코딩교육 열풍과 현주소 - 3

인공지능을 위한 오픈소스와 Python

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

많고 많은 도구 중에 왜 오픈소스일까? 그중에서 왜 Python일까? 이전 글에서 언급한 바 있지만 각종 통계 프로그램이나 RPA(Robotic Process Automation) 관련 도구가 초반에는 훨씬 편할 수 있다. 하지만 당장 일이 급한 경우가 아니라면 넓게 보아야 하겠다. 시작은 겨우 파일을 읽고 저장하는 수준일 수 있으나 나중에 컴퓨터가 혼자 알아서 척척 돌아가는 시스템을 만들게 된다면 그만큼 뿌듯한 일도 없을 것이다. 심지어 단순 조건의 조합이 아니라 사람의 눈과 귀 같은 감각기관을 대체하는 수준의 기능까지 들어가면 금상첨화 아닐까? 필자가 아무리 이렇게 말을 해도 프로그래밍이나 관련 기술을 경험해보지 못한 분들은 갸우뚱하신다. 그러면서 공통으로 질문을 주시는데 당시 답변했던 내용과 더불어 보다 자세하게 설명한 후 Python을 시작해보고자 한다.

최근 Python 강의를 들으시던 수강생분이 마지막 쉬는 시간에 이런 질문을 하셨다. "엑셀을 놔두고 왜 Python 같은 오픈소스를 공부해야 합니까?" 평소에 이런 질문을 상당히 자주 받는다. 평소 같았으면

"오픈소스는 공짜죠!"

라고 말하지 않았을까? 하지만 이 말은 아주 가볍게 질의하는 사람에게 해주는 답변이다. 그냥 막연하게 배우는 사람들. 돈 잘 번다고, 섹시한 직업이라고 어디서 들어서는 공부량이 얼마나 되는지 모른 채 천진난만한 얼굴을 한 사람들. 하지만 이때는 조금 달랐다. 표정도 진지했고 앞의 내용도 충실히 따라온 수강생이었기에 조금은 진지하게 답변을 하기로 했다.

'대용량 처리', '재현성', '자동화', '확장성'

네 단어를 말해주었다. 오픈소스는 매우 다양하지만 Python이나 R 같은 데이터 분석으로 많이 사용되는 언어 기준으로 생각하면 되겠다. 다음 글을 읽기 전까지 독자분들도 왜 오픈소스를 공부해야 하는지 조금 고민해보셨으면 한다.

대용량 처리

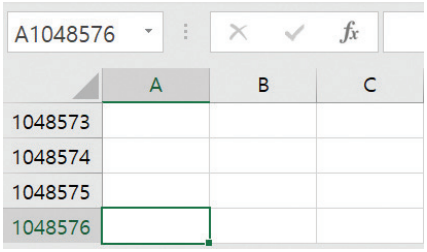

그림 1. 엑셀의 마지막 행

엑셀을 사용하면 약 105만 줄 정도 데이터를 불러올 수 있다. 이보다 더 큰 데이터는 엑셀이 불러오다가 '파일을 완전하게 로드하지 못했습니다.'라는 메시지를 띄우면서 그 뒤에 있는 데이터는 잘려서 불러올 수 없다.

물론 엑셀의 경우 Microsoft의 Access 또는 Power BI를 사용하거나 데이터베이스와 같이 사용한다면 보다 수월하게 대용량의 데이터를 다룰 수 있겠으나 컴퓨터 언어를 사용하여 직접 다루는 것보다 속도가 훨씬 느리다. 점점 데이터가 커지는 요즈음에 1,000만 줄 이상이나 GB(기가바이트) 단위 데이터를 다루기 어렵다는 것은 다가오는 미래를 대비하기 어렵다는 것을 뜻한다. 물론 엑셀을 비롯하여 좋은 프로그램은 참 많다. 그런 프로그램은 사용자로 하여금 제법 편하게 사용할 수 있도록 설계되었지만, 직전에 언급한 대용량 데이터를 다루기 어려운 것과 같이 성능이 상대적으로 좋지 않다. 이렇게 말하면 혹자는 다음과 같은 질문을 한다.

"그럼 빠르고 쓰기 편한 프로그램은 없나요?"

딱히 없는 것은 아닌데 엑셀만큼 편하고 다재다능한 프로그램은 없으며, 가격은 수천만 원 이상 호가하는 프로그램이 많다. 그러나 대용량 처리와 편의성의 관점에서 Python과 같은 오픈소스를 사용하게 되면, 코드를 직접 힘들게 입력하는 등 편의성을 잃는 대신 막강한 데이터 처리능력을 얻는다고 보면 되겠다. 그럼에도 이 부분만 보면 굳이 언어를 배워야 할 이유가 없어 보인다. 그렇기에 오픈소스의 다른 매력을 소개하고자 한다.

재현성

엑셀로 자료를 정리해본 적 있는가? 자료 정리라고 하면 상당히 광범위하지만 본인이 이전에 했던 작업을 떠올려보자. 데이터를 정리하는 작업은 여러 가지 하위 작업단위로 나눌 수 있다. 데이터를 분리하거나, 합치거나, 특정 값을 기준으로 요약하는 등 최종 결과물을 생산하기 위해 자잘한 작업단위뿐만 아니라 때로는 매우 반복적인 일이 있을 수도 있다. 이런 작업을 하기 위해서는 학습을 통해 기술을 익혀야 하고 일반적으로는 일회성으로 끝나기 마련이다. 그런데 다른 사람이 이와 비슷한 작업을 한다면 또 똑같이 관련 기술을 공부해서 일을 처리할 것이다. 물론 두 사람이 이런 일을 했다면 두 사람의 업무 능력은 이전보다 향상되었을 수 있다고 기대할 수 있다. 하지만 과연 이와 같은 작업 방식이 효율적인가?

직전에 언급한 작업유형은 일의 시작과 끝만 보이기 때문에 일을 처리하는 사람이 바뀌거나 기존의 작업 방식을 잊어버린 경우 해당 작업을 위해서 다시 공부하고 주변 사람에게 물어보는 등 같은 일을 할지라도 의도한 시간보다 더 걸릴 가능성이 크다. 그리고 이와 같은 작업은 키보드와 마우스를 수십, 수백 번 조작해서 최종 산출물을 생산할 것이다. 하지만 이를 코드로 남겨서 공유한다면 어떨까?

모든 작업을 한 줄 한 줄에 기록하여 처음 데이터를 열람하는 것부터 새로 저장하고 기록까지 코드에 고스란히 담겨서 저장된다. 즉, 엑셀이었다면 마우스와 키보드로 하던 작업을 코드로 바꿔서 처리하게 되는 것이다. 이는 마치 일의 시작부터 끝까지 동영상을 찍는 것과 같다. 내가 일하는 모습을 고스란히 영상에 담는다면, 다시 영상을 되돌려보며 어떤 부분에서 잘못되었는지, 불필요한 작업은 없었는지, 보다 객관적으로 확인하고 필요 시 개선할 수 있다.

작성된 코드의 수준에 따라 다르겠지만, 코드가 작성된 이후에는 본인뿐만 아니라 조직은 엄청난 생산성을 확보하게 된다. 이제 비슷한 자료를 코드에 통과시키면 작은 데이터의 경우 수 초 내에 처리되기 때문에 작업자는 상

대적으로 작업 결과물 검토나 다른 중요한 일에 많은 시간을 할애할 수 있게 된다.
재현성은 특히 학술 분야에서 강조되고 있는 추세이다. 이전에 수행했던 연구가 번번이 재현에 실패하면서 학계는 점점 연구에 사용된 데이터나 코드를 제출하도록 권장하고 분위기로 가고 있다. 이는 단순 작업을 코드로 구현하는 것을 넘어 결과의 신뢰성을 확보하기 위한 움직임이며 학계뿐만 아니라 업계에서도 이는 매우 중요한 사안이다. 다음으로 재현성만큼 강력한 자동화를 알아보자.

자동화

기존 작업을 코드로 작성하는 것. 물론 이 부분도 자동화의 일부라고 할 수 있다. 여기서 생각하는 자동화는 한 걸음 더 나아가서 보다 고급스러운 업무를 말한다. 예를 들어 이메일 자동 발송, 메신저 알림, 뉴스 댓글 수집, 파일 자동 분류, 수백 개 파일 자동처리 등 단순 계산의 자동화를 넘어 보다 다양한 업무의 자동화를 의미한다.
업무의 자동화를 위해 복잡하고 긴 코드를 작성할 수 있지만, API를 이용하기도 한다. 필자가 주로 이용하는 API[1]는 공공데이터 포털과 Google Map이다. 공공데이터 포털에서는 각 분야의 다양한 자료를 조회하고 추출할 수 있는 API를 제공한다.

그림 2. 공공데이터 포털 API 화면

이 중에서 날씨 정보, 위치 정보, 부동산 정보는 사람들에게 단연 인기 있는 자료이다. 만약 API를 이용하지 않는다면 자료를 제공하는 담당자도 매번 자료를 최신화하여 포털사이트에 등록해야 하고, 받아보는 사람도 매번 포털사이트에 등록된 최신 파일을 내려받아야 하는 번거로움이 있다. 하지만 API를 활용한다면 자료가 쌓여 있는 데이터베이스에 접근하는 권한[2]을 얻어 사용자가 데이터를 바로바로 받아올 수 있기에 중간 관리자를 거치거나 일일이 사이트에 접속하는 번거로움을 덜 수 있다.
추가로 필자가 애용하는 Google Map API의 경우 지도상에 특정 주소를 표기하고자 할 때 활용한다. 지하철역이나 특정 시설의 위치는 보통 한글 도로명 주소로 기록되어있다. 하지만 지도에 이를 표기해야 하는 경우 위도와 경도로 이루어진 지리 좌표계로 변환시켜야 한다. 다음 그림은 그 예시이다.

	hos_kr	group	long	lat
서산시 우리요양병원		1	126.4532	36.77105
순천시 은성복지회		1	127.4872	34.95064
광양시 노경회		1	127.7470	34.98656
상주시 하은실버홈		1	128.2964	36.36536
아산시 행복한쉼터		1	127.0741	36.85755
군산시 원광효도요양병원		1	126.7357	35.96751
청송군 청송노인종합센터		1	129.0721	36.45703
김천시 경북도립김천노인전문요양병원		1	128.0669	36.23306
성주군 가나안노인전문요양병원		1	128.2819	35.91917
고령군 고령효사랑요양병원효사랑복지센터		1	128.2708	35.72850

그림 3. Google Map API를 활용한 지오코딩 예제

첫 번째 자료인 '서산시 우리요양병원'을 API를 통해서 정보를 전달하면, 126.4532와 36.77105라는 위경도 좌표 정보를 준다. 이것을 기반으로 지도상에 빨간 점을 찍은 것이다. 하지만 주소만 주어지는 경우 사람이 직접 위경도 위치 정보를 뽑기는 매우 어렵다. 소수점 셋째 자리에서 숫자가 하나만 달라져도 기존 위치에서 꽤 멀리 떨어

1) API(Application Programming Interface, 응용 프로그램 프로그래밍 인터페이스)는 응용 프로그램에서 사용할 수 있도록, 운영체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스
2) 정확하게는 API를 사용할 수 있는 API 키를 발급받는 것을 뜻한다.

진 곳의 좌표로 인식되기 때문에 체계화된 시스템으로 추출하지 않으면 잘못 처리할 가능성이 매우 높다. 그리하여 이런 경우는 기존에 구축된 자료와 시스템을 활용하여 일을 처리하는 편이 정확하고 빠르고 효과적이다. 단, API는 무료뿐만 아니라 유료도 있기 때문에 잘 알아보고 사용하는 것이 좋다.
이렇게 코드를 활용해서 업무를 보다 효율적이고 빠르고 정확하게 할 수 있어 최근 기업에서는 R이나 Python을 활용한 업무 자동화 강의를 많이 요청하고 있다. 주 근무 52시간 제도의 영향도 있고, 조직 구성원이 보다 중요한 업무에 집중할 수 있도록 전사적으로 역량을 강화하는데 관심이 많기 때문이기도 하다.

확장성

Python의 경우 본래 개발용 언어이다. 정확하게는 범용 언어이다. 즉, 이곳저곳에서 활용될 수 있다는 뜻이다. 본래의 기능을 넘어 데이터 분석 분야로 크게 영역을 확장하면서 프로그래머뿐만 아니라 학생도 직장인도 많이 접하는 언어가 되었다. 덕분에 사람들은 데이터 분석을 위해서 Python을 배웠다면 프로그램을 만들거나 웹페이지를 만들거나 다양한 방면으로 확장이 가능하다. 경우에 따라서는 간단한 게임 개발이나 라즈베리 파이[3]를 조작하는 코드를 작성할 수 있다.

몇 년 전에 지인이 본인의 코딩 실력을 한껏 뽐내서 데이터 분석 이외의 것을 개발했는데 바로 중고거래 알림이(가칭)라는 프로그램이다. 당시 지인은 특정 컴퓨터 부품을 사기 위해서 중고거래 사이트를 자주 접속했다고 한다. 그런데 물건이 언제 등록될지 모르니 접속을 하더라도 허탕을 치는 경우가 많고, 혹여나 물건이 등록되더라도 가격이 맞지 않는 경우가 있어서 시간을 많이 빼기는 상황이 발생했다. 그리하여 개발한 프로그램이 중고거래 알림이다.

해당 프로그램의 동작 원리는 다음과 같다.

- 주기적(5분~30분)으로 최근에 등록된 게시글 감지
- 해당 게시글 제목에 내가 원하는 제품명이 있는지 확인
- 설정 가격보다 싼 제품이라고 판단되는 경우 메신저 알람
- 신규 게시글 제목 추출
- 가격 정보 추출
- 직접 설정한 가격 정보와 비교

필자는 지인이 해당 프로그램을 개발했다는 말을 듣고 재능 낭비 아니냐며 놀리기도 했지만, 만약 이 사람이 일회성으로 사용하지 않고, 다음에도 재활용한다면 충분히 개발에 투자한 시간과 노력을 회수할 수 있다고 본다. 그리고 해당 프로그램을 개발할 때 사용한 코드는 다른 유사한 프로그램을 개발할 때 유용하게 활용할 수 있으므로 충분히 가치 있는 프로그램이라고 할 수 있다. 앞의 사례처럼 단순히 내 컴퓨터 내부에서만 동작하는 코드를 작성하는 것이 아니라 외부의 데이터를 수집하고, 스마트폰에 설치된 메신저에 메시지까지 보내도록 하는 것이 바로 데이터 분석에 국한된 사용을 넘어선 언어의 확장[4]이라고 할 수 있겠다.

Python 설치

이제 드디어 코딩 공부가 필요하다. 좋다. 이런 말 말고, 직접 시작해보도록 하자. 그러면 어떤 언어를 할 것인가? 사실 이번뿐만 아니라 이전의 글에서도 언급을 많이 했던 언어가 하나 있다. 바로 Python이다. Python을 설치하자. 그리고 향후 모든 설명은 Windows 10 운영체제를 기준으로 하며 이보다 낮은 버전의 운영체제를 가지고 있다면 제발 업그레이드를 하도록 하자. Python을 설치하는 방법은 다양하다. 그중에서 가장 간편한 아나콘

3) 라즈베리 파이(Raspberry Pi): 영국 잉글랜드의 라즈베리 파이 재단이 학교와 개발도상국에서 기초 컴퓨터 과학의 교육을 증진하기 위해 개발한 신용카드 크기의 싱글 보드 컴퓨터이다.
4) 사실 파이썬은 개발용 언어이기 때문에 오히려 파이썬으로 전문적인 데이터 분석을 하는 그 자체가 확장이라고 할 수 있다.

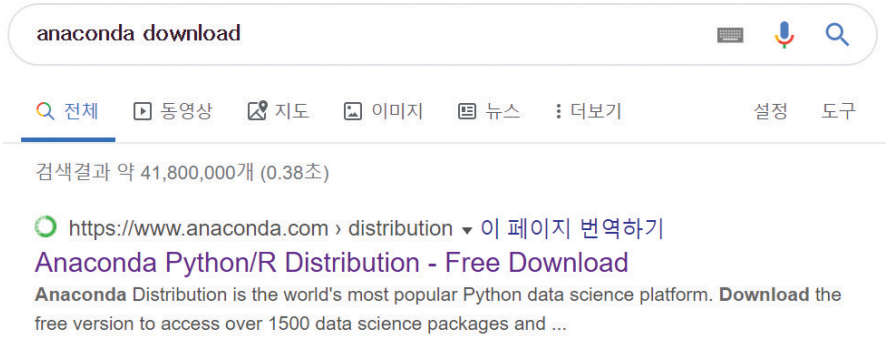

그림 4. 아나콘다 다운로드 경로 검색

다(Anaconda)를 설치하려고 한다. Google에서 'anaconda download'로 검색을 해보자.
본인의 운영체제에 맞는 버전의 파일을 다운로드받아 설치하자. 설치 방법은 블로그 등 다른 곳에 친절하게 나와 있어 여기에서 별도의 기술을 하지는 않겠다. 단, 본인 PC가 32bit인지 64bit인지 잘 모른다면 지금 Python을 공부할 게 아니라 컴퓨터를 공부해야 하는 상황이니 너무 무리하지 말자.

그나저나 왜 아나콘다를 설치한 것일까? 물론 순수하게 Python을 설치하고 이것저것 설정할 수 있긴 하지만 아나콘다를 설치하면 다양한 설정과 추가기능 설치를 한 번에 해결할 수 있기 때문이다. 특히 초심자는 순수하게 Python을 설치하는 경우 복잡한 설정과 명령어 입력 때문에 자칫 시작도 전에 포기하는 경우가 많으니 되도록 아나콘다를 설치하도록 하자. 아나콘다를 설치하면 [그림 5]와 같은 아이콘을 볼 수 있다.

그림 5. 아나콘다 설치 후 확인 가능한 아이콘 모음

앞으로 설명하는 내용은 주피터 노트북(Jupyter Notebook) 기준으로 기술될 예정이다. 주피터 노트북 말고도 스파이더(Spyder)도 Python을 사용할 수 있는 도구이긴 한데 필자는 한동안 해당 프로그램을 사용하다가 UI에 실망해서 더 이상 사용하지 않는다. 혹시나 향후 개선이 된다면 다시 사용해볼 의향이 있지만 당장은 아니다.
주피터 노트북이나 스파이더에서만 Python을 사용할 수 있는 것은 아니다. 파이참(PyCharm)도 있고, Microsoft의 VS Code(Visual Studio Code)도 있다. 하지만 향후 설명하는 내용이 Python 프로그램 개발이 아닌 데이터 분석과 인공지능에 맞춰질 예정이기도 하고, 초심자가 사용하기 편한 주피터 노트북 기준으로 기술되는 점 양해 바란다.

일단 여기까지 온 것만으로도 자신에게 큰 변화를 준 것이다. 너무 서두르진 말자. 빨리 배우고 써먹으면 좋지만 무리하다가 의지가 꺾이면 그 나름대로 문제가 된다. 이제 할 일은 다음 글을 기다리면서 읽을 책을 구하러 서점으로 가는 것이다. 그리고 서점에서 본인의 수준과 관심사에 맞는 책을 구매하여 조금씩 실습을 해보는 것을 권장한다. 단, 주의점은 Python 책의 경우 개발자를 위한 책과 데이터 분석가를 위한 책으로 크게 양분이 되니 이 점 주의해서 선택하길 바란다. 표지만 보고 현혹되지 말고 목차까지 같이 꼼꼼하게 읽고 선택하길 바란다.

그럼 공부는 어떻게 해야 할까? 막상 책을 산다고 끝나는 것은 아니다. 필자의 경우는 책으로 하는 공부를 선호

하는 편이다. 편하기로 따지면 동영상 강의를 듣는 것이 최고이긴 한데, 동영상 강의를 들으면 워낙 산만해서 집중하기가 어렵다. 게다가 동영상 강의는 내가 원하는 부분만 쏙쏙 뽑아서 취하기가 어려우므로 원하는 부분이 나오기를 기다리다 보면 졸기 일쑤다. 그렇다고 동영상 강의가 나쁘다는 것은 아니다. 최근 유튜브가 유행하듯 동영상 강의의 경우 특히 사전지식이 전무할 경우 차근차근 실행화면이나 애니메이션을 같이 보면서 이해할 수 있어 정적인 글을 읽고 함의를 유추하는 등 머리가 덜 피로하다는 장점이 있다. 이 부분은 본인의 성향에 따라 결정되는 것이기 때문에 더는 언급하지 않겠다.

공부 방법과 효율은 누구나 고민하는 주제이다. 필자가 수천 시간 이상 교육을 하면서 코딩을 빨리 배우는 사람의 특징과 습관은 다음과 같다.

키보드와 친해져라

거의 모든 버튼이나 작업을 마우스로 하려고 하는 사람이 많다. 그런데 마우스로 이래저래 누르다 보면 작업의 흐름이 끊기고 결국 일 처리가 늦어진다. 최소한의 코드 실행이라도 단축키를 사용하도록 하자. 그리고 영문 키보드 타자 속도가 빠른 경우 굉장히 유리하다. 본인이 생각하는 것을 거침없이 표현할 수 있기 때문이며, 영타 속도가 느린 경우(보통 오타도 많이 낸다) 답답함을 느껴 쉬이 짜증도 나고 금방 싫증이 날 것이다.

영어는 나의 힘

정보의 보고는 영문서에 있다. 최근에는 한글 번역이 잘 되어있거나 블로그를 작성하는 사람이 많아서 이전보다는 괜찮긴 하지만, 여전히 고급 지식은 영어로 되어있다. 특히 최신 논문이 그렇고, 대부분의 프로그래밍 언어 공식문서가 그러하다. 게다가 코드 작성도 영어이며 특수문자가 덕지덕지 붙어있는 것은 덤이다. 함수는 보통 함수 기능을 설명하거나 대표하는 영단어의 일부나 첫 글자들을 따와서 짓는 경우가 많기에 영어를 잘하면 함수명을 보고 그 기능을 유추하기가 훨씬 용이하다.

백문이 불여일타(百聞而不如一打)

일단 쳐봐야 실력이 는다. 쉬운 내용은 눈으로 대충 훑어도 이해가 가니까 이를 간과하고 넘어가는 경우가 많다. 프로그래밍 언어를 배운다고는 하지만 크게 보면 언어이다. 그러니까 보고, 듣고, 말하고, 쓰는 연습이 필요하다. 코딩도 마찬가지로 단순히 눈으로 보는 것이 아니라 직접 입력하고 그 결과를 눈으로 확인해보고 느끼는 것이 매우 중요하다. 확실히 기초 내용이라도 차근차근 입력하던 사람들이 향후 고급내용을 다루어도 오탈자를 덜 내는 경향이 있다.

강력한 인간지능

언어를 새로 배울 때 공식문서를 독파하는 것은 매우 바람직하며 좋은 행동이다. 하지만 그 양이 방대하며 보통 영문으로 작성되어있어 부담되는 것이 사실이다. 그래서 새로운 함수가 있을 때 값을 하나씩 바꿔가면서 어떤 함수가 어떤 동작을 하는지 유추해 보는 것이 학습에 매우 도움이 된다. 책을 사서 살펴보면 온 세상 모르는 것 투성이다. 물론 책에서는 나름 설명을 해주기는 하겠지만 언제나 완벽하다고 할 수 없고, 유추를 해야 하는 상황이 오기도 한다. 마치 영어문장 독해를 할 때 모르는 단어를 주변 단어나 문장을 통해 뜻을 유추하듯 함수에 들어가는 숫자나 문자를 바꿔보고, 그 결과값의 변화를 관찰함으로써 보다 쉽게 함수를 이해할 수 있을 것이다.

그럼 앞에서 언급한 내용을 숙지하고 기초 서적으로 공부하면서 다음 글을 기다리는 것은 어떨까? 다음 호에는 Python에 대해 본격적으로 알아보고 각종 고급 기능 알아보는 시간을 가져보도록 하겠다.