CA4010 Data Mining Report

Group 11

Chee Kang Kong - 12369711

David Monteiro - 10364119

1. Idea and Dataset Description

We chose to work with a poker hand dataset for our Data Mining Project. The dataset was taken from the UCI Machine Learning Repository website. The dataset contains a million entries of randomly generated poker hands. To elaborate, each record is an example of five playing cards drawn from a standard deck of 52. Each card is described using two attributes, suit and rank, for a total of 10 predictive attributes. In addition, there is an attribute called class which describes the type of poker hand that the five cards make up.

	Suit of card #1	Rank of card #1	Suit of card #2	Rank of card #2	Suit of card #3	Rank of card #3	Suit of card #4	Rank of card #4	Suit of card #5	Rank of card #5	Class of Hand
Hand#1											

Suit is represented by 1-4 {Hearts, Spades, Diamonds, Clubs}.
Rank is represented by 1-13 {Ace, 2, 3, … , Queen, King}.
Class is represented by 0-9 where:

0: Nothing in Hand
1: One Pair; one pair of equal ranks
2: Two Pairs; two pairs of equal ranks
3: Three of a Kind; three equal ranks
4: Straight; five cards sequentially ranked with no gaps
5: Flush; five cards with the same suit
6: Full House; one pair + three of a kind
7: Four of a Kind; four equal ranks
8: Straight Flush; straight + flush
9: Royal Flush; {Ace, King, Queen, Jack, Ten} + flush

An important thing to note is that Suits can't really be compared, one suit is not necessarily better than the other, whereas Ranks can be compared, generally higher being better. Although bare in mind that Ace is considered the highest card, with King being second highest, and both are represented by 1 and 13 in our dataset respectively.

Our goal was to advise the player whether they should place a bet with their given hand in a game of Texas Hold Em. To do this, we will need to impose classification algorithms for each hand and to determine whether it would be safe or risky to place a bet. As the dataset currently is, the class attribute would be incorrect. This is due to the fact that an actual game of Texas Hold Em has a hand size of only two cards. Therefore, the next three cards would need to be considered as part of the "flop" (three community cards available for all other players). To achieve this, cleaning of our dataset would be necessary and will be detailed in the next few sections.

2. Data Preparation

We chose the second workshop for our presentation, which was on the topic of measuring data dispersion. We felt that this workshop suited us best due to the statistical nature of our dataset. Our goal was to examine our dataset entries to see how much they are spread out. To do this, we needed to calculate values for the mean, variance and standard deviation. Before we calculated these values, we analysed our dataset to get an overview of the results. We wrote a java program to count the occurrences of each type of hand and these were our results:

Nothing in Hand:	501,209
One Pair:	422,498
Two Pairs:	47,622
Three of a Kind:	21,121
Straight:	3,885
Flush:	1,996
Full House:	1,424
Four of a Kind:	230
Straight Flush:	12
Royal Flush:	3
Total:	1,000,000

Mean:

The mean is the average result of a set of numbers and the formula to calculate it is:

$$\overline{X} = \frac{\sum X}{N}$$

Σ x = Total value of all Hand types
N = Total number of inputs

Result:

Mean = 616,902 / 1,000,000
Mean = 0.616902

Variance:

The variance is the measure of how far each number in the set is from the mean, the formula being:

$$\sigma^2 = \frac{\Sigma(X-\mu)^2}{N}$$

$\Sigma (X - \mu)^2$ = Sum of variance of all values
N = Total number of inputs

Result:

Variance = 0.598~

Standard Deviation:

The standard deviation is a measure of how spread out numbers are. Calculating it is basically square rooting the value for average variance.
Result:

Standard Deviation = 0.773~

Data Cleaning:

In addition to calculating the dispersion, we also performed a bit of data cleaning. We felt that it was important to include this to make it apply more to an actual game of Texas Hold Em, rather than just being five cards randomly drawn from a deck.

For example, it is common to get a One Pair in five cards. However, the One Pair is only significant in a game of Texas Hold Em when it consists of at least one card from your hand. Otherwise, all other players have access to the One Pair because it's in the Flop. Therefore, taking the classifications of hand types directly from our dataset is not enough. With this in mind, we wanted to clean our dataset and classify hand types based on whether your hand interacts with the Flop.

Here are the results after the cleaning:

Nothing in Hand:	639,089
One Pair:	296,260
Two Pairs:	38,113
Three of a Kind:	18,988
Straight:	3,885
Flush:	1,996
Full House:	1,424
Four of a Kind:	230
Straight Flush:	12
Royal Flush:	3
Total:	1,000,000

We will be drawing conclusions from the calculations and data cleaning in the final section (Results and Analysis).

3. Algorithm Description

As explained earlier, we wanted to clean our dataset and reclassify the hand types so that it more accurately reflects a game of Texas Hold Em. To do this, we needed to consider the first two cards as cards in your hand, and the next three as cards in the flop. The suit and rank attributes for the hand would be stored in an array called "myHand", with the suit and rank for
the flop being stored in a "flop" array.

Example 1:

myHand = [3,12,3,2]
flop = [3,11,4,5,2,5]
class = [1]

Hand is [3,12] and [3,2] ---> Queen of Diamonds and Two of Diamonds
Flop is [3,11] [4,5] [2,5] ---> Jack of Diamonds, Five of Clubs and Five of Spades
Class is One Pair

From example 1, we can see that the One Pair occurs in the flop, so we know that pair is available to all other players. The "havePair" method is called to check if the pair is a result of at least one card in hand. If not, the hand type is reclassified as 0, which is Nothing in Hand.

Example 2:

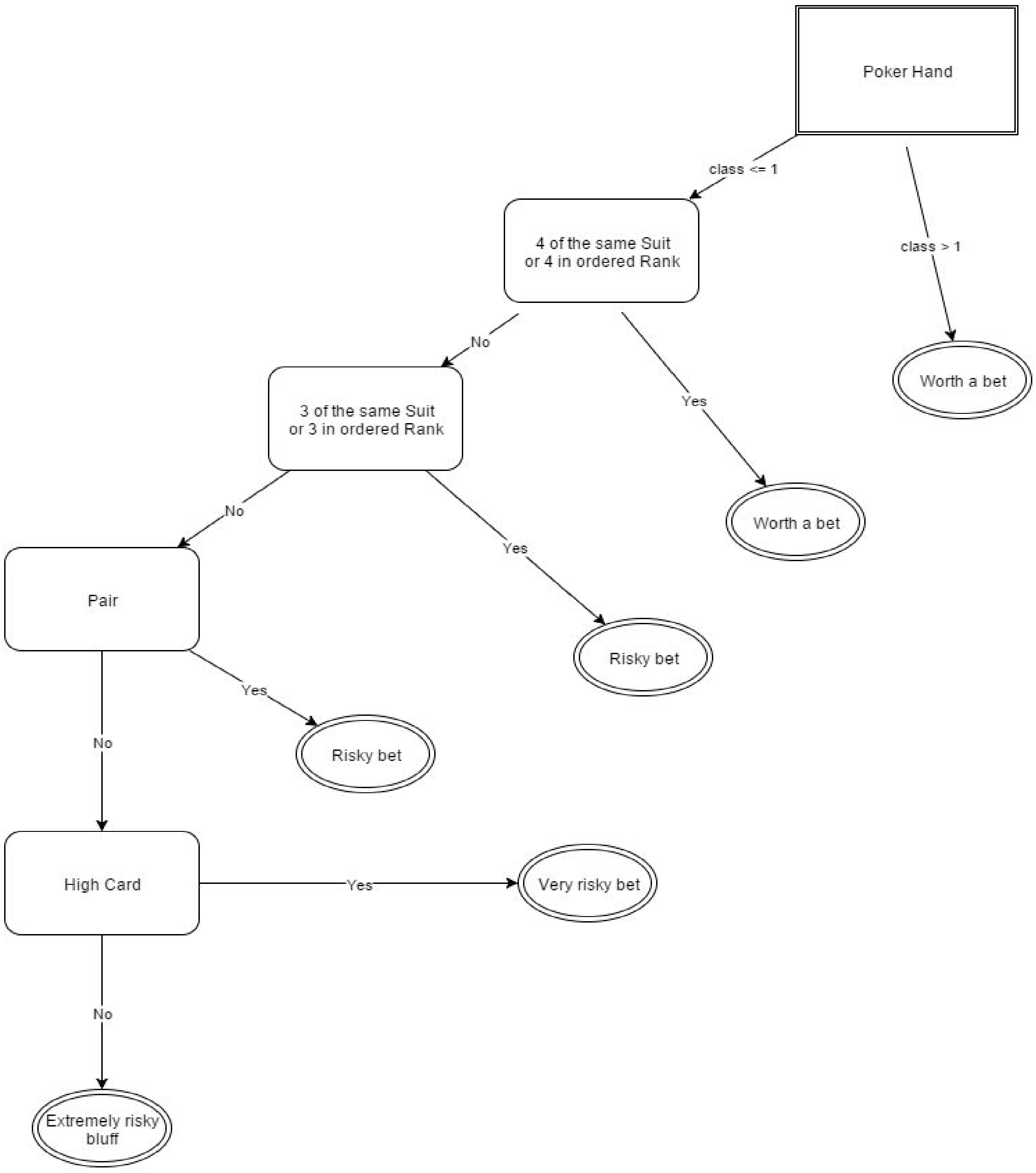
3,9,2,9,4,1,4,2,4,9,3
myHand = [3,9,2,9]
flop = [4,1,4,2,4,9]
class = [3]

Hand is [3,9] and [2,9] ---> Nine of Diamonds and Nine of Spades
Flop is [4,1] [4,2] [4,9] ---> Ace of Clubs, Two of Clubs and Nine of Clubs
Class is Three of a Kind

From example 2, the hand classification is Three of a Kind. The "haveThree" method is called which checks if at least one card in hand makes part of the Three of a Kind. In this case, it does, so no reclassification is required.

The reclassification methods only apply to the hand types 0-3 (Nothing in Hand -> Three of a Kind). This is because they are the only hand types where it does not require hand interaction, they can all occur just in the flop. The other hand types (Straight -> Royal Flush) require at least four cards to interact, which naturally includes at least one from hand.

Decision Tree:



Our basis for the first branch is that, from analysing our dataset, getting a Two Pair or better has only a 6% chance of occurring in 1,000,000 attempts. This already puts the player in a commanding position and is worth placing a bet. In the next two branches, we check if the player has a slim/good chance to get a Straight or Flush in the upcoming turns. At four ordered cards or of the same suit, the player has a very high chance of completing the Straight or Flush, so it is worth a bet. At three, there is a slim chance so the bet would be more risky. The next branch checks if the player has a pair. If yes, the bet is still regarded as risky at a 30% chance based on our data. Otherwise, go to the next branch and check for a High Card in the player's hand. At this point, any bet would be regarded as very risky as the player essentially has Nothing in Hand.

Classification Rules:

Rules
IF class > 1 THEN "Worth a bet"
IF 4 of the same Suit or 4 in ordered Rank THEN "Worth a bet"
IF 3 of the same Suit or 3 in ordered Rank THEN "Risky bet"
IF class == 1 THEN “Risky bet"
IF High Card THEN “Very Risky bet"
ELSE "Extremely risky bluff"

The classification rules above are derived from the decision tree we created. It summarises the algorithm we used to determine the risk involved in place a bet with a given hand.

Program Description:

We created a program called “RiskCalculator.java” that reads a set of attributes, these attributes are inserted in the same format as the ones from the cleaned dataset. The program implements the decision tree that contains our classification rules. Users will get feedback of whether it is safe to continue playing with the set of attributes inserted.

There are only four expected outputs:

- Worth a bet
- Risky bet
- Very Risky bet
- Extremely risky bluff

The “RiskCalculator.java” contains the following functions:

equalCards - This function takes an array of integers and an integer as parameters. The array represents all the cards from the player’s hand and the table flop. Here the array is analysed in order to decide whether or not there are two equal cards.

orderedCard - This function reads cards from the player’s hand and table, saves them into a created array that will be sorted afterwards. Using the third parameter “numApart” the function returns true if the lowest card of the array is at least X number of cards apart from the highest card.

sameSuit - This function takes one integer called “num” and two array of cards, one array represent the player’s cards and the other represents the table cards. The function extracts

the suit of all the cards and compares them. If the number of equal suits matches the "num" parameter the function will return true otherwise it will return false.
highCard - This function takes one parameter that represents the player's hand. We analyse it to see if the player has a card that is a ten, a jack, a queen, a king or an ace. If any card matches those ranks, the function will return true.

4. Results and Analysis

The result for the mean was 0.616902 before we reclassified the hand types, and 0.465247 after. Almost 94% of the 1,000,000 entries are either Nothing in Hand or One Pair, 0 and 1 respectively, so the result for the mean above is what we expected. As a follow on, we would expect to see very low values for the variance and standard deviation. This proved to be the case. The variance was 0.598~ before reclassifying, and 0.579 after. Standard deviation was 0.773~ and reduced to 0.761~ after reclassifying. These low values for variance and standard deviation point to the fact that our data is closely distributed around the mean value. As for cleaning the data itself, we saw a jump in the occurrences of Nothing in Hand, from 50% to 64%, and a decrease in occurrences of One Pair, Two Pair and Three of a Kind.

From our experience of reviewing the data and creating the decision tree and classification rules, we can see that in a real game of Texas Hold Em, it is very rare to have a hand better than One Pair within the first two rounds of betting. If a player manages to get a Two Pair or better, he/she has a very good chance of winning the pot. Even with One Pair or less, there is potential to win the pot at later stages. There are varying levels of strength within hand rankings, for example, a player who has a Pair of Aces is in a much better position than a player who has a Pair of Twos. In this case, the player with the Pair of Aces should elect to continue playing whereas the player with the Pair of Twos should take higher pairs into consideration before deciding. The classifications also cannot tell if a player is extremely close to getting a Straight or Flush. At four cards of the same suit or in ordered rank, the player has a great chance in finishing the Straight or Flush at later stages.