Using Multiple Linear Regression to predict Total Charge of a hospital stay

Introduction

Research question

The of our research will be to build a model for predicting the total charge of a patient's stay at a hospital. We will look at patient demographics and health conditions to make this prediction.

We're making the assumption that the data we are using is clean and valid.

Multiple Linear regression is an appropriate technique for this question because we are looking to predict a numerical outcome using many contributing variables.

Tools

Python in Jupyterlab was used to write the code for this analysis. The code can be found in the notebook `multiple-linear-regression.ipynb` or in script form in `multiple-linear-regression.py`.

Libraries

`Numpy` and `pandas` were used for standard dataframe and numerical operations. `Sklearn` and `statsmodels` were used for building the regression models. `matplotlib` and `yellowbrick` were used for visualizations.

Data Preparation

We begin the data preparation by removing irrelevant variables. We removed the following:

- CaseOrder
- Customer_id
- Interaction
- UID
- City
- State
- County
- Zip
- Lat
- Lng
- Interaction
- TimeZone
- Additional_charges

```
df = df.drop(['CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City',
'State', 'County', 'Zip', 'Lat', 'Lng', 'Interaction', 'TimeZone',
'Additional_charges'], axis=1)
```

We then convert the categorical values to numeric.

```
cat_columns = df.select_dtypes(exclude="number").columns

# Give categorical columns a numeric value
for col in cat_columns:
    df[col] = pd.Categorical(df[col])
    df[col] = df[col].cat.codes
```

Our target variable (`y`) is `TotalCharge`, and our initial set of predictor variables (`X`) are:

- Population
- Area
- Job
- Children
- Age
- Income
- Marital
- Gender
- ReAdmis
- VitD_levels
- Doc_visits
- Full_meals_eaten
- vitD_supp
- Soft_drink
- Initial_admin
- HighBlood
- Stroke
- Complication_risk
- Overweight
- Arthritis
- Diabetes
- Hyperlipidemia
- BackPain
- Anxiety
- Allergic_rhinitis
- Reflux_esophagitis
- Asthma
- Services
- Initial_days
- Item1
- Item2
- Item3
- Item4
- Item5

- Item6
- Item7
- Item8

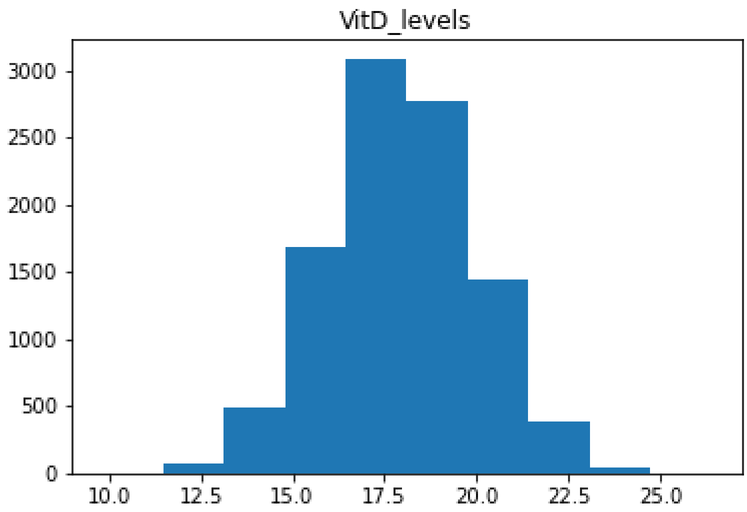
```
outcome = 'TotalCharge'

X = df.loc[:,df.columns!=outcome]
y = df[outcome]
```

There a lot of variables here, but they will be reduced later on.

The modified dataset can be found in `data/medical_prepared.csv`

Univariate Analysis

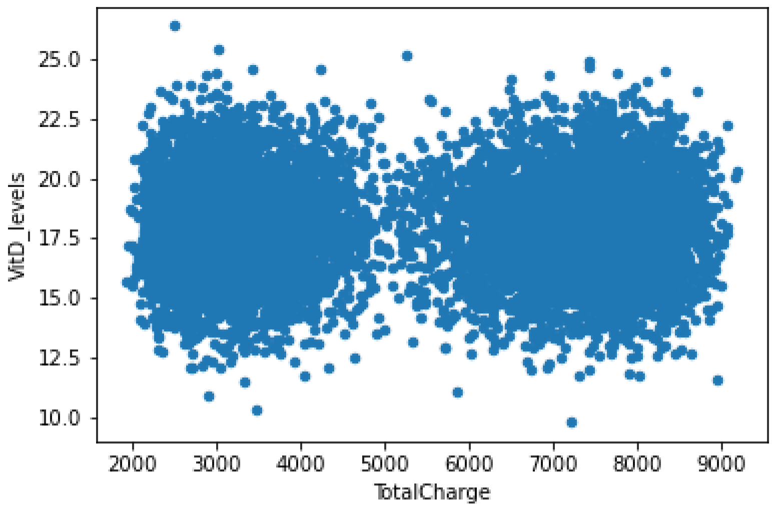
We perform univariate analysis on our variables. Here is the plot for `VitD_levels`



This plot as well as the plots for the rest of the univariate analysis can be round in `plots/` with the prefix "univariate".

Bivariate Analysis

Here is a bivariate plot of `VitD_levels` compared to `TotalCharge`



The rest of the bivariate pltos can be found in `plots/` with the prefix "bivariate".

Models

Initial Model

First we construct a model using all of the variables we picked out:

```
Xc = sm.add_constant(X)

initial_model = sm.OLS(y,Xc)
results = initial_model.fit()
results.summary()
```

OLS Regression Results

Dep. Variable:	TotalCharge	**R-squared:**	0.985
Model:	OLS	**Adj. R-squared:**	0.985
Method:	Least Squares	**F-statistic:**	1.757e+04
Date:	Fri, 13 Aug 2021	**Prob (F-statistic):**	0.00
Time:	12:20:00	**Log-Likelihood:**	-70095.
No. Observations:	10000	**AIC:**	1.403e+05
Df Residuals:	9962	**BIC:**	1.405e+05
Df Model:	37		
Covariance Type:	nonrobust		

Reduction

We'll use a reduction technique called "Backward Elimination", in which we start by training a model with all of our variables then remove variables one by one until we have removed the inconsequential ones.

The code below itterably trains a linear model then scores it using the AIC metric, removing a variable at each step until only the relevant variables are left.

```
# Adapted from lesson 3.3 of predictive modeling notes
linear_regression = LinearRegression(normalize=False,fit_intercept=True)

# Scores a model for a given X, y pair
def r2_est(X,y):
    return r2_score(y,linear_regression.fit(X,y).predict(X))

# Make a list of all features and their impact on the r2 score
r2_impact = list()
for j in range(X.shape[1]):
    selection = [i for i in range(X.shape[1]) if i!=j]
    r2_impact.append(((r2_est(X,y) - r2_est(X.values [:,selection],y))
,X.columns[j]))

# Make a list of the most impactful features called 'best_variables'
best_variables = list()
for imp, varname in sorted(r2_impact, reverse=True):
    if imp >= 0.001:
        best_variables.append(varname)
    print ('%6.3f %s' %  (imp, varname))
```

We end up with a reduced variable set consisting of:

- Initial_days
- Complication_risk
- HighBlood
- Hyperlipidemia

The reduced set can be found in `data/medical_reduced.csv`

The Reduced Model

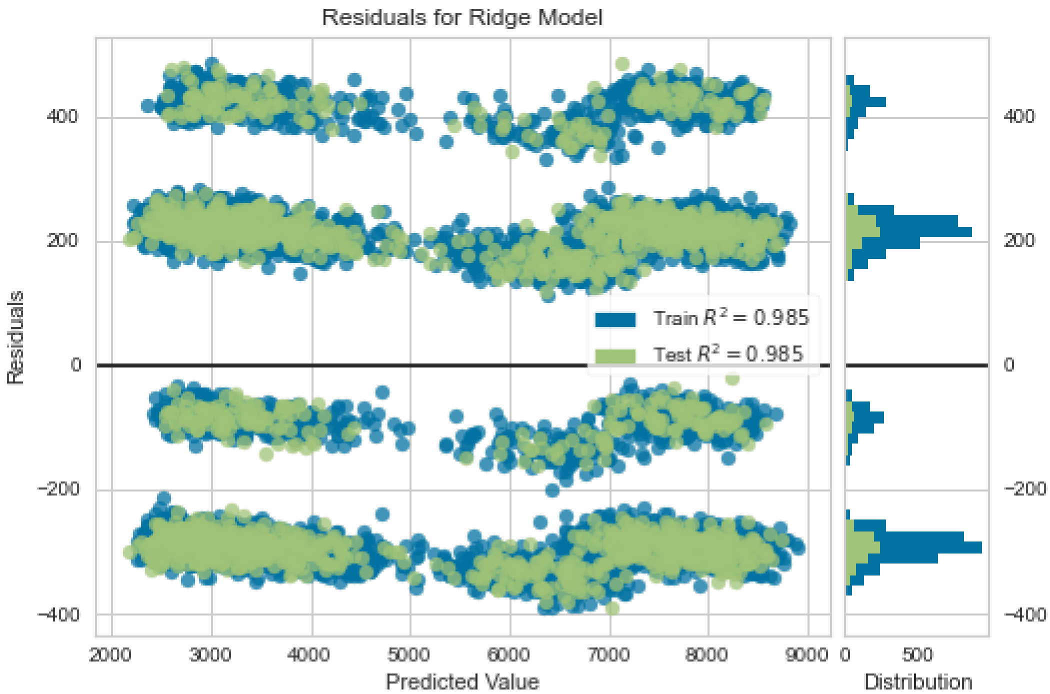
```
X_reduced = df_reduced.loc[:,df_reduced.columns!=outcome]
Xc_reduced = sm.add_constant(X_reduced)

model_reduced = sm.OLS(y,Xc_reduced)
results = model_reduced.fit()
results.summary()
```

OLS Regression Results			
Dep. Variable:	TotalCharge	**R-squared:**	0.983
Model:	OLS	**Adj. R-squared:**	0.983
Method:	Least Squares	**F-statistic:**	1.466e+05
Date:	Fri, 13 Aug 2021	**Prob (F-statistic):**	0.00
Time:	12:33:08	**Log-Likelihood:**	-70618.
No. Observations:	10000	**AIC:**	1.412e+05
Df Residuals:	9995	**BIC:**	1.413e+05
Df Model:	4		
Covariance Type:	nonrobust		

We were able to get similar accuracy with only 4 of the original variables!

Residuals

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

model = Ridge()
visualizer = ResidualsPlot(model)
visualizer.fit(X_train, y_train)
visualizer.score(X_test, y_test)
residual = visualizer.poof()
```



Summary

Coefficients

Here is a table of the coefficients, sorted by absolute value:

Coefficient	Variable
-202.0708	Complication_risk
118.2329	HighBlood
103.9143	Hyperlipidemia
81.8533	Initial_days

It seems that `Complication_risk` is the most influential variable at play.

Regression Equation

```
TotalCharge = 2633.9585 + 81.8533(Initial_days) + (-202.0708)
(Complication_risk) + 118.2329(HighBlood) + 103.9143(Hyperlipidemia)
```

Statistical significance

We were able to get within 2 thousandths of the accuracy of the full model with only the 4 variables we reduced too, which is an inconsequential change. With this information we now know that if a hospital wants to estimate the total cost of a visit they need only look at ta patient's Complication risk, whether they have high blood pressure, whether they have Hyperlipidemia, and how many days they are projected to stay.

Limitations

With only 10,000 patient records to work with there is a chance this model has been biased towards a particular conclusion. For example in another set of patients a different pre-existing condition such as Anxiety may contribute more to total cost.

Recomendations

Pay close attention to the complication risk, high blood pressure, and Hyperlipidemia in patients as well as how long they are projected to be in the hospital in order to calculate the cost of a visit.