E6690 Final Project

Shutong Jin, Tian Yang

Classification on Student Knowledge Dataset

Overview

1. Background & Motivation:
- Problem Definition
- Baseline Work

2. Results Reproduction
- k-NN
- Genetic Algorithms

3. Further Study
- Evaluation Method
- Performance Analysis

4. Conclusion

1. Introduction

Background & baseline work

Problem Definition

User Modeling Systems(UMS)

- track and model the learning activities of students
- interested features including visited pages, time consumption, exam performance, and keystrokes.

Baseline Work

The development of intuitive knowledge classifier and the modeling of domain dependent data (Kahraman et al, 2013)

Focuses on intuition-based models

- Naive Bayes
- k-NN
- Genetic Algorithms (GA)

Dataset Information

Number of Instances: 403
Training Set: 259
Testing Set: 146

	STG	SCG	STR	LPR	PEG	y	UNS
1	0.000	0.000	0.000	0.00	0.000	0	very_low
2	0.080	0.080	0.100	0.24	0.900	3	High
3	0.060	0.060	0.050	0.25	0.330	1	Low
4	0.100	0.100	0.150	0.65	0.300	2	Middle

STG (The degree of study time for goal object materials),
SCG (The degree of repetition number of user for goal object materials)
STR (The degree of study time of user for related objects with goal object)
LPR (The exam performance of user for related objects with goal object)
PEG (The exam performance of user for goal objects)

UNS (The current knowledge of students)

Implementation

3.
Result Reproduction

The k-NN knowledge Classifier

Key parameters:

- 'k': The number of nearest neighbors
- Distance: The evaluation of distance

The majority-vote method is commonly used to determine the class of the instance sample among the k- neighbors

Used distance metrics:

$$Eucledion\ Distance; d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

m=2

$$Manhattan\ Distance; d(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n}|\mathbf{x}_i - \mathbf{y}_i|$$

m=1

$$Minkowski\ Distance; d(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{n}|\mathbf{x}_i - \mathbf{y}_i|^m\right)^{1/m}$$

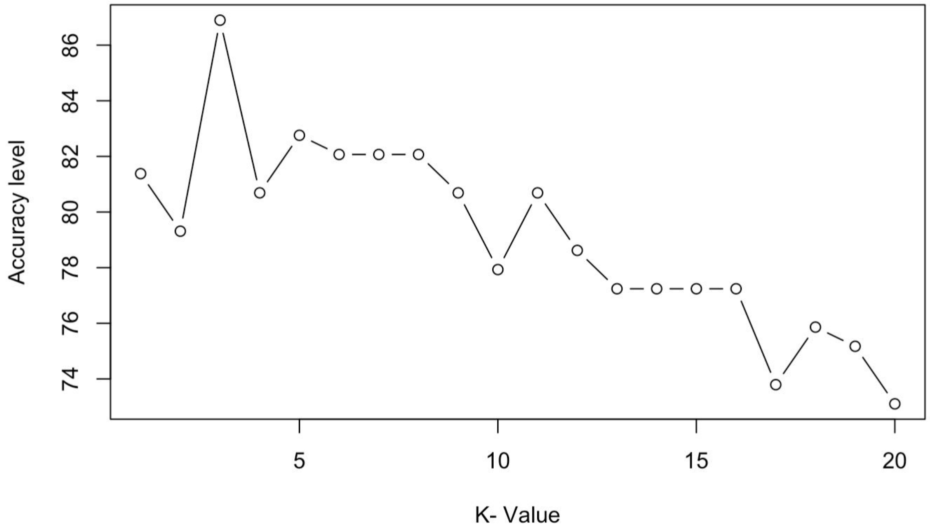
Implementation

```r
```{r}
library(class)
i=1
k.optm=1
for (i in 1:20){
  knn.mod_eu <- knn(train.input,test.input,cl,k=i)
  k.optm[i] <- 100 * sum(test.target == knn.mod_eu)/NROW(test.target)
  k=i
  cat(k,'=',k.optm[i],'
      ')
}
plot(k.optm, type="b", xlab="K- Value",ylab="Accuracy level")
```

Choice of m ?
```

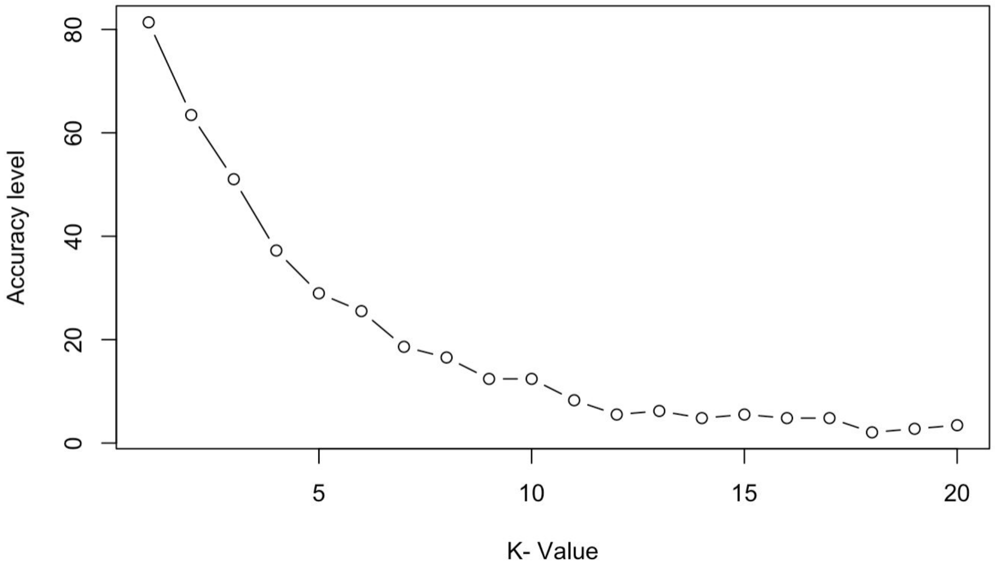
Use Euclidean Distance as metric:

k.value	Pencentages_of_Average_Error_Rates
1	20.00000
3	22.75862
5	29.65517
7	37.93103



Use Manhattan Distance as metric:

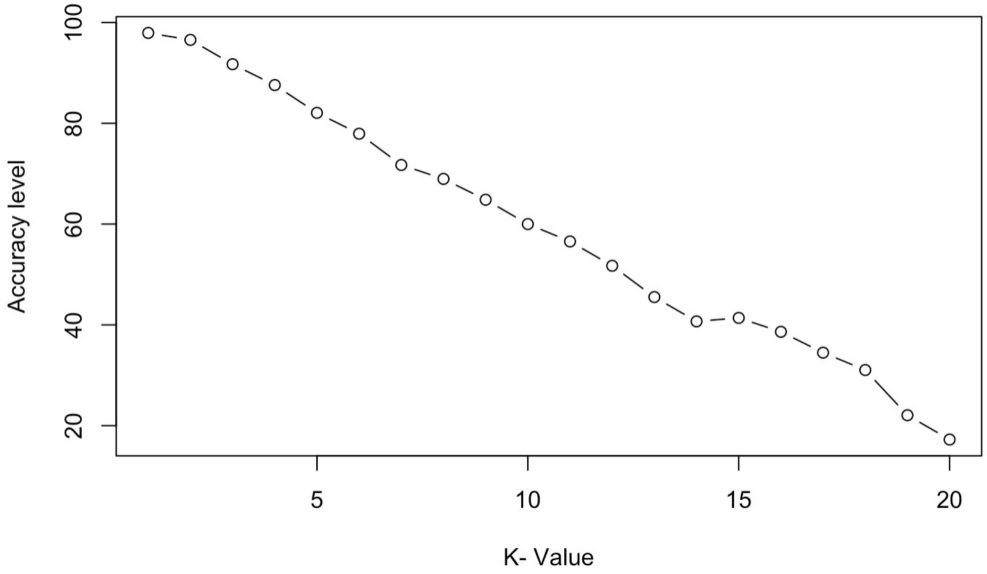
k.value	Pencentages_of_Average_Error_Rates
1	18.62069
3	48.96552
5	71.03448
7	81.37931



Use Minkowski Distance(m=0.75) as metric:

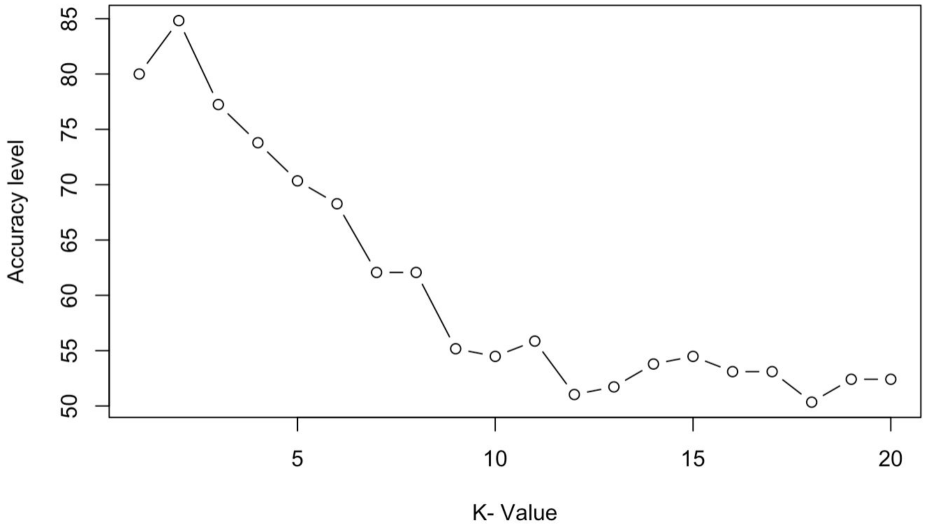
k.value	Pencentages_of_Average_Error_Rates
1	2.068966
3	8.275862
5	17.931034
7	28.275862

.

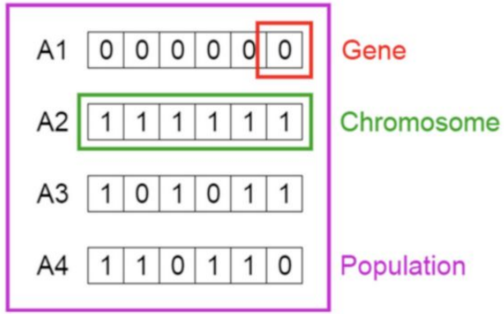


Use Minkovski Distance(m=1.414) as metric:

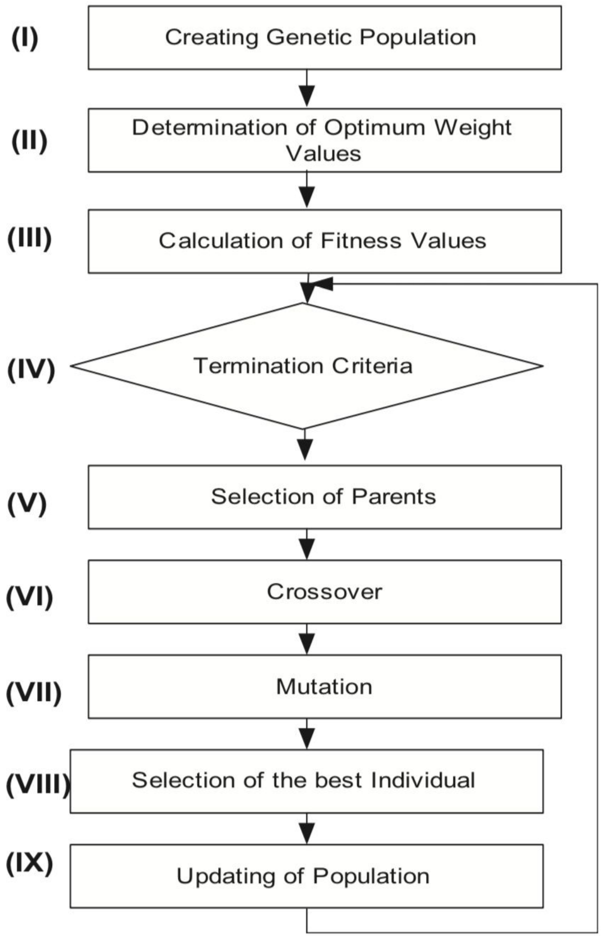
k.value	Pencentages_of_Average_Error_Rates
1	20.00000
3	22.75862
5	29.65517
7	37.93103



The Genetic Algorithm



- **Selection**: Pick up the most fitted individuals in a generation (i.e.: the solutions providing the highest ROC).
- **Cross-over**: Create 2 new individuals, based on the genes of two solutions. These children will appear to the next generation.
- **Mutation**: Change a gene randomly in the individual (i.e.: flip a 0 to 1)



Combination with k-NN methods

Fitness: How good a chromosome is

Calculation of Fitness:

1/(Number of Misclassified Instances)

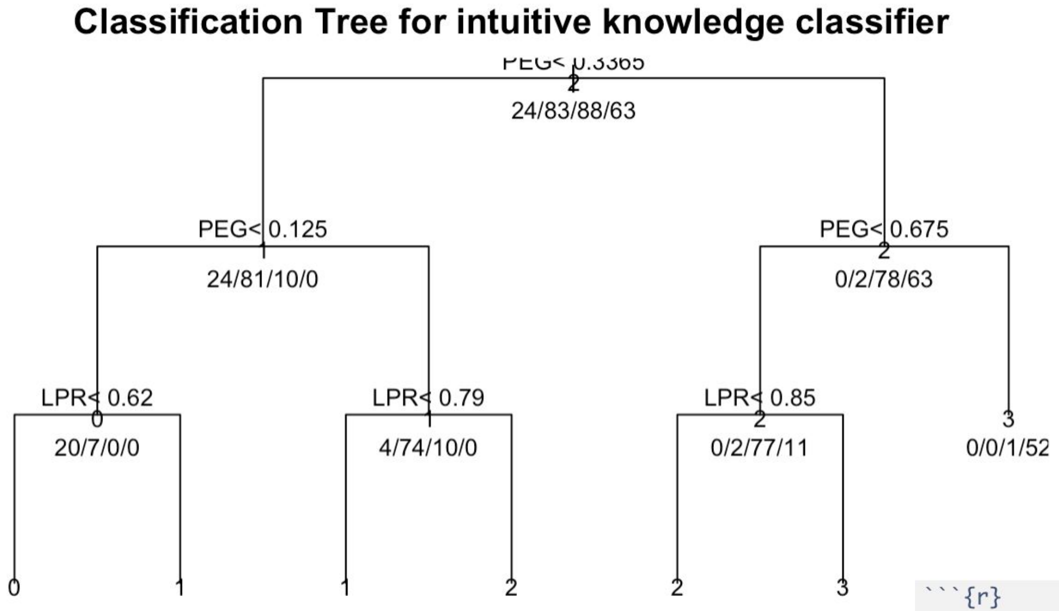
Classification results are generated by k-NN

$$\prod_{l=1}^{26} dw_{K_{uns[l]}} = \sqrt{W_{STG} * (STG_{Kuns} - STG_l)^2 + \ldots W_{PEG}(PEG_{Kuns} - PEG_l)^2}$$

3.
Further Study on the Dataset

Tree methods and comparison

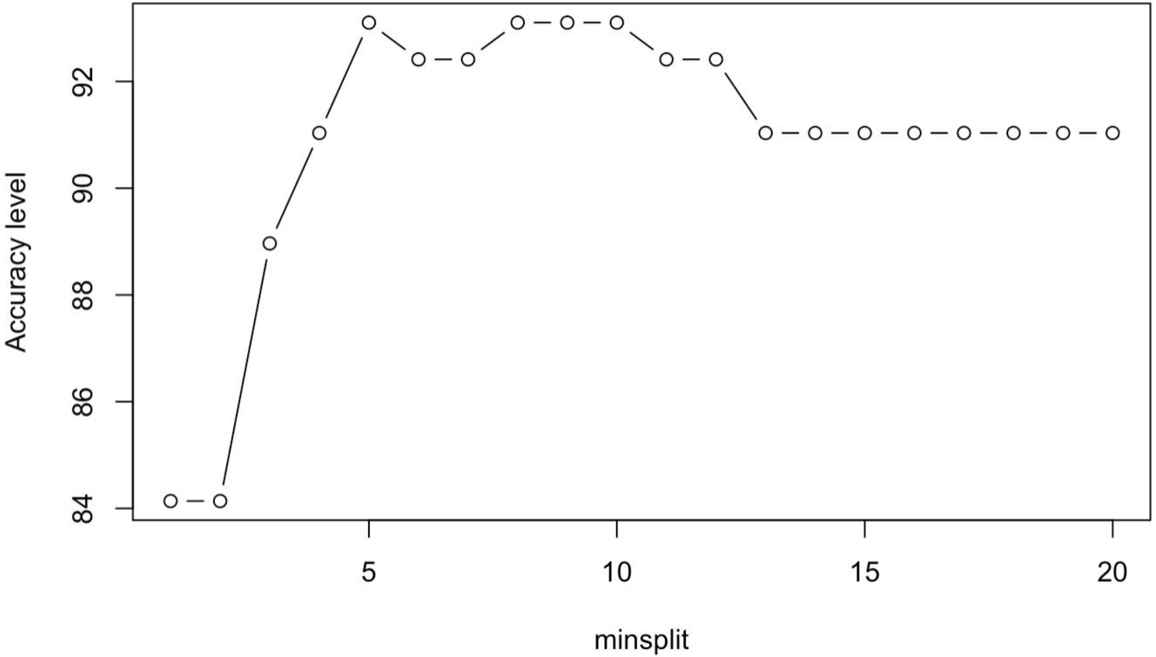
Decision Tree




```{r}
result <- predict(fit,test[1:5],type="class")
sum(test$y == result)/NROW(test$y)
```

```
[1] 0.9103448
```

Find the best minsplit




```{r}
result_[which.max(result_)]
```

```
[1] 93.10345
```

Random Forest

Confusion Matrix and Statistics

Prediction	Reference: High	Low	Middle	very_low
High	39	0	1	0
Low	0	43	3	3
Middle	0	3	30	0
very_low	0	0	0	23

Overall Statistics

Accuracy : 0.931

The highest accuracy score is obtained with a value of maxnode=10, ntrees=250,mtry=4

Maximum 10 nodes in the terminal nodes

250 trees will be trained

4 features is chosen for each iteration

Class: High	Class: Low	Class: Middle	Class: very_low
1.0000000	0.9347826	0.8823529	0.8076923

Support Vector Machine(SVM)

Using different kernels:

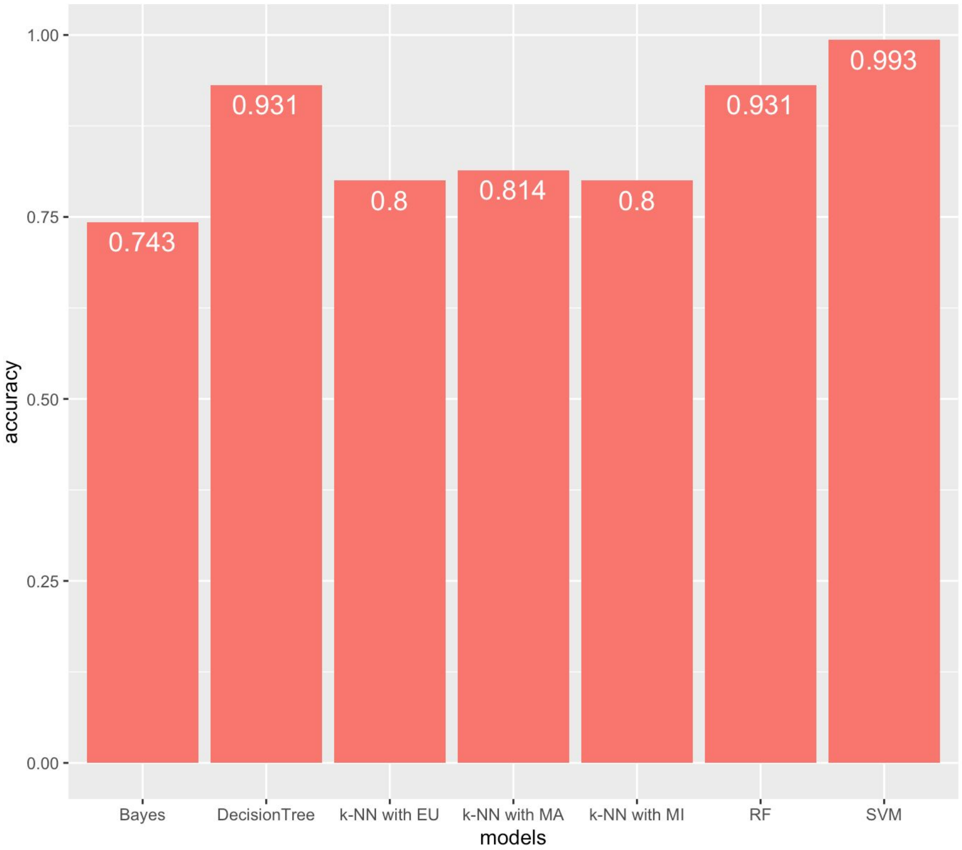
```
model1 <- ksvm(as.matrix(train[,1:5]), as.factor(train[,6]), type="C-svc", kernel="vanilladot", C=100,
scaled=TRUE)
```

```
model2 <- ksvm(as.matrix(train[,1:5]), as.factor(train[,6]), type="C-svc", kernel="anovadot", C=100,
scaled=TRUE)
```

```
model3 <- ksvm(as.matrix(train[,1:5]), as.factor(train[,6]), type="C-svc", kernel="rbfdot", C=100,
scaled=TRUE)
```

4.
Reflection & Conclusion

Accuracy comparison



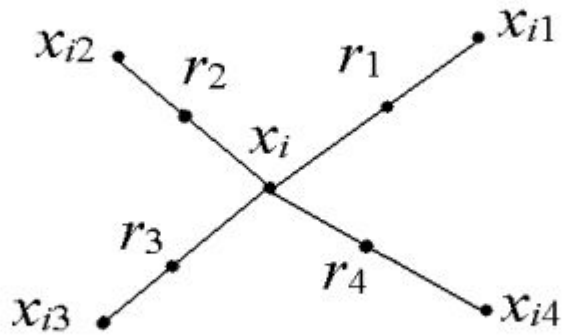
	models	accuracy
1	Bayes	0.743
2	k-NN with EU	0.800
3	k-NN with MA	0.814
4	k-NN with MI	0.800
5	DecisionTree	0.931
6	RF	0.931
7	SVM	0.993
	GA	**0.997**

Is Accuracy a Good Evaluation Metric?

```
> cm[["byClass"]][ , "Sensitivity"]
 Class: 0  Class: 1  Class: 2  Class: 3
0.1923077 0.9130435 0.6764706 0.9487179
```

The Imbalanced Dataset

Solution: Oversampling



- With SMOTE, a new sample r is created by taking a existing sample x and its nearest k samples within the same class (neighbours).

THANKS!

Any questions?

Algorithm 1. The pseudocode of GA-based weight-tuning method to explore the optimum weight values of the domain dependent data/features of users

1: ***for tt = 0 to h*** (calculate the fitness values of the individuals in $\mathbf{P}_{\mathbf{IKC[tt]}}$)

2: ***for jj = 0 to l*** (use the all observations in the set of **U**)

3: Use the Eq. (8) for EU metric and calculate the distances between the jjth observation $\mathbf{U}_{\mathbf{jj}}$ and the other observations in set **U** depending on the real-values of observations and weight values of $\mathbf{P}_{\mathbf{IKC[tt]}}$

4: Determine the class of jjth observation U_{jj} using the k-NN knowledge classifier

5: Compare the determined class of U_{jj} with real class of it

a. ***If the comparison is true then*** the fitness value of ***tt***th individual in the $\mathbf{P}_{\mathbf{IKC}[tt]}$ is increased

b. Else the fitness value of ***tt***th individual in the $\mathbf{P}_{\mathbf{IKC}[tt]}$ is decreased

6: for $qq = 0$ to generation number of population (termination criteria)

7: Select the parents

8: Achieve the crossover and the mutation operations

9: Calculate the fitness values of new individuals

10: Select most valuable individual

11: Update $\mathbf{P}_{\mathbf{IKC}}$ population

12: Finalize the searching process and save the weight values of the most suitable individual in the set of $\mathbf{P}_{\mathbf{IKC}[tt]}$