AI in Kakao

카카오내비 예측의 정확성 그리고 AI

사람들은 이동한다. 직장에 출근하고 친구를 만나고 여행을 떠나고 물건을 나른다. 이처럼 이동은 우리 생활에서 매우 중요한 일 중 하나다. 그렇다면 이동비용에 대해 생각해 본적이 있는가? 꽉 막힌 도로 위에서 보내는 시간, 자원 낭비, 탄소 배출로 인한 환경 오염은 개인의 문제가 아니라 사회적으로 큰 손실이다. 누구나 꽉 막힌 도로 위에서 시간을 허비한 경험이 있을 것이다. 그럴 때면 우리는 스스로 다음과 같은 질문을 하곤 한다.

이 길 말고 다른 길로 갔으면 덜 밀렸을까? 정말 이 길이 최선의 선택인가? 덜 막히는 길은 과연 어디일까? 지금은 정체되었지만 조금만 기다린다면 정체가 풀릴 곳은 어디일까?

글 | 윤지상 jason.yoon@kakaocorp.com

카카오의 추천 기술을 연구하고 개발하는 일을 하고 있습니다. 아직 답이 없는 새로운 문제들을 부딪혀서 해결할 때 큰 즐거움을 느낍니다. 제가 가진 아이디어가 세상에 많은 이로움을 주었으면 좋겠습니다.

글 | 김성진 nick.kim@kakaocorp.com

건담(Gundam) 열혈 팬입니다. 마음 같아선 건담의 팔, 다리, 머리 다 만들고 싶지만 일단 머리부터 만들어보자는 생각에 머신러닝을 시작했습니다. 현재 카카오 추천팀에서 다양한 머신러닝 시스템을 개발하고 있습니다. Practical한 머신러닝 시스템 만드는 것을 좋아합니다.

글 | 권영주 sami.kwon@kakaocorp.com

검색 개발자로 시작해 지금 카카오내비 교통 정보 분석, 예측모듈을 개발하고 있습니다. 개발자에서 빅데이터 처리&분석을 거쳐 지금은 인공지능 분야에서 새내기의 마음으로 시작하고 있습니다. 개발자로 일과 삶의 조화를 유지하는 것이 늘 어려웠는데 인공지능의 발달로 미래에는 그런 삶이 될수 있게 만들고 싶습니다.

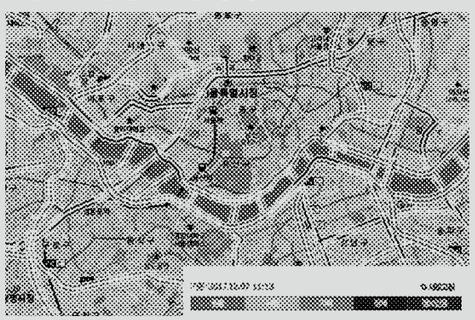
카카오모빌리티가 이동에 관한 문제들을 풀어가는 과정에 사용된 카카오내비의 도착시간 예측 시스템에 대해 이야기 해보고자 한다.

[그림 1] 도착시간 예측이 사용되는 서비스의 예시

Q: 카카오내비의 도착예정 시간은 현재 속도로 계산되나요?
A: 아니요. 미래의 교통 흐름을 예측하여 도착예정 시간을 구합니다

도로 위 교통 상황은 수시로 변한다. 카카오내비 시스템은 실시간으로 수집되는 교통정보를 이용하여 사용자에게 도로의 상태정보(원활, 지체, 정체 등)를 제공하고, 이 과정에서 누적된 빅데이터는 미래 교통 정보 예측의 초석이 된다. 카카오내비는 오랜 기간 서비스를 통해 다양한 데이터를 축적해 왔다. 데이터가 부족했던 과거에는 다양한 속도 예측 방식을 쓰는 데 한계가 있었다. 그러나 충분한 데이터가 모인 현재는 축적된 데이터로 다양한 기법을 통해 속도 예측 모델을 실험하고 고도화할 수 있게 됐다.

[그림 2] 카카오맵의 실시간 도로별 교통 현황[*1]

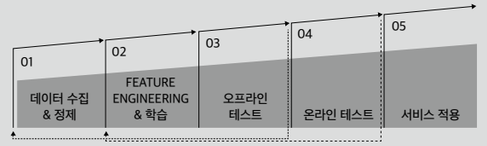


머신러닝을 통한 예측

카카오내비는 사용자에게 더 정확한 도착예정 시간을 제공 하는 것을 주요 목표로 삼고 있다. 도착예정 시간은 출발지에서 도착지까지 걸리는 전체 소요 시간을 의미하며 도로별 주행소요 시간, 회전 시간, 신호대기 시간 등의 합으로 예측된다. 이 중 도로별 주행시간은 가장 큰 비중을 차지한다. 카카오내비는 도로별 속도를 예측하는 모델을 통해 산출된 속도를 활용하여, 도로별 주행소요 시간을 계산한다.

학습의 입력으로 사용되는 실시간 교통정보는 상당히 불완전하다는 특성이 있다. 우리는 다양한 머신러닝 방법론 중 캐글(kaggle) 등 여러 머신러닝 대회에서 그 성능이 검증된 GBT(gradient boosting tree) 알고리즘[*2]을 사용하였다. 카카오내비는 이 알고리즘을 적용해 도로별 주행 속도를 예측하는 실험을 했고, 이를 서비스에 적용했다.

[그림 3] 카카오내비 내 머신러닝 서비스 적용 과정



과거에는 제한된 영역에서 설치된 카메라 혹은 도로 위 내장된 센서등을 통해서만 교통 정보를 수집할 수 있었다. 이 과정에는 큰 비용이 소요됐다. 카카오내비는 서비스 출시 때부터 크라우드(crowd) 방식으로 카카오내비로 주행한 모든 도로의 교통정보(주행시간, 주행거리 등)를 '카카오내비 이용약관'의 '위치정보 수집 및 이용 동의'에 동의한 사용자에 한해 실시간으로 수집해왔다.[*3] 교통정보는 전국 도로에 분산되어 있는 카카오내비 이용자들의 주행정보로부터 실시간으로 갱신된다. 2012년 국도, 지방도로 중심으로 수집된 교통정보를 이용한 실시간 속도정보를 제공하기 시작했고, 2013년 모든 '교통정보'를 '수집된 교통정보'로 전환하고 누적된 데이터를 기반으로 도착예정 시간을 산출했다. 교통정보의 양은 카카오내비 사용자 증가와 동시에 증가했다. 이에 따라, 데이터의 정확도 역시 높아졌다.

충분한 크기의 데이터 확보 뿐만 아니라, 데이터 상의 노이즈를 제거하는 것은 머신러닝에서 아주 중요한 작업이다. 목적 없이 쌓인 데이터는 무용지물이다. 필요한 데이터를 어떤 형태로 쌓을지 고민하고 정확한 데이터를 쌓는 작업이 필요하다. 그리고 쌓여진 데이터로 부터 정제된 데이터를 추출할 수 있어야 한다. 카카오내비에서 수집하는 교통정보 안에는 정차, 사고, GPS오차, 터널 주행 등의 다양한 변수를 포함하고 있기 때문에 정제하지 않은 데이터를 사용하게 되면 정확한 예측 결과를 얻을 수 없다.

예를 들어 정차 중인 차량의 상태를 정체로 판단한다면 도착

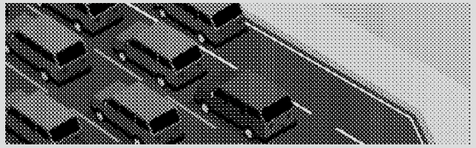
예정시간이 실제와 다르게 예측될 수 있다. 같은 고속도로 상에서도 직진 차선과 출구 쪽 차선의 속도가 다르듯, 직진, 우회전, 좌회전에 대한 구분 없이 속도 데이터를 분리하여 정제하지 않는다면 올바른 예측값을 만들어내기 어려울 것이다. 노이즈 처리가 잘못되더라도 모델 성능에 안 좋은 영향을 끼칠 수 있다. 데이터 정제 로직(logic)은 계속 고도화되고 있다.

도로, 신호등, 감시카메라, 과속방지턱 등은 늘 생성되고 소멸되는데 그때마다 교통 흐름에 변화가 생긴다. 교통 흐름은 요일, 시간, 날씨 그리고 계절에 따라 달라진다. 명절이나 공휴일, 집회나 행사에 따라서도 교통 흐름은 변한다. 다양한 변수 속에서도 정확한 예측을 하려면, 최신의 데이터를 확보하고, 알고리즘의 학습과 평가를 주기적으로 진행하여야 한다. 예측 모델의 최신성을 위해서는 데이터의 지속적인 수집 및 정제, 모델의 성능 제고를 위한 추가 수집 데이터가 고민되어야 한다. 다음은 정확한 속도 예측을 위해 데이터 수집 및 처리에 있어 고려해야하는 사항들이다.

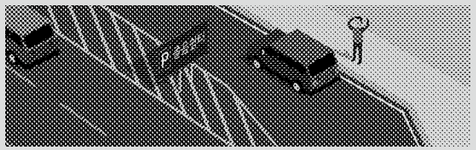
정차와 정체의 구분

위 이미지의 경우 정상적인 경우이지만 아래 이미지의 정차차량은 정제 대상이다.

[그림 4] 정체와 정차의 구분



정체 : 같은 도로위에서 정체된 차량으로 속도가 느리게 측정됨

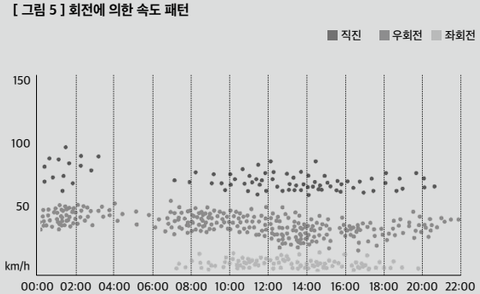


정차 : 도로 위 차량의 소통은 원활하지만 정차(졸음쉼터내) 차량으로 인해 속도가 낮게 측정됨

회전에 의한 속도 변화

같은 도로에서 진행 방향(좌회전, 우회전, 직진)에 따라 다른 속도 패턴으로 분류된다. 각 패턴별로 다르게 속도를 측정해야 한다.

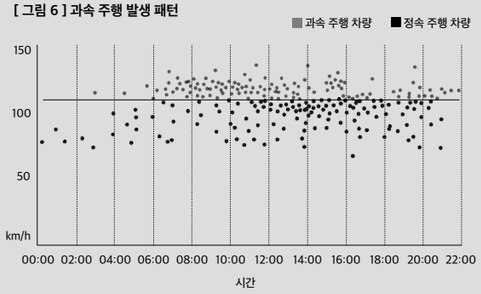
[그림 5] 회전에 의한 속도 패턴



과속에 의한 속도 오류

교통 법규 내의 최고 속도와 실제 주행에서 과속으로 인한 최고 속도의 차이로 인해 예측 시간이 실제 소요 시간보다 길게 예상되는 경우이다. 카카오내비는 최고 제한 속도 이하로만 안내하기 때문에 과속에 의한 주행 시간이 최초 예측시간보다 짧은 경우가 발생될 수 있다.

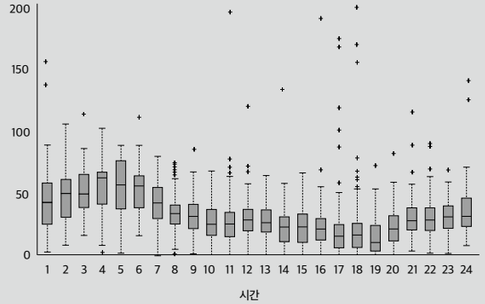
[그림 6] 과속 주행 발생 패턴



노이즈 처리

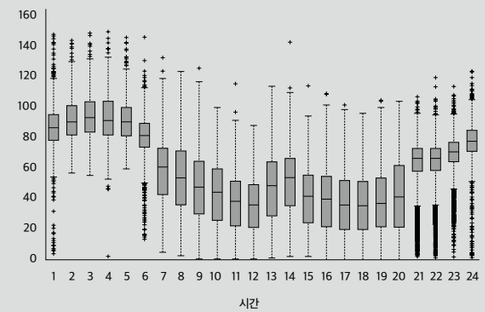
수집되는 교통 정보는 같은 도로이지만 사용자의 성향, 도로 위에서의 위치(길이가 긴 도로에서는 위치에 따라 교통 상황이 다름) 등에 따라 다양한 범위로 수집된다.

[그림 7] 일반 도로 노이즈



수집된 데이터에서 노이즈를 처리하고 박스의 데이터 속도를 계산한다.

[그림 8] 갈림길 도로 노이즈



노이즈 분류 작업에서는 21시 이후에 속도 패턴이 두개로 나누어지는 현상을 볼수 있다. 이런 경우는 갈림길일 확률이 높다. 갈림길은 진행 방향에 따라 노이즈 제거를 분리해서 한다.

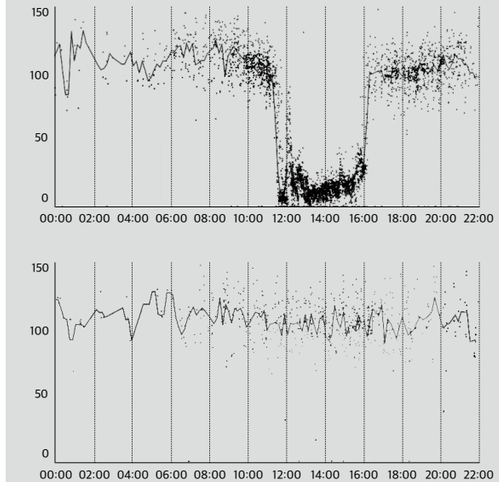
Feature engineering & 학습

1) 학습 모델

학습 모델은 교통정보와 같은 불완전한 특성을 갖는 데이터에서도 높은 성능이 검증된 GBT 알고리즘을 사용하였다. GBT 알고리즘은 목적함수를 최적화하는 방향으로 트리(tree)를 더해가면서 모델을 만드는 알고리즘이라 할 수 있다. GBT 알고리즘은 유명 머신러닝 경진대회인 캐글에서 상점의 매출을 예측하는 대회[*4]의 우승자가 사용했던 모델이기도 하다. 상점의 미래 매출액을 예측하는 문제와 도로의 미래 속도를 예측하는 문제는 공통점이 많다고 판단되어, GBT 알고리즘으로 실험 해보기로 결정했다.

교통 흐름은 요일과, 명절과 공휴일을 비롯한 연휴, 길의 모양에 따라 다른 패턴을 보인다([그림 4], [그림 9] 참고). 그러하기에, 평일 패턴에 최적화된 모델로 징검다리 휴일의 교통 흐름을 예측하면 제대로된 예측이 될 수 없다. 우리는 다른 패턴의 교통 흐름에 대해 다수의 모델을 사용해서 문제를 해결했다. GBT 알고리즘 적용으로 인한 도로별 속도 오류(mean relative error)의 개선 정도는 글의 말미에 있는 [표 2]에서 확인할 수 있다.

[그림 9] 중앙고속도로, 부산방향 중 일부 명절과 평일의 교통흐름 차이



2) 유의미한 특성 선택

모델을 학습시키려면, 데이터에서 특성(feature)을 뽑아내야 한다. 데이터 분석이나 실험 중 얻은 통찰력(insight)을 통해 아래와 같은 특성 집합(feature set)을 구성하고 그에 맞는 특성들을 데이터로부터 추출한다. 특성 추출(feature extraction)할 때는 전술한 상점 매출의 예측 대회 우승자가 사용한 방법[*5*6]을 참고했다. 우승자는 해당 대회에서 최근 관측 정보에 중점을 둔 특성, 오랜 기간에 걸쳐 관측된 트렌드에 중점을 둔 특성 등을 다양하게 사용했다.

카카오내비는 이러한 특성들이 교통정보 예측에도 도움이 될 거라 판단했다. 물론, 앞서 추출된 특성이 교통정보 예측용으로 모두 유의미한 것은 아니다. 그래서, 교통정보 예측에 대한 각 특성의 성능 기여도를 확인해 본 뒤 불필요한 특성은 제외한 후 모델을 학습하는게 중요하다. 여기에는 다른 머신러닝 대회에서 사용했던 피쳐 엔지니어링(feature engineering) 방식[*7*8]에서 힌트를 얻어, 2단계 특성 선택(2-step feature selection) 방식을 사용하였다. GBT는 모델을 한 번 학습해보면 트리 분기점(split point)에서 사용된 빈도 수에 의해 각 특성의 중요도를 파악할 수 있다. 이 방식은 전체 데이터로 학습해 본 뒤 특성의 중요도를 파악하기엔 시간도 오래 걸리고 비용이 많이 요구되는 문제의 해결책이 될 수 있다. 카카오내비는 작은 데이터셋으로 먼저 학습해서 각 특성의 중요도를 파악한 뒤 중요도가 높은 특성들로 최종 버전 모델을 학습하는 방법을 사용했다.

[표 1] Feature set

최근 30주치 타임슬라이스별 속도 표준편차
최근 30주치 타임슬라이스별 속도 평균
예측값을 만드는 시간 기준으로 ×분 전 관측 속도
주별 Daily 속도 평균
최근 속도와 30주 이내의 각 시간대별 데이터의 속도차
현재 속도, 최근 1시간 이내 속도
도로 길이(내부정의)
도로 타입(내부정의)
평균 교통량(high, med, low)

3) 목적 함수

회귀(regression) 문제에서 목적 함수로는 평균 제곱근 오차(root mean square error, RMSE)가 많이 쓰이지만, 우리는 RMSE 대신 Log RMSE를 사용했다. 도착 시간 예측의 문제에서는 고속보다는 저속의 도착 시간을 정확하게 계산하는 것이 중요하며, 이를 위해서는 절대적인 에러보다 상대적인 에러를 이용해야 하기 때문이다.

RMSE를 이용할 경우 절대적인 에러값을 계산한다. 예를 들어 정답 속도가 90km/h 일 때 100km/h로 예측하는 것과 정답이 20km/h 일 때 10km/h로 예측하는 것의 loss를 동일하게 보고 모델이 학습을 하게 된다.

하지만, 소요 시간 관점에서 본다면 이야기가 달라진다. 도로의 길이가 100km라고 할 때, 전자는 정답이 1.11 시간(100/90)인데 1시간(100/100)으로 예측한 것이지만, 후자는 5시간(100/20)이 정답인데 10시간(100/10)으로 예측한게 된다. 즉, 똑같이 절대값으로 10km/h 속도 에러가 발생하더라도 고속이냐

저속이냐에 따라 소요시간 오차 측면에서 큰 차이(0.11시간, 5시간)가 있을 수 있다.

도착 시간 예측 문제에서는 저속을 얼마나 정확하게 맞추느냐가 중요했으며, 그것을 위해 절대적인 에러보다는 상대적인 에러를 학습시키기 위해 Log RMSE를 사용하였다

4) 관측 값이 없는 경우에 대한 대처

전국의 도로별 주행량을 보면 지방도로나 생활도로*9는 실제 통행량이 없는 경우가 비일비재하다. 지방에 있는 도로, 차들이 없는 도로를 운전해본 경험은 누구나 가지고 있을것이다. 이렇듯 관측값이 충분치 않거나 관측값이 없는 경우에도 예측값은 실시간으로 생성되고 있다. 관측값이 없을 경우, 평균, 중간값, 선형보간법(interpolation)등의 방법을 이용해서 빈 관측값들을 채워나가는 방법을 많이 사용하는데, 교통 데이터에서는 이런 전통적인 방법들이 잘 동작하지 않았다. 소요 시간 예측 문제에서는 고속구간보다 저속구간을 정확하게 잘 맞추는 것이 중요한데, 값을 채우는 방법들이 이 부분에서 개선점을 찾아주지 못한게 원인이라고 생각된다.

내부 실험 결과, null 값 자체를 모델이 학습할 수 있도록 만드는 방식의 성능이 가장 좋았다. “null 값 자체를 학습한다”는 말은 GBT 학습할 때 트리의 각 분기점 즉 스플릿 포인트(split point)에 null 값이 들어왔을 때 어떻게 대처해야하는지 학습하는 알고리즘이 있는데, 그 알고리즘을 최대한 활용했다는 것이다. 더 자세한 내용은 티안취 첸(Tianqi chen)과 카를로스 구에스트린(Carlos Guestrin)의 논문에서 “3.4 Sparsity-aware Split Finding”를 참고하기 바란다.*10

5) 학습 데이터 샘플의 비중

모든 데이터 샘플에 대해 동일한 비중을 주는게 좋은 방법일까? 다른 비중을 준다면 어떻게 다르게 주는게 좋을까? 속도 에러가 동일하더라도, 소요 시간 관점에서 본다면 길이가 긴 도로에서 더 큰 오차가 발생하며, 교통량이 많은 도로에서 발생하는 에러가 서비스 전체 오차에 더 많은 영향을 준다. 우리는 도로의 길이와 교통량 모두를 고려하여 데이터 샘플 비중을 결정하는 로직을 사용했다.

검증: 오프라인 시뮬레이션

온라인 테스트는 확실하게 모델 간 성능 비교를 할 수 있다는 장점이 있지만, 단점도 많이 갖고 있다. 첫번째 단점은 시간이 오래 걸린다는 것이다. 예를 들어 신규 로직을 적용했다면, 최소한 하루는 지켜보면서 출·퇴근 시간의 에러를 확인해봐야 전반적인 성능 개선치를 확인할 수 있기 때문이다. 두 번째 단점은 동일한 시간에 테스트해볼 수 있는 모델 개수가 유한하다는 것이다. 온라인 테스트에서는 버킷 테스트(bucket test) 방식*11이나 서비스 환경과 동일한 환경을 만들어서 테스트하는 방식을 사용할 수 있는데, 두 방식 모두 이러한 문제점을 갖는다.

위와 같은 특성 때문에, 모델 최적화 과정에 필요한 수많은 실험을 모두 온라인으로 진행하는 것은 현실적으로 불가능했다. 이러한 점을 해결하기 위해 우리는 온라인 테스트를 최대한 비슷하게 재현할 수 있는 오프라인 시뮬레이터를 개발했다. 오프라인 시뮬레이터를 사용하면 위 2개의 단점을 모두 극복할 수 있다. 온라인으로는 하루, 이틀 걸려야 확인할 수 있는 결과를 오프라인에서는 1분도 안 걸려서 실험해 볼 수 있으며, 동일한 시간에 테스트할 수 있는 모델 수에도 제한이 없다.

오프라인 시뮬레이터는 온라인 테스트를 얼마나 잘 재현할 수 있느냐가 중요했다. 오프라인에서 시뮬레이션 했을때 성능이 우수했던 모델이 온라인 테스트에서 성능이 나쁘게 나오면 시뮬레이션을 한 의미가 없기 때문이다. 시스템적으로 온라인 테스트 환경을 오프라인에서 완벽하게 재현하는 것은 어려운 작업이었고, 우리는 온·오프라인 실험 결과의 차이를 줄여나가기 위해 여러가지 실험을 하였다. 신뢰할 수 있는 오프라인 시뮬레이터 개발에 성공했으며, 오프라인 시뮬레이션은 이번 모델 성능 개선에 매우 중요한 역할을 했다.

오프라인 테스트를 통해 성능이 검증된 모델은 실서비스에서 A/B 테스트를 통해 최종 검증 단계를 거치게 된다. A/B테스트를 통해 무작위로 뽑힌 10~30%의 사용자는 실험 모델의 예측값을 받게 되며, 이 사용자의 실직적인 피드백(경로 이탈*12, 고객 불만)과 내부적 품질 지표의 상대적인 변화를 확인해 전체 서비스 반영 여부를 결정 한다.

결과: 도로별 속도 에러 개선

GBT 알고리즘으로 인한 교통 패턴별 속도의 상대 에러(mean relative error)는 14~30% 개선되었다. 속도의 상대 에러를 측정한 이유는 위에서 설명한 것처럼 고속과 저속에서의 에러의 비중을 구분하기 위해서다. 자세한 내용은 [표 2]을 참고하기 바란다.

[표 2] 교통 패턴별 에러 개선율

통행량/날짜	월	화	수	목	금	토	일
많음	25.78%	28.27%	27.27%	27.75%	28.73%	29.98%	25.99%
중간	14.61%	14.27%	14.33%	15.13%	15.19%	14.39%	14.12%
적음	18.34%	18.35%	17.93%	18.19%	18.24%	19.75%	20.04%
갈림길	17.29%	18.99%	20.94%	20.67%	20.74%	20.52%	31.14%

소요 시간 오차는 mean absolute error를 기준으로 측정했다. end-to-end 소요 시간 오차는 이상적인 모델을 사용하더라도 0이 될 수 없다. 같은 도로, 같은 시각에서도 다양한 속도가 관측되기 때문이다. 우리는 오프라인 시뮬레이션을 통해 정답을 모두 알고 있는 모델을 이용해서 에러 개선치의 최대값을 확인하였고, GBT 모델 적용으로 인해 그 중 28% 개선을 이루었다.

앞으로

위의 실험으로 도착예정 시간 계산에 머신러닝을 적용해서 성능을 개선하였다. 이러한 개선은 아직 시작에 불과하다. 카카오내비는 도로의 특성, 도로와 도로의 관계를 파악하여, 상습 정체구역의 원인을 더욱 깊이 분석하고, 빠르게 진화하는 기술을 도입하여 도착시간 예측 시스템을 더욱 발전시킬 계획이다. 카카오내비는 고도화된 시스템이 카카오모빌리티의 여러 서비스에 적용되어 사용자에게 더욱 가치 있는 길안내와 정보가 제공되는 미래를 목표로 한다.

*1 참고 | 데이터는 카카오내비에서 제공 *2 논문 | Chen, T. & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. doi : arXiv:1603.02754 *3 참고 | 카카오내비의 전신은 ‘김기사’이다. 2015년 5월 카카오에서 김기사를 인수했다. *4 참고 | https://www.kaggle.com/c/rossmann-store-sales *5 참고 | http://blog.kaggle.com/2015/12/21/rossmann-store-sales-winners-interview-1st-place-gert/ *6 참고 | https://www.kaggle.com/c/rossmann-store-sales/discussion/18024 *7 참고 | https://ttvand.github.io/Second-place-in-the-Santander-product-Recommendation-Kaggle-competition/ *8 참고 | https://github.com/ttvand/Santander-Product-Recommendation/ *9 참고 | 보도와 차도가 구분되지 않은 좁은 도로를 말한다. *10 논문 | Chen, T. & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. doi : arXiv:1603.02754. *11 참고 | 버킷 테스트 방식이란 일부 그룹(bucket) 에게만 새로운 기능을 적용하여 반응을 살펴보는 테스트이다. 일부 그룹에만 적용하기 때문에 문제 발생시 전체 성능을 크게 떨어뜨리지 않는 장점이 있다. *12 설명 | 내비가 안내한 경로로 주행하지 않고 다른 길로 주행