CS 3110: A6 Design

Jackson Palmer(jrp328), Yunjie Bi(yb89), Maxwell Milne(mjm686)

November 13, 2015

1 System Description

1.1 Key Idea

To develop a system that predicts the category of crimes that occurred in San Francisco using various supervised Machine Learning classification algorithms.

1.2 Key Features

1. CSV data (training set and testing set)
2. k-NN classifier
3. Bayes classifier
4. Decision tree (TDIDT)
5. Classifier Evaluation (EG and cross validation)

1.3 Description

Our system aims to classify crimes in San Francisco.

We plan to use three classic classification algorithms in Machine Learning to build a system that can provide a decent accuracy in predicting the category of a crime with given attributes. The algorithms that we think would be suitable for this task are k-NN (k-Nearest Neighbor), Bayes classifier and decision tree (with TDIDT). All three algorithms will be implemented in OCaml from scratch.

To eventually output a result with the three algorithms, we integrate 4-fold cross validation with Exponential Gradient Algorithm for Expert Setting (EG). We will explain how that works in details in later sections.

2 Architecture

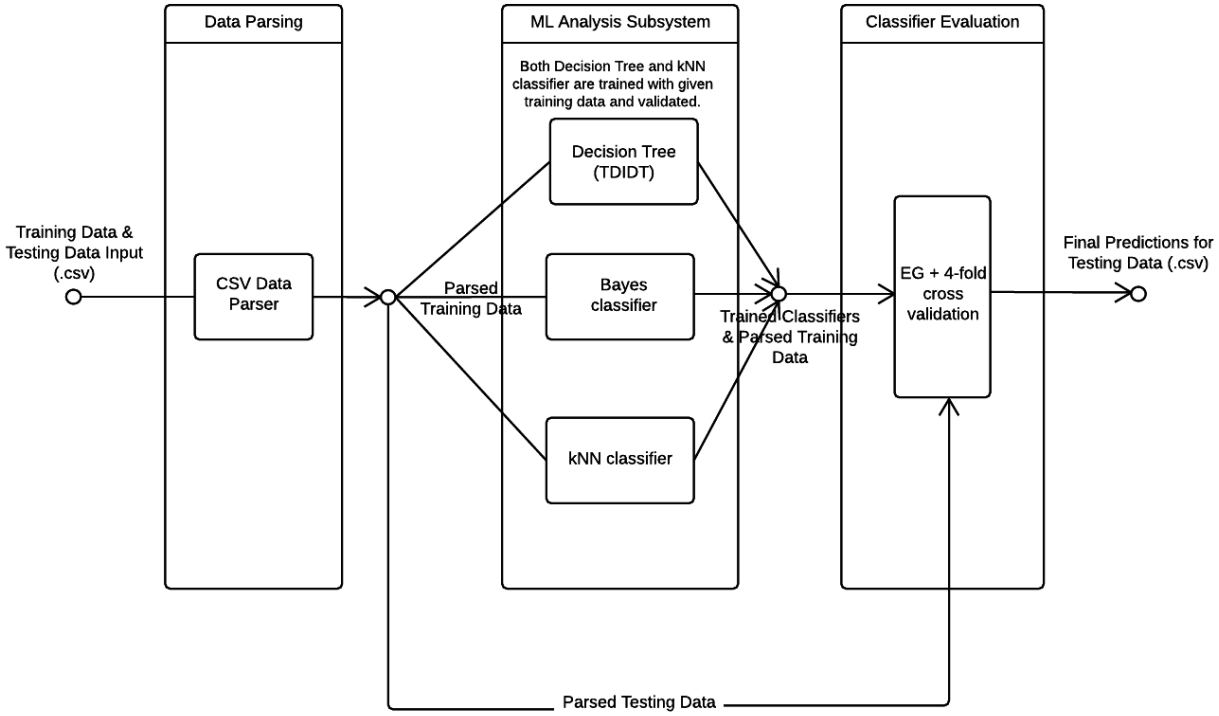
Our system can be seen as having three stages or subsystems.

- **Data Parsing**: As the name suggests, this stage is responsible for parsing the csv data we download from kaggle.com into potentially records with properly formatted attribute values.
- **Machine Learning Analysis**
  - Decision tree (TDIDT): a supervised learning algorithm. For a more detailed description of how the algorithm works, click here http://www.ke.tu-darmstadt.de/lehre/archiv/ws0809/mldm/dt.pdf
  - kNN classifier: a supervised learning algorithm. For a more detailed description of how the algorithm works, click here https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

  - Bayes classifier: a supervised learning algorithm. For a more detailed description of how the algorithm works, click here https://en.wikipedia.org/wiki/Naive_Bayes_classifier

- **Classifier Evaluation**: Since our ML Analysis stage has three algorithms, there must be a way for us to decide which result to take if they give different predictions/classifications for the same data point. We plan on doing this with 4-fold cross validation and EG.

  The intuition behind EG is that all algorithms start with the same weights. Here, we define a concept of loss, which is simply a binary value of correct classification (0) or incorrect classification (1) for this algorithm's prediction. Then we update the weight of each algorithm based on the loss incurred by its prediction on that particular data point. For a more detailed explanation of how we update the weight, please look here http://www.mit.edu/~9.520/spring08/Classes/online_learning_2008.pdf In the end, after going through all the training data, we would have different weights for these three algorithms based on their performances. Essentially, the algorithms that performed badly would have less weights. We can get four sets of weights from the 4-fold cross validation process, and the final weight of each algorithm would be the average of their four obtained weights. When we are actually running it on the testing data, we simply assign the probability of a class that an algorithm predicts as the algorithm's weight. All the predictions for one data point in the testing data will be outputted into a csv file.

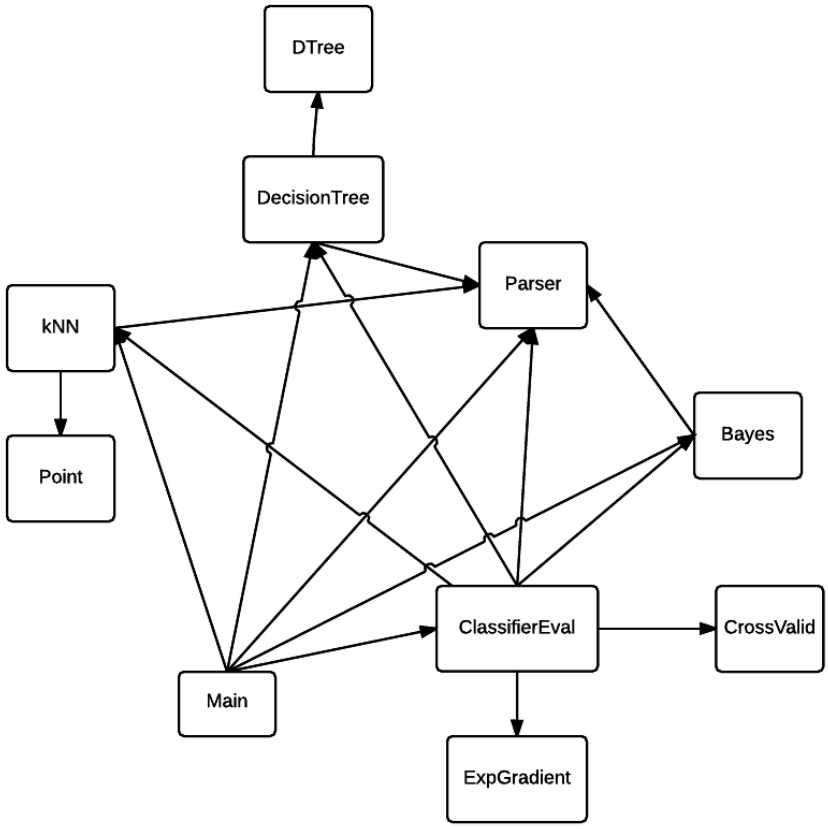


3 System Design

What are the important modules that will be implemented? What is the purpose of each module? Include a module dependency diagram (MDD).

- Parser: responsible for parsing .csv files into data instances of a record type.
- Point: data structure for kNN, provides important functions for operations on each point, each point represents one data instance.
- kNN: implementation of the kNN classifier.

- Bayes: implementation of the Bayes classifier.
- DTree: data structure for the decision tree, provides important functions for operations on constructing a decision tree.
- DecisionTree: implementation of the TDIDT algorithm.
- ExpGradient: implementation of EG algorithm which adjusts the weight of each algorithm based on their performance.
- CrossValid: implementation of 4-fold cross validation that shuffles the training data into 4 partitions of which 1 will be a testing set and the other 3 will be training sets. This is a way of evaluating the classifiers.
- ClassifierEval: assigns weight to each algorithm for testing.
- Main: takes in testing data and outputs prediction file in csv. This is the module that binds all the modules together and also the entry point for an user.



4 Module Design

Please find the zip file in our submission.

5 Data

Our system should of course maintain the training data and testing data that can be found here www.kaggle.com/c/sf-crime/data. All data points will be read in from the .csv files and parsed into instances of the same type of record. Training data and testing data will be stored into two lists. The final output file

(containing predictions for the testing data) will be in .csv format. More specific format can be found here https://www.kaggle.com/c/sf-crime/details/evaluation.

Aside from that, in kNN and decision tree's implementations, we need to have a point representation of a data point for plotting the training data and making predictions on the testing data and a tree type that represents the decision tree constructed with the training data.

6 External Dependencies

- ocaml-csv (data parsing and outputting csv files)
- sexplib (serializing records and pretty printing for debugging)

7 Testing plan

Important functions will be unit tested in a separate test module for each module (except for Main). Each team member is responsible for several modules as shown in the following list.

- Yunjie (Maggie): Parser, ClassifierEval, CrossValid, ExpGradient, Main
- Max: Point, kNN, Bayes
- Jackson: DTree, DecisionTree

We will meet at set times during the week and everyone will give an update on their progress of their parts. We will also use git as version control to keep records of all the different files.