Thesis no: MECS-2015-19

Active learning via Transduction in Regression Forests

Kim Hansson

Erik Hörlin

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Engineering: Game and software engineering. The thesis is equivalent to 20 weeks of full-time studies.

Contact Information:
Authors:
Kim Hansson
E-mail: kiha10@student.bth.se

Erik Hörlin
E-mail: erhd10@student.bth.se

University advisors:
Dr. Anton Borg
Dept. Computer Science & Engineering
Prof. Niklas Lavesson
Dept. Computer Science & Engineering

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Context. The amount of training data required to build accurate models is a common problem in machine learning. Active learning is a technique that tries to reduce the amount of required training data by making active choices of which training data holds the greatest value.
Objectives. This thesis aims to design, implement and evaluate the Random Forests algorithm combined with active learning that is suitable for predictive tasks with real-value data outcomes where the amount of training data is small. machine learning algorithms traditionally requires large amounts of training data to create a general model, and training data is in many cases sparse and expensive or difficult to create.
Methods.The research methods used for this thesis is implementation and scientific experiment. An approach to active learning was implemented based on previous work for classification type problems. The approach uses the Mahalanobis distance to perform active learning via transduction. Evaluation was done using several data sets were the decrease in prediction error was measured over several iterations. The results of the evaluation was then analyzed using nonparametric statistical testing.
Results. The statistical analysis of the evaluation results failed to detect a difference between our approach and a non active learning approach, even though the proposed algorithm showed irregular performance. The evaluation of our tree-based traversal method, and the evaluation of the Mahalanobis distance for transduction both showed that these methods performed better than Euclidean distance and complete graph traversal.
Conclusions. We conclude that the proposed solution did not decrease the amount of required training data on a significant level. However, the approach has potential and future work could lead to a working active learning solution. Further work is needed on key areas of the implementation, such as the choice of instances for active learning through transduction uncertainty as well as choice of method for going from transduction model to induction model.

Keywords: Active learning, Regression, Random Forests, Semi-supervised learning, Transduction

List of Figures

List of Tables

List of Equations

Contents

Chapter 1

Introduction

Machine learning is the study of algorithms that improve their performance with experience (Mitchell, 1997). This improvement is referred to as training and represents the learning in machine learning. Machine learning programs are central to data mining, where they are used to automatically build predictive models and gain new knowledge by identifying patterns in large data sets (Fayyad et al., 1996). Data science is the study of the generalizable extraction of knowledge from data. It adopts theory and methods from for example: statistics, machine learning, data mining, artificial intelligence, databases, and optimization to solve real-world data centric problems (Dhar, 2013).

Many machine learning algorithms require large amounts of training data ranging from hundreds to thousands of samples (Settles, 2009). And the amount and quality of the training data have a large impact on the accuracy of the estimations and generalizations of the model (Demir and Bruzzone, 2014) (Flach, 2012). However, it is expensive to obtain the required amount of training data for many real-world problems (Flach, 2012).

Manually finding the correct outcome for a data sample is often impractical due to time and cost. For example, speech recognition requires linguistic experts (Zhu et al., 2005) and information extraction in biomedicine is done by PhD-level biologists (Settles, 2009). In other cases, data is related to real-world phenomena that are rare, for example: oil spills or volcanic eruptions (Witten et al., 2011).

Active learning algorithms are used to reduce the amount of required training data. Active learning allows the algorithms to submit data samples with an unknown outcome to an oracle which in turn will return the same sample but now with the outcome known. An oracle can be represented by a human expert or an intelligent system. Using some heuristic, the algorithms choose the data that provides the highest training value and thus decreases the amount of training required.

This thesis is based on the work of Criminisi et al. (2011) who present an active learning approach for Random Forests (Breiman, 2001). This approach divides similar data samples into clusters based on a chosen criterion. These clusters can contain data samples with known outcomes as well as data samples with unknown outcomes. It then assigns temporary outcome values to data with unknown outcomes based on their similarity to the data with already known outcomes. The uncertainty of these temporary outcomes can then be used for making an active choice of which data to send the oracle. The work of Criminisi et al. only cover classification problems while the aim of this thesis is to find a solution for prediction problems, the main difference being in how temporary outcomes can be assigned to continuous numeric predictions as opposed to discrete classifications.

Chapter 2

Background

Typically, a machine learning algorithm solves a problem by training on a set of data. A single sample from a data set is referred to as an instance, and contains a number of input attributes as well as zero or more outcomes which defines the task. If the desired outcome of an instance is known, it is said to be labeled, otherwise it is referred to as unlabeled. Depending on the type of training data that is used, learning problems are divided into three common categories; supervised, unsupervised and semi-supervised (Flach, 2012).

Supervised learning requires labeled data and is used to solve machine learning tasks such as classification and regression (Flach, 2012). In supervised learning, the algorithm uses the data and attempts to generate the correct outcome, changing algorithm specific parameters until the desired outcome has been achieved. The difference between classification and regression is that for classification the outcome is a discrete number of classes. The model needs to assign the data to one or more of these classes. For regression the outcome is continuous and thus the model has to be able to predict continuous data.

Unsupervised learning problems use unlabeled data to solve clustering tasks. Here, since the outcome is not known, the algorithm instead needs to produce a model capable of grouping together data by some definition of similarity (Flach, 2012).

Learning problems that try to combine the usage of labeled and unlabeled training data are called semi-supervised learning (Zhu et al., 2005), active learning is a subgroup to semi-supervised learning (Flach, 2012).

The outcome of a machine learning algorithm is a model that is used to solve the task in accordance with the terminology above. It is important to be aware of the difference between algorithms and models, tasks are addressed by models, learning problems are solved by learning algorithms that produce models (Flach, 2012).

The fundamental goal of machine learning is to generalize beyond the training data (Domingos, 2012). The idea is to construct a model using training data, and then use that model on new data it has not seen before. This means that the machine learning algorithm must build a model that not only recognizes the structure found in the training data, but also the general structure of all data that the training set represents. The problem of a model becoming to closely fitted to the training data and thus not usable for a general task is referred to as overfitting (Flach, 2012).

2.1 Ensemble Learning

Ensemble learning is a machine learning approach that combines several models, all produced to solve the same task. The theory is that the combined prediction of the ensemble is often better than the prediction of a single model (Dietterich, 1997) (Rokach, 2009).

An important aspect of ensemble learning is that each model in the ensemble is constructed different so that the models in the ensemble do not give the same prediction each time, this can be done by for example bootstrapping or boosting. Bootstrapping means training each ensemble member with slightly different training data. Boosting on the other hand builds the ensemble based on the mismatches of previous iterations.

Random Forests is an ensemble of multiple random decision trees introduced by Breiman (2001). Apart from the randomness of attribute selection native to random trees, the Random Forests can also add further randomness by using bootstrapped data (Denil et al., 2014). This algorithm makes it possible to take a complex nonlinear prediction task and split it into smaller tasks that are handled by simpler models (Criminisi et al., 2011).

Using the mean value of a large amount of trees aids in negating the issue with random trees becoming overfitted to the training data (Flach, 2012).

When creating a forest there are in most cases three variables that needs to be defined, the number of trees, the maximum depth and how many attributes to be randomly selected for each tree.

Higher amounts of trees in a forest improve the accuracy while increasing computational time, so when choosing the amount one should take into consideration how valuable the performance increase is compared to computational time. The maximum depth variable enforces how deep a tree is allowed to split nodes, although a tree can decide to stop splitting before the maximum depth is reached if the split would not improve the model. The maximum depth directly affects the maximum number of leaf nodes in a tree:

$$M_L = 2^{M_d} \tag{2.1}$$

Where M_L is the maximum number of leaf nodes and M_d is the maximum depth. Shotton et al. (2011) shows that the accuracy of a tree increases with the depth up to a certain point, which depends on the size of the training data. After that point a tree begins to shows signs of overfitting and the accuracy begins to decrease when a higher depth is used. Shotton et al. notes that up until depth 10 the tree avoids overfitting regardless of size of the data set, which implies that depth 10 can be useful when conducting tests on data sets with different sizes. Attributes are randomly selected to increase randomness in the trees which has been shown to increase accuracy (Breiman, 2001). The number of attributes to select can be calculated dynamically as shown by Breiman (2001):

$$F = int(log_2 M + 1) \tag{2.2}$$

Where F is the number of attributes to be selected and M is the total number of attributes in the data set. In this context $int()$ means the first integer lower or equal to the input.

2.2 Transductive Learning

Transductive learning is a training setting that assigns labels for unlabeled instances by using the known labels in that sample (Goldberg, 2009). This approach can be useful

in semi-supervised learning problems, since they often work with more unlabeled than labeled training data. The method stands in contrast to inductive learning, which uses the training data to define a function that can predict future unseen instances.

The difference between induction and transduction has been compared to an in-class exam and a take home exam. Induction being similar to an in-class exam since the questions are unknown, so a student should prepare for all possible questions. Transductive learning is in contrast more similar to a take-home exam, where the student knows all questions and only needs to prepare for those (Goldberg, 2009).

2.3 Statistical testing

Statistics is a mathematical field that involves the summary and analysis of data (Sheskin, 2007). Statistics is usually divided into two branches, descriptive statistics and inferential statistics. Descriptive statistics consists of methods and procedures for presenting and summarizing data. The procedures most commonly used in descriptive statistics are the use of tables and graphs, and the computation of measures of central tendency and variability. Inferential statistics employs data to draw conclusions or make predictions. Typically, inferential statistics uses sample data from a population to draw conclusions that are valid for the whole population. A sample is a set of subjects or objects while a population is the total sum of subjects or objects that have something in common with one another.

When conducting an inferential statistical test with one or more samples drawn from one or more populations, the test may make certain assumptions about the underlying population distribution. The most commonly encountered assumption in this regard is that a distribution is normal (Sheskin, 2007). Figure 2.1 shows the shape of the normal distribution, or Gaussian distribution as it is also known. The closer a score is to the mean of the distribution, the more frequently it occurs. Knowing if a population is normally distributed is important when choosing the type of statistical test to use for evaluation. This will be covered in greater detail in section 5.3.

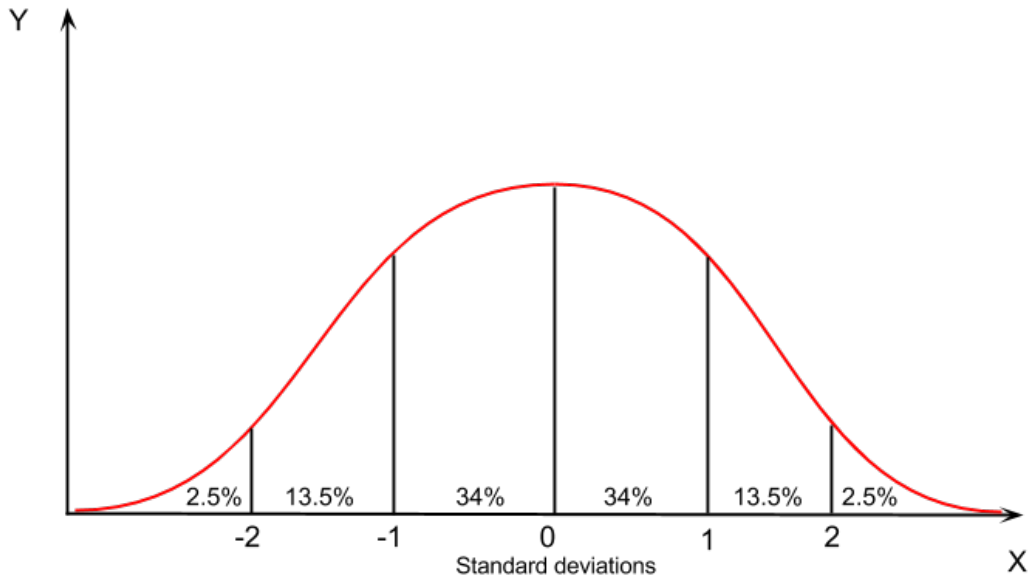

Figure 2.1: Example image of the normal distribution shape. 0 marks the distribution mean

Another important aspect of inferential statistics is the hypothesis. A hypothesis can be defined as a prediction about a single population or about the relationship between two or more populations (Sheskin, 2007). To evaluate a hypothesis, it is generally

restated as two separate hypothesis, the null hypothesis and the alternative hypothesis. The null hypothesis is a statement of no difference and is generally a hypothesis that the researcher expects to be rejected. The alternative hypothesis represents a statistical statement indicating the presence of an effect or a difference. The alternative hypothesis is one that the researcher in general expects to be supported (Sheskin, 2007).

The alternative hypothesis can be either non-directional or directional depending on how it is stated. A non-directional hypothesis does not make a prediction in a specific direction, it only states that there will be a difference. A directional hypothesis predicts a specific direction, meaning it predicts a differences that is greater or lower than the null hypothesis.

To reject or support a hypothesis, data is evaluated using a specified inferential statistical test. The test yields a test statistic and the value of this statistic is interpreted by employing special tables which contain critical values for that test statistic. These values are highly unlikely to occur if the null hypothesis is true. These tables allow the researcher to determine whether or not the result of a study is statistically significant (Sheskin, 2007). Statistical significance implies that one is determining whether or not an obtained difference in an experiment is likely to be due to chance or to the presence of a genuine experimental effect (Sheskin, 2007).

2.4 Related Work

There has been some work on combining active learning with Random Forests, but to our knowledge no work has been done on combining active learning with Random Forests for regression tasks. We want to contribute to active learning research by combining active learning using transduction for Random Forests.

Burbidge et al. (2007) presents an active learning solution for regression using Query-by-committee, but this solution lacks a random element, instead they achieve the disagreement needed for query-by-committee by using different subsets of data for different ensemble members. In comparison, Random Forests uses random subsets to achieve less correlation between committee members.

Su et al. (2009) presents work conducted on combining active learning with Random Forests but focuses only on classification problems. The work is still relevant because the differences between classification trees and regression trees concerns how the trees are constructed, not how the uncertainty of the given result is measured, which is the value used to decide what needs to be labeled. It also does not only label new data using an oracle, but also uses "soft labeling". Soft labeling is a technique where data can belong to several labels with a different amount of certainty. Applying soft labeling on regression could be seen as using a weighted mean for the final regression label, this concept will be further explained in section 4.2.

Basit and Wechsler (2013) works with combining active learning and transduction with association mining to explain and predict nsSNP-induced pathology. Here the transduction is used to generate a more consistent test error. We would instead like to use the transduction as a base for active learning.

Maiora et al. (2014) combines active learning with Random Forests for classification. The approach adds new training data over iterations to minimize the amount or required training data. Their method of using the disagreement or the Random Forests ensemble to choose the training data to be labeled is similar in that it is based on uncertainty of the classification, but does not use transduction.

Demir and Bruzzone (2014) presents an active learning training solution for regression problems solved with Support Vector Machines which focuses on selecting new data for labeling. This is done by clustering the data together and labeling data from

clusters based on three criteria; relevance, diversity and density. This is similar to the method proposed in this work in that the active learning solution is based around clustering, however our work will form clusters based on the node splits found in tree structures and use transduction to assign new labels. With this, labeling of new data is based on the uncertainty of the transduction instead of the cluster criteria.

From the articles reviewed above, we see that there is a research gap when it comes to implementing and evaluating an approach to active learning for regression forests. The differences when working with regression compared to classification, as well as between Random Forests and alternative algorithms such as Support Vector Machines make this research relevant.

Chapter 3

Aim and Objectives

This thesis aims to extend earlier work in machine learning by proposing a new method for building predictive models in data science scenarios where target variables to predict are real-valued and the amount of examples are scarce.

The objectives of this work can be divided into two stages; implementation and experimentation. In the implementation stage we reconstruct the method described by Criminisi et al. (2011), but with modifications for regression type problems. In the experimentation stage we evaluate the effect of our algorithms over several data sets and compare it to a supervised Random Forests algorithm. We also evaluate the efficiency of the Mahalanobis distance for transduction when compared to using euclidean distance. Using the Mahalanobis distance for transduction is proposed method by Criminisi et al. and will be further explained in chapter 4. We define the following research questions for these objectives:

RQ 1 *Is there a significant difference in the amount of required training data when using our active learning approach compared to random sampling?*

Random Forests is an algorithm that traditionally requires large amounts of training data. However since constructing training data is expensive it is relevant to research if active learning can be used to reduce the amount of required training data. This question is analyzed to see whether our method is a valid approach to active learning for regression forests.

This is tested by measuring the difference of our approach compared to a supervised Random forests algorithm. Our algorithm will make active choices of which data the receive as labeled while the supervised algorithm will be assigned new labeled data selected at random. The mean absolute error of their predictions are then measured over several iterations. This method of measuring the effectiveness of active learning is similar to the method described by Demir and Bruzzone (2014).

RQ 2 *Is there a significant difference between using the Mahalanobis distance for transduction compared to euclidean distance?*

Criminisi et al. (2011) uses the Mahalanobis distance to perform transduction. It is however a computational expensive algorithm because of the need to do inverse matrix calculations. This leads to long training times and does not scale well when the amount of data instances or instance attributes grow. Criminisi et al. also suggests that other distances could be used. Euclidean distance could provide an alternative to reduce the training time of the algorithm.

By measuring the transduction error rate on both approaches over the same data sets we can test whether the expensive calculations of the Mahalanobis distance can be justified.

The expected outcome of the two research questions can be formalized into the two following statements:

1. *Active learning can reduce the amount of required training data for Random Forests solving regression tasks*

2. *Mahalanobis distance gives a lower transduction error than using euclidean distance*

With this our main contribution will be the implementation and evaluation of a active learning approach for Regression Forests as well as an evaluation of the efficiency of the transduction method proposed by Criminisi et al. (2011).

Chapter 4

Approach

This chapter will cover the details of how our algorithms works over the different learning stages, ending with a summary of the changes implemented compared to the original classification method proposed by Criminisi et al. (2011).

4.1 Semi-supervised decision forest

The first step of our approach is to construct a semi-supervised forest combining regression and density clustering which will be the base for the transductive learning. The forest is constructed one tree at a time by maximizing the information gain of each node split, recursively working through each node and stopping once the information gain of a split is lower than the gain of keeping the node intact. More detailed equations describing this procedure can be found in appendix A.

Before the information gain can be calculated for a node, the algorithm needs to find the best attribute to perform a split on. Our solution will go through all the available instances that arrived at the node and test their value on the attribute as a split. The value of a split is calculated in a similar matter as gain, with the main difference that the objective is not to maximize gain but instead to minimize the regression and cluster entropy of the two subsets from the split. Algorithm 1 shows pseudo code for finding the best split.

Algorithm 1 Find best split

```
1: Input : trainingData
2: Output : splitAttribute, splitValue
3: for each attribute attr in data do
4:    for each instance inst with attribute attr do
5:       currSplit ← value of attr in inst
6:       subsets ← splitData(data, currSplit)
7:       regressionGain ← calculateRegGain(subsets)
8:       densityGain ← calculateDensGain(subsets)
9:       if (regressionGain + (α * densityGain)) < currentBest then
10:         currentBest ← (regressionGain + densityGain)
11:         splitValue ← (currSplit + prevSplit) / 2
12:         splitAttribute ← attr
13:         prevSplit ← currSplit
14:      end if
15:   end for
16: end for
17:
```

The α variable in algorithm 1 is the same as defined by Criminisi et al. In their work, it is not specified how the alpha value should be calculated, only that it is user

defined. For this thesis we did empirical testing to find an optimal alpha value. The results of these tests are presented in chapter 6.

Once a tree have been constructed, each leaf will hold the mean regression value of the instances as well as the covariance matrix constructed from the combined labeled and unlabeled data that reached the leaf. The mean regression value is used to predict regression values for new data instances, while the matrix is used to perform the transduction described in the next section. Algorithm 2 shows pseudo-code for the recursive function used in the construction of a tree.

Algorithm 2 Tree construction

```
1: Input : trainingData
2: if trainingData > 0 then
3:    totalInformationGain ← calculateRegGain(trainingData) + calculateDens-
      Gain(trainingData)
4:    splitPoint ← FindBestSplit(trainingData) /*Use algorithm 1 to find best split
      */
5:    subSets ← splitData(data, currSplit)
6:    splitInformationGain ← calculateRegGain(subsets) + calculateDens-
      Gain(subsets)
7:    if totalInformationGain - splitInformationGain > 0 then
8:       constructChildren(subsets) /*create child nodes and send them parts of the
         training data */
9:    end if
10: else
11:    designateLeaf() /*if there was no gain from splitting, make this node a leaf node
       */
12: end if
```

4.2 Transduction

As mentioned in section 2.2, transduction can be used to assign labels to unlabeled instances by using known labels. Criminisi et al. (2011) applies transduction by computing a geodesic path from an unlabeled instance to all labeled instances, assigning it the label of the closest instance. The distance of a geodesic path is calculated by computing the sum of the Mahalanobis distances between the points in the path.

The Mahalanobis distance is a measure of dissimilarity between two points. It is calculated by using the inverse of the covariance matrix from their cluster. The exact equations used by Criminisi et al. to calculate the geodesic path and Mahalanobis distance can be found in appendix A. It should be noted that the Mahalanobis distance is only defined for points in the same cluster. However the Mahalanobis equation described by Criminisi et al. calculates it over two separate cluster. This is done by calculating the normal Mahalanobis distance for both covariance matrices and then using the mean of the two values. Criminisi et al. does not explain how this cluster spanning version of the equation was derived and gives no proof of its concept. But since this equation seems to work in practice, no alterations have been made to the equation for our implementation.

In this thesis the geodesic path is interpreted as the shortest distance between two points in a complete graph, and therefore transduction will be applied using graph theory. Dijkstra's algorithm is used for shortest path detection, which for complete

graph is computationally heavy as can be seen by its big O notation:

$$O\left(\frac{V*(V-1)}{2}+V*log(V)\right) \tag{4.1}$$

Where V is the number of nodes in a graph. It should be noted that the algorithm is performed for each labeled instance in a graph. To reduce the computational time of the shortest path calculations, a tree based graph is used instead of a complete graph. This means that the graph will be split into sub-graphs using the structure of the decision tree that created the graph. Each leaf from the decision tree will be represented by a graph containing the instances that reached that leaf.

If a leaf does not contain any labeled data the tree structure will be traversed in reverse, merging together sub-graphs until a labeled instance is found. When merging, a new graph will be created once more using Mahalanobis distance as edge length.

As earlier mentioned, Criminisi et al. assigns an unlabeled instances with the same label as the closest labeled instance. This works in classification problems since classification data have a discrete number of labels, meaning that they often span areas in the data, while regression data has continuous labels (Flach, 2012). Therefore we propose a different equation than Criminisi et al. for applying transduction, instead using the labels from all labeled instances in the local graph and applying a weighted mean:

$$p^u = \sum_{i=0}^{n} \frac{w_i P_i^l}{n-1} \tag{4.2}$$

Where p^u is an unlabeled point, P^l is a collection of all labeled points and w_i is the weight for the labeled point P_i^l.

The weights are calculated as the shortest distance between each label and the unlabeled point, divided by the sum of all shortest distances to labeled points:

$$w_i = 1 - \frac{d_i}{D} \tag{4.3}$$

Where d_i is the shortest distance between the unlabeled point and the label calculated by using the geodesic-path equation described by Criminisi et al.[1], and D is the sum of all shortest distances to the labeled points. This means that the weight between points is a percentage describing how much each point affects the final label for the unlabeled point. Thus the closer a labeled point is, the more it will affect the final label.

This solution is similar to that of Troyanskaya et al. (2001), in which their KNNimpute algorithm uses a weight based on similarity to decide the contribution of each instance to the missing value.

4.3 Induction

Once new labels have been transducted onto the unlabeled training instances the goal is to use these temporary labels to create an induction model. This will enable the model to make prediction on data it has not seen before.

To achieve induction, the tree is traversed with all instances now treated as fully labeled. Once arriving at a leaf the instances are used to update the regression value of the leaf.

[1]The complete equation can be found in appendix A as equation A.8

When predicting regression values for new data, the model will traverse the tree using the attributes of the new data point. Once a leaf has been reached, the tree will make a prediction using the stored regression value of that leaf. The final prediction of the whole forest will be the mean of all the tree predictions.

4.4 Active learning

While conducting active learning the forest will select a number of unlabeled instances and send them to an oracle for labeling. The data selected for labeling can be selected at random or by using a heuristic, for example the instance(s) furthest away from a label. After the data has been returned labeled, the forest will be rebuilt with the new labeled data. This can continue until a target accuracy has been reached or a set number of instances has been labeled by the oracle.

For our approach, each tree will vote for the instance that had the longest shortest distance to a labeled instance in that tree during the transduction. the instances with the most votes are sent to the oracle. The number of instances to be sent is user defined and can be any number between one and the number of trees.

4.5 Summary of algorithm modifications

Criminisi et al. (2011) assigns new labels via transduction by using the closest label to an unlabeled instance. Our solution instead assigns a new label by calculating a weighted mean of all available labels.

To find the shortest distance to a label, Criminisi et al. checks against the distance to all labels. In our solution we instead check against the local cluster first and then gradually checks more clusters by reverse tree-traversal until at least one label is found.

Our first approach tried to reproduce the regression entropy calculation described by Criminisi et al. and covered in appendix A. It also used their prediction calculation approach which has an optimal line function stored in each leaf to predict new values. However, testing failed to produce stable results and we concluded that our implementation attempt had failed to reproduce these concepts. Instead, we calculate the regression entropy for information gain using the variance of the labeled instances as the base for the regression entropy. The main drawback with our approach is that it cannot handle multivariate regression which would have been possible with the calculations proposed by Criminisi et al. And as an alternative to using the optimal line function, we use the mean regression value of all instances that reach a leaf node.

For an active learning scenario, Criminisi et al. suggests that the built in probabilistic uncertainty of the regression formula could be used to select instances for labeling. We first calculate the shortest paths between each unlabeled and labeled point, and then use the distance of the longest of these paths as uncertainty measurement for new label selection.

Chapter 5

Method

This section describes the implementation aspect of our solution as well as the experimental design and statistical analysis description of the thesis.

5.1 Implementation

For our implementation we chose to use the Weka machine learning library for java (Witten et al., 2011) and the Eclipse Juno IDE as the base from which to build and make improvements. Weka provides a stable open source framework with good basic implementations of Random Forests as well as built in support for covariance and matrix calculations.

5.2 Experimental Design

To get a general result of the solution performance, we used a number of different data sets from the UCI data base (Lichman, 2013), shown in table 5.1. The UCI data base is a commonly used source for data sets in machine learning research (Demšar, 2006)(Demir and Bruzzone, 2014)(Johansson et al., 2014). Because of limited testing time and the long training time of the algorithm, we had to limit testing to data sets with a size limit of 12000. The data sets were chosen to represent a wide variety of fields to give the experiment a higher external validity.

Table 5.1: Data sets

id	size[a]	number of instances[b]	number of attributes[c]
d_1	2464	308	8
d_2	4509	751	6
d_3	5877	489	12
d_4	6721	517	13
d_5	6912	768	9
d_6	7084	506	14
d_7	7343	293	25
d_8	9270	1030	9
d_9	11512	959	12

Note. $d_1 - d_9$, Yacht Hydrodynamics, Airfoil self-noise, winequality-white, Forest Fires, Energy Efficiency, Housing, Parkinsons Telemonitoring, Concrete compressive strength, winequality-red
$^a = {}^b * {}^c$

The performance of the algorithm was measured using mean absolute error(MAE):

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|F_t - A_t| \tag{5.1}$$

Where F_t is the predicted value and A_t is the actual value. MAE was chosen since it is the standard error measurement for Random Forests in Weka and has been used in previous regression research (Demir and Bruzzone, 2014).

We use an independent samples design (Sheskin, 2007) where subjects are selected at random from each data set and assigned to k separate groups, called folds, used for training and validation. This method is called k-fold cross-validation and is used to avoid external validation threats such as over-fitting the model and training bias. Each fold is used once as validation data and k-1 for training. The mean results over all folds are then used as the general result of a test (Flach, 2012). Once the data set has been divided into a training set and a validation set, the training set is again divided into one set treated as labeled data and one treated as unlabeled data. To give a more visible error difference we start each test with a small amount of labeled data, 5% of the original training set chosen at random. The other 95% has their true label hidden to allow the algorithm to use them as true unlabeled data. This also allows us to simulate the effect of an oracle. When an instance has been selected to be labeled by an oracle, we reveal the label for that instance again. The instance is then added to the labeled set and removed from the unlabeled set.

As a baseline algorithm to test against, we used a supervised regression forest algorithm that share the same regression information gain calculation and prediction calculation as our semi-supervised approach. This increases internal validity since it narrows down the internal variables that can affect the experiment outcomes to variables related to the unsupervised aspects of the approach.

5.3 Statistical Analysis

To answer the research question stated in chapter 3, we evaluate the measured results over each data sets using statistical testing. Statistical tests can be divided into two categories, nonparametric and parametric. The difference between the two is that a parametric test assumes that the sample is normally distributed over the population whereas a nonparametric test makes no such assumption. The drawback with a nonparametric test is that it requires the data to be in rank-order format. If it is not, it has to be transformed into such a format, which means sacrificing information (Sheskin, 2007).

The population in this case refers to all possible tasks while a sample is a collection of data sets, each describing a separate task. To guarantee sample normality over a population, at least 30 separate data sets are needed (Demšar, 2006). This means that since we only conduct our tests over 9 different data sets, a parametric test cannot be used.

For comparing two different algorithms using nonparametric testing, Demšar (2006) recommends the Wilcoxon rank-sum test (Wilcoxon, 1946). The Wilcoxon tests answers the null hypothesis that there is no difference in performance of the two algorithms. If this hypothesis is rejected it is possible to formulate an alternative directional hypothesis to answer which algorithms has the better score. Using equations, the Wilcoxon test can be formulated in the following way: Let c_i^1 and c_i^2 be the performance score of two separate algorithms over the i-th data set. The difference between the two scores, $d_i = c_i^1 - c_i^2$, is calculated for each data set and ranks are assigned according to the absolute value of d_i. Let R^+ be the sum of ranks where algorithm 1 outperforms algorithm 2 and R^- be the sum of the opposite. Ranks of $d_i = 0$ are split evenly among the sums. We then set $T = min(R^+, R^-)$ and use a statistical table to find a critical value for comparison as described in section 2.3. The null hypothesis is rejected if T is equal to or less than the critical value.

Chapter 6

Results

This chapter will cover the results of the evaluation. First the alpha value tests are covered followed by the evaluation of the active learning performance. The chapter ends with the evaluation of the tree-based traversal and the Mahalanobis distance.

Figure 6.1 and 6.2 shows the most promising results of our alpha value analysis. The evaluation was conducted on data set d_5. d_5 was chosen because it represents a middle ground with regards to size and we could not, because of time constrains run the alpha tests over all data sets. Figure 6.1 shows the results of the initial alpha test. Here we did 10 active learning iterations over a wide selection of alpha values. From these results we selected the three most promising candidates; 1.0, 1.5 and the weight equation for further testing.

The weight equation works by recalculating the alpha value for each individual node in the tree in the following way:

$$\alpha = \frac{number\ of\ unlabeled\ instances}{total\ number\ of\ instances} \tag{6.1}$$

The idea being that each node should have an individual alpha value representing the weight between unlabeled/labeled instances reaching the node.

Figure 6.2 shows the results of analyzing alpha values centered around the three promising candidates over 20 active learning iterations. From these results we concluded that the most stable alpha value was 1.2. All subsequent tests were conducted using this alpha value.

For our Random Forests variables described in section 2.1 we choose to use a max depth of 10 since it has been shown to be a good choice when performing tests on a number of different data sets. The forest is constructed with 100 trees, which is the Weka standard amount. For the feature selection we went with the equation 2.1 described in section 2.1.

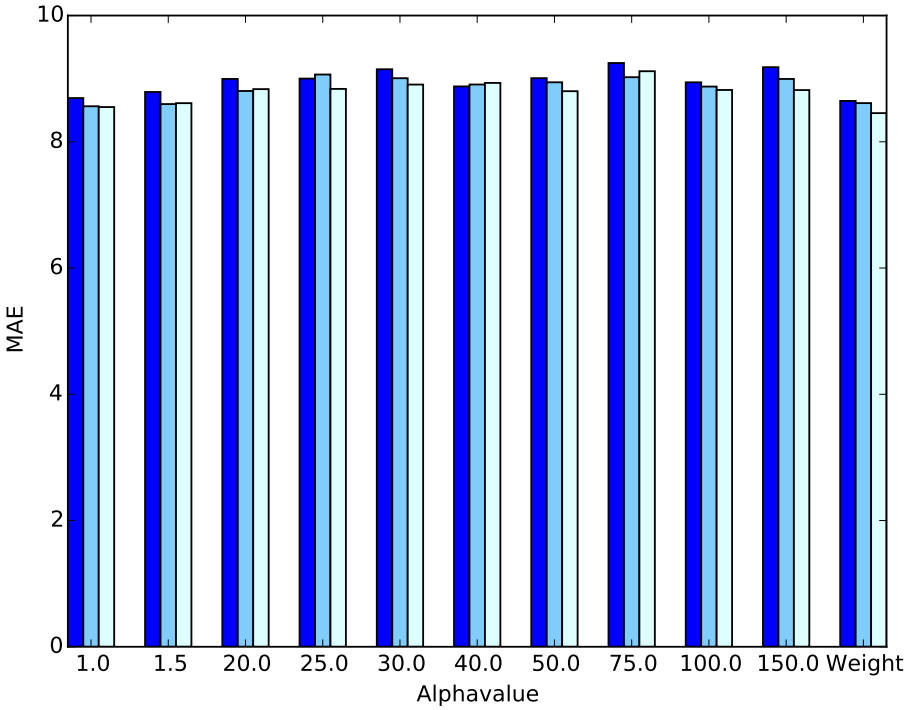

Figure 6.1: Initial comparison of different alpha values. Dark: 0 iterations, Medium: 5 iterations, Light: 10 iterations

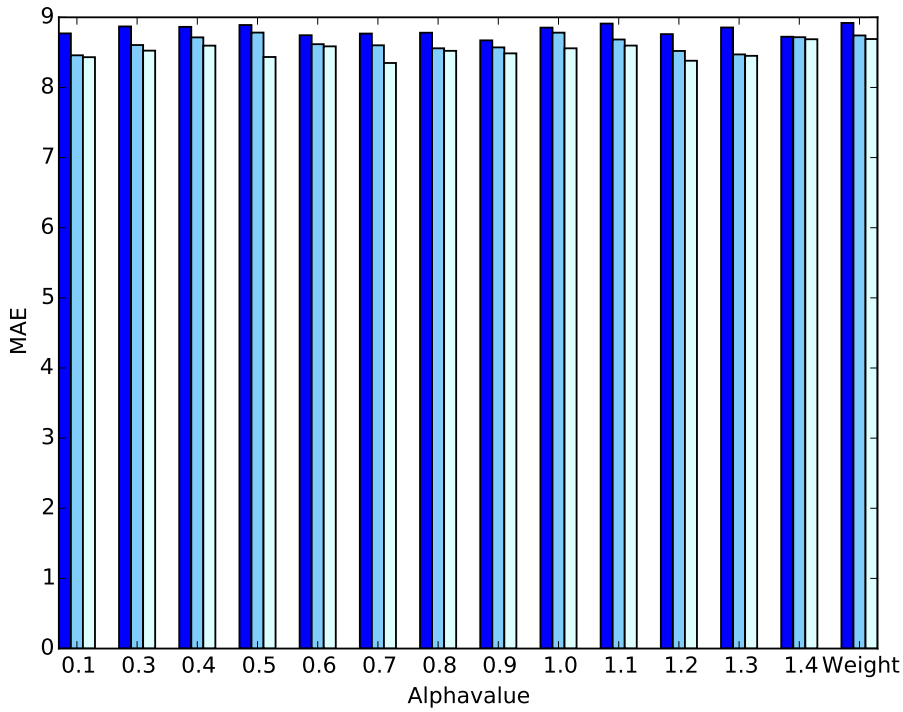

Figure 6.2: Comparison over the most promising alpha values. Dark: 0 iterations, Medium: 10 iterations, Light: 20 iterations

Table 6.1: Active learning versus Random Selection

Algorithm	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Active choice	14.563	7.3741	0.7032	15.9036	9.1999	6.6977	11.3921	14.4568	0.7002
Random selection	11.6775	6.7887	0.6666	18.2467	9.1704	6.6864	11.2774	13.2335	0.7011

The final MAE value after 50 iterations

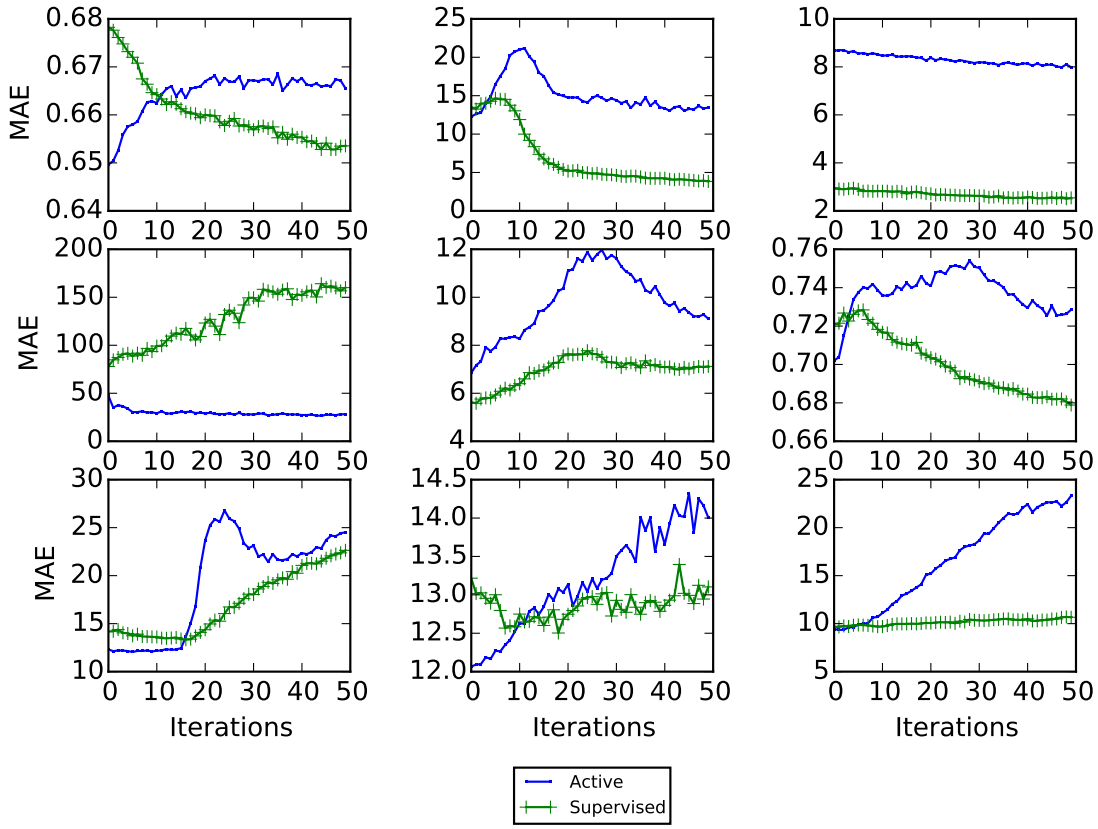

Figure 6.3: Comparison of mean absolute error of supervised and active learning. The plots corresponds to the following data sets, moving left to right: d_9, d_1, d_5, d_4, d_6, d_3, d_7, d_8, d_2

Figure 6.3 show the results from the tests comparing the MAE of active learning with randomly selecting new labels. Both algorithms use the regression entropy and regression prediction calculations as described by Criminisi et al. It should be noted that in figure 6.3 and 6.4 the Y-axes are not the same for all data sets, the reason for this is that each plot is meant to show the difference between the active and supervised version for each data set and the mean absolute error differs between the data sets. Most of the graphs in figure 6.3 are non-monotonic, this means that the performance of the algorithms are not stable even when averaged over 10 tests using ten-fold cross-validation. Furthermore even the baseline algorithm gets increasingly worse in performance as the amount of labeled data increases. From this we conclude that we failed to reproduce the mathematics of the regression calculations of Criminisi et al..

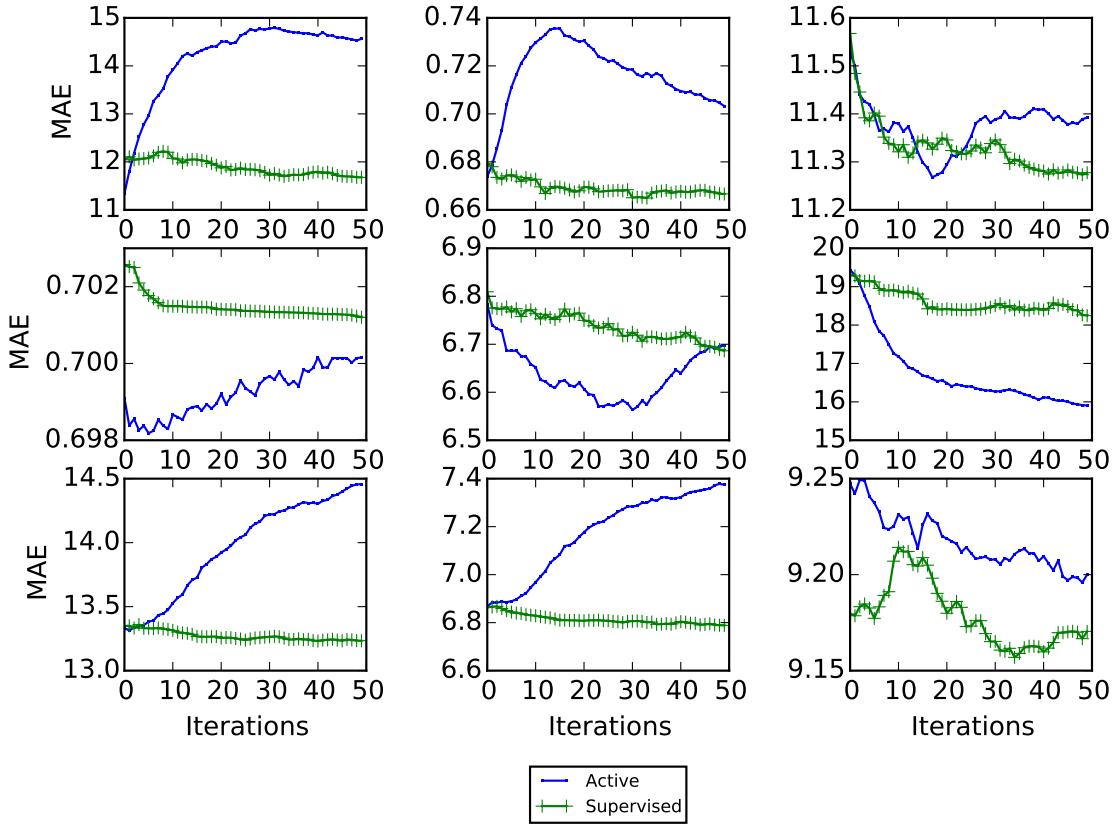

Figure 6.4: Comparison of mean absolute error of supervised and active learning. The plots corresponds to the following data sets, moving left to right: d_1, d_3, d_7, d_9, d_6, d_4, d_8, d_2, d_5

As mentioned in section 4.5, we did alterations to the regression calculations after reviewing the results from figure 6.3. The results of the second approach are shown in figure 6.4. Here, the baseline algorithm improves over all data sets, if only slightly. However, our approach still suffers from irregularities such as non-monotonic behavior and an increase in prediction error as the amount of labeled data increases.

Table 6.2: Mahalanobis distance versus Euclidean distance

Algorithm	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Mahalanobis	0.4261	0.0258	0.0587	2.6275	0.2076	0.1339	0.3493	0.1949	0.0630
Euclidean	0.6955	0.0509	0.1182	0.7421	0.5213	0.3705	4.4223	0.5949	0.1265

Mean transduction error comparison between Mahalanobis distance and Euclidean distance

Table 6.2 shows the results of comparing the Mahalanobis distance for transduction with euclidean distance. It should be noted that the Mahalanobis distance use our tree-based traversal method. However the euclidean distance calculations require no graph traversal since it is calculated straight between points. Both cases were given 20% labeled instances.

One change that was implemented to give slightly faster training time was to utilize the tree structure and build smaller graphs, one for each leaf instead of a complete graph for the whole tree. This method does however come with a potential loss of accuracy. To test how severe the loss of accuracy was, the mean transduction error rate was measured for both methods as well as the time it takes to complete the transduction step over the same data set. Table 6.3, 6.4, 6.5 show the mean transduction error of our tree-based graph traversal and that of a complete graph traversal over tree separate

amounts of labeled data. Table 6.6 shows the training time differences between the two transduction approaches.

Table 6.3: Transduction evaluation with 5% labeled instances

traversal approach	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Tree-based	0.5641	0.0301	0.0773	0.7337	0.2173	0.1523	0.4013	0.2103	0.0669
Complete-graph	0.5986	0.0266	0.0726	0.7480	0.2130	0.1556	0.3291	0.2013	0.0620

Table 6.4: Transduction evaluation with 20% labeled instances

traversal approach	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Tree-based	0.4174	0.0256	0.0650	0.7381	0.2065	0.1354	0.3258	0.1954	0.0626
Complete-graph	0.5970	0.0267	0.0582	0.7413	0.2132	0.1481	0.3231	0.1996	0.0617

Table 6.5: Transduction evaluation with 60% labeled instances

traversal approach	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Tree-based	0.3809	0.0246	0.0617	0.7448	0.2083	0.1295	0.3507	0.1911	0.0601
Complete-graph	0.5958	0.0256	0.0567	0.7481	0.2119	0.1474	0.3191	0.1991	0.0606

Table 6.6: Average test time in seconds

traversal approach	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Tree-based	99.0	791.7	541.0	488.3	245.0	848.3	1061.3	1718.6	1500.0
Complete-graph	376.0	6444.7	581.3	610.3	3754.0	1155.7	992.3	3448.0	3030.6

Chapter 7

Analysis

To answer RQ1 "*Is there a significant difference in the amount of required training data when using our active learning approach compared to random sampling?*", the results of using active learning were analyzed by applying the Wilcoxon Signed-Ranks test. The motivation for choosing this test can be found in section 5.3. In order to do so the differences between the algorithms in table 6.1 must be calculated and assigned ranks. The rankings over the different data sets can be seen in table 7.1. We state the following null hypothesis for RQ1: "*There is no significant difference between our active learning approach and sampling new training data at random*".

Table 7.1: Active learning analysis

data set	active learning results[a]	random sample results[b]	difference[c]	rank
d_1	14.5633	11.6775	-2,8858	9
d_2	7.3741	6.7887	-0,5854	6
d_3	0.7032	0.6666	-0,0366	3
d_4	15.9036	18.2467	2,3431	8
d_5	9.1999	9.1704	-0,0295	2
d_6	6.9777	6.6864	-0,2913	5
d_7	11.3921	11.2774	-0,1147	4
d_8	14.4568	13.2335	-1,2233	7
d_9	0.7002	0.7011	0,0009	1

[c] = [b] − [a]

After the ranks have been calculated, they are summarized into a positive and a negative sum depending on whether their difference was negative or positive and assign the smallest of the sums to be our statistical T value.

$$R^+ = 1 + 8 = 9$$
$$R^- = 2 + 3 + 4 + 5 + 6 + 7 + 9 = 36$$
$$T = 9$$

To reject the null hypothesis, that there is no significant difference between the two algorithms, the T value must be equal to or less than the critical value associated with the sample size used. Our sample size is the number of data sets used, 9 and our critical value for a one-tailed 0.05 level of significance is then 8 according to the table of critical T values for a Wilcoxon Signed-Ranks test (Sheskin, 2007). Since the T value is not equal to or less than 8, the null hypothesis cannot be rejected.

With this we conclude that our algorithm does not present a working approach to active learning for regression forests. The tendencies of the algorithm to become worse as the amount of training data increases could point to an error in the implementation. It is possible that we have misinterpreted the underlying mathematics presented by Criminisi et al. (2011). However, we have to the best of our abilities tried to reproduce

the work of Criminisi et al. and cannot at this time see any region of uncertainty that would cause deviations like the ones the evaluation shows. Alternatively, the approach does not work for regression in its current state because of the dissimilarities between classification and regression tasks.

It is possible that choosing a target for active learning based purely on the transduction error is inefficient. Another possibility is that the suggested approach for moving from transduction model to induction as described in section 4.3 does not work for regression type problems. This could be because of the complexity of making continuous predictions as compared to a discrete classification problem. Future work could focus on finding alternative ways to utilize the cluster structure of the approach to find optimal choices for the oracle.

Alternatively, research could look at re-implementing the proposed algorithm to not perform transduction. Instead the Mahalanobis distance would only be calculated as a measurement of how valuable a data instance is for active learning. The Induction model would then only be built from the labeled data, not the combined labeled and transducted data. The transition from transduction to induction could be the reason why the approach performs worse than the baseline algorithm, even when the decrease in mean absolute error is better.

The alpha value may also have to be empirically determined for each data set to produce optimal results. This theory is supported by the fact that the alpha value tests was performed on data set d_5, and as we can see in figure 6.4 that data set is one of the few where the mean absolute error is steadily declining. Future work could take the empirical alpha test and apply them on several data sets to see what correlation exists between the data set structure and the alpha value needed.

RQ2 "*Is there a significant difference between using the Mahalanobis distance for transduction compared to euclidean distance?*" is answered by analyzing the results from our comparison of the Mahalanobis distance and euclidean distance with regards to transduction error rate. We restate the question in the form of a the null hypothesis "*There is no significant difference between using the Mahalanobis distance compared to euclidean distance for transduction*".

Table 7.2: Mahalanobis versus Euclidean analysis

data set	mahalanobis results[a]	euclidean results[b]	difference[c]	rank
d_1	0.4261	0.6955	0.2694	6
d_2	0.0258	0.0509	0.0251	1
d_3	0.0587	0.1182	0.0595	2
d_4	2.6275	0.7421	-1.8854	8
d_5	0.2076	0.5213	0.3137	7
d_6	0.1339	0.3705	0.2366	5
d_7	0.3493	4.4223	4.0730	9
d_8	0.1949	0.5949	0.4000	8
d_9	0.0630	0.1265	0.0635	3

[c] = [b] − [a]

$$R^+ = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 9 = 37$$
$$R^- = 8$$
$$T = 8$$
$$Criticalvalue = 8$$

Again the T value is checked against the critical value of 8. For this test we can reject the null hypothesis since the T value is equal to the critical value of the test. With this, and the fact that $R^+ > R^-$ we conclude that there is a statistically significant difference between using the Mahalanobis distance compared to euclidean distance for the transduction scenario, and that the Mahalanobis distance is found to be more efficient. This was expected since the Mahalanobis distance can take advantage of the density cluster structure of the trees, making paths that better follow the structure of the data.

The results presented in table 6.3, 6.4, 6.5 are analyzed in a similar matter to see if our tree-based traversal approach lead to any improvement in training time with a minimal loss in accuracy.

Table 7.3: 5% labeled analysis

data set	tree-based traversal[a]	complete graph traversal[b]	difference[c]	rank
d_1	0.5641	0.5986	0.0345	8
d_2	0.0301	0.0266	-0.0035	2
d_3	0.0773	0.0726	-0.0047	3
d_4	0.7337	0.7480	0.0143	7
d_5	0.2173	0.2130	-0.0043	4
d_6	0.1523	0.1556	0.0033	1
d_7	0.4013	0.3291	-0.0722	9
d_8	0.2103	0.2013	-0.009	6
d_9	0.0669	0.0620	-0.0049	5

$^c = {}^b - {}^a$

$$R^+ = 8 + 7 + 1 = 16$$
$$R^- = 2 + 3 + 4 + 5 + 6 + 9 = 29$$
$$T = 16$$
$$Criticalvalue = 8$$

Table 7.4: 20% labeled analysis

data set	tree-based traversal[a]	complete graph traversal[b]	difference[c]	rank
d_1	0.4174	0.5970	0.1796	9
d_2	0.0256	0.0267	0.0011	2
d_3	0.0650	0.0582	-0.0068	7
d_4	0.7381	0.7413	0.0032	4
d_5	0.2065	0.2132	0.0067	6
d_6	0.1354	0.1481	0.0127	8
d_7	0.3258	0.3231	-0.0027	3
d_8	0.1954	0.1996	0.0042	5
d_9	0.0626	0.0617	-0.0009	1

$^c = {}^b - {}^a$

$$R^+ = 2 + 4 + 5 + 6 + 8 + 9 = 34$$
$$R^- = 1 + 3 + 7 = 11$$
$$T = 11$$
$$Criticalvalue = 8$$

Table 7.5: 60% labeled analysis

data set	tree-based traversal[a]	complete graph traversal[b]	difference[c]	rank
d_1	0.3809	0.5958	0.2149	9
d_2	0.0246	0.0256	0.001	2
d_3	0.0617	0.0567	-0.005	5
d_4	0.7448	0.7481	0.0033	3
d_5	0.2083	0.2119	0.0036	4
d_6	0.1295	0.1474	0.0179	7
d_7	0.3507	0.3191	-0.0316	8
d_8	0.1911	0.1991	0.008	6
d_9	0.0601	0.0606	0.0005	1

$^c = {}^b - {}^a$

$$R^+ = 1 + 2 + 3 + 4 + 6 + 7 + 9 = 32$$
$$R^- = 5 + 8 = 13$$
$$T = 13$$
$$Criticalvalue = 8$$

Using the same critical value as before, the null hypothesis cannot be rejected for any of the scenarios. As such, there is no significant difference in accuracy when using our proposed tree-based traversal method. This is an unexpected success for the tree-based traversal method since the loss of accuracy was expected to be significant over all amounts of labeled data.

Last we will analyze the test time difference between the tree-based and complete graph traversal approaches. Originally, the tree-based graph traversal was implemented to help speed up the transduction process. Again we use the Wilcoxon test for the two approaches.

Table 7.6: Average time analysis

data set	tree-based traversal[a]	complete graph traversal[b]	difference[c]	rank
d_1	99.0	376.0	277.0	4
d_2	791.7	6444.7	5653.0	9
d_3	541.0	581.3	40.3	1
d_4	488.3	610.3	122.0	3
d_5	245.0	3754.0	3509	8
d_6	848.3	1155.7	307.4	5
d_7	1061.3	992.3	-69.0	2
d_8	1718.6	3448.0	1729.4	7
d_9	1500.0	3030.6	1530.6	6

Note. Average time for all labeled cases
$^c = {}^b - {}^a$

$$R^+ = 1 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 43$$
$$R^- = 2$$
$$T = 2$$
$$Criticalvalue = 8$$

The T value is low enough to reject the null hypothesis. With this and the fact that $R^+ > R^-$ we conclude that the tree-based graph approach is faster than the complete graph approach on a 0.05 significance level. From the results we can also see that the tree-based traversal scales better than the complete approach as the amount of labeled data grows. This together with accuracy tests show that the tree-based traversal approach is to be preferred since it not only decreases the training time of the algorithm, but does so with no significant difference in accuracy.

Chapter 8

Conclusion

In this thesis a new approach for active learning via transduction for regression forests is presented. It is based on constructing semi-supervised density forests and using these to create a transduction model. From this model it is possible to make active choices on which training data would have the highest training value for continued training. The model was evaluated over several data sets and the decrease in error percentage was measured over a number of active learning iterations.

The usage of Mahalanobis distance for assigning labels via transduction was also compared to using euclidean distance. In addition, evaluations of two separate approaches for traversing the graph structure during transduction, one based on a complete graph traversal and one based on using a tree structure of smaller graphs were conducted.

The results from the statistical tests have shown that there was no significant difference between our approach to active learning and that of a supervised Random Forests algorithm selecting new labeled instances at random. However even if the statistical testing could not conclude any difference between the two approaches, it seems clear from the results in figure 6.4 that our suggested approach is not working as intended. The tendencies of the algorithm to become worse as more labeled data is added could point to an error in the implementations of the original approach by Criminisi et al. (2011), or simply that the approach does not work for regression in its current shape. The work of this thesis could be the foundation for future work that achieves a reduction in the required amount of training data. This represents a contribution since no previous research has been done on implementing active learning for regression forests.

The results from comparing the two distance measures showed that the Mahalanobis distance had a lower mean absolute error compared to using euclidean distance. This implies that the Mahalanobis distance is a fitting measurement for cluster based transduction, something that was not statistically analyzed by Criminisi et al. (2011).

Finally, the tests of the two traversal methods showed that our tree-based traversal approach was significantly faster than the complete graph approach, also no difference in error rate could be found between the two approaches, making tree-based traversal a valid option for transduction.

Future work will look at doing a more extensive alpha value analysis, as well as testing different implementation models for going from the transduction stage to the final induction model. This could hopefully lead to a fully working approach to active learning for regression forests.

Bibliography

N. Basit and H. Wechsler. Explanation and prediction of nssnp-induced pathology using association mining, transduction, and active learning. *Advanced Studies in Biology*, 5(5):199–214, 2013.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

R. Burbidge, J. Rowland, and R. King. Active learning for regression based on query by committee. *Lecture Notes in Computer Science*, 4881 LNCS:209–218, 2007.

A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2-3):81–227, 2011.

B. Demir and L. Bruzzone. A multiple criteria active learning method for support vector regression. *Pattern Recognition*, 47(7):2558–2567, 2014.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

M. Denil, D. Matheson, and N. De Freitas. Narrowing the gap: Random forests in theory and in practice. volume 2, pages 1006–1016, 2014.

V. Dhar. Data science and prediction. *Commun. ACM*, 56(12):64–73, Dec. 2013.

T. G. Dietterich. Machine-learning research: Four current directions. *AI magazine*, 18 (4):97, 1997.

P. Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55(10), Oct. 2012. ISSN 0001-0782.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

P. Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.

M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

J. Garcke and T. Vanck. Importance weighted inductive transfer learning for regression. *Lecture Notes in Computer Science*, 8724 LNAI(PART 1):466–481, 2014.

X. Goldberg. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6:1–116, 2009.

Y. Gu, D. Zydek, and Z. Jin. Active learning based on random forest and its application to terrain classification. *Advances in Intelligent Systems and Computing*, 1089:273–278, 2015.

R. L. Iman and J. M. Davenport. Approximations of the critical region of the fbietkan statistic. *Communications in Statistics-Theory and Methods*, 9(6):571–595, 1980.

U. Johansson, H. Boström, T. Löfström, and H. Linusson. Regression conformal prediction with random forests. *Machine Learning*, 97(1-2):155–176, 2014.

M. Lichman. UCI machine learning repository, 2013.

R. Liere and P. Tadepalli. Active learning with committees for text categorization. *Proceedings of the National Conference on Artificial Intelligence*, pages 591–596, 1997.

J. Maiora, B. Ayerdi, and M. Graña. Random forest active learning for aaa thrombus segmentation in computed tomography angiography images. *Neurocomputing*, 126: 71–77, 2014.

S. Makridakis and M. Hibon. *Evaluating Accuracy (or Error) Measures*. INSEAD working paper. INSEAD, 1995.

J. Mendes-Moreira, A. Jorge, J. De Sousa, and C. Soares. Comparing state-of-the-art regression methods for long term travel time prediction. *Intelligent Data Analysis*, 16(3):427–449, 2012a.

J. a. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa. Ensemble approaches for regression: A survey. *ACM Comput. Surv.*, 45(1):10:1–10:40, Dec. 2012b.

T. M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072.

L. Rokach. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics & Data Analysis*, 53 (12):4046 – 4072, 2009.

B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

D. J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, 4 edition, 2007.

J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. pages 1297–1304, 2011.

J. Su, S. Jelber, S. Matwin, and J. Huang. Active learning with automatic soft labeling for induction of decision trees. *Lecture Notes in Computer Science*, 5549 LNAI: 241–244, 2009.

M. Sun, P. Kohli, and J. Shotton. Conditional regression forests for human pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3394–3401, June 2012.

O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

F. Wilcoxon. Individual comparisons of grouped data by ranking methods. *Journal of economic entomology*, 39:269, 1946.

I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011.

X. Zhu, J. Lafferty, and R. Rosenfeld. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Language Technologies Institute, School of Computer Science, 2005.

Appendix A

Overview of previous work

This appendix will give an overview of the method proposed by Criminisi et al. for semi-supervised classification using transduction. A semi-supervised density Forests as described by (Criminisi et al., 2011) is an algorithm that uses two types of training data, $v^l \in L$ the labeled training data and $v^u \in U$ the unlabeled training data. Criminisi et al. (2011) gives the following equation for optimized training of a semi-supervised forest:

$$\Theta_j^* = \arg\max_{\Theta_j \in \tau_j} I_j \tag{A.1}$$

With I_j representing the information gain of one node in a tree. For semi-supervised learning, the total information gain I_j of a node has to take into account the information gain from the labeled training data as well as the separate information gain from the combined labeled and unlabeled training data. The following equation is from (Criminisi et al., 2011):

$$I_j = I_j^s + \alpha I_j^u \tag{A.2}$$

Here I_j^s is the supervised labeled information gain and I_j^u is the information gain calculated from high density regions of both unlabeled and labeled training data. The α value is a scalar parameter that specifies the relative weight between the two types of information gain.

Criminisi et al. calculates both I_j^s and I_j^u using differential entropy, entropy is an information theory measure of the average amount of information contained within a variable. Criminisi et al. calculates it using the following equations:

$$I_j^s = \sum_{v \in S} log(|\Lambda_y(v)|) - \sum_{i \in L,R} \left(\sum_{v \in S^i} log(|\Lambda_y(v)|) \right) \tag{A.3}$$

$$I_j^u = log|\Lambda(S_j)| - \sum_{i \in L,R} \frac{|S_j^i|}{|S_j|} log|\Lambda(S_j^i)| \tag{A.4}$$

Where Λ_y from equation A.3 represents the conditional covariance matrix computer from probabilistic linear fitting of the labeled data(Criminisi et al., 2011) and Λ from equation A.4 represents the standard covariance matrix calculated over both labeled and unlabeled data. A covariance matrix represents the degree of variation between attributes over a set of data points, and is calculated with the following equation:

$$Cov(X, Y) = \sum (X_i - \bar{X})(Y_i - \bar{Y})/N \tag{A.5}$$

$$A = \begin{bmatrix} Cov(x_1, x_1) & Cov(x_1, x_1) & \dots & Cov(x_1, x_n) \\ Cov(x_2, x_1) & Cov(x_2, x_1) & \dots & Cov(x_2, x_n) \\ \dots & \dots & \dots & \dots \\ Cov(x_n, x_1) & Cov(x_n, x_2) & \dots & Cov(x_n, x_n) \end{bmatrix} \tag{A.6}$$

With X and Y representing different attributes and i the instance of data to summarize over. This means that each cell in the matrix will contain the mean covariance of the data set between attribute X and Y. A conditional covariance matrix is similar but contains the conditional covariance over a certain variable, in our case the variable correlating to the regression value. The reason that the covariance matrix can be used to obtain a differential entropy comes from the properties of its determinant. Calculating the absolute value of the determinant gives the volume of the ellipsoid that covers the data used to construct the covariance matrix (Criminisi et al., 2011). A drawback with using the covariance matrix to calculate entropy is that all instances must have a value for each attribute. This means that the approach does not work for a large number of data sets which contain instances with missing values.

Once a tree has been constructed the clusters of each leaf can be used to perform transduction to assign new labels:

$$C(V^u) \leftarrow c(\arg\min_{v^1 \in L} D(v^u, v^l)) \forall v^u \in U \tag{A.7}$$

C$(\cdot)$ indicates the label for an instance. L is the set of all labeled data and U is the set of all unlabeled data. And D$(\cdot, \cdot)$ is a geodesic distance described by Criminisi et al. (2011) as:

$$D(v^u, v^l) = \min_{\Gamma \in \{\Gamma\}} \sum_{i=0}^{L(\Gamma)-1} d(s_i, s_{i+1}) \tag{A.8}$$

Where Γ is a geodesic path, $L(\Gamma)$ the length of the path, $\{\Gamma\}$ is the set of all possible geodesic paths and s_0 being the start point and $s_{L(\Gamma)}$ the end point. The local distances d$(\cdot, \cdot)$ is in Criminisi et al. work defined as a symmetric Mahalanobis distance. The Mahalanobis distance is a measure over the dissimilarity between two vectors over the same distribution. The formula used by Criminisi et al. (2011) to calculate in the following way:

$$d(s_i, s_j) = \frac{1}{2}(\boldsymbol{d}_{ij}^T \boldsymbol{\Lambda}_{l(vi)}^{-1} \boldsymbol{d}_{ij} + \boldsymbol{d}_{ij}^T \boldsymbol{\Lambda}_{l(vj)}^{-1} \boldsymbol{d}_{ij}) \tag{A.9}$$

Where s_i and s_j are the two vectors, $\boldsymbol{d}_{ij}$ is s_i - s_j, and $\boldsymbol{\Lambda}_{l(vi)}$ and $\boldsymbol{\Lambda}_{l(vj)}$ are the covariance matrices for s_i respectively s_j.