앙상블 기법을 이용한
고객의 재 구매 예측

(Prediction of Customer's Follow-on Purchase using Ensemble Methods)

지도교수 : 장 병 탁

이 논문을 공학학사 학위 논문으로 제출함.

2012년 12월 26일

서울대학교 공과대학

컴퓨터공학부

노 준 혁

2013년 2월

앙상블 기법을 이용한 고객의 재 구매 예측

(Prediction of Customer's Follow-on Purchase using Ensemble Methods)

노 준 혁

초록

온라인 상점에서의 고객들은 일반적으로 하나의 주문만을 한다. 추가 주문을 유도하기 위한 전략 중 하나로 쿠폰을 지급하는 방법이 있다. 하지만 모두에게 쿠폰을 주는 것은 경제적으로 손해를 불러일으킬 수 있다. (쿠폰이 없어도 재 구매를 할 고객들에게 쿠폰을 주는 것은 명백히 손해이다.) 그러므로 재 구매를 할 의사가 없는 고객에게 쿠폰을 주어야 한다. 본 연구에서는 고객들의 정보를 이용하여 고객들을 두 종류(재 구매하는 고객/안 하는 고객)로 나누는 다양한 모델들을 만들어보고 각각에 대해 성능을 평가해 보았다. 먼저 다양한 지도학습 기법들(로지스틱 회귀, 의사결정나무, 신경망)을 이용하여 모형을 만들어보았는데, 이 중에서 신경망 모형이 가장 좋은 성능을 보였다. 다음으로는 이번 연구에서 가장 초점을 둔 앙상블 기법들(배깅, 부스팅)을 이용하여 모형을 만들어보았는데, 그래디언트 부스팅의 경우 가장 좋은 성능을 보였다. 전체적으로 봤을 때 그래디언트 부스팅이 가장 큰 이익을 가져왔으며, 앙상블 기법을 통한 모형이 지도학습 기법을 이용한 단일 학습기에 비해 더 좋은 성능을 내는 것을 확인할 수 있었다. 향후에 변수들의 더 다양하고 신중한 범주화와 고객들의 과거 쿠폰 사용에 대한 정보가 추가된다면 더 큰 이익을 낼 수 있는 모형을 만들 수 있으리라 생각된다.

목차

1. 서론

1.1. 연구목적

올 해 온라인 쇼핑몰은 대형마트를 제치고 사상 처음 시장규모 1위에 오를 것으로 예상된다. 지난 2010년 백화점을 제치고 유통규모 2위가 된 온라인 쇼핑몰이 대형마트를 넘어 유통시장 내 최대 규모를 차지하게 되는 것이다.[1] 온라인 쇼핑몰의 특성 상 고객들의 정보를 전산화하는 것이 매우 용이한데, 이렇게 시장 규모가 급격히 커짐에 따라 쌓여가는 고객들의 정보량도 어마어마하게 커지고 있다. 이러한 '빅 데이터' 속에서 온라인 쇼핑몰 회사들은 유용한 정보를 얻어내고 이에 맞는 전략을 짜서 이익을 증진시키기 위해 노력하고 있다.

이 중 하나로 고객들에게 쿠폰을 지급하는 방법을 들 수 있다. 온라인 쇼핑몰을 이용하는 고객들의 다수는 한 번 접속하여 하나의 주문만을 한다. 판매자는 이런 고객들에게 쿠폰을 지급하여 재 구매를 유도하고자 하는데, 모든 고객에게 쿠폰을 지급하는 것은 경제적인 측면에서 효율적이지 못하다고 할 수 있다. 그 이유는 쿠폰을 주지 않아도 재 구매를 할 고객들이 있기 때문이다. 즉 고객들을 쿠폰을 주지 않아도 재 구매를 할 고객과 그렇지 않은 고객으로 나눌 수 있다면 후자에게만 쿠폰을 주는 것이 더 효율적이라고 볼 수 있다. 이번 연구의 목적은 고객들의 정보를 바탕으로 이와 같이 고객들을 두 부류로 나누는 것이 되겠다.

1.2. 연구내용

본 연구는 실제 데이터로서 'DMC 2010 Competition'에서 제공된 문제를 선택하여 진행한다.[2] 이 데이터는 온라인 미디어(media) 쇼핑몰의 다양한 고객 정보를 담고 있는데, 이를 이용하여 고객들을 앞에서 설명한 두 부류로 분류해보고자 한다. 출력 값이 2가지로 명확히 정해져 있는 만큼 다양한 지도 학습기법(supervised learning)들을 통해 모형을 구축해보고, 최종적으로는 이러한 단일 학습기들을 이용한 앙상블 기법(ensemble method)들을 이용해 더 좋은 모형을 만들어보고자 한다.

1.3. 논문의 구성

논문의 구성은 다음과 같다. 먼저 분류 문제를 푸는데 많이 쓰이는 지도학습 기법들에 대해 알아보고, 이들 보다 나은 성능을 내기 위한 앙상블 기법에 대해 알아보는데 왜 더 좋은 성능을 내는지에 대해서도 다뤄본다. 그리고 연구를 통해 해결하고자 하는 문제를 정확히 파악하고 문제를 어떤 과정을 통해 해결해 나갈 것인지 제시한다. 다음으로 실험을 통해 그 결과를 살펴보고 분석한 뒤, 마지막으로 다시 한 번 요약하고 향후 과제에 대해 논의해 보도록 하겠다.

---

1) "온라인쇼핑몰 매출, 올해 대형마트 제친다.", ETNews, 2012/12/09

2) DMC는 'Data Mining Cup'의 약자로 'prudsys AG'라는 데이터마이닝 회사에서 매년 주최하는 국제대회이다.

2. 지도학습 기법

분류 문제에서 다양한 지도학습 기법들(supervised learning methods)이 많이 쓰이는데, 이번 파트에서는 이 기법들 중 본 연구에서 이용할 로지스틱 회귀, 의사결정나무, 신경망 모형에 대해 알아보도록 하겠다.[1]

2.1. 로지스틱 회귀 (logistic regression)

출력 변수가 범주 형 변수인 경우(분류문제)에 사용하는 회귀모형 중 하나이다. 출력 변수가 이항 범주인 경우 다중 로지스틱 회귀모형은 입력변수 x에 대하여

$$P(Y=1|x) = p(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)$$

이다. 여기서 $p(x) = \exp(x)/(1+\exp(x))$[3]이고 모수 $\beta_0, \cdots, \beta_p$는 최대 우도 법으로 추정될 수 있으며 유의성 검정은 우도 비 검정을 이용한다. 입력변수가 범주형인 경우에도 선형회귀와 동일한 방법으로 가변수를 잡아서 분석할 수 있으며, 변수선택도 선형모형의 방법을 적용할 수 있다. 단, 선택기준은 선형회귀모형에서 사용하는 오차제곱 합 대신에 로그 우도함수 값을 사용한다. 예를 들어 전진선택법에서는 각 단계마다 로그 우도함수 값의 증가량이 가장 큰 변수를 추가하고 AIC 또는 BIC 등의 선택기준을 최소화하는 모형을 그 최종모형으로 선택한다.

로지스틱 회귀는 주어진 입력변수 x에 대하여 출력변수 Y가 1이 될 확률 P(Y=1|x)를 추정하는데, 0과 1사이의 적당한 수 c를 절단 값으로 선택하여, P(Y=1|x)가 c보다 크면 자료를 Y=1인 클래스로 분류하고 P(Y=1|x)가 c보다 작으면 자료를 Y=0인 클래스로 분류할 수 있다.

이때 절단 값 c의 결정은 다음과 같은 사항을 고려하여 결정한다. 첫째, 사전정보를 고려한다. 사전정보에 의하여 두 번째 범주의 자료(y=1인 자료)가 많이 나타난다면 절단 값을 작은 값으로 정할 수 있다. 둘째, 적절한 손실함수를 고려한다. 두 번째 범주의 자료를 잘못 분류하는 손실이 첫 번째 범주의 자료를 잘못 분류하는 것에 비하여 손실 정도가 심각하게 큰 경우에는 절단 값 c를 작게 잡을 수 있다. 그 밖에 전문가 의견이나 민감도와 특이도 등을 고려하여 c값을 결정할 수 있다.

2.2. 의사결정나무 (decision trees)

의사결정나무는 지도학습 기법으로 각 변수의 영역을 반복적으로 분할함으로써 전체 영역에서의 규칙을 생성한다. 의사결정나무의 예측력은 다른 지도학습기법들에 비해 대체로 떨어지나 해석력이 좋다. 즉, 의사결정나무에 의하여 생성된 규칙은 if-then 형식으로 표현되어 이해가 쉽고 SQL(structured query language)과 같은 데이터베이스 언어로 쉽게 구현되는 장점이 있다.

의사결정나무의 형성과정은 크게 성장(growing), 가지치기(pruning), 타당성 평가, 해석 및 예측으로 이루어진다. 성장 단계는 각 마디에서 적절한 최적의 분리 규칙을 찾아서 나무를 성장시키는 과정으로서 적절한 정지규칙을 만족하면 중단한다. 가지치기 단계는 오차를 크게 할

3) $p(x) = \exp(-\exp(x))$이면 검벨(Gumbel) 모형, $p(x)$가 표준정규분포의 분포함수인 경우를 프로빗(probit) 모형이라 부른다.

위험이 높거나 부적절한 추론규칙을 가지고 있는 가지 또는 불필요한 가지를 제거한다. 타당성 평가단계에서는 이익도표(gain chart), 위험도표(risk chart), 혹은 시험자료를 이용하여 의사결정나무를 평가하게 된다. 해석 및 예측 단계에서는 구축된 나무모형을 해석하고 예측모형을 설정한 후 예측에 적용한다.

의사결정나무는 출력변수가 연속형인 회귀나무(regression tree)와 범주형인 분류나무(classification tree)로 나눌 수 있다. 의사결정나무의 각 마디에서 분리변수와 분리기준은 목표변수의 분포를 가장 잘 구별해주는 쪽으로 정해야 하는데, 이에 대한 측정치로 불순도를 사용한다. 불순도의 측도로 회귀나무의 경우에는 흔히 오차제곱 합을 사용하고, 분류나무의 경우에는 카이제곱 통계량, 지니 지수(Gini index), 엔트로피 지수(entropy index) 등을 사용한다.

2.3. 신경망(neural networks)

신경망은 생물학적 신경망의 구조로부터 착안된 학습 알고리즘이다. 다음 그림은 다층신경망의 구조를 보여준다.

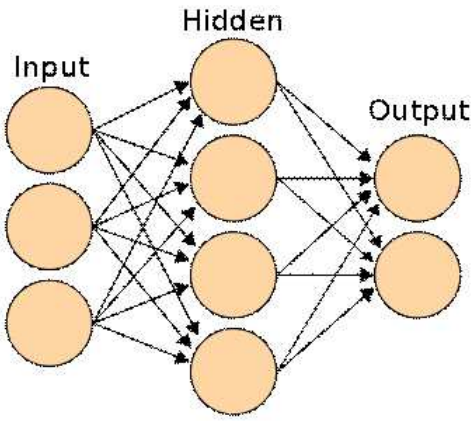


그림 1. 다층신경망의 구조

다층신경망은 입력 층(input layer), 은닉 층(hidden layer), 출력 층(output layer)으로 구성된다. 단층신경망은 은닉 층이 없이 입력 층과 출력 층만으로 구성된다. 입력 층은 각 입력변수에 대응되는 노드로 구성되며 노드의 수는 입력변수의 개수와 같다. 은닉 층은 입력 층으로부터 전달되는 변수 값들의 선형결합을 비선형함수로 처리하여 출력 층 또는 다른 은닉 층에 전달하는 역할을 하며, 출력 층은 출력변수에 대응되는 노드로서 분류모형에서는 클래스의 수만큼 의 출력노드가 생성된다.

신경망은 의사결정나무처럼 회귀와 분류문제를 모두 다룰 수 있는데, 여기서는 클래스의 수가 K인 분류 문제를 생각해 보자. 출력노드 $k(= 1, \cdots, K)$는 클래스 k에 속할 확률을 모형화하며 출력변수는 자료가 k번째 클래스에 속하는 경우 k번째 좌표는 1이고 나머지는 0으로 코딩된다.

은닉 노드 값 z_m은 입력노드 값들의 선형결합이고 출력 값은 z_m들의 선형결합 t_k들의 함수로 다음과 같이 모형 화 한다.

$$z_m = \sigma(\alpha_{0m} + \alpha_m^T x),\ m = 1, \cdots, M$$
$$t_k = \beta_{0k} + \beta_k^T z,\ k = 1, \cdots, K$$
$$f_k(x) = g_k(t),\ k = 1, \cdots, K$$
$$z = (z_1, \cdots, z_M)^T,\ t = (t_1, \cdots, t_M)^T$$

여기서 $\sigma(\bullet)$는 활성함수(activation function)라 부르며 흔히 시그모이드(sigmoid)함수를 사용한다.[4] $g_k(t)$는 출력함수(output function)라 부르고 출력 값 t에 대하여 최종적인 비선형 변환을 해주는 역할을 한다.

미지의 모수 $\alpha_{0m}, \alpha_m, m = 1, \cdots, M$과 $\beta_{0k}, \beta_k, k = 1, \cdots, K$의 벡터를 θ로 나타내자. 분류문제에서는 오차 제곱 합 또는 deviance

$$R(\theta) = -\sum_{k=1}^{K}\sum_{i=1}^{n} y_{ik} \log f_k(x_i)$$

를 비용함수로 사용하며 $G(x) = \arg\max_k f_k(x)$를 이용하여 분류한다. $R(\theta)$의 비선형성으로 인하여 전역 최소 값(global minimizer)을 찾는 것은 거의 불가능하다. 그 대신 벌점 항을 이용한 기울기 강하 알고리즘과 명시적인 벌점 화 또는 알고리즘의 조기 종료(early stopping) 등의 간접적인 벌점 화를 결합하여 좋은 국소 최소 값을 구하게 된다. 신경망에서는 $R(\theta)$를 최소화하기 위하여 역전파라 불리는 기울기 강하(gradient descent) 알고리즘을 적용한다.

---

4) 시그모이드 함수는 단극성과 양극성의 두 종류가 있다. 단극성 시그모이드 함수는 $\sigma(x) = \frac{1}{1+e^{-x}}$로 정의되고, 양극성 시그모이드 함수는 $\sigma(x) = \frac{1-e^{-x}}{1+e^{-x}}$로 정의된다.

3. 앙상블 기법을 이용한 지도학습 성능 향상

앞에서 소개한 일반적으로 잘 알려진 알고리즘인 지도학습 기법들을 통한 모형보다 더 좋은 성능을 내기 위한 알고리즘으로 앙상블 기법이라는 것이 존재한다. 이름 그대로 많은 기저 학습기들을 합치는 방법으로, 모형을 해석하기는 어렵지만 매우 좋은 성능을 낸다는 장점을 갖고 있다. 이번 파트에서는 앙상블 기법들 중 배깅과 부스팅에 대해 알아보고 왜 이 알고리즘들이 단일 학습기들에 비해 좋은 성능을 내는지 논의해 보도록 하겠다.

3.1. 배깅(bagging)

배깅은 Bootstrap **Agg**reat**ing**의 준말로서 주어진 자료에 대하여 여러 개의 붓스트랩(bootstrap) 자료를 생성하고 각 붓스트랩 자료에 예측모형을 만든 후 결합하여 최종 예측모형을 만드는 방법이다. 여기서 붓스트랩 자료란 주어진 자료로부터 동일한 크기의 표본을 랜덤 복원추출로 뽑은 자료를 말한다.

$\mathcal{L} = (x_i, y_i)_{i=1}^n$은 훈련자료를 나타낸다고 하자. 배깅 알고리즘을 정리하면 다음과 같다.[5]

⑴ B개의 붓스트랩 자료 $\mathcal{L}^{*(b)}, b = 1, \cdots, B$를 만든다.

⑵ 각 붓스트랩 자료 $\mathcal{L}^{*(b)}$에 대해서 예측모형 $f^{(b)}(x)$를 구축한다.

⑶ B개의 예측모형을 결합하여 최종 모형 $\hat{f}$을 만든다. 최종모형을 만드는 방법은

⒜ 회귀모형인 경우 $\hat{f}(x) = \sum_{b=1}^{B} f^{(b)}(x)/B$와 같이 평균을 취한다.

⒝ 분류모형인 경우 $\hat{f}(x) = \arg\max_k (\sum_{b=1}^{B} I(f^{(b)}(x) = k))$와 같이 투표를 한다.

이번에는 왜 배깅이 예측력을 크게 향상시킬 수 있는지에 대하여 알아보도록 하자.[8] 주어진 훈련자료 $\mathcal{L}$을 이용하여 구축된 예측모형 $\hat{f}(x)$는 $\mathcal{L}$에 의존한다. 이를 강조하기 위하여 $\hat{f}(x) = f(x, \mathcal{L})$이라고 쓰자. 주어진 예측모형 $f(x, \mathcal{L})$에 대하여 평균예측모형 $f_A(x)$를 $f_A(x) = E_{\mathcal{L}} f(x, \mathcal{L})$로 정의한다. 여기서 기댓값은 훈련자료가 얻어진 모집단의 분포를 이용하여 구한다는 점에 유의해야 한다. 다음의 증명은 평균예측모형의 기대손실이 단일 예측모형의 기대손실보다 항상 작다는 것을 보여준다.

(X, Y)를 $\mathcal{L}$과 독립인 미래의 관측 값이라 하자. 제곱손실함수 $L(y, a) = (y-a)^2$에 대하여 $f(x, \mathcal{L})$과 $f_A(x)$의 기대손실 R와 R_A를 다음과 같이 정의한다.

$$R = E_{(X,Y)} E_{\mathcal{L}} L(Y, f(X, \mathcal{L})), \quad R_A = E_{(X,Y)} L(Y, f_A(X))$$

제곱함수는 볼록함수이므로 Jensen 부등식에 의해서

$$E_{(X,Y)} E_{\mathcal{L}} f^2(X, \mathcal{L}) \geq E_{(X,Y)} f_A(X)^2$$

이 성립한다. 따라서

$$
\begin{aligned}
R &= E_{(X,Y)}[Y^2] - 2E_{(X,Y)}[YE_{\mathcal{L}} f(X,\mathcal{L})] + E_{(X,Y)}E_{\mathcal{L}}[f^2(X,\mathcal{L})] \\
&\geq E_{(X,Y)}[Y^2] - 2E_{(X,Y)}[Yf_A(X)] + E_{(X,Y)}[f_A(X)^2] \\
&= E_{(X,Y)}[(Y - f_A(X))^2] = R_A
\end{aligned}
$$

위의 증명에서 중요한 사실 하나를 확인할 수 있는데, $R - R_A$는

$$
E_{(X,Y)}\left\{E_{\mathcal{L}} f^2(X,\mathcal{L}) - (E_{\mathcal{L}} f(X,\mathcal{L}))^2\right\} = E_{(X,Y)}(Var_{\mathcal{L}} f(X,\mathcal{L}))
$$

이다. 즉, $f(X,\mathcal{L})$의 분산(또는 불안정성)이 크면 평균예측모형이 원래의 예측모형을 크게 향상시키며, 반대로 분산이 작으면(또는 안정적이면) 평균예측모형의 예측력의 향상 정도가 줄어든다.

훈련자료를 얻은 모집단의 분포를 모르기 때문에 실제문제에서는 평균예측모형을 구할 수 없다. 그 대신 학습 자료를 모집단으로 생각하고 이로부터 평균예측모형을 구한 것이 바로 배깅의 예측모형이라 할 수 있다. 배깅은 결국 주어진 예측모형의 평균예측모형을 구하는 것이고 이를 통하여 분산을 줄여줌으로써 예측력을 향상시킨다. 바꿔 말하면 배깅은 예측모형의 편의(bias)에는 영향을 미치지 않고 분산에만 영향을 미친다. 따라서 배깅을 적용하기에 적합한 예측모형은 편의가 없고 분산이 큰 모형이다. 일반적으로 과대 적합 된 모형이 편의가 작고 분산은 크다. 의사결정나무에 배깅을 적용할 때 나무를 최대로 성장시키고 가지치기를 하지 않는 것도 과대적합을 통해 배깅의 효과를 극대화하기 위함이다.

3.2. 부스팅(boosting)

부스팅의 기본 아이디어는 예측력이 약한 예측모형(weak learner)들을 결합하여 강한 예측모형을 만드는 것이다. 여기서 약한 예측모형이란 랜덤하게 예측하는 것보다 약간 좋은 예측력을 지닌 모형을 말한다. 반면 강한 예측모형이란 예측력이 최적에 가까운 예측모형을 지칭한다. 실제 자료 분석을 위해 제안된 최초의 부스팅 알고리즘은 이진 분류문제에서 Freund와 Schapire에 의해서 개발된 아래의 AdaBoost(Adaptive Boost) 알고리즘이다.[9]

⑴ 가중치 $w_i = 1/n,\ i = 1, \cdots, n$을 초기화 한다.

⑵ $m = 1, \cdots, M$에 대하여 다음 과정을 반복한다.

⒜ 가중치 w_i를 이용하여 분류기 $f_m(x) \in \{-1, 1\}$를 적합한다.

⒝ err_m를 다음과 같이 계산한다.

$$
err_m = \frac{\sum_{i=1}^{n} w_i I(y_i \neq f_m(x_i))}{\sum_{i=1}^{n} w_i}
$$

⒞ $c_m = \log((1 - err_m)/err_m)$로 설정한다.

⒟ 가중치 w_i를 $w_i = w_i \exp(c_m I(y_i \neq f_m(x_i)))$로 업데이트 한다.

⑶ 단계 ⑵에서 얻은 M개의 분류기를 결합하여 최종 분류기 $sign(\sum_{i=1}^{M} c_m f_m(x))$를 얻는다.

AdaBoost 알고리즘의 본래 목적은 훈련오차를 빨리 그리고 쉽게 줄이는 것이었다. 약한 학습기 f_m의 오분류 율이 항상 $0.5-\gamma\ (\gamma>0)$이면 훈련오차는 지수 적으로 빠르게 0으로 수렴함이 증명되었다.[9] 그러나 AdaBoost 알고리즘이 제안된 이후 본래의 목적과는 상관없이 예측오차도 감소한다는 것이 경험적으로 증명되었다. 특히 비슷한 시기에 개발된 배깅에 비해서도 많은 경우에 예측오차가 유의하게 향상되었다. 이후로 AdaBoost 알고리즘의 해석에 대하여 많은 연구가 이루어졌다. Friedman은 AdaBoost가 단계별 전진선택법(stepwise forward selection)을 이용한 변수선택방법이라는 것을 밝혀냈고 축소추정을 통해서 과대적합을 피할 수 있다는 것을 경험적으로 보였다.[7] Efron, Hastie와 Tibshirani는 축소추정을 포함한 부스팅 알고리즘은 lasso 추정치와 동일함을 보였다.[10] 이러한 연구결과들을 통해서 부스팅 알고리즘의 작동원리는 대부분 규명되었다.

Friedman은 부스팅 알고리즘을 최적화 알고리즘의 하나인 기울기 강하(gradient descent) 알고리즘으로 해석하였으며, 이를 통하여 지수 손실함수 이외의 다양한 손실함수에서 부스팅 알고리즘을 개발하였다.[7] 이러한 알고리즘을 그래디언트 부스팅(gradient boosting)이라고 부른다. 기울기 강하 알고리즘을 부스팅에 적용하면 다음과 같다. 주어진 손실함수 L과 주어진 함수집합 F에 대해서 경험위험함수 $R(f)=\sum_{i=1}^{n}L(y,f(x_i))/n$을 최소로 하려고 한다. 주어진 함수 f에서의 경험위험함수의 기울기는 $\nabla(f)=\dot{L}(y,f(x))$로 정의된다. 여기서 $\dot{L}(y,a)=\partial L(y,a)/\partial a$이다.

(1) 해를 $f^c=f_0$로 초기화 한다.

(2) 다음 단계를 해 f^c가 수렴할 때까지 반복한다.

(a) f^c에서 기울기 $\nabla(f)$를 계산한다.

(b) $\nabla(f)$와 가장 가까운 기저 학습기(base learner) g를 다음과 같이 찾는다.

$$g=\arg\min_{h\in F}\sum_{i=1}^{n}(h(x_i)-\nabla(f)(x_i))^2$$

(c) f^c의 이동거리 ρ를 계산한다.

$$\rho=\arg\min_{z\in R}R(f^c+zg)$$

(d) f^c를 g방향으로 ρ만큼 이동하여 새로운 해를 구한다.

$$f^c=f^c+\rho g$$

4. 문제점 및 해결전략

데이터로의 크기는 training set의 경우 고객의 수가 32,428명, test set의 경우 32,427명이 들어있다. 각 고객의 정보를 이루는 변수는 총 38개로 독립변수 37개, 종속변수 1개로 이루어져있고, 각 변수의 형태와 의미는 다음과 같다.

표 1. DMC 2010 데이터의 변수 형태와 의미

Variable	Type	Description	Variable	Type	Description
Customer number	Nominal	Unique customer number	Entry	Nominal	Entry into the shop; 0 = Shop; 1 = Partner;
Saturation	Nominal	0 = Ms ; 1 = Mr ; 2 = Company	Points	Nominal	Points redeemed; 0 = No; 1 = Yes;
Title	Nominal	0 = Not available ; 1 = available	Delivery type	Nominal	Delivery type 0 = dispatch; 1 = collection;
Domain	Nominal	Email provider domain 0 ~ 12	Invoice post code	Nominal	Invoice address postcode
Date created	Cardinal	Date account opened	Deliv post code	Nominal	Delivery address postcode
Newsletter	Nominal	Newsletter subscribed ; 0 = No; 1 = Yes	Shipping costs	Nominal	Shipping costs incurred; 0 = No; 1 = Yes;
Model	Nominal	Model 1; 2; 3	Delivery date real	Cardinal	Delivery date (real)
Payment type	Nominal	0 = Payment on invoice; 1 = Cash payment; 2 = Transfer from current account; 3 = Transfer from credit card;	Delivery date promised	Cardinal	Delivery data (promised)
Voucher	Nominal	Voucher redeemed 0 = No; 1 = Yes;	Weight	Cardinal	Shipment weight
Advertising data code	Nominal	Advertising data code	Remi	Cardinal	Number of remitted items
Case	Ordinal	Value of goods 1 = low; 5 = high;	Cancel	Cardinal	Number of cancelled items
Number items	Cardinal	Number of ordered items	Used	Cardinal	Number of used items
Gift	Nominal	Gift option; 0 = No; 1 = Yes;	w0, … , w10	Cardinal	Number of specific items ordered
			Target90	Nominal	Re-order within 90 days 0 = No; 1 = Yes;

이 중 우리가 예측하고자 하는 변수는 target90이라는 변수로 해당 고객이 90일 이내에 재 구매를 하는지의 여부를 의미하는 변수이다. 1은 재 구매를 하는 경우, 0은 재 구매를 하지 않는 경우로, training set에서는 0의 비율이 약 82% 정도를 차지한다. 고객을 이와 같이 두 부류로 분류함에 있어 모형을 평가하는 척도로는 오분류율이 아닌 총 이익이 된다. DMC 2010 문제에서는 다음과 같이 가정한다.

(1) 쿠폰의 가치는 €5.00이다.
(2) 재 주문을 하지 않는 고객의 경우에 쿠폰을 주면 약 10%의 확률로 90일 이내에 재 주문을 하는데, 평균적으로 €20.00의 구매를 한다.
(3) 쿠폰이 없어도 재 주문을 할 고객에게 쿠폰을 제공 시 쿠폰의 가치 만큼인 €5.00의 손해가 생긴다.

위의 가정들을 토대로 그려 본 이익행렬은 다음과 같다.

표 2. DMC 2010 데이터 셋에 적용할 이익행렬

		실제	
		재 구매 안하는 고객	재 구매 하는 고객
예측	쿠폰 제공 안 함	0	0
	쿠폰 제공 함	1.5	-5

이를 바탕으로 총 이익을 계산해보면 다음과 같다.

i : 고객 번호

g_i : 쿠폰 지급 여부

k_i : 고객의 재 주문 의사 여부

$$x_i = \begin{cases} 0, & g_i = 0 \\ -5, & g_i = 1 \wedge k_i = 1 \\ 1.5, & g_i = 1 \wedge k_i = 0 \end{cases}$$

$$\text{총 이익} = \sum_{i \in \text{고객 번호}} x_i$$

우리는 모형을 통해 $P(target90 = 1 | other\ variables)$을 구할 수 있는데 이 값을 절단 값과 비교함으로써 고객을 최종적으로 0과 1로 나눌 수 있을 것이다. 당연히 절단 값은 총 이익을 최대로 해주는 값으로 정해야하고 이론적으로 다음과 같음이 알려져 있다.

$$\tilde{C}(x) = I(P(y=1|x) > \frac{L_0}{L_0 + L_1})$$

여기서 L_i는 $y = i$에 속한 자료를 오분류 했을 때의 손실을 의미한다. 여기서 $L_0 = 1.5$(이익이 생기지 않는 경우이므로 손실이 1.5만큼 생겼다고 생각), $L_1 = 5$이므로 결국 $\tilde{C}(x) = I(P(y=1|x) > 3/13)$이 된다. 이는 $P(target90 = 1 | other\ variables)$의 값이 3/13보다 크면 재 구매를 할 고객이라 판단하여 쿠폰을 주지 않고, 반대의 경우 쿠폰을 주는 것을 의미한다.

다음으로는 어떠한 과정으로 위 문제를 해결할 것인지 소개하겠다. 아래의 그림은 개략적인 프로세스를 나타낸 것이다.

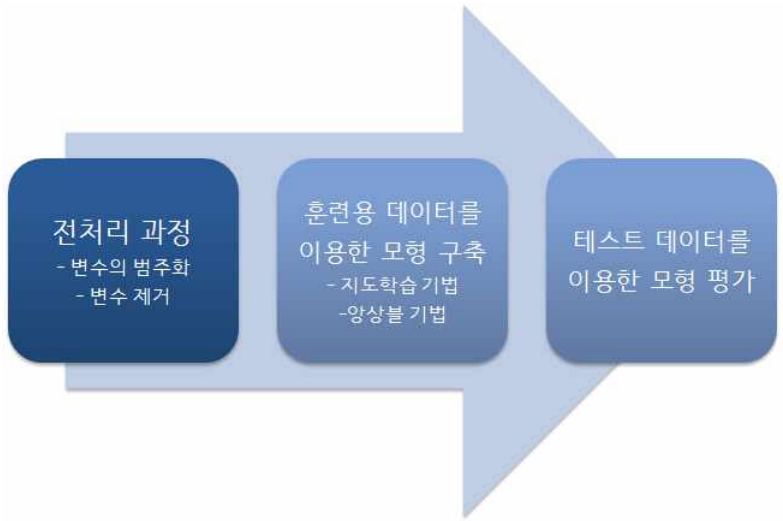


그림 2. 문제 해결의 프로세스

가장 먼저 변수의 범주화라든지 불필요한 변수를 제거하는 데이터 전처리 과정이 필요하다. 다음으로는 훈련용 데이터를 이용해서 모형을 구축하는데, 앞에서 소개한 지도학습기법과 앙상블 기법을 이용하여 만들어본다. 마지막으로 테스트 데이터를 이용하여 만든 모형들을 평가해 본다. 각각의 과정에 대한 자세한 내용은 논문의 다음 파트에서 설명하도록 하겠다.

5. 실험 및 결과

5.1. 실험 내용 및 방법

실험은 파트4에서 소개했듯이 3가지 프로세스를 통해 진행된다. 전처리 과정, 모형 구축, 모형 평가가 바로 그것인데, 각각의 과정에 대해 자세히 살펴보도록 하겠다.

5.1.1. 전처리 과정

가장 먼저 분포가 불균형한 변수들을 범주화 시켜 요인(factor) 변수화 하였다. 이에 해당하는 변수들은 w0, w1, … , w10, cancel, used, remi로 이들은 모두 아래 그림과 같이 0과 1 이상의 값으로 범주를 나누어 각 변수 명에 '.f'를 추가로 붙여 준 변수를 새로 만들어 주었다.

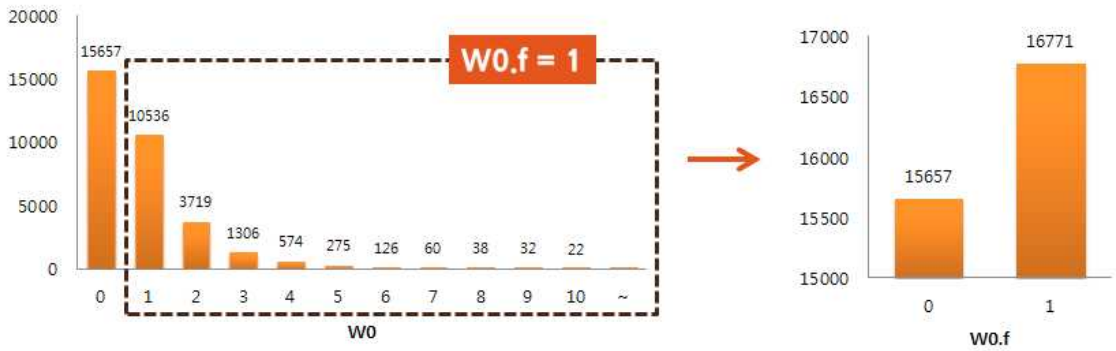


그림 3. 변수의 범주화 과정

다음으로는 일부 변수들을 제거해 주었다. 훈련용 데이터의 결측 치가 20% 이상이 되는 경우 분석을 하는데 어려움이 있어서 이에 해당하는 Delivery date promised, Delivery date real 변수를 제거해주었다. 또한 deliverytype 변수와 paymenttype 변수 사이의 상관관계가 매우 높았는데, 이렇게 상관관계가 높은 두 변수를 모두 사용할 경우 다중공선성(multicolinearity)가 발생하여 분산행렬의 행렬식이 0에 가까운 값이 되어 회귀 계수의 추정 정밀도가 매우 나빠질 수 있으므로 이 중 하나인 deliverytype 변수를 삭제하였다.[2] 실제로 두 변수의 의미를 생각해보면 왜 그런지 알 수 있는데 paymenttype은 지불방식, deliverytype은 배송방식으로, 현금으로 지급 시 직접 수령을 해야 하기 때문에 두 변수 사이에 긴밀한 관계가 있음을 짐작해 볼 수 있다.

5.1.2. 모형의 구축

모형의 구축은 훈련용 데이터를 training set과 validation set으로 나누어 training set으로 모형을 적합한 뒤 validation set으로 모형을 평가하는 방식을 취하였다. N=32,428로 충분히 크므로 교차확인(cross-validation)을 할 필요 없이 단순히 자료를 두 set으로 나눠서 하였다.

training set과 validation set의 크기는 비율을 7:3으로 하였고, 전체 훈련용 데이터의 target90 변수의 0과 1의 비율을 유지하며 두 set에 랜덤으로 나눴다. 이는 샘플링 방식에서 단순랜덤샘플링(simple random sampling)과 층화랜덤샘플링(stratified random sampling)의 차이로 생각할 수 있는데, 일반적으로 층화랜덤샘플링의 경우가 추정 값의 분산이 더 작다는 사실에 기인하였다.[3]

각 알고리즘 별로 몇 가지 parameter들을 정하여 그 값을 변화시키며 성능을 비교하였는데, parameter들의 의미가 간단히 무엇인지 알아보도록 하겠다. (단, 모든 알고리즘에 대해서 변수집합은 원래의 변형하기 전 변수와 요인 화(factorize)시킨 변수로 나누어 각각에 대해 모형을 적합하였다. 앞으로 전자를 original, 후자를 factor라고 표기하겠다.)

먼저 로지스틱 회귀 모형의 경우 모형 선택의 방법을 바꿔가며 비교해보았다. 즉, 전진선택법(forward selection), 후진소거법(backward elimination), 단계적 방법(stepwise method) 중 어떤 방식이 가장 좋은지 찾아보았다. 의사결정나무의 경우 CART알고리즘[5]을 사용하였고, 비교대상인 parameter로는 cp와 maxdepth를 선택하였다. cp는 복잡 도를 나타내는 인자로 나무의 복잡 도를 조정해주고, 이 값이 클 수 록 나무가 더 간단해진다. maxdepth는 나무의 최대 깊이를 나타내는 인자로 이 값이 클 수 록 나무가 더 커질 수 있다. 신경망 모형의 경우 층(layer)이 1개인 다층 신경망 모형으로 고정하였고, 비교대상인 parameter로는 hidden과 learningrate를 선택하였다. hidden은 은닉 노드의 수를 나타내는 변수이고, learningrate은 학습률을 나타낸다. 학습률이 크다는 것은 역전파 알고리즘을 적용하는 단계에서 각 단계별 업데이트 되는 값의 변화가 커지는 의미가 있다.

다음으로 배깅의 경우에는 의사결정나무와 신경망모형의 경우에 대해 모두 적용해보았는데, 위의 parameter에 추가로 전체 예측모형의 개수인 B를 선택하였다. 참고로 분류 문제이긴 하지만 다수결(voting) 방법을 사용하지 않고, $P(Y=1|x)$를 평균(averaging)하는 방법을 택하였다. 마지막으로 부스팅의 경우에는 ada부스팅과 gradient부스팅에 대해 모형을 적합해보았다. ada부스팅의 경우 비교대상의 parameter로 iter와 maxdepth를 선택하였다. iter는 부스팅의 반복횟수를 나타내는 변수이고, maxdepth는 ada부스팅이 단일학습기로 의사결정나무를 사용하는 만큼 이 나무의 최대깊이를 나타내는 변수이다. gradient부스팅은 비교대상의 parameter로 mstop과 nu를 선택하였다. mstop은 iter와 마찬가지로 부스팅의 반복횟수를 나타내는 변수이고, nu는 축소(shrinkage)의 정도를 나타내는 변수이다.

5) 대표적 의사결정나무 알고리즘으로 이지분류(binary split)를 이용하고, 불순도로는 목표변수가 범주형인 경우 지니 지수(gini index)를 이용한다.

다음은 위의 parameter들에 대해 어떤 값을 사용하였는지 보여주는 표이다.

표 3. 각 알고리즘 별 사용한 parameter 값

로지스틱 회귀	변수선택법		
	전진선택법, 후진소거법, 단계적방법		
의사결정나무 (CART)	cp		maxdepth
	-1, 0.00005~0.00045(0.00005씩 증가), 0.0005~0.0200(0.0005씩 증가)		1~30(1씩 증가)
신경망 (MLP)	hidden		learningrate
	3~10(1씩 증가)		0.1~1.0(0.1씩 증가)
배깅 (CART)	B	cp	maxdepth
	20~100(10씩 증가)	-1, 0.0005~0.00050(0.00005씩 증가)	5~30(5씩 증가)
배깅 (MLP)	B	hidden	learningrate
	20~100(10씩 증가)	5~10(1씩 증가)	0.1~0.5(0.1씩 증가)
부스팅 (ada)	iter		maxdepth
	30~150(10씩 증가)		1~5(1씩 증가)
부스팅 (gradient)	mstop		nu
	500~5000(500씩 증가)		0.005~0.100(0.005씩 증가)

5.1.3. 모형의 평가

training set을 이용하여 위의 parameter 값들로 적합한 모형 중 validation set에 대하여 총 이익을 최대로 만들어주는 모형을 각 알고리즘별로 대표 모형으로 하나씩 채택한다. 그리고 해당 모형에 대해 최종적으로 test set에 대하여 이익을 계산해보고 이를 통해 모형의 성능을 평가해보도록 한다.

5.2. 실험 결과

각 알고리즘 별로 validation set에 대하여 최고의 이익을 내는 parameter는 다음과 같다. 여기서 empty는 모두에게 쿠폰을 주는 경우에 해당하며, Lift는 이에 대비하여 각 모형의 이익이 몇 %나 증가하였는지 나타내주는 항목이다.

표 4. 각 알고리즘 별 validation set에 대한 최대 이익을 내는 parameter와 그 이익

Algorithm	변수집합	parameter	이익(€)	Lift(%)
empty			2789.5	0.00
로지스틱 회귀	original	FS, BW, SM	3370.0	20.81
	factor	FS, BW, SM	3552.5	27.35
의사결정나무 (CART)	original	cp=0.0002, maxdepth=8	3567.5	27.89
	factor	cp=0.0002, maxdepth=7	3523.0	26.30
신경망 (MLP)	original	hidden=10 learningrate=0.2	3578.0	28.27
	factor	hidden=9 learningrate=0.1	3548.0	27.19
배깅 (CART)	original	B=50, cp=0.0003, maxdepth=8	3616.0	29.63
	factor	B=60, cp=0.0002, maxdepth=9	3566.5	27.85
배깅 (MLP)	original	B=50, hidden=9, learningrate=0.3	3646.5	30.72
	factor	B=70, hidden=10, learningrate=0.2	3596.0	28.91
부스팅 (ada)	original	iter=60, maxdepth=3	3540.5	26.92
	factor	iter=70, maxdepth=2	3527.5	26.46
부스팅 (gradient)	original	mstop=5000, nu=0.005	3631.5	30.18
	factor	mstop=5000, nu=0.010	3573.5	28.11

최종적으로 모형을 평가할 때 변수집합도 선택하기 위하여 각 알고리즘 별로 두 변수집합 사이에 더 큰 이익을 내는 부분에 음영을 넣었다. (로지스틱 회귀 모형을 제외하고는 모두 원래의 변수집합이 선택되었다.)

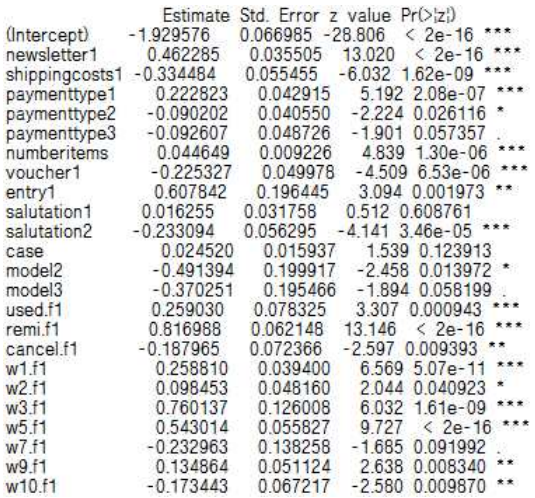
이제 각 알고리즘의 대표 모형을 갖고 test set에 대하여 이익을 계산해보면 표 5와 같다.

표 5. 각 알고리즘 별 최적 모형의 test set에 대한 이익

Algorithm	변수집합	parameter	이익(€)	Lift(%)
empty			8548.5	0.00
로지스틱 회귀	factor	FS, BW, SM	11545.0	35.05
의사결정나무 (CART)	original	cp=0.0002, maxdepth=8	11511.5	34.66
신경망 (MLP)	original	hidden=10 learningrate=0.2	11643.5	36.21
배깅 (CART)	original	B=50, cp=0.0003, maxdepth=8	11754.5	37.50
배깅 (MLP)	original	B=50, hidden=9, learningrate=0.3	11877.0	38.94
부스팅 (ada)	original	iter=60, maxdepth=3	11623.0	35.97
부스팅 (gradient)	original	mstop=5000, nu=0.005	11946.0	39.74

5.3. 결과 분석

가장 먼저 단일 학습기 중 모형의 해석이 용이한 로지스틱 회귀 모형과 의사결정나무를 통해 고객의 재 구매에 크게 영향을 끼치는 변수가 무엇인지 알아보도록 하겠다. 로지스틱 회귀 모형에서 선택된 변수와 그 변수의 계수 추정치는 그림 4와 같다.



	Estimate	Std. Error	z value	Pr(>|z|)	
(Intercept)	-1.929576	0.066985	-28.806	< 2e-16	***
newsletter1	0.462285	0.035505	13.020	< 2e-16	***
shippingcosts1	-0.334484	0.055455	-6.032	1.62e-09	***
paymenttype1	0.222823	0.042915	5.192	2.08e-07	***
paymenttype2	-0.090202	0.040550	-2.224	0.026116	*
paymenttype3	-0.092607	0.048726	-1.901	0.057357	.
numberitems	0.044649	0.009226	4.839	1.30e-06	***
voucher1	-0.225327	0.049978	-4.509	6.53e-06	***
entry1	0.607842	0.196445	3.094	0.001973	**
salutation1	0.016255	0.031758	0.512	0.608761	
salutation2	-0.233094	0.056295	-4.141	3.46e-05	***
case	0.024520	0.015937	1.539	0.123913	
model2	-0.491394	0.199917	-2.458	0.013972	*
model3	-0.370251	0.195466	-1.894	0.058199	.
used.f1	0.259030	0.078325	3.307	0.000943	***
remi.f1	0.816988	0.062148	13.146	< 2e-16	***
cancel.f1	-0.187965	0.072366	-2.597	0.009393	**
w1.f1	0.258810	0.039400	6.569	5.07e-11	***
w2.f1	0.098453	0.048160	2.044	0.040923	*
w3.f1	0.760137	0.126008	6.032	1.61e-09	***
w5.f1	0.543014	0.055827	9.727	< 2e-16	***
w7.f1	-0.232963	0.138258	-1.685	0.091992	.
w9.f1	0.134864	0.051124	2.638	0.008340	**
w10.f1	-0.173443	0.067217	-2.580	0.009870	**

그림 4. 로지스틱 회귀 모형에서의 변수의 계수 추정치

그림 5는 적합 된 의사결정나무 모형에 대한 요약(summary)결과의 내용으로 변수의 중요도를 보여준다.

```
Call:
rpart(formula = yformula(var.x), data = dmc2010.train, cp = 0.0002,
    maxdepth = 8)
  n= 22699

           CP nsplit    rel error       xerror          xstd
1  0.0010625737898      0 1.0000000000 1.0000000000 0.01385902972
2  0.0010232192050      4 0.9957497048 0.9978748524 0.01384766933
3  0.0007083825266      9 0.9893742621 0.9988193625 0.01385272131
4  0.0005903187721     11 0.9879574970 0.9990554900 0.01385398358
5  0.0004722550177     13 0.9867768595 1.0000000000 0.01385902972
6  0.0004427390791     16 0.9853600945 1.0089728453 0.01390673576
7  0.0004132231405     25 0.9811097993 1.0089728453 0.01390673576
8  0.0003935458481     29 0.9794569067 1.0089728453 0.01390673576
9  0.0003148366785     41 0.9747343566 1.0110979929 0.01391797328
10 0.0002361275089     44 0.9737898465 1.0139315230 0.01393292033
11 0.0002000000000     58 0.9704840614 1.0219598583 0.01397504605

Variable importance
      weight   paymenttype    newsletter          remi        domain shippingcosts          case         model
          15            14            13            11             9             5             5             4
 numberitems            w0            w5            w1       voucher        cancel          used    salutation
           3             3             3             2             2             2             2             2
       entry            w9           w10            w3            w2   advertising
           2             2             1             1             1             1
```

그림 5. 의사결정나무 모형에서의 각 변수들의 중요도

두 모형의 결과를 종합해보면 newsletter(뉴스레터 구독 여부), remi(반송된 상품의 수/반송 여부), paymenttype(지불 방식), shippingcosts(배송 비용 발생 여부), case(상품의 가치)등이 공통적인 주요 인자임을 확인해 볼 수 있다.

다음으로는 모두 쿠폰을 주는 경우에 비해 각 알고리즘의 모형들이 얼마만큼 이익을 증가시켰는지 살펴보겠다. 위의 표에서 볼 수 있듯이 단일 지도학습기법을 이용한 모형들의 경우, 로지스틱 회귀 모형은 35.05%, 의사결정나무는 34.66%, 신경망 모형은 36.21%의 이익이 증가하였다. 신경망 모형의 성능이 가장 좋다고 할 수 있는데, 해석이 어렵다는 단점은 있지만 예측력이 뛰어나다는 장점을 확인해 볼 수 있었다. 다음으로 앙상블 기법을 이용한 모형들의 경우, 배깅(나무)은 37.50%, 배깅(신경망)은 38.94%, 부스팅(ada)은 35.97%, 부스팅(gradient)은 39.74%의 이익이 증가하였다. 앙상블 기법으로 배깅, 부스팅, 랜덤 포레스트 등이 있는데 일반적으로 부스팅의 성능이 가장 좋다고 알려져 있고, 실험을 통해 이 역시 확인해 볼 수 있었다.

마지막으로 단일 학습기와 앙상블 기법을 통한 모형의 성능을 비교해 보겠다. 의사결정나무를 이용한 경우 이익 증가율이 34.66%였는데, 의사결정나무를 기저 학습기로 하는 배깅의 경우 37.50%, 부스팅(ada)의 경우 35.97%로 모두 성능이 향상된 것을 볼 수 있다. 또한 신경망 모형을 이용한 경우 이익 증가율이 36.21%이고, 신경망 모형을 기저 학습기로 하는 배깅의 경우 38.94%로 역시 성능 향상을 목격할 수 있다. 하지만 성능 향상의 정도가 작다고 볼 수도 있는데, 단일 학습기의 적합 과정을 생각해보면 그 이유를 알 수 있다. 의사결정나무를 적합할 때, parameter 조합의 경우의 수만 1500(=50×30)가지를 따져보았다. 즉 매우 많은 parameter 값들의 조합 중 최적의 조합을 찾은 것인데, 이는 의사결정나무로 만들 수 있는 모형 중 최적의 모형을 찾았다고 볼 수 있는 것이다. 실제로 parameter 값의 범위를 줄이거나 랜덤으로 parameter 값 들을 선택하여 모형을 적합해본다면 증가하는 이익의 양이 많이 줄어드는 것을 확인할 수 있었고, 앙상블 모형의 상대적 성능 향상이 큰 것 또한 알 수 있었다.

6. 결론

6.1. 연구 결과 요약

단일 학습기들 중 해석력이 좋은 로지스틱 회귀 모형이나 의사결정나무를 통해 고객의 재 구매 예측에 영향을 많이 주는 변수가 뉴스레터 구독 여부, 반송된 상품의 수, 지불 방식, 배송 비용 발생 여부, 상품의 가치 등임을 확인해 볼 수 있었다. 그리고 지도학습 기법들 중에는 신경망 모형이 36.21%로, 앙상블 기법들 중에서는 부스팅(gradient)이 39.74%로 이익을 가장 크게 증가시켰다. 특히 이번 연구에서 앙상블 기법을 주요 문제 해결 방법으로 채택했는데 단일 학습기들에 비해 좋은 성능을 내는 것을 직접 확인해 볼 수 있었다.

6.2. 향후 과제

본 연구에서 발전하여 쇼핑몰의 이익을 더 증가시키기 위한 방법으로는 두 가지를 생각해 볼 수 있다.

첫 번째로는 변수의 범주화를 조금 더 세분화시키는 방법이 있다. 이번 연구에서는 분포가 불균형을 이루는 변수들에 대하여 두 범주로 나누어 주었다. 하지만 이렇게 범주화하는 것은 불균형한 변수를 균형한 변수로 만들어 줌과 동시에 정보를 손실시킨다는 단점을 갖고 있다. 실제로 연구 결과를 통해서도 로지스틱 회귀 모형을 제외하고는 모두 범주화한 변수를 사용한 경우에 이익이 감소하는 것을 확인할 수 있었다. 이러한 정보의 손실 정도를 줄이는 방법으로는 범주의 항목을 늘리는 것이 있다. 그러나 더 세분화된 범주화를 하기 위해서는 데이터에 대한 고찰이 더욱 필요하다. 범주를 나누는 명확한 기준이 있고 각 범주들이 의미를 가져야 하며, 또한 범주 별로 데이터의 분포가 균형을 이뤄야 할 것이다.

두 번째로는 재 구매를 하지 않을 고객을 다시 두 부류로 나누어 쿠폰을 지급해도 재 구매를 하지 않을 고객과 할 고객으로 나누는 것이다. 문제의 가정에서 재 구매를 하지 않을 고객들 중 쿠폰을 지급했을 때 의사를 바꾸는 비율이 10%라고 명시되어 있다. 즉 90%는 의사를 바꾸지 않을 고객들이고 이들에게 쿠폰을 지급함으로써 생기는 손실이 있을 수 밖 에 없다. 하지만 현재의 데이터에서는 이들을 분류해 내는 것이 쉽지 않다. 첫 구매에 쿠폰을 사용했는지의 여부를 나타내는 voucher라는 변수가 있긴 하지만 이를 이용하여 결론을 내기에는 무리가 있다. 즉 이러한 분류를 위해서는 판매자가 제공하는 정보에 고객들의 쿠폰에 관련된 과거 정보가 더 추가되어야 할 것이다.

참고문헌

[1] 박창이, 김용대, 김진석, 송종우, 최호식, 『R을 이용한 데이터마이닝』, 교우사, 2011.

[2] Doublas C. Montgomery, Elizabeth A. Peck, G. Geoffery Vining, 『Introduction to Linear Regression Analysis』, Wiley Interscience, 2006.

[3] Richard L. Scheaffer, William Mendenhall Ⅲ, R. Lyman Ott, 『Elementary Survey Sampling』, Thomson Brooks/Cole, 2006.

[4] Benjamin Hofner, Andreas Mayr, Nikolay Robinznov, Matthias Schmid, “Model-based Boosting in R : A Hands-on Tutorial Using the R Package mboost”, 2012.

[5] David H. Wolpert, William G. Macready, “Combining Stacking With Bagging To Improve A Learning Algorithm”, 1996.

[6] Yoav Freud, Robert E. Schapire, “A Short Introduction to Boosting”, 1999.

[7] Jerome H. Friedman, “Greedy Function Approximation : A Gradient Boosting Machine”, 2001.

[8] Leo Breiman, "Bagging Predictors", 1994.

[9] Yoav Freud, Robert E. Schapire, "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting", 1997.

[10] B. Efron, T. Hastie, R. Tibshirani, “Least angle regression”, 2004.