index

January 25, 2022

1 Multiple Linear Regression in Statsmodels - Lab

1.1 Introduction

In this lab, you'll practice fitting a multiple linear regression model on the Ames Housing dataset!

1.2 Objectives

You will be able to: * Determine if it is necessary to perform normalization/standardization for a specific model or set of data * Use standardization/normalization on features of a dataset * Identify if it is necessary to perform log transformations on a set of features * Perform log transformations on different features of a dataset * Use statsmodels to fit a multiple linear regression model * Evaluate a linear regression model by using statistical performance metrics pertaining to overall model and specific parameters

1.3 The Ames Housing Data

Using the specified continuous and categorical features, preprocess your data to prepare for modeling: * Split off and one hot encode the categorical features of interest * Log and scale the selected continuous features

[19]:
```
import pandas as pd
import numpy as np

ames = pd.read_csv('ames.csv')

continuous = ['LotArea', '1stFlrSF', 'GrLivArea', 'SalePrice']
categoricals = ['BldgType', 'KitchenQual', 'SaleType', 'MSZoning', 'Street',
                'Neighborhood']
```

1.4 Continuous Features

[39]:
```
# Log transform and normalize

ames_log_norm = pd.DataFrame([])



def log_norm(data):
```

```
    data_log = np.log(data)

    data_log_norm = (data_log - np.mean(data_log)) / np.std(data_log)

    return data_log_norm



ames_log_norm = ames[continuous].apply(log_norm)

new_name = [item+"_log" for item in continuous]

ames_log_norm.columns = new_name
```

1.5 Categorical Features

[42]:
```
# One hot encode categoricals
ames_cat = pd.DataFrame([])

ames_cat = pd.get_dummies(ames[categoricals], prefix = categoricals,
                                    drop_first = True)
```

1.6 Combine Categorical and Continuous Features

[45]:
```
# combine features into a single dataframe called preprocessed
preprocessed = pd.concat([ames_log_norm, ames_cat], axis = 1)
preprocessed.head()
```

[45]:
```
   LotArea_log  1stFlrSF_log  GrLivArea_log  SalePrice_log  BldgType_2fmCon  \
0    -0.133231     -0.803570       0.529260       0.560068                0
1     0.113442      0.418585      -0.381846       0.212764                0
2     0.420061     -0.576560       0.659675       0.734046                0
3     0.103347     -0.439287       0.541511      -0.437382                0
4     0.878409      0.112267       1.282191       1.014651                0

   BldgType_Duplex  BldgType_Twnhs  BldgType_TwnhsE  KitchenQual_Fa  \
0                0               0                0               0
1                0               0                0               0
2                0               0                0               0
3                0               0                0               0
4                0               0                0               0

   KitchenQual_Gd  …  Neighborhood_NoRidge  Neighborhood_NridgHt  \
0               1  …                     0                     0
1               0  …                     0                     0
2               1  …                     0                     0
```

```
3                1  …                    0                     0
4                1  …                    1                     0

   Neighborhood_OldTown  Neighborhood_SWISU  Neighborhood_Sawyer  \
0                     0                   0                    0
1                     0                   0                    0
2                     0                   0                    0
3                     0                   0                    0
4                     0                   0                    0

   Neighborhood_SawyerW  Neighborhood_Somerst  Neighborhood_StoneBr  \
0                     0                     0                     0
1                     0                     0                     0
2                     0                     0                     0
3                     0                     0                     0
4                     0                     0                     0

   Neighborhood_Timber  Neighborhood_Veenker
0                    0                     0
1                    0                     1
2                    0                     0
3                    0                     0
4                    0                     0

[5 rows x 48 columns]
```

1.7 Run a linear model with SalePrice as the target variable in statsmodels

[53]:
```python
# Your code here
import statsmodels.api as sm

predictors = list(preprocessed.columns)
predictors.remove("SalePrice_log")
```

[57]:
```python
X = preprocessed[predictors]
X_int = sm.add_constant(X)
Y = preprocessed["SalePrice_log"]
```

[60]:
```python
model = sm.OLS(Y,X_int).fit()
model.summary()
```

[60]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:          SalePrice_log   R-squared:                       0.839
Model:                            OLS   Adj. R-squared:                  0.834
```

```
Method:                 Least Squares   F-statistic:                     156.5
Date:                Tue, 25 Jan 2022   Prob (F-statistic):               0.00
Time:                        00:49:11   Log-Likelihood:                -738.64
No. Observations:                1460   AIC:                             1573.
Df Residuals:                    1412   BIC:                             1827.
Df Model:                          47
Covariance Type:            nonrobust
================================================================================
========
                           coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--------
const                   -0.1317      0.263     -0.500      0.617      -0.648
0.385
LotArea_log              0.1033      0.019      5.475      0.000       0.066
0.140
1stFlrSF_log             0.1371      0.016      8.584      0.000       0.106
0.168
GrLivArea_log            0.3768      0.016     24.114      0.000       0.346
0.407
BldgType_2fmCon         -0.1715      0.079     -2.173      0.030      -0.326
-0.017
BldgType_Duplex         -0.4205      0.062     -6.813      0.000      -0.542
-0.299
BldgType_Twnhs          -0.1404      0.093     -1.513      0.130      -0.322
0.042
BldgType_TwnhsE         -0.0512      0.060     -0.858      0.391      -0.168
0.066
KitchenQual_Fa          -1.0002      0.088    -11.315      0.000      -1.174
-0.827
KitchenQual_Gd          -0.3822      0.050     -7.613      0.000      -0.481
-0.284
KitchenQual_TA          -0.6695      0.055    -12.111      0.000      -0.778
-0.561
SaleType_CWD             0.2286      0.215      1.061      0.289      -0.194
0.651
SaleType_Con             0.5863      0.304      1.927      0.054      -0.010
1.183
SaleType_ConLD           0.3152      0.155      2.029      0.043       0.010
0.620
SaleType_ConLI           0.0331      0.195      0.169      0.865      -0.350
0.416
SaleType_ConLw           0.0161      0.196      0.082      0.935      -0.368
0.400
SaleType_New             0.3000      0.079      3.803      0.000       0.145
0.455
```

SaleType_Oth	0.1179	0.246	0.480	0.631	-0.364	0.599
SaleType_WD	0.1749	0.065	2.676	0.008	0.047	0.303
MSZoning_FV	1.0670	0.193	5.526	0.000	0.688	1.446
MSZoning_RH	0.8771	0.194	4.512	0.000	0.496	1.258
MSZoning_RL	0.9964	0.162	6.151	0.000	0.679	1.314
MSZoning_RM	1.1027	0.152	7.264	0.000	0.805	1.400
Street_Pave	-0.2132	0.180	-1.182	0.237	-0.567	0.141
Neighborhood_Blueste	0.0530	0.318	0.167	0.868	-0.571	0.677
Neighborhood_BrDale	-0.4629	0.171	-2.711	0.007	-0.798	-0.128
Neighborhood_BrkSide	-0.6500	0.137	-4.735	0.000	-0.919	-0.381
Neighborhood_ClearCr	-0.2103	0.144	-1.456	0.146	-0.494	0.073
Neighborhood_CollgCr	-0.0761	0.119	-0.641	0.522	-0.309	0.157
Neighborhood_Crawfor	-0.0824	0.129	-0.638	0.523	-0.336	0.171
Neighborhood_Edwards	-0.7615	0.124	-6.143	0.000	-1.005	-0.518
Neighborhood_Gilbert	-0.0980	0.126	-0.777	0.437	-0.346	0.150
Neighborhood_IDOTRR	-0.9622	0.160	-6.014	0.000	-1.276	-0.648
Neighborhood_MeadowV	-0.6921	0.159	-4.351	0.000	-1.004	-0.380
Neighborhood_Mitchel	-0.2554	0.131	-1.944	0.052	-0.513	0.002
Neighborhood_NAmes	-0.4408	0.120	-3.664	0.000	-0.677	-0.205
Neighborhood_NPkVill	-0.0160	0.173	-0.092	0.927	-0.356	0.324
Neighborhood_NWAmes	-0.2677	0.126	-2.122	0.034	-0.515	-0.020
Neighborhood_NoRidge	0.3633	0.133	2.737	0.006	0.103	0.624
Neighborhood_NridgHt	0.3627	0.120	3.029	0.002	0.128	0.598
Neighborhood_OldTown	-0.9354	0.140	-6.686	0.000	-1.210	

```
-0.661
Neighborhood_SWISU      -0.7000      0.144     -4.845      0.000      -0.983
-0.417
Neighborhood_Sawyer     -0.4756      0.128     -3.727      0.000      -0.726
-0.225
Neighborhood_SawyerW    -0.2332      0.125     -1.860      0.063      -0.479
0.013
Neighborhood_Somerst     0.0951      0.145      0.658      0.511      -0.188
0.379
Neighborhood_StoneBr     0.4297      0.133      3.232      0.001       0.169
0.691
Neighborhood_Timber      0.0057      0.134      0.042      0.966      -0.257
0.269
Neighborhood_Veenker     0.1277      0.169      0.754      0.451      -0.204
0.460
==============================================================================
Omnibus:                      289.988   Durbin-Watson:                   1.967
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1242.992
Skew:                          -0.886   Prob(JB):                    1.22e-270
Kurtosis:                       7.159   Cond. No.                         109.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

1.8 Run the same model in scikit-learn

[69]:

```python
# Your code here - Check that the coefficients and intercept are
# the same as those from Statsmodels

from sklearn.linear_model import LinearRegression

model_sk = LinearRegression()
model_sk.fit(X,Y)
print("Model Coefficients: ", model_sk.coef_)
print("\nModel Intercept: ", model_sk.intercept_)
```

```
Model Coefficients:  [ 0.10327192  0.1371289   0.37682133 -0.17152105
-0.42048287 -0.14038921
 -0.05121949 -1.00020261 -0.38215288 -0.6694784   0.22855565  0.58627941
  0.31521364  0.03310544  0.01609215  0.29995612  0.1178827   0.17486316
  1.06700108  0.8771105   0.99643261  1.10266268 -0.21318409  0.0529509
 -0.46287108 -0.65004527 -0.21026441 -0.0761186  -0.08236455 -0.76152767
 -0.09803299 -0.96216285 -0.6920628  -0.25540919 -0.4408245  -0.01595592
 -0.26772132  0.36325607  0.36272091 -0.93537011 -0.70000301 -0.47559431
```

```
 -0.23317719  0.09506225  0.42971796  0.00569435  0.12766986]

Model Intercept:  -0.1317424941874447
```

1.9 Predict the house price given the following characteristics (before manipulation!!)

Make sure to transform your variables as needed!

- LotArea: 14977
- 1stFlrSF: 1976
- GrLivArea: 1976
- BldgType: 1Fam
- KitchenQual: Gd
- SaleType: New
- MSZoning: RL
- Street: Pave
- Neighborhood: NridgHt

1.10 Summary

Congratulations! You pre-processed the Ames Housing data using scaling and standardization. You also fitted your first multiple linear regression model on the Ames Housing data using statsmodels and scikit-learn!