REGULARIZED REGRESSION

Kwabena Asabere

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```python
from sklearn.linear_model import Ridge,Lasso,RidgeCV,LassoCV,ElasticNetCV,ElasticNet
from sklearn.preprocessing import StandardScaler,OneHotEncoder,PolynomialFeatures,FunctionTra
from sklearn.pipeline import Pipeline,make_pipeline
from sklearn.model_selection import train_test_split,cross_val_score,cross_validate,cross_val
from sklearn.metrics import root_mean_squared_error,mean_squared_error
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
```

```python
housing = pd.read_csv(r"C:\Users\KAsab\Desktop\PYTHON\california_housing.csv",engine = "pyar
```

```python
housing.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	m
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3

```python
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
```

```
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

```
housing.isna().sum()
```

```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

```
housing["total_bedrooms"]=housing["total_bedrooms"].fillna(housing["total_bedrooms"].median(
```

```
housing.isna().sum()
```

```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
total_bedrooms        0
population            0
households            0
```

```
median_income         0
median_house_value    0
ocean_proximity       0
dtype: int64
```

```
housing["rooms_per_house"] = housing['total_bedrooms']/housing["total_rooms"]
housing["people_per_house"] = housing["population"]/housing["households"]
```

```
housing.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	m
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3

```
numeric_features = ["rooms_per_house","people_per_house"]
log_features = housing.iloc[:,:8].columns.to_list()
cat_features = housing.select_dtypes(include = "object").columns.to_list()
```

```
X = housing.drop(columns = ["median_house_value"])
y = housing["median_house_value"]
```

```
X_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16512 entries, 14196 to 15795
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           16512 non-null  float64
 1   latitude            16512 non-null  float64
 2   housing_median_age  16512 non-null  float64
 3   total_rooms         16512 non-null  float64
 4   total_bedrooms      16512 non-null  float64
 5   population          16512 non-null  float64
 6   households          16512 non-null  float64
 7   median_income       16512 non-null  float64
 8   ocean_proximity     16512 non-null  object
 9   rooms_per_house     16512 non-null  float64
 10  people_per_house    16512 non-null  float64
dtypes: float64(10), object(1)
memory usage: 1.5+ MB
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
numeric_transformer = Pipeline( steps = [
    ("imputer",SimpleImputer(strategy = "mean")),
    ("scaler",StandardScaler())
])

log_transformer = Pipeline(steps = [
    ("imputer",SimpleImputer(strategy = "mean")),
    ("log",FunctionTransformer(np.log1p))
])

cat_transformer = Pipeline(steps = [
    ("imputer",SimpleImputer(strategy = "most_frequent")),
    ("onehot",OneHotEncoder(handle_unknown = "ignore",sparse_output = False))
])
```

```
preprocessor = ColumnTransformer(transformers =[
    ("log",log_transformer,log_features),
    ("num",numeric_transformer,numeric_features),
    ("cat",cat_transformer,cat_features)
])
```

```
pipeline = Pipeline( steps = [
    ("preprocessor",preprocessor),
    ("ridge",Ridge())
])
```

```
param_grid = {
    "ridge__alpha":[0.1,1.0,10,100,1000]
}
```

The names of the hyperparameters in the `param_grid` must contain the name of the step in the pipeline followed by 2 underscores and then the hyperparameter.