
House Prices : Advanced Regression Technique

2020년 6월 17일

20141591 최기용

20141594 최승훈

20161584 민현홍

개요

주어진 집의 정보를 통해 집의 가격을 예측하는 머신러닝 모델을 만들고, 그 정확도를 높이도록 학습시킨다. 이 때, 집의 정보가 담겨있는 Data Set은 Kaggle Knowledge Competition 항목인 House Prices에서 제공하는 Data Set을 사용한다.

목표

1. 주어진 Data set에서 적절한 Feature Engineering을 적용한다.
2. 적절한 Advanced Regression Techinques를 적용한다. (예: random forest)
3. 위 두 목표를 진행할 때 다양한 기법을 사용해보며 정확도를 높인다.

설명

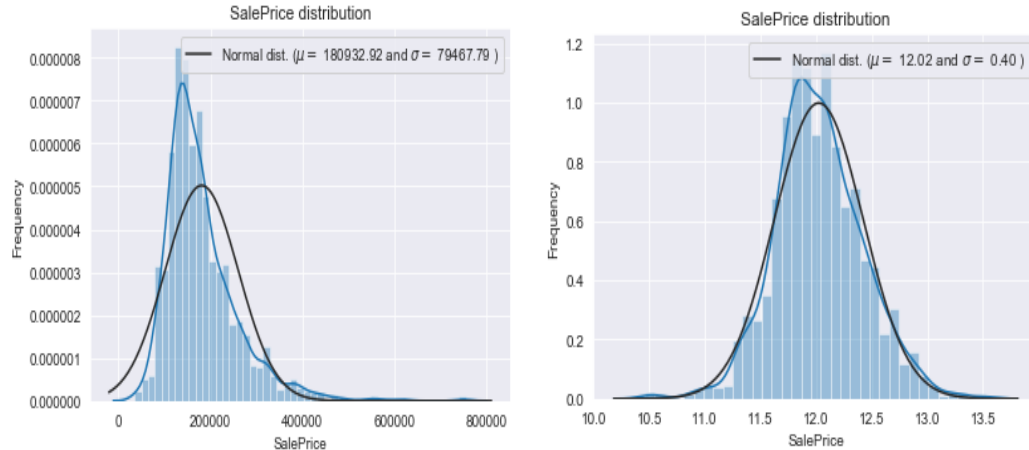
Kaggle Knowledge Competition 항목인 House Prices를 주제로 선정하였다. Data set은 해당 competition에서 제공하는 Data Set을 사용한다.

Test Data set : 1459가구 각각의 주택 관련 80개의 컬럼변수로 이루어져있다.

Train Data Set: 1460가구 각각의 주택 관련 81개(Price 추가)의 컬럼변수로 이루어져있다.

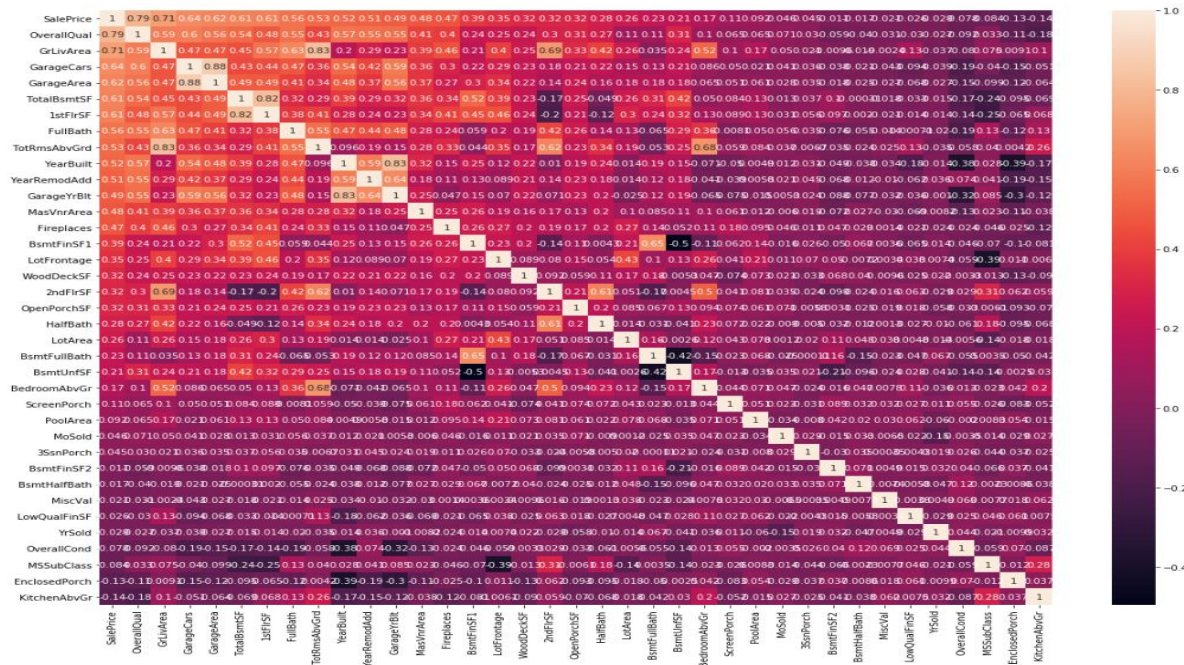
데이터분석

1. 타겟 변수 확인



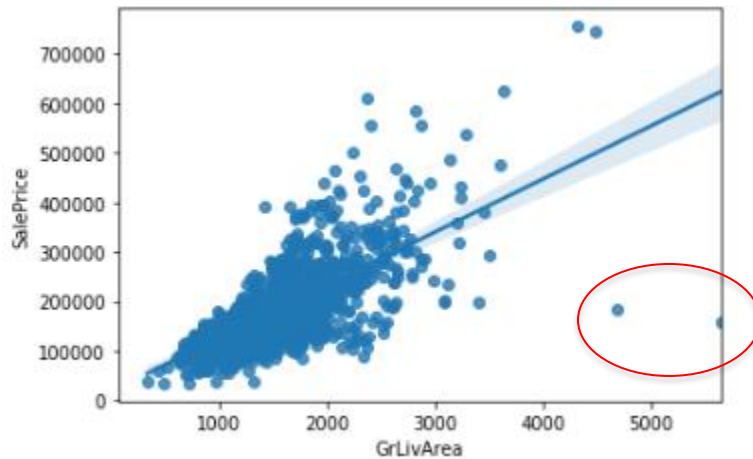
SalePrice의 값이 너무 크고, Right Skewed한 분포를 띄고 있어, 변수에 Log를 씌워 이를 정규분포에 가깝게 만들어 해결.

2. 변수 간 상관관계 확인



- Target Variable 'SalePrice'와 상관관계 상위 40개 변수 Heatmap 그래프

3. Outlier 제거



상관 관계와 전혀 다른 결과를 보여주고 있는 데이터(Outlier)들을 제거 하여 성능 향상

4. Missing Value

```
cols=list(all_data)
for col in list(all_data):
    if (all_data[col].isnull().sum())==0:
        cols.remove(col)
    else:
        pass
print(len(cols))
print(cols)
```

34

['MSZoning', 'LotFrontage', 'Alley', 'Utilities', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'MasVnrArea', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Electrical', 'BsmtFullBath', 'BsmtHalfBath', 'KitchenQual', 'Functional', 'FireplaceQu', 'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual', 'GarageCond', 'PoolQC', 'Fence', 'MiscFeature', 'SaleType']

- 34개의 Missing Value 포함하는 Feature 확인.

```
for col in ('PoolQC', 'MiscFeature', 'Alley', 'Fence', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond', 'BsmtQual'):
    all_data[col] = all_data[col].fillna('None')

for col in ('GarageYrBlt', 'GarageArea', 'GarageCars', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'BsmtFullBath', 'BsmtHalfBath'):
    all_data[col] = all_data[col].fillna(0)

for col in ('MSZoning', 'Electrical', 'KitchenQual', 'Exterior1st', 'Exterior2nd', 'SaleType', 'Functional', 'Utilities'):
    all_data[col] = all_data[col].fillna(all_data[col].mode()[0])

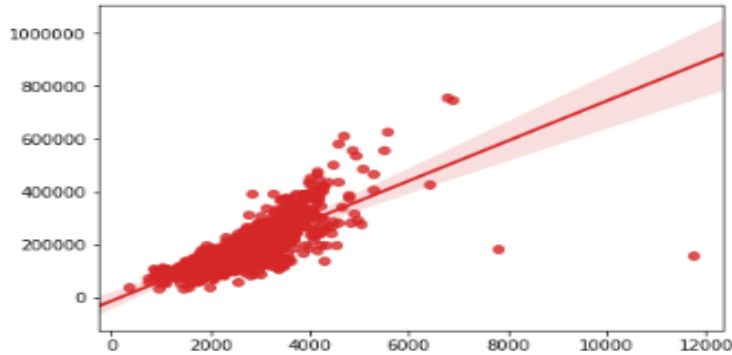
print(f"Total count of missing values in all_data : {all_data.isnull().sum().sum()}")
```

Total count of missing values in all_data : 0

- Categorical Column의 경우 'None', Numeric Column의 경우 0, 해당 Column이 아예 없다고 보기 힘든 경우에는 임의의 값(최빈값 사용)으로 Missing Value를 채운다.

5. 예측에 도움이 되는 새로운 Feature 추가

- TotalBsmtSF, 1stFlrSF, 2ndFlrSF를 모두 합한 'TotalSF' 변수 추가.



- 총 면적과 SalePrice 상관관계 표현 그래프

추가로, BsmtFullBath, FullBath, BsmtHalfBath, HalfBath를 모두 합한 TotalBath 변수 추가

6. 수치형 변수 및 비정상 변수 가공

```
all_data['MSSubClass'] = all_data['MSSubClass'].astype(str)
all_data['MoSold'] = all_data['MoSold'].astype(str)
all_data['YrSold'] = all_data['YrSold'].astype(str)
```

- 연도, 달 과 같은 Numeric 변수이지만 의미 상 Categorical 데이터 인 것들 변환

```
all_data = all_data.drop(columns=['Street', 'Utilities', 'Condition2', 'RoofMat1', 'Heating'])
```

```
all_data = all_data.drop(columns=['PoolArea', 'PoolQC'])
```

```
all_data = all_data.drop(columns=['MiscVal', 'MiscFeature'])
```

- 하나의 값의 비율이 너무 높은 변수들 삭제

전처리

Categorical 변수들에 대해 One-hot Encoding

```
non_numeric=all_data.select_dtypes(np.object)

def onehot(col_list):
    global all_data
    while len(col_list) !=0:
        col=col_list.pop(0)
        data_encoded=pd.get_dummies(all_data[col], prefix=col)
        all_data=pd.merge(all_data, data_encoded, on='id')
        all_data=all_data.drop(columns=col)
```

이 후, numeric 변수들에 대해 Right-skewed인 변수들 log통한 표준화

모델 학습

```
model_lasso = LassoCV(alphas = [1, 0.1, 0.001, 0.0005]).fit(Xtrain, Ytrain)
lasso_preds = np.exp1(model_lasso.predict(Xtest))

model_xgb = xgb.XGBRegressor(n_estimators=360, max_depth=2, learning_rate=0.1)
model_xgb.fit(Xtrain, Ytrain)
xgb_preds = np.exp1(model_xgb.predict(Xtest))

model_ridge = RidgeCV(alphas = [1, 0.1, 0.001, 0.0005]).fit(Xtrain, Ytrain)
ridge_preds = np.exp1(model_ridge.predict(Xtest))

elastic_model = ElasticNetCV(alphas = [1, 0.1, 0.001, 0.0005]).fit(Xtrain, Ytrain)
elastic_preds = np.exp1(elastic_model.predict(Xtest))

model = lgb.LGBMRegressor(objective='regression',num_leaves=5,
                           learning_rate=0.01, n_estimators=5000,
                           max_bin = 55, bagging_fraction = 0.8,
                           bagging_freq = 5, feature_fraction = 0.2319,
                           feature_fraction_seed=9, bagging_seed=9,
                           min_data_in_leaf =6, min_sum_hessian_in_leaf = 11)

lgbm_fit = model.fit(Xtrain, Ytrain)



lgb_preds = np.exp1(lgbm_fit.predict(Xtest))

preds = 0.2*lasso_preds + 0.2*xgb_preds+0.2*lgb_preds+0.2*ridge_preds+0.2*elastic_preds
```

5가지 모델

Lasso, XGBoost, Ridge, Elastic, LGBM 사용하여 학습 후 각각에 같은 비율을 적용하여, 예측

결과

600	ssapgosu	  	0.11745	7	3h
-----	----------	--	---------	---	----

현재 Kaggle 'House price' Competition에 600/5386 순위에 rank되어있다.

기타 데이터 분석에서 Categorical 변수들에 SalePrice에 영향을 미치는 정도에 따라 가중치를 더 부여하는 방법, Test 데이터에 모델 적용 시, 모델 별 가중치를 다르게 하는 방법 등을 통해 성능을 더 개선할 수 있을 것이라 예상된다.

프로젝트 공헌도

최기용 : 30%

최승훈 : 30%

민현홍 : 40%
