Google Developer Student Clubs

ML/DL Study

Code Review

Code Review

Exploring Data

```python
df.head()
```

Python

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	SaleCondition	SalePrice
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2008	WD	Normal	208500
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	5	2007	WD	Normal	181500
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	9	2008	WD	Normal	223500
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2006	WD	Abnorml	140000
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	12	2008	WD	Normal	250000

5 rows × 81 columns

```python
df = df.drop('Id', axis=1)
df.head(3)
```

Python

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	HouseStyle	OverallQual	OverallCond	YearBuilt
0	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	2Story	7	5	2003
1	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fam	1Story	6	8	1976
2	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	2Story	7	5	2001

```python
df = df.dropna(axis=1, how='any')
```

```
MSSubClass         0
MSZoning           0
LotFrontage      259
LotArea            0
Street             0
                 ...
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 80, dtype: int64
```

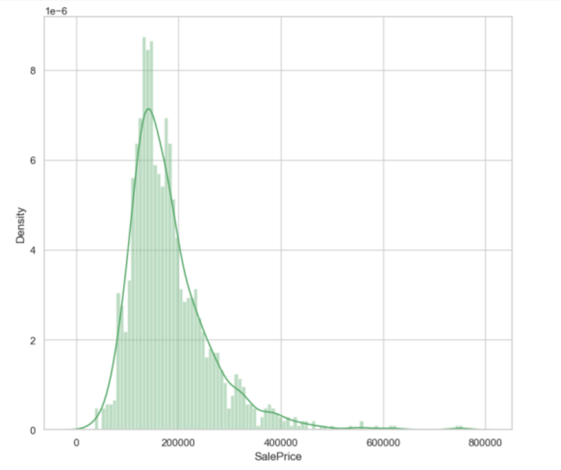
```
MSSubClass       0
MSZoning         0
LotArea          0
Street           0
LotShape         0
                ..
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
SalePrice        0
Length: 61, dtype: int64
```

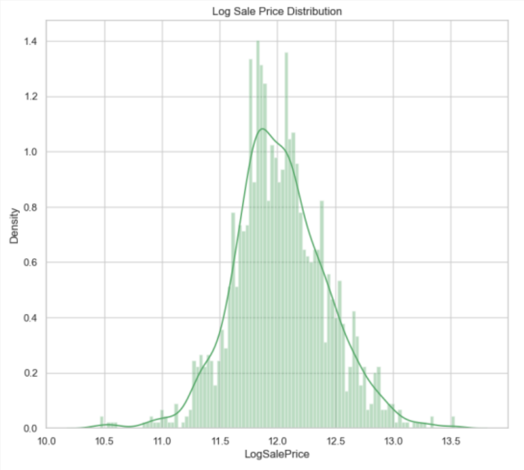
```python
def min_max_scaling(data):
    min_val = data.min()
    max_val = data.max()
    scaled_data = (data - min_val) / (max_val - min_val)
    return scaled_data

df['LogSalePrice'] = min_max_scaling(df['LogSalePrice'])
print(df['LogSalePrice'].describe())
```

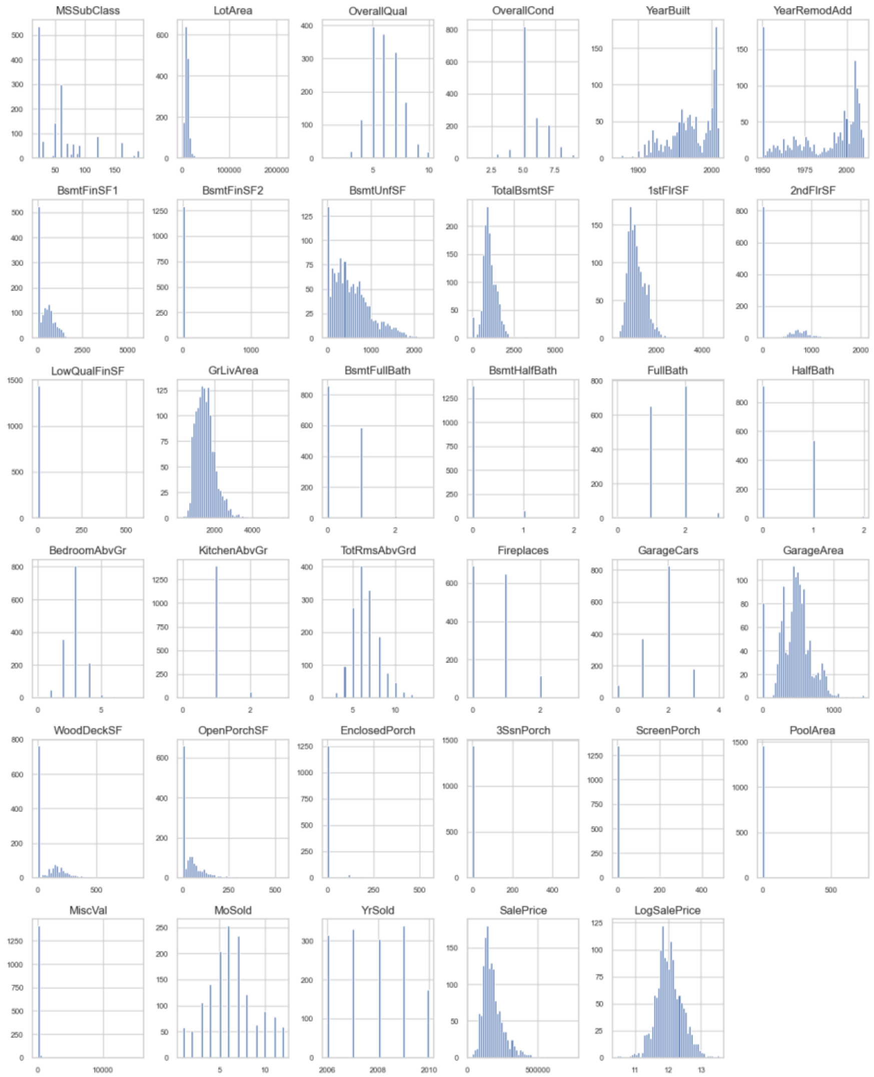
✓ 0.0s Python

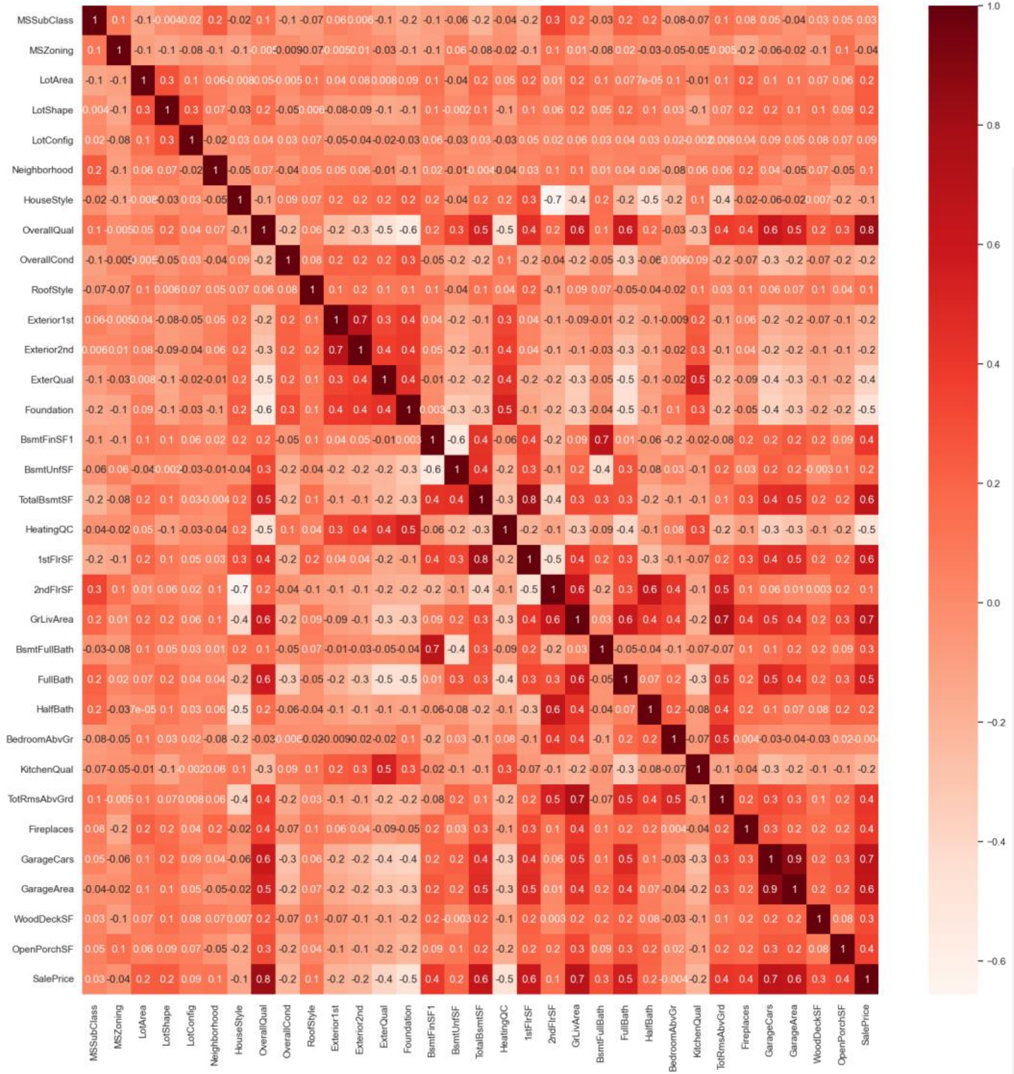
```
count    1460.000000
mean        0.508683
std         0.129936
min         0.000000
25%         0.427702
50%         0.501349
75%         0.589900
max         1.000000
Name: LogSalePrice, dtype: float64
```

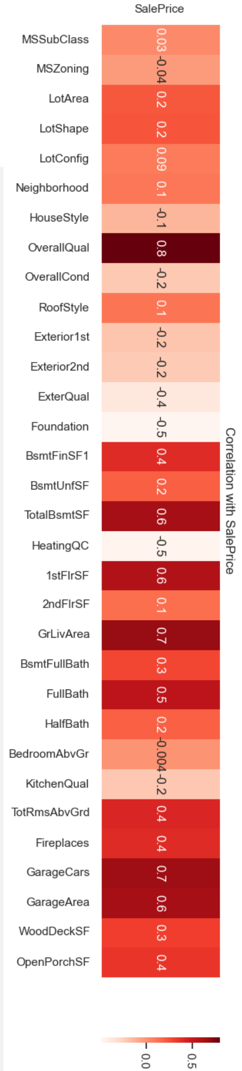
```python
df['LogSalePrice'] = np.log(df['SalePrice'])
```






MSSubClass
LotArea
OverallQual
OverallCond
YearBuilt
YearRemodAdd
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold
YrSold
SalePrice
LogSalePrice


MSSubClass
MSZoning
LotArea
LotShape
LotConfig
Neighborhood
HouseStyle
OverallQual
OverallCond
RoofStyle
Exterior1st
Exterior2nd
ExterQual
Foundation
BsmtFinSF1
BsmtUnfSF
TotalBsmtSF
HeatingQC
1stFlrSF
2ndFlrSF
GrLivArea
BsmtFullBath
FullBath
HalfBath
BedroomAbvGr
KitchenQual
TotRmsAbvGrd
Fireplaces
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
SalePrice
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


SalePrice
MSSubClass 0.03
MSZoning -0.04
LotArea 0.2
LotShape 0.2
LotConfig 0.09
Neighborhood 0.1
HouseStyle -0.1
OverallQual 0.8
OverallCond -0.2
RoofStyle 0.1
Exterior1st -0.2
Exterior2nd -0.2
ExterQual -0.4
Foundation -0.5
BsmtFinSF1 0.4
BsmtUnfSF 0.2
TotalBsmtSF 0.6
HeatingQC -0.5
1stFlrSF 0.6
2ndFlrSF 0.1
GrLivArea 0.7
BsmtFullBath 0.3
FullBath 0.5
HalfBath 0.2
BedroomAbvGr -0.004
KitchenQual -0.2
TotRmsAbvGrd 0.4
Fireplaces 0.4
GarageCars 0.7
GarageArea 0.6
WoodDeckSF 0.3
OpenPorchSF 0.4
Correlation with SalePrice
0.0
0.5

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_train)
mse = mean_squared_error(y_train, y_pred)
r2 = r2_score(y_train, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

✓ 0.0s Python

```
Mean Squared Error: 621206121.4955989
R-squared: 0.8460515711954664
```

Random Forest

```python
from sklearn.ensemble import RandomForestRegressor

model_1 = RandomForestRegressor(n_estimators=100, random_state=0)

model_1.fit(X_train, y_train)

y_pred = model_1.predict(X_train)
mse = mean_squared_error(y_train, y_pred)
r2 = r2_score(y_train, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

✓ 3.5s Python

```
Mean Squared Error: 89644858.08237426
R-squared: 0.9777840485232812
```

Gradient Boosting

```
from sklearn.ensemble import GradientBoostingRegressor

model_2 = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, random_state=0)

model_2.fit(X_train, y_train)

y_pred = model_2.predict(X_train)
mse = mean_squared_error(y_train, y_pred)
r2 = r2_score(y_train, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

✓ 0.8s Python

```
Mean Squared Error: 250821838.44492674
R-squared: 0.9378408766393095
```

XGB

```python
import xgboost as xgb

model_4 = xgb.XGBRegressor(n_estimators=100, learning_rate=0.1, random_state=0)

model_4.fit(X_train, y_train)

y_pred = model_4.predict(X_train)
mse = mean_squared_error(y_train, y_pred)
r2 = r2_score(y_train, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

✓ 0.2s Python

```
Mean Squared Error: 45084063.98391847
R-squared: 0.9888271854151449
```

#	Team	Members	Score	Entries	Last	Join
3781	**goldbabyerim**		0.18915	1	35s	

Your First Entry!
Welcome to the leaderboard!