t2_bike_reg

```
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_score, train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
```

```
You are provided hourly rental data spanning two years. For this competition, the training set is comprised of the
first 19 days of each month, while the test set is the 20th to the end of the month. You must predict the total
count of bikes rented during each hour covered by the test set, using only information available prior to the rental
period.
Data Fields

datetime - hourly date + timestamp
season -  1 = spring, 2 = summer, 3 = fall, 4 = winter
holiday - whether the day is considered a holiday
workingday - whether the day is neither a weekend nor holiday
weather - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
temp - temperature in Celsius
atemp - "feels like" temperature in Celsius
humidity - relative humidity
windspeed - wind speed
casual - number of non-registered user rentals initiated
registered - number of registered user rentals initiated
count - number of total rentals
```

```
train=pd.read_csv('data/bike_sharing/train.csv')
test=pd.read_csv('data/bike_sharing/test.csv')
```

```
train.shape,test.shape
```

```
((10886, 12), (6493, 9))
```

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  object
 1   season      10886 non-null  int64
 2   holiday     10886 non-null  int64
 3   workingday  10886 non-null  int64
 4   weather     10886 non-null  int64
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
 9   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: float64(3), int64(8), object(1)
memory usage: 1020.7+ KB
```

```python
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6493 entries, 0 to 6492
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    6493 non-null   object
 1   season      6493 non-null   int64
 2   holiday     6493 non-null   int64
 3   workingday  6493 non-null   int64
 4   weather     6493 non-null   int64
 5   temp        6493 non-null   float64
 6   atemp       6493 non-null   float64
 7   humidity    6493 non-null   int64
 8   windspeed   6493 non-null   float64
dtypes: float64(3), int64(5), object(1)
memory usage: 456.7+ KB
```

```python
# predict the total count of bikes rented during each hour
```

```python
test['datetime'].value_counts()
```

```
datetime
2011-01-20 00:00:00    1
2012-05-21 02:00:00    1
2012-05-21 12:00:00    1
2012-05-21 11:00:00    1
2012-05-21 10:00:00    1
                      ..
2011-09-21 14:00:00    1
2011-09-21 13:00:00    1
2011-09-21 12:00:00    1
2011-09-21 11:00:00    1
2012-12-31 23:00:00    1
Name: count, Length: 6493, dtype: int64
```

```python
train.isnull().sum()
```

```
datetime      0
season        0
holiday       0
workingday    0
weather       0
```

```
temp          0
atemp         0
humidity      0
windspeed     0
casual        0
registered    0
count         0
dtype: int64
```

```
train = train.drop(['casual','registered'],axis=1)
train.shape
```

```
(10886, 10)
```

```
# train.select_dtypes(include='object').columns
```

```
# help(train.select_dtypes)
```

```
test.shape
```

```
(6493, 9)
```

```
target= train.pop('count')
```

```
# 전처리 오브젝트 구분
train['holiday'].value_counts()
```

```
holiday
0    10575
1      311
Name: count, dtype: int64
```

```
cat_cols= ['season','holiday','workingday','weather']
```

```
for col in cat_cols:
    train[col]= train[col].astype('object')
    test[col]= test[col].astype('object')
```

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
```

```
 0   datetime    10886 non-null  object
 1   season      10886 non-null  object
 2   holiday     10886 non-null  object
 3   workingday  10886 non-null  object
 4   weather     10886 non-null  object
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
dtypes: float64(3), int64(1), object(5)
memory usage: 765.5+ KB
```

```
train['datetime']=pd.to_datetime(train['datetime'])
test['datetime']=pd.to_datetime(test['datetime'])
```

```
train.shape,test.shape
```

```
((10886, 9), (6493, 9))
```

```
for df in [train,test]:
    df['year']= df['datetime'].dt.year.astype("object")
    df['month']=df['datetime'].dt.month.astype("object")
    df['hour']=df['datetime'].dt.hour # 이건 수치형으로 놔둬도 될거같다 판단.
```

```
# datetime 제거
train=train.drop('datetime',axis=1)
test= test.drop('datetime',axis=1)
```

```
train.shape,test.shape
```

```
((10886, 31), (6493, 31))
```

```
train=pd.get_dummies(train)
test= pd.get_dummies(test)
train.shape,test.shape
# train.columns.equals(test.columns)
```

```
((10886, 31), (6493, 31))
```

```
# 스플릿으로 테스트
```

```
X_train,X_test,y_train,y_test = train_test_split(train,target)
```

```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

```
((8164, 31), (2722, 31), (8164,), (2722,))
```

```
# 랜덤포레스트 모델 로드
rf= RandomForestRegressor(random_state=42,n_estimators=500)
```

```
rf.fit(X_train,y_train)
pred= rf.predict(X_test)
```

```
score= metrics.r2_score(y_test,pred)
score
```

```
0.944323660933783
```

```
from sklearn.metrics import mean_squared_log_error
import numpy as np

rmsle = np.sqrt(mean_squared_log_error(y_test, pred))
rmsle
```

```
0.34075200811877315
```

```
# rmsle 값으로 판단 대회 2등성적이 나옴
```

```
# 실제 테스트
```

```
rf.fit(train,target)
pred= rf.predict(test)
pred
```

```
array([  9.336,   4.59 ,   3.958, ..., 112.54 , 105.552,  58.617])
```

```
# X_train.info()
```

```
result = pd.DataFrame({'pred':pred})
result.to_csv('data/bike_sharing/result.csv',index=False)
```