신용카드 고객의 신용 예측을 위한 지식기반 방법들: 적용 및 비교 연구

주석진* · 김재경* · 성태경* · 김중한*

Knowledge-Based Methodologies for the Credit Rating: Application and Comparison.

Seok-Chin Chu* · Jae-Kyeong Kim* · Tae-Kyung Sung* · Joong-Han Kim*

요 약

본 연구는 백화점 고객이 신용 카드 신청 요구 시에 작성되는 가입 정보 및 사용하고 있는 고객의 거래 정보의 카드 사용 패턴으로 신용도를 예측하는 여러 방법론을 제시하고 성능을 비교하였다. 가입 정보를 분석하기 위해 역전파 신경망 (Back-Propagation Neural Network, BPNN), 사례기반추론 (Case-Based Reasoning)을, 거래 정보를 분석하기 위해 역전파 신경망과 더불어 시간지연 신경망 (Time-Delayed Neural Network, TDNN)을 각각 사용하여 그 결과를 비교하였다. 또한 전체시스템의 적중률을 높이기 위하여, ID3와 신경망을 이용한 Meta-Leaning 방법을 제안하였으며, Meta-Learning 방법과 다른 방법들을 비교, 분석을 하였다.

본 연구에서는 모형 수립과 검증을 위하여 T백화점의 실제 신용 카드 가입 고객 데이터를 이용하여 실험하였다. 데이터의 성격에 따라 각 모델의 예측력에는 차이가 나타났으나, 신경망 모형의 예측력이 우수하였으며, 시간적 특성을 고려하는 시간지연 신경회로망 모형의 예측력은 더욱 우수하게 나타났다. 또한 Meta-Learning 모형을 사용하면 예측력이 더 높아진다는 것을 확인할 수 있었다.

Key Word: 신용예측, 신경망, 시간지연신경망, 사례기반추론, ID3, Meta-Learning

* 경기대학교 경영정보학과

1. 서 론

우리 사회는 신용 거래와 정보화의 진전에 따라 개인의 무형 자산인 신용에 근거하여 각종 거래와 평가가 이루어지는 개인 신용 사회의 성숙을 맞고 있다. 그러나 이러한 신용거래의 보편화와 양적 증가와는 달리 이를 뒷받침할 수 있는 각종 제도와 관리 기술, 응용 기법 등의 발전은 그에 미치지 못하고 있다.

서구의 금융기관들의 경우 각자 고유한 신용 평가 방법을 고안해내어 이를 실무에서 활용한 지 수 십년에 이르는 역사가 있으며, 이러한 방법의 고안에는 계량적 분석 기법에 관한 다양한 연구 결과가 응용되었다 [국은경제연구소, 1992.4; 국은경제연구소, 1992.12; 김다웃, 1997; 김동기, 1982; 박무송, 1988; 조홍규, 1994; 최동만, 1993; 최순식, 1995; Capon, 1982; Meyers, et al., 1963; Standard & Poors Corporation, 1986]. 이러한 분석 및 평가 기법의 발달은 신용 관리에 따른 비용과 손실의 감소는 물론, 신용 승인시의 신속한 의사결정에 따른 관리 비용의 절감 및 소비자 사회에 신용의 중요성에 대한 의식을 뿌리내리는데 공헌 하였다. 신용을 제공하는 기업으로서는 고객의 신용도를 가능한 한 정확히 예측하고, 이에 근거하여 신용 부여나 거래 중지 등의 고객 신용 관리활동을 수행하려 한다. 만약 이러한 신용평가가 부정확하게 이루어질 경우, 장기 악성 연체나 대손과 같은 금전적인 손실과 함께 고객에 대한 관리 비용의 증가를 가져온다 [권오준, 1997].

본 연구에서 사례로 사용된, 신용카드를 주요한 거래 수단으로 이용하는 T백화점은 이러한 문제를 주로 고객에 대한 과거의 신용관리 경험에 근거한 주관적인 평점부여 방법이나, 해당 실무 전문가의 경험에 근거한 판단으로 이 문제를 해결하고 있다. 그러나 이러한 방법에 따른 신용 심사에서 신용 신청 자격을 승인받은 다수의 고객들이 악성 연체를 유발시켜 기업에 손실을 끼치고 있으며, 이것은 기존의 방법이 만족할 만한 성과를 가져오지 못함을 보여준다. 따라서 고객 신용 관리자는 기업의 이익을 증가시키고 영업상의 안전을 도모하기 위해서 양호한 신용도를 가진 카드회원을 확보하여 연체금액 증가에 따른 대손발생의 위험과 자금의 고정화에 따른 자금 수급상의 불균형을 해소하여야 한다. 그러므로 고객 신용 평가의 정확성 및 객관성을 높이기 위해서는, 실질적으로 신용불량 위험이 있는 고객을 조기에 판별하여 기업의 손실을 줄이든지 아니면 기존 고객의 신용 위험도를 합리적으로 예측하여 고객관리 의사결정의 근거로 삼을 수 있는 과학적인 신용도 평가 모형이 요구된다.

본 연구에서는 이러한 요구에 따라 고객이 신용카드를 신청할 때, 작성되는 입회 정보로부터 그 고객의 인적 특성 변수를 추출하여 이를 설명 변수로 사용하였고, 목적 변수인 신용도를 예측하는 것을 그 목적으로 삼았다. 여기서 신용도는 3개월 이상 연속 연체를 불량으로 판단하고, 2개월 이하의 연체까지는 양호로 판단한다. 그리고 이미 카드를 발급 받아 사용하고 있는 고객의 거래 정보를 이용하여 카드의 사용 패턴으로 신용도를 예측하였다. 회원 가입 시점에서의 가입 정보를 통해 고객의 입회 여부를 판단하며 거래 정보를 이용하여 이미 사용 중인 고객의 신용도를 예측하여 불량 고객을 미연에 방지함으로써 2차원적으로 불량 고객을 막을 수 있는 방법을 연구하였다.

본 연구에서는 가입 정보를 분석하기 위해 역

전파 신경망 (Back-Propagation Neural Network, BPNN), 사례기반추론 (Case-Based Reasoning)을 각각 사용하여 결과를 비교하였고, 거래 정보를 분석하기 위해 역전파 신경망과 더불어 시간지연 신경망 (Time-Delayed Neural Network, TDNN)을 각각 사용하여 결과를 비교하였다 [Waibel et al., 1989]. TDNN은 음성인식과 같은 시계열 데이터에서 특정 패턴을 인식하기 위하여 개발된 것으로 가중치 개수는 적으면서도 은닉계층(Hidden Layer)이 다수이어야 하고 시간에 무관하게 인식이 가능하여야 하는 특성을 가지고 있다. 또한 이 연구에서는 전체시스템의 적중률을 높이기 위하여, ID3와 신경망을 이용한 Meta-Leaning 방법을 제안하였으며, Meta-Learning 방법과 다른 방법들을 비교, 분석을 하였다. 이 연구에서 사용되는 각 알고리즘은 자바(Java)로 프로그래밍하였다 [Boone, 1996].

본 연구에서는 모형 수립과 검증을 위하여 T 백화점의 신용 카드 가입 고객 중에서 95년 3월, 4월, 5월부터 거래를 시작한 1,529명의 가입 신청 정보와 25개월간 거래 기록 정보를 분석 자료로 사용하였다. 2장에 연구에 사용되는 데이터를 설명하였으며, 연구에 사용되는 여러 방법들 및 각 방법과 사용되는 데이터간의 관계는 3장에서 설명하였다. 실제 데이터를 이용한 각 방법들의 결과는 4장에 설명되어있다. 또한 실험 디자인 및 결과에 대한 분석도 4장에서 설명하였다. 결론 및 추후 연구과제는 5장에서 나타나 있다.

2. 데이터 분석

2.1. 전체 데이터

본 연구의 연구 자료는 T 백화점의 신용카드 고객 중 95년 3월, 4월, 5월부터 거래를 시작한 1,529명을 대상으로 하며, 이들의 가입 정보와 25개월간의 거래 정보를 분석 자료로 사용하였다. 본 연구에서는 고객 신용도의 양호 집단과 불량 집단을 구분하는 기준으로 표본기간(25개월)동안 연속된 연체를 이용하였다. 연체가 3개월 이상인 집단을 신용도 불량 집단으로, 그렇지 않은 집단을 신용도 양호 집단으로 분류하면, 전체 고객은 신용도가 양호한 집단과 불량한 집단으로 분류되는데, 그 결과는 다음 <표1>과 같다.

〈표1〉 신용도에 따른 고객분포

	3월 고객	4월 고객	5월 고객	합계
양호 고객	282명	400명	401명	1083명
불량 고객	207명	237명	102명	446명
합계	489명	537명	503명	1,529명

2.2 가입 정보

고객이 신용카드를 신청할 때 사용되는 가입 정보를 다음 <표 2>와 같이 7개의 변수로 정리하였다.

〈표2〉 모형에 사용되는 변수

변수명	내 용
나이	0 → 23~29세, 1 → 30~39세, 2 → 40~54세, 3→55세 이상
성별	0 → 여성, 1 → 남성
직종	0 → 보통 월정 급여자, 1 → 우량 월정 급여자
결혼 여부	0 → 미혼, 1 → 기혼
주거상황	0 → 기타, 1 → 본인 소유
자동이체	0 → 이체, 1 → 이체 안함
신용도	0 → 양호, 1 → 불량 (불량고객은 3개월 이상 연체 고객)

2.3 거래 정보

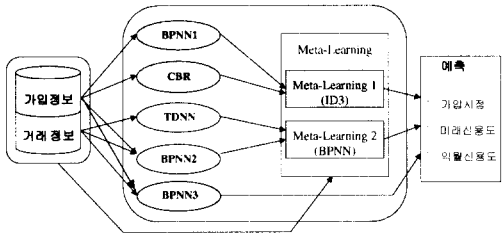
거래 정보는 가입 고객별로 25개월 동안의 매월의 매출액과 그 월의 연체 상황을 사용하였다. 25개월중 임의의 달, 혹은 임의의 달 이후의 연체여부를 예측하기 위하여, 그 직전 연속 6개월치의 데이터를 사용하였다. 6개월을 정한 이유는 현재 T 백화점에서 가입후 6개월이 지나면 과거의 매출실적이나, 연체여부를 이용하여 그 고객의 신용등급을 조정하기 때문이다. 이 연구에서는 신용등급 조정은 연구범위에서 제외하였으며, 추후 연구과제에 포함시켰다. 신용불량자로 판정하기 위하여는 3개월 이상 연속 연체를 하여야 한다. 3개월 연속 연체를 하려면, 반드시 연체 상황에 1개월 연체와 2개월 연체가 나타나게 되는데 2개월까지 연체한 고객이 다음 달에 3개월까지 연체를 할 것인지 아닌지를 판단하여야 한다. 본 연구에서는 6개월 동안의 월별 매출액을 사용하여 신용도를 파악하며 불량 고객에 대해서는 3개월 연체가 발생하기 전 6개월을 학습에 사용하였으며 양호 고객에 대해서는 무작위로 시점을 선택한 후 그 시점으로부터 과거 6개월 월별 매출액을 사용하였다. 양호 고객의 6개월간 월별 매출액 중에 마지막 2개월을 연체한 고객을 선택하여 2개월까지 연체한 고객이 3개월까지 연체할 것인가를 판단할 수 있도록 하였다. 그러므로 만약 3월에 가입한 고객은 최소한 6개월치의 데이터가 필요하므로, 9월달 이후의 연체 여부를 판단 가능하게 된다.

3. 예측 모형 시스템

본 연구에서는 분석하는 정보의 종류에 따라 가입 정보 분석 시스템과 거래 정보 분석 시스템으로 크게 나누었다. 가입정보는 고객이 카드를 신청할 때 제출하는 정보로서 예를 들면 성별, 나이, 직업, 거주형태 등을 말한다. 거래정보는 그 고객이 카드를 획득한 후, 월별 사용액에 관한 정보를 말한다. 예측하는 내용은 두 시스템 다 신용도로 같지만, 구체적으로 말하면, 가입정보를 바탕으로 가입 허가 여부를 예측 (또는 결정)하고, 거래정보를 바탕으로 그 고객의 미래 신용도 및 다음달 신용도(연체여부)를 예측하는 것이다. 각 시스템은 예측하는 시점이 다르고 그에 따른 카드회사의 대응도 달라진다. 가입 정보 분석 시스템은 가입 시점에서 얻어지는 정보를 토대로 신용도를 예측하여 새로운 고객이 가입을 원할 경우 그 고객에게 가입을 허용할 것인지를 판단하기 위한 시스템이다. 예측 시스템으로는 역전파 신경망, 시간지연 신경망, 사례기반추론을 사용하여 각 방법을 비교 분석하였다. 또한 데이터 유형에 적합한 각 방법을 결정하여 예측에 사용하는 Meta-Learning도 사용하여 다른 방법들과 예측한 결과와 실제 결과를 비교 분석하였다.

이 연구에서 사용하는 전체시스템의 개략적인 시스템 구조도는 다음 [그림 1]과 같다. 가입정보 및 거래정보는 각각의 데이터 베이스에 저장되어 있으며, 이 들 정보를 바탕으로 신용도를 예측하는 5개의 방법들을 하위 시스템을 구성하고 있다. 각 정보 및 이들 방법들의 결과를 이용하는 Meta-Learning이 또한 하위 시스템을 구성하며, 각 하위 시스템에 관한 구체적인 내용은 아래에 설명하였다. 이와 같이 가입 정보 분석 시스템과 거래 정보 분석 시스템을 하나의 시스템으로 표현한 것은 각 정보들이 서로 공유되는 부분이 있으며, 이와 같이 시스템으로 표현하는 것이 각 방법들을 비교하여 실험하기 편리하기 때문이다. 또한 결과를 바탕으로 추후

실제로 신용예측을 하기위한 시스템구축을 한다면, 이와 같은 시스템이 시작점이 될 수 있다.



〔그림 1〕 시스템 구조

3.1 가입 정보 분석 시스템

3.1.1. BPNN 1

입력으로 나이, 성별, 직종, 결혼여부, 주택 소유 여부, 자동 이체를 사용하였고 출력으로는 신용도(양호/불량)를 사용하였다. 은닉층 (Hidden Layer)의 노드 수는 3개, 활성화 함수 (Activation function)는 Sigmoid 함수를 사용하였으며, 자바로 구현하였다 [김대수, 1992; 이재규외, 1996].

3.1.2. CBR

CBR은 Neighbor의 수를 변경하며 실험을 한 결과 Neighbor의 수가 3일때가 가장 적중률이 높았다. 알고리즘은 Euclidean distance를 이용한 Nearest neighbor 방법을 사용하였으며 역시 자바로 구현하였다 [이재규외, 1996].

3.1.3. Meta-Learning 1

가입 정보에 대한 Meta-Learning은 ID3를 사용하였다. [그림 1]에 나타난 바와 같이 Meta-Learning 1은 BPNN1과 CBR의 예측 결과값과 가입 정보를 이용하여 ID3로 구현하였다. 가입정보를 입력 받아 BPNN1과 CBR 중 어느 시스템을 이용할 것인지를 결과값으로 낸다. UNIK-ID3를 이용하여 규칙 (Rule)을 도출하여 이를 자바로 코딩하여 사용하였다 [이재규외, 1996; Boone, 1996].

3.2 거래 정보 분석 시스템

거래 정보는 시계열 데이터이므로 신경망을 주로 사용하여 예측하였다 [이재규외, 1996]. 본 연구에서는 TDNN과 BPNN을 이용하였다. 각 월별 매출액은 정규화 하였다.

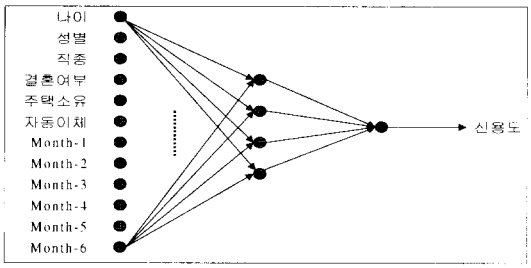
3.2.1. BPNN 2

BPNN2 모듈의 목적은 어떤 고객의 현재 시점으로부터 6개월 이전의 거래 정보를 바탕으로 다음 달에 3개월 연체를 시작할 것인가를 예측하는 것이다. 입력 노드의 수는 6, 은닉층 노드 수는 3, 출력 노드는 1, 활성화 함수는 Sigmoid 함수를 사용하였다.

3.2.2. BPNN 3

BPNN3의 목적은 어떤 고객의 현재 시점으로부터 6개월 이전의 거래 정보와 동시에 가입 정보를 보고 다음 달에 3개월 연체를 시작할 것인가를 예측하는 것이다. 학습에 사용되는 데이터는 거래 정보 중 3개월 이상 연체가 시작되기 전 6개월의 월별 매출액과 가입 정보를 사용한다. 불량 고객을 학습시키기 위해서 실제 25개월의 데이터 중 3개월 연체가 나타나기 전 6개월을 사용하였고 정상 고객에 대해서는 25개월 중 임의의 연속 6개월 데이터를 사용하였다.

입력 노드 수는 6개의 가입 정보와 6개월의 거래 정보로 12개이며 은닉층의 노드 수는 4개를 사용하였다. 활성화 함수는 Sigmoid 함수를 사용하였다.

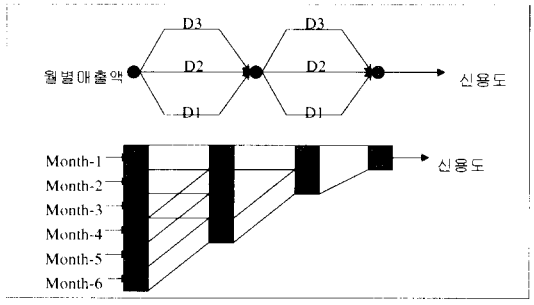


〔그림 2〕 거래 정보를 이용하는 BPNN 3

3.2.1. TDNN

[그림 2]의 BPNN3는 6개월간의 거래 정보를 보고 다음 달의 신용도를 예측하는데 목적이 있지만 TDNN은 다음 달의 신용도를 예측하는 것이 아니라 6개월의 거래 정보를 다음 달을 포함한 미래의 신용도를 예측하는 것이 목적이다. 위의 BPNN은 2개월까지만 연체를 한 고객이 다음 달에 계속 연체할 것인가에 초점이 맞추어져 있으므로 단기 예측이 된다. 따라서 Month-4, Month-5, Month-6의 값이 예측치에 많은 영향을 미치게 된다. 그러나 3개월 이상 연체를 하는 고객들의 거래 정보를 보게 되면 꼭 3개월까지 연체를 하지 않더라도 1, 2개월씩 연체를 몇 번 하고 나중에 3개월 연체를 하는 경우가 많다. 즉, 3개월 이상 연체를 할 징후를 학습하는 것이 TDNN의 목적이다. TDNN의 장점은 어느 시점에서건 특정 패턴이 나타나게 되면 이를 인식하게 된다. 6개월의 거래 정보 중 연체에 대한 징후는 초기나 중기나 후기에 아무 때나 나타날 수 있기 때문에 TDNN을 사용하였다. 그러나 TDNN을 사용하여 얻어진 예측치는 고객의 3개월 연체 잠재성을 나타내는데 그치는 한계가 있다.

TDNN의 구조는 [그림 3]과 같으며, [그림 2]에 나타난 일반 BPNN와 비교하기 바란다. 입력노드는 하나이며 은닉층은 두 개이며 윈도(window) 크기는 3이다. 윈도 크기를 3으로 정한 것은 대부분의 연체 고객의 매출 패턴이 3개월에 걸쳐 나타나는 것으로 관찰되었기 때문이다. TDNN 구현을 위하여, 역전파 알고리즘을 변형하여 다음과 같은 가중치(weight) 조정 알고리즘을 개발하였으며, 자바로 프로그래밍한 결과를 부록에 나타내었다.



〔그림 3〕 TDNN 구조

$$E_p = \frac{1}{2}(Y_p - O_p)^2$$

$$Op = f^o(net^o_p)$$

$$net^o_p = \sum_{k=1}^{2} w^o_k b_{pk}$$

$$b_{pk} = f^B(net^B_{pk})$$

$$net^B_{pk} = \sum_{j=1}^{3} w^B_j a_{p(j+k-1)}$$

$$a_{pj} = f^A(net^A_{pj})$$

$$net^A_{pj} = \sum_{i=1}^{3} w^A_i x_{p(i+j+k-2)}$$

$$w^k_o(t+1) = w^k_o(t) + \eta(Y_p - O_p) f^o(net^o_p) b_{pk}$$

$$w^B_j(t+1) = w^B_j(t) + \eta(Y_p - O_p) f^o(net^o_p) b_{pk} \sum_{k=1}^{2} w^o_k f^B(net^B_{pk}) a_{p(j+k-1)}$$

$$w^A_i(t+1) = w^A_i(t) + \eta(Y_p - O_p) f^o(net^o_p) b_{pk} \sum_{k=1}^{2} w^o_k f^B(net^B_{pk}) \sum_{j=1}^{3} w^B_j f^A(net^A_{p(j+k-1)}) x_{p(i+j+k-2)}$$

3.2.1. Meta-Learning 2

미래 시점의 신용도를 예측하는 BPNN2와

TDNN의 예측 결과값과 6개월간의 거래 정보를 이용하여 BPNN으로 구현하였다. [그림 1]에 표현한 것과 같이 6개월의 거래 정보를 입력 받아 BPNN2와 TDNN 중 어느 시스템을 이용할 것인지를 결과값으로 낸다. 6-3-1의 구조에 Sigmoid 함수를 사용하였다.

4. 실험

4.1. 데이터 분할

4.1.1. 가입 정보 분할

전체 1,529개의 데이터 중에서 1,500개를 4개의 세그먼트(segment)로 나누어 사용하였다. 세그먼트의 사용과 데이터 수는 아래 <표 3>과 같다. 데이터를 4개의 세그먼트로 나눈 이유는 각 예측 시스템의 학습에 대한 독립성을 보장하기 위한 것으로 객관적인 결과의 비교를 위한 것이다. 자세히 기술하자면, 각 방법별로 학습집단(training set)과 테스트 집단(test set)이 필요하다. 만약 Meta-learning이 존재하지 않는다면, 두개의 방법(BPNNI과 CBR)을 비교하기 위해서는 상호 교차적으로 학습시킨 다음 같은 집단(set)으로 테스트 해야 하므로 세개의 세그먼트가 필요하다. 그런데, Meta-learning1은 BPNNI과CBR의 테스트 결과를 가지고 학습하는 것이므로, 결국, BPNN1, CBR, 그리고 Meta-learning1을 위한 테스트 집단이 더 필요하다. 그러므로 4개의 세그먼트가 필요하다. 그리고 BPNN1은 6-3-1구조의 신경망이므로, 데이터는 대략 (18 +3)2 보다 커야한다 [이재규, 1996]. 그래서 세그먼트 A의 데이터는 500개로 하였고, 세그먼트 B, C는 400개씩, 그리고 테스트를 위한 세그먼트 D는 적어도 되므로200개로 하였다. 만약 데이터가 많다면, 세그먼트 A, B, C가 다 같을 수 있으나, CBR과 Meta-learning은 상대적으로 적어도 되므로 400개로 하였다.

〔표3〕 데이터 세그먼트

	Segment A(500명)	Segment B(400명)	Segment C(400명)	Segment D(200명)	합계 (1500명)
불량고객수	154 (30.8%)	120 (30%)	112 (28%)	58 (29%)	444 (29.6%)
BPNN1	학습	[N/A]	테스트	테스트	
CBR	[N/A]	Case Base	테스트	테스트	
Meta-Learning1	[N/A]	[N/A]	규칙생성	테스트	

4.1.2. 거래 정보의 분할

가입 정보의 분할과 같은 세그먼트를 사용하였다. TDNN과 BPNN2의 학습 세그먼트의 크기가 다른데 이는 TDNN의 경우 가중치의 수가 BPNN2보다는 적기 때문에 학습 세그먼트의 수가 적어도 무방하다고 보았기 때문이다.

〈표 4〉 거래 정보의 분할

	Segment A(500명)	Segment B(400명)	Segment C(400명)	Segment D(200명)	합계 (1500명)
불량고객수	154 (30.8%)	120 (30%)	112 (28%)	58 (29%)	444 (29.6%)
BPNN2	학습	[N/A]	테스트	테스트	
TDNN	[N/A]	학습	테스트	테스트	
Meta-Learning2	[N/A]	[N/A]	학습	테스트	
BPNN3	학습			테스트	

BPNN3의 경우는 모두 12개의 입력을 받기 때문에 가중치의 수가 많아 세그먼트 A~C를 모두 학습에 사용하고 세그먼트 D를 테스트에 사

용하였다. 물론 예측하는 목적이 다르기 때문에 BPNN2나 TDNN과 비교할 대상이 되지는 않지만 편의상 세그먼트 A~C를 사용하였다.

4.2 결과 및 분석

3.2.1. 가입 정보 이용 예측 결과

다음 <표 5>는 고객의 가입 정보를 이용하여 신용도를 예측한 결과이다. 각 셀에는 전체 적중률(hit ratio)과 불량 고객과, 정상 고객 각각에 대한 판별 결과가 나타나 있다. 예를 들면, 세그먼트 A를 BPNN1으로 학습하여 테스트한 결과는 전체 적중률이 76.2%이며 불량 고객 154명 중 89명을, 정상 고객 345명 중 292명을 맞게 판별하였다. 세그먼트 C를 Meta-Learning 1으로 학습한 경우는 전체 적중률은 69%이며 이를 구체적으로 나타내면 BPNN1적중한 296개 중에 184개를 적중했으며 CBR이 적중한 284개중 92개를 적중하였다. CBR보다 BPNN1의 적중률이 높은 것은 둘 다 맞는 경우에 대해서는 BPNN1의 결과를 사용하였기 때문이다.

〈표 5〉 가입 정보 이용 예측 결과

	Segment A(500명)	Segment B(400명)	Segment C(400명)	Segment D(200명)	합계 (1500명)
불량 고객수	154 (30.8%)	120 (30%)	112 (28%)	58 (29%)	444 (29.6%)
BPNN1	76.2% 89/154(.578) 292/345(.846)	[N/A]	74% 51/112(.455) 245/288(.851)	77.5% 33/58(.568) 122/142(.859)	
CBR	[N/A]	Case Base	71% 63/112(.563) 221/288(.767)	68% 34/58(.586) 102/142(.718)	
Meta-Learning1	[N/A]	[N/A]	69% 184 / BPNN1 92 / CBR	84.5% 45/58(.775) 124/142(.873)	

세그먼트 D를 보면 모든 시스템의 결과를 비교할 수 있다. CBR보다 BPNN1의 예측 적중률이 높게 나타난다. 하지만, 불량고객에 대한 적중률에서 CBR이 BPNN1보다 높은 것은, 통계학적 의미가 있는 것은 아니지만 데이터 분포에서 불량 고객 수가 적은 경우에는 BPNN1은 불량 고객을 정상고객으로 판단할 위험이 높지만 CBR은 이에 영향을 받지 않는 특성 때문인 것으로 추정된다. BPNN1과 CBR의 결과를 이용한 Meta-Learning1의 경우는 두 시스템보다 적중률이 높게 나타났다.

3.2.1. 거래 정보 이용 예측 결과

거래정보를 이용한 예측결과는 [표6]과 같으며, 각 셀에 대한 해석은 [표 5]와 마찬가지이다.

〈표 6〉 거래 정보 이용 예측 결과

	Segment A(500명)	Segment B(400명)	Segment C(400명)	Segment D(200명)	합계 (1500명)
불량 고객수	154 (30.8%)	120 (30%)	112 (28%)	58 (29%)	444 (29.6%)
BPNN2	80.6% 87/154(.565) 316/345(.916)	[N/A]	62.75% 21/112(.188) 230/288(.798)	58.5% 10/58(.172) 107/142(.753)	
TDNN	[N/A]	68% 67/120(.558) 205/280(.732)	67% 62/112(.553) 206/288(.716)	65% 27/58(.465) 103/142(.725)	
Meta-Learning2	[N/A]	[N/A]	62.5% 98 / BPNN2 152 / TDNN	61% 20/58(.345) 102/142(.718)	
BPNN3	83% 328/386(.850) 751/914(.822)			79.5% 50/58(.862) 109/142(.768)	

TDNN의 경우 BPNN2보다 나은 결과를 보이고 있다. BPNN2의 경우는 입력의 위치가 고정되어 있기 때문에 같은 패턴이 다른 입력 위치에서 발생하게 되면 같은 패턴으로 인식하지 못하지만 TDNN의 경우는 시간과 무관(Time Invariant)하게 패턴을 인식할 수 있으므로 BPNN2보다는

적중률이 높게 나타났다. BPNN3의 경우는 다음 달의 신용도를 예측한 것인데 적중률이 높게 나타나므로 2개월까지 연체한 고객에 대한 적절한 조치를 취해야 한다는 결론을 도출할 수 있다.

5. 요약 및 결론

본 연구에서는 고객의 신용도를 크게 가입 시점, 다음 달, 중장기로 구분하여 여러 방법으로 예측하였다. 많은 연구들이 가입 시점의 정보를 이용하여 신용도를 예측하는데 치중되어 있는데 이는 가입 시점에서 불량 고객을 미연에 방지하겠다는 의도이다. 그러나, 거래 정보를 살펴본 결과 많은 고객이 연체를 발생시키고 있으므로 다른 보완 대책이 필요하다고 판단되었고, 본 연구에서는 가입 시점 뿐만 아니라 고객의 사용 상황을 근거로 미래의 신용도를 예측함으로써 거래 중인 고객에 대해서도 신용도 관리를 가능하게 할 수 있다는 장점이 있다.

가입 정보를 이용하여 가입 시점에서의 신용도를 예측한 결과 BPNN1, CBR 모두 70% 이상의 적중률을 보였다. 그러나, 불량 고객에 대한 적중률은 60%를 넘지 못하고 있다. 적중률의 대부분을 정상 고객에 대한 적중률이 차지하고 있다. 따라서 가입시점의 데이터만으로 BPNN1과 CBR로 불량 고객을 예측하는 데는 무리가 있다고 할 수 있겠다. 그러나 ID3를 이용한 Meta-Learning 1을 사용한 결과 불량 고객에 대한 적중률이 77%까지 향상되었다. 또한 정상고객에 대한 적중률도 향상되었다. TDNN의 경우 BPNN2보다 나은 결과를 보이고 있다. 그 이유는 BPNN2의 경우는 입력의 위치가 고정되어 있기 때문에 같은 패턴이 다른 입력 위치에서 발생하게 되면 같은 패턴으로 인식하지 못하지만 TDNN의 경우는 시간과 무관하게 패턴을 인식할 수 있으므로 BPNN2보다는 적중률이 높게 나타났다고 볼 수 있다. BPNN1의 적중률이 떨어지는 이유는 우선 입력 변수가 수가 적어 많은 정보를 사용할 수 없었던 점과 입력되는 변수들 중에도 신용도와 밀접한 관계가 있는 변수가 적었기 때문이라고 판단된다. 이는 고객의 가입 시점에 고객이 기재해 내는 정보 중에 신용도를 판단할 수 있는 근거가 되는 변수는 없다는 결론을 의미할 수 있다. 따라서 현재의 가입 정보 이외 고객에 대한 새로운 정보를 조사하여 다시 이 방법을 사용해서 신용도에 영향이 큰 변수를 찾아 내고 이를 가입 정보에 추가하여야 하겠다. 거래 정보를 이용한 예측에 있어서 BPNN2가 상당히 저조한 적중률을 보이고 있는데 이는 임의의 6개월간 거래 정보를 이용해서는 미래의 신용도를 예측하기 어렵기 때문이라고 생각된다. 1~2개월씩 연체하는 고객이 3개월 이상 연체할 가능성이 높은데 이러한 패턴이 발생해도 그 위치가 불규칙하기 때문에 신경망이 정확하게 패턴으로 인식하지 못하기 때문으로 판단된다. 특히 초기에 카드를 사용하지 않는 고객은 정상 고객이라고 판단하지만 6개월 이외의 기간에 연체를 하는 경우도 많기 때문에 이러한 경우 예측이 정확하게 이루어지기 어렵다.

본 연구에서는 고객이 신용 카드 신청 요구시에 작성되는 가입정보 및 고객의 거래 정보를 이용한 카드 사용 패턴으로 신용도를 예측하는 것과 이를 위하여 여러 방법을 비교, 분석하는 것을 연구 목적으로 하였다. 이을 위하여 본 연구에서는 가입 정보를 분석하기 위해 역전파 신경망(BPNN), 사례기반추론(CBR)을 사용하였고, 거래 정보를 분석하기 위해서는 역전파 신경망과 더불어 시간지연 신경망(TDNN)을 사용하였

다. Meta-Leaning 기법으로는 ID3와 신경망을 사용하였다. T백화점의 실제사례를 이용하여 각 기법을 적용하였고 그 결과를 분석하였다. 또한 지금 연구중인 인터넷 환경에서 로컬 에이전트(Local agent)를 이용한 meta-learning을 활용하는 연구와 연계하기 위하여 각 알고리즘은 자바로 프로그래밍하였다.

시계열 데이터에서 시간에 관계없이 특정 패턴을 인식하는 문제에서는 기존의 BPNN 보다 TDNN이 적합할 수 있다. 이에 대한 자세한 분석 및 검증은 추후 흥미있는 연구과제이다.

참 고 문 헌

권오준, *통계적 기법과 인공신경망을 이용한 신용카드 고객 신용도 평가 모형에 관한 연구*, KAIST, 석사학위논문, 1997

국은경제연구소, *소비자금융에 있어서의 신용평가기법 및 적용에 관한 연구*, 1992.12

국은경제연구소, *신용카드산업 성장의 국민경제적 효과 분석*, 1992.4

김다웃, *신경망 분리모형과 사례기반추론을 이용한 기업 신용 평가*, KAIST, 석사학위논문, 1997

김대수, *신경망 이론과 응용*(I), 하이테크 정보, 1992

김동기, *소비자신용제도론*, 박영사, 1982

박무송, *소비자금융과 신용*, 행림출판사, 1988

이재규, 최형림, 김현수, 서민수, 주석진, 지원철, *전문가 시스템 원리와 개발*, 법영사, 1996

조홍규, *판별분석, 유사추론, 인공신경망을 이용한 도산예측*, KAIST, 석사학위 논문, 1994

최동만, *신용카드회원의 신용도 결정모형에 관한 연구*, 서강대학교 경영대학원, 석사학위논문, 1993

최순식, *신용카드시스템의 신용평가기법 적용에 관한 연구*, 숭실대 정보과학대학원, 석사학위논문, 1995

Boone, B., *C와 C++ 프로그래머를 위한 필수 자바*, (번역:장규오, 이장희), 북플러스, 1996.

Waibel, A., T. Hanzawa, G. Hinton, K. Shikano, K. J. Lang, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 37, No.3 (1989).

Capon, N., "credit scoring systems : A critical analysis", *Journal of marketing*, vol. 46(1982), 83~88

Hosmer, D, W., and Lemeshow, *Applied Logistic Regression*. New York : John Wiley and Sons, 1989

Hair , J. F.,Jr., R. E. Anderson, R. L. Tathan, W. C. Black, *Multivariate Data Analysis*, Prentice Hall, 1995

Miller, R. "Learning Financial Classifications : Stastical, Genetic, and Neural Approaches", *Computer-Aided Financial Analysis*, Addison-Wesley, 377-307

Meyers, J.H. and E.W. Forgy. "The Development o American Credit Evaluations Systems", *Journal of the American Stastical Association*, vol.58 (1963), 798~800

Standard & Poors Corporation, *Debt Ratings Criteria : Industrial Overview*, N.Y., 1986

부록 (TDNN 구현 JAVA프로그램)

```
import java.io.*;
import java.util.*;

public class tdnn{
    int s_size = 100;
    int num_input = 6;

    double W_A[] = new double[3]; // weights of hidden layer A
    double W_B[] = new double[3]; // weights of hidden layer B
    double W_O[] = new double[2]; // weights of output layer

    double X[][] = new double[s_size][num_input]; // input data
    double Y[] = new double[s_size];                    // actual output 0 or 1

    double Eta = 0.01;                          // learning rate parameter

    tdnn(){
      for(int a = 0; a < 3; a++){
          W_A[a] = -1;
          W_B[a] = -1;
      }
      for(int a = 0; a < 2; a++){
          W_O[a] = -1;
      }

      try{
      String line_str;
      double Normalized[] = new double[num_input];

      FileInputStream fi = new FileInputStream("tdnn_tr.dat");
      DataInputStream di = new DataInputStream(fi);
```

```
    for(int n = 0; n < s_size; n++){
        if((line_str = di.readLine()) != null){
            StringTokenizer st = new StringTokenizer(line_str);
            for(int m = 0; m < num_input; m++){
                X[n][m] = Double.valueOf(st.nextToken()).doubleValue();
            }
            Normalized = normalize(X[n]);
            for(int i = 0; i < num_input; i++){
                X[n][i] = Normalized[i];
            }
            Y[n] = Double.valueOf(st.nextToken()).doubleValue();
        } // end of if
    } // end of for
    di.close();
    } //end of try

    catch(IOException z){
        System.out.println("File error: " + z);
    }
} // tdnn()

public double[] normalize(double data[]){
    double maximum = 0;
    double normalized[] = new double[num_input];
    for(int i = 0; i < num_input; i++){
        if(data[i] > maximum) maximum = data[i];
    }
    for(int i = 0; i < num_input; i++){
        normalized[i] = data[i] / maximum;
    }
    return normalized;
}
public void learn(int epoch, int trainset){
    double net_out;
```

```
    double net_B[] = new double[2];
    double net_A[] = new double[4];
    double HA[] = new double[4];// hidden layer A
    double HB[] = new double[2];// hidden layer B

    double output = 0;
    double delta;

    for(int lcv = 0; lcv < epoch; lcv++){
        for(int m = 0; m < trainset; m++){
            net_A = get_net_As(X[m]);
            HA = get_As(net_A);
            net_B = get_net_Bs(HA);
            HB = get_Bs(net_B);
            net_out = get_net_out(HB);
            output = sigmoid(net_out);
            delta = get_delta(Y[m], output);
            change_wa(X[m], delta, net_A, net_B);
            change_wb(delta, net_B, HA);
            change_wo(delta, HB);
        }
    }
}

public double [] test(){
    double test_out[] = new double[30];
    for(int i = 0; i < 30; i++){
            test_out[i] = predict(X[i+70]);
    }
    return test_out;
}



public double predict(double x[]){
```

```
    double net_a[] = new double[4];
    double ha[] = new double[4];
    double net_b[] = new double[2];
    double hb[] = new double[2];
    double net_o;

    net_a = get_net_As(x);
    ha = get_As(net_a);
    net_b = get_net_Bs(ha);
    hb = get_Bs(net_b);
    net_o = get_net_out(hb);

    if(sigmoid(net_o)>= 0.5) return 1;
    else return 0;
}

public double sigmoid(double net){
    return (1 / (1 + Math.exp(-net)));
}

public double [] get_net_As(double input_X[]){
    double netA[] = new double[4];
    for(int i = 0; i < 4; i++){
        netA[i] = 0;
        for(int j = 0; j < 3; j++){
            netA[i] = netA[i] + W_A[j] * input_X[i + j];
        }
    }
    return netA;
}

public double [] get_As(double netA[]){
    double hidden_A[] = new double[4];
    for(int i = 0; i < 4; i++){
        hidden_A[i] = sigmoid(netA[i]);
```

```
    }
    return hidden_A;
}

public double [] get_net_Bs(double input_A[]){
    double netB[] = new double[2];
    for(int i = 0; i < 2; i++){
        netB[i] = 0;
        for(int j = 0; j < 3; j++){
            netB[i] = netB[i] + W_B[j] * input_A[i + j];
        }
    }
    return netB;
}

public double [] get_Bs(double netB[]){
    double hidden_B[] = new double[2];
    for(int i = 0; i < 2; i++){
        hidden_B[i] = sigmoid(netB[i]);
    }
    return hidden_B;
}

public double get_net_out(double hidden_B[]){
    double net_output = 0;
    for(int i = 0; i < 2; i++){
        net_output = net_output + hidden_B[i] * W_O[i];

    }
    return net_output;
}

public double get_delta(double y, double op){
    return ((y - op) * op * (1 - op));
}
```

```
    public void change_wo(double Delta, double hidden_B[]){
        for(int k = 0; k < 2; k++){
            W_O[k] = W_O[k] + Eta * Delta * hidden_B[k];
        }
    }

    public void change_wa(double x[], double Delta, double netA[], double netB[]){
        double Sum = 0;
        double temp1;
        double temp2;

        for(int i = 0; i < 3; i++){
            for(int k = 0; k < 2; k++){
                temp1 = W_O[k] * sigmoid(netB[k]) * (1 - sigmoid(netB[k]));
                for(int j = 0; j < 3; j++){
                    temp2 = W_B[j] * sigmoid(netA[j+k]) * (1 - sigmoid(netA[j+k]));
                    Sum = Sum + temp1 * temp2 * x[i+j+k];
                }
            }
            W_A[i] = W_A[i] + Eta * Delta * Sum;

        }
    }

    public void change_wb(double Delta, double netB[], double hidden_A[]){
        double Sum = 0;
        double temp;

        for(int j = 0; j < 3; j++){
            for(int k = 0; k < 2; k++){
                temp = W_O[k] * sigmoid(netB[k]) * (1 - sigmoid(netB[k]));
                Sum = Sum + temp * hidden_A[j+k];
            }
            W_B[j] = W_B[j] + Eta * Delta * Sum;
        }
    }
}
```