O 사전 세팅

1. [워드] 단락 뒤 공백 제거
2. 캡처도구 켜두기
3. C:\windows\fonts에서 맑은 고딕 파일(malgun.ttf) Jupyter Notebook으로 옮겨두기  
   (fonts 폴더에서 ‘맑은’ 검색하면 됨)
4. 미리 코드 쳐두기(라이브러리 등)
5. 문제 파일 확인

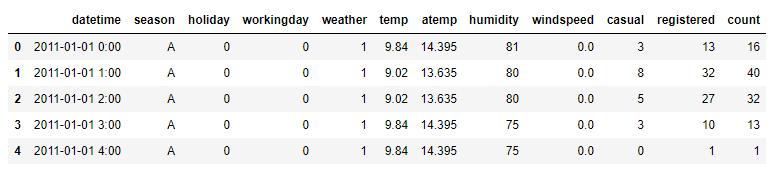
O 서식

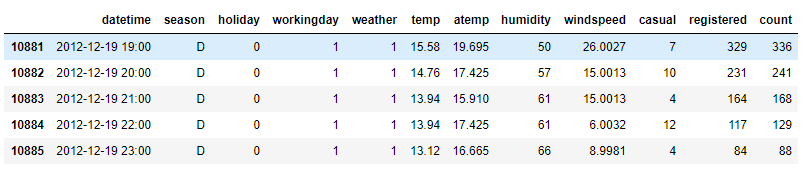
1. **대분류(Bold, 14pt)**
   1. 중분류
      1. 소분류

**2021. 3. 22**

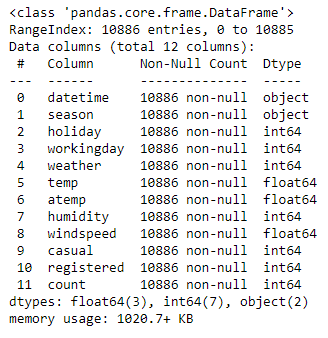
**1. 기계학습 1.1 분류**

1. **데이터 구조 확인 및 EDA**
   1. head, tail을 통해 데이터 처음과 마지막 5행 확인

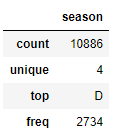
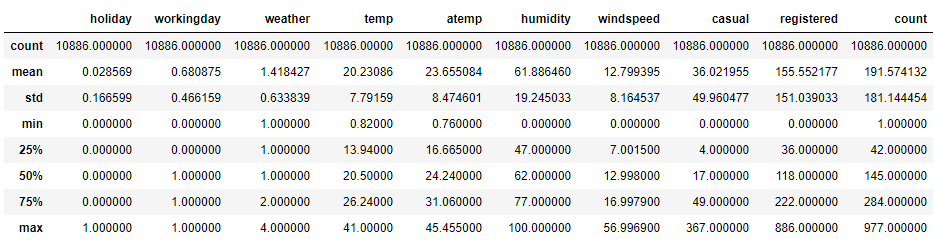




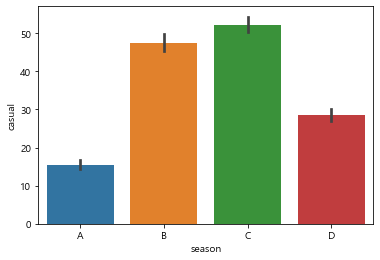
* 1. shape로 데이터 구조 확인 : 10,886행 x 12열
  2. info로 컬럼별 데이터 타입 확인



* 1. describe로 빈도와 분포 확인



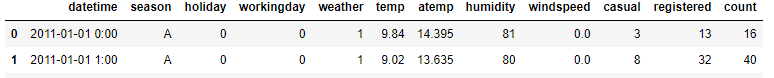
* 1. 시각화를 통한 데이터 구조 확인  
     (전체 컬럼 중 몇 가지 컬럼만 선별하여 첨부함)
     1. Season class에 따라 casual 평균 값에 차이가 있음을 확인했으며, season이 C일 경우 casual의 평균은 50이 넘어 전체 season 중 가장 높았으며, 반대로 A의 경우 약 15로 가장 낮았음



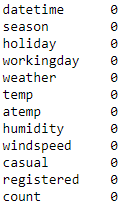
* + 1. Season별 빈도는 모두 2500 정도로 고르게 분포하고 있음



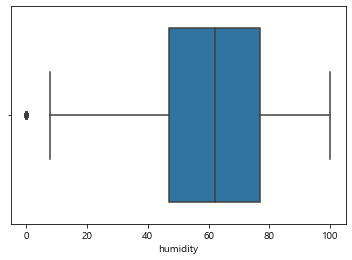
1. **데이터 마트 DQ check 및 변수 선택 @@전 버전 ‘분류’ 참고**
2. **데이터 전처리**
   1. 중복값 처리
      1. ~~ 사유로 중복이 발생한 것으로 생각됨

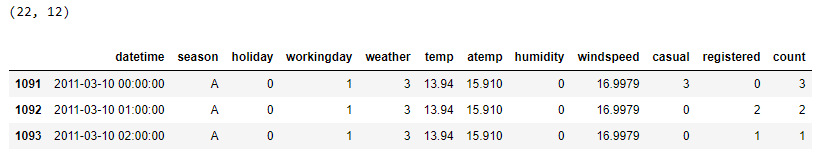


* + 1. ~~ 하여 중복값은 마지막 데이터만 남기기로 함
  1. 결측값 처리
     1. ~~와 @@에 결측값이 n개 있음을 확인함



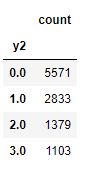
* + 1. ~~ 사유로 결측값은 0으로 대체함 / 결측값이 있는 데이터는 제거함 / 중앙값
  1. 이상치 처리
     1. 각 컬럼별로 boxplot을 그려 IQR 방식으로 이상치를 확인함
     2. 본 보고서에는 일부 변수에 대해서만 plotting함
        1. humidity 변수에서는 총 22개의 이상치를 확인함





* + 1. 데이터를 최대한 활용하기 위해 min/max값 경계 밖의 데이터를 min/max로 보정함 / 제거함

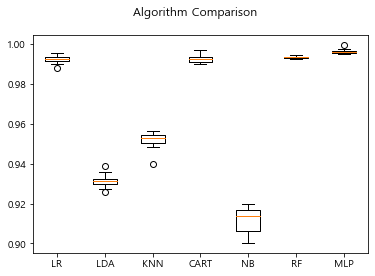
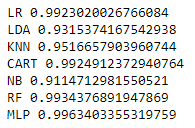
1. **파생변수 생성 @@전 버전 ‘분류’ 참고**
2. **종속변수 전처리(n개 클래스로 변환) @@전 버전 ‘분류’ 참고**
3. **변수 더미화**
   1. 특정 알고리즘을 활용한 모델링 과정에서의 에러 발생을 방지하기 위해 범주형 변수에 대해 더미화를 진행하였음
4. **다중공선성 제거 @@전 버전 ‘분류’ 참고**
5. **종속변수 분포 확인**



1. **Train, Test set 분리**
   1. 데이터 마트를 Train, Test set 7:3 비율로 분리함
2. **데이터 정규화**
   1. Standard Scaler로 데이터를 정규화함
3. **오버샘플링 진행**
   1. 종속변수의 class별 빈도가 불균형하여 SMOTE 방식 오버샘플링을 진행함

 🡪 

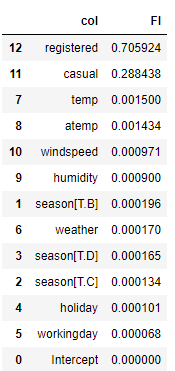
1. **군집화를 통한 파생변수 생성 @@전 버전 ‘분류’ 참고**
2. **모델링 및 기초 모델 선정 @@XGB는 필요시 전 버전 참고**
   1. 다양한 알고리즘을 활용하여 기본 모델을 구축함
      1. Logistic Regression
      2. Linear Discriminant Analysis
      3. K Neighbors Classifier
      4. GaussianNB
      5. RandomForest Classifier
      6. MLP Classifier
   2. Kfold=5 Cross validation을 통한 분류 정확도 기준으로 기준으로 보았을 때 MLP Classifier가 평균 0.996으로성능이 가장 높았음
   3. Training 시간을 감안하여 평균 정확도 0.993으로 두 번 째로 성능이 좋은 RandomForest Classifier를 최종 모델로 사용하기로 함



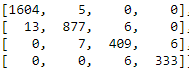
1. **파라미터 튜닝**
   1. 앞서 선택한 RandomForest Clasifier의 파라미터를 Gridsearch CV로 튜닝하여 성능을 개선함



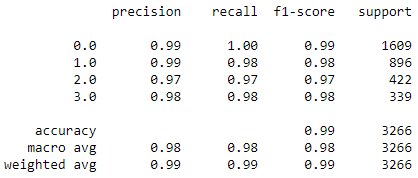
* 1. best parameter로 max\_feature : None, n\_estimators : 40이 선정됨
  2. 본 모델의 변수중요도는 registered, casual 순으로 높았음

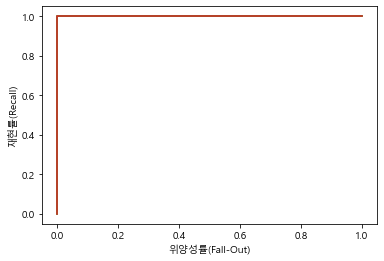
1. **Test set 활용하여 예측 수행 및 성능평가**
   1. Confusion Matrix
      1. Index 축이 실제(y\_test), col\_name 축이 예측(y\_pred)  
         엑셀로 하던가 해서 잘 그려보자



* 1. Classification Report
     1. Class별 F1-score는 class0 = 0.99, class1 = 0.98, class2 = 0.97
     2. 전체 F1-score는 0.98



* 1. ROC, AUC
     1. 4개 class에 대해 ROC를 도식화하고 ROC\_AUC를 계산함



AUC는 Class0: 1.0, Class:0.9 … 임.

//

1. **앙상블 모델 만들기**
   1. 1번 방법 : Randomforest로 그냥 제출
   2. 2번 방법 : 이항 분류 전제일 때, 모델 n개 골라서 각각 예측한 predict\_proba()의 평균을 0.5 기준으로 사용(결과 array중 왼쪽이 0일 확률, 오른쪽이 1일 확률)