Build a Feature Engineering Regression Pipeline

Alexander Bonnet

Grand Canyon University

DSC - 530

Brian Stout

7/10/2024

1. Formulate a prediction question that you want to answer by applying regression modeling.

How accurately can I predict the median house value of a neighborhood based on the given factors?

2. Search and locate a dataset that is relevant to the question you created in the previous step. You may search repositories such as Data.gov, UCI Machine Learning, Kaggle, or Scikit-Learn. Find dataset with no less than 10 variables, mostly quantitative.

California Housing Prices dataset from Kaggle. There are 10 factors. 9 numerical and 1 categorical.

3. Explain and describe your dataset's variables. List your dependent and independent variables, and identify which scale is used to measure each variable (interval, ordinal, or nominal).

Dependent –

Median-house-value (Interval): The median value of the houses in the neighborhood.

Independent –

Longitude (Interval): A measure of how far west a house is; a higher value is farther west

Longitude (Interval): A measure of how far north a house is; a higher value is farther north

housingMedianAge (Interval): Median age of a house within a block; a lower number is a newer building

totalRooms (Interval): Total number of rooms within a block

totalBedrooms (Interval): Total number of bedrooms within a block

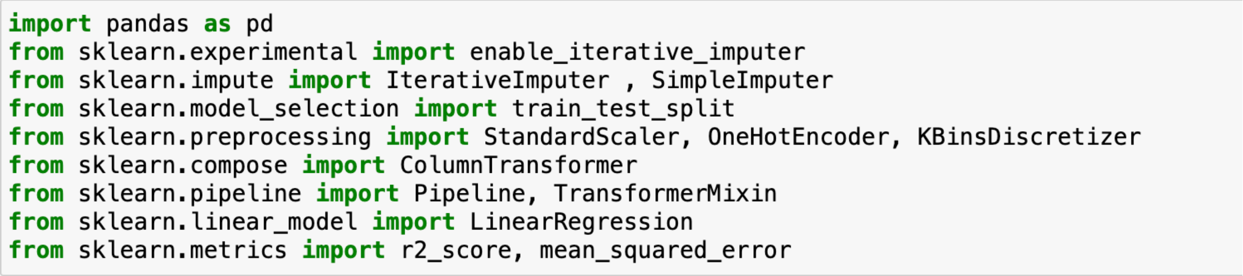
population (Interval): Total number of people residing within a block households (Interval): Total number of households, a group of people residing within a home unit, for a block

medianIncome (Interval): Median income for households within a block of houses (measured in tens of thousands of US Dollars)

medianHouseValue (Interval): Median house value for households within a block (measured in US Dollars)

oceanProximity (nominal): Location of the house w.r.t ocean/sea

4. Import all the necessary libraries and load your dataset into a data frame.



```
import pandas as pd
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer , SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder, KBinsDiscretizer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline, TransformerMixin
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
```

5. Use the feature-engine transformers available in the scikit-learn library to feature engineer your dataset variables.

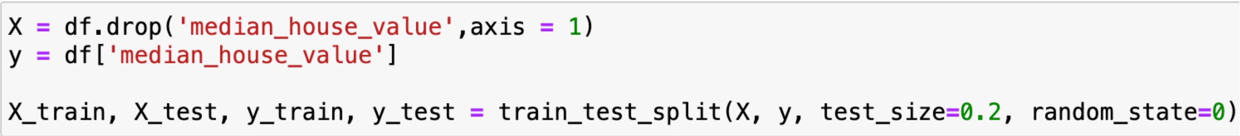
Missing data imputation: For numerical data I used IterativeImputer. This allowed me to match the numerical missing values using regression. This will give a more accurate representation of what the missing value might have been. For categorical variables I chose to use the most common response. There were no missing data points in this variable, but if there I was, I would choose this as it is an easy way to fill the missing responses while choosing the most likely value.

Categorical variable encoding – For encoding I chose to use One Hot Encoding so that I could assign each categorical category to a numerical value. This allows the data to have some form of hierarchy in the categorical variable. Since the categorical variable is the proximity to the ocean ranking them like this will prove beneficial, as the proximity to the ocean tends to lead to higher prices.

Discretization: KBinsDiscretizer was used to slim down to generalize the data. I chose to use 25 bins, since the data has a large range. I used ordinal encoding because the data is based on growing values. I used kmeans as the strategy so that I can group the like data together. When trying to use the different strategies kmeans performed the best.

Variable transformation: I standardized all the factors. This allows me to make all the predictors to be centered around the same value.

6. Split your dataset into training and testing sets.



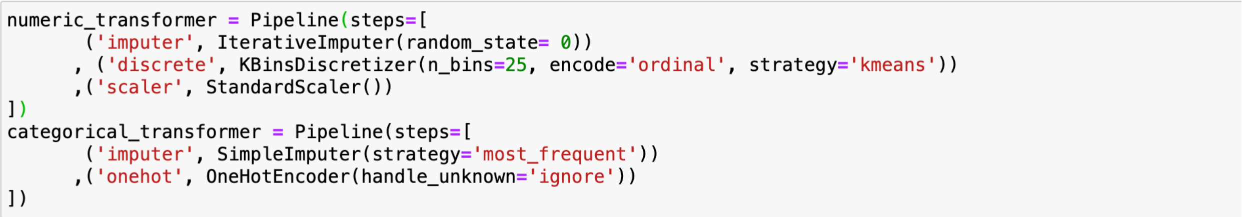
```
X = df.drop('median_house_value',axis = 1)
y = df['median_house_value']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

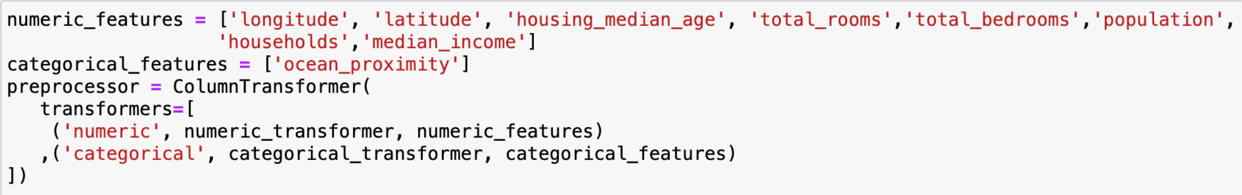
7. Import the Pipeline class from the sklearn.pipleline library

Done

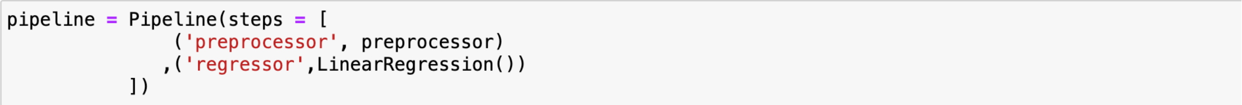
8. Create a pipeline object and pass all the transformers you created in step 5 and a regression model.



```
numeric_transformer = Pipeline(steps=[
       ('imputer', IterativeImputer(random_state= 0))
      , ('discrete', KBinsDiscretizer(n_bins=25, encode='ordinal', strategy='kmeans'))
      ,('scaler', StandardScaler())
])
categorical_transformer = Pipeline(steps=[
       ('imputer', SimpleImputer(strategy='most_frequent'))
      ,('onehot', OneHotEncoder(handle_unknown='ignore'))
])
```

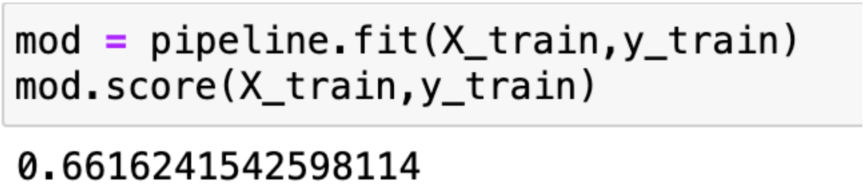


```
numeric_features = ['longitude', 'latitude', 'housing_median_age', 'total_rooms','total_bedrooms','population',
                   'households','median_income']
categorical_features = ['ocean_proximity']
preprocessor = ColumnTransformer(
   transformers=[
    ('numeric', numeric_transformer, numeric_features)
   ,('categorical', categorical_transformer, categorical_features)
])
```



```
pipeline = Pipeline(steps = [
               ('preprocessor', preprocessor)
              ,('regressor',LinearRegression())
           ])
```

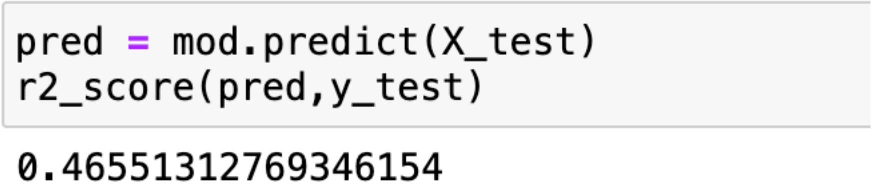
9. Fit the pipeline on the training dataset.



```
mod = pipeline.fit(X_train,y_train)
mod.score(X_train,y_train)
```

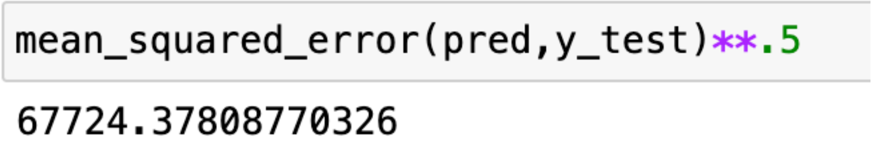
```
0.6616241542598114
```

10. Make predictions and evaluate the performance of your model using the cross-validation technique. Report the RMSE and R2 values and explain the results.



```
pred = mod.predict(X_test)
r2_score(pred,y_test)
```

```
0.46551312769346154
```



```
mean_squared_error(pred,y_test)**.5
```

```
67724.37808770326
```

The model was able to account for 46.55% of the variation in the median house value and had a root mean squared error of 67724 dollars. This means that the model does not perform very well.

Reference

Nugent, C. (2017, November 24). California housing prices. Kaggle. https://www.kaggle.com/datasets/camnugent/california-housing-prices?resource=download