**10장. 데이터 분석 및 시각화**

데이터가 너무 많아서 포기한다는 것인가? 어떻게 하면 데이터를 쉽게 분석하고 통찰력을 얻을 수 있는지 궁금해 한 적이 있는가? 데이터 분석 프로세스에 대해 궁금한가? 자, 제대로 찾아왔다.

10장은 다음을 살펴볼 것이다

-시각화를 사용한 데이터 읽기, 선택 및 해석

-데이터 필터링 및 집계를 사용한 통찰력 생성

-기업을 위한 소셜 미디어 분석 자동화

**소개**

우리는 하나님을 신뢰한다. 다른 모든 사람들은 데이터를 가져와야 한다.

- 에드워즈 데밍(Edwards Deming), 통계학자

오늘날 기업은 고객이 필요로 하는 정보, 구매할 채널 등을 파악하기 위해 데이터에 크게 의존한다. 이렇게 하면, 기업은 신제품을 출시하거나 새로운 제안을 내는 것에 대해 정보에 근거한 결정을 내릴 수 있다. 그러나 기업들은 어떻게 이것을 달성할까? 의사 결정은 실제로 무엇을 포함하는가?

데이터 기반 의사결정은 통찰력의 생성, 유용한 정보의 발견 및 결론 도출을 목적으로 데이터 위의 데이터 검사, 스크러빙(scrubbing) 혹은 정리(cleaning), 데이터 변환 및 모델을 생성하는 과정을 말한다. 예를 들어 전자상거래 회사는 이 프로세스를 사용하여 소비자 구매 패턴을 분석하고 특정 제품 그룹에 대한 프로모션 오퍼를 제안하기 위한 적절한 시간대를 제안한다. 실제로 기업은 추세 생성, 예측 모델 구축 혹은 원시 데이터에서 구조화된 정보 추출과 같은 여러 목적을 위해 정적 혹은 실시간 데이터를 분석한다. 데이터 분석에는 다양한 측면과 접근 방식이 있으며 비즈니스 인텔리전스, 예측 분석 및 텍스트 마이닝에 따라 간단하게 분류할 수 있다.

비즈니스 인텔리전스(BI, Business intelligence)는 대용량의 구조화된 데이터와 비정형 데이터를 처리할 수 있으므로 이러한 대량의 데이터를 쉽게 해석할 수 있다. 데이터에 대한 통찰력을 기반으로 새로운 기회를 파악하면 기업에서는 경쟁 우위와 안정성을 제공할 수 있다.

예측 분석(Predictive analytics)은 미래 혹은 알려지지 않은 사건에 대한 예측을 하기 위해 과거 데이터 및 현재 경향을 분석하기 위한 기계 학습과 같은 다양한 통계 모델의 적용을 포함한다. 여기에는 위험 평가 및 의사결정을 위해 모델을 생성하고 데이터 기능 간의 관계를 포착하는 작업이 포함된다.

텍스트 분석(Text analytics)은 구조화된 혹은 비정형화된 텍스트 데이터에서 품질 정보를 유도하는 프로세스이다. 텍스트 분석은 비즈니스를 위한 정보를 추출하고 분류하는 언어적, 통계적 및 문맥적 기법을 포함한다.

하지만 데이터 기반 의사결정은 쉽지 않으며 쉽게 수행할 수 없다. 데이터 기반 의사결정은 여러 작업을 포함하는 단계별 프로세스이다. 다음 절에서 전체 프로세스를 자세히 살펴보자.

**데이터 기반 의사결정 단계**

높은 수준에서 프로세스는 다음 단계로 분류할 수 있다. 물론, 목표에 맞게 프로세스를 커스터마이징할 수 있다.

-가설 및 데이터 요구사항 정의: 프로세스를 시작하기 전에 SMART(Specific, Measurable, Acceptable, Relevant and Timely)로 비즈니스 목표를 명확히 파악해야 한다. 문제를 해결하지 않고 데이터 수집을 시작하고 싶지 않다. 가능한 한 “지난 3분기 동안 소비자 공간에서 모바일 판매의 추세는 무엇인가?”와 같은 명확한 문제 진술 혹은 “이번 겨울에 150%의 이익으로 전자 제품을 판매할 수 있게 될 것인가?”와 같은 미래를 제시한다. 여러분의 회사에 이런 진술을 할 수 있는가?

-데이터 소스: 데이터 소스에 대해서도 분명히 밝혀야 한다. 데이터 분석을 수행하기 위해 회사 데이터베이스에 의존하고 있는가? 또한 분석을 기반으로 하는 서드 파티 시장 조사 혹은 동향에 의존하고 있는가? 서드 파티 데이터를 사용하는 경우, 소스(가능하면 API를 통해)에서 데이터를 추출하고 데이터 저장소에 저장하려면 어떻게 계획해야 하는가?

-데이터 수집: 이제 통찰력을 생성하려는 대상이 무엇인지 확실히 알았으니 다음 단계는 필요한 형식으로 데이터를 수집하는 것이다. 예를 들어, 모바일 판매 추세에 대한 데이터를 원하는 경우, 신제품 소개(제품), 오퍼(가격), 지불 옵션, 구매 날짜/시간, 다른 관련 요인 등 모바일 판매에 영향을 미치는 요소에 대한 데이터를 수집해야 한다. 또한 데이터를 저장하는 데 동의하거나 표준적인 방법을 사용해야 한다. 예를 들어, 단위당 모바일 매출을 EUR이 아닌 USD로 저장할 수 있거나 매출을 시간 단위가 아닌 일 단위로 저장할 수 있다. 이러한 경우에는 대표 표본(representative sample)을 확인하는 것이 매우 유용하다. 대표 표본은 전체 인구를 정확하게 반영하고 분석에 확실히 도움이 된다.

-데이터 변환: 이제 데이터를 어디에서 가져올 것인지, 어떤 형식으로 데이터를 수집해야 하는지 알게 되었으므로 데이터를 읽는 위치를 결정해야 한다. 일반적인 오래된 CSV일 수도 있고 SQL 데이터베이스 일 수도 있다. 가장 좋은 방법으로 데이터를 구성하고 분석할 수 있도록 사전에 알아야 한다. 이 단계는 원본 데이터 시스템에서 대상 데이터 시스템으로 데이터를 추출하기 때문에 변환이라고 할 수 있다. 큰 규모에서는 데이터웨어하우스 시스템에 저장된다.

-데이터 정리: 데이터를 처리하고 정리한 후에는 데이터의 이상 유무를 살펴봐야 한다 전송된 데이터는 호환되지 않거나, 중복되거나, 적어도 측정, 샘플링 및 데이터 입력 오류를 포함할 수 있다. 데이터 정리는 부정확한 데이터의 제거, 누락 데이터에 대한 기본 값 추가, 이상 징후 제거, 기타 데이터 불일치 해결이 포함된다. 이상 징후 제거는 주의해야 한다. 제거할 방법을 결정해야 한다. 단순히 레코드를 삭제하는 것인가? 아니면 다른 관찰의 평균/양식을 전가하는 것인가? 여러분은 이 경우 최고의 의사결정자이다.

-데이터 분석: 일단 데이터를 정리하고 사용할 준비가 되면 더 깊이 분석할 시간이다. 로지스틱 회귀분석(Logistic Regression)과 같은 통계 기법을 사용하여 비즈니스 인텔리전스를 위한 데이터를 분석하거나 예측 모델을 생성할 수 있다. 통찰력을 생성하고 의사결정에 이르기 위해 텍스트 분석도 수행할 수 있다.

-데이터 시각화: 일단 분석이 완료되면 분석을 효과적으로 청중에게 전달할 수 있도록 다양한 형식으로 보고할 수 있다. 데이터 시각화는 표와 차트와 같은 정보 표시를 사용하여 데이터에 포함된 주요 메시지를 전달하는 데 도움을 준다. 또한 시각화는 사용자가 분석을 통해 가정을 해석하고 의미있는 정보를 생성하는 데 도움이 된다.

-데이터 해석 및 피드백: 이 단계는 세 가지 주요 질문에 답하는 데 도움이 된다. 분석이 여러분이 시작한 질문에 대한 답변을 제공하는가? 가설을 수락하거나 거절하는 것을 증명하는 데 도움이 되는가? 모델 혹은 결론을 향상시키기 위해 더 많은 데이터가 필요한가? 결론이 시스템에 다시 전달되지 않으면 완료되지 않는다. 피드백 루프(feedback loop)는 예측 모델이 향후 사용을 위해 더욱 강화되고 잘 훈련되도록 한다.

그래, 좋은 출발이다! 통찰력을 얻기 위한 데이터 수집을 위해 완전한 프로세스에 대한 올바른 생각을 가지고 있어야 한다. 목표 정의, 데이터 수집 및 데이터 변환과 같은 몇 가지 단계는 시장 상황에 맞게 커스터마이징되며, 해결되고 있는 문제들이라는 점을 깨닫게 될 것이다.

10장에서는 실시간 데이터 수집, 데이터 읽기, 데이터 분석 수행 및 데이터 시각화와 같은 몇 가지 일반적인 측면을 살펴볼 것이다. 데이터를 효과적으로 읽고 통찰력을 생성하기 위해 데이터를 분석하는 데 도움이 되는 인기있는 파이썬 모듈을 다룰 것이다. 또한 데이터를 분석하고 시각화(차트)를 생성하는 데 도움이 되는 파이썬 모듈을 살펴볼 것이다.

10장의 마지막 부분에서는 10장에서 다룬 레시피를 학습한 지식으로 자동화할 수 있는 일반적인 비즈니스 프로세스에 대해서도 살펴볼 것이다. 10장은 데이터 과학자로서의 여정을 시작하는 데 도움이 되지만 통계 기법 혹은 예측 모델링과 같은 광범위한 주제는 다루지 않는다.

10장에서는 다음과 같은 파이썬 모듈을 사용할 것이다.

-pandas(http://pandas.pydata.org/pandas-docs/version/0.15.2/tutorials.html)

-numpy(http://www.numpy.org/)

-matplotlib(http://matplotlib.org/)

-seaborn(https://pypi.python.org/pypi/seaborn/)

**시각화를 사용한 데이터 읽기, 선택 및 해석**

이 레시피는 알려진 데이터셋의 도움을 받는다. 1,460개의 회사 펀딩 후원 목록이 포함된 TechCrunch의 Continental USA CSV 파일을 사용한다. 여기에는 회사 이름, 직원 수, 펀딩 날짜, 모금 금액 및 펀딩 유형(시리즈 A 혹은 엔젤 펀딩)과 같은 데이터 포인트가 포함된다.

<그림>

1. 이제 이 CSV 파일에서 데이터를 읽고 선택하는 데 사용할 모듈을 설치하자. 그렇게 하기 전에 가상 환경을 설정하고 활성화할 것이다.

chetans-MacBookPro:ch11 Chetan$ virtualenv analyze

New python executable in analyze/bin/python2.7

Also creating executable in analyze/bin/python

Installing setuptools, pip, wheel...done.

chetans-MacBookPro:ch11 Chetan$ source analyze/bin/activate

(analyze)chetans-MacBookPro:ch11 Chetan$

2. 그래, 좋아! 이제, pandas를 설치하자. CSV 파일을 읽고 분석할 데이터를 선택하기 위해 pandas를 사용할 것이다. 가장 좋아하는 유틸리티인 python-pip로 pandas를 설치한다. 다음은 Mac OSX의 pandas의 설치 로그이다.

(analyze)chetans-MacBookPro:ch11 Chetan$ pip install pandas

Collecting pandas

Collecting pytz>=2011k (from pandas)

Using cached pytz-2016.7-py2.py3-none-any.whl

Collecting python-dateutil (from pandas)

Using cached python\_dateutil-2.6.0-py2.py3-none-any.whl

Collecting numpy>=1.7.0 (from pandas)

Collecting six>=1.5 (from python-dateutil->pandas)

Using cached six-1.10.0-py2.py3-none-any.whl

Installing collected packages: pytz, six,

python-dateutil, numpy, pandas

Successfully installed numpy-1.11.2 pandas-0.19.1

python-dateutil-2.6.0 pytz-2016.7 six-1.10.0

<<중요표시시작>>

pandas 모듈을 설치하면 numpy 모듈도 설치된다. 사실, 이 모듈을 내 머신에 더 빨리 설치했다. 따라서 많은 모듈이 캐시에서 선택된다. 컴퓨터에서, 설치 로그는 다를 수 있다.

<<중요표시종료>>

3. 다음으로, matplotlib와 seaborn을 설치하자. 시각화를 위해 사용할 라이브러리이다. 다음은 matplotlib의 처음 머신의 설치 로그이다.

(analyze)chetans-MacBookPro:ch11 Chetan$ pip install matplotlib

Collecting matplotlib

Requirement already satisfied (use --upgrade to upgrade):

numpy>=1.6 in ./analyze/lib/python2.7/site-packages

(from matplotlib)

Requirement already satisfied (use --upgrade to upgrade):

pytz in ./analyze/lib/python2.7/site-packages

(from matplotlib)

Requirement already satisfied (use --upgrade to upgrade):

python-dateutil in ./analyze/lib/python2.7/site-packages

(from matplotlib)

Collecting cycler (from matplotlib)

Using cached cycler-0.10.0-py2.py3-none-any.whl

Collecting pyparsing!=2.0.0,!=2.0.4,!=2.1.2,>=1.5.6

(from matplotlib)

Using cached pyparsing-2.1.10-py2.py3-none-any.whl

Requirement already satisfied (use --upgrade to upgrade):

six>=1.5 in ./analyze/lib/python2.7/site-packages

(from python-dateutil->matplotlib)

Installing collected packages: cycler, pyparsing,

matplotlib

Successfully installed cycler-0.10.0

matplotlib-1.5.3 pyparsing-2.1.10

이러한 모듈이 내 머신에 설치되어 있으므로 이 모듈을 머신에 처음으로 설치할 때 설치 로그가 달라질 수 있다. 여기에 seaborn에 대한 로그가 있다.

(analyze)chetans-MacBookPro:ch11 Chetan$ pip install seaborn

Collecting seaborn

Collecting scipy (from seaborn)

Requirement already satisfied (use --upgrade to upgrade):

numpy>=1.7.1 in ./analyze/lib/python2.7/site-packages

(from scipy->seaborn)

Installing collected packages: scipy, seaborn

Successfully installed scipy-0.18.1 seaborn-0.7.1

**실행방법**

1. 먼저, https://support.spatialkey.com/spatialkey-sample-csv-data/에서 CSV 파일을 다운로드하자. TechCrunch 파일의 직접 다운로드 링크는 http://samplecsvs.s3.amazonaws.com/TechCrunchcontinentalUSA.csv이다. 다음과 같이 wget 커맨드를 사용하여 이 파일을 다운로드할 수 있다.

(analyze)chetans-MacBookPro:ch11 Chetan$

wget <http://samplecsvs.s3.amazonaws.com/>

TechCrunchcontinentalUSA.csv

--2016-11-20 16:01:57--

<http://samplecsvs.s3.amazonaws.com/>

TechCrunchcontinentalUSA.csv

Resolving samplecsvs.s3.amazonaws.com... 54.231.97.224

Connecting to samplecsvs.s3.amazonaws.com

|54.231.97.224|:80... connected.

HTTP request sent, awaiting response... 200 OK

Length: 93536 (91K) [application/x-csv]

Saving to: 'TechCrunchcontinentalUSA.csv'

TechCrunchcontinentalUSA.csv 100%

[=======================================================

=========================================>]

91.34K 20.3KB/s in 4.5s

2016-11-20 16:02:03 (20.3 KB/s) -

'TechCrunchcontinentalUSA.csv' saved [93536/93536]

2. 이제, CSV 파일을 읽기 위한 첫 번째 파이썬 코드를 작성하자. CSV 파일을 읽고 처음 다섯 행을 인쇄한다.

import pandas as pd

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

df = pd.read\_csv('TechCrunchcontinentalUSA.csv')

print "First five rows:\n", df[:5]

앞의 코드 예제에서는 CSV 파일의 처음 다섯 레코드를 읽는다.

<그림>

3. pandas 모듈은 파일의 콘텐츠를 읽고 행과 열의 데이터 프레임으로 변환한다. 이제, 앞의 코드 결과를 살펴보면, 인덱스 열이 파일 콘텐츠에 추가된다는 것을 알 수 있다. pandas로 날짜를 파싱하고, CSV 파일이 첫 번째 일 혹은 첫 번째 월(UK 혹은 US 형식)인지 여부를 확인하고, 날짜 열을 색인 열(index column)로 만들 수 있다.

import pandas as pd

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

df = pd.read\_csv('TechCrunchcontinentalUSA.csv', index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True,)

print "Top five rows:\n", df[:5]

앞의 코드를 실행하면 다음 스크린샷과 같이 색인 열인 fundedDate를 확인할 수 있다.

<그림>

4. 깔끔하다! 이제, 데이터를 읽을 수 있지만 그 위에 몇 가지 분석을 할 수 있도록 일부 데이터를 선택한다. 회사가 모금한 금액을 나타내는 열을 선택하자(raisedAmt 열).

import pandas as pd

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

df = pd.read\_csv('TechCrunchcontinentalUSA.csv', index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True,)

raised = df['raisedAmt'][:5]

print "Funding Raised by Companies over time:\n", raised

다음 스크린샷에는 모금 금액으로 회사의 상위 5개 레코드가 인쇄됐다.

<그림>

5. 그래, 좋아! 따라서 선택한 컬럼을 선택하고 분석을 위한 데이터를 얻었다. 멋진 시각화를 생성할 수 있는지 살펴보자. 다음 레시피는 모금 금액(y 축)을 기준으로 모든 연도(x 축)에 대해 보고된 펀딩 후원에 대한 선형 차트(line chart)를 생성한다.

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

df = pd.read\_csv('TechCrunchcontinentalUSA.csv')

print "First five rows:\n", df[:5]

df = pd.read\_csv('TechCrunchcontinentalUSA.csv', index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True,)

print "Top five rows:\n", df[:5]

raised = df['raisedAmt'][:5]

print "Funding Raised by Companies over time:\n", raised

sns.set\_style("darkgrid")

sns\_plot = df['raisedAmt'].plot()

plt.ylabel("Amount Raised in USD")

plt.xlabel("Funding Year")

plt.savefig('amountRaisedOverTime.pdf')

다음 스크린샷에서 펀딩 비율(혹은 보고 비율)이 어떻게 증가했는지 확인하고, 인상된 금액도 꾸준히 증가한 것도 확인한다.

<그림>

6. 환상적이다! 이미 여기서 하고 있는 것을 좋아하기 시작한 것을 안다. 더 나아가 CSV 파일에서 여러 열을 선택할 수 있는지 살펴보자. 다음 예제에서는 열 이름이 company, category 및 fundedDate인 50 개의 행에 대한 데이터를 얻는다.

import pandas as pd

from matplotlib import pyplot as plt

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

fundings = pd.read\_csv('TechcrunchcontinentalUSA.csv')

print "Type of funding:\n", fundings[:5]['round']

# 여러 열을 선택한다.

print "Selected company, category and date of funding:\n",\

fundings[['company', 'category', 'fundedDate']][600:650]

앞의 코드의 출력은 다음과 같다.

<그림>

7. 좋아! 이제 이 열 중에서 하나를 선택하고 그 위에 몇 가지 분석을 수행하자. 다음 코드 예제에서는 보고된 모든 펀딩 후원의 범주를 제공하는 category 열을 선택한다. 그런 다음 선택한 열을 처리하여 펀딩된 회사의 가장 일반적인 카테고리를 가져온다.

import pandas as pd

from matplotlib import pyplot as plt

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

fundings = pd.read\_csv('TechcrunchcontinentalUSA.csv')

print "Type of funding:\n", fundings[:5]['round']

# 여러 열을 선택한다.

print "Selected company, category and date of funding:\n",\

fundings[['company', 'category', 'fundedDate']][600:650]

# 펀딩된 회사의 가장 일반적인 카테고리

counts = fundings['category'].value\_counts()

print "Count of common categories of company that raised funds:\n", \

counts

앞의 코드의 출력은 다음과 같다.

Count of common categories of company that raised funds:

web 1208

software 102

mobile 48

hardware 39

other 16

cleantech 14

consulting 5

biotech 4

Name: category, dtype: int64

8. 데이터와 숫자는 많은 정보를 제공하지만, 실제로 시각화를 통해서만 영향을 확인할 수 있다. 위의 데이터를 수평 막대 차트로 그릴 수 있는지 살펴보자. 다음의 레시피는 우리를 위해 수행한다.

import pandas as pd

from matplotlib import pyplot as plt

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

fundings = pd.read\_csv('TechcrunchcontinentalUSA.csv')

print "Type of funding:\n", fundings[:5]['round']

# 여러 열을 선택한다.

print "Selected company, category and date of funding:\n",\

fundings[['company', 'category', 'fundedDate']][600:650]

# 펀딩된 회사의 가장 일반적인 카테고리

counts = fundings['category'].value\_counts()

print "Count of common categoris of company that raised funds:\n", \

counts

counts.plot(kind='barh')

plt.xlabel("Count of categories")

plt.savefig('categoriesFunded.pdf')

y축에는 펀딩된 회사의 카테고리가 있고 x축은 주어진 카테고리에서 회사의 총 수이다. 또한 도표를 categoriesFunded.pdf라는 PDF 파일로 저장한다.

<그림>

와! 이렇게 많은 웹 회사가 펀딩을 얻었는가? 굉장해! 자금을 얻을 확률이 높아지기 위해 나도 또한 웹 회사를 시작해야 한다.

**실행방법**

이 절에서는 데이터 분석의 두 가지 주요 측면을 다뤘다. 먼저 CSV 파일에서 데이터셋을 읽고 데이터셋에서 적절한 데이터(행 혹은 열)를 선택하는 방법을 살펴봤다.

첫 번째 코드에서는 pandas 모듈의 도움으로 CSV 파일을 읽었다. pandas에는 CSV 파일 경로를 인자로 사용하는 read\_csv(csv\_file) 메소드가 있다. pandas 모듈은 파일을 읽고 데이터 프레임으로 파일 콘텐츠를 저장한다. 데이터 프레임은 잠재적으로 다른 유형의 열을 가진 2차원이고 라벨이 있는 인메모리 데이터 구조이다. 스프레드시트의 구조 혹은 SQL 테이블 혹은 시리즈 객체 딕셔너리를 복사한다. 그것은 데이터를 선택, 색인 및 필터링을 위한 훌륭한 메소드 및 속성을 제공한다. 첫 번째 레시피는 CSV 파일을 읽고 데이터프레임 객체 df를 생성했다. 객체 df로 df[: 5]를 사용하여 CSV 파일의 처음 다섯 행을 선택했다. pandas로 CSV 파일의 행을 선택하는 것이 얼마나 쉬운 지 살펴봤다.

read\_csv() 메소드로 몇 가지 작업을 수행할 수 있다. 기본적으로 pandas는 데이터셋에 여러 색인 열을 추가하지만, CSV 파일의 어떤 열을 색인에 사용해야 하는지 지정할 수 있다. index\_col 매개변수를 read\_csv() 메소드에 전달하여 이 작업을 수행했다. 또한 CSV 파일의 fundedDate 열의 문자열 날짜를 parse\_dates 매개변수로 datetime 형식으로 변환했으며 date는 dayfirst 매개변수로 날짜의 첫 부분 day인 형식을 보여준다.

데이터프레임을 가져와서 fundedDate를 색인으로 사용한 후, raisedAmt 열을 선택하고 처음 다섯 개의 행을 인쇄하기 위해 df['raisedAmt'][:5]를 사용했다. 그런 다음 seaborn 라이브러리를 사용하여 sns.set\_style("darkgrid")로 스타일을 설정하고 plot() 메소드로 막대 차트를 생성했다. seaborn 라이브러리는 멋진 시각화를 생성하는 데 사용되며 matplotlib에서 구현된다.

matplotlib 라이브러리를 사용하여 ylabel() 및 xlabel() 메소드로 차트에 라벨을 지정하는 데 사용된 plt 객체를 생성했다. plt 객체는 또한 savefig() 메소드를 사용하여 결과 차트를 PDF 형식으로 최종 저장하는 데 사용됐다.

두 번째 예제에서는 fundings[['company','category' and 'fundedDate']]으로 여러 열을 선택했다. 한 줄의 코드로 CSV 파일에서 세 개의 열을 선택했다. 그런 다음 plot() 메소드로 가로 막대형 차트를 생성하고 kind=barh로 차트 유형을 지정했다. 마지막으로, xlabel() 메소드로 x축에 라벨을 지정하고, savefig() 메소드로 차트를 저장하기 위해 matplotlib 라이브러리를 사용했다. 보시다시피, 차트를 그리기 위해 seaborn 라이브러리를 사용할 필요가 없었다. 단순히 matplotlib로 함께 처리할 수 있다.

**데이터 필터링 및 집계를 사용한 통찰력 생성**

Pandas로 CSV 파일을 읽고 여러 열을 쉽게 선택할 수 있다. 이 절에서는 본래 pandas로 데이터 필터링, 데이터 분할하는 방법을 살펴본다.

**준비하기**

이 절에서는 이전 레시피에서 사용한 것과 동일한 라이브러리(다음 라이브러리)를 사용한다.

-필터링 및 데이터 분석을 위한 pandas

-차트 생성 및 PDF 파일의 데이터 저장을 위한 matplotlib 및 seaborn

**실행방법**

1. 필요한 라이브러리를 가져와서 read\_csv() 메소드로 CSV 파일을 읽는 것으로 시작하자. 다음 코드는 작업을 수행한다.

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

fundings = pd.read\_csv('TechcrunchcontinentalUSA.csv', index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True)

2. 이제 데이터프레임으로 필터링하고 여러 열을 사용하여 데이터를 필터링하자. category, state 및 city의 펀딩 레코드를 필터링한다고 가정한다. 다음 코드를 사용하면 이 작업을 수행할 수 있다.

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

funding = pd.read\_csv('TechcrunchcontinentalUSA.csv', index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True)

# CA의 웹 펀딩

web\_funding = funding['category'] == 'web'

in\_CA = funding['state'] == 'CA'

in\_city = funding['city'].isin(['Palo Alto', 'San Francisco',

'San Mateo', 'Los Angeles', 'Redwood City'])

위의 코드는 Palo Alto, San Francisco, San Mateo, Los Angeles, 및 Redwood City에 있는 웹 회사의 모든 펀딩 레코드를 반환한다. 다음은 출력의 스크린샷이다.

<그림>

3. 좋아. 이제 도시 이름별로 웹 카테고리의 회사에 대한 펀딩 수를 확인할 수 있는지 살펴보자. 다음 코드는 필요한 세부 정보를 제공한다.

web\_funding = funding[web\_funding & in\_CA & in\_city]

web\_counts = web\_funding['city'].value\_counts()

print "Funding rounds for companies in 'web' category by cities in CA:\n", web\_counts

앞의 코드 결과는 선택한 도시(city)의 웹(web) 카테고리의 회사가 받은 펀딩 후원 수이다.

<그림>

좋아! 분석은 매우 유용했다. 샌프란시스코(San Francisco) 도시의 웹 회사는 195회(보유한 데이터를 통해) 펀딩을 제공 받았음을 알게됐다. 샌프란시스코의 웹 회사라면, 모든 자금 문제는 끝난 것 같다. 그래, 그건 논리적이고 단순해 보인다.

4. 그러나 잠깐, 이 정보는 불완전하지 않는가? 어쩌면 웹(web)을 포함한 모든 카테고리의 회사에 대한 데이터를 수집한 다음 웹 카테고리의 회사에 대한 데이터를 모든 카테고리의 비율로 나타내는 것은 어떤가? 이런 식으로 하면, 'San Francisco' 혹은 다른 도시에 회사를 두어야 하는지를 알게 될 것이다. 좋아, 그러면, CA에서 선택한 모든 도시의 모든 카데고리(웹 포함)에 속하는 회사의 펀딩 후원의 수를 계산하자.

total\_funding = funding[in\_CA & in\_city]

total\_counts = total\_funding['city'].value\_counts()

print "Funding rounds for companies in 'all'

categories by cities in CA:\n",total\_counts

앞의 코드의 결과는 다음과 같다.

<그림>

5. 좋아! 이제 웹 카테고리의 회사에 대한 데이터를 CA의 선택된 도시에 대한 모든 카테고리의 회사 비율로 살펴보자. 웹 카테고리의 데이터를 모든 카테고리로 나누고 백분율로 100을 곱하면 이 데이터를 백분율로 나타낼 수 있다. 다음 코드는 이 경우에 도움이 될 것이다.

sns.set\_style("darkgrid")

sns\_plot = (web\_counts\*100/total\_counts.astype(

float)).plot(kind='barh')

6. 이제 이 데이터를 다음 코드로 가로 막대 차트로 표현하자.

plt.xlabel("(Funding Rounds in Web Category) / (

Funding Rounds in All Categories) \* (100)")

plt.savefig('webFundedByCity.pdf')

다음 스크린샷은 다른 모든 카테고리의 회사 및 캘리포니아의 도시에 대한 펀딩 후원과 관련하여 웹 회사의 펀딩 후원을 비교하는 데 도움이 된다.

<그림>

수치는 무엇을 보여주는가? 샌프란시스코에 아직도 회사를 설립하고 싶은가? 로스앤젤레스(Los Angeles)의 웹 회사인 경우 펀딩 후원이 제한되지만, 적어도 샌프란시스코(0.855)에 있을 때 보단 펀딩을 확보할 확률(0.925)이 더 높다.

이제 다른 예제를 살펴보자. 데이터를 분석하여 역사적으로 어느 달이 다른 달보다 더 많은 펀딩을 지원 받았는지 확인하고 싶다고 가정한다. 추가적으로, 시리즈 A 혹은 엔젤 펀딩과 같은 펀딩 후원과 관련시킬 수 있는가? 재미있는 생각이다! 하지만 거기에서 얻을 수 있는 방법이 있는가? pandas 모듈은 데이터와 데이터 집합을 그룹화하는 것을 지원하고 있으며, 이 분석을 위해 도움이 될 것이다. 이 문제를 단계적으로 살펴보자.

1. 먼저 CSV 파일을 읽고 raisedAmt 및 rounds라는 두 개의 열을 선택하자. 이 데이터프레임에 색인 열로 다른 열인 month도 추가하자. 다음 코드는 추가 분석을 위해 데이터 프레임을 준비한다.

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

plt.style.use('default')

pd.set\_option('display.line\_width', 5000)

pd.set\_option('display.max\_columns', 60)

df = pd.read\_csv('TechCrunchcontinentalUSA.csv',

index\_col='fundedDate', \

parse\_dates=['fundedDate'], dayfirst=True,)

funds = df[['raisedAmt', 'round']]

funds['month'] = funds.index.month

print "Funding Rounds with Month Index:\n", funds

2. 이제 그 달을 기준으로 모아진 펀딩에 대한 데이터를 얻을 필요가 있다. 다음 코드는 필요로 하는 것을 정확하게 수행한다.

funding\_by\_month = funds.groupby('month').aggregate('sum')

funding\_by\_month.index = ['Jan', 'Feb', 'Mar',

'Apr', 'May', 'June', 'July', \

'Aug', 'Sept', 'Oct', 'Nov', 'Dec']

print "Funding Rounds Grouped By Month:\n", funding\_by\_month

3. 이제 이 분석을 위한 데이터를 활용하면 보유한 모든 년도 데이터에 대한 월간 펀딩 변동 추이를 살펴볼 수 있다.

<그림>

좋아, 1월에 자금을 요청하는 것이 더 좋다. 아마도 투자자들은 크리스마스와 새해 휴가 이후에 좋은 분위기일 것이다. 어떻게 생각하는가?

4. 이제 월별, 발생한 금액 및 펀딩 후원에 상관 관계를 분석하고 작성하기 위해, 다음 코드로 데이터를 가져올 수 있다.

funds['month'] = funds.index.month

funding\_by\_stage = funds.groupby(['month',

'round']).aggregate('sum')

print funding\_by\_stage

앞의 코드의 출력은 다음 스크린샷과 같이 정렬된 데이터프레임이다. 데이터는 month 및 ground 열로 그룹화되며, raisedAmt는 이에 따라 집계된다.

<그림>

**작동원리**

첫 번째 문제의 경우 read\_csv() 메소드로 데이터프레임으로써 CSV 파일의 데이터를 로드한다. 이 후, 주(state)가 CA이고 회사 카테고리가 web이고 도시가 Palo Alto, San Francisco, SanMateo, Los Angeles 및 Redwood City인 것을 여러 요인을 기준으로 데이터를 필터링한다. 필터링은 열 기반 연산이며 꽤 간단하다. 기준을 적용한 후에 관련 데이터프레임을 가져온다.

그런 다음, value\_counts() 메소드로 웹 회사의 도시별로 그룹화된 펀딩 후원 수를 계산했다. 웹을 포함한 모든 카테고리의 회사에서 펀딩 후원을 위한 동일한 훈련을 했다.

마지막으로, 단순히 데이터를 나눠서 모든 카테고리의 데이터 백분율로 웹 회사의 데이터를 얻었다. pandas 모듈은 이 작업을 원활하게 처리했다. 신경쓰지 않고도 분석을 위해 같은 도시의 데이터 포인트를 모두 사용했다.

마지막으로 plot() 메소드로 가로 막대 차트를 생성하여 각 도시의 백분율을 개별적으로 나타내었고 찾고있는 통찰력을 얻었다.

두 번째 예제에서는 raisedAmt 및 round라는 여러 열을 선택하여 우선 데이터프레임을 얻었다. month 데이터프레임에 새 열도 추가했고 이를 색인 열로 처리했다.

그런 다음, groupby() 메소드를 사용하여 month를 기반으로 데이터를 그룹화했다. 이 후 month로 모금된 펀딩 금액을 얻어서, 펀딩 금액을 집계했다. 총 펀딩을 얻기 위해, aggregation() 메소드를 사용했고 필요한 정보를 얻기 위해 데이터 포인트를 추가했다.

또한 month 및 round로 모금된 펀딩 간의 상관 관계를 구축하기 위해 month 및 round로 데이터프레임을 그룹화하고 raisedAmt에 집계를 다시 적용했다.

**더 알아보기**

앞의 두 가지 레시피에서, 데이터 분석 및 시각화를 다뤘고, 광범위하게 레시피에 있는 pandas 파이썬 모듈을 사용했다. pandas 모듈은 매우 포괄적인 라이브러리이며 시계열(time series) 작업, 고급 색인 기술, 객체 병합 및 연결, 다른 데이터셋(JSON 및 Excel) 작업 등의 기능을 제공한다. 이 멋진 라이브러리에 대해 자세히 알아 보려면 pandas API 레퍼런스를 살펴보기를 추천한다.

다음 레시피에서는 주디(Judy)가 자신의 작업을 자동화할 수 있도록 도와줌으로써 지금까지 10장에서 얻은 지식을 적용할 수 있는지 살펴보자.

**기업의 소셜 미디어 분석 자동화**

주디는 런던의 한 유명 잡지의 칼럼니스트이다. 작가로서, 주디는 항상 최근 주제에 관심이 있다. 주디는 데이터를 수집하고 분석하여 독자들에게 흥미로운 통찰력을 제시한다.

현재, 주디는 애플(Apple)의 아이폰(iPhone)과 삼성(Samsung)의 노트(Note) 간의 경쟁에 관심이 있고, 잡지에 기사를 게시하려고 한다. 주디는 거리에서 사람들과 이야기하고 블로그 게시물을 읽음으로써 데이터를 수집할 계획이지만, 주디는 소셜 미디어로부터 엄청난 양의 정보를 얻을 것이라는 것을 알고있다. 주디는 사람들이 요즘 트위터에 제품을 사용하는 것에 대한 즐거움이나 실망을 표현하고, 소셜 미디어에 있는 친구들에게도 제품을 소개한다는 사실을 안다. 그러나 주디는 자신의 기사를 위해 엄청난 양의 소셜 미디어 데이터를 검토해야 한다는 사실에 대해 우려한다.

여러분은 데이터 과학자이자 주디의 동료이다. 주디에게 도움을 줄 수 있을까? 데이터 기술을 과시할 수 있는 기회이다.

주디의 문제를 분석하기 시작하자. 우선 주디는 트위터(Twitter)와 같은 지속적으로 성장하는 소셜 미디어 플랫폼에서 데이터를 수집해야 한다. 둘째, 흥미로운 통찰력을 얻기 위해 이 데이터를 분석해야한다. 따라서 주디의 문제를 모두 해결할 수 있는 시스템을 구축할 수 있어야 한다. 또한 자신의 현재 요구 사항만 해결하는 시스템을 구축하는 것이 좋지만 향후 모든 프로젝트에 이 시스템을 사용할 수 있어야한다.

**준비하기**

이 문제를 해결하는 데 필요한 모든 모듈을 설치하자. 이미 pandas, matplotlib 및 seaborn를 설치했다. 이 문제를 해결하기 위해 트위터 데이터를 처리할 모듈인 tweepy도 설치한다. 자신의 python-pip로 tweepy를 설치하자.

(analyze)chetans-MacBookPro:ch11 Chetan$ pip install tweepy

You are using pip version 7.1.0, however version 9.0.1 is available.

You should consider upgrading via the 'pip install --upgrade pip'

command.

Collecting tweepy

Downloading tweepy-3.5.0-py2.py3-none-any.whl

Collecting requests>=2.4.3 (from tweepy)

Downloading requests-2.12.1-py2.py3-none-any.whl (574kB)

100% |████████████████████████████████| 577kB

161kB/s

Requirement already satisfied (use --upgrade to upgrade): six>=1.7.3 in

./analyze/lib/python2.7/site-packages (from tweepy)

Collecting requests-oauthlib>=0.4.1 (from tweepy)

Downloading requests\_oauthlib-0.7.0-py2.py3-none-any.whl

Collecting oauthlib>=0.6.2 (from requests-oauthlib>=0.4.1->tweepy)

Downloading oauthlib-2.0.0.tar.gz (122kB)

100% |████████████████████████████████| 122kB

345kB/s

Building wheels for collected packages: oauthlib

Running setup.py bdist\_wheel for oauthlib

Stored in directory:

/Users/chetan/Library/Caches/pip/wheels/e4/e1/92/68af4b20ac26182fbd623647af

92118fc4cdbdb2c613030a67

Successfully built oauthlib

Installing collected packages: requests, oauthlib, requests-oauthlib,

tweepy

Successfully installed oauthlib-2.0.0 requests-2.12.1 requestsoauthlib-

0.7.0 tweepy-3.5.0

**실행방법**

좋아, 이제 모든 모듈의 준비가 끝났다. 이제 트위터(Twitter)에서 데이터를 수집하여 시작하자. 트위터에는 개발자가 실시간으로 트윗을 수집하는 데 유용한 놀라운 스트리밍 API가 있다. 데이터 수집 요청에도 이 라이브러리를 사용할 것이다.

1. 다음 코드는 트위터 스트리밍(Twitter Streaming) API를 사용하여 데이터를 수집하고, 각 트윗(tweet)을 JSON형식으로 저장하는 텍스트 파일에 저장한다. 두 개의 키워드인 iPhone 7과 Note 5를 찾는다. 10장에서는 약 5분 동안 코드를 실행했지만, 주디를 위해 정확한 통찰력을 얻기 위해 최대 데이터를 수집하는 몇시간 혹은 며칠 동안 코드를 실행해야 할 수도 있다.

from tweepy import Stream

from tweepy import OAuthHandler

from tweepy.streaming import StreamListener

import json

#consumer key, consumer secret, access token, access secret.

ckey="<>"

csecret="<>"

atoken="<>"

asecret="<>"

tweets\_data\_path = 'twitter\_data.txt'

f = open(tweets\_data\_path, "w")

class listener(StreamListener):

def on\_data(self, data):

print data

f.write(data)

#all\_data = json.loads(data)

#tweet = all\_data["text"]

#lang = all\_data["lang"]

#username = all\_data["user"]["screen\_name"]

#print "username:%s, tweet:%s,

language:%s" %(username, tweet, lang)

return True

def on\_error(self, status):

print "Error:", status

auth = OAuthHandler(ckey, csecret)

auth.set\_access\_token(atoken, asecret)

twitterStream = Stream(auth, listener())

twitterStream.filter(track=["iPhone 7","Note 5"])

f.close()

자, 이제 트위터에서 온 데이터가 들어왔으니, 이 데이터를 분석하고 주디가 그녀의 기사에 대해 공유할 수 있는 흥미로운 것을 찾을 수 있는지 알아보기 위해 코드를 작성하자.

2. 애플 아이폰(iPhone) 및 삼성 노트(Note)는 전 세계의 사람들이 이야기하는 세계적 수준의 인기 제품이다. 소비자들이 트위터에서 이런 제품들에 대해 이야기할 때 사용하는 다양한 언어들을 발견하는 것은 정말 흥미로운 일일 것이다. 다음 코드는 트위터 데이터로 하기를 원하는 것과 정확히 일치한다. 그것은 저장된 트윗을 통해 모든 트윗의 언어를 살펴보고 그것을 그룹화하여 상위 4개 언어로 생성한다.

import json

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

tweets = []

fh = open("twitter\_data.txt", "r")

for data in fh:

try:

tweets.append(json.loads(data))

except:

continue

tweet\_df = pd.DataFrame()

tweet\_df['lang'] = map(lambda x: x['lang'], tweets)

tweets\_by\_lang = tweet\_df['lang'].value\_counts()

fig, axis = plt.subplots()

sns.set\_style("darkgrid")

axis.set\_xlabel('Languages', fontsize=15)

axis.set\_ylabel('Tweets' , fontsize=15)

clrs = ['green', 'blue', 'red', 'black']

sns\_plot = tweets\_by\_lang[:4].plot(ax=axis, kind='bar', color=clrs)

plt.savefig('language.pdf')

앞의 코드를 실행하면, 사람들이 아이폰 7과 노트5에 대해 트윗을 할 때 사용했던 상위 언어에 대한 막대 차트를 나타낸다.

<그림>

굉장하다! 나는 주디가 이 분석을 좋아할 것이라고 생각한다. 상위 언어가 예상대로 영어(en)이지만 다른 세 언어는 이탈리아어(it), 스페인어(es) 및 포르투갈어(pt)이다. 이것은 기사에 대한 멋진 피드일 것이다.

이 연습에서 얻을 수 있는 결과는 데이터 수집 프로그램을 실행할 때 결정된다. 예를 들어, GMT 시간이 오전 2시부터 8시까지 짧은 기간 동안 실행하면 이러한 나라에서는 낮시간이기 때문에 중국어 혹은 일본어로 더 많은 트윗을 살펴볼 수 있다. 그런데, 사람들이 트윗을 하는 가장 좋은 시간이 언제인지 분석하는 것은 흥미로운 일이 아닌가? 몇 가지 상관 관계를 발견할 수 있다.

3. 이 데이터를 가지고 더 자세히 살펴보고 더 멋진 것들을 해보자. 이런 제품에 대한 소비자의 감정을 파악하기 위해 트윗에 대한 텍스트 기반 분석을 실시하는 것은 어떨까? 감정에 의해, 이런 제품들을 흠잡는 트윗인지, 아니면 제품의 특징을 높이 평가하는지, 아니면 그냥 지나가는 코멘트인지 파악할 수 있을까? 하지만 기다린다. 이런 종류의 데이터를 얻을 수 있을까? 그래, 절대적이다. 다음 코드는 파이썬의 NTLK 기반 API를 사용하여 긍정, 부정 혹은 중립적인 감정을 결정하기 위해 트윗(텍스트)에 대한 감정 분석을 수행한다. 그런 다음 이 데이터를 그룹화하여 막대 차트로 표시하고 PDF 파일로 저장한다.

import json

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import requests

tweets = []

fh = open("twitter\_data.txt", "r")

for data in fh:

try:

tweets.append(json.loads(data))

except:

continue

probablities = pd.DataFrame()

prob = []

for tweet in tweets:

text = tweet['text']

r = requests.post(url="http://text-processing.com/api/sentiment/", data={"text":text},)

print r.text

if r.status\_code == 200:

ans = json.loads(r.text)

prob.append(ans["label"])

probablities['data'] = map(lambda x: x, prob)

p\_df = probablities['data'].value\_counts()

fig, axis = plt.subplots()

sns.set\_style("darkgrid")

axis.set\_xlabel('Sentiments', fontsize=15)

axis.set\_ylabel('Tweets' , fontsize=15)

clrs = ['green', 'yellow', 'red']

sns\_plot = p\_df.plot(ax=axis, kind='bar', color=clrs)

plt.savefig('sentiments.pdf')

앞의 코드를 실행하면 저장된 트윗 모두에 대한 감정 데이터로 막대 차트를 얻을 것이다. 그래프를 살펴보면, 소비자가 일반적으로 두 제품에 대해 긍정적으로 말한 것을 알 수 있다.

<그림>

그것들에 대한 약간의 불평이 있기 때문에, 부정적인 극성을 가진 감정이 있지만, 단지 제품에 대한 언급이나 코멘트가 될 수 있는 중립적인 코멘트가 있다. 산뜻하다.

이것이 좋긴하지만, 여러분은 주디가 이러한 부정적인 트윗 중 몇개가 아이폰 혹은 노트에 관한 것인지 알고 싶어한다고 생각하지 않을까? 글쎄, 여러분에게 이것을 남겨둘 것이다. 여러분이 주디에게 그것을 알아낼 것이라고 확신한다.

**작동원리**

첫 번째 문제의 경우, 먼저 분석에 필요한 데이터를 수집했다. 트위터의 스트리밍 API는 실시간으로 이 정보를 수집할 수 있다. 스트리밍 API를 사용하려면 트위터에 개발자 앱에 등록하고 소비자 키(consumer key), 소비자 시크릿(consumer secret), 인증 토큰(auth token) 및 인증 시크릿(auth secret)을 수집해야한다. 그것은 간단한 과정이며 https://dev.twitter.com에서 쉽게 찾을 수 있다. 따라서, 데이터 수집 예제(이 레서피의 첫 번째 예제)에서 OAuthHandler 클래스를 인스턴스화하여 트위터의 권한 객체인 auth를 얻은 다음 set\_access\_token() 메소드를 사용하여 권한 토큰(authorization token)과 시크릿(secret)을 설정하는 데 사용했다. 트위터의 스트리밍 API의 Stream 클래스는 listener 클래스에 바인딩되어 twitterStream 객체를 반환한다. listener 클래스는 StreamListener 클래스를 상속하고 이 클래스는 들어오는 트윗을 모니터링하고 도착하는 트윗에 대해 on\_data() 메소드에서 작업을 수행한다. 트윗의 필터링은 twitterStream.filter() 메소드를 사용하여 수행된다.

이제, 들어오는 트윗이 on\_data() 메소드에서 사용가능하다는 것을 안다. 트윗을 twitter\_data.txt 파일에 저장하기 위해 그것에 사용했다. 이것을 위해 write(w) 모드에서 파일을 열고 write() 메소드를 사용하여 JSON 형식의 파일에 트윗을 작성했다. 이를 통해 첫 번째 레시피를 완성하고 주디가 요구하는 데이터를 수집했다. 이제 분석을 수행할 차례이다.

언어를 가져오는 것에 대한 첫 번째 통찰력을 얻기위해, twitter\_data.txt 파일을 read(r) 모드로 여는 것으로 시작했다. 모든 트윗(JSON 형식)을 읽고 tweets 배열에 추가했다. pandas를 사용하여 pd.DataFrame()로 빈 DataFrame 객체인 tweet\_df를 생성했다. 파이썬의 map() 메소드를 사용하여, tweets 배열에서 작업하고 빈 DataFrame에 새로운 열 lang을 추가했다. 그런 다음 value\_counts() 메소드를 사용하여 분석중인 트윗의 모든 언어 수를 가져와서 변수 tweets\_by\_lang에 저장했다.

코드의 다른 부분은 평소대로 matplotlib에서 plt 객체를 생성하고 seaborn 라이브러리를 사용하여 막대 차트를 생성하기 위해 plot() 메소드를 사용했다. set\_xlabel() 및 set\_ylabel() 메소드로 축 라벨을 설정하고 green, blue, red 및 black 색상을 사용하여 다른 언어를 나타낸다. 마지막으로 savefig() 메소드를 사용하여 PDF 파일에 저장했다.

감정 분석을 포함하는 두 번째 통찰력을 위해, twitter\_data.txt의 모든 트윗을 읽고 tweets 배열에 저장하는 것으로 시작했다. 이 후 빈 데이터프레임 probabilities을 생성하고, 모든 감정 분석의 트윗을 처리하고, prob 배열에 분석을 저장했다. 그런 다음 prob 배열의 map() 메소드를 사용하여 빈 데이터프레임에 text 열을 추가했다.

데이터프레임 probabilities['text']은, 이제 분석한 모든 트윗에 대한 감정이 포함된다. 정규 작업에 따라, 분석된 트윗에 대한 positive, negative 및 neutral의 값 집합을 얻고, 막대 차트로 그린다.

이 레시피의 모든 예제를 살펴보면, 데이터 수집 및 분석 작업을 별도의 프로그램으로 나눴다. 시각화가 방대하면 그것들을 분리할 수도 있다. 이렇게 하면, 주디가 데이터 수집 파이썬 프로그램을 사용하여, 미래의 기사를 위해, 다른 키워드 집합에 대한 정보를 수집할 수 있다.

또한 프로그램의 분석 및 시각화 매개변수를 약간 변경하여 자신의 데이터 집합에 대한 분석을 실행할 수도 있다. 따라서 미래의 모든 기사에 대해, 주디를 위해 데이터 수집, 분석 및 시각화 프로세스를 자동화할 수 있었다.

주디의 얼굴은 이미 미소를 짓고있다. 여러분이 10장을 즐겼기를 바란다. 10장에서 얻은 지식을 통해 데이터 및 시각화 세계를 시작할 수 있다고 확신한다.

**더 알아보기**

이전 레시피에서 텍스트 기반 감정 분석의 겉모습을 살펴봤다. 감정 분석은 텍스트 분류, 토큰화, 의미론적 추론 및 훨씬 더 흥미로운 내용이 포함된다. <http://www.nltk.org/book/>에서 파이썬 텍스트로 작업하는 방법을 자세히 살펴보기 위해 NLTK에 대한 책을 보는 것을 강력히 추천한다.