핸즈온머신러닝
1,2장

2018-11-08

김성헌

목차

1. 한눈에 보는 머신러닝

1.1 머신러닝이란

- 데이터로부터 학습하도록 컴퓨터를 프로그래밍하는 과학(또는 예술)
- 명시적인 프로그래밍 없이 컴퓨터가 학습하는 능력을 갖추게 하는 연구 분야 - 아서사무엘, 1959
- 어떤 작업 T에 대한 컴퓨터 프로그램의 성능을 P로 측정했을 때 경험 E로 인해 성능이 향상됐다면, 이 컴퓨터 프로그램은 작업 T와 성능 측정 P에 대해 경험 E로 학습한 것이다. - 톰 미첼, 1997

1.2 왜 머신러닝을 사용하는가?

- 기존 솔루션으로는 많은 수동 조정과 규칙이 필요한 문제 : 하나의 머신러닝 모델이 코드를 간단하고 더 잘 수행되도록 할 수 있습니다.
- 전통적인 방식으로는 전혀 해결 방법이 없는 복잡한 문제 : 가장 뛰어난 머신러닝 기법으로 해결 방법을 찾을 수 있습니다.
- 유동적인 환경 : 머신러닝 시스템은 새로운 데이터에 적응할 수 있습니다.
- 복잡한 문제와 대량의 데이터에서 통찰 얻기

1.3 머신러닝 시스템의 종류

- 지도, 비지도학습
  - 지도학습
  - 비지도학습
    - clustering: k-means, HCA, Expectation Maximization
    - 시각화, 차원축소: PCA,...
      - 시각화: 대규모 고차원데이터를 도식화 가능한 2D,3D로 표현
    - 이상치탐지: 부정신용카드 거래 감지, 학습데이터셋에서 이상한 값 자동제거
    - 연관규칙학습: 대량의 데이터에서 특성 간의 흥미로운 관계 탐색. Apriori, Eclat
  - 준지도학습(예:구글 포토 사람분류)
  - 강화학습
- 배치학습, 온라인학습: 점진적 학습가능 여부
  - 배치학습(오프라인학습): 점진적으록 학습 불가
  - 온라인학습: 미니배치 단위로 훈련. 변화하는 데이터에 빠르게 적응
- 사례기반, 모델기반 학습: 어떻게 일반화 되는가에 따른 분류
  - 사례기반 예: 스팸메일 판단할 때 스팸단어많으면 스팸으로 분류
  - 모델기반: 일반적인 학습방식

1.4 머신러닝의 주요 도전 과제

- 훈련데이터 부족
- 대표성부족
  - 한쪽으로 치우친 데이터
- 저품질 데이터
  - 에러, 이상치, 잡음이 많은 데이터는 내재된 패턴을 찾기가 어렵다.
  - 대부분의 데이터과학자는 데이터정제에 많은 시간을 쓰고 있다.
- 관련없는 특성
- overfitting
  - 데이터에 있는 잡음의 양에 비해 모델이 너무 복잡할 때 발생
  - 해결: 모델단순화, 데이터의 특성수 줄이기, 더많은 데이터 확보, 데이터잡음제거
- underfitting
  - 너무 단순한 모델 사용이 원인

1.5 테스트와 검증

- 학습세트
  - 학습용 데이터
- 검증세트
  - 하이퍼파라미터 튜닝용
- 테스트세트
  - 최종 테스트 용

1.6 연습문제

1. 머신러닝을 어떻게 정의할 수 있나요?
2. 머신러닝이 도움을 줄 수 있는 문제 유형 네 가지를 말해보세요.
3. 레이블된 훈련 세트란 무엇인가요?
4. 가장 널리 사용되는 지도 학습 작업 두 가지는 무엇인가요?
5. 보편적인 비지도 학습 작업 네 가지는 무엇인가요?
6. 사전 정보가 없는 여러 지형에서 로봇을 걸어가게 하려면 어떤 종류의 머신러닝 알고리즘을 사용할 수 있나요?
7. 고객을 여러 그룹으로 분할하려면 어떤 알고리즘을 사용해야 하나요?
8. 스팸 감지의 문제는 지도 학습과 비지도 학습 중 어떤 문제로 볼 수 있나요?
9. 온라인 학습 시스템이 무엇인가요?
10. 외부 메모리 학습이 무엇인가요?
11. 예측을 하기 위해 유사도 측정에 의존하는 학습 알고리즘은 무엇인가요?
12. 모델 파라미터와 학습 알고리즘의 하이퍼파라미터 사이에는 어떤 차이가 있나요?
13. 모델 기반 알고리즘이 찾는 것은 무엇인가요? 성공을 위해 이 알고리즘이 사용하는 가장 일반적인 전략은 무엇인가요? 예측은 어떻게 만드나요?
14. 머신러닝의 주요 도전 과제는 무엇인가요?
15. 모델이 훈련 데이터에서의 성능은 좋지만 새로운 샘플에서의 일반화 성능이 나쁘다면 어떤 문제가 있는 건가요? 가능한 해결책 세 가지는 무엇인가요?
16. 테스트 세트가 무엇이고 왜 사용해야 하나요?
17. 검증 세트의 목적은 무엇인가요?
18. 테스트 세트를 사용해 하이퍼파라미터를튜닝하면 어떤 문제가 생기나요?
19. 교차 검증이 무엇이고, 왜 하나의 검증 세트보다 선호하나요?

2. 머신러닝 프로젝트 처음부터 끝까지

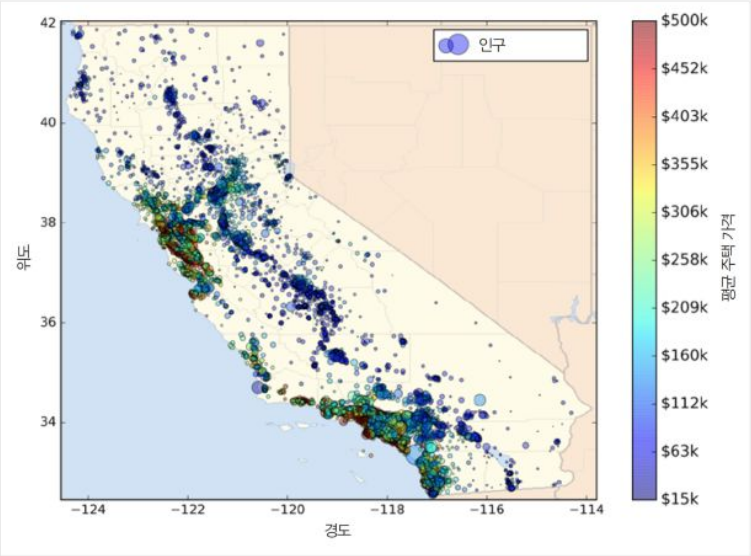
주요단계

1. 큰 그림을 봅니다.
2. 데이터를 구합니다.
3. 데이터로부터 통찰을 얻기 위해 탐색하고 시각화합니다.
4. 머신러닝 알고리즘을 위해 데이터를 준비합니다.
5. 모델을 선택하고 훈련시킵니다.
6. 모델을 상세하게 조정합니다.
7. 솔루션을 제시합니다.
8. 시스템을 론칭하고 모니터링하고 유지 보수합니다.

2.1 실제 데이터로 작업하기

- StatLib 저장소[2]에 있는 캘리포니아 주택 가격California Housing Prices 데이터셋
  - 1990년 캘리포니아 인구조사 데이터를 기반

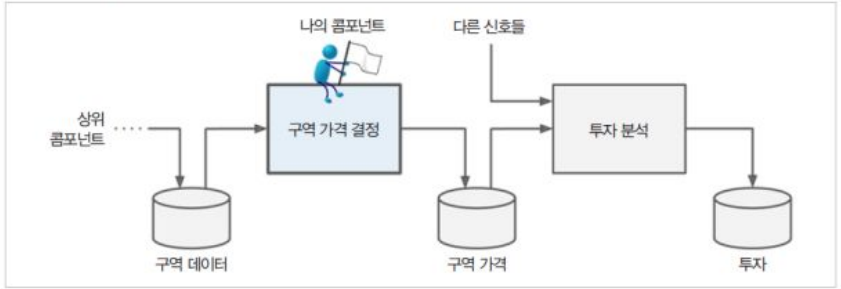
그림 2-1 캘리포니아 주택 가격



2.2 큰 그림 보기

- 목표: 특정 구역의 중간 주택 가격 예측
  - 학습데이터: 캘리포니아 구역마다 인구, 중간소득, 중간 주택 가격 등.
- 문제정의
  - 비즈니스의 목적이 무엇인가요?
  - 현재 솔루션은 어떻게 구성되어 있나요?
    - 전문가가 수동 추정. 정확도가 낮음
  - 문제정의 (지도/비지도, 온라인/오프라인)
    - 지도, 오프라인

그림 2-2 부동산 투자를 위한 머신러닝 파이프라인



- 성능측정지표 선택
  - $\mathrm{RMSE}(\boldsymbol{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2}$ $\mathrm{MAE}(\boldsymbol{X}, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h(\boldsymbol{x}^{(i)}) - y^{(i)}\right|$
- 가정검사
  - 전체적으로 시뮬레이션

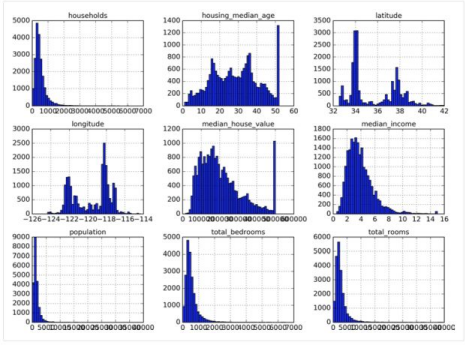
2.3 데이터 가져오기

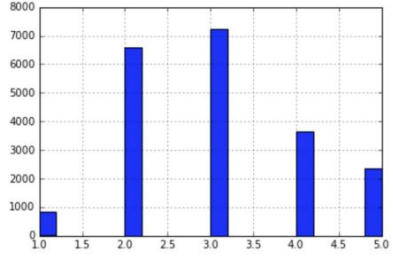
- 작업환경 만들기
- 데이터 다운로드
- 데이터 구조 훑어 보기
  - NULL 유무 파악
  - 데이터형태 검토
- 테스트 세트 만들기
  - 데이터 스누핑 편향: 테스트 세트를 학습에 반영해서 론칭 후 기대성능이 나오지 않는 것
  - 랜덤하게 테스트 세트 분리
    - 처음 한번 테스트세트 저장 후 사용
    - 난수 발생기 초기값 고정후 매번 실행
    - 샘플의 식별자(행의 인덱스 id)로 테스트 세트를 분리하자.
  - 샘플링 편향 방지 -> 계층적 샘플링
    - 전체모수는 계층이라는 동질의 그룹으로 나뉘고, 테스트 세트가 전체 모수를 대표하도록 각 계층에서 올바른 수의 샘플을

```
In [8]: housing.describe()
```

Out[8]:

	longitude	latitude	housing_median_age	total_rooms	total_bedr
count	20640.000000	20640.000000	20640.000000	20640.000000	20433.000(
mean	-119.569704	35.631861	28.639486	2635.763081	537.870553
std	2.003532	2.135952	12.585558	2181.615252	421.38507(
min	-124.350000	32.540000	1.000000	2.000000	1.000000
25%	-121.800000	33.930000	18.000000	1447.750000	296.00000(
50%	-118.490000	34.260000	29.000000	2127.000000	435.00000(
75%	-118.010000	37.710000	37.000000	3148.000000	647.00000(
max	-114.310000	41.950000	52.000000	39320.000000	6445.00000(

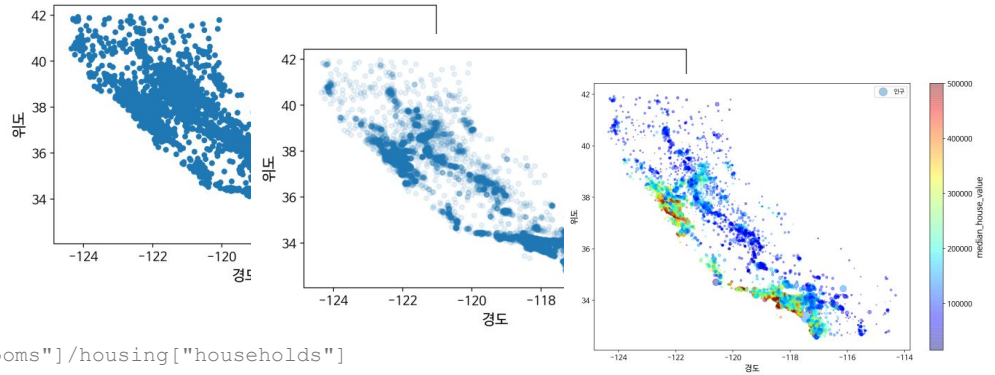




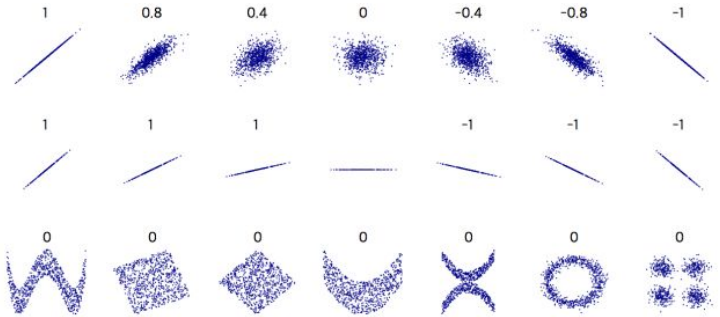
	전체	무작위 샘플링	계층 샘플링	무작위 샘플링 오류율	계층 샘플링 오류율
1.0	0.039826	0.040213	0.039738	0.973236	-0.219137
2.0	0.318847	0.324370	0.318876	1.732260	0.009032
3.0	0.350581	0.358527	0.350618	2.266446	0.010408
4.0	0.176308	0.167393	0.176399	-5.056334	0.051717
5.0	0.114438	0.109496	0.114369	-4.318374	-0.060464

2.4 데이터 이해를 위한 탐색과 시각화

- 지리적 데이터 시각화
- 상관관계 조사
  - 상관관계
    - -1 ~ 1
    - 1에 가까우면 강한 양의 상관관계
    - -1에 가까우면 강한 음의 상관관계
    - 0에 가까우면 상관관계 없음
- 특성 조합으로 실험
  - `housing["rooms_per_household"] = housing["total_rooms"]/housing["households"]`
  - `housing["bedrooms_per_room"] = housing["total_bedrooms"]/housing["total_rooms"]`
  - `housing["population_per_household"]=housing["population"]/housing["households"]`




```
>>> corr_matrix = housing.corr()
>>> corr_matrix["median_house_value"].sort
median_house_value 1.000000
median_income 0.687160
rooms_per_household 0.146285
total_rooms 0.135097
housing_median_age 0.114110
households 0.064506
total_bedrooms 0.047689
population_per_household -0.021985
population -0.026920
longitude -0.047432
latitude -0.142724
bedrooms_per_room -0.259984
Name: median_house_value, dtype: float64
```



2.5 머신러닝 알고리즘을 위한 데이터 준비

- 데이터 정제
  - 값이 없는 경우
    - 해당구역제거, 전체특성 삭제, 특정값을 채움(0, 평균, 중간값)
- 텍스트와 범주형 특성 다루기
  - One-hot 인코딩 사용
  - 희소행렬(sparse matrix): 0이아닌 원소의 위치만 저장. 카테고리가 많을 때 효율적임.
  - Tip: 카테고리수가 많을 때 ont-hot은 학습을 느리게 하고 성능감소 가능성이 있음. 이런 경우 임베딩이라고 하는 조금 더 조밀한 표현을 사용
- 나만의 변환기
- 특성 스케일링
  - min-max 스케일링, 표준화(standardization)
- 변환 파이프라인
  -

```
num_pipeline = Pipeline([
        ('selector', DataFrameSelector(num_attribs)),
        ('imputer', Imputer(strategy="median")),
        ('attribs_adder', CombinedAttributesAdder()),
        ('std_scaler', StandardScaler()),
    ])

cat_pipeline = Pipeline([
        ('selector', DataFrameSelector(cat_attribs)),
        ('cat_encoder', CategoricalEncoder(encoding="
    ])
```

```
full_pipeline = FeatureUnion(transformer_list=[
        ("num_pipeline", num_pipeline),
        ("cat_pipeline", cat_pipeline),
    ])
```

2.6 모델 선택과 훈련

- 훈련 세트에서 훈련하고 평가하기
- 교차 검증을 사용한 평가
  - k-fold cross-validation
- 다양한 모델에 대한 평가를 수행하고 최적 모델 선택
  - linear-regression
  - decision tree regressor
  - random forest regressor

2.7 모델 세부 튜닝

- 그리드 탐색
  - 수동으로 하이퍼파라미터를 조정
- 랜덤탐색
  - 탐색공간이 커지면 그리드탐색 보다 좋다.
- 앙상블 방법
  - 최상의 모델을 연결해 보는 것
- 최상의 모델과 오차 분석
  - random forest의 각 특성별 중요도 분석 -> 덜 중요한 특성을 제외 -> 더 정확한 모델
- 테스트 세트로 시스템 평가하기
  - 테스트세트로 최종 평가
  - 테스트세트로는 하이퍼파라미터를 튜닝하지 마라.

2.8 론칭, 모니터링, 그리고 시스템 유지 보수

- 입력 데이터소스를 시스템에 연결하고 테스트 코드 작성
- 모니터링 코드 작성
  - 새로운 데이터를 사용해서 주기적으로 훈련시켜야 함
- 시스템 성능 평가
  - 전문분석가의 분석이 필요
- 시스템의 입력데이터 품질 평가
  - 저품질 데이터의 입력은 시스템 성능 모니터링에서 알람이 울릴 정도까지 다소 시간이 걸린다
  - 시스템의 입력을 모니터링하면 이보다 일찍 알 수 있다. 아주 중요
- 새로운 데이터를 사용해 정기적으로 모델 훈련
  - 가능하면 자동화 해야 함.