1 데이터와 표본분포

빅데이터 시대가 되면서 데이터의 질과 적합성을 일정 수준 이상으로 담보할 수 없기 때문에 표본추출의 필요성이 더 커지고 있다.

1.1 임의표본추출과 표본편향

표본은 데이터의 부분집합.
큰 데이터 집합을 모집합
임의표본추출 : 대상이 되는 모집단 내의 선택 가능한 원소들을 무작위로 추출하는 과정.
단순임의표본 : 위 과정의 결과물
복원 추출, 비복원 추출
데이터의 품질이란 완결성, 형식의 일관성, 깨끗함 및 각 데이터의 정확성.
층화표본추출 : 모집단을 층으로 나눈 뒤, 각 층에서 무작위로 표본을 추출하는 것.
표본 편향의 사례
알프레드 랜던이 프랭클린 루스벨트를 제치고 대선에 승리하리라 예언했던 리터러리 다이제스트의 설문 조사.
조사자 선정에 문제가 있어 더 많은 사람을 대상으로 여론 조사했음에도 불구하고 승리를 예측하지 못함.

1.2 편향

통계적 편향은 측정 과정 혹은 표본추출 과정에서 발생하는 계통적인 오차를 의미.

1.3 임의 선택

임의표본추출
층화표본추출

1.4 크기와 품질 : 크기는 언제 중요해지는 가

빅데이터에 가치가 있을 것이다 라는 일반적인 예상은 데이터가 크고 동시에 희소할 때이다. 예제 구글의 검색 쿼리

1.5 표본평균과 모평균

$\bar{x}$는 모집단의 표본평균. μ는 모집단의 평균// 왜 구분해야하나? 표본에 대한 정보는 관찰을 통해 얻어지고 모집단에 대한 정보는 주로 작은 표본들로부터 추론하기 때문.

2 선택편향

데이터를 의식적이든 무의시적이든 선택적으로 고르는 관행.
데이터 스누핑이란 뭔가 흥미로운 것을 찾아 광범위하게 데이터를 살피는 것.
방대한 검색효과란 큰 데이터 집합을 가지고 반복적으로 다른 모델을 만들고 다른 질문을 하다보면, 언젠가 흥미로운 것을 발견하기 마련일때.
성능을 검증하기 위해 둘 이상의 홀드아웃 세트를 이용
엘더는 데이터마이닝 모델에서 제시하는 예측을 검증하기 위해, 목푯값(순열검정)라는 것을 추천.

2.1 평균으로의 회귀

평균으로의 회귀란 주어진 어떤 변수를 연속적으로 측정했을 때 나타나는 현상.

2.2 통계학에서의 표본분포

표본분포라는 용어는 하나의 동일한 모집단에서 얻은 여러 샘플에 대한 표본통계량의 분포를 나타냄. 주요 관심사는 표본의 변동성.
파이썬 코드에서는 히스토그램을 표시하기 위해 seaborn의 FAcetGrid를 사용

2.3 중심극한정리

모집단이 정규분포가 아니더라도 표본의 크기가 충분하고 데이터가 정규성을 크게 이탈하지 않는 경우, 여러 표본에서 추출한 평균은 종모양의 정규곡선을 따른다.

2.4 표본오차

표본오차는 통계에 대한 표본분포의 변동성을 한마디로 말해주는 단일 측정 지표.
표준오차 = Standard Error = SE = $\frac{s}{\sqrt{n}}$
표본오차를 추정하기 위해 새 샘플을 수집하는 접근방식은 일반적으로 불가능.
현대 통계에서는 부트스트랩은 표준오차를 추청하는 표준방법이 됨

3 부트스트랩

현재 있는 표본에서 추가적으로 표본을 복원추출하고 각 표본에 대한 통계량과 모델을 다시 계산하는 것.
데이터나 표본통계량이 정규분포를 따르지 않아도 됨.
복원추출
부트스트랩 재표본추출 알고리즘

1. 샘플 갑을 하나 뽑아서 기록하고 다시 제자리(복원추출)

2. n번 반복
3. 재표본추출된 값의 평균을 기록
4. 1 3단계 R번 반복
5. R개의 결과를 사용하여 표준편차를 계산하고 히스토그램 또는 박스플롯을 그리고 신뢰구간을 찾는다.

파이썬에서는 scikit learn의 resample 메서드를 사용.
배깅 : 분류 및 회귀 트리를 사용할 때 여러 부트스트랩 샘플을 가지고 트리를 여러개 만든 다음 각 트리에서 나온 예측값을 평균내는 것(분류 문제에서는 과반수 투표 보팅)
크기 n 에 따라 표본분포가 어떻게 달라지는지 알보기 위한 실험을 통해 표본크리를 결정하는데에 부트스트랩을 사용할 수 있다.

bootstrap

July 18, 2022

[1]:
```
import pandas as pd

df = pd.read_csv('data2.csv')
df.head()
```

[1]:

0	16465	2018	9800000
1	42082	2015	6500000
2	62265	2016	26700000
3	38233	2019	25800000
4	56155	2017	24200000

[12]:
```
from sklearn.utils import resample
```

[17]:
```
resample(df
        , replace=False
        , n_samples=60
        , random_state=42
        , stratify=None
        )
```

[17]:

418	43045	2016	8800000
474	48205	2017	21000000
181	32582	2019	48900000
446	60277	2015	17700000
297	19865	2021	45200000
148	104379	2011	5700000
380	68351	2016	18700000
386	50308	2018	10800000
479	92211	2012	9500000
227	47921	2019	32800000
9	10347	2021	11800000
175	98132	2017	14300000
478	55881	2019	35300000
323	106236	2016	9500000
211	107535	2014	10500000
427	15760	2018	8500000

84	61099	2018	25100000
30	77342	2016	16700000
238	85118	2016	14300000
90	12353	2015	9200000
247	85178	2018	25500000
477	17785	2021	48700000
281	4221	2020	28900000
463	50477	2017	12400000
33	45139	2017	7100000
153	28543	2019	60800000
475	51543	2019	14700000
469	36958	2018	9300000
77	117018	2014	9200000
70	118908	2018	32400000
73	55554	2019	23400000
461	31190	2020	33400000
438	110255	2018	20600000
78	86685	2018	18900000
0	16465	2018	9800000
11	12970	2019	49800000
104	60457	2017	37400000
117	74802	2016	9900000
360	21795	2018	14500000
390	16725	2021	53800000
131	17241	2020	15500000
222	45212	2018	24200000
55	36706	2017	9200000
462	57829	2016	13300000
18	25449	2021	46200000
63	78525	2019	16500000
68	106816	2015	13500000
124	80754	2015	13000000
456	82866	2013	5300000
39	37193	2020	38900000
430	84280	2013	7800000
398	30854	2020	21900000
185	103620	2019	28900000
250	37465	2020	27100000
383	38605	2020	17800000
431	36845	2020	13400000
209	2034	2022	17800000
261	40168	2017	10800000
101	13070	2020	18700000
341	88602	2016	20900000