




Kaggle Competition

Boston Housing
Prediction

김대관
김정현
김희아
남유선



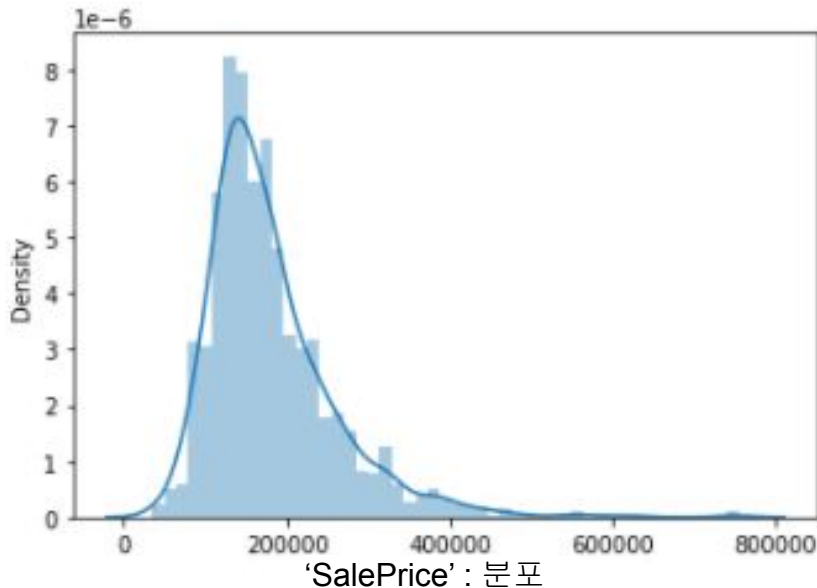
Index

- Data 분석 및 전처리
- Model 및 결과 분석

Data 분석 - 숫자형

총 38개의 int, float 자료형 feature

index인 'Id'와 우리가 예측하고자 하는 y label인 'SalePrice'를 제외하면 총 36개의 숫자형 feature



Data 분석 - 숫자형

```
1 #일단 변수간 관련성 보자
2 cor = train.corr()
3 (cor['SalePrice']).sort_values(ascending=False)
```

OverallQual	0.790982
GrLivArea	0.708624
GarageCars	0.640409
GarageArea	0.623431
TotalBsmtSF	0.613581
1stFlrSF	0.605852
FullBath	0.560664
TotRmsAbvGrd	0.533723
YearBuilt	0.522897
YearRemodAdd	0.507101
GarageYrBlt	0.486362
MasVnrArea	0.477493
Fireplaces	0.466929
BsmtFinSF1	0.386420
LotFrontage	0.351799
WoodDeckSF	0.324413
2ndFlrSF	0.319334
OpenPorchSF	0.315856

‘OverallQual’, ‘GrLivArea’,
‘GarageCars’, ‘GarageArea’,
‘TotalBsmtSF’ 등의 순으로 판매
가격과 상관관계가 높았습니다.

Data 분석 - 숫자형

```
1 # 일단 index, SalePrice 빼주기
2 num_var = train[num_var].drop('Id', axis=1)
3
4 num_var = num_var.drop('SalePrice', axis=1)

1 from sklearn.pipeline import Pipeline
2 from sklearn.preprocessing import StandardScaler
3
4 num_pipeline = Pipeline([
5     ('imputer', SimpleImputer(strategy="median")),
6     ('std_scaler', StandardScaler()),
7 ])
8
9 num_tr = num_pipeline.fit_transform(num_var)
```

결측치는 중간값으로 넣고,
standardscaler로 scaling하여 모델에
포함하였습니다.

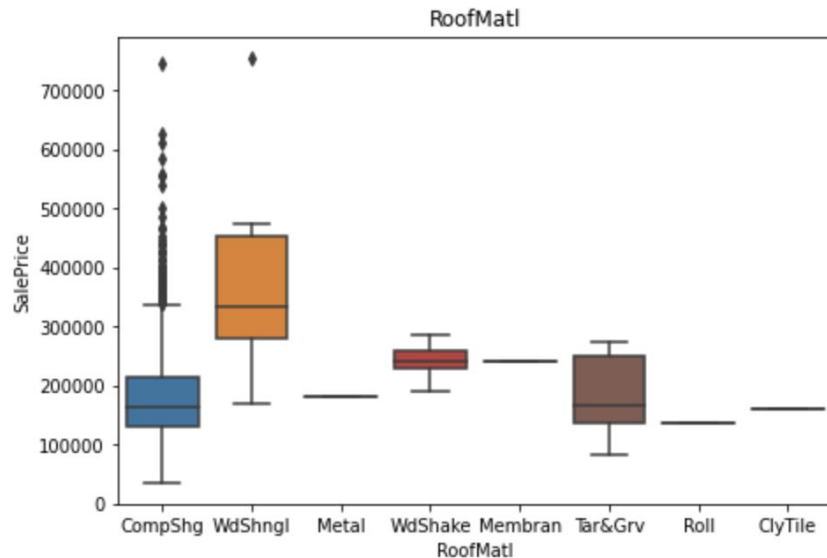
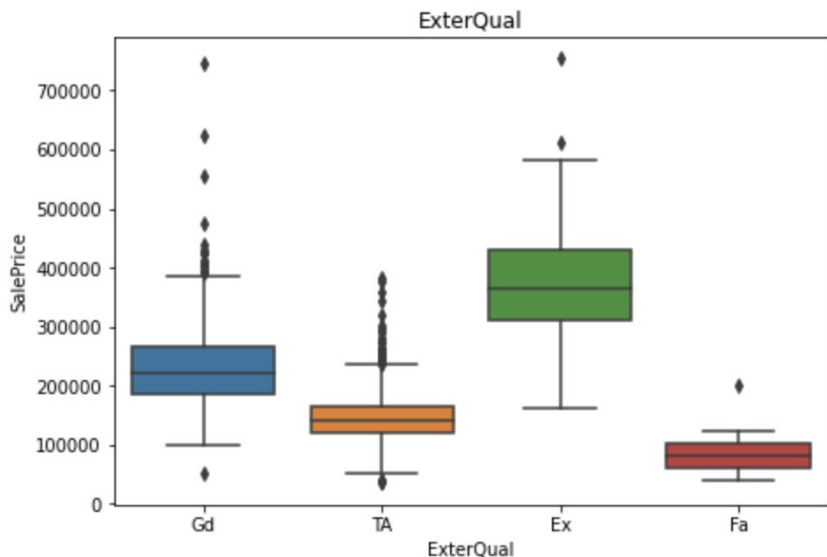
Data 분석 - 카테고리형

- 총 43개의 Column 존재

'MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'LandSlope',
'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl',
'Exterior1st', 'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating', 'HeatingQC',
'CentralAir', 'Electrical', 'KitchenQual', 'Functional', 'FireplaceQu', 'GarageType',
'GarageFinish', 'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence', 'MiscFeature',
'SaleType', 'SaleCondition'

Data 분석 - 카테고리형

- 대체적으로 카테고리의 값들이 **SalePrice**의 평균값에 따라 다름을 알 수 있다.
- 따라서 **SalePrice**의 평균이 큰 값에 대해 큰 숫자를 부여하는 방식으로 결정



Data 분석 - 카테고리형

1. 카테고리 값에 대해서 평균을 기준으로 정렬

```
grades = [[df[df[column] == v]['SalePrice'].mean(), v] for v in uniques]
grades.sort()
```

2. 평균 값이 작은 값 기준으로 mapping ex) 0, 1, 2, ...

- 평균이 가장 작은 값에 대해서 0, 그 다음 값은 1로 바꿔줌

```
mapping = {v: i for i, (_, v) in enumerate(grades)}
df[column] = df[column].map(mapping)
test_df[column] = test_df[column].map(mapping)
```


Data 분석 - 결측치

PoolQC	1453
MiscFeature	1406
Alley	1369
Fence	1179
FireplaceQu	690
LotFrontage	259
GarageType	81
GarageCond	81
GarageFinish	81
GarageQual	81
GarageYrBlt	81
BsmtFinType2	38
BsmtExposure	38
BsmtQual	37
BsmtCond	37
BsmtFinType1	37
MasVnrArea	8
MasVnrType	8
Electrical	1

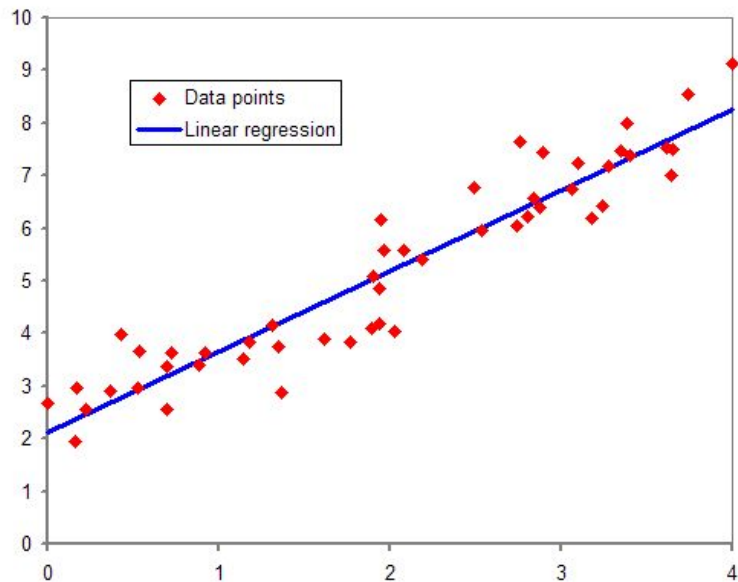
총 19개의 Column에서 결측치 관찰

Data 분석 - 결측치

PoolQC	1453
MiscFeature	1406
Alley	1369
Fence	1179
FireplaceQu	690
LotFrontage	259
GarageType	81
GarageCond	81
GarageFinish	81
GarageQual	81
GarageYrBlt	81
BsmtFinType2	38
BsmtExposure	38
BsmtQual	37
BsmtCond	37
BsmtFinType1	37
MasVnrArea	8
MasVnrType	8
Electrical	1

- 맨 마지막 Electrical Column의 결측치는 단 하나로, 해당 행만 Drop함
- Text형 데이터의 결측치에는 'None'이란 값으로 대체
- Numerical형 데이터의 결측치에는 0의 값으로 대체

Model



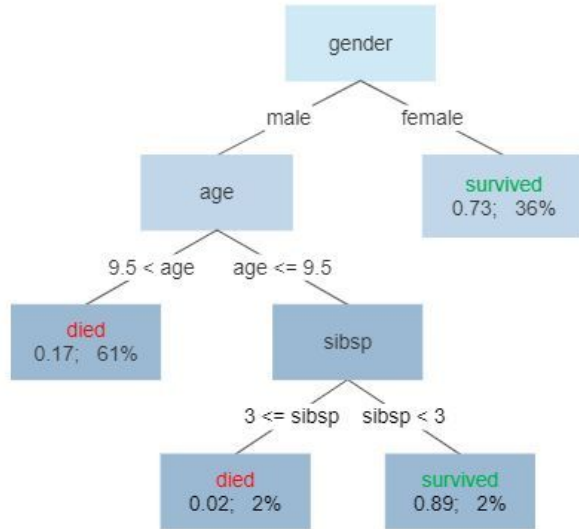
Linear Regression

RMSE : 0.1327

R^2 Score : 0.89776

Model

Survival of passengers on the Titanic



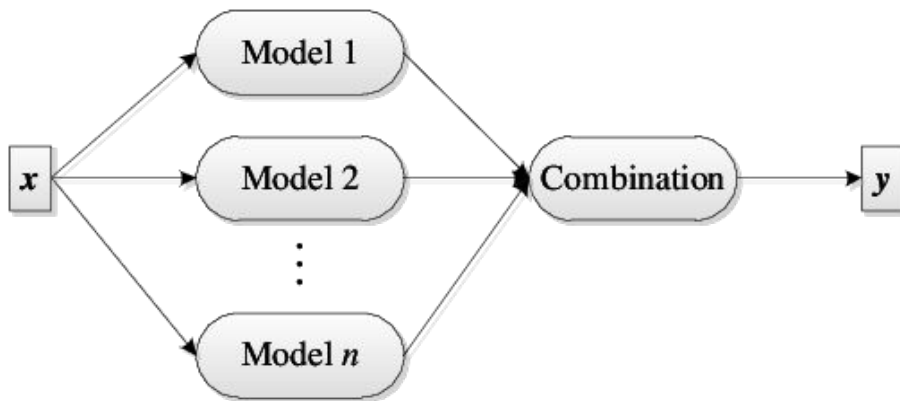
Decision Tree Regression

RMSE : 0.19047

R^2 Score : 0.78936

Model

XGBoost Regression



RMSE : 0.22195

R^2 Score : 0.90879



Thank You!

