6주차 ML/DL 스터디
(basic)

2020077329
김남호
ML/DL core

Kaggle - House Price 코드리뷰 & 성능개선

1. 코드리뷰

- 데이터 분석 및 전처리
  - (이미 데이터 수집은 완료)
- 모델 선정 및 학습(선형회귀모델)
- 모델 평가

House price 예측

- 주어진 여러 개의 데이터 set을 이용해서 주택의 가격을 예측해 보자.

데이터 분석 및 전처리

- 데이터 수집 이미 완료.
- 결측치 존재 항목 제거
  col (81 -> 62)

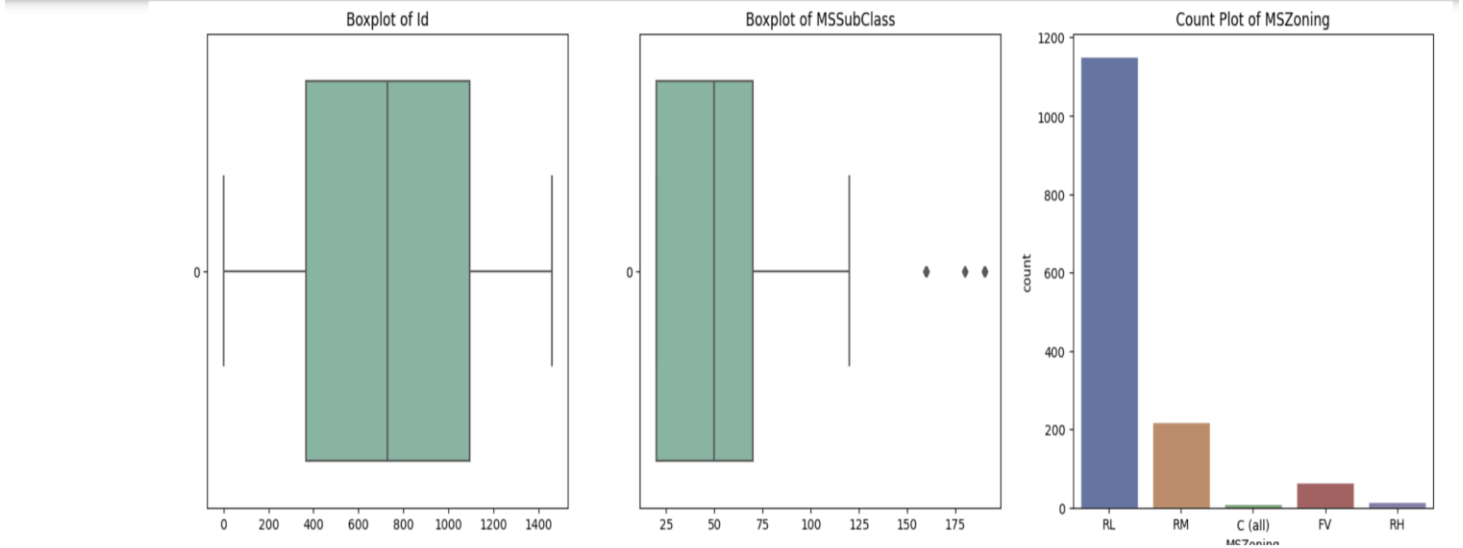
```
In [8]: df.isnull().sum()

Out[8]: Id               0
        MSSubClass       0
        MSZoning         0
        LotFrontage    259
        LotArea          0
                      ...
        MoSold           0
        YrSold           0
        SaleType         0
        SaleCondition    0
        SalePrice        0
        Length: 81, dtype: int64
```

데이터 분석 및 전처리

- Outlier 처리, 제거

- Boxplot 이용
  (이상치 제거에 용이)

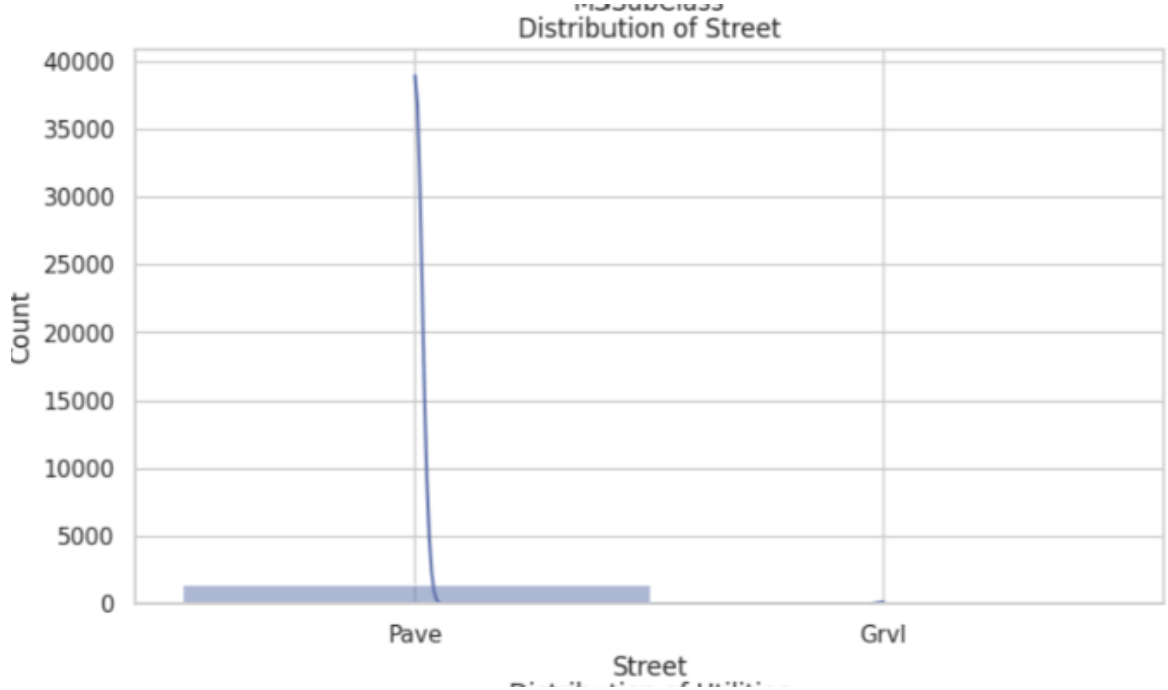
```
In [15]: fig, axes = plt.subplots(21, 3, figsize=(20, 150))
for i, (column_name, column_data) in enumerate(ptr_df.items()):
    sns.set(style="whitegrid")
    row = i // 3
    col = i % 3
    if ptr_df[column_name].dtype == 'int64' or ptr_df[column_name].dtype == 'float64':
        sns.boxplot(data=column_data, orient="h", palette="Set2", ax=axes[row, col])
        axes[row, col].set_title(f'Boxplot of {column_name}')
    else:
        sns.countplot(x=column_name, data=ptr_df, ax=axes[row, col])
        axes[row, col].set_title(f'Count Plot of {column_name}')
plt.show()
```



데이터 분석 및 전처리

- Sub플롯도 이용

- 데이터 정제, 분류에 큰 도움 X



데이터 분석 및 전처리

- 총 24개의 열 column_are_not_helpful_in_prediction 제거.

- 박스플롯을 이용해 outlier 행 모두 제거 (약 300행)

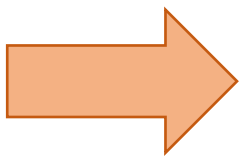
데이터 분석 및 전처리

- 특성 스케일링

- Int64 타입 vs obj 타입

=> 문자열 분류해서 int타입으로 구분.

a	LotShape	LotConfig	Neighborhood	HouseStyle	OverallQ
0	Reg	Inside	CollgCr	2Story	
0	IR1	Inside	CollgCr	2Story	
0	IR1	Corner	Crawfor	2Story	
5	IR1	Inside	Mitchel	1.5Fin	
4	Reg	Inside	Somerst	1Story	
..	...	...	...	...	
0	Reg	Inside	Somerst	1Story	
7	Reg	Inside	Gilbert	2Story	
5	Reg	Inside	NWAmes	1Story	
7	Reg	Inside	NAmes	1Story	
7	Reg	Inside	Edwards	1Story	



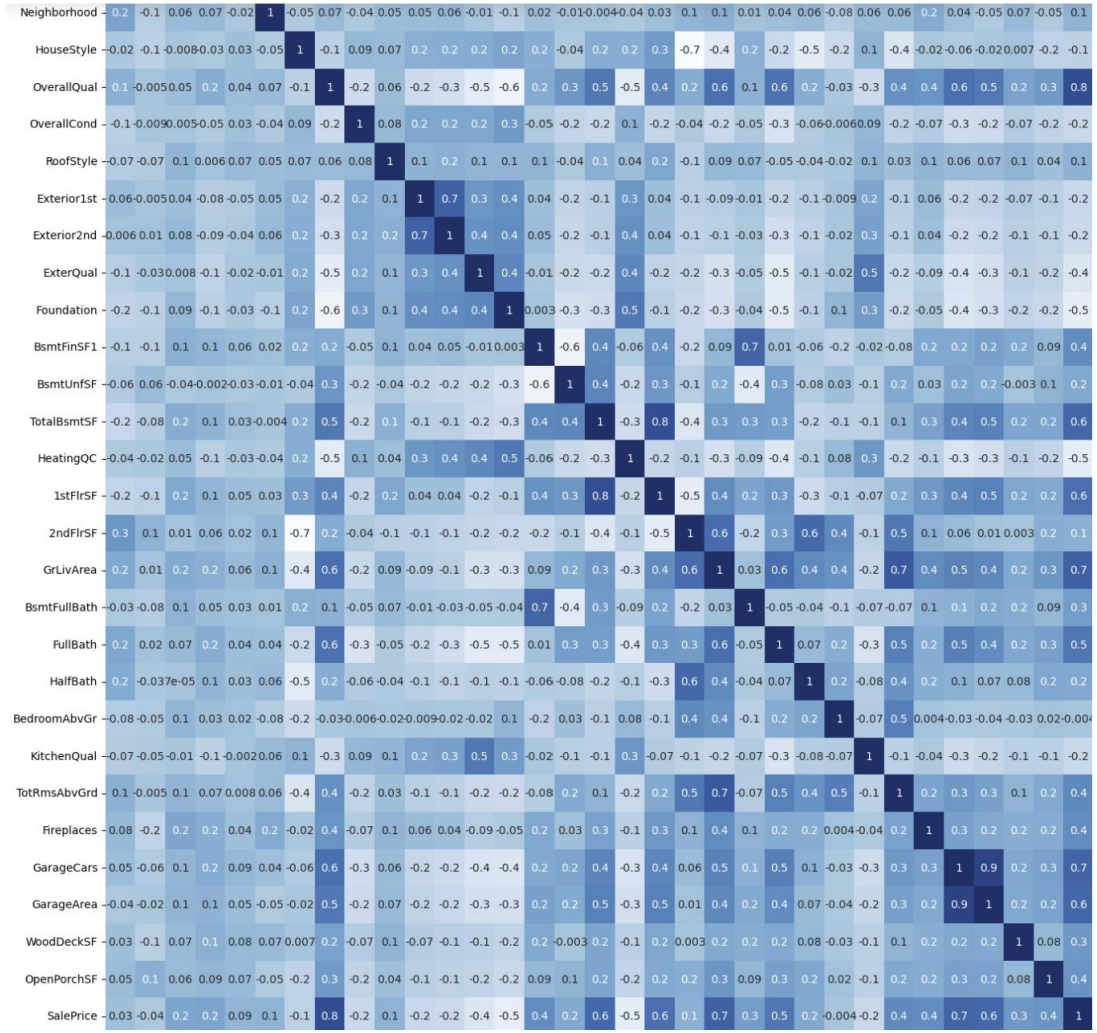
ea	LotShape	LotConfig	Neighborhood	HouseStyle	OverallQ
50	0	0	0	0	
50	1	0	0	0	
50	1	1	1	0	
15	1	0	2	1	
34	0	0	3	2	
...	...	...	...	...	
00	0	0	3	2	
17	0	0	14	0	
75	0	0	4	2	
17	0	0	7	2	
37	0	0	12	2	

데이터 분석 및 전처리

- 그 外 saleprice(Y값), ID 등 input data가 아닌 열 직접 제거
- 총 33 col

데이터 분석 및 전처리

- 상관계수 확인(heatmap)
- Saleprice와 상관계수확인
- 높은 것만 유의미한 데이터
- (21col)



데이터 분석 및 전처리

- VIF지수(다중 공선성) 확인
- VIF 낮은 feature 체크.

	VIF	features
0	39.947857	GarageCars
1	2.875315	HeatingQC
2	15.704668	BsmtFinSF1
3	170.777187	2ndFlrSF
4	68.715139	TotRmsAbvGrd
5	4.900665	HouseStyle
6	54.374619	SalePrice
7	76.980054	TotalBsmtSF
8	3.365545	Exterior1st
9	1141.777385	1stFlrSF
10	64.515602	OverallQual
11	24.102569	FullBath
12	4.084638	Exterior2nd
13	2.989799	KitchenQual
14	3.448951	BsmtFullBath
15	31.928066	GarageArea
16	2.913176	HalfBath
17	3.765041	Foundation
18	23.652166	BsmtUnfSF
19	4.578160	ExterQual
20	1655.578624	GrLivArea

데이터 분석 및 전처리

데이터 전처리 결과 검정 및 확인.

OLS Regression Results

Dep. Variable:	SalePrice	**R-squared (uncentered):**	0.982
Model:	OLS	**Adj. R-squared (uncentered):**	0.981
Method:	Least Squares	**F-statistic:**	2653.
Date:	Wed, 08 Nov 2023	**Prob (F-statistic):**	0.00
Time:	12:07:45	**Log-Likelihood:**	-11740.
No. Observations:	1014	**AIC:**	2.352e+04
Df Residuals:	994	**BIC:**	2.362e+04
Df Model:	20		
Covariance Type:	nonrobust		

모델 선정 및 학습

- 단순 선형 회귀 모델 선정
- 학습(fit)
- 결정 계수 확인

In [68]:
```
r2=ridge_model.score(X,y)
r2
```

Out[68]:
```
0.8460512004265616
```

In [69]:
```
X.shape
```

Out[69]:
```
(1014, 20)
```

In [70]:
```
r2=ridge_model.score(X, y)
n=X.shape[0]
p=X.shape[1]
adj_r2=1-(1-r2)*(n-1)/(n-p-1)
adj_r2
```

Out[70]:
```
0.8429505196697955
```

Test_data 전처리

- test_ data를 학습과정과 동일하게 전처리
- 예측에 사용되는 feature만 선정
- 이후 전처리

  (중간값, 최빈값)

```
def pre_process_test(df_test_sel):
    n_df_t=df_test_sel.select_dtypes(include='number')
    n_cols_t=n_df_t.columns
    for col in n_cols_t:
        df_test_sel.loc[:, col] = df_test_sel[col].fillna(df_test_sel[col].median())  #숫자형의 누락
    s_df_t=df_test_sel.select_dtypes(include='object')
    s_cols_t=s_df_t.columns
    for col in s_cols_t:
        df_test_sel.loc[:, col] = df_test_sel[col].fillna(df_test_sel[col].value_counts().idxmax())
    for col in s_cols_t:
        df_test_sel.loc[:,col]=pd.factorize(df_test_sel[col])[0]
    return df_test_sel

ptr_df_t=pre_process_test(df_test_sel)
ptr_df_t.isnull().sum()
#전처리 이후 누락된 값 있는지 확인
```

모델 예측

- 학습한 선형회귀 모델 이용해서 예측

0.53875

성능 개선

- 내가 생각한 낮은 성능의 원인
  - 데이터 전 처리 과정에서 outlier를 단순 배제하는 것.
    => 가중치 부여로 반영

  - 단순 선형 회귀 모델자체의 문제
    =>모델 변경 (ridge, Decision tree, ~~SVM~~, NN)

  - 하이퍼 파라미터 튜닝
    =>경험적 기반..?

제출

Submissions

All | Successful | Errors | Recent

Submission and Description	Public Score
submission_NN.csv Complete · 4m ago · 뉴럴넷 적용	**5.56544**
submission_Lasso.csv Complete · 19m ago · LAsso regression is submitted	**0.53878**
submission3.csv Complete · 1h ago · I use Decision tree classification model.	**0.30423**
submission2.csv Complete · 2h ago · it is just edited with ridge regression	**0.53873**
submission.csv Complete · 2h ago · for the test. (it is already uploaded by other people) This is not my code	**0.53875**

추가 학습 필요 내용

- Vif지수 의미 파악
- 데이터 전처리 과정에 능동적으로 개선 및 참여하는 법

발표를 들어주셔서 감사합니다.