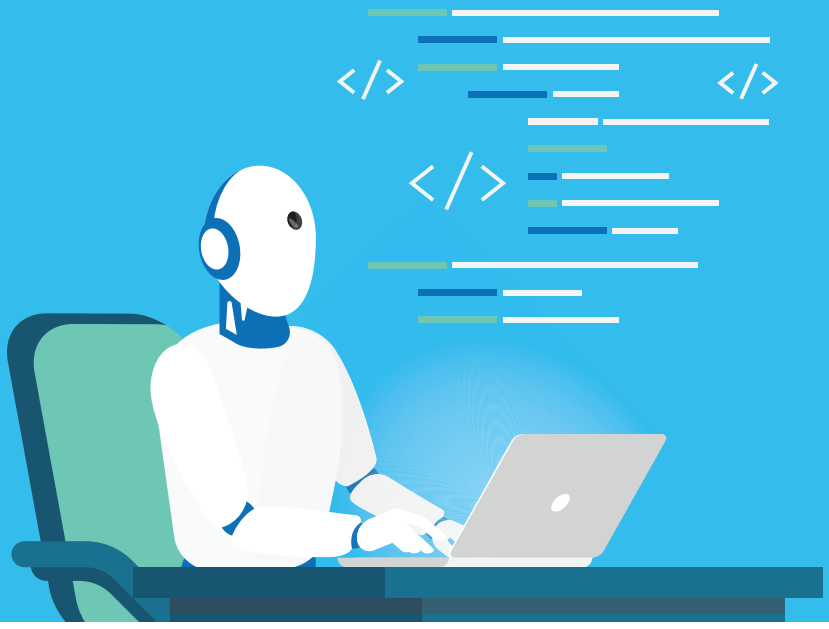
코딩교육 열풍과 현주소 - 4

인공지능을 위한 데이터 수집

글. 김승욱 Rloha 대표, 데이터 분석 교육 및 컨설팅
'빅데이터 분석, R좀 R려줘', 'R 데이터 분석' 등 관련 칼럼 및 강의 진행



인공지능? 인간지능? 여태 사람이 하던 것을 기계가 하게끔 시키려면 어떻게 해야 할까? 기계라고 하면 로봇팔을 떠올리는 사람도 있고, 컴퓨터를 떠올리는 사람도 있다. 아무래도 대부분의 사람은 컴퓨터보다는 로봇팔을 떠올리지 않을까 하여 로봇팔을 기준으로 설명하도록 하겠다.

그림 1. 커피 제조 중인 로봇팔 (www.fnnews.com/news/201903141021428628)

일단 로봇팔을 만들었다고 하자. 여기서 바로 전원을 연결하면 로봇팔이 일을 시작할까? 예를 들어 전원이 연결되었다는 빨간 불이 들어올 수 있겠지만 별다른 일을 하지 않을 것이다. 여기서 별다른 일을 하지 않는다는 문장이 어색하게 보일 수 있다. 하지만 여기서 언급하는 로봇팔에 특정한 동작 또는 기능을 수행하도록 조치를 취했다고 하지는 않았다. 그럼 다시 처음으로 돌아가서 특정 동작을 수행하도록 조치가 된 로봇팔을 조립하고 전원을 넣으면? 그리고 특정 버튼을 누르면? 이제 로봇팔은 아메리카노를 내린다던가, 카페라떼를 내린다던가 하는 지정된 작업을 수행할 것이다.

조금 더 들어가 보자. 앞에서 언급한 동작, 기능, 조치라는 단어를 보다 전문용어로 바꿀 필요가 있다. 여기서의 동작과 기능은 알고리듬, 조치는 프로그래밍을 뜻한다. 그런데 커피를 내리는 알고리듬을 로봇팔에 프로그래밍한다고 해서 로봇팔이 인공지능을 가진 것이라고 할 수 있을까? 빈 종이컵을 들어서 특정 위치에 가져다 놓고, 물을 받는 등 매우 단순하고 정해진 순서를 따를 뿐이다. 인공지능이라는 단어의 정의를 넓게 잡는다면 이 또한 인공지능이라고 할 수 있겠지만, 요즈음은 최소한 인공신경망을 활용한 기술 기반으로 특정 작업을 수행해야 충분히 인공지능이라고 할 수 있다. 그렇다면, 이런 단순한 작업 말고 인공신경망을 활용한 기술이 들어간다면, 이 로봇팔은 어떤 작업을 할 수 있을까?

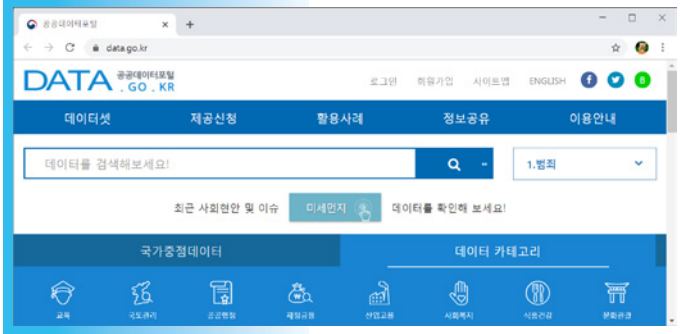
- 버튼이 아닌 음성인식 기반 주문접수
- 정해진 위치가 아닌 실시간으로 움직이는 사람의 손에 정확하게 커피잔 전달

위와 같은 작업을 수행하기 위해서는 사람의 경우 어떤 감각기관이 필요할까? 바로 귀와 눈이 되겠다. 단순히 어느 위치에 어떤 물건을 옮기는 작업은 여러 규칙을 가지는 코드의 조합으로 해결 가능하지만, 귀와 눈과 같은 감각기관을 대체하도록 만드는 것은 결코 쉽지 않다. 귀를 대체하려면 많이 들어야 하고, 눈을 대체하려면 많이 보아야 한다. 그래서 해당 기능을 구현하려고 하면 수많은 음성파일과 이미지(또는 영상) 파일이 필요하다. 자. 이제 그럼 데이터 수집에 대해 알아보도록 하자.

데이터를 확보하는 방법은 다양하다. USB 같은 이동식 저장장치와 이메일로 건네받는 방법 이외에 대표적으로 데이터베이스(이하 DB), API, 크롤링(crawling)이 있겠다. DB로부터 가져오는 경우는 보통 특정 조직에 속한 경우에 해당하기 때문에 생략한다. 그리고 API를 이용하는 경우는 이전 글에서 언급한 바 있지만, 보통 공공데이터 포털(www.data.go.kr)을 이용하는 경우가 많다. 크롤링 부분은 먼저 API를 활용한 데이터 수집을 먼저 알아본 이후에 하도록 하겠다.

공공데이터 포털의 국토교통부 아파트 실거래가 자료 API를 이용하려면 우측과 같이 따라 하도록 한다.

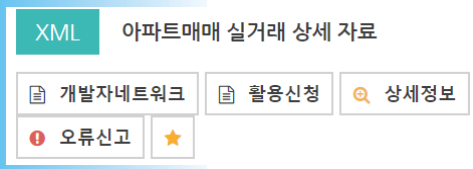
1. 공공데이터 포털 회원가입 및 로그인



2. 검색창에 [국토교통부 실거래] 검색

여기서 스크롤을 내리면 API가 나온다. 파일데이터에 있는 실거래가 정보를 사용하는 것이 아니다.

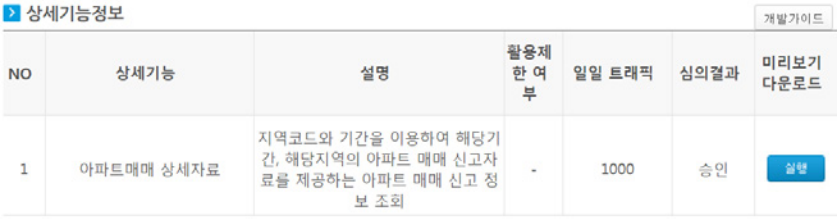
3. 아파트매매 실거래 상세자료 활용신청



4. 각종 설정

여기에서 시스템 유형은 일반, 활용 목적은 웹 사이트 개발이나 앱 개발을 제외한 나머지 선택지를 누르고, 상세기능정보와 라이선스표시의 체크박스를 클릭해주면 신청이 완료된다.

앞의 절차를 따라 하면 실거래가 정보를 받아올 수 있는 권한을 획득하게 된다. 정말로 내가 권한을 얻었는지 알아보도록 하자. **[마이페이지] > [오픈 API]** 위치로 이동하면 내가 신청한 API 목록을 확인할 수 있다. 여기서 직전에 신청한 API 페이지에 들어가서 스크롤을 내려 보면 다음과 같은 화면을 볼 수 있다.



상세기능정보

개발가이드

NO	상세기능	설명	활용제한 여부	일일 트래픽	심의결과	미리보기 다운로드
1	아파트매매 상세자료	지역코드와 기간을 이용하여 해당기간, 해당지역의 아파트 매매 신고자료를 제공하는 아파트 매매 신고 정보 조회	-	1000	승인	실행

[일일 트래픽]에는 1000이라는 숫자가 적혀 있는데, 이는 하루에 아파트매매 상세자료를 1,000번 요청 가능함을 뜻한다. 해당 항목은 관계기관과 합의로 조정이 가능한 경우가 대부분이고, 개인이 사용하는 경우 하루에도 1,000번씩 호출하는 경우는 드물기에 크게 신경 쓰지 않아도 된다. 그리고 **[미리보기 다운로드]** 항목에 실행 버튼을 눌러보도록 하자.

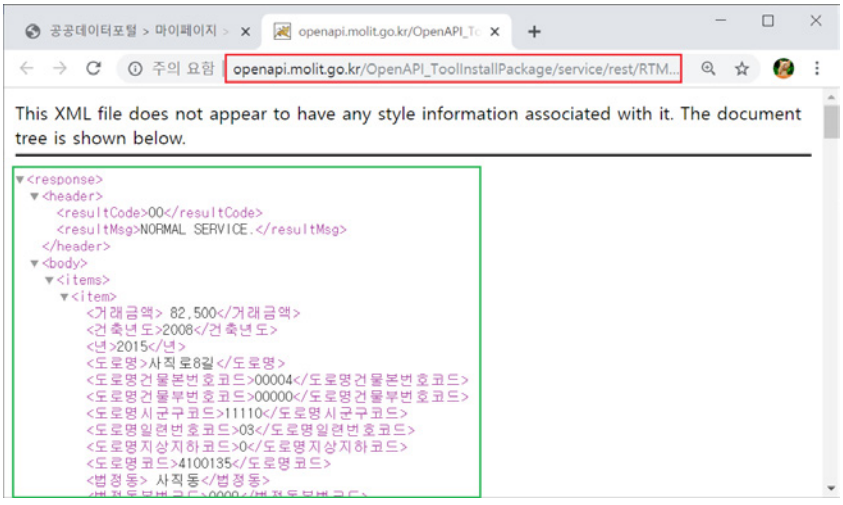
요청변수(Request Parameter)

초기화 닫기

항목명	샘플데이터	설명
ServiceKey	-	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호
numOfRows	10	한 페이지 결과 수
LAWD_CD	11110	지역코드
DEAL_YMD	201512	계약월

미리보기 XLS XML JSON CSV RDF

요청변수 항목의 [미리보기] 버튼을 또 눌러보면 다음과 같은 화면이 나온다.

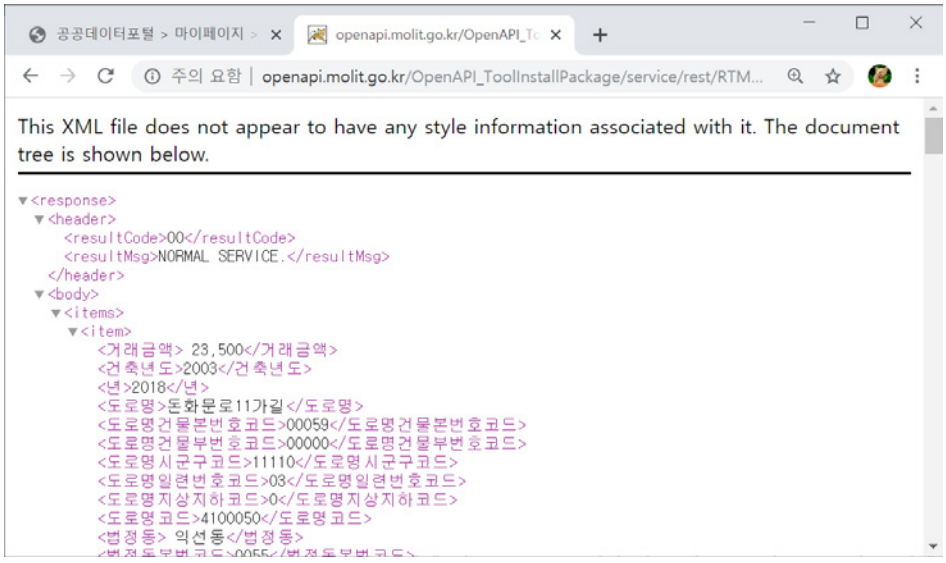


화면의 초록색 네모 부분이 아파트 실거래 정보이다. 조금 복잡해 보이는 이 형식은 XML이라는 것으로 인터넷 웹페이지의 코드를 구성하는 표준 형식이다. 하지만 여기서 문제가 있다. 내가 원하는 지역과 날짜의 아파트 실거래가 정보를 수집하려면 어떻게 해야 할까? 이제 그 방법을 알아보기 위해서 앞의 그림의 빨간 네모는 실거래가 정보를 조회할 수 있는 고유한 주소(URL)이다. 해당 주소를 한 번 살펴보자.

실선의 네모상자 안에 있는 주소는 우리가 바로 얻을 수 있는 주소이고, 아래 점선 네모상자 안에 추가로 줄 바꿈을 실시한 것을 보면 대략 어떤 정보가 API 정보 제공측(국토교통부 서버)에 전달되었는지 알 수 있다. 그리고 빨간 실선의 밑줄 내용은 **serviceKey**를 제외하고, 앞의 **[미리보기 다운로드]**에서 보았던 요청변수 목록과 일치한다는 것을 알 수 있다. 그렇다면, 각 항목에 기입되는 값을 바꿔주면 다른 값을 받을 수 있다. 다음 그림은 **DEAL_YMD**에 입력되는 값을 '201512' 대신 '201812'로 입력했을 때 나오는 화면이다.

```
http://openapi.molit.go.kr/OpenAPI_ToolInstallPackage/service
/rest/RTMSOBJSvc/getRTMSDataSvcAptTradeDev?serviceKey=2j
phw%2BFyE88rPf0gz5nICc9SyRbsIbYjD0VsGUiHplpkCUE%2FCQ0
2IP2vjacyoHETquky1enBlKXrTjpH%2BTsIZw%3D%3D&pageNo=1
&numOfRows=10&LAWD_CD=11110&DEAL_YMD=201512
```

```
http://openapi.molit.go.kr/OpenAPI_ToolInstallPackage/service
/rest/RTMSOBJSvc/getRTMSDataSvcAptTradeDev?
serviceKey=2jphw%2BFyE88rPf0gz5nICc9SyRbsIbYjD0VsGUiHplp
kCUE%2FCQ02IP2vjacyoHETquky1enBlKXrTjpH%2BTsIZw%3D%3
D&
pageNo=1&
numOfRows=10&
LAWD_CD=11110&
DEAL_YMD=201512
```



익선동의 아파트 매매정보가 가장 위에 출력되는 것을 확인할 수 있다. 요청변수의 보다 상세한 내용은 **[서비스정보]**의 참고문서(아파트 매매 상세자료 조회 기술문서.hwp)를 보면 알 수 있다.

서비스정보

일반 인증키 (UTF-8)	2jphw%2BFyE88rPf0gz5nICc9SyRbsIbYjD0VsGUiHplpkCUE%2FCQ02IP2vjacyoHETquky1enBlKXrTjpH%2BTsIZw%3D%3D 복사
End Point	http://openapi.molit.go.kr/OpenAPI_ToolInstallPackage/service/rest/RTMSOBJSvc/getRTMSDataSvcAptTradeDev?_wadl&type=xml
데이터포맷	XML
참고문서	아파트 매매 상세자료 조회 기술문서.hwp

요청변수 입력값 변경을 해보았으니 **serviceKey**를 보도록 하겠다. 해당 변수에 입력되는 값의 정식 명칭은 API 키(key)가 되겠다. 간단하게 말해서 비밀번호와 유사하다. 누구나 제한 없이 아파트 매매정보를 얻기 위해 요청을 많이 넣는다면, 서버에 부하가 갈 수 있다. 예를 들어서 어떤 사람이 1분에 10만 번의 요청을 넣는다고 하면, 해당 요청을 처리하느라 다른 이용자가 정상적으로 서비스를 이용하기 어렵거나, 서버에 장애를 유발하여 사용을 하지 못하게 되기도 한다. 그래서 이런 식으로 회원가입 및 개인별 별도의 API 키를 제공하는 것이다. 그리고 본 API의 경우 누구나 사용할 수 있기도 하고 무료이기 때문에 해당 키가 유출되더라도 큰 문제가 되지 않는다.

하지만 만일의 보안사고를 막기 위하여 언제나 API 키는 외부에 유출되지 않도록 관리에 유의하길 바란다. [1]

API를 활용하여 데이터 수집 및 정제하는 코드는 좌측의 QR 코드를 통해서 전체 내용을 확인할 수 있고 각 코드 블록마다 설명을 적어 놓았다. 아나콘다(Anaconda)를 설치하면서 같이 설치되는 주피터 노트북(Jupyter Notebook)으로 해당 코드를 실행시켜보면 되겠다.

이제 크롤링을 알아보자. 이전 글에서 살짝 다뤘던 크롤링(crawling)은 하베스팅(harvesting), 스크레이핑(scraping) 등 여러 명칭이 있으며 보통 앞에 인터넷을 뜻하는 웹(web)을 붙여서 언급한다. 애당초 크롤링은 웹에서 대량의 자료를 수집한 후 검색엔진에서 풍부한 검색 결과를 제공하기 위해 기술이 발전하였는데, 최근에는 데이터 분석과 인공지능 학습 목적으로 다양한 데이터를 수집하는 목적으로도 사용된다.
우선 가장 간단한 크롤링부터 해보도록 하겠다. 아나콘다의 주피터 노트북에서 다음의 코드를 실행하도록 하자.

```
1  import requests
2  from bs4 import BeautifulSoup
3
4  keywords = BeautifulSoup(requests.get("https://www.naver.com").text,
5                           "html.parser").select('[class=ah_k]')
6  [word.text for word in keywords]
```

앞의 코드는 네이버의 실시간 급상승 검색어를 수집하여 출력해주는 코드이다. 코드를 크게 세 부분으로 나누자면 다음과 같다.

- 데이터 수집 및 처리에 필요한 모듈 불러오기
- 목표 사이트의 소스코드 수집 및 필터링
- 코드 내부에 있는 텍스트(실시간 급상승 검색어) 추출

크롤링은 API보다 난이도가 높은 것이 일반적이지만, 네이버와 비슷한 포털사이트의 실시간 급상승 검색어를 가져오는 것은 크롤링 난이도 최하에 속한다. 앞의 코드와 API 활용 코드의 차이점이라면 별도의 API 키 없이 내가 원하는 대상 사이트의 정보를 가져올 수 있다는 것이다. 하지만 목표 사이트의 정보를 무작정 수집하면 안 된다. 크롤링은 데이터를 수집하는 방법 중 비공식적이지만 기술만 있다면 연구자에게 상당히 매력적인 방법이기 때문에 무턱대고 사용하는 경향이 있는데, 다음의 질문을 자기 자신이나 주변 사람들에게 물어보도록 하자.

- 수집 데이터양이 많은가?
- 상업적으로 사용할 목적이 있는가?

먼저 수집 데이터양이 많다면 정식으로 서비스 제공 측에 정당한 대가를 지불하고 요청을 해야 한다. 본래는 양이 적건 많건 해당 서비스의 지적재산이기 때문에 양해를 구하는 것이 맞지만 일단 양이 많을 경우, API 부분에서 비슷한 예시를 들었는데, 서비스에 부하(트래픽 등)를 유발하여 서비스에 장애가 발생할 수 있다. 그리하여 일반 서비스 제공 측에서는 과도한 크롤링 시도가 발생하였을 때 해당 IP 주소를 차단할 수 있다. 예를 들어서

1) 이 글에서는 필자의 API 키를 공개하고 있지만, 여러분이 이 글을 읽기 전에 변경 예정이라 해당 키가 동작하지 않을 것이다.

네이버의 실시간 급상승 검색어를 1초당 한 번씩 한 시간 동안 수집하는 코드를 실행하면 매우 높은 확률로 일정 기간 IP 주소 차단을 당해서 한동안 네이버 사이트에 접속을 하지 못할 가능성이 크다. 심지어 어떤 사이트는 이러한 크롤링 시도가 적발되는 동시에 영구적으로 접속을 하지 못하도록 차단하기도 한다.

앞의 내용을 참고해서 수집을 조금만 한다고 마냥 괜찮은 것도 아니다. 크롤링을 어디까지 허용범위를 기술한 문서는 각 사이트 최상위 위치에 robots.txt라는 파일이다.
예를 들어 다음 그림은 네이버의 robots.txt 문서인데 www.naver.com/robots.txt 주소로 접근하면 볼 수 있는 내용이다.

이 문서에서 안내하는 내용은 모든 크롤링 봇(수집하는 자동화 로봇)은 단순하게 '모든 에이전트에 대하여 사이트 내부 어떠한 정보도 수집할 수 없고, 어떠한 정보도 허용하지 않는다.'라는 뜻이다. 그리고 온라인 쇼핑몰 사이트인 gmarket의 경우 '모든 에이전트는 사이트의 모든 정보를 수집할 수 있다.'라는 완전히 반대의 문서를 제시하고 있다.

이처럼 특정 사이트를 크롤링하는 경우 robots.txt를 꼭 확인하고 실시해야 불필요한 법적 분쟁을 피할 수 있고, 허용하는 범위 내에서 크롤링을 실시하더라도 막대한 양을 크롤링하면 문제가 생길 수 있으니 유의하기 바란다. 꼭 네이버 실시간 급상승 검색어가 아니라도 daum.net, zum.com 사이트에서도 비슷한 검색어를 수집할 수 있다. 그런데 과연 인공지능을 구현하고 관련 시스템을 구축하는데 이런 급상승 검색어가 쓸모가 있을까? 억지로 찾는다면 있기야 하겠지만 단편적으로 봤을 때는 도저히 사용 용도를 알 수 없다. 그러면 우리는 어떤 자료를 수집해야 할까? 문제 정의가 가장 중요하다. 이렇게 크롤링의 기술을 안다고 해도 문제해결을 위하여 적절한 데이터를 선정하지 못한다면 전혀 쓸모가 없다. 크롤링 기술은 시중에 책이 많으니 기술 관련 자세한 서술 보다는 해당 기술을 어떻게 활용할지 사례를 들어 보다 자세하게 알아보도록 하자.

영화

영화의 흥행 여부를 아는 것은 매우 중요하다. 그것도 미리 알 수 있다면 투자자 관점에서는 제법 유용한 정보가 될 것이다. 그렇다면 투자자의 관점에서 생각해보자. 투자는 영화가 상영되기 전에 집행이 되어야 할 것이다. 투자가 진행될 때 즈음에 알 수 있는 정보는 어떤 것이 있을까? 상황에 따라 다르겠지만, 영화 제목, 출연진, 감독, 상영시간, 제작비용 등이 되겠다. 해당 정보를 수집하는 것이 단편적인 수집 범위가 되겠다. 어떻게 보면 이 정보만 가지고 얼마나 흥행을 할지 알 수 있다는 것이 불가능하다고 생각될 수 있다. 그리고 아직 모호한 것이 있다. 과연 흥행의 기준은 무엇인가? 여러분도 잠깐 생각을 해보셨으면 좋겠다. 과연 흥행의 기준을 어떤 것으로 잡아야 할 것인가? 투자자의 관점에서는 투자 수익이 기간 및 시중 금리 대비 높게 나오는 것이 흥행의 기준이 될 수 있고, 감독의 입장에서는 기존 제작 영화보다 많은 관객유치일 수 있고, 배급사 입장에서는 영화 제작비 규모 대비 높은 매출일 수 있다. 이처럼 기준이 다양하기에 기준을 명시하고 그에 알맞은 자료를 수집하고 평가를 해야 한다.
데이터 수집의 범위를 좀 더 넓혀보자. 웹상에 있다고 하기는 어렵지만, 대본이 있겠다. 대본을 읽어보면 주인공의 비중은 어떠한지, 극적인 묘사가 있는지 다양한 정보를 얻을 수 있다. 하지만 이것은 사람이 직접 눈으로 읽었을 때 알 수 있는 것이다. 주인공의 비중이야 주인공이 전체 대본 중에서 몇 번 말하고 타인이 몇 번 언급하

는지 데이터 처리를 하여 산출할 수 있다. 하지만 극 중의 긴장감이나 특정 배역의 배신이나 이런 부분은 적당히 코드를 작성해서는 알 수 없다. 이때 들어가는 것이 인공지능 기반의 자연어 처리(NLP, Natural Language Processing)가 필요하다. 그리고 꼭 인공지능의 기술이 들어가지 않더라도 배우나 감독의 이전 작품정보를 수집하는 것도 방법이다. 예를 들어 감독의 경우 천만 관객 영화가 몇 건이 있었는지, 배우의 경우 기존에 촬영한 광고나 드라마가 몇 건 있었는지 등이다.

부동산

문제 정의를 해보자. 누구를 위한 분석인가? 정책 결정자? 부동산 구매자? 부동산 임대업자? 정책 결정자라고 하면 부동산 거래가 어디서 과열되는지 상승률, 거래 건수를 중점적으로 수집할 수 있다. 그리고 새집 마련을 고민하는 신혼부부의 경우 집을 사더라도 향후에 가격이 오를 만한 집을 찾을 수 있기에 부동산 거래 정보(특히 단위 면적당 단가 등)를 중점적으로 수집할 수 있다. 그리고 임대업자의 경우 독특한 정보를 수집해야 할 수 있는데, 예를 들어 부동산 임대업자가 오피스텔 소유자라면 소유 부동산의 공실 여부가 문제가 될 수 있다. 예를 들어 원룸 오피스텔 한 채에서 나오는 월세가 60만 원이라고 할 경우 두 달만 누적이 되어도 120만 원의 기회비용이 발생하기에 어떤 세입자가 계약 기간 이전에 빠져나갈지 판단하는 모델이 필요하다. 이를 위해서는 계약 시점, 계약 금액, 세입자의 성별/나이/출신/학력 등 다양한 정보가 필요할 수 있다. 이와 비슷하게는 특정 서비스의 사용자 이탈예측이 있으니 관련 직종에 종사하시는 분들은 참고하면 되겠다.

영상/이미지

안면 인식을 예를 들어보자. 그리고 문제 정의. 무엇을 위한 인공지능을 만들 것인가? 범죄자 식별? 면접자 표정 인식? 그 외에도 다양한 선택지가 있지만 면접자의 표정 인식을 주제로 해보자.

다시 고민해보자. 면접자의 표정을 왜 인식해야 하는가? 면접자가 본인의 소개를 하던지 이전에 수행한 프로젝트를 설명하는 부분은 언어의 영역이기 때문에 앞에서 언급한 자연어 처리에 음성인식을 더한 인공지능 기술이 들어가야 한다. 표정은 눈으로 보고 인지하는 것이기에 사람의 눈을 대체할 수 있는 이미지 및 영상을 인식하는 인공지능 기술이 사용되어야 한다. 그래서 면접자의 사진이나 영상을 수집해야 하는데 여기서 면접자의 표정을 잡아내야 한다. 그런데 무작정 면접의 처음부터 끝까지 표정을 잡아내야 할까? 그것도 아니다. 면접자가 특정 단어를 들었을 때 표정이 일그러지는지, 기쁜 표정을 짓는지 알아내야 할 것이다. 이 부분은 굉장히 어려운 기술이 들어가니 기획자는 매우 조심해서 접근해야 한다. 이 정도의 기술을 활용하고자 하는 기업의 경우 면접자가 수십 명이 될 가능성이 매우 높다. 그리고 정확한 판단을 위해서 면접관이나 면접자가 말하는 단어 단어마다 영상을 끊어야 할 수 있는데 이렇게 하려면 단순 영상처리가 아닌 음성인식까지 수반되어야 한다. 아무튼 모든 기술적 허들을 넘었을 때, 결과적으로 판단할 수 있는 것은 특정 주제에 대하여 면접자가 얼마나 진실된 답변을 하는가? 정도가 되겠다. 사실 이 정도는 고도의 인공지능보다는 기존의 인간지능을 활용하는 것이 당장의 현실적인 해결책에 가깝다고 할 수 있다.

앞에 소개한 예시 말고도 정말 많은 예시가 있겠지만, 전부 다루기에는 다소 독자분들의 흥미가 떨어질 듯하여 3개 예시만 제시하였다. 아직은 고전적인 방법을 쓰는 곳이 많기에 이렇게 인공지능을 접목한다면 그 성능이 비약적으로 좋아진다고는 하지만, 앞에서 예시를 들었던 면접자의 표정 인식처럼 인공지능을 활용하는 것은 생각보다 높은 기술과 많은 자원이 소요되기 때문에 신중하게 결정을 해야 한다. 물론 인공지능을 중점으로 많은 것을 소개하고 있지만 절대적으로 인공지능이 만능이 아니라는 것을 숙지해야 한다. 향후에는 인공지능이 만능이 될지 몰라도 지금은 차라리 손으로 직접 입력하는 것이 훨씬 빠르고 정확할 수 있다. 그럼 다음에는 이렇게 수집한 데이터를 제대로 활용하기 위한 인공지능 기초와 체계적인 데이터 관리에 대해서 알아보도록 하겠다.