KIPS Trans. Softw. and Data Eng.
Vol.10, No.11 pp.529~534
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2021.10.11.529

A Comparative Study on Game-Score Prediction Models Using Compuational Thinking Education Game Data

Yeongwook Yang†

ABSTRACT

Computing thinking is regarded as one of the important skills required in the 21st century, and many countries have introduced and implemented computing thinking training courses. Among computational thinking education methods, educational game-based methods increase student participation and motivation, and increase access to computational thinking. Autothinking is an educational game developed for the purpose of providing computational thinking education to learners. It is an adaptive system that dynamically provides feedback to learners and automatically adjusts the difficulty according to the learner's computational thinking ability. However, because the game was designed based on rules, it cannot intelligently consider the computational thinking of learners or give feedback. In this study, game data collected through Autothikning is introduced, and game score prediction that reflects computational thinking is performed in order to increase the adaptability of the game by using it. To solve this problem, a comparative study was conducted on linear regression, decision tree, random forest, and support vector machine algorithms, which are most commonly used in regression problems. As a result of the study, the linear regression method showed the best performance in predicting game scores.

Keywords : Computational Thinking, Prediction, Game Based Learning, Regression

컴퓨팅 사고 교육 게임 데이터를 사용한 게임 점수 예측 모델 성능 비교 연구

양 영 욱†

요 약

컴퓨팅 사고는 21세기에 필요한 중요한 소양 중 하나로 여겨지면서 여러 국가에서 컴퓨팅 사고 교육 과정을 도입하여 시행하고 있다. 컴퓨팅 사고 교육 방법 중 교육용 게임 기반 방법은 학생들의 참여와 동기를 증대시키고 컴퓨팅 사고에 대한 접근성을 높여준다. Autothinking은 학습자들에게 컴퓨팅 사고 교육을 제공하기 위한 목적으로 개발한 교육용 게임으로 학습자들에게 동적으로 피드백을 제공하고, 학습자의 컴퓨팅 사고 능력에 따라서 난이도를 자동으로 조절하는 적응적 시스템이다. 하지만 규칙기반으로 게임을 디자인하여 지능적으로 학습자들의 컴퓨팅 사고를 고려하거나 피드백을 주지 못한다. 본 연구에서는 Autothikning을 통해 수집한 게임 데이터를 소개하고, 이를 활용하여 해당 게임의 적응성을 높이기 위해 컴퓨팅 사고를 반영하는 게임 점수의 예측을 수행한다. 이 문제를 해결하기 위해 회귀 문제에 가장 많이 사용되는 선형 회귀, 결정 트리, 렌덤 포레스트, 서포트 벡터 머신 알고리즘에 대한 비교연구를 수행하였다. 연구 수행결과 선형회귀 방법이 게임 점수 예측에 가장 좋은 성능을 보여주었다.

키워드 : 컴퓨팅 사고, 예측, 게임 기반 학습, 회귀

1. 서 론

컴퓨팅 사고(Computational Thinking)는 컴퓨터 과학의 추론 과정을 적용하여 문제에 대한 해결책을 디자인할 수 있는 인지능력이다[1]. 이러한 개념은 컴퓨터 과학 분야뿐만 아니라 다양한 분야에서 21세기에 필요한 중요한 소양 중 하나로 여겨지고 있다. 이에 따라서 세계 주요 국가의 교육 기관들은 1차, 2차 및 고등 교육에 학생들에게 컴퓨팅 사고를 함양하도록 하기 위한 교육 과정을 도입하여 시행하고 있다[2].

컴퓨팅 사고 교육 중 대표적인 방법은 프로그래밍 언어 또는 프로그램 명령어 블록을 사용하여 주어진 문제를 해결하게 함으로 개념화 및 논리적 사고를 학습하도록 하는 방법이다. 하지만 이러한 방법은 프로그래밍 언어의 문법 습득 및 프로그램 명령어들의 기능을 학습하게 함으로 컴퓨팅 사고의 주요 초점이 분산되고 학습 동기가 약해진다. 이로 인하여 학

※ 이 논문(작품)은 한신대학교 학술연구비 지원에 의하여 연구(창작)되었음.
† 정 회 원 : 한신대학교 컴퓨터공학부 조교수
Manuscript Received : October 12, 2021
First Revision : October 27, 2021
Accepted : October 27, 2021
* Corresponding Author : Yeongwook Yang(yeongwook.yang@gmail.com)

생에게 컴퓨팅 사고 교육에 대한 부정적 인식을 준다.

다른 방법으로는 교육용 게임 기반의 컴퓨팅 사고 교육 방법으로 학생들의 참여와 동기를 증대시키고 컴퓨팅 사고에 대한 접근성을 높여준다[3-6]. 하지만 게임 플레이 및 교육 과정에 있어서 컴퓨팅 사고에 대한 개념 및 기능에 대해서 충분히 설명하지 않거나, 학생들의 컴퓨팅 사고 측면은 고려되지 않는다. 효과적인 컴퓨팅 사고 교육을 위한 교육용 게임이 개발되기 위해서는 게임을 플레이하는 동안 학습자들에게 컴퓨팅 사고에 대한 개념이 노출되어야 하며, 학습자들의 지식 수준 및 학습 능력의 차이를 보완하기 위해 맞춤형 교육 및 피드백을 해당 시스템이 지원해야 한다.

Autothinking[7]은 학습자들에게 컴퓨팅 사고 교육을 제공하기 위한 목적으로 개발된 게임이다. 학습자들에게 동적으로 피드백을 제공하고, 학습자의 컴퓨팅 사고에 따라서 난이도를 자동으로 조절하는 적응적 시스템이다. 하지만 해당 게임이 규칙기반으로 디자인되어 지능적으로 학습자들의 컴퓨팅 사고를 고려하거나 피드백을 제공할 수 없다.

본 연구에서는 이전에 개발한 Autothinking 게임의 지능적 피드백 도입 가능성을 실험하기 위해 2019년 12월부터 2021년 4월까지 수집된 데이터를 회귀 문제에 가장 많이 사용되는 선형 회귀, 결정 트리, 렌덤 포레스트, 서포트 벡터 머신 알고리즘을 활용하여 학습자들이 획득할 수 있는 점수를 예측하는 연구를 진행하였다. 해당 게임은 컴퓨팅 사고의 개념 및 스킬을 반영하여 디자인하였으며, 게임에서 주어진 과제를 컴퓨팅 사고와 관련된 기능 및 명령어들을 활용하여 해결하는 것을 목적으로 디자인되어 어떤 기능 및 명령어들을 활용했는지, 주어진 과제를 얼마나 잘 해결했는지, 입력한 명령어의 정확하게 입력하였는지에 따라서 획득하는 점수가 달라진다. 즉, 컴퓨팅 사고에 대한 개념을 잘 이해하고, 활용할수록 높은 점수를 얻을 수 있는 구조이기 때문에 학습자들의 점수를 예측하는 것은 컴퓨팅 사고의 이해를 예측하는 것으로 이해할 수 있다.

본 연구의 기여는 첫째, 컴퓨팅 사고 교육 게임에 대한 수집 데이터 소개로 해당 도메인에 대한 관심도 증가, 둘째, 교육용 게임 데이터에 대한 기계학습 방법 적용 가능성 확대, 셋째, 기계학습 방법을 통해 게임 점수를 예측함으로 컴퓨팅 사고의 학습 정도 예측 가능성 증가이다.

논문의 구성은 다음과 같다. 2장은 해당 연구와 관련된 연구에 대해 제공하고, 3장에서는 Autothinking 게임에 대한 설명, 4장에서는 실험 및 실험 결과, 5장에서는 결론 및 제언을 제공한다.

2. 관련 연구 및 배경 지식

2.1 컴퓨팅 사고

컴퓨팅 사고에 대한 아이디어는 컴퓨터 과학에 사용된 지식과 기술이 다른 분야에 적용할 수 있다는 개념을 통해서 제안되었다. 해당 용어를 처음으로 사용한 Wing은 컴퓨팅 사고가 다른 기초 학문들과 같이 필수적으로 배워야 하는 미래 역량 중 하나라고 주장하였다[8]. 다른 연구자들의 큰 문제를 작은 문제로 나누고 관련 패턴과 변수를 식별하고 구조화된 알고리즘을 개발하여 주어진 문제에 대한 일반적인 솔루션을 만드는 것을 컴퓨팅 사고라고 주장하였다. 즉, 컴퓨팅 사고는 논리적으로 문제를 해결하는 과정과 개념을 체계화시켜 문제 해결 방법을 배우는 것이다.

컴퓨팅 사고를 연구하는 연구자들은 다양한 측면에서 컴퓨팅 사고를 체계화시켰지만, 합의된 개념은 존재하지 않는다. Wing[8]은 조건부 개념(Optional Concept), 알고리즘 개발(Algorithm development), 디버깅(debugging), 시뮬레이션(Simulation) 및 분산(Distributed Calculation)의 5가지 요소로 컴퓨팅 사고를 체계화시켰으며, Denning[9]은 조정(coordination), 커뮤니케이션(communication), 컴퓨팅(computing), 기억(recollection), 설계(design), 평가(assessment) 및 자동화(automation)를 포함하여 컴퓨팅 사고를 체계화시켰다. National Research Council 보고서[10]에 따르면 컴퓨팅 사고의 범위를 문제 분해(problem decomposition), 병렬 처리(parallelism), 디버깅, 검색 전략(search strategy) 및 시뮬레이션을 포함하는 여러 기술로 정의를 하였다. Kazimoglu외 3인[11]은 문제 해결(problem solving), 알고리즘 구축(building algorithm), 디버깅, 시뮬레이션 및 사회화(socializing)로 정의하였다. Brennan과 Resnick[12]는 CT의 개념을 순차(sequence), 반복(loop), 병렬 처리, 이벤트(events), 조건(conditionals), 연산자(operators) 및 데이터(data)로 정의하고, CT 기술을 패턴 인식(pattern recognition), 디버깅, 시뮬레이션으로 정의하였다.

이 외에도 다양한 개념 및 방법으로 정의되었지만, 관련 연구를 통하여 네 가지 기술과 세 가지 개념을 발견하였다. 기술은 문제 해결 또는 알고리즘 사고, 알고리즘 작성 디버깅 및 시뮬레이션으로 구성되며, 개념은 순차, 조건, 반복으로 구성된다. 이러한 기술과 개념은 다른 연구자들에게 다른 용어로 사용되는 경우도 있지만, 컴퓨팅 사고의 핵심 요소로써 보편적으로 받아들여지고 있다.

2.2 Autothinking

Autothinking[13-15]은 학생들의 컴퓨팅 사고를 증진시키기 위하여 저자와 연구팀이 개발한 적응형 교육 게임으로 기존의 교육용 게임들과는 달리 게임 플레이와 학습 과정에 적응력을 포함하고 있다. 이 새로운 게임 방식은 Table 1에서 표현하고 있는 것처럼 4가지 컴퓨팅 사고 스킬과 3가지 컴퓨팅 사고에 대한 개념을 게임을 플레이하면서 자연스럽게 학습할 수 있다.

게임상에서 사용자의 목표는 두 마리 고양이를 피해 미로 안에 있는 76개의 치즈를 생쥐가 수집하는 것이다. 사용자는 총 20번의 기회가 주어지고, 주어진 기회 동안 순차, 반복, 조건에 해당하는 명령어 조합과 디버깅, 시뮬레이션 기능을 활용하여 주어진 문제 상황에 대한 해결전략을 수립한다. 이

Table 1. CT Skill and Concept

CT skill	CT concept
1) Problem identification and decomposition 2) Algorithm(Pattern recognition and generalization) 3) Debugging 4) Simulation	1) Sequence 2) Condition 3) Loop

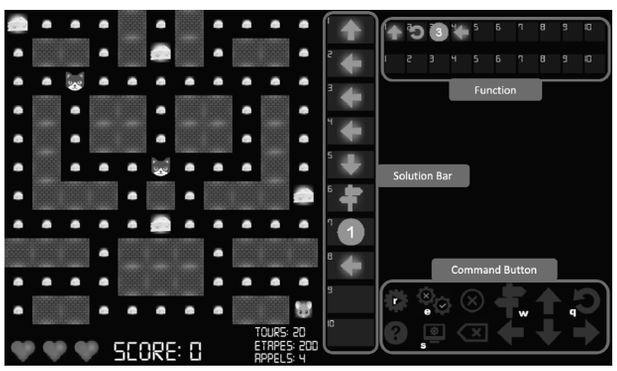

Fig. 1. Autothinking

과정을 반복하면서 사용자는 Table 1에서 제시하고 있는 컴퓨팅 사고의 스킬과 개념을 자연스럽게 학습한다.

Fig. 1은 해당 게임의 인터페이스로 Command Button창에 있는 기능들을 활용하여 Solution Bar에 명령어들을 조합하고 실행 버튼으로 명령어를 실행한다. 게임은 사용자가 조합한 명령어 조합의 완성도, 컴퓨팅 사고 개념의 이해도, 사용자의 방법의 위험도를 규칙기반의 방법으로 계산한다.

해당 게임의 적응성은 베이지안 네트워크를 통해 결정된다. 베이지안 네트워크는 해당 분야의 전문가에 의해서 모델링되며, 사용자의 플레이와 사용자의 컴퓨팅 사고의 학습 정도를 확률적으로 계산하는 것에 따라서 적응적으로 피드백과 게임 플레이의 난이도를 조절한다[13].

게임 난이도는 고양이의 움직임과 관련되어 있다. 사용자가 생성한 명령어에 따라서 무작위, 도발적, 공격적, 회피적으로 고양이를 움직인다. 무작위는 고양이가 무작위로 움직이는 것을 의미하며, 도발적은 고양이가 생쥐에게 가까이 다가가나 잡지 않고(최대 한 칸 앞까지 전진) 물러나는 것을 의미한다. 공격적은 고양이가 생쥐를 잡기위해 최단 경로로 움직인다. 회피적은 고양이가 생쥐로부터 6칸 이상 떨어지는 것을 의미한다. 즉, 게임의 난이도가 동적으로 조정된다.

사용자 명령어의 단기 및 장기적 기억에 따라서 이 알고리즘 내에서 고양이의 움직임이 결정되나, 또 다른 고양이의 경우에는 솔루션 바에 입력된 명령어의 개수에 따라서 무작위로 움직이기 때문에 고양이의 모든 움직임을 예측할 수 없다. 따라서 가능한 모든 상황을 고려하도록 사용자의 사고방식을 이끈다.

적응적 피드백은 사용자가 현재 또는 이전에 생성한 솔루션을 통해 현재 상태의 컴퓨팅 사고 학습 정도를 확률적으로 판단하고, 이에 따라서 적시에 사용자에게 필요한 피드백을 제공해준다. 시스템이 제공해주는 피드백의 형태는 텍스트와 그래픽, 동영상이다.

3. 실험 방법

전체적인 실험 절차는 Fig. 2에서 확인할 수 있는 것처럼 먼저 게임을 통해 사용자가 플레이한 로그 데이터를 Json 파일 형태로 저장하여 수집한다. 첫 번째 전처리 과정에서는 수집된 Json 파일을 파싱하여 데이터의 형태를 수치형 데이터로 변형한다. 두 번째 전처리 과정은 노이즈 필터링 및 특성 추출을 수행한다. 마지막으로 게임 점수 예측을 위한 모델 학

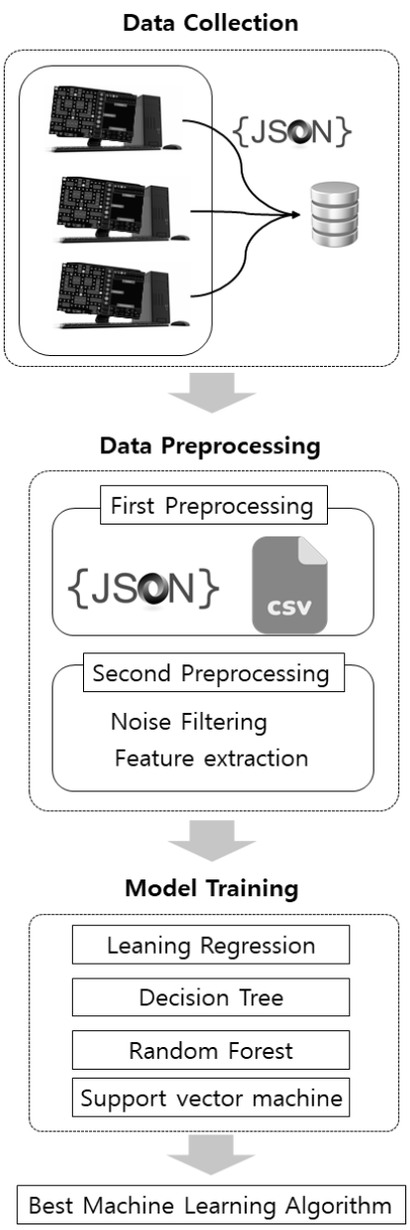

Fig. 2. Process of the experiment

습 과정 및 학습된 모델들 비교하여 최적의 기계학습 알고리즘을 찾는 과정으로 구분할 수 있다.

3.1 데이터 수집

본 연구에서 사용한 데이터는 8개의 국가에서 수집되었으며, 이상치를 제외하고 아프리카와 멕시코에서 한 명씩, 에스토니아에서 113명, 프랑스 12명, 홍콩 및 한국에서는 각각 7명, 태국 264명, 말레이시아 4명으로 총 409명에 대한 데이터이다. 여성과 남성은 각각 269명과 140명으로 구성되어 있으며, 10대 미만 17명, 10대 318명, 20대 이상 44명으로 구성되어 있다.

데이터 수집 방법은 게임 내에서 수집 가능한 정보들을 학습자가 명령어를 입력할 때마다 내부 메모리를 사용하여 저장을 한다. 이후 게임이 끝나는 시점에 서버로 데이터를 전송하고, Json 형태로 게임 데이터를 서버에 저장한다.

수집된 정보는 아이디, 이름, 나이, 성별, 국가, 생쥐 객체의 위치, 치즈 위치, 고양이1 객체의 위치, 고양이2 객체의 위치, 큰 치즈의 위치, 생명, 점수, 함수 사용, 제어문 사용횟수, 반복문 사용횟수, 디버깅 횟수, 시뮬레이션 횟수, 베이지안 확률 점수이다.

3.2 데이터 전처리

데이터 전처리에서는 먼저 서버에 저장되어 있는 Json 형태의 데이터를 테이블 형태의 데이터로 만든다. 이 과정에서 사용자의 입력에 따라 구성되어 있는 데이터를 한 번의 명령어 실행 단위로 변환한다.

두 번째 데이터 전처리에서는 데이터의 이상치 제거, 집계, 정규화를 수행하였다. 이상치 제거는 데이터의 전송 오류를 통해 데이터가 잘못 전달된 데이터들을 제거하였고, 치즈 개수, 함수 사용횟수, 제어문 사용횟수, 반복문 사용횟수, 디버깅 횟수, 시뮬레이션 횟수, 명령어 횟수, 고양이1 객체와 고양이2 객체가 움직인 횟수, 생쥐 객체가 움직인 횟수들을 집계하였다. 정규화는 각 특성이 숫자 값이고 값의 범위를 맞춰주기 위하여 z-score를 사용하였다.

최종적으로 본 연구에서 사용한 데이터 특성은 총 12개로 치즈의 수, 명령어 횟수, 실행 횟수, 움직인 횟수, 함수의 사용횟수, 생명, 방향 명령어 사용횟수, 반복문 사용횟수, 조건문 사용횟수, 생쥐 객체의 움직임 횟수, 고양이1 객체의 움직임 횟수, 고양이2 객체의 움직임 횟수이다. 모든 특성은 수치형 데이터이며, 사용자가 게임 시작부터 종료까지의 수집된 데이터들을 집계한 값이다.

추가로 실험시에 유용한 특성의 선별을 위하여 단변량 특성 선택 방법을 사용하였고, 선별된 특성들만을 활용하여 게임 점수 예측을 수행하였다.

3.3 실험 모델

본 연구 문제는 특정한 값을 예측하는 회귀 문제로 회귀 알고리즘에서 대중적으로 사용되는 선형회귀(Linear Regression), 결정 트리(Decision Tree), 랜덤 포레스트(Random Forest), 서포트 벡터 머신(Support Vector Machine, SVM) 기계학습 모델들을 선택하였다. 데이터의 양이 적기 때문에 딥러닝 기반의 알고리즘들은 고려대상에서 제외하였다.

선형회귀[16]는 종속 변수와 한 개 이상의 독립변수와의 선형 상관관계를 모델링하는 방법으로 선형 예측 함수를 사용해 회귀식을 만들고, 회귀식을 구성하는 파라메터들을 데이터로부터 추정하는 방법이다. 데이터의 특성이 많이 사용될수록 강력한 성능을 보인다.

결정 트리[17]는 독립변수들을 바탕으로 종속 변수의 값을 예측하는 트리를 생성하는 방법론이다. 트리를 구성하는 노드들은 분할 적합성을 측정하는 기준에 의해 분할되며, 각 노드에는 분할을 위한 기준 또는 조건이 정의되어 있다. 즉, 결정 트리는 이러한 기준 또는 조건들로 정의된 노드를 생성하며 트리를 구성하고, 마지막 노드(리프 노드)에 소속되어 있는 데이터들의 특성값들을 통해 값을 예측한다.

랜덤 포레스트[18]는 결정트리의 약점인 과대적합(Overfitting)될 가능성을 줄여주는 방법으로 여러 개의 결정 트리를 생성하여 새로운 데이터를 각 트리에 동시에 적용한 후 가장 좋은 예측 결과를 채택하는 방식을 말한다. 다수의 결정 트리를 생성함으로 과대적합을 예방한다.

서포트 벡터 머신[19]은 특징 공간에서 독립변수들이 종속변수를 반영하는 최상의 분리 초평면(Hyperplane)을 찾는 방법이다. 이때 주어진 데이터에서 분류하고자 하는 데이터 사이의 간격이 최대가 되는 마진(margin)을 갖도록 학습된다. 마진의 경계에 있는 값들을 서포트 벡터라고 하며, 이에 따라 마진이 달라진다. 서포트 벡터 머신에서 회귀 문제를 다루기 위해 Radial basis function(RBF) 커널을 사용하였다. 커널 함수는 데이터를 선형으로 분류할 수 없는 경우에 비선형으로 데이터를 분류하기 위하여 사용한다.

본 연구에서는 이 4가지 회귀 알고리즘을 사용하여 게임 점수 예측을 수행하였으며, 그리드 검색으로 데이터에 적합한 하이퍼파라미터를 선택하였다.

3.4 성능 평가 방법

본 연구의 예측성능을 평가하기 위하여 R^2(R-squared)를 사용하였다. R^2는 0과 1사이의 값을 가지며 1에 가까울수록 모델이 데이터에 대해 높은 연관성을 가지고 있다고 해석할 수 있다. 수식은 Equation (1)과 같다.

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}^{(i)} - \mu_y)^2}{\sum_{i=1}^{n}(y^{(i)} - \mu_y)^2} \quad (1)$$

Equation (1)에서 $\hat{y}$는 모델을 통한 예측값을 의미하고, y는 목표 데이터값을 의미하며, μ는 평균을 의미한다. 실제 값의 분산 대비 예측값의 분산 비율로 정의할 수 있다.

4. 실험 결과

각 기계학습 모델에 대한 성능평가는 정규화를 적용하지 않은 특성들을 사용한 비정규화(Non-normalization)와 정규화를 적용한 특성들을 사용한 정규화(Normalization)로 나누어 학습을 진행하였다. 실험 결과는 Table 2와 같다.

선형회귀와 결정 트리는 z-score 정규화에 영향을 받지 않아 같은 값인 0.890와 0.743의 결과를 보인다. 랜덤 포레스트의 경우 비정규화일 때 0.847의 값을 정규화를 적용했을 때 0.839의 값을 보인다. 서포트 벡터 머신의 경우 비정규화일 때 0.887의 값을 정규화를 적용했을 때 0.877의 값을 보인다. 정규화를 적용하지 않았을 때 랜덤 포레스트의 경우 0.008, 서포트벡터 머신의 경우 0.010만큼 더 높은 성능을 보인다. 가장 좋은 성능을 보인 알고리즘은 선형회귀 알고리즘으로 0.890의 성능을 보였다.

특성 선택 방법을 적용하여 게임 점수 예측에 맞는 특성 정보를 추출하고자 하였다. 특성 선택은 단변량 특성 선택 방법을 사용하였고, 그중 상위 n개의 특성을 추출하여 성능평가를 진행하였다. 실험 결과는 Table 3과 같다. 실험 결과에서 확인할 수 있는 것처럼 가장 좋은 성능을 보이고 있는 모델은 선형회귀 방법이고, 특성을 거의 다 사용하거나 모두 사용했을 때 가장 좋은 성능을 보이는 것을 확인할 수 있다. 즉, 실험에서 사용된 특성들이 모두 게임의 점수를 예측하는 것에 필요한 특성들로 해석할 수 있다.

Table 2. Experimental Results

	Linear Regression	Decision Tree	Random Forest	SVM
Non-normalization	0.890	0.743	0.847	0.887
Normalization	0.890	0.743	0.839	0.877

Table 3. Experimental Results with Feature Selection

	n	Linear Regression	Decision Tree	Random Forest	SVM
Non-normalization	2	0.716	0.648	0.707	0.714
	3	0.716	0.648	0.706	0.714
	4	0.716	0.648	0.705	0.714
	5	0.722	0.647	0.702	0.716
	6	0.760	0.664	0.729	0.756
	7	0.809	0.651	0.755	0.800
	8	0.809	0.706	0.777	0.806
	9	0.878	0.741	0.844	0.878
	10	0.880	0.736	0.840	0.848
	11	0.890	0.751	0.848	0.887
Normali-zation	2	0.716	0.648	0.711	0.714
	3	0.716	0.648	0.705	0.704
	4	0.716	0.648	0.708	0.705
	5	0.722	0.642	0.705	0.700
	6	0.760	0.669	0.720	0.765
	7	0.809	0.648	0.761	0.791
	8	0.809	0.711	0.770	0.806
	9	0.878	0.739	0.836	0.867
	10	0.880	0.736	0.845	0.841
	11	0.890	0.751	0.848	0.874

이러한 실험 결과는 데이터의 특성이 게임 점수와 선형적인 관계를 보인다는 것을 의미한다. 게임 점수가 높다는 의미는 남아 있는 치즈의 수가 적고, 명령어 횟수, 실행 횟수, 움직인 횟수, 함수의 사용횟수, 생명, 방향 명령어 사용횟수, 반복문 사용횟수, 조건문 사용횟수, 생쥐 객체의 움직임 횟수, 고양이1 객체의 움직임 횟수, 고양이2 객체의 움직임 횟수가 많다는 의미를 가진다.

5. 결론 및 향후 연구 방향

본 연구는 컴퓨팅 사고 교육을 위한 적응적 게임인 Autothinking에서 수집된 데이터에서 컴퓨팅 사고와 관련된 게임 점수를 예측하는 연구를 수행하였다. 수행결과 선형회귀 알고리즘이 비정규화한 12개의 특성들을 모두 사용하였을 때, 가장 좋은 성능을 보인다는 것을 확인하였다. 특성 선택을 수행하여 게임 점수를 예측하였을 때도 선형회귀 알고리즘의 성능이 가장 좋았으며, 모든 특성을 사용하는 것이 게임 점수 예측에 가장 좋다는 것을 확인하였다. 이 실험 결과를 통하여 선형회귀가 데이터를 가장 잘 반영하는 알고리즘이라는 것을 확인하였다.

본 연구의 한계점으로는 데이터의 양이 적기 때문에 큰 데이터셋을 필요로 하는 인공신경망 및 딥러닝 알고리즘에 적용하지 않았다는 점과 게임의 점수가 일부 컴퓨팅 사고 능력을 포함하지만 게임의 특성들과 세부적 관계분석이 미흡하다는 점이다. 또한 게임의 적응성 향상을 위해 새로운 알고리즘을 제안하여 실험군과 대조군을 명확히 할 필요성이 있다.

향후 연구 방향으로는 더 많은 양의 데이터를 수집하여 일반화된 모델을 생성하고, 게임 데이터의 특성들과 컴퓨팅 사고 능력에 대한 해석을 높이는 것이다. 해당 논문 결과인 선형 알고리즘과 또한, 더 지능적인 컴퓨팅 사고 게임으로 업그레이드시키기 위해 동적 피드백 및 명령어 추천을 위한 모델을 연구하는 것이다.

References

[1] C. Selby and J. Woollard, “Computational thinking: The developing definition,” *In Presented at the 18th Annual Conference on Innovation and Technology in Computer Science Education, Canterbury*, 2013.

[2] M. Román-González, J.-C. Pérez-González, and C. Jiménez-Fernández, “Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test,” *Computers in Human Behavior*, Vol.72, pp.678-691, 2017.

[3] H. P. Pontes, J. B. F. Duarte, and P. R. Pinheiro, "An educational game to teach numbers in Brazilian Sign Language while having fun," *Computers in Human Behavior*, Vol.107, pp.105825, 2020.

[4] J. Zumbach, L. Rammerstorfer, and I. Deibl, "Cognitive and metacognitive support in learning with a serious game about demographic change," *Computers in Human Behavior*, Vol. 103, pp.120-129, 2020.

[5] J. Asbell-Clarke, E. Rowe, V. Almeda, T. Edwards, E. Bardar, S. Gasca, R. S. Baker, and R. Scruggs, "The development of students' computational thinking practices in elementary- and middle-school classes using the learning game, Zoombinis," *Computers in Human Behavior*, Vol.115, pp.106587, 2021.

[6] C. Kazimoglu, "Enhancing confidence in using computational thinking skills via playing a serious game: A case study to increase motivation in learning computer programming," *IEEE Access*, Vol.8, pp.221831-221851, 2020.

[7] D. Hooshyar, Y. Yang, Autothikning [Internet], http://www.autothinking.ut.ee

[8] J. M. Wing, "Computational thinking," *Communications of the ACM*, Vol.49, No.3, pp.33-35, 2006.

[9] P. J. Denning, "The profession of IT Beyond computational thinking," *Communications of the ACM*, Vol.52, No.6, pp.28-30, 2009.

[10] N. R. Council, *Report of a workshop on the scope and nature of computational thinking*: National Academies Press, 2010.

[11] C. Kazimoglu, M. Kiernan, L. Bacon, and L. Mackinnon, "A serious game for developing computational thinking and learning introductory computer programming," *Procedia-Social and Behavioral Sciences*, Vol.47, pp.1991-1999, 2012.

[12] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada, Vol.1, pp.25. 2012.

[13] D. Hooshyar, H. Lim, M. Pedaste, K. Yang, M. Fathi, and Y. Yang, "AutoThinking: An adaptive computational thinking game," In International Conference on Innovative Technologies and Learning, pp.381-391, 2019.

[14] D. Hooshyar, M. Pedaste, Y. Yang, L. Malva, G.-J. Hwang, M. Wang, H. Lim, and D. Delev, "From gaming to computational thinking: An adaptive educational computer game-based learning approach," *Journal of Educational Computing Research*, Vol.59, No.3, pp.383-409, 2021.

[15] D. Hooshyar, L. Malva, Y. Yang, M. Pedaste, M. Wang, and H. Lim, "An adaptive educational computer game: Effects on students' knowledge and learning attitude in computational thinking," *Computers in Human Behavior*, Vol.114, pp.106575, 2021.

[16] G. K. Uyanik and N. Güler, "A study on multiple linear regression analysis," *Procedia-Social and Behavioral Sciences*, Vol.106, pp.234-240, 2013.

[17] B. Kamiński, M. Jakubczyk, and P. Szufel, "A framework for sensitivity analysis of decision trees," *Central European Journal of Operations Research*, Vol.26, No.1, pp.135-159, 2018.

[18] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural computation*, Vol.9, No.7, pp.1545-1588, 1997.

[19] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, Vol.20, No.3, pp.273-297, 1995.

양 영 욱

https://orcid.org/0000-0003-3219-7250
e-mail : yeongwook.yang@gmail.com
2009년 한신대학교 소프트웨어학과(학사)
2011년 고려대학교 컴퓨터교육과(석사)
2018년 고려대학교 컴퓨터학과(박사)
2021년 ~ 현 재 한신대학교 컴퓨터공학부 조교수
관심분야 : Recommender system, Educational data mining