Fundamentals of Software Analytics

Summer Term 2019
Assignment Sheet # 4

Faculty of Digital Engineering
Computer Graphics Systems Group
Prof. Dr. Jürgen Döllner

Objectives

With these assignments you will learn how to:

- use different Machine Learning techniques for the prediction of values and binary classification, and
- use dimensionality reduction techniques and computational geometry for similarity assessment.

For your programming tasks, you may use one of the following languages: Bash, Python, batch, PowerShell, C, C++, or Java.

Task 4.1: Linear Regression for Metric Prediction (3 Points)

Some software metrics of interest are hard to compute or have intrinsic disambiguities. For such cases, Machine Learning techniques can effectively cope with the uncertainties of data accuracy and algorithm design and implementation.

Task Implement a command-line tool that predicts source code metrics from other source code metrics. Therefore, perform linear regression[1] using the columns that are passed as a semicolon-separated list in the `--training-columns` parameter from a test dataset (passed with the `--train` parameter) to train a weight vector to predict the metric in the column identified through the `--prediction-column`. Use this weight vector to predict the metric values for the columns of the prediction dataset that is passed with `--predict`. The filenames and the predicted metric values should be printed to standard output with semicolon as separator. Both datasets are encoded as CSV file with semicolon as the column separator. The first line contains the column names.

Dataset The dataset is hosted on moodle[2]. It consists of a training dataset and a prediction dataset.

Example

```
> ./4-1-complexity-prediction --train training-data.csv --training-columns "F-Args;Classes;RLoC;Nl;NlMean" --
    prediction-column Mccc --predict prediction-data.csv
source/cpplocate/source/cpplocate.cpp;33
source/cpplocate/source/ModuleInfo.cpp;12
source/cpplocate/source/utils.cpp;17
```

Remarks You don't have to implement linear regression from scratch. You can use a Machine Learning library that supports linear regression.

Deliverables The submission should contain the following artifacts within the directory `4_1_complexity_prediction`.

Source Code the sources of the tool

[1] A mathematical introduction is available on moodle at https://moodle.hpi3d.de/mod/resource/view.php?id=4019. The matrix invert operations can be approximated using its pseudo inverse.

[2] The training dataset is hosted at https://moodle.hpi3d.de/mod/resource/view.php?id=4020 and the prediction dataset is hosted at https://moodle.hpi3d.de/mod/resource/view.php?id=4021.

Task 4.2: Binary Classification on NASA Software Dataset (9 Points)

Task Implement a command-line tool that can perform training and prediction on the NASA public domain defect dataset for error-proneness of software systems. The tool should support the two Machine Learning techniques Logistic Regression and Support Vector Machines[3]. Further, the program should be able to output either the predicted values or the error rates for training and prediction to standard output.

a) Logistic Regressions Use Logistic Regression to train the model on the file passed as `--train` parameter to predict error-proneness on the file passed as `--predict` parameter. The last column is the target for prediction.

b) Support Vector Machines Use Support Vector Machines to train the model on the file passed as `--train` parameter to predict error-proneness on the file passed as `--predict` parameter. The last column is the target for prediction. The fixed parameters for the training are the kernel function (a Gaussian kernel) with $\sigma = 0.5$ and $C = 1$.

c) Qualitative Comparison When the command line option `--output-error-values` is passed, omit the output of the predicted values. Instead, compute and output the error values in percent for both the training and the prediction dataset for the current type of Machine Learning method. Compile a text file where these values are documented for the NASA dataset.

Dataset The required NASA public domain defect dataset is hosted on moodle[4]. It consists of a training dataset and a prediction dataset. More datasets are available on the GitHub mirror https://github.com/klainfo/NASADefectDataset/tree/master/OriginalData/MDP.

Example

```
> ./4-2-defect-prediction --type=logistic-regression --train MC2-train.arff --predict MC2-predict.arff
N
N
N
> ./4-2-defect-prediction --type=support-vector-machine --train MC2-train.arff --predict MC2-predict.arff
Y
N
N
> ./4-2-defect-prediction --type=logistic-regression --train MC2-train.arff --predict MC2-predict.arff --
    output-error-values
Logistic regression error report
train error: 5%
prediction error: 30%
> ./4-2-defect-prediction --type=support-vector-machine --train MC2-train.arff --predict MC2-predict.arff --
    output-error-values
Support vector machine error report
train error: 1%
prediction error: 6%
```

Remarks You don't have to implement logistic regression or support vector machines from scratch. You can use a Machine Learning library that supports these techniques. Beware, that the task covering the logistic regression requires the use of a Gaussian kernel.

Deliverables The submission should contain the following artifacts within the directory `4_2_binary_classification`.

Source Code the sources of the tool

Report A text file containing the four values of train error and test error for both techniques

[3] A mathematical introduction is available on moodle at https://moodle.hpi3d.de/mod/resource/view.php?id=4018

[4] The training dataset is hosted at https://moodle.hpi3d.de/mod/resource/view.php?id=4022 and the prediction dataset is hosted at https://moodle.hpi3d.de/mod/resource/view.php?id=4023.

Task 4.3: Author Prediction for Software Modules (5 Points)

Finding a suitable developer for a source code module is a common task in software engineering. Especially in the cases of sick leaves, dismissals or restructured teams, finding a substitute for a former developer is crucial.

Task Implement a command-line program that gives suggestions for developers in a git repository (the repository can be expected to be in the current working directory). The program should have two phases:

Initialization Phase Analyze developer contributions by means of changed lines for each author. These changes should be converted to Bag-of-Words vectors with a length of 256 entries.

Query Phase After initialization, the program should read file names from standard input. For each line, perform the following query:

- Derive the Bag of Words vector for the source code file.
- Use the dimensionality reduction technique t-SNE to reduce all Bag-of-Words vectors to two dimensions.
- Construct a Voronoi diagram for the projected author data points (do not add the source code file data point as site for the Voronoi diagram).
- Perform a lookup for the nearest author of the source code file's Bag-of-Word vector by use of the Voronoi cells (effectively performing a nearest-neighbor search).

The program should output the filename and the most similar author separated by semicolon to the standard output.

Dataset Use the open source Github repository `https://github.com/microsoft/TypeScript`. It can be accessed by cloning it:

```
1 > git clone https://github.com/microsoft/TypeScript.git
```

Example

```
1 > cd TypeScript
2 > echo "scripts/bisect-test.ts" | ./4-3-author-prediction
3 scripts/bisect-test.ts;RyanCavanaugh
```

Remarks You should not implement t-SNE or Voronoi dissection by yourself. Use libraries that support these algorithms.

Deliverables The submission should contain the following artifacts within the directory `4_3_author_prediction`.

Source Code the sources of the tool

Instructions

Pair Programming On these assignments, you are encouraged (not required) to work with a partner provided you practice pair programming. Pair programming "is a practice in which two programmers work side-by-side at one computer, continuously collaborating on the same design, algorithm, code, or test." One partner is driving (designing and typing the code) while the other is navigating (reviewing the work, identifying bugs, and asking questions). The two partners switch roles every 30–40 minutes, and on demand, brainstorm.

Upload Results All described artifacts are submitted to the course moodle system `https://moodle.hpi3d.de/course/view.php?id=119` as a zipped archive with the following naming convention: `assignment_4_matrikNr1.zip` or `assignment_4_matrikNr1_matrikNr2.zip` whether or not pair programming was applied. Compiled, intermediate, or temporary files should not be included. If pair programming is used, the results are turned in only once. The assignments #4 are due on July 15th, 11:00 a.m.

Violation of Rules a violation of rules results in grading the affected assignments with 0 points.

- Writing code with a partner without following the pair programming instructions listed above (e.g., if one partner does not participate in the process) is a serious violation of the course collaboration policy.
- Plagiarism represents a serious violation of the course policy.