CITS3401 Data Exploration & Mining Project 2

Classifying Poker hands using Weka

Aleck Greenham 20362627
Ash Tyndall 20915779

27th May 2013

Abstract

This document details the analysis of how different data classification algorithms handle the classification of different Poker "hands" (collections of 5 playing cards) as one of the nine well-defined classes. Where the analysis requirements [1] were ambiguous or incomplete, reasonable assumptions were made and documented.

Introduction

Limitations

Requirements

Assumptions

The following assumptions were made:

- item

1. item

Classification Method Evaluation

Accuracy

Speed

Classification Time

Robustness

Scalability

Interpretability

Implementation

Data Restructing

The originally provided data was in CSV format, while Weka only easily supports its own ARFF format. The tool linked to in the design document [2] unfortunately could not handle the large testing set (presenting error 500), so a suitable Python replacement was devised; `csv_convertor.py`.

Furthermore, the initial data produced had all of its attributes described as "numeric" in the produced ARFF format. Noting that the explanation of attributes document [3] described the "suit of card" and "class of hand" attribute as "ordina" rather than "numerical" (as the rank of card was), we elected to modify the data set so that attributes C1 through C5 and CLASS were represented in Weka as a "nominal specification" (the Weka form of an enumeration) instead.

The Python script was thusly modified to map certain numbers to representative letter combinations to assist in the interpretation of the enumeration.

Creating Smaller Training Sets

As per the assignment instructions, we devised a program that creates smaller training sets that have equivalent ratios to the larger training set. This program is described in `training_ratio_splitter.py`. The program functions by counting the number of Royal Flushes present in the larger training set (5), which is the lowest probable action, and divides the training set into a number of different smaller sets that each contain one Royal Flush.

The ratio of classes is then calculated against Royal Flushes, and a segment of each class equivalent to that ratio is placed in each file. Due to the fact that the training data set does not divide perfectly into integers, some small amount of card plays are discarded (36 of 25K). We do not anticipate this causing any issue.

Splitting Testing Set

As per the assignment instructions, we devised a program that randomly splits the test data into a configurable amount of sets with a configurable amount of elements; `test_splitter.py`.

We choose to generate with the recommanded amount of sets (10) and elements (5000).

To prevent any result ordering bias, we import the entire test set into memory, randomise it, then export the required number of elements to their own file in a subfolder.

Data Import

Classification Method Selection

Classification methods were selected from those discussed in lectures based on how suitable each was for the classification necessary to correctly identify Poker hands. The in-program documentation of the classification methods (accessibly by right-clicking on the classifier name and selecting properties from the resultant context menu) was also taken into consideration.

Bayesian Classification

The Naïve Bayesian Classifier was chosen as a base classification to be used for comparison with the other classifiers tested. The classifier is based on Bayes' Theorem:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Where:

- $P(A|B)$ is the probability that an element in a set B is also in set A
- $P(B|A)$ is the probability that an element in a set A is also in set B
- $P(A)$ is the probability that an element from the universal set is also in set A
- $P(B)$ is the probability that an element from the universal set is also in set B

When applied to classification, $P(A|B)$ becomes the probability that a record belongs to a particular class, A. Naïve Bayesian Classification calculates the probability for every record of it belonging to each of the available classes, and places it in the most probable class.

To reduce the computational load of calculating the Bayesian classification, record attributes are assumed to be independent. This is an appropriate assumption when the attributes of a playing hand are the cards that it comprises of. As the same card cannot appear more than once in any given hand, each attribute must be unique for that hand. However, each attribute may have 52 possible values, so excluding 4 other cards is a sufficiently small dependence to justify the approximation.

Decision Tree Classification

Decision tree classification algorithms iteratively divide records based on attributes selected to increase information gain, until a given threshold is reached. Decision tree

classification algorithms differ in the way they calculate information gain, and thus select attributed to use for classification.

Attributes are assumed to be, or converted to, categorical values for use with decision tree classification. This is suitable for classifying poker hands as the attributes are categorical values (face value and suite).

Gain Ratio(C4.5) J48 in Weka

Rule-based Classification

Backpropagation (Multilayer Percepeptron in Weka)

Experimentation

Initial Experiment

For our intial experiment, we decided to use the NaiveBayes, MultilayerPerceptron and the J48 algorithm to test against our full sized training set `poker_hand/training`. For the experimental parameters, we elected to cross-validate 10-fold, and repeat the number of interations of the algorithms 10 times.

References

[1] CITS3401 Data Exploration and Mining - Project 2, http://undergraduate.csse.uwa.edu.au/units/CITS3401/labs/proj2-2013.html

[2] Online CSV to ARFF conversion tool, http://slavnik.fe.uni-lj.si/markot/csv2arff/csv2arff.php

[3] Explanation of Poker Hand data attributes, http://undergraduate.csse.uwa.edu.au/units/CITS3401/labs hand.names

[4] Attribute-Relation File Format, http://www.cs.waikato.ac.nz/ml/weka/arff.html