# 다섯째마당
# 딥러닝 활용하기

# 22장 캐글로 시작하는 새로운 도전

1 캐글 가입 및 대회 선택하기

2 데이터 획득하기

3 학습하기

4 결과 제출하기

5 최종 예측 값 제출하기

# 캐글로 시작하는 새로운 도전

- 캐글로 시작하는 새로운 도전

# 캐글로 시작하는 새로운 도전

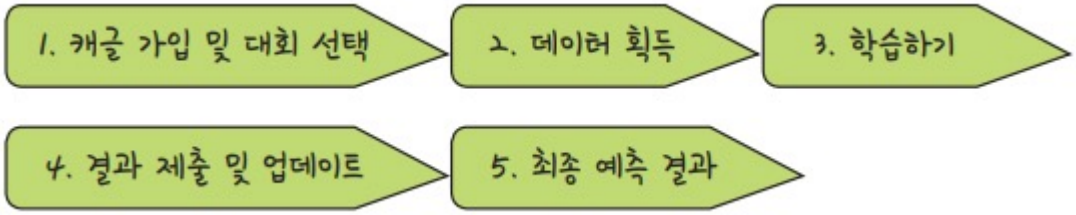

- **캐글로 시작하는 새로운 도전**
  - 캐글은 2010년 4월부터 지금까지 전 세계 데이터 과학자 15만 명 이상이 참가해 온 데이터 분석 경진대회
  - '데이터 사이언스를 스포츠처럼!'이라는 구호 아래 데이터 분석 기술을 스포츠와 같이 경쟁할 수 있게 만든 것이 특징
  - 대회는 상금과 함께 상시 열리고 있으며, 각 경쟁마다 풀어야 할 과제와 평가 지표 그리고 실제 데이터가 주어짐
  - 주어진 데이터를 사용해 정해진 시간 안에 가장 높은 정확도로 예측하는 것이 목표
  - 분석 결과를 업로드하면 보통 몇 분 안에 채점이 끝나며, 평가 지표에 근거해 참가자 간의 순위가 매겨짐
  - 실제 데이터를 사용해 다양한 기술을 구현하므로 자신의 데이터 과학 수준을 확인할 수 있을 뿐만 아니라 최신 기술과 트렌드를 배울 수 있는 기회를 제공

# 캐글로 시작하는 새로운 도전


- **캐글로 시작하는 새로운 도전**
  - 이 장에서 우리는 캐글에 가입하는 방법과 캐글에 예측된 결과를 업로드하는 방법을 배울 것
  - 앞서 '15장. 실제 데이터로 만들어 보는 모델'에서 사용한 데이터가 실은 캐글에서 배포하는 유명 벤치마크 학습셋
  - 이제 우리가 실행한 학습의 결과를 캐글에 업로드하고 평가 지표에 따른 순위를 확인해 보자

# 캐글로 시작하는 새로운 도전

- **캐글로 시작하는 새로운 도전**
  - 캐글에 참여하는 순서는 다음과 같음



# 1 캐글 가입 및 대회 선택하기

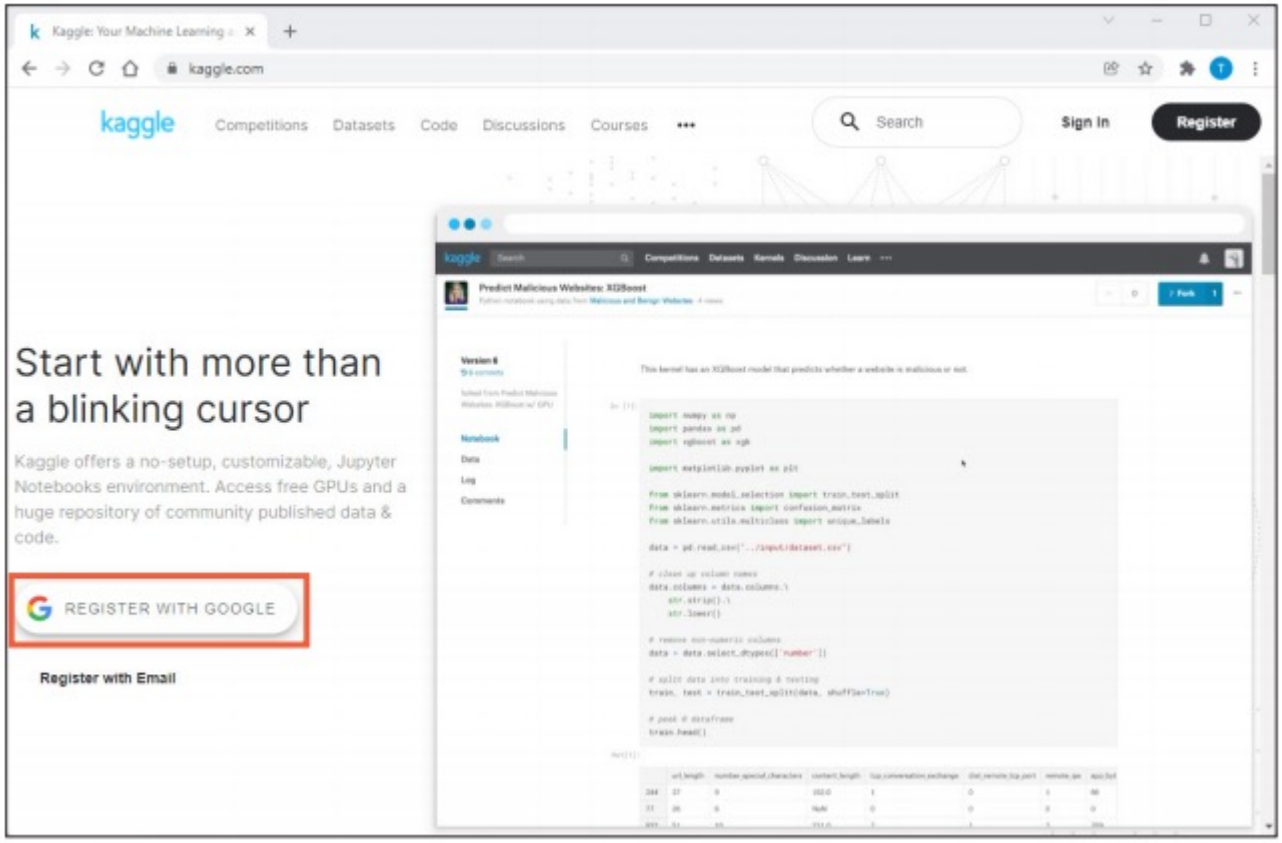
# 1 캐글 가입 및 대회 선택하기



- **캐글 가입 및 대회 선택하기**
  - 먼저 캐글 웹 사이트에 방문해 회원 가입을 함
  - 구글 계정이 있으면 간단히 회원에 가입할 수 있음

# 1 캐글 가입 및 대회 선택하기

▼ 그림 22-1 | 캐글 웹 사이트



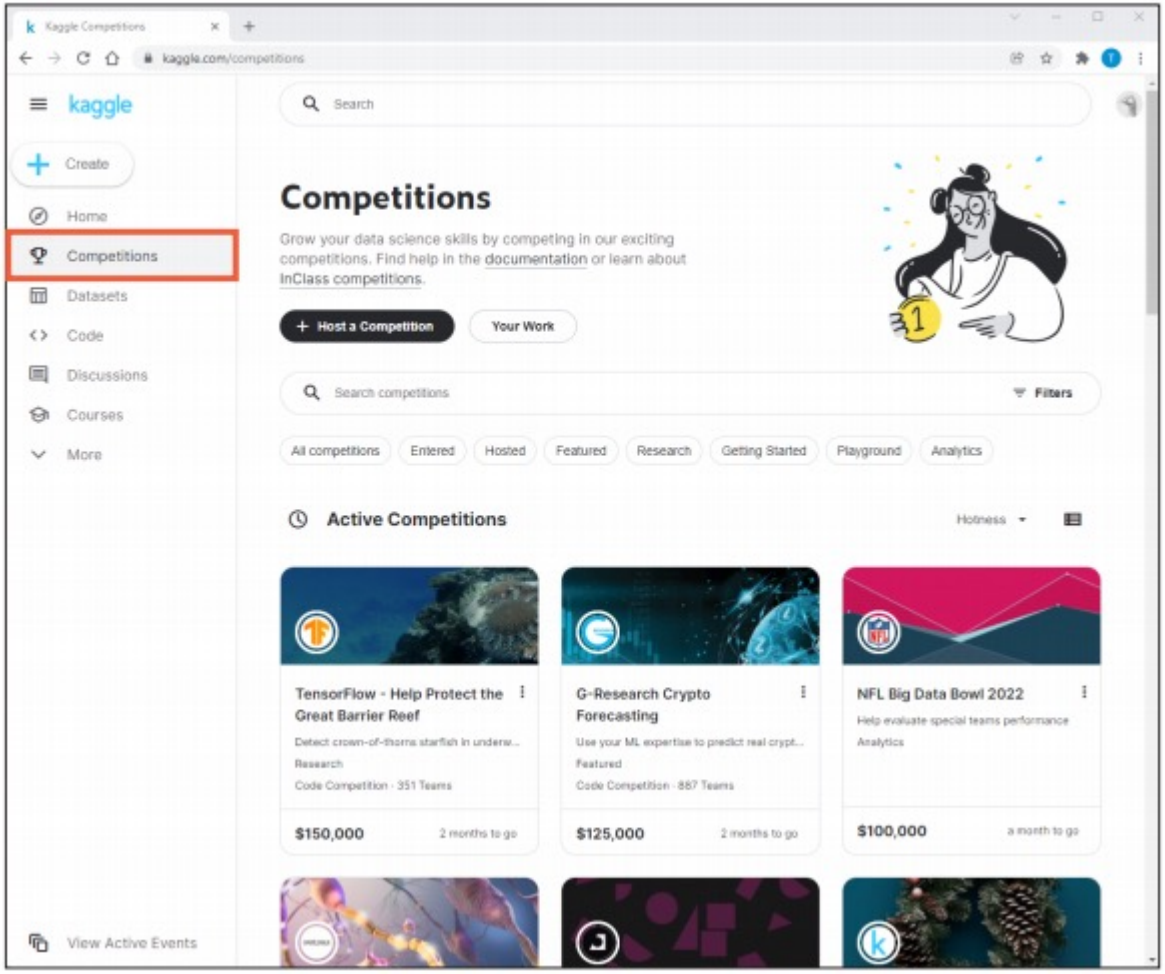
# 1 캐글 가입 및 대회 선택하기



- **캐글 가입 및 대회 선택하기**
  - 가입이 완료되면, 캐글에 공지된 대회 중 참가할 만한 대회를 선택
  - 메인 화면에서 Competitions를 클릭하면 현재 진행 중인 경진대회의 목록이 보임

# 1 캐글 가입 및 대회 선택하기

▼ 그림 22-2 | 경진대회 목록 확인하기

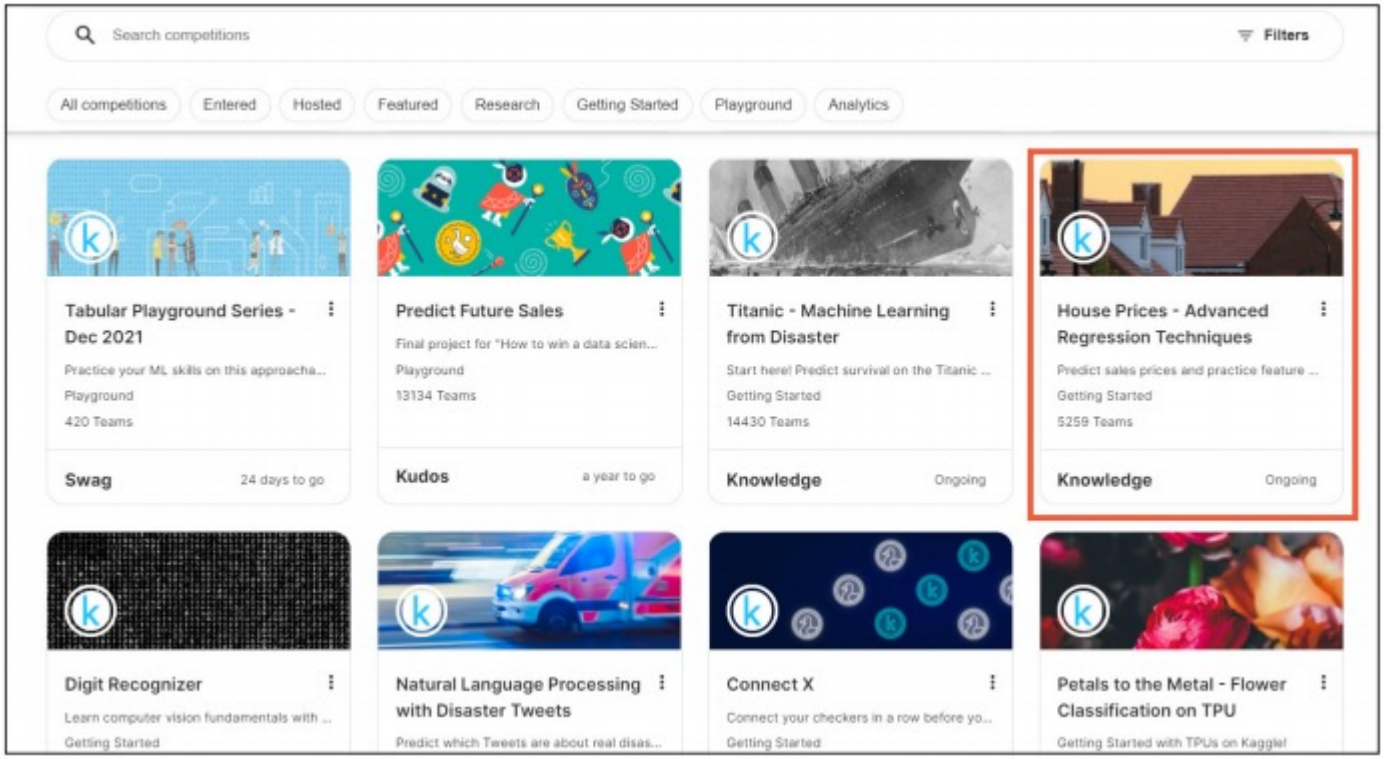


# 1 캐글 가입 및 대회 선택하기


- **캐글 가입 및 대회 선택하기**
  - 우리는 스터디를 목적으로 하므로 캐글에서 누구나 테스트할 수 있게끔 준비한 HousePrices - Advanced Regression Techniques를 클릭
  - 해당 대회로 바로 이동하는 주소는 https://www.kaggle.com/c/house-prices-advanced-regression-techniques

# 1 캐글 가입 및 대회 선택하기

▼ 그림 22-3 | House Prices 폴더로 이동

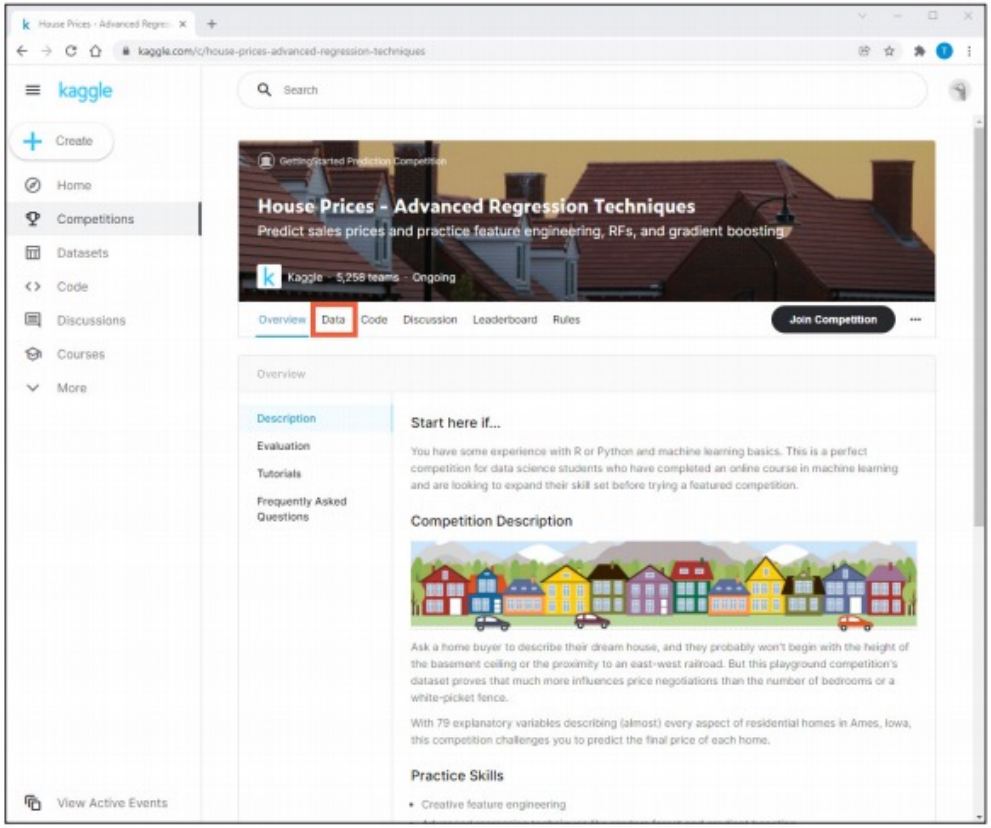


# 2 데이터 획득하기

# 2 데이터 획득하기

- **데이터 획득하기**
  - 해당 경진대회에 접속을 완료하면 대회에 대한 내용을 숙지하고 **Data**를 클릭해 데이터에 접근

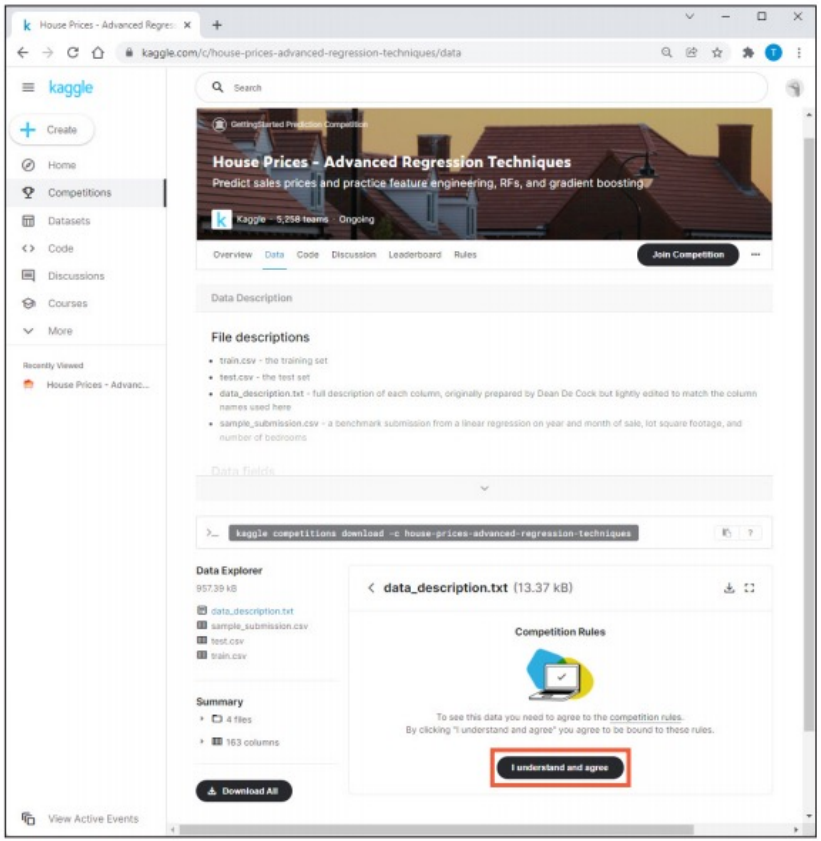
▼ **그림 22-4 | 데이터에 접근하기**



# 2 데이터 획득하기

- **데이터 획득하기**
  - 데이터 화면이 나오면 **I understand and agree**를 클릭해 데이터를 내려받을 준비

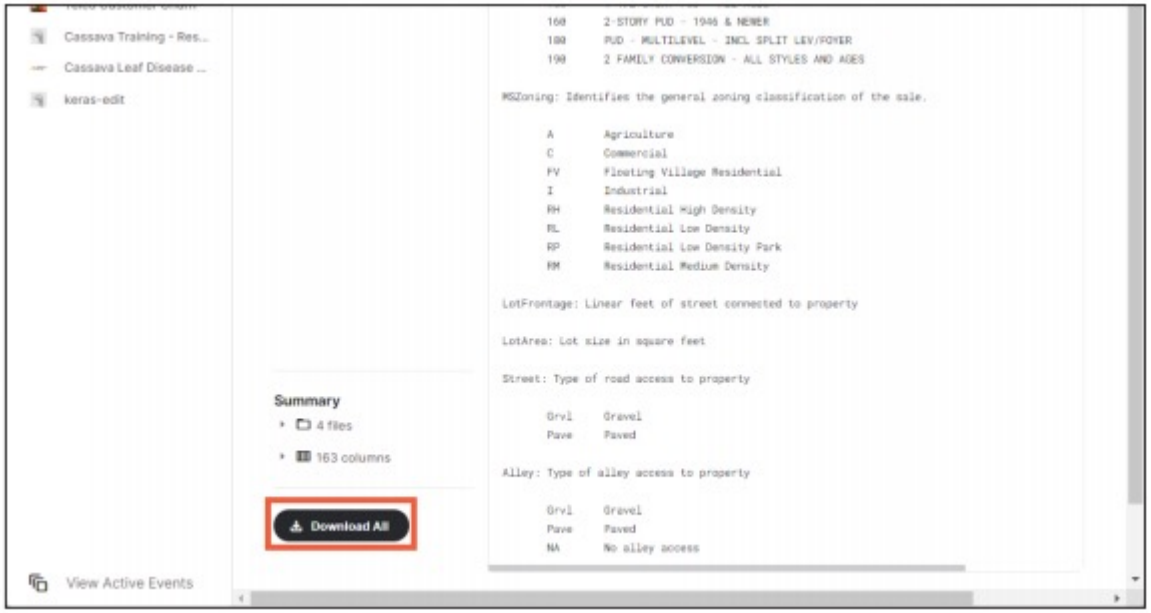
▼ **그림 22-5 | 데이터를 내려받을 준비**



# 2 데이터 획득하기

- **데이터 획득하기**
  - 데이터 화면이 바뀌고 해당 데이터에 대한 설명이 나옴
  - **Download All**을 클릭

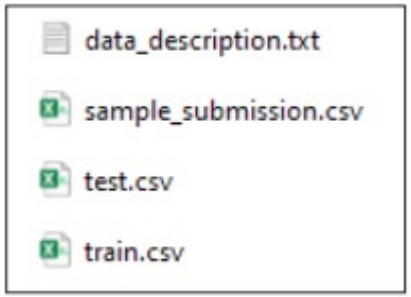
▼ **그림 22-6 | 데이터 내려받기**



# 2 데이터 획득하기

- **데이터 획득하기**
  - 내려받은 데이터를 확인해 보자

▼ **그림 22-7 | 데이터 확인하기**



# 2 데이터 획득하기


- **데이터 획득하기**
  - data_description.txt 파일은 내려받은 데이터의 각 속성이 무엇을 의미하는지 설명하고 있음
  - train.csv 파일은 집 값과 해당 집이 어떤 속성을 가졌는지 정리된 파일
  - test.csv 파일은 이 train.csv 파일을 이용해 학습한 결과를 테스트하기 위한 데이터
  - train.csv 파일과 모든 항목이 같지만 맨 마지막 집 값(SalePrice) 항목만 빠져 있음
  - 이 항목을 예측하는 것이 우리의 과제
  - sample_submission.csv 파일은 Id와 SalePrice 두 개의 열만 존재하는 파일
  - 각 Id별로 우리가 예측한 SalePrice를 채워 넣어 캐글에 업로드하면 됨

# 3 학습하기

# 3 학습하기


- **학습하기**
  - 데이터를 확인했으면 이제 딥러닝 또는 머신 러닝 기법을 활용해 모델을 만들고 학습을 시작하면 됨
  - 여기서는 15장에서 실시한 학습 모델을 가져와 어떻게 테스트셋에 적용하는지 공부해 보자

# 3 학습하기


- **학습하기**
  - 먼저 필요한 라이브러리를 불러옴
  - 케라스의 load_model과 판다스를 불러오자

```
from tensorflow.keras.models import load_model
import pandas as pd
```

# 3 학습하기


- **학습하기**
  - 캐글에서 배포하는 house_test.csv 파일은 data 폴더에 이미 저장되어 있음
  - 해당 테스트셋을 불러오자

```
kaggle_test = pd.read_csv("./data/house_test.csv")
```

# 3 학습하기

▼ 그림 22-8 | kaggle_test 파일 내용 미리 보기

| | Id | MSSubClass | MSZoning | LotFrontage | LotArea | Street | Alley | LotShape | LandContour | Utilities |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1461 | 20 | RH | 80.0 | 11622 | Pave | NaN | Reg | Lvl | AllPub |
| 1 | 1462 | 20 | RL | 81.0 | 14267 | Pave | NaN | IR1 | Lvl | AllPub |
| 2 | 1463 | 60 | RL | 74.0 | 13830 | Pave | NaN | IR1 | Lvl | AllPub |
| 3 | 1464 | 60 | RL | 78.0 | 9978 | Pave | NaN | IR1 | Lvl | AllPub |
| 4 | 1465 | 120 | RL | 43.0 | 5005 | Pave | NaN | IR1 | HLS | AllPub |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1454 | 2915 | 160 | RM | 21.0 | 1936 | Pave | NaN | Reg | Lvl | AllPub |
| 1455 | 2916 | 160 | RM | 21.0 | 1894 | Pave | NaN | Reg | Lvl | AllPub |
| 1456 | 2917 | 20 | RL | 160.0 | 20000 | Pave | NaN | Reg | Lvl | AllPub |
| 1457 | 2918 | 85 | RL | 62.0 | 10441 | Pave | NaN | Reg | Lvl | AllPub |

# 3 학습하기


- **학습하기**
  - 테스트셋의 속성은 학습셋과 동일한 상태로 변형되어야 해당 모델을 적용할 수 있음
  - 이를 위해 학습셋과 동일하게 전처리되어야 함
  - 먼저 카테고리형 변수를 0과 1로 이루어진 변수로 바꾸어 주겠음

```
kaggle_test = pd.get_dummies(kaggle_test)
```

# 3 학습하기


- **학습하기**
  - 결측치를 전체 칼럼의 평균으로 대체해 채워 줌

```
kaggle_test = kaggle_test.fillna(kaggle_test.mean())
```

# 3 학습하기

- 학습하기
  - 업데이트된 데이터 프레임을 출력해 보면 그림 22-9와 같음

▼ 그림 22-9 | 수정된 데이터 프레임 보기

| | Id | MSSubClass | LotFrontage | LotArea | OverallQual | OverallCond | YearBuilt | YearRemodAdd |
|---|---|---|---|---|---|---|---|---|
| 0 | 1461 | 20 | 80.0 | 11622 | 5 | 6 | 1961 | 1961 |
| 1 | 1462 | 20 | 81.0 | 14267 | 6 | 6 | 1958 | 1958 |
| 2 | 1463 | 60 | 74.0 | 13830 | 5 | 5 | 1997 | 1998 |
| 3 | 1464 | 60 | 78.0 | 9978 | 6 | 6 | 1998 | 1998 |
| 4 | 1465 | 120 | 43.0 | 5005 | 8 | 5 | 1992 | 1992 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1454 | 2915 | 160 | 21.0 | 1936 | 4 | 7 | 1970 | 1970 |
| 1455 | 2916 | 160 | 21.0 | 1894 | 4 | 5 | 1970 | 1970 |
| 1456 | 2917 | 20 | 160.0 | 20000 | 5 | 7 | 1960 | 1996 |
| 1457 | 2918 | 85 | 62.0 | 10441 | 5 | 5 | 1992 | 1992 |
| 1458 | 2919 | 60 | 74.0 | 9627 | 7 | 5 | 1993 | 1994 |

# 3 학습하기


- **학습하기**
  - 이제 학습에 사용된 열을 K_test로 저장

```
cols_kaggle = ['OverallQual','GrLivArea','GarageCars','GarageArea',
'TotalBsmtSF']
K_test = kaggle_test[cols_kaggle]
```

# 3 학습하기



- **학습하기**
  - 앞서 15장에서 만든 모델을 불러옴

```
model = load_model("./data/model/Ch15-house.hdf5")
```

# 3 학습하기


- **학습하기**
  - model.predict()를 이용해 불러온 모델에 조금 전 만든 K_test를 적용하고 예측 값을 만들어 봄

```
ids = [] # ID와 예측 값이 들어갈 빈 리스트를 만듭니다.
Y_prediction = model.predict(K_test).flatten()
for i in range(len(K_test)):
    id = kaggle_test['Id'][i]
    prediction = Y_prediction[i]
    ids.append([id, prediction])
```

# 3 학습하기


- **학습하기**
  - 테스트 결과의 저장 환경을 설정
  - 앞서 만든 내용과 중복되지 않도록 현재 시간을 이용해 파일명을 만들어 저장
  - 파일은 별도 폴더에 저장

```
import time

timestr = time.strftime("%Y%m%d-%H%M%S")
filename = str(timestr) # 파일명을 연월일-시분초로 정합니다.
outdir = './'           # 파일이 저장될 위치를 지정합니다.
```

# 3 학습하기


- **학습하기**
  - 앞서 만들어진 실행 번호와 예측 값을 새로운 데이터 프레임에 넣고 이를 csv

```
df = pd.DataFrame(ids, columns=["Id", "SalePrice"])
df.to_csv(str(outdir + filename + '_submission.csv'), index=False)
```

# 3 학습하기



- 학습하기
  - 모든 내용을 한 번에 정리하면 다음과 같음

### 실습 | 캐글에 제출할 결과 만들기

```
from tensorflow.keras.models import load_model

import pandas as pd
import time

# 깃허브에 준비된 데이터를 가져옵니다.
!git clone https://github.com/taehojo/data.git

# 캐글에서 내려받은 테스트셋을 불러옵니다.
kaggle_test = pd.read_csv("./data/house_test.csv")
```

# 3 학습하기


- 학습하기

```
# 카테고리형 변수를 0과 1로 이루어진 변수로 바꿉니다.
kaggle_test = pd.get_dummies(kaggle_test)

# 결측치를 전체 칼럼의 평균으로 대체해 채워 줍니다.
kaggle_test = kaggle_test.fillna(kaggle_test.mean())

# 집 값을 제외한 나머지 열을 저장합니다.
cols_kaggle = ['OverallQual','GrLivArea','GarageCars','GarageArea','TotalB
smtSF']
K_test = kaggle_test[cols_kaggle]
```

# 3 학습하기



- 학습하기

```
# 앞서 15장에서 만든 모델을 불러옵니다.
model = load_model("./data/model/Ch15-house.hdf5")

# ID와 예측 값이 들어갈 빈 리스트를 만듭니다.
ids = []

# 불러온 모델에 K_test를 적용하고 예측 값을 만듭니다.
Y_prediction = model.predict(K_test).flatten()
for i in range(len(K_test)):
    id = kaggle_test['Id'][i]
    prediction = Y_prediction[i]
    ids.append([id, prediction])
```

# 3 학습하기



- 학습하기

```
# 테스트 결과의 저장 환경을 설정합니다.
timestr = time.strftime("%Y%m%d-%H%M%S")
filename = str(timestr) # 파일명을 연월일-시분초로 정합니다.
outdir = './'           # 파일이 저장될 위치를 지정합니다.

# Id와 집 값을 csv 파일로 저장합니다.
df = pd.DataFrame(ids, columns=["Id", "SalePrice"])
df.to_csv(str(outdir + filename + '_submission.csv'), index=False)
```

# 3 학습하기

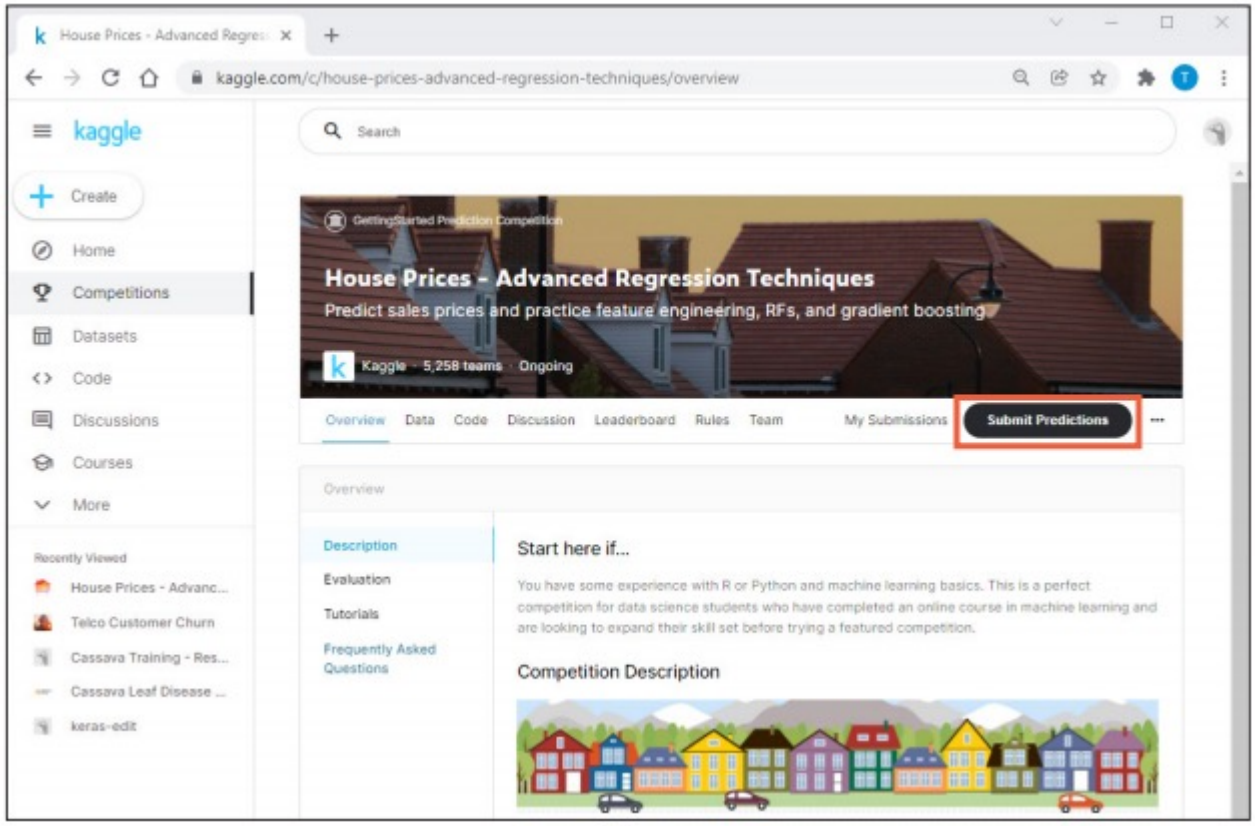

- **학습하기**
  - 이 코드를 실행해 구글 코랩 폴더에 (연도)(월)(일)-(시)(분)(초)_submission.csv 파일이 만들어졌다면 결과를 캐글에 제출할 준비가 됨

# 4 결과 제출하기

# 4 결과 제출하기

- **결과 제출하기**
  - 다시 경진대회 웹 페이지로 돌아가서 이번에는 **Submit Predictions**를 클릭

▼ **그림 22-10 | 결과 제출하기**

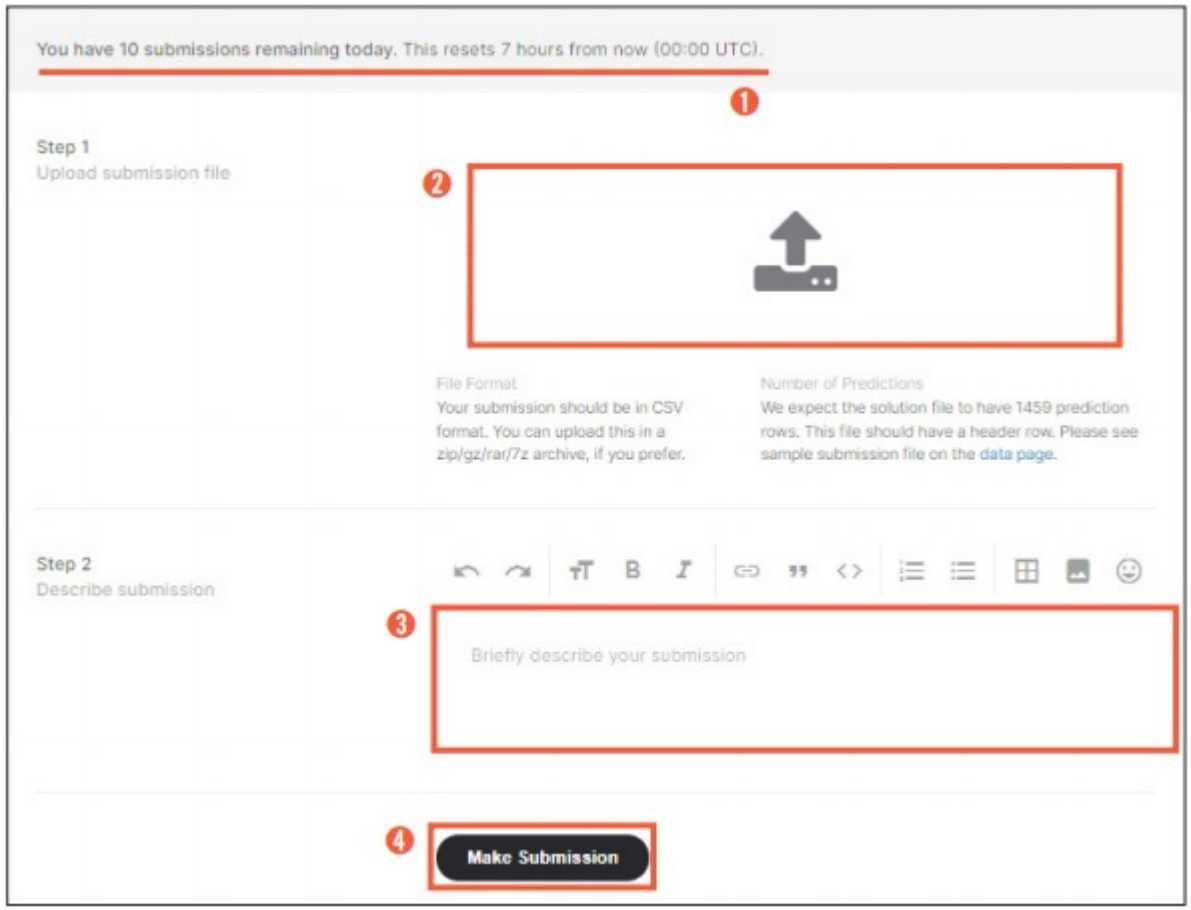


# 4 결과 제출하기


- **결과 제출하기**
  - ❶ 하루에 몇 번의 제출이 가능한지, 제출 횟수는 언제 리셋되는지에 대한 설명과 함께 예측 결과를 요약하고 업로드할 수 있는 페이지가 나옴
  - ❷ 를 눌러 조금 전 예측한 csv 파일을 선택
  - ❸ 해당 제출본이 어떤 모델이었는지 차후에 확인할 수 있도록 제출 전 간단히 모델에 대한 설명을 추가
  - ❹ **Make Submission**을 클릭해 제출을 마침

# 4 결과 제출하기

▼ 그림 22-11 | 결과 업로드하기



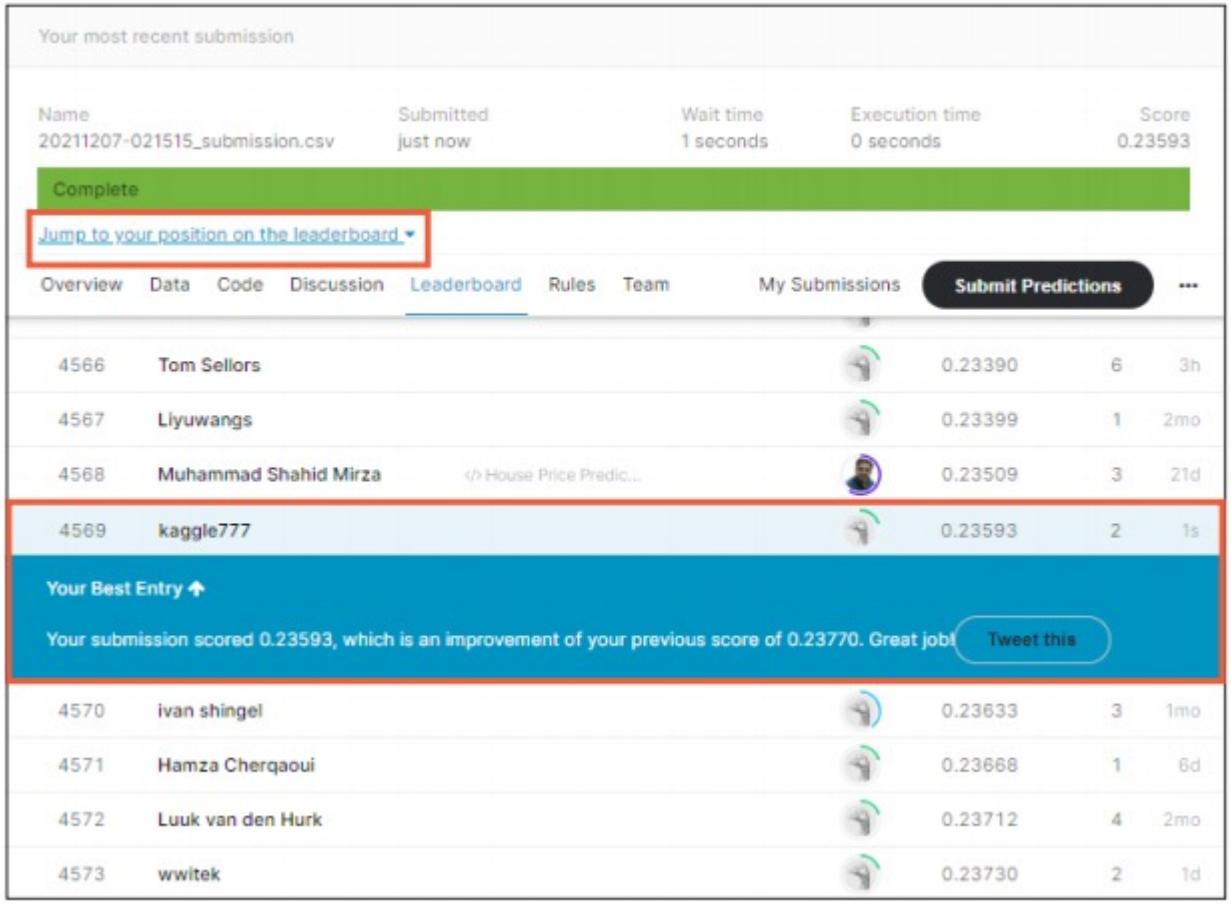
# 4 결과 제출하기



- **결과 제출하기**
  - 모든 과정을 무사히 마쳤다면 그림 22-12와 같이 작업이 완료되었다는 안내가 나옴
  - **Jump to your position on the leaderboard**를 클릭하면 내가 만든 예측 정확도와 순위가 표시

# 4 결과 제출하기

▼ 그림 22-12 | 내 예측 순위 보기



# 4 결과 제출하기


- **결과 제출하기**
  - 순위와 함께 이것이 내 몇 번째 제출인지, 이전 결과와 비교해서 어떤 차이가 있는지 알려줌
  - 결과를 이전 결과와 비교하고 또 다른 참가자들과 비교해 가면서, 모델을 지속적으로 업데이트
  - **Code**나 **Discussion** 메뉴를 선택하면 타 참가자들이 제출한 코드나 토론 내용 등이 나타남
  - 이를 읽어 보면 이전에 발견하지 못한 것들을 찾고 내 구현 방법을 업데이트하는 데 도움이 될 만한 기술적 조언들을 얻을 수 있음
  - 실제 대회라면 수정 후 제출을 대회 종료일까지 반복
  - 하루에 결과를 제출할 수 있는 횟수에 제한이 있으므로 전략을 잘 수립하는 것이 중요

# 5 최종 예측 값 제출하기

# 5 최종 예측 값 제출하기

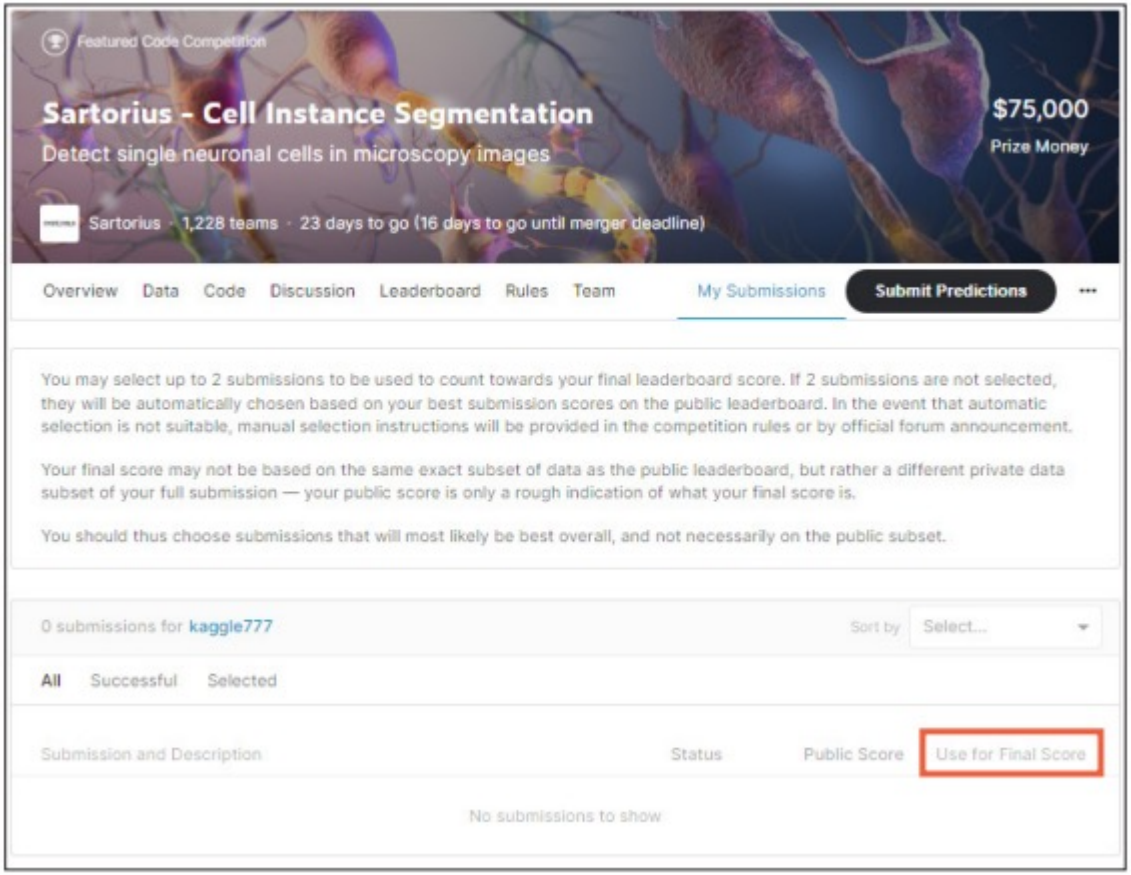

- **최종 예측 값 제출하기**
  - 우리가 제출한 대회는 상금이 걸린 실제 대회가 아니므로 최종 예측 값 제출의 개념이 없음
  - 상금과 마감일이 정해진 대회라면 마감 전에 그동안 제출했던 결과 중 하나를 최종 예측 값으로 결정해 제출
  - My Submissions를 클릭하면 그동안 제출한 결과를 한눈에 볼 수 있는데 학습용 대회는 'Public Score' 열 하나만 보이지만, 실제 대회에서는 'Use for Final Score'열이 있음
  - 여기에 체크하면 내 최종 결과가 제출

# 5 최종 예측 값 제출하기

▼ 그림 22-13 | 학습용 대회

# 5 최종 예측 값 제출하기

▼ 그림 22-14 | 실제 대회



# 5 최종 예측 값 제출하기


- **최종 예측 값 제출하기**
  - 최종 결과를 제출하면 며칠 후 모든 참가자의 결과를 합산해서 계산한 최종 순위가 발표