**문제 개요**

삼성전자 A공장은 전기료 효율화를 통한 원가절감 방안을 찾고 있다.

공장 가동 시 일시적인 과부하로 최대수요전력이 증가하면 불리한 기본요금이 산정되고 이는 원가상승 요인이 된다.

전기 요금은 최대수요전력을 바탕으로 시간 단위로 산정된다. 최대수요전력을 예측하여 선제적으로 원가상승에 대응하고자 한다.

당일 데이터를 바탕으로, 그 다음 날의 최대수요전력을 예측하는 모델을 개발한다.

보유한 데이터는 A공장에서 2021년 1월 1일 ~ 2021년 9월 14일 (9개월) 동안 변압기에 감지센서를 부착하여 15분 단위로 최대수요전력을 관측한 데이터이다.

데이터는 2종류이며, ‘elec.csv’ 데이터는 최대수요전력을 측정한 데이터이며, ’info.csv’ 데이터는 측정 당시 기상정보 및 생산량과 인건비율 등의 정보가 포함되어 있다.

- 데이터 파일: elec.csv (24,642 rows, 4 columns, 데이터셋명-elec)

|  |  |  |
| --- | --- | --- |
| 컬럼명 | 내용 | 타입 |
| Date | 측정 날짜 | 문자열 |
| DateHour | 측정 날짜와 시간 | 문자열 |
| Minute | 15분 간격으로 측정  (15분, 30분, 45분, 60분) | 문자열 |
| Value | 최대수요전력(피크전기 사용량)  : 최대수요전력이란 공장 내 설치된 변압기를 통해 일정 시간 단위로 누적 계산되는 전력을 의미 | 정수형 |

- 데이터 파일: info.csv (6,168 rows, 9 columns, 데이터셋명-info)

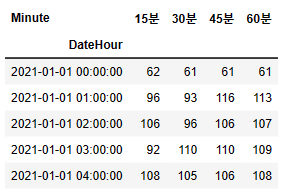
|  |  |  |
| --- | --- | --- |
| 컬럼명 | 내용 | 타입 |
| Date | 측정 날짜와 시간 | 문자열 |
| 생산량 | 해당 시점에 생산해야 할 생산량 | 정수형 |
| 기온 | 해당 시점의 기온 | 실수형 |
| 풍속 | 해당 시점의 풍속 | 실수형 |
| 습도 | 해당 시점의 습도 | 정수형 |
| 강수량 | 해당 시점의 강수량 | 실수형 |
| 전기요금(계절) | 해당시점의 전기요금 | 실수형 |
| 공장인원 | 해당 시점에 공장이 보유한 생산력 | 실수형 |
| 인건비 | 야간 및 주간 인건비에 대한 비율값 | 실수형 |

**전처리(Preprocessing)**

데이터를 분석하기 위하여 다음에 안내된 순서대로 데이터를 가공하시오.

**단계 1**: 데이터셋 elec를 이용하여 다음 그림과 같이 재구성하시오.

* 최대수요전력 (Value)을 시간당 15분 간격으로 관측하여 얻은 4번의 결과
* DateHour 변수 기준으로 시간당 한 행에 4번 측정결과가 나오도록 재구성함.
* 재구성된 데이터셋은 elec1로 표기(6,168 rows, 5 columns)



**단계 2**: 데이터셋 elec1의 DateHour 변수를 이용하여 아래의 파생변수를 생성한다.

* 요일명(DayName): DateHour 변수에서 요일명을 숫자로 추출, 0(월)~6(일)으로 반환
* 시간(Hour): DateHour 변수에서 시간 추출
* 오전/오후(AM): 생성한 시간(Hour) 변수를 기준으로 12시 전(< 12:00)은 오전(0), 이후(>= 12:00)는 오후(1)로 표기, 이때 표기는 숫자 0, 1로 표기
* 주말유무(Weekend\_yn): 생성한 요일명(DayName) 변수를 이용하여 5,6이면 1(주말), 그 외는 0(평일) 로 표기
* 공휴일(Holiday\_yn): 다음은 공휴일 목록임  
  ["2021-01-01", "2021-02-11", "2021-02-12", "2021-03-01", "2021-05-05", "2021-05-19", "2021-08-16"], 목록에 해당하면 1, 아니면 0로 표기한다. DateHour의 일자를 기준으로 판별한다.
* 최대수요전력 평균(Avg): 시간당 15분 간격(15분, 30분, 45분, 60분)으로 측정한 최대수요전력 (Value)의 평균값
* 최대수요전력 합계(TotalHour): 시간당 15분 간격(15분, 30분, 45분, 60분)으로 측정한 최대수요전력 (Value)의 총합

**단계 3**: info데이터셋에 결측치가 존재한다. 이 때 발생된 결측치들은 0으로 변경하시오.

* 풍속, 강수량은 바람이 불지 않는 경우, 비가 오지 않는 경우 기록되지 않는 경우가 존재함, 이는 0의 의미임
* 공장인원의 경우 생산량이 없다던가 인력이 필요하지 않은 경우 기록되지 않는 경우가 존재함, 이는 0의 의미임
* 결측치를 처리한 후 데이터셋명은 info1로 표기함

**단계 4**: elec1와 info1 데이터프레임을 결합하시오. 결합 시 elec1데이터셋에서는 DateHour 변수, info1데이터셋에서는 Date 변수를 기준으로 inner join하시오.

* elec1 데이터셋에 info1 데이터셋을 결합
* 결합 방식: inner join, elec1.DateHour == info1.Date
* 결합 후 Date와 DateHour 같다. 두 변수를 이름을 DateHour 하여 하나만 남긴다.
* 결합 후 생성된 데이터셋명 basetable1로 표기(6,168 rows, 20 columns)

**[문제 1]**

basetable1을 이용하여 다음 질문에 답하시오.

**[단계 1-1]** 시간당 최대수요전력 합계(TotalHour)의 최대값과 최소값의 차이를 구하고, 이 값을 A라고 한다.

**[단계 1-2]** 일별로 시간당 최대수요전력 합계(TotalHour)의 최대값을 찾고 최대치가 빈번하게 발생하는 시간을 찾고자 한다. 일별 기준(DateHour 변수 사용)으로 최대수요전력 합계(TotalHour)의 최대값이 첫 번째로 빈번하게 발생하는 시간(Hour)을 B, 두 번째로 빈번하게 발생하는 시간(Hour)을 C라고 한다. 하루에 최대값에 해당하는 시간(Hour)이 여러 번 나올 수 있음을 고려하라.

**[단계 1-3]** 일별 기준(DateHour 변수 사용)으로 시간당 최대수요전력 합계(TotalHour) 의 범위(최대값-최소값)가 가장 큰 값을 찾고 이 값을 D라고 한다.

**[단계 1-4]** 공휴일과 주말을 제외한 평일 데이터만 추출을 한 후, 일별 기준으로 시간당 최대수요전력 합계(TotalHour) 의 범위(최대값-최소값)가 가장 큰 일자의 요일명(DayName)을 E라고 한다.

A + B + C + D + E를 적으시오.

**[문제 2]**

최대수요전력 평균(Avg) 값이 오전/오후에 따라 차이가 있는지 검정하시오.

**[단계 2-1]** basetable1 에서 공휴일과 주말을 제외하고 prob2 데이터프레임을 만든다.

- prob2에서 시간(Hour)이 8과 11사이 (8, 11 포함) 또는 13과 16 (13, 16 포함) 사이의 행들만 남긴다..

- prob2에서 Avg가 130이상인 행들만을 남긴다.

Hint] 최종 1233개 행만 남는다.

**[단계 2-2]** 오전(AM = 0) 인 행들을 대상으로 Avg에 대하여 Shapiro-Wilk 검정을 사용하여 정규성 검정한다. 검정 결과에서 p-value를 A라고 한다.

**함수 가이드**

scipy.stats 제공 기능 활용

문제 지시사항 외 Default 값 사용

**[단계 2-3]** 오후(AM = 1) 인 행들을 대상으로 Avg에 대하여 Shapiro-Wilk 검정을 사용하여 정규성 검정한다. 검정 결과에서 p-value를 B라고 한다.

**[단계 2-4]** 오전/오후에 따라 Avg의 모평균에 차이가 있는지 검정을 한다. 검정 방법은 T-검정을 사용하며, Bartlett 검정을 하고 결과를 T-검정 시 반영한다(유의 수준 5%). 검정 결과에서 검정통계량의 절대값을 C라고 한다.

**함수 가이드**

scipy.stats 제공 기능 활용

문제 지시사항 외 Default 값 사용

**[단계 2-5]** 단계 2-2와 2-3의 검정 결과를 바탕으로, 단계 2-4의 T-검정의 가정을 만족한다면 D 값을 1 아니면 0으로 한다. (유의 수준은 5%이다.)

**[단계 2-6]** 오전/오후에 따라 Avg를 구분하여 Mann-Whitney U 검정을 하라. 검정 결과 p-value를 E값으로 한다. 대립 가설은 '오전의 Avg와 오후의 Avg의 중앙값은 같지 않다' 이다.

**함수 가이드**

scipy.stats.mannwhitnyeu, alternative를 문제 요건에 반드시 맞추어 설정합니다.

문제 지시사항 외 Default 값 사용

A + B + C + D + E를 소수점 넷째 자리에서 반올림하여 셋째 자리까지 적으시오

Ex) 3.142

**[단계 2-7, Optional]** Avg에 대하여 AM, DayName 두 범주에 대하여 이원산 분산분석(Two-way ANOVA)을 수행하고 상호작용항(교호항)의 검정통계량을 구하라.

**함수 가이드**

statsmodels.formula.api.ols,

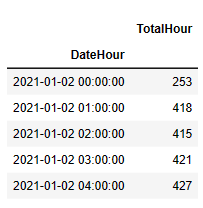
statsmodels.api.stats 를 활용합니다.

문제 지시사항 외 Default 값 사용

**[문제 3]**

DateHour를 기준으로 다음날 같은 시각의 최대전력사용량을 예측하려고 한다.

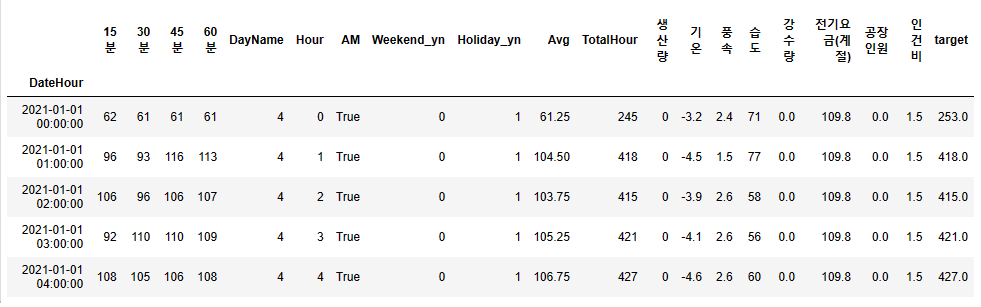
**[단계 3-1]** basetable1에 아래 그림과 같이, DateHour를 기준으로 다음날 같은 시각의 TotalHour를 이용하여 파생 변수 target을 생성한다. target에 결측치가 있는 행은 모두 제거하여 prob3 데이터셋을 만든다.



DateHour 기준으로 다음날 같은 시각의 TotalHour를

파생변수로 추가

2021-01-02 00:00:00 이후 TotalHour



**[단계 3-2]** 학습용과 테스트용 데이터셋을 2021년 8월 14일 전과 이후로 분리

. prob3\_train: DateHour 변수 기준으로 2021년 8월 14일 전(8월 14일 미포함) 데이터 (행의 수: 5400개)

. prob3\_test: DateHour 변수 기준으로 2021년 8월 14일 이후(8월 14일 포함) 데이터 (행의 수: 744 개)

**[단계 3-3]** 결정트리 회귀모델(Decision Tree Regression Model)을 사용하여 target을 예측하는 모델을 prob3\_train으로 학습시킨다. 요건은 다음과 같다.

성능 측정 지표:

입력 변수: 15분, 30분, 45분, 60분, DayName, Hour, AM, Weekend\_yn, Holiday\_yn, Avg, TotalHour, 생산량, 기온, 풍속, 습도, 강수량, 전기요금(계절), 공장인원, 인건비

※ 입력 변수의 순서는 반드시 나열한 순서로 해야 함.

대상 변수: target

결정 트리 구성 요건:

트리의 최대 깊이: 10

리프 노드(Leaf Node)의 최소 샘플 수: 4

랜덤 시드: 123

**함수 가이드**

sklearn.tree 제공 기능 활용, 문제 지시사항 외 Default 값 사용

모델의 학습 결과 가장 중요도가 높은 변수의 입력 순서를 A (순서는 1부터 시작)에 저장한다.

그리고, 결정 트리의 prob3\_test에 대한 성능을 B라고 한다.

**[단계 3-4]** 결정트리 대신에 경사 부스팅(Gradient Boosting) 회귀모델을 사용한다. 입력변수와 대상 변수는 3-3과 같다.

경사 부스팅(Gradient Boosting) 회귀모델구성 요건:

손실함수: huber

트리의 수: 75

트리의 최대 깊이: 7

리프 노드(Leaf Node)의 최소 샘플 수: 4

랜덤 시드: 123

**함수 가이드**

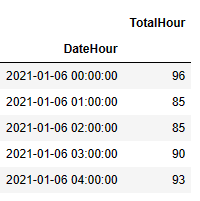
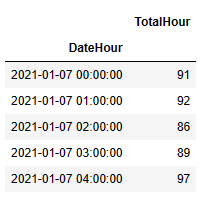
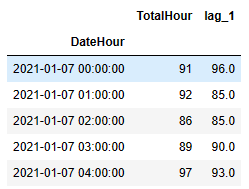
sklearn.ensemble 제공 기능 활용, 문제 지시사항 외 Default 값 사용

A + B + C를 소수점 넷째 자리에서 반올림하여 셋째 자리까지 적으시오

Ex) 3.142

**[문제 4]**

DateHour를 기준으로 동일한 시각의 n일 이전의 TotalHour를 파생 변수 lag\_n이라고 한다. 예를 들어 DateHour가 2021-01-07 00:00:00이라면, lag\_1은 2021-01-06 00:00:00의 TotalHour값인 96이 된다.

   
**[단계 4-1]** prob3에 lag\_1 ~ lag\_6까지의 6개의 파생 변수를 추가하여 prob4를 만든다. prob4에 lag\_1~6까지 하나라도 결측치가 있다면 삭제한다.

**[단계 4-2]** 학습용과 테스트용 데이터셋을 2021년 8월 14일 전과 이후로 분리

. prob4\_train: DateHour 변수 기준으로 2021년 8월 14일 전(8월 14일 미포함) 데이터 (행의 수: 5256개)

. prob4\_test: DateHour 변수 기준으로 2021년 8월 14일 이후(8월 14일 포함) 데이터 (행의 수: 744 개)

**[단계 4-3]** prob4\_train에서 TotalHour와 lag\_1~6 를 표준화하고, prob4\_train의 표준화 설정으로 prob4\_test의 TotalHour와 lag\_1~6를 표준화한다.

**함수 가이드**

sklearn.preprocessing 제공 기능 활용, 문제 지시사항 외 Default 값 사용

**[단계 4-4]** SVR 모델을 아래 요건에 3가지 커널(Kernel)로 prob4\_train을 학습하고 prob4\_test로 성능을 측정한다. 그 중 가장 좋은 성능을 보인 커널의 번호(No.)를 A, 그 때의 성능을 B라고 한다.

성능 측정 지표:

입력 변수: TotalHour, lag\_1 ~ 6

대상 변수: target

커널함수:

|  |  |
| --- | --- |
| No. | 커널 함수 |
| 1 |  |
| 2 |  |
| 3 |  |

**함수 가이드**

sklearn.svm.SVR, C=10

문제 지시 사항 외 Default 값 사용

A + B를 소수점 셋째 자리에서 반올림하여 둘째 자리까지 적으시오

Ex) 3.14

**[단계 4-5, Optional]** 퍼셉트론 회귀모델을 다음과 같이 3가지 설정으로 prob4\_train으로 학습하고, prob4\_test로 성능을 측정한다. 가장 좋은 성능을 보인 경우의 번호(No.) C, 그 때의 성능을 D라고 한다.

성능 측정 지표:

입력 변수: TotalHour, lag\_1 ~ 6

대상 변수: target

요건] sklearn.neural\_network.MLPRegressor,

learning\_rate\_init=0.05, random\_state=123, max\_iter=500

|  |  |  |
| --- | --- | --- |
| No. | 은닉층의 수 | 은닉층의 활성화 함수 |
| 1 | [8] |  |
| 2 | [8, 4] |  |
| 3 | [8, 4] |  |

**[문제 5]**

info에서 기온, 풍속, 습도 강수량 데이터에서 이상치를 탐색하고자 한다.

**[단계 5-1]** info1에서 기온, 풍속, 습도, 강수량 만을 떼어 weather1 데이터프레임을 만든다.

**[단계 5-2]** weather1의 기온, 풍속, 습도, 강수량을 표준화 한다.

**함수 가이드**

sklearn.preprocessing 제공 기능 활용, 문제 지시사항 외 Default 값 사용

**[단계 5-3]** DBSCAN 군집화를 입실론(epsilon) 반경을 0.5, 주변 영역에 포함하는 최소 샘플수를 7로하여 군집화를 수행 한다. weather1에 DBSCAN 결과에서 소속 군집이 없는 샘플들을 이상점으로 간주한다. 이상점 여부를 파생변수 outlier\_dbscan로 하여 weather1에 추가한다. (소속 군집이 있으면: False, 없으면 True)

입력변수: 기온, 풍속, 습도, 강수량

**함수 가이드**

sklearn.cluster 제공 기능 활용, 문제 지시사항 외 Default 값 사용

**[단계 5-4]** IsolationForest를 사용하여 weather1의 이상점을 탐색한다. 학습시 bootstrapping은 하지 않고, 트리의 수는 30개를 사용한다. IsolationForest으로 탐지한 이상점 여부는 파생변수 outlier\_isof로 하여 weather1에 추가한다. (이상치라면 True, 아니면 False)

입력변수: 기온, 풍속, 습도, 강수량

**함수 가이드**

sklearn.ensemble.IsolationForest random\_state=123, contamination=’auto’, behaviour=’auto’

문제 지시사항 외 Default 값 사용

**[단계 5-5]** LocalOutlierFactor를 사용하여 weather1의 이상점을 탐색한다. 관찰할 주변의 샘플은 10개이다. 그리고, 샘플간의 거리는 유클리디언 거리를 사용한다. LocalOutlierFactor로 탐지한 이상 여부를 outlier\_lof로 하여 weather1에 추가한다.

입력변수: 기온, 풍속, 습도, 강수량

**함수 가이드**

sklearn.neighbors.LocalOutlierFactor, contamination=’auto’ 문제 지시사항 외 Default 값 사용

**[단계 5-6]** outlier\_dbscan가 True인 행의 개수를 A, outlier\_isof가 True인 행의 개수 B

, outlier\_lof가 True인 행의 개수 C 이다.

**[단계 5-7]** outlier\_dbscan, outlier\_isof 그리고 outlier\_lof 모두가 True인 행의 개수는 D이다.

A + B + C + D를 적는다.

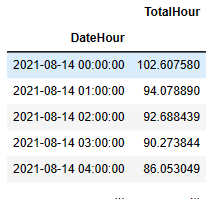
**[문제 6]**

(kaggle 형 문제) prob4 를 바탕으로 아래 데이터셋을 만든다.

. prob6\_train: DateHour 변수 기준으로 2021년 8월 14일 전(8월 14일 미포함) 데이터 (행의 수: 5256개)

. prob6\_test: DateHour 변수 기준으로 2021년 8월 14일 이후(8월 14일 포함) 데이터 (행의 수: 744 개)

일 때, prob6\_train으로 target을 예측하는 모델을 만들어, prob6\_test의 target에 대한 MAE를 최소화하는 모델을 만든다. prob6\_test의 예측 결과를 아래와 같은 형식으로 출력한다. 파일명은 answer6.csv 이다.



※ 실제 상황에 가까운 연습을 위해prob6\_test의 target 값은 모른다는 상황을 가정하고 문제를 푼다.

(정답)

|  |  |  |
| --- | --- | --- |
| 문제 | 시험 버전 | BDC 버전 |
| 1 | 1555 | 1555 |
| 2 | 8.348 | 8.348 |
| 3 | 78.322 | 78.223 |
| 4 | 33.02 | 33.02 |
| 5 | 1417 | 1520 |
| 6 | 27~35 | 27~35 |