



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

공학석사 학위논문

하둡에 기반한 빅데이터 관리 시스템 구축 및 사례 연구

Building a Big Data Management System
Based on Hadoop and Case Study

지도교수 박휴찬

2020년 2월

한국해양대학교 대학원

컴퓨터공학과
김정래

본 논문을 김정래의 공학석사 학위논문으로 인준함.

위원장 신 옥 근



위 원 이 장 세



위 원 박 휴 찬



2019년 12월 26일

한국해양대학교 대학원

목 차

Abstract	iii
제 1 장 서 론	1
제 2 장 빅데이터의 이해	
2.1 빅데이터 개념	3
2.2 빅데이터 현황	4
2.3 빅데이터 목적	6
2.4 빅데이터 구현 기술	7
제 3 장 빅데이터 관리 시스템 구축	
3.1 시스템 아키텍처	10
3.2 수집 프로그램	11
3.3 적재 프로그램	12
3.4 탐색 프로그램	15
3.5 분석 프로그램	18
3.6 구축환경	20

제 4 장 구축한 시스템의 적용 사례	
4.1 적용 사례	21
4.2 수집 레이어	23
4.3 적재 레이어	25
4.4 탐색 레이어	26
4.5 분석 레이어	31
제 5 장 결론 및 향후 연구	42
참고문헌	43



Building a Big Data Management System Based on Hadoop and Case Study

Jeong-Rae Kim

Department of Computer Engineering,
Graduate School of Korea Maritime and Ocean University

Abstract

데이터 생성량의 증가로 인해 다양한 분야에서 빅데이터의 활용에 대한 필요성이 커지고 있다. 그중 IOT 기반 데이터는 센서로부터 실시간으로 대량의 데이터를 수집하기 때문에 데이터 생성량이 큰 비율을 차지하고 있으며, 이를 활용할 수 있는 빅데이터 시스템의 필요성이 커지고 있다. 본 논문에서는 대용량 로그 데이터를 처리, 빅데이터 분석할 수 있는 빅데이터 시스템을 설계 및 구축하였다. 임의로 생성된 스마트카 상태 데이터를 주축으로 빅데이터의 수집, 적재, 탐색, 분석 과정에서 사용되는 프로그램들의 역할과 기능을 설명하고, 하둡을 중심으로 데이터 처리 과정을 분석하여 최종적으로는 추천 및 군집 분석 등 머신러닝 기술을 적용하였다.

제 1 장 서 론

최근 개인용 스마트 기기의 보편화와 다양한 분야로부터 디지털 데이터에 접근하는 기회가 많아지고, 데이터를 생성하는 주체가 증가함에 따라 빅데이터 관련 산업에 대한 관심이 급증하고 있다. 맥킨지에 따르면 빅데이터가 생산성, 혁신, 경쟁력의 핵심요소로서 의료 및 공공행정 등의 6대 분야에서 6천억 달러 이상의 가치를 창출할 것이며, 미래 글로벌 비즈니스 지형을 바꿀 3가지 기술로 스마트자산, 클라우드, 그리고 빅데이터라 하였다. [1]

빅데이터가 사회적 이슈로 등장한 이유는 IT를 활용한 다양한 산업분야에 활용할 수 있는 대용량의 데이터가 축적되어 있고, 가공되지 않은 데이터의 활용과 유용성 등의 데이터 가치가 무궁무진하기 때문이다. [2] 그러나 우리나라는 아직 기술적인 측면에서 세계 수준에 비교하여 뒤처져 있으며, 관련 기술자들도 매우 부족한 상황이다.

국내 기업의 빅데이터 도입현황을 살펴보면, 포털사와 이동통신사 등 소수의 대기업들이 자사가 보유한 데이터를 바탕으로 빅데이터 서비스 제공을 시작하는 초기 단계이며, 소셜 분석, 시각화 기술, 데이터 관리 등 분야별 전문기업들이 등장하고 있다. 향후 IT 패러다임이 클라우드 컴퓨팅 중심으로 변화하고, 빅데이터의 가치가 중요해질 전망이기 때문에 이에 대비한 국내 플랫폼 기술의 연구개발과 개발자 생태계의 활성화가 시급한 시점이다.

이에 본 논문에서는 빅데이터의 처리와 연구를 시작하는 단계에서 저렴한 비용과 실용적인 환경으로 하둡, SQL과 같은 빅데이터 처리 기술을 이용할 수 있는 빅데이터의 병렬분산처리환경 구축을 위한 실용적인 학습 시스템을 설계 및 구현하였다. 사용되는 데이터는 자동차 내부의 엔진 및 브레이크 등 고장을 판별할 수 있는 장비들의 정보를 사용하며, 빅데이터 처리 과정 단계에서 데이터가 어떤 방식으로 처리되는지 과정을 보여준다. 추가적으로 마스터 데이터와 구매 이력 데이터를 입력하여, 머신러닝 기술을 적용할 수 있는 새로운 데이터

셋을 생성한다.

본 논문의 구성 및 주요 내용은 다음과 같다. 제 2 장에서는 빅데이터 관련 현황 및 기술들을 설명하여 본 논문의 주요 목적인 빅데이터 시스템 구축과 사례분석에 관한 필요성을 설명한다. 제 3 장에서는 앞서 언급한 필요성에 기반한 빅데이터 시스템 구축을 위한 소프트웨어 아키텍처와 활용하게 될 각 처리 과정 별 프로그램들의 기능들을 설명한다. 제 4 장에서는 빅데이터 시스템에서 작동하는 각 프로그램들이 직접적인 데이터 처리 과정을 데이터셋과 연관지어 분석 결과값을 보여준다. 마지막 제 5 장에서는 본 논문의 결론 및 향후 연구 내용을 제시한다.



제 2 장 빅데이터의 이해

2.1 빅데이터 개념

스마트 기기, SNS, 사물인터넷의 확산으로 시작된 빅데이터가 현대인의 생활 요소에 직간접적으로 큰 영향을 주면서 중요한 사회적 현상에 빅데이터가 빠지지 않고 등장하고 있다. 이는 불과 몇 년 전 많은 전문가들이 예견했던 빅데이터의 시대가 현실로 다가왔음을 의미한다.

2010년 인터넷/모바일 시대를 기점으로 데이터의 양이 폭발적으로 증가했고, 2016년부터는 사람, 사물, 정보가 하나로 연결되는 초연결의 시대, 즉 4차 산업혁명이 시작됐다. 4차 산업혁명은 인공지능, 사물인터넷, 무인자동차, 로봇산업 등으로 메인스트림이 만들어지는데, 이때 필요한 핵심 기반 기술로 모두 빅데이터를 주목하고 있다. 빅데이터의 시대는 데이터를 단순 정보로만 보지 않는다. 과거로부터 현재까지 쌓인 데이터를 분석해 현재를 이해하고 이 정보에서 만들어지는 다양한 패턴들을 해석하며 미래를 예측하는 것을 목적으로 한다. 이를 통해 조직의 중요한 의사결정에 빅데이터가 활용되면서 빅데이터가 단순히 대규모의 데이터 집합에서 기술, 분석, 통찰력까지 총칭하는 용어로 사용되고 있다.

Fig 1은 빅데이터의 의미와 목적을 담고 있다. 2011년 메타그룹의 애널리스트인 더그 레이니는 당시 정립되지 않은 빅데이터의 정의를 3V라는 표현으로 매우 명확하게 정리했는데, 이는 데이터의 크기(Volume), 데이터의 입출력 속도(Velocity), 데이터 종류의 다양성(Variety)이라는 세 개의 차원으로 빅데이터를 정의한 것이다. 이후 IBM이 진실성이라는 요소를 더해 4V를 정의했고, 이후에 시각화와 가치가 추가로 정의되면서 6V까지 확장됐다. 빅데이터에 대한 정의는 방대한 크기(Volume)의 다양한(Variety) 데이터들이 빠른 속도(Velocity)로 발생하고 있으며, 빅데이터는 3V(Volume, Variety, Velocity)를 수용하여, 데이터의 진실성(Veracity)을 확보하고, 분석 데이터를 시각화(Visualization)함으로써 새로운 효익을 가져다줄 가치(Value)를 창출하는 것으로 정의할 수 있다.

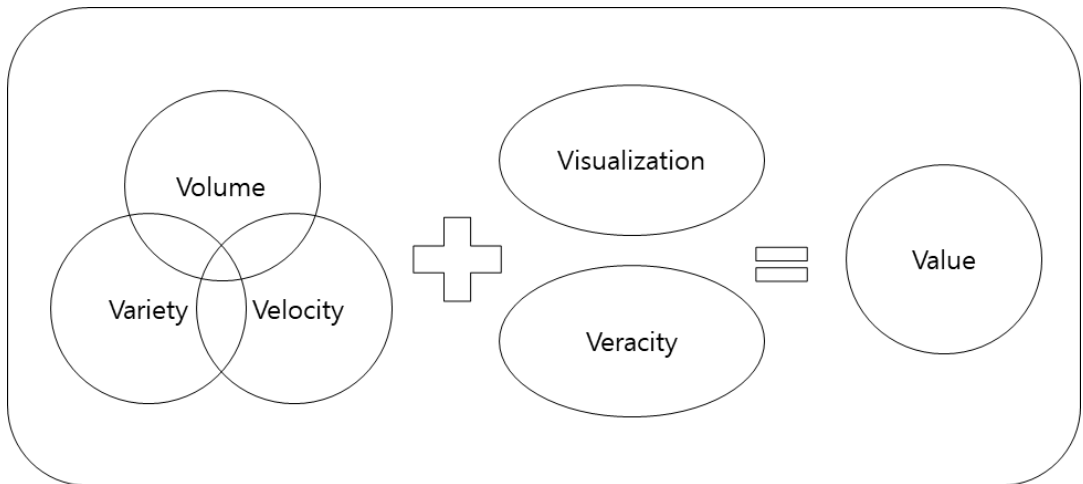


Fig 1 빅데이터의 6V

2.2 빅데이터 현황

최근 2년 동안 발생한 데이터가 전 세계 데이터의 80%를 차지한다고 하며, 향후 지구상에서 발생하는 데이터의 양은 Fig 2와 같이 2025년까지 현재보다 10배 늘어난 163 ZB 수준까지 증가할 것으로 보인다. 이 가운데 80%가 빅데이터 분석이 필요한 비정형 데이터로 만들어질 것으로 예측하고 있다.

빅데이터의 중요성이 부각 되면서 관련 시장도 큰 폭으로 성장하고 있다. 빅데이터가 기존 시스템들이 기술적 한계로 해결하지 못한 수 많은 현안들을 해결하기 시작하면서 그 활용범위가 넓어졌고, 빅데이터의 엄청난 체계모니를 선점하기 위한 전 세계 국가와 기업들간의 경쟁이 치열해지고 있다. Fig 3을 보면 자사의 이익을 위한 기업들의 적극적인 투자로 데이터 생성량의 비율이 가장 높은 것을 확인할 수 있다. 국내 빅데이터 시장은 2020년까지 90억 달러 규모로 성장할 것으로 예상하고 있으며, 매년 20% 이상의 높은 성장세를 보일 것으로 전망된다.

지난 2015년 빅데이터의 국내 시장조사 현황을 보면 아직은 초기시장 형성 단계로 보고 있다. 하드웨어가 50% 이상을 점유하고 있으나 빅데이터의 핵심

경쟁력은 소프트웨어 및 서비스 영역은 상대적으로 적은 점유율을 보이고 있기

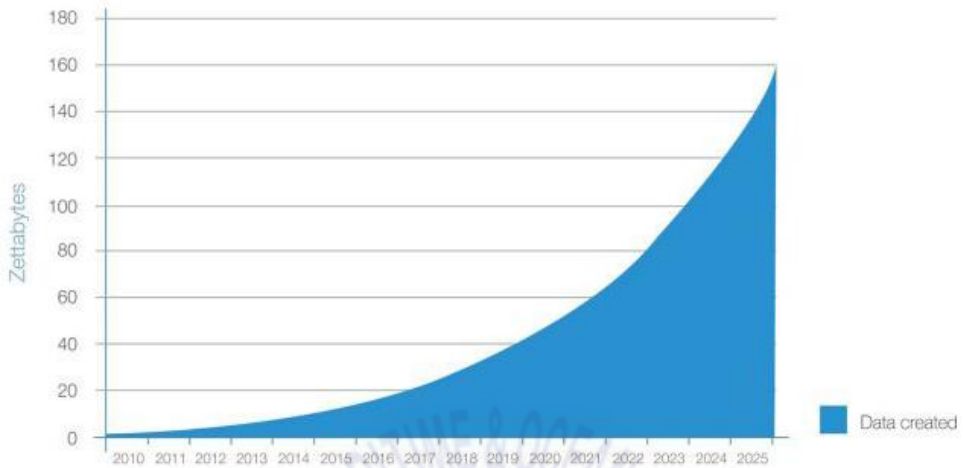


Fig 2 연도별 데이터 생성량

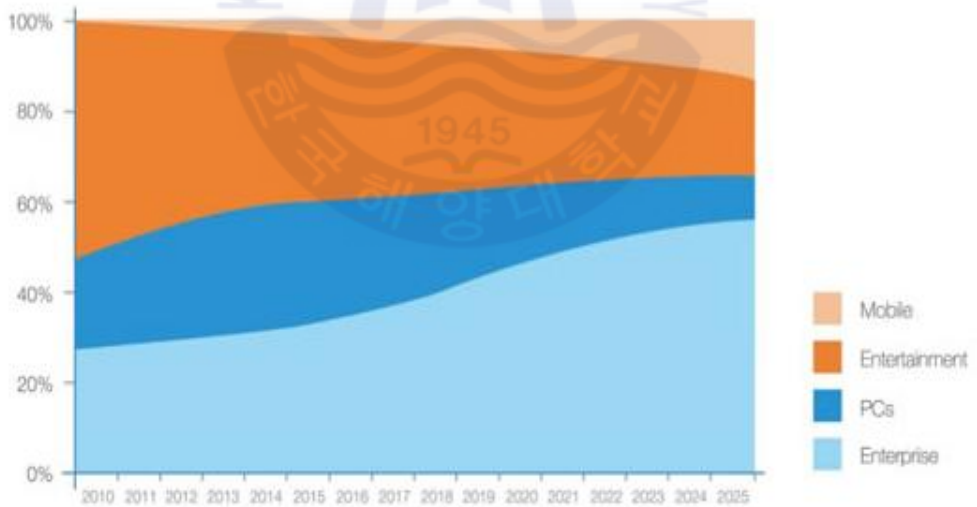


Fig 3 데이터 생성 위치

때문이다. 국내 빅데이터 도입 사례를 보면 일부 사이트를 제외하며 대부분은 실험적인 PoC 수준에서 빅데이터 도입이 추진됐고, 조사 대상 기업 중 빅데이

터 도입에 대해 논의조차 없음이 67.8%로 조사되어 정부와 기업들의 적극적인 투자와 지원이 필요한 상황이다. 빅데이터의 세계 시장 규모를 보면 2026년까지 850억 달러 규모로 전망하고 있고, 2020년을 기점으로 600억 달러로 커지면서 국내 대비 약 60배가 넘는 시장 규모로 보고 있다. 특히 선진국에서는 ICT 글로벌 경쟁력을 강화하기 위해 빅데이터의 사업과 R&D를 국가적 차원에서 지원하고 있어 국가 간의 빅데이터 기술 격차가 크게 벌어지고 있는 상황이다. 2013년 해외 빅데이터 시장의 분야별 점유율을 보면 소프트웨어(22%)와 서비스(40%) 분야가 60% 이상을 차지하고 있어 이미 빅데이터 인프라 구축을 완료하고 빅데이터의 응용 서비스 단계로 시장이 전환된 것을 확인할 수 있다. 하드웨어 중심의 국내 데이터 시장과는 다소 상반된 모습이다. [3]

2.3 빅데이터 목적

빅데이터 전문기업의 화려한 기술과 마케팅에 그 목적을 잊어버리기 쉬우나 결국에는 다른 IT 시스템과 마찬가지로 기존의 문제점을 개선해 비용을 절감하거나 새로운 사업 모델을 만들어 수익을 창출하기 위해서이다. Fig 4를 보면 조직은 성공적인 빅데이터 시스템의 구축과 운영 시 반드시 내재화하고 자산화해야 할 세 가지 요소가 있는데, 바로 사람, 기술, 데이터이다. 이 가운데 빅데이터의 목적 달성에 가장 기본적이면서 중요한 것이 바로 데이터이다. 사람과 기술은 초기엔 외부 파트너에 의존하며 시간을 두고 확보해 나갈 수 있지만 데이터가 없을 때는 대안이 없기 때문이다. 이러한 빅데이터 시스템에서 많이 오해하는 것 중 하나가 빅데이터 시스템이 직접 인사이트와 가치를 만들어 낼 것이라는 기대이다. 하지만 빅데이터 시스템은 경영진 또는 주변 시스템에 신뢰성 있는 정보를 제공해서 인사이트를 갖게 만들고, 빠르게, 의사결정을 내릴 수 있도록 지원하는 정보 제공 시스템일 뿐이다.

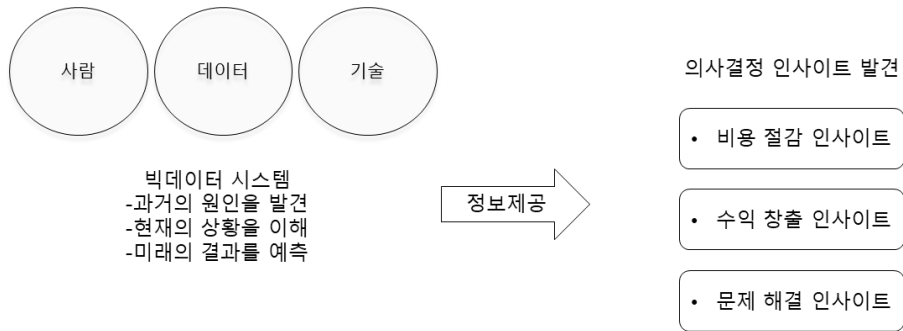


Fig 4 빅데이터의 3가지 요소 및 인사이트 도출

2.4 빅데이터 구현 기술

초기 빅데이터 기술은 낮은 비용의 스토리지를 구축하기 위한 솔루션으로 인식됐다. 하지만 빅데이터가 기존 RDBMS의 기술적 한계를 수행하지 못했던 대규모 작업들을 낮은 비용으로 완수하기 시작했다. 이어서 머신러닝, 텍스트 마이닝 등 고급 분석을 통해 금융, 의료, 방송, 제조, 통신 등 다양한 산업 분야에 관여하며 가치를 만들어 내자 빅데이터 기술을 단순히 스토리지 기술이 아닌 이머징 기술로 주목하기 시작했다. 가트너의 2014년 하이프 사이클을 보면 빅데이터 기술이 빠르게 각성기에 진입했는데, 이는 빅데이터의 거품이 빠지고 시장에서는 메인 플레이어만이 살아남고 잠재적 플레이어가 새롭게 등장하는 단계로 빠르게 이동했음을 시사한다.

빅데이터 아키텍처는 역할별로 수집, 적재, 처리 및 탐색, 분석 및 응용이라는 6개의 레이어로 나눌 수 있고, 각 단계별로 주요 역할 및 기술은 Fig 5와 같다. 구축 순서도 나열한 순서 그대로 진행되며, 이 가운데 처리 및 탐색, 분석 및 응용 과정은 필요 시 반복 진행하면서 데이터의 품질과 분석 수준을 향상 시킬 수 있다.

단계	역할	활용 기술
수집	<ul style="list-style-type: none"> • 내·외부 데이터 연동 • 내·외부 데이터 통합 	Crawling, FTP, Open API RSS, Log Aggregation, DB Aggregation Streaming
적재	<ul style="list-style-type: none"> • 대용량/실시간 데이터 처리 • 분산 파일 시스템 저장 	Distributed File, No-SQL Memory Cached Message Queue
처리	<ul style="list-style-type: none"> • 데이터 선택, 변환, 통합, 축소 • 데이터 워크플로 및 자동화 	Structured Processing UnStructured Processing WorkFlow, Scheduler
탐색	<ul style="list-style-type: none"> • 대화형 데이터 질의 • 탐색적 Ad-Hoc 분석 	SQL Like Distributed Programming Exploration Visualization
분석	<ul style="list-style-type: none"> • 빅데이터 마트 구성 • 통계 분석, 고급 분석 	Data Mining Machine Learning Analysis Visualization
응용	<ul style="list-style-type: none"> • 보고서 및 시각화 • 분석 정보 제공 	Data Export/Import Reporting Business Visualization

Fig 5 빅데이터 단계별 핵심 기술

수집 기술은 조직의 내외부에 있는 다양한 시스템으로부터 원천 데이터를 효과적으로 수집하는 기술이다. 빅데이터 수집에는 기존의 수집 시스템(EAI, ETL, ESB 등)에서 다뤘던 데이터보다 더 크고 다양한 형식의 데이터를 빠르게 처리해야 하는 기능이 필요한데, 이 때문에 빅데이터 수집 아키텍처는 선형 확장이 가능하면서 분산 처리가 가능한 형태로 구성된다.

적재 기술은 수집한 데이터를 분산 스토리지에 영구 또는 임시 저장하는 기술이다. 빅데이터 분산 저장소로 크게 4가지가 있다. 대용량 파일 전체를 영구적으로 저장하는 HDFS, 대규모 메시징 데이터를 영구 저장하기 위한 NoSQL, 대규모 메시징 데이터의 일부만 임시 저장하기 위한 인메모리 캐시, 대규모 메시징 데이터 전체를 버퍼링 처리하기 위한 Message Oriented Middleware가 있다.

처리/탐색 기술은 대용량 저장소에 적재된 데이터를 분석에 활용하기 위해

데이터를 정형화 및 정규화하는 기술이다. 탐색적 분석에는 SQL on Hadoop이 주로 사용되며, 대화형 애드혹 쿼리로 데이터를 선택, 변환, 통합, 축소 등의 작업을 수행한다. 또한 정기적으로 발생하는 처리/탐색의 과정들은 워크플로로 프로세스화해서 자동화하고, 작업이 끝나면 데이터셋들은 특화된 데이터 저장소 (D/W, Mart)로 옮겨진다.

분석/응용 기술은 대규모 데이터로부터 새로운 패턴을 찾고, 그 패턴을 해석해서 통찰력을 확보하기 위한 기술이다. 빅데이터 분석은 활용 영역에 따라 통계, 데이터마이닝, 텍스트마이닝, 소셜미디어 분석 등 다양하게 분류된다. 분석/응용 기술로는 임팔라, 제플린, 머하웃, R, 텐서플로를 다루며, 군집, 분류, 회귀, 추천 등의 고급 분석 영역까지 확장할 수 있다. 마지막으로 스킵을 응용해서 외부 RDBMS에 데이터를 제공한다.



제 3 장 빅데이터 관리 시스템 구축

3.1 시스템 아키텍처

Fig 6은 빅데이터 시스템 구축을 위한 소프트웨어 아키텍처이다. 표기된 메인 오픈소스 소프트웨어는 총 10개를 사용한다. 각 오픈소스 프로그램은 크게 수집, 적재, 처리 및 탐색, 분석 및 응용 영역으로 분류될 수 있다. 이미 많은 빅데이터 시스템에 활용되고 있는 소프트웨어 아키텍처는 크게 2개의 영역으로 나뉘어서 설명할 수 있는데, 하둡을 중심으로 앞쪽을 수집/적재(전처리) 영역, 하둡 뒤쪽을 탐색/분석(후처리) 영역으로 나누어진다. [4]

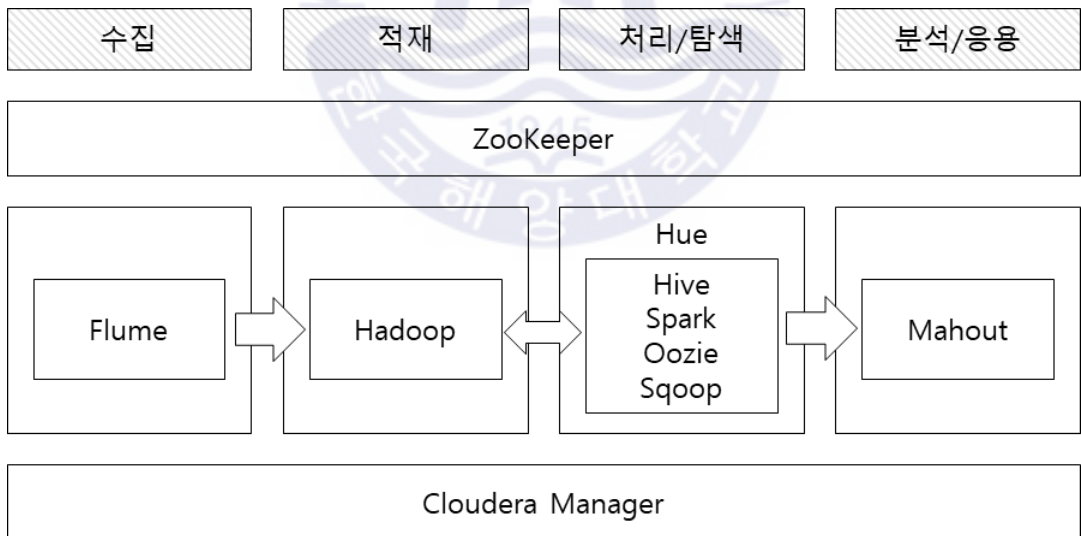


Fig 6 소프트웨어 아키텍처

분석 대상의 데이터를 수집하기 위해 플럼을 사용하고, 적재대상은 하둡이며, 대용량 로그 형식의 파일은 수집과 동시에 플럼에서 하둡으로 적재된다. 하둡에 적재된 데이터는 하이브를 이용해 정제/변형/통합/분리/탐색 등의 작업을 수행하고, 데이터를 정형화된 구조로 정규화해 데이터 마트를 만든다. 그리고 가공/분석된 데이터를 외부로 제공하기 위해 스콧을 이용하며, 필요 시 분석/응용 단계에서도 사용한다. 이러한 처리/탐색 프로세스는 데이터의 품질을 높이는 단계로서, 길고 복잡한 과정을 거치게 되는데, 이때 우지의 워크플로로 프로세스를 구성해 복잡도를 낮추고 자동화할 수 있다. 처리/탐색 과정에서는 데이터를 정규화하고, 더 나아가 데이터 마트를 구성한다. 머하웃에서는 하둡에 적재된 데이터를 이용해 분석 대상에 대한 군집 분석, 분류/예측 분석 등을 통해 다양한 패턴을 찾을 수 있다.

3.2 수집 프로그램

빅데이터 구축은 수집에서부터 시작된다. 빅데이터 프로젝트에서는 여러 공정 단계가 있는데, 그중 수집이 전체 공정 과정의 절반 이상을 차지한다. 빅데이터 수집은 일반적인 수집과 달리 수집 영역이 조직 내의 전체 시스템에서부터 외부 시스템에 이르기까지 매우 광범위하고 다양하다. 플럼은 빅데이터를 수집할 때 다양한 수집 요구사항들을 해결하기 위한 기능으로 구성된 소프트웨어이다. 데이터를 원천으로 수집할 때 통신 프로토콜, 메시지 포맷, 발생 주기, 데이터 크기 등의 기능과 아키텍처를 제공한다. 플럼의 메커니즘은 Source, Channel, Sink만을 활용하는 Fig 7과 같은 단순한 구조이다. 플럼의 Source에서 데이터를 로드하고, Channel에서 데이터를 임시 저장해 놓았다가, Sink를 통해 목적지에 최종 적재한다. 이러한 메커니즘을 기반으로 플럼은 수집 요건에 따라 다양한 분산 아키텍처 구조로 확대할 수 있다.

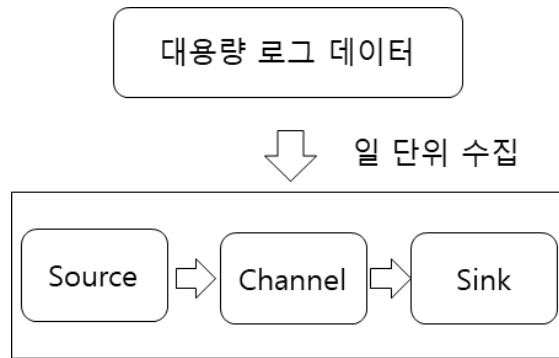


Fig 7 플럼의 수집 과정

3.3 적재 프로그램

적재 과정은 앞서 수집된 데이터를 어디에 어떻게 저장할 것인지 결정하는 부분이다. 수집한 데이터는 데이터의 성격에 따라 처리 방식과 적재 위치가 달라질 수 있다. 크기는 데이터의 발생 주기에 따라 일괄 배치성 데이터인지, 실시간 스트림 데이터인지를 판단해야 하고, 데이터의 형식에 따라 가공처리나 사전 검증 작업을 할 것인지도 판단해야 한다. 적재한 데이터를 어떤 비즈니스 요건에서 활용 하는냐에 따라 적재 대상 위치가 달라질 수도 있는데, 이는 데이터 적재 후 데이터 분석 방식과 데이터 활용성 및 연관된 업무 시스템의 성격에 따라 적재 저장소가 분산 파일, NoSql, 메모리 캐시 등으로 달리 구성되어야 함을 의미한다.

하둡은 이미 보편화된 빅데이터의 핵심 소프트웨어이다. [5] 빅데이터의 에코 시스템들은 대부분 하둡을 위해 존재하고 하둡에 의존해서 발전해 가고 있다 해도 과언이 아니다. 하둡은 크게 두 가지 기능이 있는데, 첫 번째가 대용량 데이터를 분산 저장하는 것이고, 두 번째는 분산 저장된 대용량 데이터를 분석하는 기능이다. 이 가운데 대용량 데이터 처리를 위해 분산 병렬 처리 기술을 사용하는데, 분산 컴퓨팅 기술은 하둡이 처음 개발되기 시작한 2005년 이전부터 이미 사용되어왔지만 높은 투자 비용으로 특정 분야에서만 활용되고 있었다. 분산 병렬 처리에서의 기본은 수많은 컴퓨터에 분산 저장되어있는 데이터로부터

터 어떻게 효율적으로 일을 나눠서(Map) 실행시킬 수 있는냐고, 다음으로 수많은 컴퓨터가 나눠서 실행한 결과들을 어떻게 하나로 모으냐(Reduce)는 것이다. 이를 쉽고 편리하게 지원하는 프레임워크가 Fig 8과 같은 하둡의 맵리듀스이다. [6]

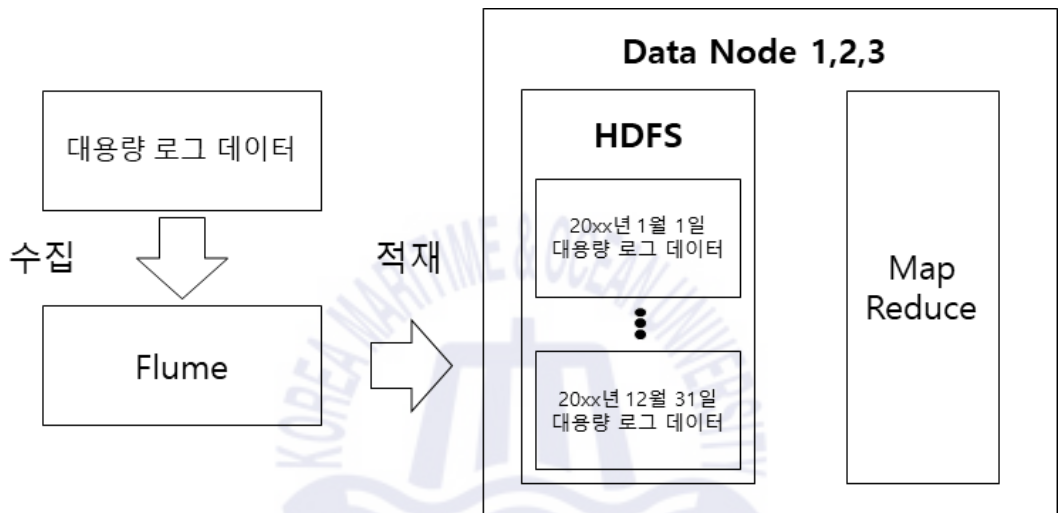


Fig 8 HDFS 대용량 로그 데이터 적재

수십~수천 대의 서버에 설치되어 있는 빅데이터 분산 환경을 더욱 효율적으로 관리하기 위해서는 서버 간의 정보를 쉽고 안전하게 공유해야 한다. 공유된 정보를 이용해 서버 간의 중요한 이벤트를 관리하면서 상호작용을 조율해 주는 코디네이터 시스템이 필요한데, 이것이 바로 분산 코디네이터인 아파치 주키퍼이다. 주키퍼는 하둡, Hbase, 카프카, 스톱 등의 분산 노드 관리에 사용된다. 주키퍼는 Fig 9와 같이 구성되어 있다. 3대 이상의 홀수 개의 서버로 구성되어야 하며, 그중 반드시 1대는 리더 서버가 되고 나머지는 팔로워 서버가 되어야 한다. 팔로워 서버 1에 저장된 ZNODE 정보는 리더 서버에 전달되고, 리더 서버는 다른 모든 팔로워 서버에 요청받은 ZNODE 정보를 브로드캐스트한다. 본 논문에서는 주키퍼를 직접적으로 사용하지는 않으나 하둡 내부에서 주

키퍼를 활용해 클러스터 멤버십 기능 및 환경설정의 동기화 등을 사용하고 있어 없어서는 안 될 중요 소프트웨어이다.

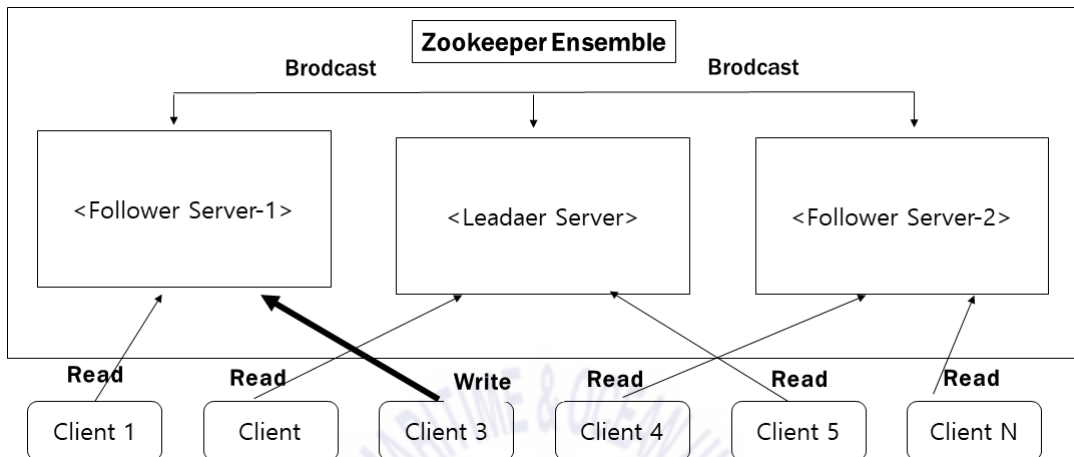


Fig 9 주키퍼 아키텍처

대용량 로그 파일을 적재하기 위한 요건으로서 아키텍처는 다음과 같은 구성으로 진행된다. 플럼의 Source 컴포넌트로 대용량 파일을 읽고 Sink를 이용해 HDFS의 특정경로에 적재하는 구성이다. HDFS에 적재할 때는 데이터의 포맷, 경로, 파티션 값을 신중하게 설정해야 하는데, 적재된 데이터 형식에 따라 뒤에서 이어질 탐색/분석을 위한 후처리 작업량과 복잡도가 커질 수 있기 때문이다. 또한 HDFS에 적재된 데이터는 부분 수정 및 삭제가 어렵기 때문에 초기 적재 레이어는 원천을 그대로 유지하고 2, 3차 가공 작업을 통해 데이터의 품질을 높이는 데이터 설계가 중요하다.

플럼에서 가장 중요한 컴포넌트가 HDFS Sink다. 플럼의 Source에서 읽어들이는 데이터를 하둡에 적재해야 하는데 이때 플럼의 HDFS Sink에서 다양한 옵션과 기능들을 사용할 수 있다. HDFS Sink의 기본 기능은 수집한 데이터를 HDFS의 특정 경로에 적재하는 것이다. 적재할 때 사용될 파일 타입, 파일명, 배치 크기, 생성 파일 크기 등의 정보를 설정할 수 있다. 이때 사용하는 옵션은 주변의 환경과 요구사항에 따라 최적화해야 하는데, 수집되는 데이터 양과 주기, 포

맷, 향후 분석 형태 등을 고려해 설정한다.

HDFS의 적재 경로를 하이브에서 인지할 수 있는 특정한 구분값(날짜, 시간, 코드 등)으로 파티셔닝 한다. 파티션은 주로 날짜별 디렉터리로 만들어 관리하는데, 업무코드 + 날짜를 조합해서 고유한 파티션 경로를 구성한다. 향후 적재한 데이터를 하이브에서 사용하는데, 파티션 디렉터리를 조건으로 데이터 조회 시 전체 파일을 스캔하지 않고 조건에 해당하는 디렉터리를 직접 참조할 수 있어 효율성이 좋아진다.

3.4 탐색 프로그램

데이터 처리 및 탐색 영역은 적재된 데이터를 가공하고 이해하는 단계이다. 특히 데이터를 이해하는 과정에서 데이터들의 패턴, 관계, 트렌드 등을 찾게 되는데, 이를 탐색적 분석이라고도 한다. 탐색 과정은 분석에 들어가기에 앞서 빅데이터의 품질과 인사이트를 확보하는 매우 중요한 단계임에도 많은 기업들이 이를 무시하고 곧바로 분석단계로 넘어가 빠른 성과 창출을 원하는 경향이 있다. 결국 평이한 빅데이터 분석 결과가 만들어지면서 빅데이터 프로젝트의 위기를 초래한다. 빅데이터 탐색에서는 덩치 큰 비정형 데이터를 정교한 후 처리 작업으로 정형화해서 데이터의 직관성을 확보하고, 업무 도메인에 대한 이해를 바탕으로 충분한 탐색적 분석을 진행했을 때 빅데이터를 통한 미래의 통찰력과 비즈니스 가치의 창출이 가능해진다. 탐색 결과는 곧바로 분석 마트를 위한 기초 데이터로 활용되며, 이러한 일련의 처리/탐색, 분석/응용 과정을 거쳐 빅데이터 웨어하우스가 만들어진다.

Fig 10을 보면 빅데이터 기반 DW는 크게 3개의 영역으로 구성되어 있는데, 이는 전통적인 RDBMS 기반 DW 구성과 크게 다르지 않다. 빅데이터 External 영역은 전처리와 후처리가 만나서 데이터를 서로 공유하는 영역으로, 원천 데이터의 형식을 최대한 유지한다. External의 데이터셋은 처리/가공 단계를 통해 Managed 영역으로 이동하고 데이터의 주제 영역별 처리/탐색 과정을 거치면서 빅데이터 분석용 마트가 최종적으로 만들어진다.

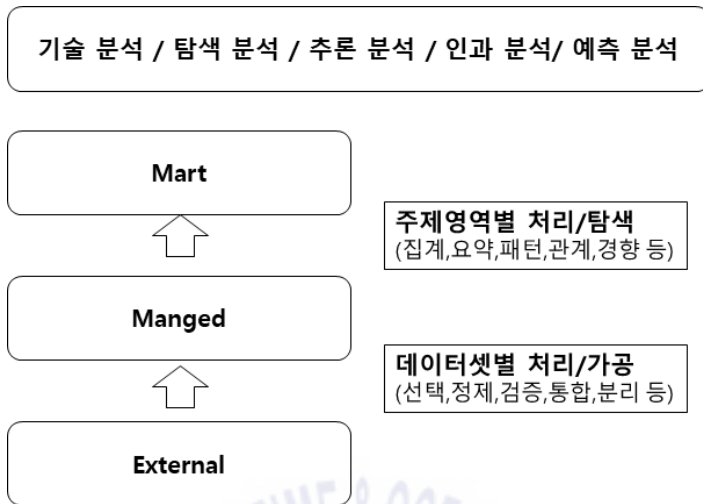


Fig 10 빅데이터 웨어하우스

하둡 초창기에는 적재된 데이터를 탐색/분석하기 위한 도구로 맵리듀스 (MapReduce)를 주로 이용했다. 하지만 맵리듀스는 복잡도가 높은 프로그래밍 기법이 필요했고, 이는 업무 분석가 및 관리자들에게 빅데이터에 접근하는 것을 어렵게 만들었다. 이를 해결하기 위해 페이스북에 SQL과 매우 유사한 방식으로 하둡 데이터에 접근성을 높인 하이브를 개발하였다.

하이브의 아키텍처에서 가장 큰 특징은 하이브 클라이언트에서 작성한 QL이 맵리듀스 프로그램으로 변환되어 실행된다는 것이다. CLI, 웹 콘솔 등을 통해 하이브 QL을 작성하면 쿼리 엔진에 있는 SQL 파서가 하이브 QL을 맵리듀스 프로그램으로 변환하고, 이 맵리듀스 프로그램이 하둡 클러스터에 전송되어 여러 데이터 노드에서 분산 실행된다. 하이브는 복잡한 맵리듀스를 하이브 QL로 래핑해 접근성을 높일 수 있었지만 맵리듀스 코어를 그대로 사용함으로써 성능면에서 만족스럽지 못했다. 그로 인해 반복적인 대화형 연산 작업에서는 하이브가 적합하지 않았다. 이러한 하이브의 단점을 극복하기 위한 하나의 방법으로서 스파크가 채택되었다. [7]

스파크의 가장 큰 특징은 고성능 인메모리 분석이다. 기존 맵리듀스 기반의 하이브 경우 대량 데이터를 처리할 때 디스크에 적재된 데이터를 대상으로 연산을 수행하게 되어 과도한 디스크 I/O가 발생해 수 분에서 수 시간씩 결과를 기다려야 한다. 하지만 스파크는 이러한 단점을 극복하기 위해 데이터 가공 처리를 인메모리에서 수행함으로써 스파크 SQL, 스파크 스트리밍, 머신러닝 등의 대용량 데이터 작업에도 빠른 성능을 보장한다. 또한 스파크는 파이썬, 자바, 스칼라, SQL 등의 클라이언트를 라이브러리를 제공하여 접근성을 높였고, 스파크 코어 연동을 통해 작업 스케줄링 메모리 관리, 장애 복구 및 RDD 등을 관리할 수 있는 API를 제공한다.

스파크는 다양한 클라이언트 프로그래밍 언어를 지원하고, SQL을 이용해 데이터에 접근할 수 있다. [8] 하이브, 스파크 등을 이용한 빅데이터의 처리/탐색하는 과정은 복잡한 선후 과정을 반복적으로 진행된다. 대규모 빅데이터 시스템에서는 수집 및 적재된 수백 개 이상의 데이터셋을 대상으로 다양한 후처리 작업이 데이터 간의 의존성과 무결성을 유지하며 복잡하게 실행된다. 반복적이면서 복잡한 후처리 작업을 처리하기 위해 방향성 있는 비순환 그래프로 작업의 시작, 처리, 분기, 종료점 등의 액션 등을 정의하는 워크플로 역할의 필요해졌고 그 역할을 우지가 수행하게 되었다.

우지 클라이언트에서 작성한 워크플로는 우지 서버에 곧바로 전송되며, 관련 워크플로 메타 정보는 RDBMS에 별도로 관리된다. 우지 서버에 있는 Coordinator는 우지에 등록된 워크플로를 스케줄링해주며, 이때 워크플로 엔진이 Action 노드와 Control 노드의 정보를 해석하면서 관련 태스크를 하둡의 클러스터에 실행시킨다. 본 과정에서 우지를 활용하여 후처리 작업을 정의하고 프로세스화한다. 적재된 데이터를 DW 처리과정을 통해 데이터를 이동시키는 과정에서 다양한 하이브 SQL들이 이용되고, 이를 약속된 시간에 따라 스케줄링해서 실행해야 하는데 이때 우지의 워크플로를 활용한다.

빅데이터 탐색/분석은 장기간의 반복 작업이면서 그 과정에 있어 많은 도구들이 활용된다. 주로 하둡을 기반으로 하이브, 피그, 우지, 스크립, 스파크 등이 해당되며 이를 접해보지 못한 일반 분석가 또는 업무 담당자들이 직접 사용하

기에는 어려움이 많다. 빅데이터 기술이 성숙해지면서 이러한 기술의 복잡도를 숨기고 접근성과 편의성을 높인 소프트웨어들이 만들어 졌는데, 그중 하나가 바로 클라우드에서 만든 휴이다. 휴는 다양한 하둡의 에코시스템의 기능들을 웹 UI로 통합 제공한다. 휴는 하둡 에코시스템들을 통합하기 위해 자체 플러그인을 설치하거나 API를 연동해서 에코 시스템들의 주요 기능들을 웹 UI로 제공한다. 휴의 데이터베이스에서 휴에 로그인하는 사용자의 계정 관리와 휴에서 사용할 컴포넌트(잡, 하이브, 우지 등)의 메타 정보를 관리한다.

3.5 분석 프로그램

빅데이터 탐색 단계가 데이터를 관찰하고 이해는 과정이라면 빅데이터 분석은 탐색과 분석을 반복하며 의미 있는 데이터를 추출해 문제를 명확히 정의하고 해결하는 과정이다.

다양한 분석 기술을 통해 빅데이터의 가치는 “RAW 데이터 정보 통찰력 가치” 순으로 변하게 된다. 이 가운데 통찰력을 갖게 되는 단계에서 빅데이터의 활용 효익이 만들어지기 시작하는데, 주로 상품 및 서비스 개발, 마케팅, 및 캠페인 지원, 리스크 관리 등의 영역에서 주요 의사결정을 내릴 때 빅데이터를 이용한다. 머하웃은 하둡 생태계에서 머신러닝 기법을 이용해 데이터 마이닝을 수행하는 툴이다. 머신러닝을 다루는 프레임워크 머하웃 말고도 다양한 분야에서 발전해 오고 있다. 가장 유명한 R을 비롯해 래피드마이너, 웨카, 텐서플로 등이 이에 해당한다. 하지만 이러한 제품들은 대규모의 데이터셋을 분석할 수 있게 설계 되지 않았고 하둡의 분산 환경에서 실행하기 어려웠다.

머하웃은 2008년 검색엔진 루씬의 서브 프로젝트로 시작됐고 하둡의 분산 아키텍처를 바탕으로 텍스트 마이닝, 군집, 분류 등과 같은 머신러닝 기반 기술이 내재화 되면서 아파치 최상위 프로젝트로 승격됐다. 머하웃은 하둡의 분산환경 위에 맵리듀스를 기반으로 고급 분석을 지원하는 라이브러리 패키지이고, 하둡 클러스터 관점에서 보면 아키텍처는 Fig 11과 같이 매우 단순한 구조이다. 주

요 관련 라이브러리는 추천, 분류, 군집이 있다. RDBMS에 있는 데이터를 특별한 전처리 없이 곧바로 HDFS에 적재하거나, 반대로 HDFS에 저장된 데이터를 RDBMS로 제공해야 하는 경우가 있다. RDBMS와 HDFS 사이에서 데이터를 편리하게 импорт하거나 익스포트해주는 소프트웨어가 스쿱이다. 스쿱의 기본 아키텍처는 스쿱의 CLI로 импорт, 익스포트 명령을 하둡에 전달하면 맵 태스크가 병렬로 실행되어 외부 데이터베이스와 HDFS 사이에서 대량의 데이터를 импорт 및 익스포트할 수 있는 아키텍처를 제공한다.

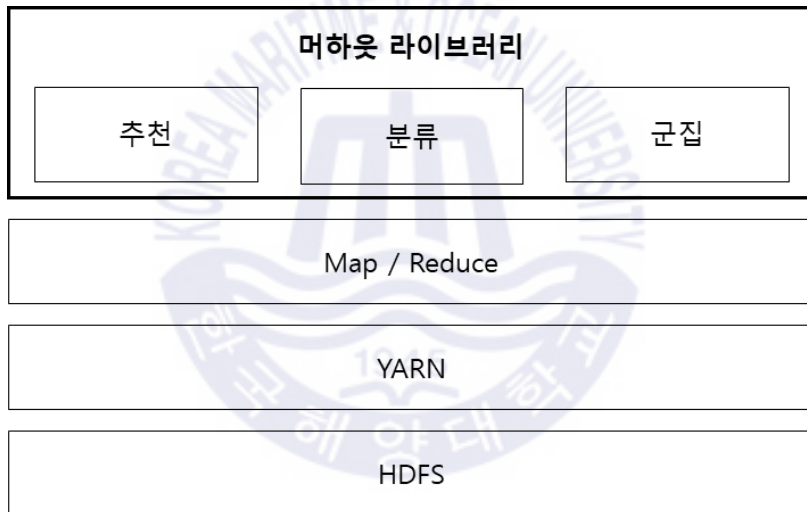


Fig 11 머하웃 아키텍처

3.6 구축 환경

Fig 12은 앞서 설명한 소프트웨어 아키텍처를 만들기 위해 구축한 PC 환경이다. 3대의 가상머신을 만들고, 가상머신에 소프트웨어를 설치하였다. 빅데이터 자동화 관리툴인 클라우데라의 CM(Cloudera Manager)를 이용하여 하둡을 포함한 에코시스템을 설치 및 관리한다.

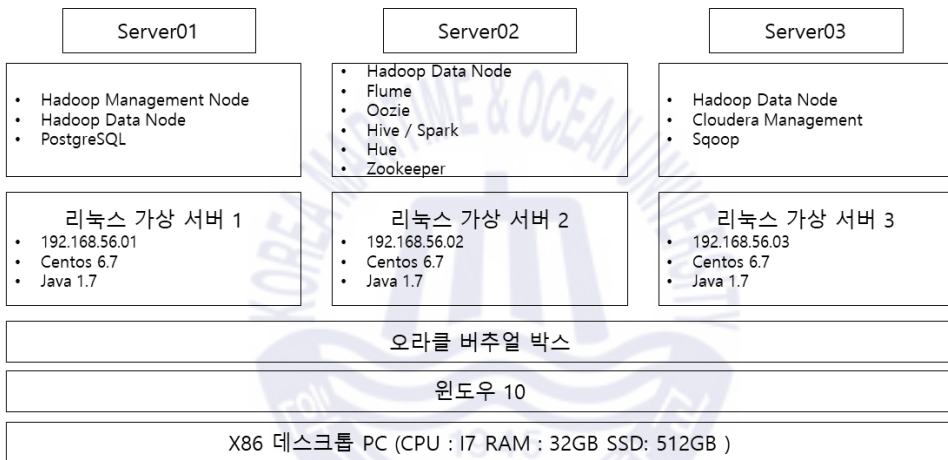


Fig 12 PC 구축 환경

제 4 장 구축한 시스템의 적용 사례

4.1 적용 사례

본 논문에서 다루고자 하는 빅데이터 도메인은 자동차의 첨단 전자장치와 무선통신을 결합한 스마트카 서비스이다. 실제 스마트카를 대상으로 빅데이터 시스템을 진행한다면 앞서 도출된 요구사항을 해결하기 위해 수십~수백 대의 하둡 클러스터 노드를 구성할 필요가 있다. 하지만 대규모 빅데이터 환경을 구성하는 것은 현실적으로 어려울뿐더러 기업에서조차 대규모 투자비가 발생하는 빅데이터 시스템 구축을 선뜻 진행하는 것은 쉽지가 않다. 그래서 개인용 PC를 활용할 수 있는 수준으로 소규모 빅데이터 파일럿 환경을 구성했고, 파일럿 환경에서도 빅데이터 핵심 기술과 기능들을 모두 사용할 수 있도록 아키텍처를 구성했다.

이제 우리가 일상에서 타고 다니는 자동차 안에 컴퓨터, 무선 인터넷, 전자장치들이 설치되어 자동차 안에서 이메일을 주고받고, 인터넷을 통해 각종 정보도 검색한다. 또한 무선 네트워크를 통해 차량을 원격 진단하고 운전 습관, 날씨 교통 정보 등을 분석해서 운전자의 안전과 편의를 도모한다. 국내외 주요 IT 기업들이 최첨단 기술을 이용해 스마트카 산업을 주도하기 시작했으며 무인자동차의 테스트베드 성공이 가시화되고 있다. 스마트카 안에는 수백 개의 IOT 센서가 장착되어 있으며, 자동차의 상태를 모니터링 하면서 수많은 차량 상태 정보를 실시간으로 만들어낸다.

스마트카의 센싱 정보는 네트워크를 타고 중앙의 빅데이터 시스템으로 전송되며, 이 데이터들을 수집 적재 처리 및 탐색 분석 및 응용 단계를 거치면서 운전자에게 편의성과 안전성을 지원하는 스마트카 서비스로 제공된다. 이러한 스마트카에서 발생하는 수많은 데이터로부터 가치와 통찰력을 찾기 위한 빅데이터 시스템을 구축하고자 한다.

본 논문에서는 스마트카의 빅데이터 분석을 위해 ‘차량의 다양한 장치로부터 발생하는 로그 파일을 수집해서 기능별 상태를 점검한다.’ 라는 요구사항을 제시한다. Table 1은 앞으로 사용할 스마트카 데이터의 상태 정보의 상세 내용을 담고 있다.

데이터 발생 위치	100대의 시범 운행 차량
발생 데이터 종류	대용량 로그 파일
데이터 발생 주기	3초
데이터 수집 주기	24시간
데이터 수집 규모	1MB/1대(1일 수집 규모: 약 100MB/100대)
데이터 타입	텍스트(UTF-8), 반정형
데이터 분석 주기	일/주/월/년
데이터 처리 유형	배치
데이터 구분자	coma(,)
데이터 스키마	로그 발생일시, 차량 고유번호, 차량 타이어 상태, 차량 라이트 상태, 차량 엔진 상태, 차량 브레이크 상태, 차량 배터리 충전 상태, 수집 작업 요청일

Table 1 스마트카 데이터 분석

실제 스마트카의 빅데이터 분석 요건은 훨씬 더 많고 복잡하다. 하지만 여기서는 논문의 요구사항과 분석 요건을 파일럿 프로젝트 수준인 100대의 차량으로 한정하고, 스마트카의 수집 정보도 누구나 쉽게 이해할 수 있는 수준으로 단순화했다. 실제로 100대의 스마트카를 운영하면서 파일럿 프로젝트를 수행하는 것은 불가능하므로 스마트카를 시뮬레이션하는 스마트카 로그 시뮬레이터 프로그램을 사용한다.

로그 시뮬레이터로 생성된 스마트카 데이터 수집은 수집 적재 처리 및 탐색 분석 및 응용 프로세스를 거치고, 각 단계마다 파일럿에서 활용하기 쉬운 데이터셋으로 재구성한다. 마지막으로 탐색과 분석, 머신러닝 기법 등을 적용해 분

류예측, 군집, 추천에 이르는 데이터 마이닝 작업까지 진행한다.

4.2 수집 레이어

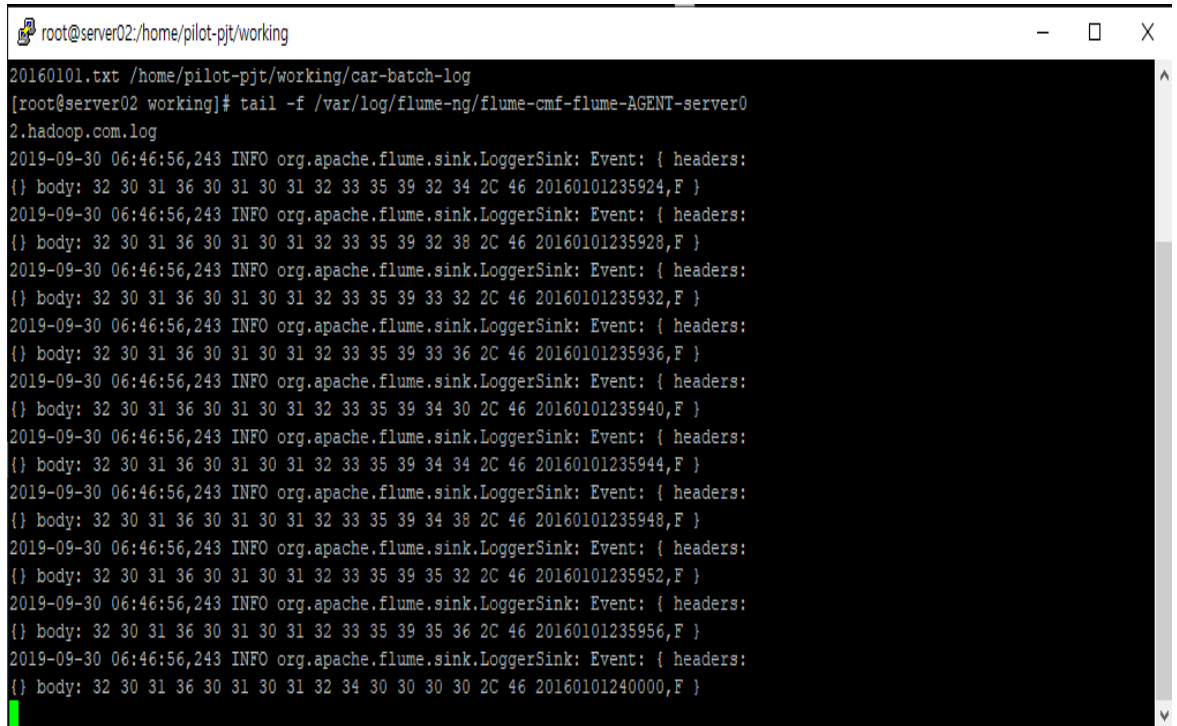
플럼은 스마트카에서 발생하는 로그를 직접 수집하는 역할을 담당한다. 100대의 스마트카에 대한 상태 정보 로그 파일이 로그 시뮬레이터를 통해 매일 생성된다. 이렇게 만들어진 상태 정보 파일을 플럼 에이전트가 일 단위로 수집해서 하둡에 적재하고 향후 대규모 배치 분석에 활용한다. 본 장에서는 수집 과정 내부에서 플럼을 사용하여 데이터 처리 과정을 설명한다.

원천 데이터는 로그 시뮬레이터를 사용하며, 스마트카의 상태 정보 로그를 가상으로 만드는 자바 로그 발생기이다. 스마트카 상태 정보 데이터는 100대 스마트카 장치들의 상태 정보를 3초 간격으로 발생시키며, 1일 100MB의 로그 파일이 만들어진다. 우선 플럼에서 에이전트를 생성한다. 그리고 에이전트에서 사용할 Source, Channel, Sink의 각 리소스 변수를 정의한다. Source 부분에서는 “Spooldir” 을 설정하여 특정 디렉토리를 모니터링하고 있다가 새로운 파일이 생성되면 이벤트를 감지해서 “batchsize” 의 설정값 만큼 읽어서 Channel에 데이터를 전송한다.

Channel의 type은 “memory”, “file” 이 있는데 Memory는 Source로부터 받은 데이터를 메모리상에 중간 적재하므로 성능이 높지만 안정성이 낮다. File은 Source에서 전송한 데이터를 받아 로컬 파일시스템 경로인 “dataDirs” 에 임시 저장했다가 Sink에게 데이터를 제공하므로 성능은 낮지만 안정성이 높다. 최종목적지 type은 Logger로 지정해서 수집한 데이터에 대한 테스트 및 디버깅을 할 수 있다. 마지막으로 각 노드들을 이어주는 에이전트 리소스를 하나로 연결해 준다. Interceptor는 Source와 Channel의 중간에서 데이터를 가공하는 역할을 한다.

플럼의 Source에서 수집되는 데이터 중 일부 데이터를 수정하거나 필요한 데이터만 필터링하는 등 중간에 데이터를 추가/가공/정제 하는데 사용된다. 플럼에서 데이터 전송 단위를 Event라 하는데, Event의 구조는 다시 Header와 메시

본문인 Body로 구성된다. Fig 13를 통해 수집되는 데이터의 형식을 확인할 수 있다.

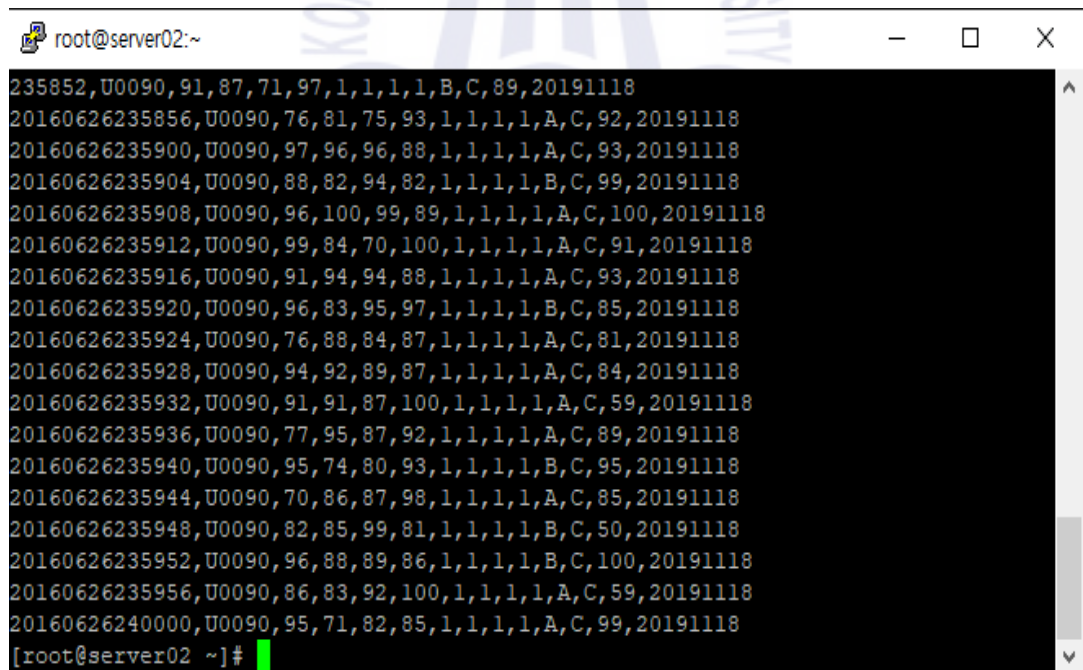


```
root@server02:/home/pilot-pjt/working
20160101.txt /home/pilot-pjt/working/car-batch-log
[root@server02 working]# tail -f /var/log/flume-ng/flume-cmf-flume-AGENT-server0
2.hadoop.com.log
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 32 34 2C 46 20160101235924,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 32 38 2C 46 20160101235928,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 33 32 2C 46 20160101235932,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 33 36 2C 46 20160101235936,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 34 30 2C 46 20160101235940,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 34 34 2C 46 20160101235944,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 34 38 2C 46 20160101235948,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 35 32 2C 46 20160101235952,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 33 35 39 35 36 2C 46 20160101235956,F }
2019-09-30 06:46:56,243 INFO org.apache.flume.sink.LoggerSink: Event: { headers:
{} body: 32 30 31 36 30 31 30 31 32 34 30 30 30 2C 46 20160101240000,F }
```

Fig 13 수집 데이터 확인

4.3 적재 레이어

스마트카 상태 정보 로그는 비교적 큰 크기(100MB 이상)의 파일로서, HDFS의 특정 디렉토리에 일자 단위로 파티션해서 적재한다. 이렇게 일 단위로 장기간 적재된 데이터는 일/주/월/년별로 스마트카의 다양한 시계열 집계 분석을 할 수 있다. 이때 하이브가 활용되고, 분산 병렬 처리 작업을 위해 맵리듀스 프로세스가 내부적으로 작동한다. 하이브로 분석된 결과는 다시 HDFS의 특정 영역에 저장되고, 이 데이터가 스마트카의 고급 분석으로까지 확장해서 사용된다. 적재 과정에서는 수집 과정에서 생성하였던 플럼의 에이전트 내용을 다음과 같이 수정한다. 구성 파일에 Logger Sink의 구성요소를 HDFS Sink로 교체하고, 데이터 가공을 위한 인터셉터를 추가한다. 로그 시뮬레이터를 실행하여, Fig 14의 적재 데이터를 직접 확인할 수 있다.



```
root@server02:~  
235852,U0090,91,87,71,97,1,1,1,1,B,C,89,20191118  
20160626235856,U0090,76,81,75,93,1,1,1,1,A,C,92,20191118  
20160626235900,U0090,97,96,96,88,1,1,1,1,A,C,93,20191118  
20160626235904,U0090,88,82,94,82,1,1,1,1,B,C,99,20191118  
20160626235908,U0090,96,100,99,89,1,1,1,1,A,C,100,20191118  
20160626235912,U0090,99,84,70,100,1,1,1,1,A,C,91,20191118  
20160626235916,U0090,91,94,94,88,1,1,1,1,A,C,93,20191118  
20160626235920,U0090,96,83,95,97,1,1,1,1,B,C,85,20191118  
20160626235924,U0090,76,88,84,87,1,1,1,1,A,C,81,20191118  
20160626235928,U0090,94,92,89,87,1,1,1,1,A,C,84,20191118  
20160626235932,U0090,91,91,87,100,1,1,1,1,A,C,59,20191118  
20160626235936,U0090,77,95,87,92,1,1,1,1,A,C,89,20191118  
20160626235940,U0090,95,74,80,93,1,1,1,1,B,C,95,20191118  
20160626235944,U0090,70,86,87,98,1,1,1,1,A,C,85,20191118  
20160626235948,U0090,82,85,99,81,1,1,1,1,B,C,50,20191118  
20160626235952,U0090,96,88,89,86,1,1,1,1,B,C,100,20191118  
20160626235956,U0090,86,83,92,100,1,1,1,1,A,C,59,20191118  
20160626240000,U0090,95,71,82,85,1,1,1,1,A,C,99,20191118  
[root@server02 ~]#
```

Fig 14 적재 데이터 확인

4.4 탐색 레이어

본 논문에서는 하이브를 스마트카 데이터셋을 다양한 각도로 탐색 및 가공하는데 활용하였다. 하이브 QL로 스마트카 데이터에 대한 조회, 결합, 분리, 변환, 정제 등의 작업을 수행해 스마트카 DW를 구성하고, 다시 DW에서 2,3차 탐색과 고급 분석을 거쳐 스마트카 분석 마트를 만든다. 앞서 수집 및 적재한 데이터가 하이브의 External 영역에 적재되어 있는데, 이를 정제해서 Managed 영역으로 옮기고 주제 영역별 Mart를 구성하는데 하이브가 주로 사용된다.

휴에서는 HDFS, 하이브를 편리하게 사용하기 위한 웹 에디터를 기본적으로 제공한다. 웹 에디터를 통해 “스마트카 상태 데이터”를 직관적으로 탐색할 수 있다. 또한 추가적으로 사용할 데이터셋인 “스마트카 마스터 데이터”, “스마트카 차량 물품 구매 이력 데이터”를 임포트하는 작업도 진행한다. 스파크 셸에서 스파크 SQL API를 이용해 추가로 적재될 “스마트카 마스터” 데이터를 조회 및 정제하는 작업을 추가로 하였다. 스마트카 분석 마트를 만들기 위해서는 3가지 데이터셋 주제 영역별 데이터웨어하우스 작업이 필요한데, 이때는 휴의 Job Designer를 이용해 우지의 워크플로를 3가지로 나누어 작성하고 실행한다.

탐색 레이어에서는 스마트카의 기능별 상태를 점검하기 위해 적재된 데이터를 탐색 및 가공해서 분석하기 쉬운 데이터셋으로 재구성한다. 수집 및 적재된 원천 데이터를 탐색해서 최종 분석 마트 데이터까지 만들어가는 과정을 설명한다. 먼저 External에 적재된 데이터를 휴에서 제공하는 Hive Editor를 이용해 SQL과 유사한 방식으로 조회한다. External에 적재된 데이터를 작업일자 기준으로 후처리 작업을 하며, 하이브의 Managed 영역에는 스마트카로부터 발생한 생성일자를 기준으로 2차 적재한다. 필요 시 데이터를 필터링 및 클린징 처리하거나 통합, 분리 작업을 수행한다. Managed에 만들어진 데이터는 곧바로 분석에 활용된다. External 영역보다 데이터가 구조화됐고, 내용들이 정제되면서 정형 데이터로 전환한다. 이 단계에서 업무 요건에 맞는 필요한 애드혹 조회를 하며 탐색적 분석을 수행한다.

마트영역에서는 개발자 또는 분석가들이 실행한 애드혹 작업들은 많은 시간과 노력을 들여 만들어지는 가치 있는 데이터이다. 이러한 애드혹 작업의 결과는 재사용이 가능하도록 마트화하고, 외부 업무 시스템에 제공할 수 있어야 한다. 이때 우지의 워크플로를 이용해 프로세스화하고 자동화하게 된다. HDFS에 적재된 데이터를 확인한다. 이 데이터는 플럼에서 수집한 데이터를 1차로 적재한 곳이며, 하이브의 External 영역에 해당한다. 여기서 하이브를 이용해 이곳의 데이터를 탐색하기 위해서는 적재된 데이터를 하이브의 테이블 구조와 매핑 시킨 메타 정보가 필요하다.

하이브 쿼리를 이용해 테이블을 생성한다. 쿼리 내용은 앞서 적재한 “스마트카 상태 정보”의 파일 위치, 형식, 구조 등의 정보를 하이브 테이블로 정의하고 있다. 즉, 스키마가 없는 데이터를 하이브의 스키마로 정해 하이브의 메타스토어에 등록하는 작업이다. 추가적으로 작업 일자를 기준으로 파티션 정보를 생성한다. Alter Table 명령을 실행해 작업 일자 기준으로 파티션 정보를 추가한다. 파티션 정보는 하이브의 메타스토어에서 관리 되는데 파티셔닝 대상 데이터가 적재되면 그에 해당하는 하이브 테이블에 Add Partition을 반드시 수행해야 한다. External에 생성된 테이블은 SELECT 쿼리를 실행하여 간단히 조회할 수 있다. Fig 15, 16는 하이브에서 쿼리를 작성하고 조회한 결과를 보여준다.

이제 앞선 설명한 나머지 두 데이터셋을 추가로 적재한다. “스마트카 마스터 데이터”, “스마트카 차량용품 구매이력 데이터”이다. 해당 데이터는 외부 데이터로서 휴 파일 브라우저의 업로드 기능을 이용해 적재하고 하이브의 External Table로 정의한다. 추가한 데이터셋을 앞서 만든 스마트카 상태 정보 데이터 테이블과 같이 테이블을 생성 및 저장한다. 스파크를 활용해서 테이블을 생성할 수 있는데 “스마트카 마스터 데이터”를 활용한다. Fig 17은 스파크 셸을 활용한 쿼리 검색 결과이다.

지금까지의 과정은 수집된 후 적재된 데이터를 탐색하는 방법들이었다. 다음 과정으로는 데이터 탐색 자동화와 마트를 구성한다. 주로 External에 적재된 데이터를 Managed로 통합해서 이동시키는 우지의 워크플로 잡을 만드는 것이다.

```

1 create external table if not exists SmartCar_Status_Info (
2 reg_date string,
3 car_number string,
4 tire_fl string,
5 tire_fr string,
6 tire_bl string,
7 tire_br string,
8 light_fl string,
9 light_fr string,
10 light_bl string,
11 light_br string,
12 engine string,
13 break string,
14 battery string
15 )
16 partitioned by( wrk_date string )
17 row format delimited
18 fields terminated by ','
19 stored as textfile
20 location '/pilot-pjt/collect/car-batch-log/'
21

```

Fig 15 Hive 쿼리 작성

	smartcar_status_info.reg_date	smartcar_status_info.car_number	smartcar_status_info.tire_fl	smartcar_status_info.tire_fr	smartcar_status_info.tire_bl
1	20160101000000	10001	86	83	90
2	20160101000004	10001	96	96	94
3	20160101000008	10001	87	97	89
4	20160101000012	10001	80	70	80
5	20160101000016	10001	87	94	86
6	20160101000020	10001	74	89	93
7	20160101000024	10001	90	96	97
8	20160101000028	10001	75	91	90
9	20160101000032	10001	97	88	82
10	20160101000036	10001	99	93	83
11	20160101000040	10001	92	80	79
12	20160101000044	10001	76	85	96
13	20160101000048	10001	76	73	82
14	20160101000052	10001	83	75	79
15	20160101000056	10001	76	79	92
16	20160101000100	10001	89	81	81
17	20160101000104	10001	94	99	90
18	20160101000108	10001	100	92	85
19	20160101000112	10001	85	79	95
20	20160101000116	10001	84	72	96

Fig 16 Hive 쿼리 검색 결과

```

root@server02:~
scala> smartcar_master_df.show()
-----+-----+-----+-----+-----+-----+-----+-----+-----+
|car_number|sex|age|marriage|region|job|car_capacity|car_year|car_model|
-----+-----+-----+-----+-----+-----+-----+-----+-----+
|A0001|여|32|미혼|서울|프리랜서|1000|2009|F|
|A0002|남|53|미혼|충남|주부|2500|2015|A|
|A0003|여|62|기혼|대전|회사원|2500|2012|B|
|A0004|남|31|미혼|광주|공무원|2000|2010|D|
|A0005|남|67|미혼|대구|공무원|1700|2002|C|
|A0006|여|30|미혼|인천|전문직|2000|2016|D|
|A0007|남|61|미혼|전남|개인사업|1700|2003|E|
|A0008|여|20|미혼|충북|개인사업|1500|2013|G|
|A0009|여|60|미혼|경남|프리랜서|3500|2015|D|
|A0010|여|69|미혼|제주|개인사업|1200|2003|A|
|A0011|남|29|기혼|충남|주부|1000|2008|G|
|A0012|여|53|미혼|세종|학생|2500|2006|E|
|A0013|여|43|미혼|인천|전문직|1500|2016|E|
|A0014|남|63|미혼|대구|공무원|1500|2006|C|
|A0015|남|45|미혼|충남|프리랜서|1700|2012|F|
|A0017|남|48|미혼|충북|프리랜서|1000|2010|B|
|A0018|여|70|기혼|경북|개인사업|2000|2004|H|
|A0019|남|32|기혼|인천|주부|1700|2004|C|
|A0020|남|65|기혼|대구|주부|3500|2009|F|
|A0021|여|22|기혼|대구|전문직|2000|2001|A|
-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

```

Fig 17 스파크를 활용한 쿼리 검색 결과

관련 데이터 마트를 만들기 위해 휴의 Job Designer에서 우지의 워크플로를 이용한다. 우선 로그 시뮬레이터를 이용해 “스마트카 상태 정보 데이터”를 생성한다. 그리고 플럼과 HDFS에 적재되었는지 확인한다. 그리고 각 데이터셋 별로 스마트카 상태 정보 모니터링, 긴급 점검이 필요한 스마트카 정보, 스마트카 운전자 차량용품 구매 이력 정보의 워크플로를 작성한다. 각 워크플로는 Fig 18, Fig 19, Fig 20으로 스크립트가 구성되어 있다. 공통적으로, 테이블을 생성하는 create_table 함수와 데이터를 입력하는 insert_table 함수를 가지고 있다. 스마트카 상태 정보 모니터링 워크플로는 작업일자를 기준으로 파티션 정보를 추가하는 스크립트가 추가로 포함되어 있고, 스마트카 운전자 차량용품 구매 이력 정보 워크플로에는 차량별로 구매한 상품 리스트를 조회해서 로컬 파일시스템의 특정 위치에 파일을 생성하는 하이브 스크립트를 포함하고 있다.

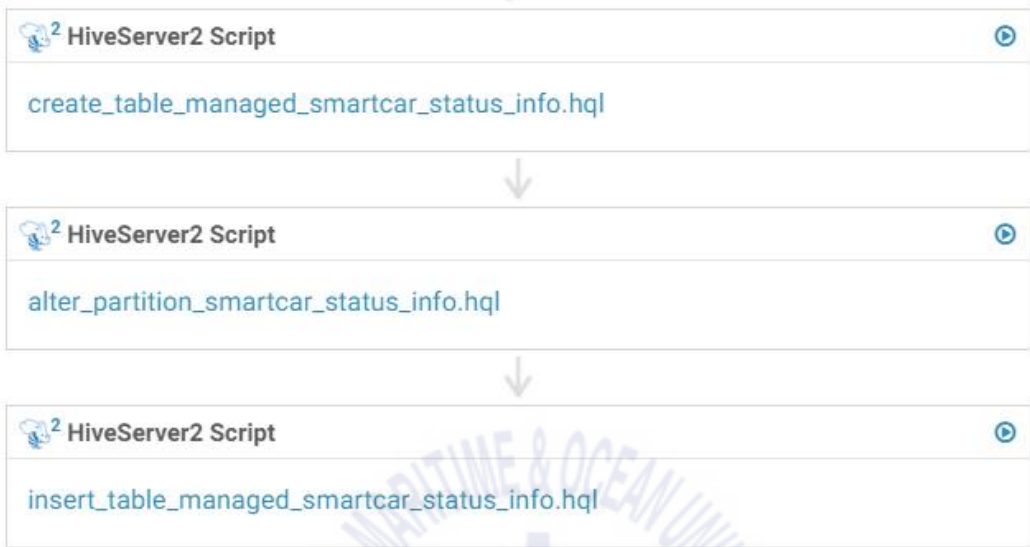


Fig 18 스마트카 상태 정보 모니터링



Fig 19 긴급 점검이 필요한 스마트카 정보

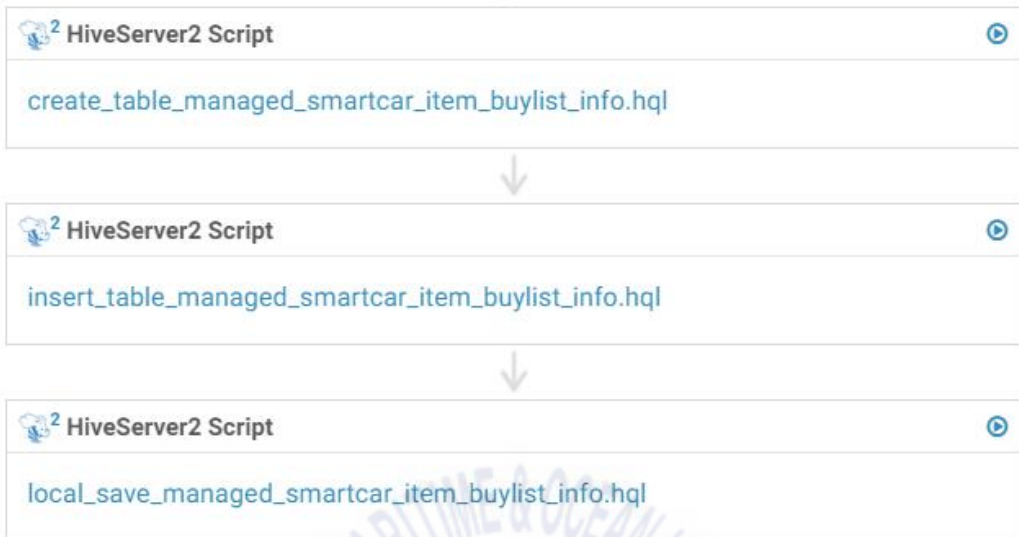


Fig 20 스마트카 운전자 차량용품 구매 이력 정보

4.5 분석 레이어

머하웃을 이용한 데이터 마이닝은 빅데이터에서 사람이 인지하기 어려운 패턴을 발견하고 통찰력을 갖게 만든다. 이를 통해 앞으로 발생할 일들을 예측하면서 신속한 의사결정을 지원하게 된다. 이번 장에서는 지금까지 수집, 적재, 처리, 분석한 스마트카의 데이터셋을 가지고 머하웃의 세 가지 마이닝 기법인 추천, 분류, 군집 기능을 활용하여 보다 가치 있는 데이터 분석을 결과를 만드는 것을 목표로 한다. 머하웃의 라이브러리를 통해 스마트카의 Managed 영역에 적재된 “스마트카 상태 데이터”를 이용해 감독학습-분류, 비감독학습-군집의 머신러닝을 수행한다. 또한 추천 기능을 활용해 “차량용품 구매 이력 데이터”를 분석하고 스마트카 운전자 가운데 유사 그룹 간의 구매 선호도에 따라 차량용품을 추천하는 작업을 진행한다. [9]

1) 추천- 스마트카 차량용품 추천

데이터 마이닝의 추천에 사용될 데이터셋은 “스마트카 차량용품 구매 이력” 정보로서 하이브의 Managed 영역에 있는 구매 이력 테이블에서 약 10만건의 데이터가 적재되어 있다. Fig 21 에 보여지는 것처럼 필요한 항목은 3개의 필드로서 차량 고유 번호, 구매 용품 아이템 코드, 사용자 평가점수이다. 특히 사용자평가점수는 구매한 아이템에 대한 사용자의 긍정 또는 부정을 스코어링하고 있어 사용자의 취향을 알 수 있는 중요한 필드이다. 각 사용자들의 취향에 따라 구매한 이력을 분석하여 새로운 패턴을 발견하고, 유사한 패턴을 보이는 사용자 간의 유사성을 계산하여, 그 결과로부터 유사 사용자 간의 선호하는 아이템을 예측해서 추천하는 것이 사용자 기반 협업 필터링 모델이다. [10]

	smartcar_item_buylist.car_number	smartcar_item_buylist.item	smartcar_item_buylist.score
35	K0095	Item-018	5
5	Y0042	Item-020	2
30	W0023	Item-030	2
7	Y0036	Item-023	3
83	T0026	Item-028	1
27	Q0044	Item-008	1
77	Q0056	Item-014	5
24	N0095	Item-002	5
72	M0051	Item-030	2
22	D0021	Item-005	2
76	B0084	Item-002	3
6	I0085	Item-021	5
47	L0066	Item-029	2

Fig 21 추천 사용 데이터셋

우선 기존 데이터를 머하웃의 추천기에서 사용 가능한 형식으로 재구성한 파일을 만들어야 한다. “스마트카 차량용품 구매 이력” 테이블에서 차량번호, 아이템 코드, 평가점수 데이터를 조회하여 새로운 파일을 Server 2에 생성한다. 머하웃 라이브러리의 입력 데이터로 사용하기 위해 데이터의 type을 Long으로 형변환 하였고, 하이브의 hash() 함수를 사용하였다. 생성된 데이터를 확인하고 HDFS에 경로를 생성하여 저장한다. 머하웃의 추천 분석기를 실행하면 분석기에서 필요한 옵션을 설정 할 수 있는데 입력 경로, 출력 경로, 추천을 위한 유사도 알고리즘 설정, 추천할 아이템 개수를 지정할 수 있다.

명령 결과는 여러 개의 잡과 관련된 맵리듀스가 반복적으로 실행되면서 수분의 시간의 소요되었다. 분석결과를 휴의 파일 브라우저에 저장하여 확인하면, 그 결과 Fig 22에는 3개의 추천 각 차량번호별로 3개의 상품이 추천되었고, 상품 ID와 추천의 강도를 나타내는 수치가 포함된 것을 알 수 있다. 추천 데이터는 추천 정보가 필요한 유관 시스템에 제공되어지는데, 데이터 변환작업이 필요하다. 숫자로 변환되어 나타난 값들은 머하웃의 추천기 데이터로 사용하려고 hash() 함수를 이용해 각각 대체키와 대체코드로 변경된 것이므로, 추가적으로 원래의 차량번호, 상품코드로 복원작업이 필요하다.

2) 분류-스마트카 상태 정보 예측/분류

데이터 마이닝의 분류에 사용될 데이터셋은 “스마트카 상태 정보”로 하이브의 Managed 영역에 약 200만 건의 데이터가 적재되어 있다. Fig 23은 스마트카의 주요 장치에 대한 상태를 기록한 값으로, 차량의 상태를 진단하기 위한 중요한 데이터셋이다. 이 값들을 이용해 차량의 정상/비정상을 분류하는 모델을 만들고 사용자가 임의의 각 장치들의 값을 랜덤으로 입력하여 예측 결과를 확인한다. 랜덤포레스트 알고리즘을 활용하며, 과거의 운행됐던 스마트카의 상태를 판단할 수 있는 Classify 프로그램을 생성한다. 더 나아가 이 프로그램을 통해 실시간으로 운행 중인 스마트카 시스템에 적용해서 차량의 상태를 예측하는 어플리케이션을 만들 수 있다. [11]

추천받은 차량번호	첫번째 - 추천상품 상품ID : 추천 값	두번째 - 추천상품 상품ID : 추천 값	세번째 - 추천상품 상품ID : 추천 값
61506498	[1240943802:4.413618]	[1240943804:4.41339]	[1240943803:4.4094567]
61506499	[1240943833:3.73822,1240943769:3.7255893,1240943803:3.7229412]		
61506500	[1240943802:3.9372115,1240943835:3.934659,1240943833:3.9247985]		
61506501	[1240943833:3.7379262,1240943774:3.7302492,1240943768:3.7280393]		
61506502	[1240943774:3.7492416,1240943800:3.7375124,1240943767:3.7374246]		
61506503	[1240943832:4.6025167,1240943828:4.597181,1240943801:4.5880203]		
61506504	[1240943837:3.942884,1240943767:3.9236326,1240943859:3.9147122]		
61506505	[1240943774:4.1257434,1240943768:4.124374,1240943837:4.117005]		
61506506	[1240943771:4.6092715,1240943837:4.6058173,1240943800:4.6036468]		
61506528	[1240943834:3.60105,1240943770:3.592803,1240943798:3.5921888]		
61506529	[1240943772:4.715842,1240943837:4.7087755,1240943800:4.7086225]		
61506530	[1240943805:4.0446095,1240943771:4.037118,1240943772:4.0236454]		
61506531	[1240943837:3.2082965,1240943831:3.201726,1240943829:3.1996403]		
61506532	[1240943775:3.9573083,1240943799:3.9510791,1240943837:3.9499652]		
61506533	[1240943828:3.703165,1240943773:3.6960275,1240943804:3.694208]		
61506534	[1240943771:4.5152473,1240943837:4.510224,1240943803:4.50887]		
61506535	[1240943804:4.121136,1240943769:4.1156893,1240943859:4.1122084]		
61506536	[1240943773:3.8700159,1240943830:3.866614,1240943768:3.858679]		
61506537	[1240943836:4.3760905,1240943799:4.372197,1240943859:4.3638077]		
61506559	[1240943859:4.479543,1240943769:4.4730077,1240943832:4.471402]		
61506560	[1240943832:3.2655122,1240943771:3.2444124,1240943798:3.2351766]		
61506561	[1240943805:3.866831,1240943803:3.8640647,1240943828:3.8608942]		

Fig 22 추천 결과

우선 하이브를 이용하여 트레이닝 데이터셋을 만드는 작업을 한다. “스마트카 상태 정보” 데이터를 머하웃의 분류기의 입력 데이터로 사용하기 위해 하이브로 재구성한다. 휴의 Hive Editor에서 QL를 실행한다. “스마트카 상태 정보” 데이터로부터 예측 변수(차량 내부 기기들)들의 상태 값으로 임의로 설정하였다. 예측 변수에는 보정치 변수와 가중치 변수가 있고, 이 모든 예측변수의 값을 합산해서 “6” 미만인 경우 “비정상” 을 “6” 이상인 경우에 “정상” 인 값을 가지는 목표 변수를 정의하였다.

	managed_smartcar_status_info_car_capacity	managed_smartcar_status_info_car_year	managed_smartcar_status_info_tire_A	managed_smartcar_status_info_tire_B	managed_smartcar_status_info_tire_C
1	3500	2015	96	77	100
2	2500	2011	93	78	100
3	3500	2015	90	98	100
4	3500	2015	93	95	100
5	2500	2011	80	72	100
6	3500	2015	97	88	100
7	3500	2015	98	74	100
8	3500	2015	100	93	70
9	3500	2015	79	82	70
10	3500	2015	86	96	70
11	2500	2011	95	97	70
12	2500	2011	96	91	71
13	3500	2015	89	88	71
14	3500	2015	88	83	71
15	3500	2015	79	100	72
16	2500	2011	80	95	72
17	2500	2011	100	71	72
18	2500	2011	74	88	72
19	2500	2011	74	88	72

Fig 23 분류 데이터 셋

예측 변수와 목표변수값이 포함된 “스마트카 상태 정보” 입력 데이터셋을 HDFS에 저장한 후 머신러닝에 활용할 디스크립터를 생성한다. 디스크립터 파일은 학습할 데이터의 형식을 정의하는 파일로서 앞서 생성한 입력 데이터셋 파일을 이용한다. 그리고 머신러닝에 활용할 학습 및 테스트 데이터를 설정한다. 입력 데이터셋은 7:3 비율로 분할 생성하였다. 최종적으로 스마트카의 상태 정보 예측 모델을 만들기 위해 트레이닝 데이터셋과 분류 알고리즘 중 하나인 랜덤 포레스트 모델을 사용하여 데이터를 학습시킨다.

입력 데이터로는 앞서 만든 디스크립터 파일과 트레이닝 파일을 사용하였다. Fig 24에 나와 있는 분류기의 모델 평가 결과를 보면 테스트 데이터를 이용한 분류 성공률이 97.199%, 실패율이 2.801%으로 나왔다. Confusion Matrix는 비정상, 정상의 분류 케이스를 매트릭스로 표현한 것이다. (a,a)는 비정상을 비정상으로 성공 분류한 케이스, (b,a)는 정상을 비정상으로 실패 분류한 케이스로 매트릭스를 이해할 수 있다.

지금까지는 머하웃으로 분류기를 이용해 Classify 프로그램을 만들고 평가를 하였다. 다음은 프로그램에 새로운 “스마트카 상태 정보” 데이터셋에 적용해 차량의 상태가 정상인지 비정상인지 분류하는 과정이다. Java를 사용하며, 새로운 데이터셋에 대한 예측프로그램을 만든다. Java 프로그램이 주 내용으로는 스마트카 상태 데이터로 머신러닝을 수행한 후, 만들어진 분류기 모델이 저장된 위치를 설정하고, 분석할 데이터셋의 데이터 형식을 정의한다. 마지막으로 스마트카 상태 정보 입력값을 분류하기 위한 벡터 데이터셋을 설정한다. 그리고 분류기 모델을 작동시켜 스마트카 상태 정보 입력 값에 대한 분류를 예측한다. 이제 Java를 실행시켜 스마트카에 포함된 14종류의 기기들의 상태 값을 순서대로 입력한다. Fig 25은 임의로 지정한 서로 다른 두 값을 넣어 정상, 비정상으로 분류된 결과를 나타낸다.

```

19/11/28 06:08:59 INFO mapreduce.TestForest:
=====
Summary
-----
Correctly Classified Instances      :    629879    97.199%
Incorrectly Classified Instances    :     18151     2.801%
Total Classified Instances          :    648030
=====

Confusion Matrix
-----
a      b      <--Classified as
167103 15892   | 182995   a      = 비 정상
2259   462776 | 465035   b      = 정상
=====

Statistics
-----
Kappa                -0.6073
Accuracy              97.199%
Reliability           63.6099%
Reliability (standard deviation) 0.5524

```

Fig 24 분류기

```

[root@server02 mahout-data]# java -cp bigdata.smartcar.mahout-1.0.jar com.wikibook.bigdat
a.smartcar.mahout.ClassifySmartCarStatus 2000 2000 A 80 80 80 80 1 1 1 1 A B 50
SmartCar Status Prediction :비 정상
[root@server02 mahout-data]# java -cp bigdata.smartcar.mahout-1.0.jar com.wikibook.bigdat
a.smartcar.mahout.ClassifySmartCarStatus 2000 2000 A 80 80 80 80 1 1 1 1 A B 80
SmartCar Status Prediction :정상

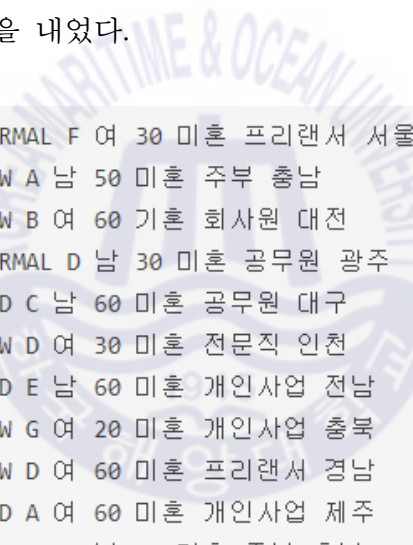
```

Fig 25 분류기를 적용한 스마트카 예측 결과

3) 군집-스마트카 고객 정보 분석

데이터 마이닝 중 3번째인 군집 분석을 진행한다. 사용될 데이터는 “스마트카 마스터 정보” 로 하이브의 External 영역의 테이블로 총 2600명의 스마트카 사용 고객 정보가 적재되어 있다.

데이터셋은 Fig 26과 같이 스마트카의 차량번호, 차량용품, 차량용량, 차량모델 정보와 스마트카 사용자의 개인정보들이 포함되어 있다. 군집 분석은 이러한 속성 정보를 벡터화하고 유사도 및 거리를 계산해 데이터의 새로운 군집을 발견하는 마이닝 기법이다. 군집 분석도 다양한 알고리즘이 있다. 이번 과정에서는 Canopy의 빠른 군집 분석을 적용 후 군집 개수와 중심점을 파악한 후 K-means를 적용하여 값을 내었다.



A0001	소형	NORMAL	F	여	30	미혼	프리랜서	서울
A0002	중형	NEW	A	남	50	미혼	주부	충남
A0003	중형	NEW	B	여	60	기혼	회사원	대전
A0004	중형	NORMAL	D	남	30	미혼	공무원	광주
A0005	소형	OLD	C	남	60	미혼	공무원	대구
A0006	중형	NEW	D	여	30	미혼	전문직	인천
A0007	소형	OLD	E	남	60	미혼	개인사업	전남
A0008	소형	NEW	G	여	20	미혼	개인사업	충북
A0009	대형	NEW	D	여	60	미혼	프리랜서	경남
A0010	소형	OLD	A	여	60	미혼	개인사업	제주
A0011	소형	NORMAL	G	남	20	기혼	주부	충남
A0012	중형	OLD	E	여	50	미혼	학생	세종
A0013	소형	NEW	E	여	40	미혼	전문직	인천
A0014	소형	OLD	C	남	60	미혼	공무원	대구
A0015	소형	NEW	F	남	40	미혼	프리랜서	충남
A0016	중형	OLD	G	여	10	미혼	회사원	강원
A0017	소형	NORMAL	B	남	40	미혼	프리랜서	충북
A0018	중형	OLD	H	여	70	기혼	개인사업	경북

Fig 26 군집 데이터셋

하이브를 이용해 “스마트카 마스터 정보” 데이터 셋을 조회해서 데이터셋을 로컬 디스크에 저장하고 HDFS에도 따로 저장한다. 하이브 QL은 차량 번호, 차량 용량, 차량 연도, 모델, 성별, 나이, 결혼 여부, 직업, 거주지 데이터로 조회한다. 이때 범주가 큰 용량, 연도, 운전자 나이의 경우 범주화함으로써 군집의 정확도를 올릴 수 있다.

군집 분석을 기존 텍스트 형식의 “스마트카 마스터 데이터”를 시퀀스 파일로 변환하는 작업이 필요하다. 시퀀스 파일은 키/값 형식의 바이너리 데이터셋으로 분산 환경에서 성능과 용량 면에서 효율성을 높인 데이터 포맷이다. 시퀀스 파일을 로우별로 n-gram 기반의 TF(Term Frequency) 가중치가 반영된 벡터 데이터로 변환한다. n-gram의 벡터 모델은 단어의 분류와 빈도수를 측정하는 알고리즘이다.

이 과정에서 차량번호별 각 항목의 단어를 분리해 벡터화한다. 스마트카 마스터 데이터를 다차원의 공간 벡터로 변환한다. Canopy로 군집 개수를 파악하기 위해서는 중심점으로부터 거리를 나타내는 t1, t2 옵션을 바꿔가며 반복적인 군집 분석을 수행해야 한다. 스마트카 사용자의 수는 2600명으로 설정하였기 때문에 그에 맞는 군집 개수를 찾기 위해 거리 값을 변경하며 반복적인 수행이 필요하다. k-means로 앞선 구한 군집 개수 및 중심점 데이터를 바탕으로 군집 분석을 한다.

임의로 지정한 VL-1921에 대한 군집 결과를 확인한다. Fig 27에 나온 각 키워드 값을 확인하면 “Old”, “남”이라는 키워드가 1.0의 값을 가지며 이는 군집에 포함된 사용자들이 모두 나이가 많고 남자임을 나타낸다. Fig 28은 군집 VL-1921에 세부 내용을 포함하고 있으며, 여기서 차량번호를 파악하여, 쿼리를 조회할 수 있는데 결과적으로 군집의 다양한 인사이트를 도출할 수 있다. Fig 29은 VL-1921 군집에 대한 쿼리를 조회한 결과이며, 도출할 수 있는 결과는 60대 미혼 남성들이며, 차량은 중형 차량으로 소득 수준이 중간으로 추측할 수 있고, VL-1921 군집과 유사한 고객 군에는 스마트카 E 모델 또는 유사 모델로 타겟 마케팅 할 수 있다.

```

VL-1921{n=
  Top Terms:
old => 1.0
old => 1.0
old => 0.8181818181818182
old => 0.8181818181818182
old => 0.8181818181818182
60 => 0.8181818181818182
60 => 0.8181818181818182
e => 0.7272727272727273
e => 0.7272727272727273
old e => 0.7272727272727273

```

Fig 27 키워드 분석

```

root@server02:~
[root@server02 ~]# grep "Key: 1921" /home/pilot-pjt/mahout-data/clustering/output/seqdump ^
er_result.txt
Key: 1921: Value: wt: 1.0 distance: 5.834710743801654 vec: B0100 = [5:1.000, 10:1.000, 1
6:1.000, 25:1.000, 28:1.000, 29:1.000, 41:1.000, 44:1.000, 59:1.000, 68:1.000, 95:1.000,
146:1.000, 150:1.000, 154:1.000, 201:1.000]
Key: 1921: Value: wt: 1.0 distance: 7.8347107438016526 vec: C0041 = [2:1.000, 10:1.000,
16:1.000, 22:1.000, 25:1.000, 29:1.000, 37:1.000, 42:1.000, 53:1.000, 68:1.000, 95:1.000,
129:1.000, 143:1.000, 152:1.000, 204:1.000]
Key: 1921: Value: wt: 1.0 distance: 4.380165289256201 vec: C0045 = [5:1.000, 10:1.000, 1
6:1.000, 25:1.000, 29:1.000, 42:1.000, 44:1.000, 46:1.000, 59:1.000, 68:1.000, 95:1.000,
146:1.000, 156:1.000, 204:1.000, 238:1.000]
Key: 1921: Value: wt: 1.0 distance: 4.925619834710744 vec: C0063 = [5:1.000, 16:1.000, 2
5:1.000, 29:1.000, 37:1.000, 41:1.000, 42:1.000, 59:1.000, 99:1.000, 146:1.000, 154:1.000
, 197:1.000, 204:1.000]
Key: 1921: Value: wt: 1.0 distance: 5.289256198347108 vec: E0001 = [5:1.000, 16:1.000, 2
5:1.000, 29:1.000, 39:1.000, 41:1.000, 42:1.000, 59:1.000, 99:1.000, 146:1.000, 154:1.000
, 198:1.000, 204:1.000]
Key: 1921: Value: wt: 1.0 distance: 7.834710743801654 vec: E0006 = [5:1.000, 16:1.000, 2
3:1.000, 25:1.000, 28:1.000, 29:1.000, 41:1.000, 59:1.000, 99:1.000, 146:1.000, 150:1.000
, 154:1.000, 189:1.000]
Key: 1921: Value: wt: 1.0 distance: 8.198347107438016 vec: H0039 = [2:1.000, 10:1.000, 1
6:1.000, 22:1.000, 25:1.000, 29:1.000, 36:1.000, 42:1.000, 53:1.000, 68:1.000, 95:1.000,
128:1.000, 143:1.000, 152:1.000, 204:1.000]
Key: 1921: Value: wt: 1.0 distance: 4.198347107438021 vec: P0086 = [5:1.000, 10:1.000, 1
6:1.000, 25:1.000, 29:1.000, 35:1.000, 42:1.000, 46:1.000, 59:1.000, 68:1.000, 95:1.000,
146:1.000, 156:1.000, 204:1.000, 232:1.000]

```

Fig 28 VL-1921 군집 세부 내용

	smartcar_master.sex	smartcar_master.age	smartcar_master.marriage	smartcar_master.car_capacity	smartcar_master.car_year	smartcar_master.car_m
1	남	66	미혼	3500	2003	E
2	남	36	미혼	2000	2002	E
3	남	61	미혼	2500	2002	E
4	남	65	미혼	2000	2002	A
5	남	67	미혼	2500	2001	A
6	남	63	미혼	3000	2000	A
7	남	38	미혼	2500	2006	E
8	남	67	미혼	2000	2006	E
9	남	65	기혼	2000	2007	E
10	남	67	기혼	2500	2000	E
11	여	70	기혼	2000	2004	E

Fig 29 VL-1921에 대한 군집 결과

스마트카 데이터셋에 대한 탐색 및 분석 결과는 다양한 외부 시스템에 공유되고 활용되어야 한다. 머하웃에서 분석된 결과를 외부 RDBMS 시스템에 편리하게 제공하기 위한 도구로 스쿱을 활용하였다. 스쿱은 원래 하둡 생태계에서 импорт 기술로 분류된다. 하지만 본 논문에서는 분석 결과를 외부에 제공하는 용도로만 사용하였다. 스쿱을 이용해 외부 RDBMS에 데이터를 제공하는 작업도 시간별로 수행해야 하는 정기적인 비치 프로세스 작업이다. 이때는 휴의 워크플로의 작업 노드를 스쿱 작업으로 생성할 수 있는데, 여기에 앞서 사용했던 스쿱 내보내기 명령을 입력해 워크플로를 생성하면 정기적인 내보내기 워크플로로 작업을 수행할 수 있다.

제 5 장 결론 및 향후 연구

본 논문에서는 빅데이터 시스템 구축 과정과 스마트카 관련 데이터를 활용한 분석 과정을 다루었다. 빅데이터 시스템 구축 과정에서는 설치한 에코시스템 소프트웨어들의 기능들과 프로그램들 간에 연관성을 다루며 전체적인 시스템을 이해하는 단계를 거쳤고, 여기에 실제 스마트카 데이터셋들을 적용하여, 빅데이터 처리 과정에서 각 레이어 단계별로, 프로그램별로 어떤 처리 과정을 거쳤는지 설명하였다. 본 논문에서는 대용량 배치 데이터를 주로 다루었으며, 향후 연구로서 다른 데이터 타입인 실시간 데이터에 접근하는 것이 빅데이터를 활용한 보다 질 높은 분석 결과를 만드는 것을 목표로 하고 있다. 대용량 데이터에 비해 실시간 데이터는 더 많은 프로그램을 필요로 하며 보다 복잡한 처리 과정을 가지고 있다. 하지만 서로 다른 두 타입의 데이터를 융합할 수 있으면 다양한 분석 도구들을 사용할 수 있으며, 보다 가치 있는 결과 값을 얻을 수 있을 것이다.

참고문헌

- [1]McKinsey Global Institute, June 2011, “Big Data: The Next Frontier for Innovation, Competition, and Productivity,” by James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Hung Byers.
- [2]김재생. (2014). 빅데이터 분석 기술과 활용사례. 한국콘텐츠학회지, 12(1), 14-20.
- [3]어윤봉 (2012. 8. 16), “빅데이터 동향 및 시사점”, 《IT SPOT ISSUE》, SPOT 2012-03, 정보통신산업진흥원.
- [4]이영훈, 김용일. (2016). Mi Band와 MongoDB를 사용한 생체정보 빅데이터 시스템의 설계. 스마트미디어저널, (), 2287-1322
- [5]전성환, 정혜진, 나연목. (2017). 하둡 맵리듀스 튜닝을 통한 기계학습 성능 개선. 한국정보과학회 학술발표논문집, (), 731-733.
- [6]김직수. (2019). 하둡 기반 대규모 작업처리 프레임워크에서의 Adaptive Parallel Computability 기술 연구. 방송공학회논문지, 24(6), 1122-1133.
- [7]이태현, 유은섭, 박개명, 유성상, 박진표, 문두환. (2019). 하둡 및 스파크 기반 빅데이터 플랫폼을 이용한 선박 운항 효율 이상 상태 분석. 한국기계가공학회지, 18(6), 82-90.
- [8]이장열, 남춘성, 신동렬. (2017). Apache Spark를 이용한 머신러닝 빅데이터 교육 플랫폼. 한국정보과학회 학술발표논문집, (), 1531-1533.
- [9]박정민, 노재춘. (2017). 데이터 처리를 위한 빅데이터와 머신러닝 플랫폼 조사 및 성능 비교. 대한전자공학회 학술대회, (), 1164-1166.
- [10]최도현, 박중오. (2015). 빅데이터 환경에서 사용자 거래 성향분석을 위한 머신러닝 응용 기법. 한국정보통신학회논문지, 19(10), 2232-2240.
- [11]정훈, 박문성. (2018). 차륜 및 차축베어링 고장진단을 위한 빅데이터 기반 머신러닝 기법 연구. 한국산학기술학회 논문지, 19(1), 75-84.

감사의 글

먼저 논문을 끝낼 수 있게 도와주신 많은 분들에게 감사의 말을 전합니다.

대학원 생활을 하게 된 이유가 학부 때 부족하였던 전공 공부를 하고 싶어서였는데, 돌이켜보니 아직 부족하고 아쉬움이 많이 남습니다. 방황하던 시간도 있었지만, 저의 연구 방향과 논문지도를 성심성의껏 이끌어 주신 박휴찬 교수님께 감사드립니다. 질타가 아닌 칭찬으로 항상 저를 격려해주셔서 제가 포기하지 않고 여기까지 올 수 있었습니다. 그리고 항상 반갑게 웃으시며, 인사해주신 신옥근 교수님, 항상 제 안부를 물어봐 주신 이장세 교수님, 임베디드 사업과 관련하여 많은 조언과 관심을 가져주신 김재훈 교수님 감사드립니다. 김경언 조교님과 강군호 조교님 항상 챙겨주셔서 감사드립니다.

말레이시아에서 와서 새로운 경험을 시켜주었던 하직, 야나, 자라 덕분에 외롭지 않게 대학원 생활을 할 수 있어서 고마워. 그리고 먼저 간 진이 아직 남은 정민이도 항상 잘되길 바랄게. 대학원 생활을 하면서 서로 의지가 되었던 동규와 정욱이도 남은 학기 열심히 해. 그리고 함께 고생한 다른 대학원 분들 너무나 감사드립니다. 마지막으로 멀리서 항상 지켜봐 주시고 힘이 되어 주신 부모님께 감사드립니다.