

졸업프로젝트 보고서

Real-time Personal Course Recommendation System

2020 년 12 월 1 일

2015410031 김재훈
2016320199 권지혜
(지도교수 : 유헌창)

고려대학교 정보대학 컴퓨터학과

Contents

- ❑ Introduction(프로젝트 개요)
- ❑ Related Work (Project)
- ❑ Main Idea
- ❑ Experiment / 성능평가
- ❑ Conclusion

1. Introduction(프로젝트 개요)

□ 개요

- 급속도로 발전하고 있는 정보화 사회 속에서 데이터의 양은 기하급수적으로 늘어나고 있다. 이러한 정보의 홍수 속에서 개인에게 유의미한 정보를 추출하는 과정은 무엇보다 중요하게 되었으며, 이러한 흐름에 따라 유저의 활동에 기반하여 맞춤형 정보를 제공할 수 있는 추천시스템을 구현하고자 한다.
- 고려대학교에는 매 학기 수많은 수업이 개설된다. 이러한 수업들 중 학생들은 자신과 가장 알맞은 수업을 듣기 위해 수강평을 검색하곤 한다. 그러나, 모든 수강평을 열람하기엔 한계가 존재하며, 자신에게 정말 필요한 수업을 발견하지 못하는 경우가 빈번하다.
- 따라서 학생들이 보다 편리하고 효과적으로 수업을 선택할 수 있도록 수강평을 종합 및 분석하여 개인 맞춤형 교내 수업 추천시스템 웹서비스를 구현하고자 한다.
- 수강평은 고려대학교 최대 수강평가 사이트인 KLUE를 사용하고자 한다.

1. Introduction(프로젝트 개요)

□ 목적

- 서비스 사용자들의 서비스 사용 로그 데이터, KLUE 리뷰 작성 이력 등의 정보를 고려해 **실시간**으로 적절한 강의를 추천해주는 **개인 맞춤형 서비스**를 구현한다.

□ 기대효과

- 시간 단축 : 검색 등의 중간과정 없이 direct로 콘텐츠를 추천할 수 있다.
- 커버리지 확보 : 이용자가 예측하지 못한 강의를 추천함으로써 콘텐츠의 폭을 확장한다.




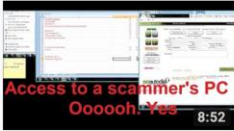








1. Introduction(프로젝트 개요)

□ Importance

- 현재 많은 e-commerce 기업의 profit 발생 요인 중 추천시스템은 큰 부분을 차지하고 있고, 데이터의 효율적 축적, 추천 알고리즘의 전문화가 경쟁요소로 자리잡은 만큼 추천시스템의 중요성은 더욱 대두되고 있다.
- KLUE사이트 역시 약 10년 동안 운영이 되어온 만큼 많은 데이터가 축적되어 있으며 학생들이 수강한 교과목과 강의평 사이에 유의미한 상관관계가 있을 것이라 생각하여, 이를 극대화하기 위한 추천시스템을 구현하고자 한다.
- 한정된 시간과 수강 인원 안에 강의를 선택해야 하는 학생의 입장에서 많은 선택지는 결정에 어려움을 주기 때문에 실시간 개인 맞춤 강의 추천시스템은 꼭 필요한 서비스가 될 것이라 생각한다.

2. Related Work (Project)

Recommended

 <p>"His Ideas Are Idiotic" Jordan Peterson DESTROYS Justin Conservative Network 403K views • 1 month ago</p>	 <p>JORDAN PETERSON HYPER-INTELLECTUALS PhilosophyInsights 865K views • 8 months ago</p>	 <p>Jordan Peterson Dissects the Mind of a Mass Murderer Cheap Virtue 671K views • 1 year ago</p>	 <p>Access to a scammer's PC Jim Browning 2.1M views • 1 year ago</p>	 <p>PETERSON GENIUS ScienceNET 458K views • 8 months ago</p>	 <p>PETERSON VS. MUHAMMAD Acts17Apologetics 504K views • 1 month ago</p>
 <p>"All White men are R@CIST" Smart Man OWNS Race- 50 Stars 159K views • 5 months ago</p>	 <p>Jordan Peterson: Milo is a walking Contradiction and He Conservatism 277K views • 1 month ago</p>	 <p>Jordan Peterson Destroys Gender Denying Ideologue AustralianRealist 733K views • 2 years ago</p>	 <p>Jordan Peterson: My Encounter With Hells Angels Clash of Ideas 282K views • 6 months ago</p>	 <p>Leftist Host SNAPS At Jordan Peterson, Instantly Conservative Network 993K views • 1 week ago</p>	 <p>How to Easily Overcome Social Anxiety - Prof. Jordan Psyche Matters 908K views • 7 months ago</p>

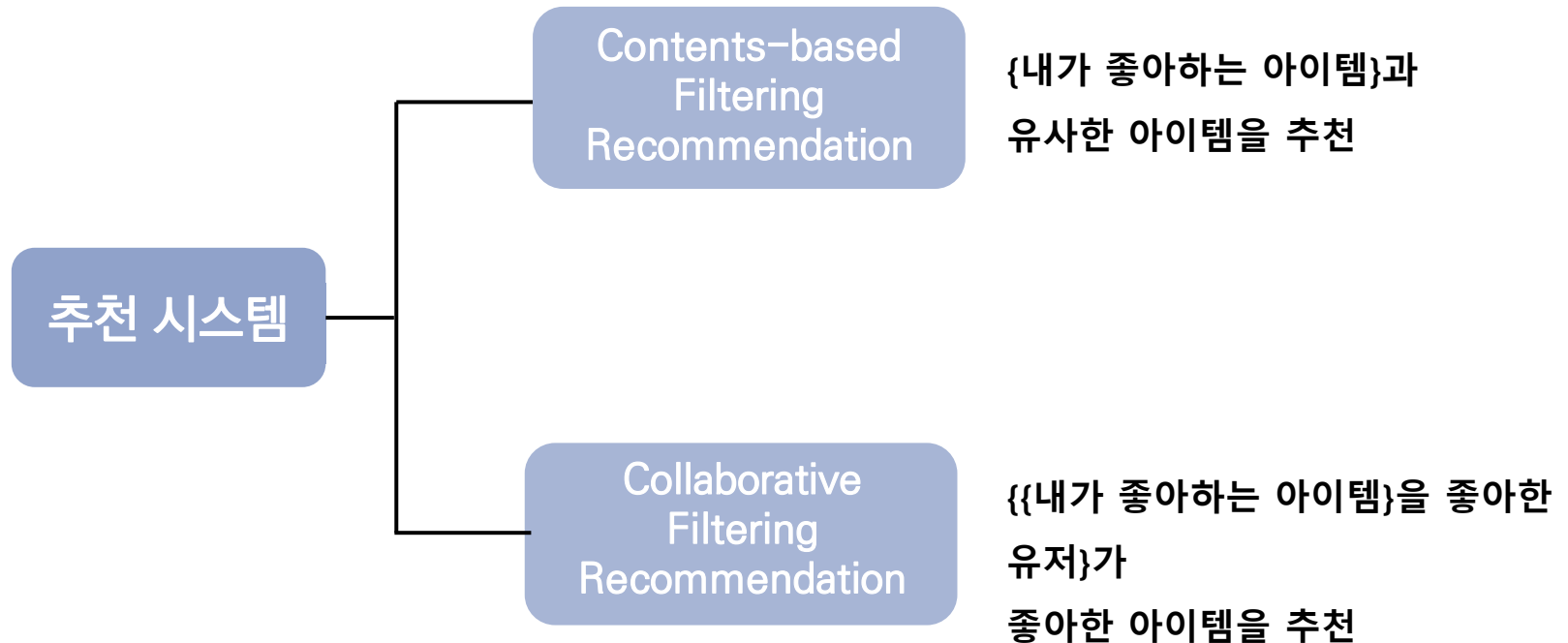
SHOW MORE

□ 최근 전 세대에 걸쳐 가장 많이 소비되고 있는 유튜브는 추천 알고리즘으로 유명하다. 이는 실시간으로 로그의 시청 기록을 활용해 시청한 영상의 description, keywords 등을 분석하고, 유사한 영상을 추천한다.

□ 이러한 점에서 착안하여, KLUE 수강평가 사이트의 강의 텍스트를 분석하여 실시간 유저 로그 수집을 통해 유저맞춤 추천 알고리즘을 구현하고자 한다.

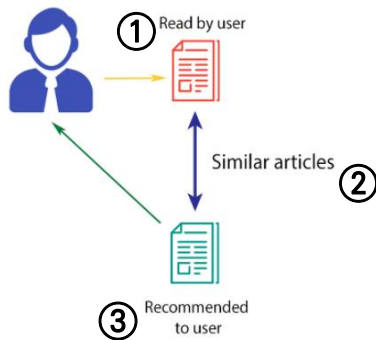
3. Main Idea

□ 추천 알고리즘 유형



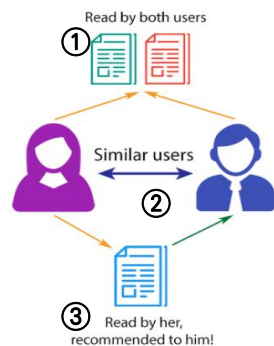
3. Main Idea

□ 추천 알고리즘 유형



□ 1. Contents-based Filtering Recommendation (내용 기반 추천)

- 콘텐츠 내용의 유사도를 바탕으로 추천을 하는 알고리즘
- 이용자가 조회하거나 구매한 콘텐츠 자체의 특징을 분석하여 그와 유사한 콘텐츠 추천
- ex) 클래식 음악을 듣는 이용자에게 최근 발매된 클래식 음악이나, 평소 듣는 음악가의 다른 음반 추천



□ 2. Collaborative Filtering Recommendation (협업 필터링 추천)

- 이용자와 콘텐츠 사이의 유사성을 기준으로 비슷한 성향의 이용자들이 선호하는 콘텐츠 추천
- ex) 모차르트 음악을 듣는 이용자가 베토벤의 음악도 들었다면, 모차르트 음악을 듣는 다른 이용자에게도 베토벤 추천

3. Main Idea

□ Collaborative Filtering Recommendation (협업 필터링 추천)

- 2009년 넷플릭스 추천 시스템 경진대회에서 해당 알고리즘이 우승하면서 주목 받음
- **행렬 분해**(Matrix Factorization)를 활용해 user-item 간 **잠재 요인**을 파악하는 알고리즘

예시) 이용자-강의 조회수 행렬에 잠재되어 있는 어떠한 요인 검출 → 이를 기준으로 다른 강의 추천

	강의1	강의2	강의3	강의4	강의5
이용자1	4			2	
이용자2		5		3	1
이용자3			3	4	4
이용자4	5	2	1	2	

=

	요인1	요인2	요인3
이용자1	0.96	0.47	-0.76
이용자2	-0.03	0.84	-2.47
이용자3	2.38	0.11	-1.20
이용자4	0.59	1.10	-1.06

×

	강의1	강의2	강의3	강의4	강의5
요인1	1.62	-0.79	1.04	1.07	1.43
요인2	1.51	0.45	-0.06	0.12	-0.21
요인3	-2.22	-1.85	0.43	1.18	-0.50

- 각 이용자별 강의에 대한 조회수
- 빈 칸 = 아직 조회되지 않은 강의

- (이용자, 강의) = (이용자, 잠재요인) × (잠재요인, 강의)
- **잠재요인** : blackbox element로 구체적인 정의는 어려우나, 추천의 근거가 됨
- 분해된 두 행렬의 곱은 원 행렬의 조회수 예측치 제공

3. Main Idea

□ 데이터 수집

강의번호, 강의명, 교수명, 강의평, 평점 등의 강의정보를 크롤링

KLUE

BeautifulSoup
Crawling

데이터 (7) lectures											
	id	lectid	lectname	professor	userid	username	review	starttotal	stardifficulty	stardstudytime	starattendance
53		53	COSE156 모두들위한파이...	유길상	49498	고대나오	파이썬을 쉽게 배...	4	1	1	1
54		54	COSE156 모두들위한파이...	유길상	52149	rust2c	교양으로 파이썬 ...	5	3	3	1
55		55	COSE156 모두들위한파이...	유길상	46014	메추리알막	중간고사까지 는 ...	5	3	2	1
56		56	COSE156 모두들위한파이...	유길상	46955	꿀강활력터	프로그래밍을 차...	5	2	2	3
57		57	COSE156 모두들위한파이...	유길상	46955	꿀강활력터	프로그래밍을 차...	5	2	2	3
58		58	COSE156 모두들위한파이...	유길상	52212	빈님	한번도 파이썬을 ...	1	4	3	3
59		59	COSE156 모두들위한파이...	유길상	51748	자미	학생이 직접 출석...	5	2	2	5
60		60	COSE156 모두들위한파이...	유길상	30632	코른쟈지	잘 가르치십니다...	1	1	1	1
61		61	COSE156 모두들위한파이...	유길상	51601	미미미딩1030	파이썬 중간 고사...	4	5	4	1
62		62	COSE156 모두들위한파이...	유길상	43924	복숭아마시평	교수님은 아주 착...	4	3	3	1
63		63	COSE156 모두들위한파이...	유길상	49218	jhw	파이썬을 배우고 ...	5	5	4	3
64		64	COSE156 모두들위한파이...	유길상	53031	정예홍	출석체크는 교수...	5	4	4	3
65		65	COSE156 모두들위한파이...	유길상	37720	DDeo	교수님이 수업을 ...	5	1	2	2
66		66	COSE159 소프트웨어관리	이원규	51179	k2mss5	교수님이 한분이 ...	3	3	3	5
67		67	COSE159 소프트웨어관리	이원규	45918	스타박스	저는 수업을 거의...	3	3	3	5

□ 데이터 전처리

데이터 내 결측치 제거
특수문자 제거
불용어(Stopwords) 제거

1단계 Data Cleansing

한글 형태소 분석기인
KoNLPy의 Komoran을
활용해
Tokenizing / Normalizing /
Stemming 진행

2단계 Tokenizing

Scikit-Learn. TFidfVectorizer
를 활용해
데이터토큰을 TFIDF 벡터로
변환

3단계 Embedding

3. Main Idea

□ 추천 알고리즘 구축

Contents-based Filtering Recommendation

```
with open('cosine_sim_final.pkl', 'rb') as f:
    cosine_sim=pickle.load(f)

x = pd.Series(cosine_sim[1]).index+1

def get_recommendations(i): #insert ID
    sim_scores = list(enumerate(cosine_sim[i-1]))
    sim_scores = sorted(sim_scores, key = lambda x : x[1], reverse=True)
    sim_scores = sim_scores[1:21]
    recommend = x[[s[0] for s in sim_scores]]
    return recommend
```

강의정보를 활용해 lecture embedded vector
간의 cosine similarity 추출한 뒤, 정렬

Collaborative Filtering Recommendation

```
def svd_recommend(user_id, num_recom=5):
    index = user_id-1

    #sort expected ratings
    sorted_index = svd_pred.loc[user_id].sort_values(ascending=False).index

    #List of Lectures viewed
    seen_lects = user_item.loc[user_id][user_item.loc[user_id]!=0].index
    seen_lects = list(seen_lects.unique())

    #get Lectures not viewed
    sorted_notseen = [x for x in sorted_index if x not in seen_lects]
    recom_title = list(user_item.columns[sorted_notseen][:num_recom])

    return recom_title
```

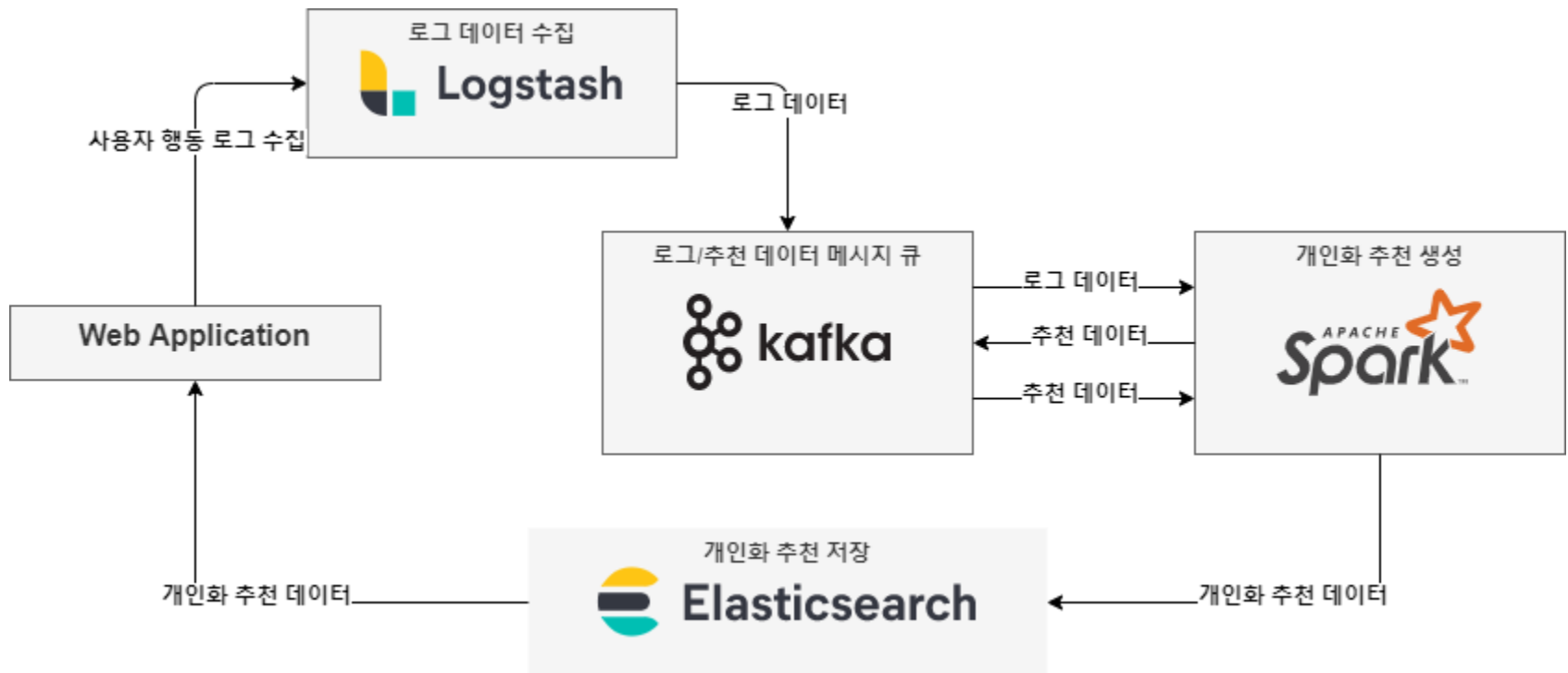
수집된 유저 조회 테이블을 pivot table로 변환한
뒤, SVD 진행하여 높은 예측 조회도 순으로 정렬



Hybrid Recommendation Algorithm

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – 시스템 구성도



3. Main Idea

□ 실시간 추천 서비스 아키텍처 - Components

Web Application

실제 사용자들이 서비스를 사용하며, 데이터를 남기고 추천 서비스를 제공받는
User Interface

Data Pipeline

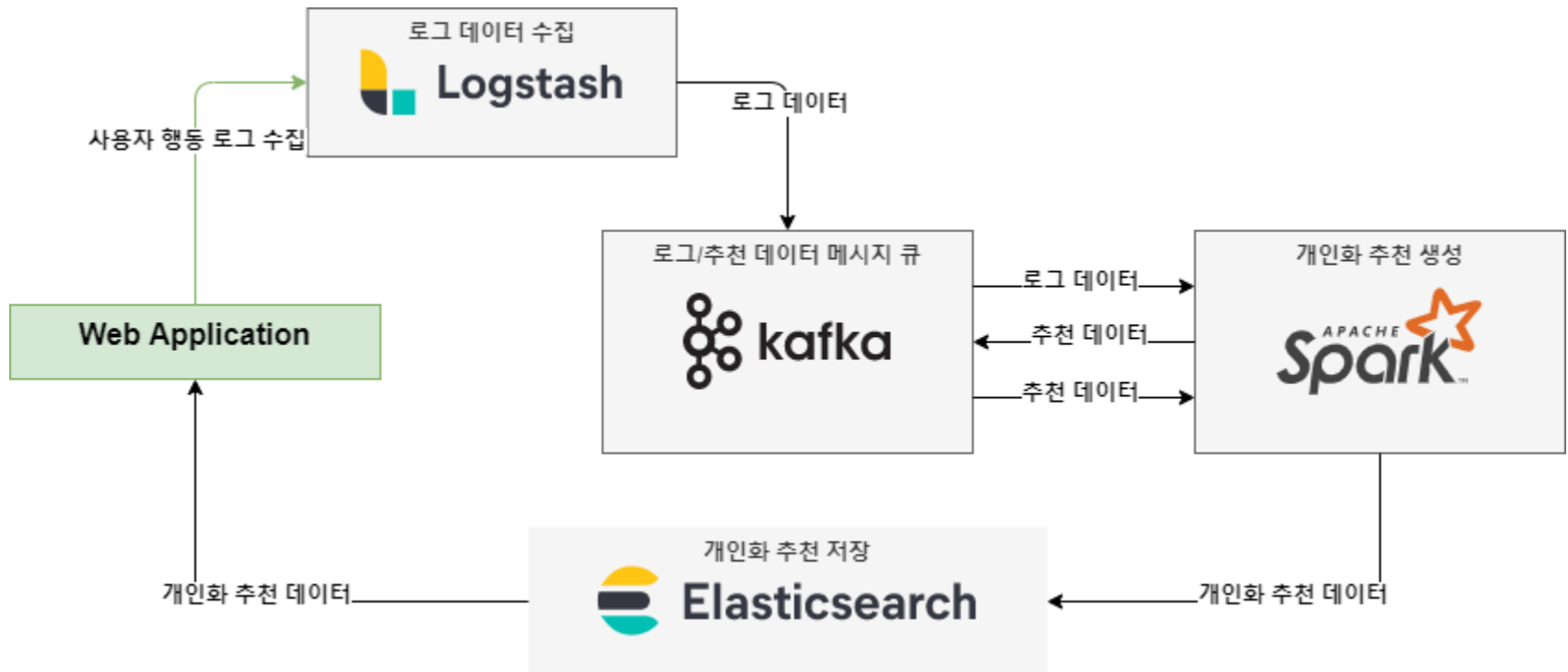
Web Application에서 발생하는 사용자 로그 데이터들을 실시간으로 수집하고,
생성된 추천 데이터를 Web Application으로 전달하는 Component들의
집합(Logstash, Kafka, Spark, Elasticsearch 등)

Recommender Engine

Data Pipeline을 거치는 로그 데이터와 추천 알고리즘을 바탕으로 개인화 추천을
생성하는 Component (Spark)

3. Main Idea

□ 실시간 추천 서비스 아키텍처 - Web Application



3. Main Idea

□ 실시간 추천 서비스 아키텍처 - Web Application

고려대학교 교양 강의 추천 □

- [ab123](#) (로그아웃) [마이 페이지](#)

학수번호	강의명	담당교수	
CHIN105	중국문학산책	김지선	담아두기
CHIN120	중국어번역연습	김영민	담아두기
COSE156	모두를위한파이썬프로그래밍	유길상	담아두기
COSE159	소프트웨어윤리	이원규	담아두기
ECON110	경제학개론	최성환	담아두기
EGRN203	과학기술과지식재산	함병현	담아두기
FRAN133	프랑스문화탐색	조주은	담아두기
FRAN134	프랑스문학탐색	유재화	담아두기
FRAN135	프랑스역사와사회탐색	유재화	담아두기
FRAN136	프랑스공연예술탐색	송태효	담아두기
FRAN137	프랑스영화탐색	송태효	담아두기
GEOG101	도시와국토	손승호	담아두기
GEOG104	관광의역사와문화	천종호	담아두기

Django Framework를 기반으로 개발한 Web Application.

아래 3가지 기능을 수행한다.

1. 정보를 조회하고 싶은 강의를 클릭하면 해당 강의의 Klue 페이지로 redirect 한다.
2. 관심있는 강의를 담아둘 수 있다.
3. 마이페이지에서 담아 둔 강의를 조회할 수 있으며, 개인화 추천을 제공받을 수 있다.

3. Main Idea

□ 실시간 추천 서비스 아키텍처 - Web Application

```

'logstash': {
  'level': 'INFO',
  'class': 'logstash.TCPLogstashHandler',
  'host': LOGSTASH_SERVER,
  'port': 5959,
  'version': 1,
},

'loggers': {
  'lecture_app.views': {
    'handlers': ['logstash', 'file'],
    'level': 'INFO',
  },
  'member_app.views': {
    'handlers': ['logstash', 'file'],
    'level': 'INFO',
  },
},

```

사용자의 강의 정보 조회 행동에서 발생하는 Click Event,

사용자가 관심있는 강의를 담아두는 행동에서 발생하는 Wish Event

위 두가지 Event에서 발생하는 로그 데이터들을 Logstash가 수집.

```

"level" => "INFO",
"@version" => "1",
"type" => "logstash",
"user_id" => 1,
"message" => "1 CLICK 2",
"logger_name" => "lecture_app.views",

```

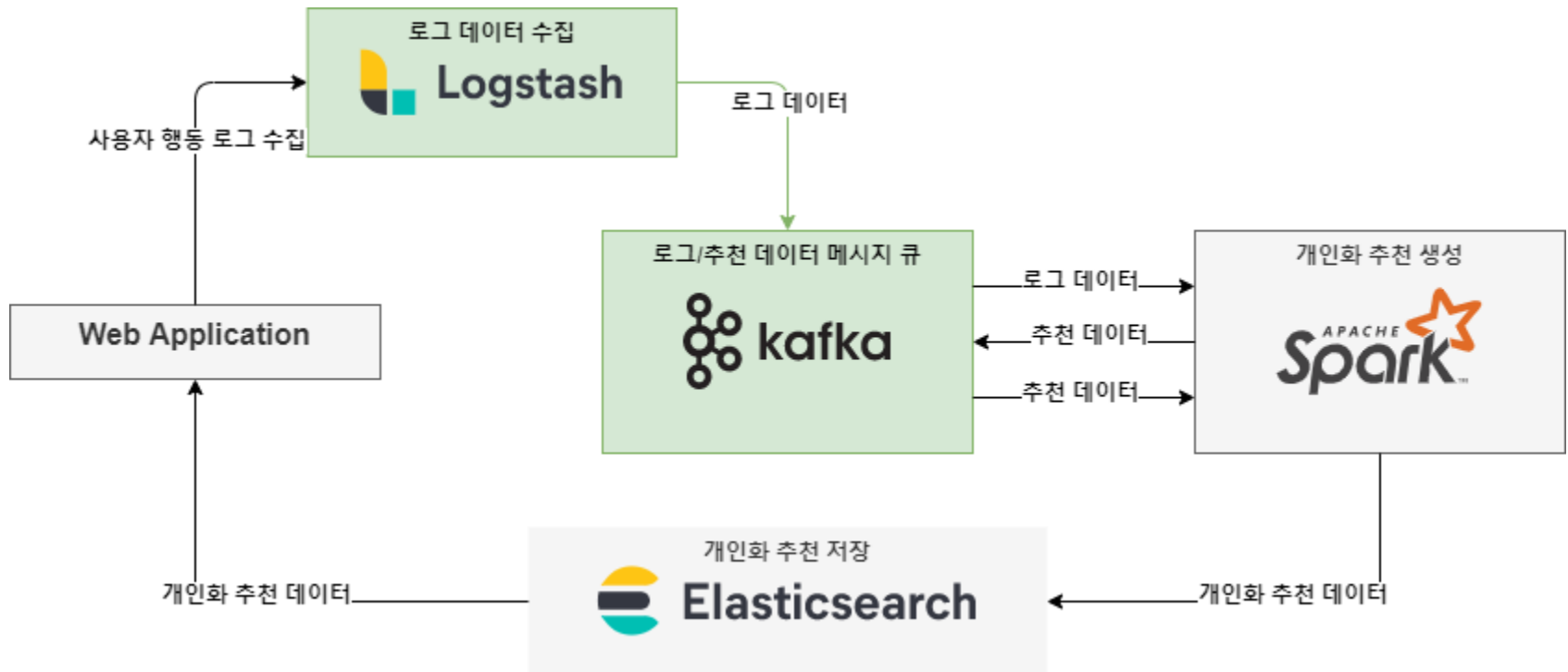
```

"level" => "INFO",
"@version" => "1",
"type" => "logstash",
"user_id" => 1,
"message" => "1 WISH 8",
"logger_name" => "member_app.views",

```


3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Data Pipeline (Collecting, Messaging)



3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Data Pipeline (Collecting, Messaging)



다양한 소스에서 데이터들을 실시간으로 수집할 수 있는 파이프라인 오픈소스
본 프로젝트에서는 Google Cloud Platform VM 환경에서 실행하며, Django Web
Application에서 발생하는 사용자 로그 데이터를 수집한다.

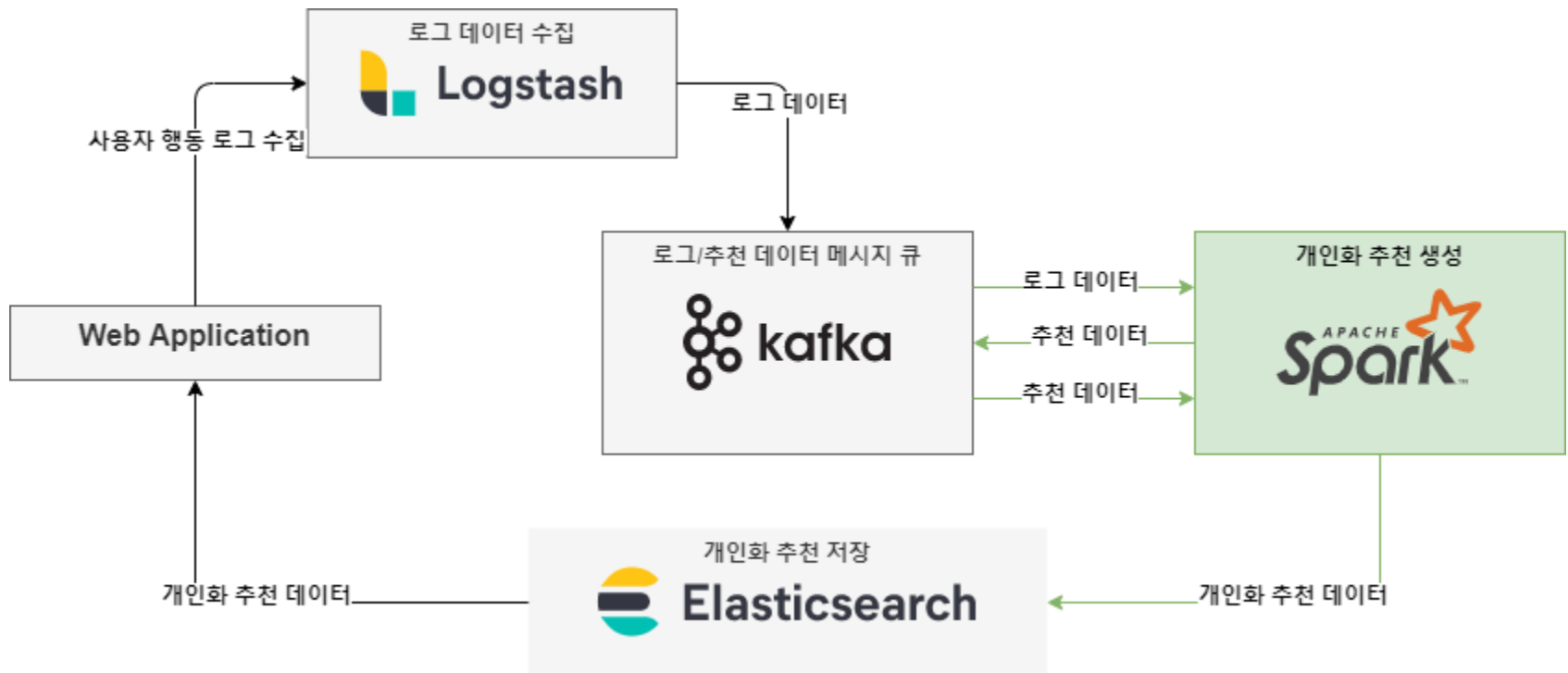


분산 환경 기반 메시지 큐 오픈소스. 본 프로젝트에서는 Google Cloud Platform
VM 환경에서 1 Kafka Broker / 1 Zookeeper를 standalone으로 실행하며, 두 가지
데이터 Topic을 관리한다.

1. Logstash가 수집하여 produce하고 Spark가 consume하는 사용자 로그
데이터 Topic
2. Spark가 추천 알고리즘을 통해 produce하고, 다시 consume하여
Elasticsearch에 적재하는 개인화 추천 결과 데이터 Topic

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Recommender Engine



3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Recommender Engine



대용량 데이터 처리에 사용되는 분산 처리 엔진 오픈소스

본 프로젝트에서는 Kafka와 연동하여, 실시간 로그 데이터 처리와 개인화 추천 생성을 수행. (Spark Streaming, Spark SQL 활용) On-premise 환경, 1 Master / 3 Worker를 standalone으로 실행

1. Kafka의 웹 어플리케이션 로그 데이터 Topic을 consume한다.
2. Kafka에 개인화 추천 결과 데이터 Topic을 produce한다.
3. Produce한 추천 결과 데이터 Topic을 다시 consume하여 Elasticsearch에 개인화 추천 결과 데이터를 적재한다.

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Recommender Engine

Spark Streaming에서의 read stream과 write stream

```
# 사용자 행동 로그 데이터 메시지 consume
df = spark \
    .readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_BROKER) \
    .option("subscribe", "lecture-recsys-log") \
    .option("failOnDataLoss", "false") \
    .load()
ds = df.selectExpr("CAST(value AS STRING)")

# 추천 결과 데이터 메시지 consume
df2 = spark \
    .readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_BROKER) \
    .option("subscribe", "lecture-recommenders") \
    .option("failOnDataLoss", "false") \
    .load()
ds2 = df2.selectExpr("CAST(value AS STRING)")
```

```
# 개인화 추천 결과 produce
query1 = final_recsys_df_sql \
    .writeStream \
    .trigger(processingTime='30 seconds') \
    .outputMode("update") \
    .format("kafka") \
    .option("kafka.bootstrap.servers", KAFKA_BROKER) \
    .option("checkpointLocation", "pyspark/streaming/checkpointLocation4") \
    .option("topic", "lecture-recommenders") \
    .option("failOnDataLoss", "false") \
    .start()

# 개인화 추천 결과 Elasticsearch에 적재
query2 = logdf2 \
    .writeStream \
    .outputMode("append") \
    .format("org.elasticsearch.spark.sql") \
    .option("checkpointLocation", "pyspark/streaming/checkpointLocation5") \
    .option("es.resource", "recommenders_db") \
    .option("es.nodes", ELASTICSEARCH_DB) \
    .option("failOnDataLoss", "false") \
    .start()
```

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Recommender Engine

개인화 추천 생성 로직

1. Spark Structured Streaming을 통해 실시간 로그 데이터(click, wish)를 batch 형태의 table에 추가하여 마치 정적 데이터를 다루듯이 dataframe 연산을 수행 => 사용자 별로 로그 데이터들을 group by 집계.
2. 사용자들이 click or wish한 강의와 유사한 강의를 Contents-based Filtering을 통해 구하고, 새로운 batch table을 생성 (Spark UDF 활용)
3. 사용자가 click 했다면, 1점 / wish 했다면 3점의 점수를 각 유사 강의들에게 부여
4. 30초 트리거 동안 쌓인 batch table 내의 강의들의 점수를 바탕으로 상위 5개 강의를 선정 (Spark UDF 활용)
5. {user id, 해당 사용자를 위한 추천 강의 list}의 형태로 추천 결과를 Kafka Topic에 produce
6. Kafka의 추천 결과 Topic을 다시 consume하여, Elasticsearch에 적재

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Recommender Engine

- Spark UDF(User Defined Function) 사용을 위한 함수 정의

```
# 유사 강의 목록 구하기 + 점수 부여
def get_recommendations(i, action):
    sim_scores = list(enumerate(cosine_sim[i-1]))
    sim_scores = sorted(sim_scores, key = lambda x : x[1], reverse=True)
    sim_scores = sim_scores[1:6]
    recommend = list(x[s[0]] for s in sim_scores)

    if action == 'CLICK':
        for i in range(len(recommend)):
            recommend[i] = [recommend[i], 1]
    elif action == 'WISH':
        for i in range(len(recommend)):
            recommend[i] = [recommend[i], 3]

    return recommend
```

```
def recsys_count(recommenders):
    temp = dict()
    for rec in recommenders:
        if rec[0] in temp:
            temp[rec[0]] = temp[rec[0]] + rec[1]
        else:
            temp[rec[0]] = rec[1]

    rec_list = list()
    for _ in range(5):
        max_key = max(temp.keys(), key=lambda k: temp[k])
        rec_list.append(max_key)
        temp.pop(max_key)

    return rec_list
```

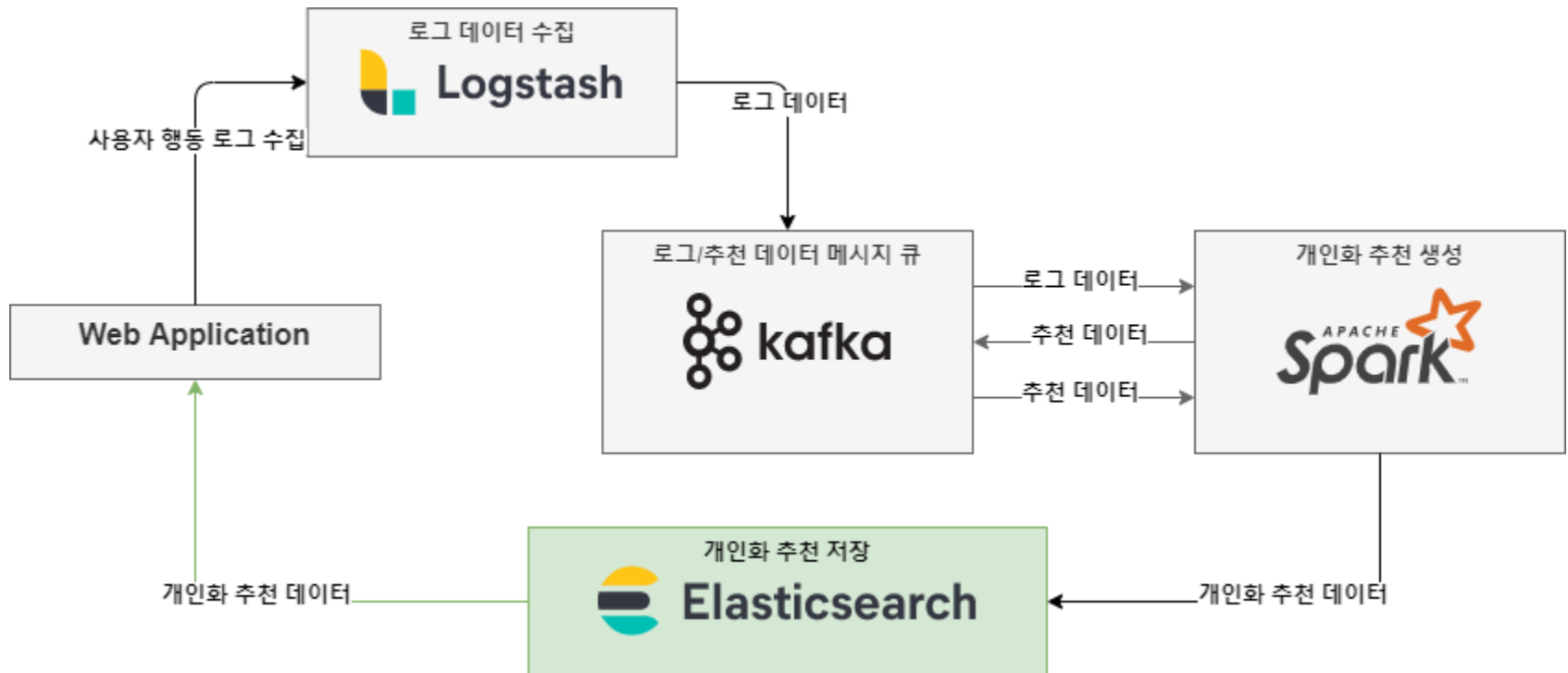
수집한 로그 데이터
batch table의 강의들을
대상으로

Contents-based Filtering
으로 유사한 강의들을
구하고, 사용자
행동(click, wish) 유형에
따라 점수를 부여한다.

사용자별로 집계한 추천
점수에 기반하여 사용자
맞춤 상위 5개의 강의들을
선정한다.

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Data Pipeline (Storage, Serving)



3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Data Pipeline (Storage, Serving)



분산 검색 엔진 오픈소스. 표준 JSON document 형식으로 데이터를 저장할 수 있으며, REST API 방식으로 데이터에 접근이 가능.

위 특징을 바탕으로 해서 본 프로젝트에서는 Elasticsearch를 두 가지 방식으로 활용하였다.

Storage: 추천 결과 데이터들을 저장하는 NoSQL 데이터베이스로서 Elasticsearch를 활용.

```
user_id: 1 best: 451, 129, 391, 398, 465
```

Serving: Web Application에서 Elasticsearch에 대한 HTTP 프로토콜 기반 요청을 통해 추천 결과 데이터를 제공 받음.

3. Main Idea

□ 실시간 추천 서비스 아키텍처 – Data Pipeline (Storage, Serving)

- 결과적으로 사용자는 자신의 과거 서비스 사용 history를 바탕으로 개인화 추천을 제공받는다.

고려대학교 교양 강의 추천 □

- [ab123 \(로그아웃\) 마이 페이지](#)

ab123님이 좋아할 수도 있는 강의들

[인물로본일본의역사](#) [현대사회의윤리적쟁점들](#) [성의과학적이해](#) [미생물과미래융합기술](#) [전통시대동양의법과사회](#)

8개의 강의를 담았습니다.

학수번호	강의명	담당교수
COSE159	소프트웨어윤리	이원규 삭제
GEOG104	관광의역사와문화	천종호 삭제
GERM136	독일의대안문화	송민정 삭제
GERM139	독일예술의이해	전희원 삭제
HANM110	동양의지혜	김윤섭 삭제
HEED102	부모되기교육	김경미 삭제
HOEW125	한일교류의역사	김상준 삭제
HOKA103	한국전통문화의이해	김소연 삭제

사용자는 마이페이지에서 개인화 추천을 조회할 수 있다.

4. Experiment / 성능 평가

추천 알고리즘 평가

1. Contents-based Filtering Recommendation

Klue에 다수의 텍스트 리뷰들이 등록된 강의들의 경우, 키워드 추출과 유사도 기반 추천을 통해 유의미한 추천이 가능했다. 하지만 텍스트 리뷰가 매우 적게 등록된 강의들의 경우에는 쉽지 않았다.

⇒ 즉, Contents-based Filtering의 전통적인 문제점인 특정 아이템들의 메타 데이터 부족으로 인한 문제를 확인할 수 있었다.

2. Collaborative Filtering Recommendation

본 프로젝트에서 설계한 웹서비스는 상용화 단계가 아니므로, 결국 모델을 학습할 만한 대용량의 서비스 사용 history 데이터를 수집하는 데에 한계가 있음.

=> 즉, 학습 데이터의 부족으로 모델 학습에 한계가 있었으므로, 본 프로젝트 환경에서 활용할 수 없었다.

4. Experiment / 성능 평가

실시간 추천 서비스 아키텍처 평가

1. Data Pipeline

Kafka 메시지 큐 활용을 통해 비동기적으로 로그/추천 데이터를 처리하는 것이 가능했다. 또한, 파일 시스템에 메시지 데이터들을 보관하는 특성을 통해 향후 대용량 데이터가 발생하는 환경에서 실시간 데이터 유실 방지, 보관된 데이터에 대한 분석 등이 가능한 안정성을 확보할 수 있을 것으로 예상된다.

현재 Elasticsearch에 추천 결과를 저장하는 방식을 사용하고 있다. 향후 트래픽이 방대해지는 상황에서 실시간 추천 결과를 지속적으로 저장하게 되면, 추천 스토리지에 대한 Scale-out이 급격하게 요구될 것으로 예상된다.

2. Recommender Engine

본 프로젝트에서는 Spark를 on-premise 환경에서 실행하였지만, 추후 트래픽이 방대해지고, 원활한 서비스를 보장하기 위해 public cloud를 활용한 Scale-out이 필요할 것으로 예상된다.

5. Conclusion

□ Summary

- 수강평가 사이트 KLUE 내 강의 정보를 수집하고, 이를 바탕으로 contents-based filtering과 collaborative filtering을 종합한 hybrid recommendation system을 구현하였다. 이를 spark에 탑재해 서비스 사용자들의 로그 데이터를 실시간 처리하여, 개인 맞춤형 강의를 추천해주는 웹서비스를 구현하였다.
- 추천 결과를 확인하며 유저 취향과 강의평가 사이에 유의미한 상관관계가 존재함을 확인할 수 있었다.

□ Future work

- Collaborative Filtering Recommendation의 경우, 유저 로그 history를 바탕으로 모델을 발전시킨다는 특징이 있다. 현재 해당 알고리즘의 구축은 완료되었으나, 서비스가 초기 단계에 존재하기 때문에 조회 테이블의 크기가 부족해 실제 적용이 어려운 상태이다. 따라서 이는 추후 서비스 상용화 시 유저 기록이 어느정도 축적된 후에 서비스에 사용하고자 한다.
- 향후 서비스가 상용화된다면, 분산 환경 기반 오픈소스들의 Scale-out을 수행하여, 원활한 서비스를 보장하고자 한다.