

Bigdata Analytics

Final Report

INE5015 – 22057

DO NOT LEAK THIS DOCUMENT

빅데이터 애널리틱스 8조

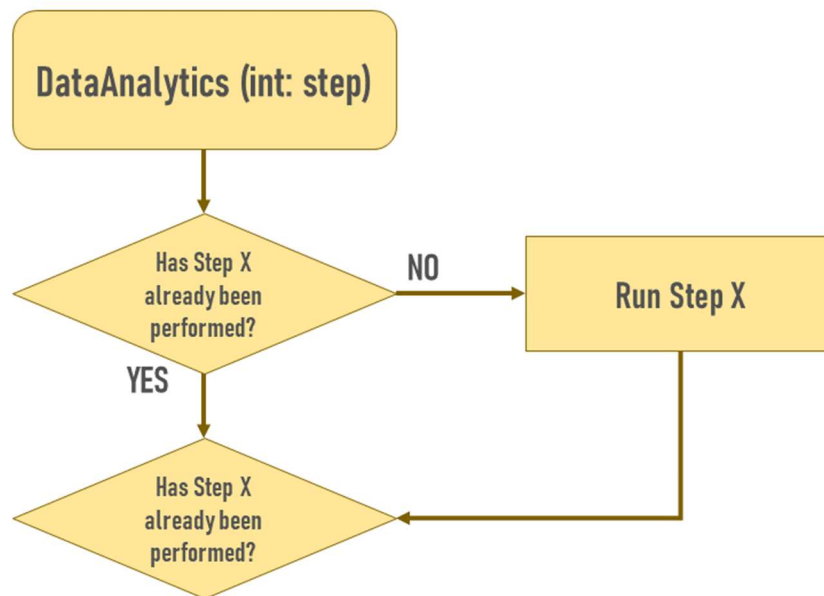
| 최준희 | 이강산 | 장혜연 | 황태영 |

List of Contents

- Logic of Data Preprocessing
- step-by-step process
 - Step 0 ~ Step 2 : raw file refining
 - Data Cleaning Overview (3 Steps)
 - ◆ Step 3 : correlation check and correction
 - ◆ Step 4 : Missing Value Imputation
 - ◆ Step 5 : Outlier corrections
 - Step 6 : Data Scaling
- Feature Selection
- Data over/down Sampling overview
- Performance
 - The performance of two samplings
 - Prediction accuracy according to each algorithm
 - ◆ Logistic Regression
 - ◆ Decision Tree
 - ◆ Random Forest
 - ◆ Boosting

Logic of our Data Processing

전반적인 Preprocessing 과정은 기능, 목적에 따라 Step으로 구별했고, 매 Step마다 시각적, 통계적 데이터를 확인하면서 진행하였다. 아래는 Step에 대한 구성 Logic이다.



Proceed with 8 steps as above

DataAnalytics 함수에서 필요한 함수를 호출하여 사용하는 구조이며, 각 단계마다 csv 파일을 저장하도록 하여 변화를 직관적으로 확인하기 용이하도록 디자인하였다.

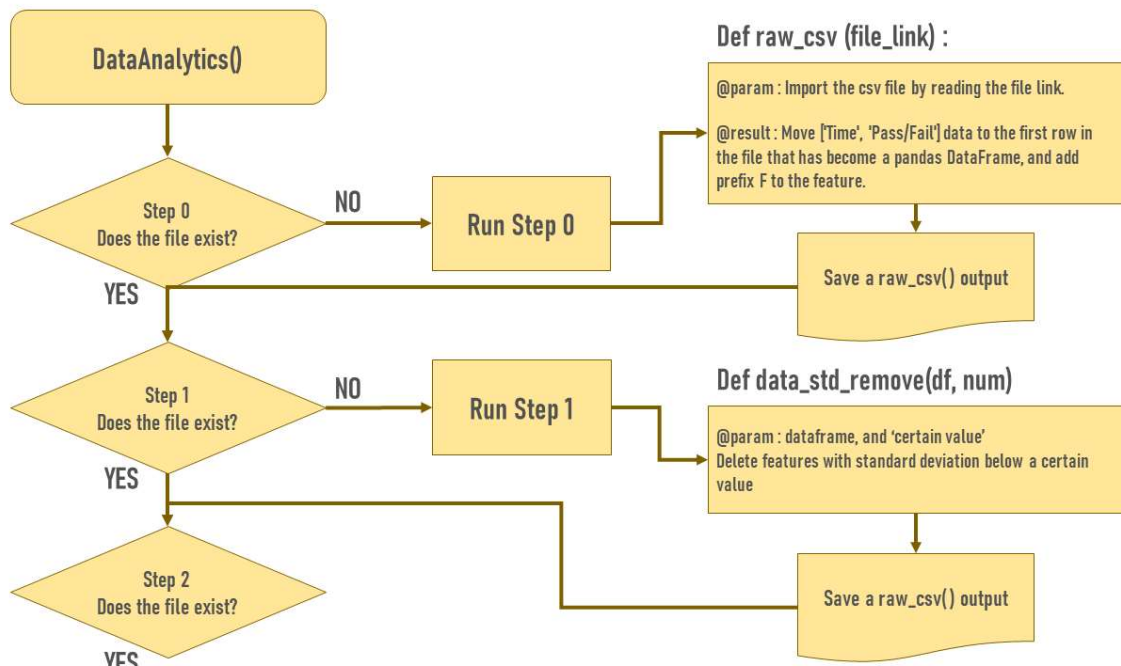
각 Step을 크게 나누어 보면 Raw data를 가공에 용이하도록 조정하는 부분, Pass/Fail 데이터를 분리해 총 3가지 데이터셋으로 나누어 진행할 수 있도록 하는 부분이 있다.

이후 각 Feature (독립변수)간 종속성을 확인하고, 제거하기 위해 진행하는 Correlation 과, 결측치 처리/보정 과정, 이상치 처리/보정 과정. 즉, 데이터 클리닝 과정이 있다.

마지막으로 Feature Selection과 샘플링을 통해 최종 데이터 셋 후보를 선정하고, 퍼포먼스를 확인하는 과정으로 마무리한다.

Step 0 ~ Step 2 : Ready to Preprocess

이 단계는 데이터 파일을 불러오고, 이후 보정에 있어 편의성을 높이기 위해 보정한다.



Explanation of Steps

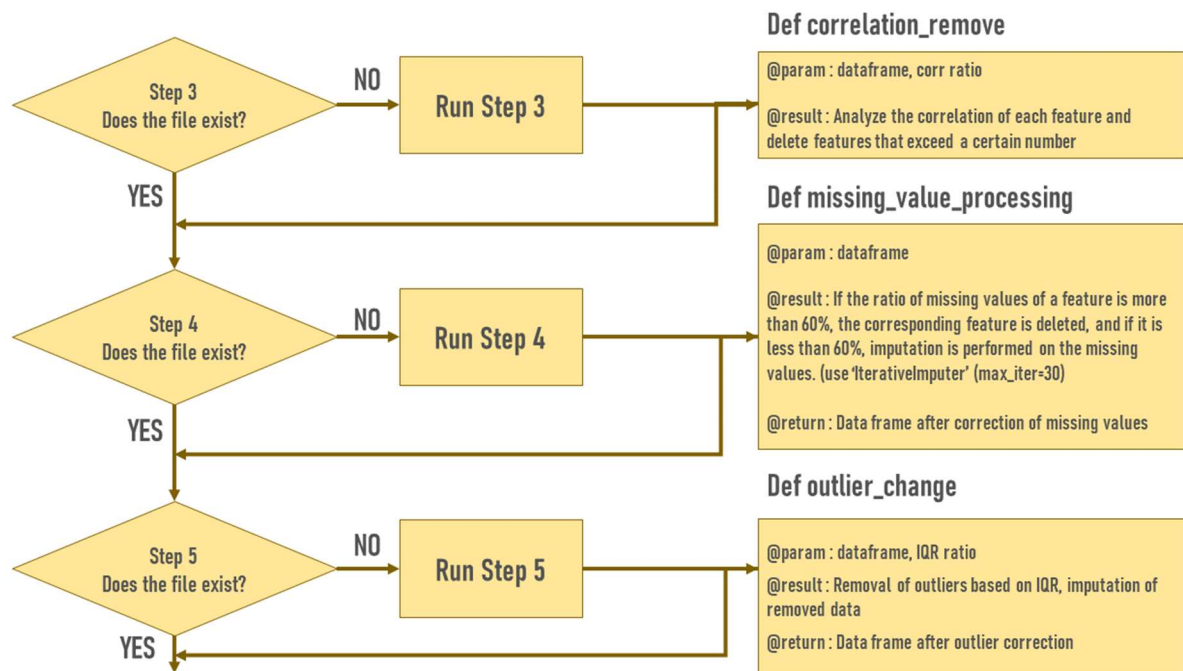
STEP	STEP DESCRIPTION
STEP 0	raw_csv 함수를 통해, Pandas Read_CSV를 실행하여 데이터 셋을 받아온다. 이후 작업의 편의를 위해 행 위치 변경, Prefix 추가 등의 과정을 거친 Dataframe 파일을 받고, 저장한다.
STEP 1	일정 계수 미만의 표준편차를 가진, 특징 선택의 필요도가 낮은 Feature들을 제거하고, Dataframe 파일을 저장한다. 그리고 데이터 셋을 3개로 나누는 과정을 거친다.
STEP 2	데이터 셋은 Pass Data, Fail Data, Both Data로 나뉘며, 이렇게 데이터 셋을 나누는 것은 프로젝트 초기의 아이디어 중 유용한, 유의미한 결과를 가져오는 계기가 된다.

Step 0 부터 Step 2까지의 3 과정은 데이터 전 처리를 위한 준비 단계에 불과하다.

> 위 과정에서 1566개 (1463+104)의 테스트 케이스, 247개의 Feature Data가 남았다.

Step 3 ~ Step 5 : Data Cleaning

이 단계에서는 데이터셋의 노이즈를 제거하기 위해 데이터 클리닝을 진행하는 과정이다. 크게 독립변수 간 종속성 (비율), 결측치 제거 및 보정, 이상치 제거 및 보정으로 나뉜다.



Explanation of Steps

STEP	STEP DESCRIPTION
STEP 3	각 독립변수 (Feature)간 상관성이 높다는 것은 다중공선성 발생 소지가 있고, 정확한 예측을 방해하는 요인으로 작용할 수 있다. 상관관계가 높은 Feature들을 제거하는 과정이다.
STEP 4	데이터에 결측치가 많아도 정확한 예측을 방해한다. 독립변수의 수를 최대한 유지하기 위해, 결측치가 60% 이상인 Feature를 제거하고, 60% 미만은 IterativeImputer를 통해 보정한다.
STEP 5	각 Feature의 사분위 값을 확인하고, IQR 방식을 통해 양 극단에 있는 이상치들을 제거한다. 이 때, Fail 데이터셋은 오버샘플링을 진행하지 않았기 때문에 가중치를 달리 설정한다.

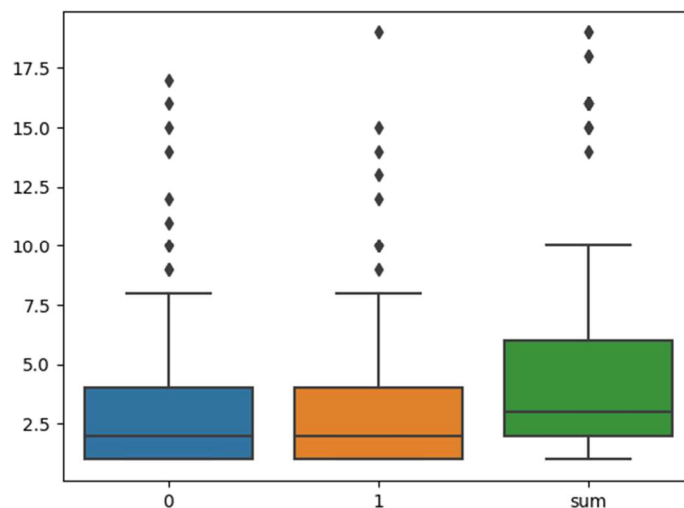
다음 페이지에는 Step3, 4, 5의 구체적인 내용과 편차 제거의 효용성을 비교한다.

Data Cleaning - Overview

데이터 전처리 과정에 있어, 데이터 클리닝 과정은 매우 중요하다. 특히 결측치와 이상치에 대해 처리하는 것은 데이터 분석 및 모델링 결과를 크게 변화할 수 있기 때문에, 데이터의 불안전성과 잡음, 불일치 등을 최대한 효과적으로 처리해야 한다.

Step 3 - Correlation

상관관계를 분석 (Correlation Analysis)을 한다는 것은 통계학적으로 두 변수간 선형적 관계를 분석하는 것이다. 독립 변수 (=Feature)간 상관관계가 높다면 두 변수의 연관성이 높다는 것이고, 필요치 않은 연산을 할 뿐만 아니라 Clustering 에도 방해가 된다.



우리는 이 프로젝트 과정에서 상관관계가 높은 Feature들을 계산했고 아래 예시처럼 선택된 Feature들을 제거했다. (여기서, Fail 데이터와 All/Pass 데이터간 상관관계 분석 내용이 달라 약 20여개의 Feature가 Fail 데이터에는 남아있게 되었다.

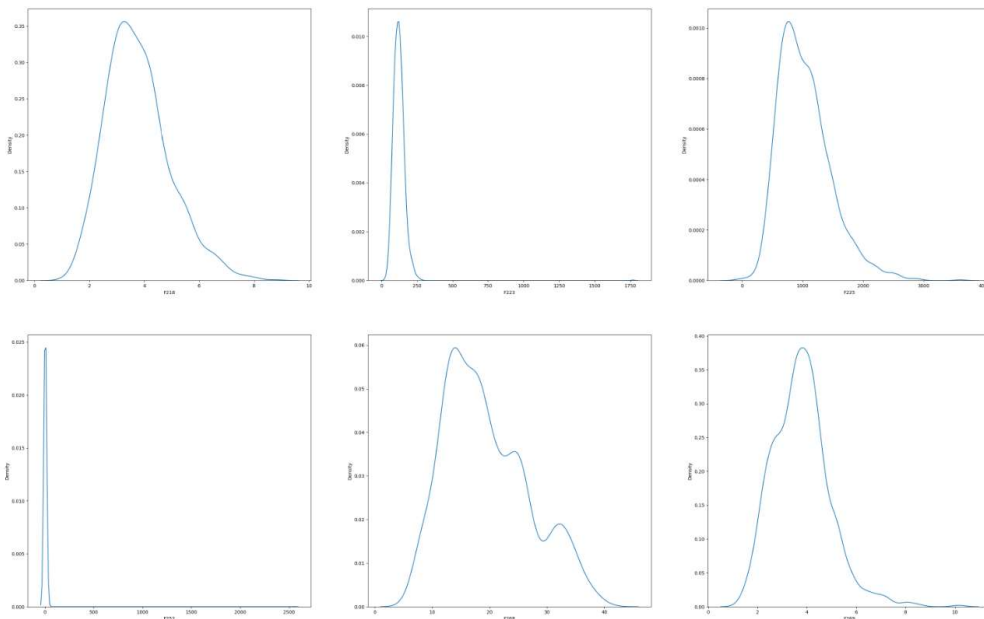
(시각화 자료 들어갈 위치)

Step 4 – Missing Value

이 프로젝트에 있는 결측치는 패턴이 없는, Random Missing Feature 라 가정하고 진행하였다. 결측치를 처리하는 방법에는 삭제, 대치, 예측이 있는데, 결측치들의 특성이 패턴을 가지고 있다는 가정 하에 예측 모델을 구현해야 하기 때문에 이 프로젝트에서는 Deletion, Imputation 두 가지 방법을 사용하였다.

Scikit Learn에서 제공하고 있는 impute 중 Iterative Imputer를 사용하였다. 다른 모든 특성에서 개별 특성을 추정하는 다변량 대치 방식이며, Round Robin 알고리즘으로 각 Feature를 모델링 하여 결측값을 대치하는 기능을 한다.

또한 Iterative Imputer는 KNN 알고리즘으로 결측치를 예측하여 채워 넣는 방식인데, Max-Iter를 30으로 조정함으로써 최종 라운드 동안 계산된 결과를 반환하기전에, 수행할 Round의 수를 늘림으로써 데이터의 완전성이 높아지기를 기대하였다.

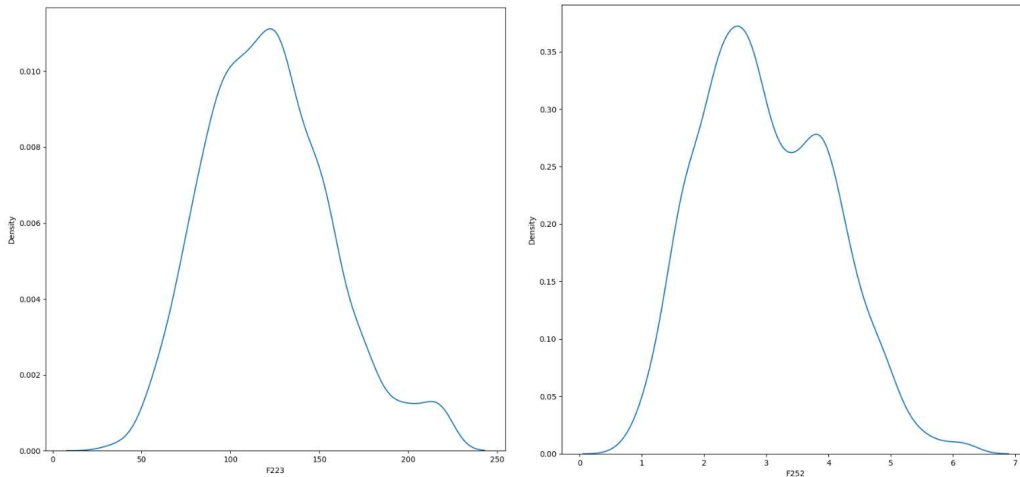


위 사진은 결측치 보정이 완료된 Feature들의 일부 사진이다. 위 사진에서 있는 F223, F252의 경우 이상치 처리와 보정이 이루어지지 않아 편차의 정도를 알 수 없으며, 다음 단계인 이상치 보정 (삭제, 보정, 편차제거) 단계를 통해 결과를 확인할 수 있다.

기존에는 Fail 데이터에 IQR 가중치를 높여 20여개의 Feature 개수 차이가 났으나, 데이터 왜곡이 우려되어 동일하게 적용하였다.

Step 5 - Outlier Value

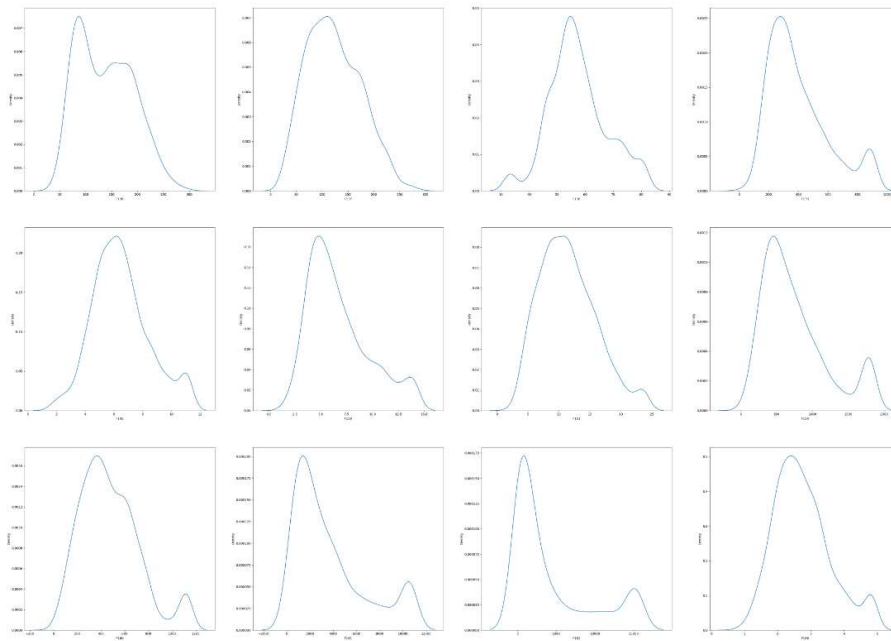
이상치의 기준은 IQR Weight 1.5 를 적용하였으며, 일반적인 이상치들을 보정할 수 있었다. 기준 이상인 데이터는 기준치의 최대값으로 대체 하였고, 기준 이하인 데이터는 기준치의 최소값으로 대체 하였다. 이후 편차 제거를 통해 대체 방법의 단점을 상쇄하였다.



Step 4에서는 편차가 작아 제거되어야 할 대상으로 보였던 Feature 223, 252의 이상치 처리 이후의 그래프다. 이상치를 제거하면서 생긴 편차의 변동을 반영하기 위해 편차 제거를 약한 단계 (표준편차 기준 0.1 0.5 1.0)부터 높은 단계로 여러 번 시각화를 통해 확인한 결과 중간 단계가 이 단계에서의 최적의 값이라고 판단 되었다.

테스트 해본 결과, 편차의 변동이 생기는 STEP5에서만 Feature 수의 변화가 있었다.

STEP3	STEP4	STEP5
<pre> 119 ## corr 0.3 / 0.6 큰 의미 없음 / std3 진행 120 s3_c30_all = correlation_remove(std30_all, 121 print("----- 필터링 -----") 122 print(s3_c30_all) 123 124 s3_c30_all_2 = data_std_remove(s3_c30_all, 125 print("----- 필터링 -----") 126 print(s3_c30_all_2) 127 print("----- 필터링 -----") 128 return 129 df.to_csv('./[step 0] - rawfile_low_refine 130 131 s3_c30_all.to_csv('./[step 3] - correlatio 132 </pre> <p>1567 rows x 202 columns]</p> <pre> Time Pass/Fail F0 ... 2008-07-19 11:55 -1 3030.93 ... 2008-07-19 12:32 -1 3095.78 ... 2008-07-19 13:17 1 2932.61 ... 2008-07-19 14:43 -1 2988.72 ... 2008-07-19 15:22 -1 3032.24 562 2008-10-16 15:13 -1 2899.41 ... 563 2008-10-16 20:49 -1 3052.31 ... 564 2008-10-17 5:26 -1 2978.81 ... 565 2008-10-17 6:01 -1 2894.92 ... 566 2008-10-17 6:07 -1 2944.92 ... </pre> <p>1567 rows x 202 columns]</p>	<pre> 158 s4_all = pd.read_csv('./[step 4] - D 159 s4_pass = pd.read_csv('./[step 4] - D 160 s4_fail = pd.read_csv('./[step 4] - D 161 162 print("----- 필터링 -----") 163 print(s4_all) 164 165 s4_all_2 = data_std_remove(s4_all, 1 166 print("----- 필터링 -----") 167 print(s4_all_2) 168 print("----- 필터링 -----") 169 return 170 171 #이상치 처리 -> fail 데이터가 너무 작아 172 173 s5_w15_p50_pass = outlier_change(s4_p 174 </pre> <p>1567 rows x 181 columns]</p> <pre> Time Pass/Fail F0 ... 2008-07-19 11:55 -1 3030.93 ... 2008-07-19 12:32 -1 3095.78 ... 2008-07-19 13:17 1 2932.61 ... 2008-07-19 14:43 -1 2988.72 ... 2008-07-19 15:22 -1 3032.24 562 2008-10-16 15:13 -1 2899.41 ... 563 2008-10-16 20:49 -1 3052.31 ... 564 2008-10-17 5:26 -1 2978.81 ... 565 2008-10-17 6:01 -1 2894.92 ... 566 2008-10-17 6:07 -1 2944.92 ... </pre> <p>1567 rows x 181 columns]</p>	<pre> 168 print("----- 필터링 -----") 169 print(s5_w15_p50_all) 170 171 s4_all_1 = data_std_remove(s5_w15_p50_all, 0.5 172 print("----- 필터링 -----") 173 print(s4_all_1) 174 print("----- 필터링 -----") 175 return 176 177 # 이상치 제거 과정에서 데이터가 너무 작아 (표준 편차 기준) 178 # IQR 값과 같은 기준을 사용 하겠다 179 180 s5_w15_p50_all.to_csv('./[step 5] - D 181 s5_w15_p50_pass.to_csv('./[step 5] - D 182 s5_w15_p50_fail.to_csv('./[step 5] - D 183 184 print("[*] Step 5 - Complete.") 185 step = 5 186 187 (step == 0): 188 # 시작점 설정 </pre> <p>1567 rows x 181 columns]</p> <pre> Time Pass/Fail F0 ... F57 2008-07-19 11:55 -1 3030.93 ... 24.950 2008-07-19 12:32 -1 3095.78 ... 10.900 2008-07-19 13:17 1 2932.61 ... 9.272 2008-07-19 14:43 -1 2988.72 ... 8.503 2008-07-19 15:22 -1 3032.24 ... 10.869 ... 562 2008-10-16 15:13 -1 2899.41 ... 11.725 563 2008-10-16 20:49 -1 3052.31 ... 17.027 564 2008-10-17 5:26 -1 2978.81 ... 17.726 565 2008-10-17 6:01 -1 2894.92 ... 19.210 566 2008-10-17 6:07 -1 2944.92 ... 22.918 </pre> <p>1567 rows x 181 columns]</p>
변화 없음	변화 없음	변화 있음



Step 5를 진행한 Feature들의 모습이다.

데이터 클리닝에 대한 내용

STEP3 ~ 5의 히스토그램 등 변화를 알 수 있는 시각화 데이터

Feature Selection

여기서는 RFE, KBS 내용 설명

DO NOT LEAK THIS DOCUMENT

Sampling

샘플링 소개 (언더샘플링, 단순 오버샘플링, SMOTE, ADASYN 등 알고리즘)

SMOTE (KNN기반)

ADASYN(SMOTE + @)

Cost-Sensitive Learning (가중치 주는거...) Xgboost 사용할 때도 weight 으로 지정 가능...

오버샘플링 SMOTE 랜덤 100회 평균 (test_size = 0.2, LogisticRegression)

acc, pre, rec, f1, roc_curve

```
RFE_MMS_ALL
0.8272 0.8061 0.8624 0.8332 0.8947
```

```
KBS_MMS_ALL
0.8112 0.7843 0.8595 0.8201 0.8804
```

```
RFE_STD_ALL
0.8264 0.8065 0.8596 0.8321 0.8941
```

```
KBS_STD_ALL
0.8112 0.783 0.862 0.8205 0.8817
```

MMS / STD간 큰 차이를 보이지 않았음.

여기서 Fail 데이터에 맞추어 다운 샘플링한 결과도 좀 비교하고 싶은데...

시간이 되면 해보겠습니다.

Prediction accuracy

Logistic Regression

Decision Tree

Random Forest

Boosting

DO NOT LEAK THIS DOCUMENT