsen, W., Kint, V., Van Orshoven, J., Muys, B., 2011. Evaluation of modeling techniques for forest site productivity prediction in contrasting ecoregions

using stochastic multicriteria acceptability analysis (SMAA). Environ. Modell. Software 26 (7), 929–937.
Blanchard, B.J., Mcfarland, M.J., Schmugge, M.J., Rhoades, E., 1981. Estimation of soil-moisture with API algorithms and microwave emission. Water Resour. Bull. 17, 767–774.
Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. Wadsworth International Group, Belmont, CA.
Breiman, L. et al., 1993. Classification and Regression Trees. Chapman & Hall, Boca Raton.
Brodley, C.E., Freidl, M.A., 1997. Decision tree classification of land cover from remotely sensed data. Remote Sens. Environ. 61 (3), 399–409.
Chico-Santamarta, L., Richards, T., Godwin, R.J., 2009. A laboratory study into the mobility of travelling irrigators in air dry, field capacity and saturated sandy soils. Am. Soc. Agric. Biol. Eng. Annu. Int. Meeting 4 (2009), 2629–2646.
Capehart, W.J., Carlson, T.N., 1994. Estimating near-surface soil moisture availability using a meteorologically driven soil water profile model. J. Hydrol. 160, 1–20.
Choudhury, B.J., Blanchard, B.J., 1983. Simulating soil water recession coefficients for agricultural watersheds. Water Resour. Bull. 19, 241–247.
Coopersmith, E.J., Minsker, B., Montagna, P., 2011. Understanding and forecasting hypoxia using machine learning algorithms. J. Hydroinformatics 13 (1), 64–80.
Coopersmith, E.J., Minsker, B., Sivapalan, M., 2014. Using hydro-climatic and edaphic similarity to enhance soil moisture prediction". Hydrol. Earth. Sys. Sci. Discuss. vol. 11, pp. 2321–2353. (doi:10.5194/hessd-11-2321-2014). <http://www.hydrol-earth-syst-sci-discuss.net/11/2321/2014/>.
Coopersmith, E., Yaeger, M., Ye, S., Cheng, L., and Sivapalan, M., 2012. Exploring the physical controls of regional patters of flow duration curves – Part 3: a catchment classification system based on regime curve indicators. Hydrol. Earth Syst. Sci. (doi:10.5194/hess-16-1-2012).
Entekhabi, D., Rodriguez-Iturbe, I., 1994. Analytical framework for the characterization of the space-time variability of soil moisture. Adv. Water Resour. 17, 35–45.
Farago, T., 1985. Soil moisture content: statistical estimation of its probability distribution. J. Clim. Appl. Meteorol. 24 (4), 371–376.
Fix, E., Hodges, J.L. Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci., 55.
Gamache, R.W., Kianirad, E., Alshawabkeh, A.N., 2009. An automatic portable near surface soil characterization system. Geotechnical Special Publication 192, 89–94.
Haywood, A., Stone, C., 2011. Mapping eucalypt forest susceptible to dieback associated with bell miners (Manorina melanophys) using laser scanning, SPOT 5, and ancillary topographical data. Ecol. Model. 222 (5), 1174–1184.
Hamon, R.W., 1963. Computation of direct runoff amounts from storm rainfall. International Association of Scientific Hydrology. Publication 63, Wallingford, Oxon., U.K.
Jensen, M. E., Burman, R. D., Allen, R.G., 1990. Evapotranspiration and irrigation water requirements. Manuals and Reports of Engineering Practice, No.70. New York.
Jones, H.G., 2004. Irrigation scheduling: advantages and pitfalls of plant-based methods. J. Exp. Bot. 55, 2427–2436.
Koppen W., 1936. Das Geographisca System der Klimate. In: 734 Koppen, W., Geiger, G., (Eds.), Handbuch der Klimatologie. 1. C. Gebr, Borntraeger, pp. 1–44.
Kumar, P., Alameda, J., Bajcsy, P., Folk, M., Markus, M., 2006. Hydroinformatics. CRC Press, Taylor & Francis Group, New York, USA.
Lailiberte, A.S., Fredrickson, E.L., Rango, A., 2007. Combining decision trees with hierarchical object-oriented image analysis for mapping arid rangelands. Photogr. Eng. Remote Sens. 73 (2), 197–207.
Lamande, M., Schjonning, P., 2008. The ability of agricultural tyres to distribute the wheel load at the soil-tyre interface. J. Terrramech. 45 (4), 109–120.
Lebert, M., Brunotte, J., Sommer, C., Boken, H., 2006. Protecting soil structure against compaction: proposed solutions to safeguard agricultural soils. J. Plant Nutr. Soil Sci. 169 (5), 633–641.
Li, H., Sun, J., Wu, J., 2010. Predicting business failure using classification and regression tree: an empirical comparison with popular classical statistical methods and top classification mining methods. Exp. Syst. Appl. 37, 5895–5904.
Li, Y., Jiao, Y., Reid, K., 2011. Assessment of landed and non-landed by-catch of walleye, yellow perch, and white perch from the commercial gillnet fisheries of Lake Erie, 1994–2007. J. Great Lakes Res. 37 (2), 325–334.
Lee, J.H., Wang, W., 2009. Characterization of snow cover using ground penetrating radar for vehicle trafficability – experiments and modeling. J. Terrramech. 46 (4), 189–202.
Meliker, J.R., AvRuskin, G.A., Slotnick, M.J., Goovaerts, P., Schottenfeld, D., Jacquez, G.M., Nriagu, J.O., 2008. Validity of spatial models of arsenic concentrations in private well water. Environ. Res. 106 (1), 42–50.
McRoberts, R., Tomppo, E., Finley, A., Heikkinen, J., 2007. Estimating areal means and variances of forest attributes using the k-nearest technique and satellite imagery. Remote Sens. Environ. 111 (4), 466–480.
Monteith, J.L., 1965. Evaporation and environment. In: Proceedings of the 19th symposium of the Society for Experimental Biology. Cambridge Universality Press, New York, pp. 205–233.
Mosaddeghi, M.R., Hajabbasi, M.A., Hemmat, A., Afyuni, M., 2000. Soil compactibility as affected by soil moisture content and farmyard manure in central Iran. Soil Tillage Res. 55 (1–2), 87–97.
Nemes, A., Roberts, R., Rawls, W., Pachepsky, Y., Van Genuchten, M., 2008. Software to estimate – 33–1500 kPa soil water retention using the non-parametric k-nearest-neighbor technique. Environ. Modell. Softw. 23 (2), 254–255.
Pan, F., 2012. Estimating daily surface soil moisture using a daily diagnostic soil moisture equation. J. Irrigation Drainage Eng. 138 (7), 625–631.
Pan, F., Peters-Lidard, C.D., 2008. On the relationship between the mean and variance of soil moisture fields. J. Am. Water Resour. Assoc. 44 (1), 235–242.
Pan, F., Peters-Lidard, C.D., Sale, M.J., 2003. An analytical method for predicting surface soil moisture from rainfall observations. Water Resour. Res. 39 (11) (Art).
Peel, C., Finlayson, B., McMahon, T., 2007. Updated world map of the Koppen-Geiger climate 756 classification. Hydrol. Earth Syst. Sci. 11, 1633–1644.
Pena-Barragan, J.M., Ngugi, M.K., Plant, R.E., Six, J., 2011. Object-based crop identification using multiple vegetation indices, textural features and crop phenology. Remote Sens. Environ. 115 (6), 1301–1316.
Priestley, C.H.B., Taylor, R.J., 1972. On the assessment of surface heat flux and evaporation using large-scale parameters. Mon. Weather Rev. 100, 81–92.
Pytka, J., 2009. Determining and analyzing the stress state under wheeled-vehicle loads. Proc. Inst. Mech. Eng. Part D: J. Automobile Eng. 223 (2), 233–253.
Quinlan, J.R., 1986. Induction of Decision Trees. Mach. Learn 1(1), 81–106.
Raper, R.L., 2005. Agricultural traffic impacts on soil. J. Terrramech. 42 (3–4), 259–280.
Russell, S., Norvig, R., 2010. Artificial Intelligence: A Modern Approach. Prentice Hall, Indianapolis.
Sahu, R.K., Raheman, H., 2008. A decision support system on matching and field performance prediction of tractor-implement system. Comput. Electr. Agric. 60 (1), 76–86.
Saxton, K.E., Lenz, A.T., 1967. Antecedent retention indexes predict soil moisture. J. Hydraul. Div. Proc. Am. Soc. Civ. Eng. 93, 223–241.
Shepard, D., 1968. A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM national conference, pp. 517–524 (doi: 10.1145/800186.810616).
Shoop, S., Kestler, M., Stark, J., Ryerson, C., Affleck, R., 2002. Rapid stabilization of thawing soils: Field experience and application. J. Terrramech. 39 (4), 181–194.
Sharifat, K., Kushwaha, R.L. 2000. Sinkage simulation model for vehicles on soft soil. ASAE Annual Intenational Meeting, Technical Papers: Engineering Solutions for a New Century 1, pp. 2549–2553.
Sliva, R.B., Lancas, K.P., Miranda, E.E.V., Silva, F.A.M., Baio, F.H.R., 2009. Estimation and evaluation of dynamic properties as indicators of changes on soil structure in sugarcane fields of Sao Paulo State – Brazil. Soil Tillage Res. 103 (2), 265–270.
Stavrakoudis, D.G., Theocharis, J.B., Zalidis, G.C., 2011. A boosted genetic fuzzy classifier for land cover classification of remote sensing imagery. ISPRS J. Photogram. Remote Sensing 66 (4), 529–544.
Thornthwaite, C.W., 1948. An approach toward a rational classification of climate. Geogr. Rev. 38, 55–94.
Tullberg, J.N., Yule, D.F., McGarry, D., 2007. Controlled traffic farming: from research to adoption in Australia. Soil Tillage Res. 97 (2), 272–281.
Wetzel, P.J., Chang, J.T., 1988. Evapotranspiration from nonuniform surfaces— A 1st approach for short-term numerical weather prediction. Mon. Weather Rev. 116, 600–621.