**Project 2 – Decision Trees, Linear Regression, Model Trees, Regression Trees**

**CS548 / BCB503 Knowledge Discovery and Data Mining - Fall 2018**

**Prof. Carolina Ruiz**

**Students:** Paritosh Goel, Lei Shi

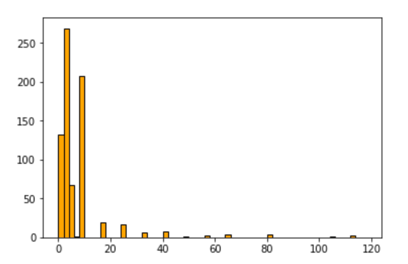
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Classification** | | **Regression** | |
| **Dataset :**   * Dataset Description * Data Exploration * Initial Data Preprocessing (if any) | /05  /10  /05 | | | |
| **Code Description:** | **Weka**  /10 | **Python**  /10 | **Weka**  /10 | **Python**  /10 |
| **Experiments:**   * Guiding Questions | /10 | | /10 | |
| * Sufficient & coherent set of experiments | /10 | /10 | /10 | /10 |
| * Objectives, Parameters, Additional Pre/Post-processing | /10 | /10 | /10 | /10 |
| * Presentation of results | /10 | /10 | /10 | /10 |
| * Analysis of individual experiments’ results | /10 | /10 | /10 | /10 |
| Summary of Results, Analysis, Discussion, and Visualizations | /20 | | /20 | |
| Advanced Topic | /30 | | | |
| Total Written Report | /310 = /100 | | | |

**Dataset Description, Exploration, and Initial Preprocessing: (at most 1 page)**

**[05 points] Dataset Description: (e.g., dataset domain, number of instances, number of attributes, distribution of target attribute, % missing values, …)**

This dataset represents absence instances for 36 employees. Each row corresponds to an absence activity of an employee in hours with some other details which can be responsible for the employee absence. The total number of instances is 740 and total number of attributes is 21 including class attribute which is called “Absenteeism time in hours”. The target attribute has a mean value of 6.92, standard deviation of 13.33, maximum value of 120, and minimum value of 0. The Q1, Q2 and Q3 values of target attribute are 2, 3, 8, respectively. According to the description of dataset, we found 44 instances with missing values, which is 6% of the whole dataset.

**[10 points] Data Exploration: (e.g., comments on interesting or salient aspects of the dataset, visualizations, correlation, issues with the data, …)**

1. The employee whose ID number is 3 has 113 records which is much larger than others. And this employee also has the longest absence hours, 482 hours. On the other hand, the employee whose ID number is 14 has 476 hours of absence time with only 29 records.
2. Most frequent reasons for absence are medical consultation and dental consultation.
3. Majority employees shown in the dataset have high school level education.
4. Distribution of target class: “Absenteeism time in hours”. From the plot, we can see that most of absence time is under 12 hours. 
5. Correlation matrix shows that “Service time” and “Age” are also highly related to each other. “Service time” and “Age” are both highly related to “Body mass index”. Furthermore, “Month of absence” is highly related to “Seasons”.
6. The range of Absenteeism time in hours is 0-120. We notice that 44 rows have 0 of target attribute, which is a corrupted value, out of these 43 rows has 0 of “reason for absence”. In addition, one row has 0 of “month of absence”.

**[05 points] Initial data preprocessing, if any, based on data exploration findings: (e.g., removing IDs, strings, necessary dimensionality reduction, …)**

*Note: Keep initial data preprocessing to a minimum – just the necessary – and instead experiment with data preprocessing in the following sections.*

1. We remove “ID” attribute, “Height” and “Weight”. Because “Body mass index” is computed according to height and weight.
2. We remove 44 instances with missing values.
3. Foe classification, we discrete target attribute into 6 equal-frequency bins. For regression, we encode the “Reason for absence” with dummy numbers. Because there are 28 unique values in “Reason for absence”, we create 28 new predictors.

**Weka Code Description: Inputs, output, and process followed by Weka’s code to construct the trees (at most 2/3 page)**

**[10 points] J4.8 Code Description:**

Input - (Instances), flags like - use reduced pruning, multiway split or binary split etc. Evaluation - Evaluation eval = new Evaluation(data); eval.evaluateModel (classifier, data); Output - tree when we build the classifier and class when we use classifyInstancMethod.

Sample code - J48 classifier = new J48(); classifier.buildClassifier(data); classifier.classifyInstance(instance);

Process - Choose a split selector class based on binary split or multiple split. Use the split selector object to choose the split condition. Recursively follow this approach until we get a leaf node. Each node will contain a model object which has the distribution information/ split condition for that node. Each node will have a reference to its sons to reach out to child nodes.

Implementation Classes - **Split Selector classes** - BinC45ModelSelection (if only binary splits) otherwise, C45ModelSeletion (multiple splits) **Model classes** - C45Split, ClassifierSplitModel **Node classes** - C45PruneableClassifier (If reduced error pruning is not used) otherwise PruneableClassifier

**[10 points] M5P Code Description:**

Input - instances; flags like -R to Build regression tree etc.

Output - classifier tree will be built in the object of M5P, which could be used to classify a new instance.

Process followed - Internally the implementation uses rules which created while we build the classiffer, later on these rules helps to predict the correct values

Sample code used - M5P retval = new M5P(); retval.buildClassifier(data); retval.classifyInstance(inst);

**[20 points] Python Packages and Functions used (decision trees, linear regression, model/regression trees). Describe inputs & outputs (at most 1/3 page)**

Package: numpy, pandas, sklearn.

KFold.split(inputs: data to be split; outputs: indices for training and indices for testing).

For decision tree: precision\_score (inputs: test targets, predictions on the test predictors; outputs: precisions for each target classes), recall\_score (inputs: test targets, predictions on the test predictors; outputs: recalls for each target classes), roc\_auc\_score (inputs: test targets, prediction probabilities on the test predictors; outputs: ROC AUC for each target classes).

For linear regression: mean\_squared\_error (inputs: test targets, predictions on the test predictors; outputs: mean squared error), mean\_absolute\_error (inputs: test targets, predictions on the test predictors; outputs: mean squared error).

For regression tree: DecisionTreeRegressor (outputs: a regression tree without training)

**[10 points] Three Guiding Questions for the Classification Experiments: (at most 1/3 page)**

1. What is the most common value of Absenteeism in hours?
2. Which particular factor is most responsible for deciding in hours of Absenteeism?
3. How are other factors related to absenteeism?

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[40 points] Summary of Classification Experiments in Weka. Use 10-fold cross-validation** *At most 2/3 page.* | | | | | | | | | | |
| **Tech.** | **Guiding**  **questions** | **Pre-process** | **Parameters** | **Post-process &**  **Pruning** | **Accuracy,** **Precision, Recall, ROC Area** | **Time to build model** | **Size of model** | **Interesting patterns in the model** | **Analysis & observations about experiment** |  |
| ZeroR | 1 | Discretize Filter in Weka | Whole data set | None | Accuracy: 28.11%  Precision: 0.281  Recall: 1  ROC Area for ‘(7,5 - 12]’: 0.495 | 0 sec | 1 node | max instances lie in range 7.5 - 12. | ZeroR predicts class value: '(7.5-12]' |  |
| OneR | 2 | Discretize Filter in Weka | Whole Data Set | None | Accuracy: 48.9189 %  Precision:  0.878, 0.384, 0.063, ?, 0.571, 0.583  Recall:  0.326 ,0.885 ,0.009,  0, 0.827 ,0.111  ROC Area:  0.658 ,0.751,0.493,0.500 ,0.792 ,0.552 | 0.01 sec | 1 parent node and  28 leaf nodes | -Neoplasm/Diseases of the circulatory system -> max hours of Absenteeism  - Reasons of absence (0, 23, 28, 25 , 27, 16, 4) leads to lower hours of absenteeism  -No prediction - [3.5-7.5] | Recall of 3rd category is not good |  |
| Decision Tree | 3 | 1. Discretize  2. Remove correlated attributes  3. Remove attributes which were not showing up in the tree  total attributes = 13+ | same as above | reduced error pruning = true | Accuracy - 49.459%  Precision - 0.573, 0.416, 0.283, 0.346, 0.589, 0.474  Recall - 0.508, 0.631, 0.152, 0.132, 0.793, 0.143, 0.495  ROC - 0.808, 0.770, 0.643, 0.744, 0.845, 0.817 , 0.780 | 0.01 sec | No of leaves = 42  size of tree = 54 | Employee with unjustified absence (no 27) have more absence if their hit target is low | +minNumOfObjects -> better accuracy |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[40 points] Summary of Classification Experiments in Python. Use 10-fold cross-validation** *At most 2/3 page.* | | | | | | | | | | |
| **Tech.** | **Guiding**  **questions** | **Pre-process** | **Parameters** | **Post-process &**  **Pruning** | **Accuracy,** **Precision, Recall, ROC Area** | **Time to build model** | **Size of model** | **Interesting patterns in the model** | **Analysis & observations about experiment** |  |
| Decision Tree | 2, 3 | Nothing | 1. Criterion: ”entropy” 2. Random\_state: 1 3. Min\_samples\_split: 10 | Implement pre-pruning. Stop splitting when number of instances is less than 10. | * Accuracy: 0.37 * Precision: 0.21, 0.45, 0.35, 0, 0.44, 0.4 * Recall: 0.35, 0.43, 0.35, 0, 0.52, 0.28 * ROC Area: 0.72, 0.64, 0.67, 0.59, 0.68, 0.66 | 0.024 sec | 187 | The precision and recall for class “(3.5, 7.5)” are 0. | Size of this tree is smaller because we implement pre-pruning. In addition, this model predicts totally wrong on class “(3.5, 7.5)”. |  |
| Decision Tree | 2, 3 | * Discrete predictors * Select k best predictors | 1. Predictor number: 10 2. Criterion: ”entropy” 3. Random\_state: 1 | Implement pre-pruning. Stop splitting when number of instances is less than 10. | * Accuracy: 0.37 * Precision: 0.43, 0.44, 0.33, 0.13, 0.52, 0.06 * Recall: 0.43, 0.44, 0.4, 0.14, 0.43, 0.08 * ROC Area: 0.67, 0.70, 0.69, 0.51, 0.60, 0.47 | 0.021 sec | 493 | The precision and recall for class “(12.0, inf]” are much lower than other classes. | Size of the tree is large because it not pruned. In addition, this model predicts worse on class ”(12.0, inf]”. |  |
| Decision Tree | 2, 3 | Select some best predictors | 1. Predictor number: 10 2. Criterion: ”entropy” 3. Random\_state: 1 4. Min\_samples\_split: 10 | Implement pre-pruning. Stop splitting when number of instances is less than 10. | * Accuracy: 0.42 * Precision: 0.45, 0.48, 0.42, 0, 0.51, 0.18 * Recall: 0.65, 0.46, 0.25, 0, 0.63, 0.16 * ROC Area: 0.79, 0.74, 0.69, 0.44, 0.79, 0.58 | 0.016 sec | 163 | The precision and recall of class “(3.5, 7,5]” are 0. | Size of the tree is small. In addition, the accuracy increases. However, this model predicts totally wrong on class “(3.5, 7,5]”. |  |

**[20 points] Summary of Weka and Python Classification Results, Analysis, Discussion, and Visualizations (at most 1/3 page)** 1. Analyze the effect of varying parameters/experimental settings on the results. 2. Analyze the results from the point of view of the dataset domain, and discuss the answers that the experiments provided to your guiding questions. 3. Include (a part of) the best classification model obtained.

1. Pruning reduces the complexity of a decision tree and hence improves the accuracy of the same.

2. Most important deciding factor is Reason for absence. And people who are social drinker, tend to increase absenteeism.

3. We got the best classification model in Weka with 49.45% accuracy. Parameters - total no of attributes used 13, - reduced error pruning = true.

**[10 points] Three Guiding Questions for the Regression Experiments: (at most 1/3 page)**

1. What is the mathematical expression that could be used to define absenteeism in terms of factors provided?
2. What are the different mathematical expressions that can define absenteeism under different circumstances?
3. Can we generalize the conditions for rules? Can we improve our efficiency in predicting the absenteeism?

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[40 points] Summary of Regression Experiments in Weka. Use 10-fold cross-validation.** *At most 2/3 page.* | | | | | | | | | | |
| **Tech.** | **Guiding**  **questions** | **Pre-process** | **Parameters** | **Post-**  **process**  **& Pruning** | **Correlation**  **Coefficient**  **and Error Metric(s)** | **Time to build model** | **Size of model** | **Interesting patterns in the model** | **Salient observations about experiment** |  |
| Model Tree with M5P | 1 | Nominal attribute (Reason for absence) to Binary | Build Regression Tree = false | None | 1. Correlation coefficient - 0.361 2. Mean absolute error - 5.4734  3. Root mean squared error - 12.4661 | 0.13 sec | Number of rules = 1 | Apart from Reason of absence, the rule has 4 attributes -  1. Day of the week  2. Age  3. Children  4. Pet | We can turn of the tree flag and get a single equation from M5P |  |
| Regression Tree with M5P | 2 | Nominal attribute (Reason for absence) to Binary | M5P with prune = false  Build Regression Tree = True | None | 1. Correlation coefficient - 0.2527  2. Mean absolute error - 5.6312  3. Root mean squared error - 13.2425 | 0.14 sec | Number of Rules = 229 | Nothing | with pruning = false, size of the tree is exponentially large |  |
| Regression Tree with M5P | 2, 3 | Nominal attribute (Reason for absence) to Binary | Build Regression Tree = True | pruning = true | 1. Correlation coefficient - 0.0866  2. Mean absolute error - 5.8894  3. Root mean squared error - 13.2959 | 0.08 sec | Number of Rules = 2 | only on one attribute - reason of absence =23 | Pruning can drastically reduce the complexity of tree.  This tree is predicting only 2 outputs |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[40 points] Summary of Regression Experiments in Python. Use 10-fold cross-validation.** *At most 2/3 page.* | | | | | | | | | | |
| **Tech.** | **Guiding**  **questions** | **Pre-process** | **Parameters** | **Post-**  **process**  **& Pruning** | **Correlation**  **Coefficient**  **and Error Metric(s)** | **Time to build model** | **Size of model** | **Interesting patterns in the model** | **Salient observations about experiment** |  | |
| Linear Regression | 1 | Nothing | Default | Nothing | * Correlation Coefficient: 0.108 * Mean squared error: 207.44 * Mean absolute error: 6.34 | 0.012 second | Not apply | More than half of the predictors are negatively related to the target attribute. | Most of predictors are negatively related to the target attribute. |  | |
| Linear Regression | 1 | Implement PCA. | PCA: 25 components | Nothing | * Correlation Coefficient: 0.146 * Mean squared error: 198.52 * Mean absolute error: 6.41 | 0.008 second | Not apply | When number of PCA components is less than 15, correlation coefficient is negative. | Implementing PCA increases the correlation coefficient a lot and decreases the time to build the model. |  | |
| Regression Tree | 2, 3 | Implement PCA. | PCA: 35 components  Min\_samples\_split: 40 | Implement pre-pruning. | * Correlation Coefficient: 0.121 * Mean squared error: 204.39 * Mean absolute error: 5.53 | 0.120 second | 73 | Nothing | Implementing PCA and pre-pruning increases the time of building model. |  | |

**[20 points] Summary of Weka and Python Regression Results, Analysis, Discussion, and Visualizations (at most 1/3 page)** 1. Analyze the effect of varying parameters/experimental settings on the results. 2. Analyze the results from the point of view of the dataset domain, and discuss the answers that the experiments provided to your guiding questions. 3. Include (a part of) the best regression model obtained.

1. Implementing PCA and pre-pruning increases the correlation coefficient of the model.

2. Some reasons of absences are positively related to target attribute and some are negatively related to target attribute. The reason with largest coefficient is No. 19, which is Injury, poisoning and certain other consequences of external causes.

3. The best regression model obtained is model tree implementing M5P in Weka. Its correlation coefficient is 0.361.

**Advanced Topic (AT MOST 1 PAGE): AdaBoost**

**[7 points] List of sources/books/papers used for this topic (include URLs if available):**

* AdaBoost Classifier: <https://medium.com/machine-learning-101/https-medium-com-savanpatel-chapter-6-adaboost-classifier-b945f330af06>
* Sklearn ensemble methods: <http://scikit-learn.org/stable/modules/ensemble.html#adaboost>
* Boosting and AdaBoost for Machine Learning: <https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning>

**[20 points] In your own words, provide an in-depth, yet concise, description of your chosen topic. Make sure to cover all relevant data mining aspects of your topic.** *Your description here should be in-depth and at the graduate level.*

AdaBoost is an ensemble method that creates a strong classifier from a number of weak classifiers. Each weak classifier has a weight and AdaBoost use the weighted average prediction of all weak classifiers as the final prediction. More specifically, a weak classifier with 50% accuracy is given a weight of zero, and a weak classifier with less than 50% accuracy is given a negative weight. This is because when a classifier accuracy is 50%, it contributes nothing to the strong classifier and if a classifier has an accuracy of less than 50% it has negative effect on the strong classifier.

During the training process, there are two kinds of weight. One is the weight of data point. At the beginning of the training, AdaBoost will initialize the weight for each data point. The initial weight is 1/n (n is the number of data points). And after training a weak classifier at any level, AdaBoost assigns new weight to each data point. Misclassified data point is assigned higher weight so that it is more likely to appear in the training subset of next weak classifier. The other weight is the weight of weak classifier. After training each classifier, a new weight is assigned to this classifier based on accuracy. Therefore, weak classifier with higher accuracy will contribute more to the strong classifier.

Last but not least, AdaBoost require the high-quality training data because it continues to attempt to correct misclassifications in the training data. Before implementing AdaBoost, we better remove outliers and noise of training data.

**[3 points] How does this topic relate to trees and the material covered in this course?**

AdaBoost is an ensemble method for classification. We can use decision trees as the weak classifiers. Combining these decision trees using AdaBoost, we can acquire a better classifier than each of decision tree.

**Authorship:** Although each student on the team is expected to be involved in every aspect of the project, describe in detail here the main contributions that each of the team members made to this project. This authorship description must accurately reflect the work done by each team member, and must be approved by all of the members of the team (at most 1/3 page)

Paritosh Goel and Lei Shi both experimented in Weka and Python with all the aspects.

Paritosh Goel focused more on Weka experiments and Java Code. Paritosh also did basic Python experiments such as data pre-processing, data exploration, and basic classification problems in Python.

Lei Shi focused more on data pre-processing, data exploration, and Python experiments. Lei also tried some experiments in Weka.