Jhirsc21

March 30, 2024

1 Lab 10 - Linear Models

```
[1]: %matplotlib inline
```

1.1 Directions

The due dates for each are indicated in the Syllabus and the course calendar. If anything is unclear, please email EN685.648@gmail.com the official email for the course or ask questions in the Lab discussion area on Blackboard.

The Labs also present technical material that augments the lectures and "book". You should read through the entire lab at the start of each module.

Please follow the directions and make sure you provide the requested output. Failure to do so may result in a lower grade even if the code is correct or even 0 points.

1.1.1 General Instructions

1. You will be submitting your assignment to Blackboard. If there are no accompanying files, you should submit *only* your notebook and it should be named using *only* your JHED id: fsmith79.ipynb for example if your JHED id were "fsmith79". If the assignment requires additional files, you should name the *folder/directory* your JHED id and put all items in that folder/directory, ZIP it up (only ZIP...no other compression), and submit it to Blackboard.
   - do **not** use absolute paths in your notebooks. All resources should appear in the same directory as the rest of your assignments.
   - the directory **must** be named your JHED id and **only** your JHED id.
2. Data Science is as much about what you write (communicating) as the code you execute (researching). In many places, you will be required to execute code and discuss both the purpose and the result. Additionally, Data Science is about reproducibility and transparency. This includes good communication with your team and possibly with yourself. Therefore, you must show **all** work.
3. Avail yourself of the Markdown/Codecell nature of the notebook. If you don't know about Markdown, look it up. Your notebooks should not look like ransom notes. Don't make everything bold. Clearly indicate what question you are answering.
4. Submit a cleanly executed notebook. It should say `In [1]` for the first codecell and increase by 1 throughout.

1.2 Linear Regression

In a previous module (Lab 5), you performed EDA on the insurance data set. In this Lab, you should build a linear regression model trying to estimate `charges`.

[2]:
```
import numpy as np
import random as py_random
import numpy.random as np_random
import time
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats

sns.set(style="whitegrid")
```

[3]:
```
import models
```

2 Answer

Let's start by reading in the data and looking at the variables we have in the dataframe.

[4]:
```
insurance = pd.read_csv("https://raw.githubusercontent.com/
 ↪fundamentals-of-data-science/datasets/master/insurance.csv", header=0)
```

[5]:
```
insurance.head()
```

[5]:
```
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
```

So we have 6 variables not including our target variable. We can view their types as well.

[6]:
```
insurance.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

We see no missing entries, age and children are numeric, as is bmi but technically is a float. Smoker and sex are binary categorical variables, and region is categorical with 4 values (from our previous EDA analysis). Let's take a quick look at region again.

[7]:
```
insurance['region'].value_counts()
```

[7]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

We'll first start by converting the categorical variables into a one hot encoding. We'll look at transforming region first.

[8]:
```
regions = insurance['region'].unique()
region = {'region': ['southeast', 'southwest', 'northwest', 'northeast' ]}
data = pd.DataFrame(region)

insurance = pd.concat([insurance, pd.get_dummies(insurance['region'])], axis=1)
insurance.head()
```

[8]:
```
   age     sex     bmi  children smoker     region      charges  northeast  \
0   19  female  27.900         0    yes  southwest  16884.92400      False
1   18    male  33.770         1     no  southeast   1725.55230      False
2   28    male  33.000         3     no  southeast   4449.46200      False
3   33    male  22.705         0     no  northwest  21984.47061      False
4   32    male  28.880         0     no  northwest   3866.85520      False

   northwest  southeast  southwest
0      False      False       True
1      False       True      False
2      False       True      False
3       True      False      False
4       True      False      False
```

Looks pretty good. Now we can do a similar transformation for sex and smoker. We'll make sure the transformations don't cause unexpected changes.

[9]:
```
pd.DataFrame(insurance['sex'].value_counts())
```

[9]:
```
        count
sex
male      676
```

```
female    662
```

[10]:
```
insurance['female'] = insurance['sex'].apply(lambda x: 1 if x == 'female' else␣
 ↪0)

pd.DataFrame(insurance['female'].value_counts())
```

[10]:
```
        count
female
0         676
1         662
```

Look like we have the same value counts after the encoding. We can apply the same thing for smoker.

[11]:
```
pd.DataFrame(insurance['smoker'].value_counts())
```

[11]:
```
        count
smoker
no       1064
yes       274
```

[12]:
```
insurance['smokes'] = insurance['smoker'].apply(lambda x: 1 if x == 'yes' else␣
 ↪0)

pd.DataFrame(insurance['smokes'].value_counts())
```

[12]:
```
        count
smokes
0        1064
1         274
```

Looks good. For putting our encoded variables in our regression model, we are leaving out one value of the encodings. For region, we will leave out southeast, for sex we will leave out male, and for smoker we will leave out no.

[13]:
```
insurance.head()
```

[13]:
```
   age     sex     bmi  children smoker     region      charges  northeast  \
0   19  female  27.900         0    yes  southwest  16884.92400      False
1   18    male  33.770         1     no  southeast   1725.55230      False
2   28    male  33.000         3     no  southeast   4449.46200      False
3   33    male  22.705         0     no  northwest  21984.47061      False
4   32    male  28.880         0     no  northwest   3866.85520      False

   northwest  southeast  southwest  female  smokes
0      False      False       True       1       1
1      False       True      False       0       0
```

```
2      False       True      False        0        0
3       True      False      False        0        0
4       True      False      False        0        0
```

Here we just see all our encodings should be present in our dataframe.

Before we create our model, we can think about how each variable could affect charges (note we've already seen the pairwise EDA, so I will just summarize here):

age - positive most likely, as age increases, so do health issues

bmi - positive, charges would increase for obese people

children - positive, as number of children increase, charges are more costly

northeast - ? we don't have enough to go on here, possibly positive if there are higher population centers in the northeast

northwest - again we don't know

southwest - let's guess negative, just to oppose the north

female - negative, women tend to have higher life expectancy than men

smokes - positive, we know from EDA that smoking largely increases charges

Let's start with the all variables in model:

[14]:
```
model = '''charges ~ age + bmi + children + northeast
        + northwest + southwest + female + smokes'''

result = models.bootstrap_linear_regression(model, data=insurance)
```

[15]:
```
models.describe_bootstrap_lr(result)
```

[15]:

Model: charges $\sim$ age + bmi + children + northeast + northwest + southwest + female + smokes

			95% BCI	
Coefficients		**Mean**	**Lo**	**Hi**
	β_0	-13104.87	-14837.54	-10946.95
age	β_1	1035.02	189.98	1909.53
bmi	β_2	682.06	-333.47	1577.56
children	β_3	74.97	-851.58	1194.93
northeast	β_4	256.86	231.51	285.69
northwest	β_5	339.19	275.96	398.15
southwest	β_6	475.50	199.87	732.26
female	β_7	131.31	-359.07	783.29
smokes	β_8	23848.53	22713.14	25178.08

Metrics	**Mean**	**Lo**	**Hi**
σ	6062.10	5762.44	6475.10
R^2	0.75	0.72	0.78

We see the R^2 is 0.75, so the model explains 75% of the variability in charges. We should also look at the adjusted R^2 since we have so many variables in the model. I will copy the `adjusted_r_squared` function from *Fundamentals, page 779.*

[16]:
```
def adjusted_r_squared(result):
    adjustment = (result['n'] - 1)/ (result['n'] - len(result['coefficients'])␣
↪- 1 - 1)
    return 1 - (1 - result['r_squared']) * adjustment
```

[17]:
```
adjusted_r_squared(result)
```

[17]:
```
0.7490359662835133
```

We see about the same result here, but note the adjusted R squared could be handy to look at as we proceed to refine our model.

Let's describe our initial charges category as a reminder

[18]:
```
insurance['charges'].describe()
```

[18]:
```
count     1338.000000
mean     13270.422265
std      12110.011237
min       1121.873900
25%       4740.287150
50%       9382.033000
75%      16639.912515
max      63770.428010
Name: charges, dtype: float64
```

We see a std of 12,110, while our σ is 6062.10, almost half. Coefficients are all positive, but only charges hugely so. Aside from region, the other variable credible intervals contain 0. We can look at our strong/weak/mixed our evidence is that we predicted these variables correctly

[19]:
```
predictions = {'age': '+', 'bmi': '+', 'children': '+', 'northeast': '+',
            'northwest': '+', 'southwest': '-', 'female': '-', 'smokes': '+'}

models.evaluate_coefficient_predictions(predictions, result)
```

```
age P(>0)=0.980 (strong)
bmi P(>0)=0.920 (strong)
children P(>0)=0.520 (mixed)
northeast P(>0)=1.000 (strong)
northwest P(>0)=1.000 (strong)
southwest P(<0)=0.000 (weak)
female P(<0)=0.280 (weak)
smokes P(>0)=1.000 (strong)
```

We have strong evidence that age, bmi, northeast, northwest, and smokes are all positive. Weak evidence that southwest is negative, but we weren't really positive (heh), and then mixed results for children and weak results for female. From our EDA, we saw men and women didn't have much of a difference in charges, and we had a weak correlation between number of children and charges.

According to our table for evaluating coefficients, we should remove variables that had an unexpected sign and includes 0 - female falls into that category, so I will take it out of the model (we should not expect to see much of a difference between men and women anyway based on our EDA).

Let's try a model without the female variable

[20]:
```
model = '''charges ~ age + bmi + children + northeast
        + northwest + southwest + smokes'''

result = models.bootstrap_linear_regression(model, data=insurance)

models.describe_bootstrap_lr(result)
```

[20]:

Model: charges ~ age + bmi + children + northeast + northwest + southwest + smokes

			95% BCI	
Coefficients		**Mean**	**Lo**	**Hi**
	β_0	-13024.63	-15175.37	-10959.93
age	β_1	1034.36	354.42	2080.73
bmi	β_2	682.18	-121.01	1596.41
children	β_3	74.99	-752.59	969.07
northeast	β_4	256.97	234.61	274.26
northwest	β_5	338.66	267.45	400.28
southwest	β_6	474.57	301.71	651.22
smokes	β_7	23836.30	22598.83	24901.77

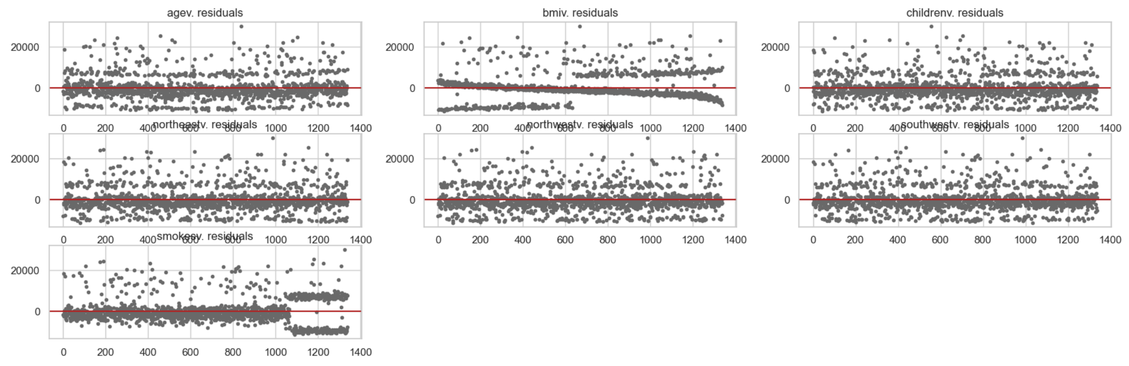
Metrics	**Mean**	**Lo**	**Hi**
σ	6060.18	5695.64	6404.91
R^2	0.75	0.72	0.78

We have the same R^2 value of 0.75 and a very similar, slightly smaller σ of 6060. This likely means the female variable does not contribute much to the overall model, and is thus fine to exclude.

Our coefficients and credible intervals are mostly the same as well.

Next we can look at residuals for each variable in the model

[21]:
```
figure = plt.figure(figsize=(20,6))

variables = ['age', 'bmi', 'children', 'northeast', 'northwest', 'southwest',␣
 ↪'smokes']

plots = len(variables)
rows = (plots // 3) + 1

for i, variable in enumerate(variables):
    axes = figure.add_subplot(rows, 3, i+1)
    keyed_values = sorted(zip(insurance[variable].values, result['residuals']),
                            key=lambda x: x[0])
    residuals = [x[1][0] for x in keyed_values]

    axes.plot(list(range(0, result['n'])), residuals, '.', color='dimgray')
    axes.axhline(y=0.0, xmin=0, xmax=result['n'], c='firebrick')
    axes.set_title(variable + 'v. residuals')

plt.show()
plt.close()
```



Definitely not ideal. Bmi shows almost a mirrored residual plot, while smokes shows higher variability for larger values. I could make an argument for the other variables being 'okay', but it is a little subjective here.

Our next step would be to attempt a transformation of the variables. For bmi in particular, I think we could try a square root transformation.

[22]:
```
insurance['bmi_sqrt'] = insurance['bmi'] ** 0.5
```

[23]:
```
model = '''charges ~ age + bmi_sqrt + children + northeast
        + northwest + southwest + smokes'''

result = models.bootstrap_linear_regression(model, data=insurance)

models.describe_bootstrap_lr(result)
```

[23]:

Model: charges ∼ age + bmi_sqrt + children + northeast + northwest + southwest + smokes

Coefficients		**Mean**	**95% BCI** **Lo**	**Hi**
	β_0	-23367.69	-27421.08	-20049.69
age	β_1	1031.63	198.92	2084.22
bmi_sqrt	β_2	653.40	-196.29	1715.98
children	β_3	49.00	-764.13	878.75
northeast	β_4	256.33	237.44	272.73
northwest	β_5	3768.69	3244.72	4414.98
southwest	β_6	474.89	238.69	746.19
smokes	β_7	23842.20	22985.17	24933.10

Metrics	**Mean**	**Lo**	**Hi**
σ	6055.01	5681.42	6336.64
R^2	0.75	0.72	0.79

Hmm so we don't see much of a change here. We might try then to perform a log transform instead

```
[24]: insurance['bmi_log'] = np.log((insurance['bmi']))
```

Now let's try the model again

```
[25]: model = '''charges ~ age + bmi_log + children + northeast
              + northwest + southwest + smokes'''

      result = models.bootstrap_linear_regression(model, data=insurance)

      models.describe_bootstrap_lr(result)
```

[25]: **Model: charges ~ age + bmi_log + children + northeast + northwest + southwest + smokes**

Coefficients		**Mean**	**95% BCI** **Lo**	**Hi**
	β_0	-37501.85	-43194.84	-30313.39
age	β_1	1014.43	-10.21	1775.65
bmi_log	β_2	609.45	-418.56	1514.82
children	β_3	13.70	-912.85	999.37
northeast	β_4	255.87	233.53	276.38
northwest	β_5	10267.33	8405.99	11979.28
southwest	β_6	475.38	239.46	705.28
smokes	β_7	23847.42	22693.79	25022.22

Metrics	**Mean**	**Lo**	**Hi**
σ	6054.18	5680.03	6421.50
R^2	0.75	0.72	0.78

Again not much of a difference here. I think overall there are too many variables in this model. Reducing the number of variables would perhaps be best. From our EDA, we know that region does not appear to have a huge effect on charges. We could consider either removing those variables or maybe combining them in some way.

Let's try a model without the region variables just for fun

[26]:
```
model = '''charges ~ age + bmi_log + children
        + smokes'''

result = models.bootstrap_linear_regression(model, data=insurance)

models.describe_bootstrap_lr(result)
```

[26]:

Model: charges $\sim$ age + bmi_log + children + smokes

			95% BCI	
Coefficients		**Mean**	**Lo**	**Hi**
	β_0	-35521.06	-40951.76	-28996.91
age	β_1	256.73	234.38	284.07
bmi_log	β_2	9793.93	7918.37	11549.33
children	β_3	473.31	272.55	747.00
smokes	β_4	23826.87	22527.52	24748.09

Metrics	**Mean**	**Lo**	**Hi**
σ	6061.68	5645.56	6423.30
R^2	0.75	0.72	0.77

This does not improve the model at all, but also does not make it worse. I'm not really sure where to go from here, it seems like any transformation or removal of variables does not improve the model in any significant way.

From EDA, we saw that age is sort of split into 3 subgroups, and so we might try to discretize this variable by turning into a categorical one and create a one hot encoding. Overall we know that most of the variation in the model comes from the smokes variable, as evident below

[27]:
```
model = '''charges ~ smokes
        '''

result = models.bootstrap_linear_regression(model, data=insurance)

models.describe_bootstrap_lr(result)
```

[27]:

Model: charges $\sim$ smokes

			95% BCI	
Coefficients		**Mean**	**Lo**	**Hi**
	β_0	8434.27	8153.03	8776.85
smokes	β_1	23615.96	22262.97	24932.47

Metrics	**Mean**	**Lo**	**Hi**
σ	7470.22	7163.50	7765.34
R^2	0.62	0.58	0.65

Since smokers is a binary categorical variable, we can't really transform it in a traditional way.

But let's try changing age first and see how that model performs.

```
[28]: from tabulate import tabulate
```

```
[29]: middle = insurance[(insurance['age'] >= 30) & (insurance['age'] < 50)]
```

```
[30]: young = len(insurance[insurance['age'] < 30])
      middle = len(insurance[(insurance['age'] >= 30) & (insurance['age'] < 50)])
      old = len(insurance[insurance['age'] >= 50])

      tabulate([[young], [middle], [old]], tablefmt='html')
```

```
[30]: '<table>\n<tbody>\n<tr><td style="text-align: right;">417</td></tr>\n<tr><td
      style="text-align: right;">536</td></tr>\n<tr><td style="text-align:
      right;">385</td></tr>\n</tbody>\n</table>'
```

```
[31]: insurance['age_group'] = insurance['age'].apply(lambda x: 'young' if x < 30␣
       ↪else ('middle' if x < 50 else 'old'))

      pd.DataFrame(insurance['age_group'].value_counts().sort_index())
```

```
[31]:            count
      age_group
      middle       536
      old          385
      young        417
```

Our data was transformed correctly. We will leave the young group out of the model. Let's check we have our dummies in the dataframe

```
[32]: insurance = pd.concat([insurance, pd.get_dummies(insurance['age_group'])],␣
       ↪axis=1)
      insurance.head()
```

```
[32]:    age     sex     bmi  children smoker     region      charges  northeast  \
      0   19  female  27.900         0    yes  southwest  16884.92400      False
      1   18    male  33.770         1     no  southeast   1725.55230      False
      2   28    male  33.000         3     no  southeast   4449.46200      False
      3   33    male  22.705         0     no  northwest  21984.47061      False
      4   32    male  28.880         0     no  northwest   3866.85520      False

         northwest  southeast  southwest  female  smokes  bmi_sqrt   bmi_log  \
      0      False      False       True       1       1  5.282045  3.328627
      1      False       True      False       0       0  5.811196  3.519573
      2      False       True      False       0       0  5.744563  3.496508
      3       True      False      False       0       0  4.764976  3.122585
      4       True      False      False       0       0  5.374012  3.363149
```

```
  age_group  middle    old  young
0     young   False  False   True
1     young   False  False   True
2     young   False  False   True
3    middle    True  False  False
4    middle    True  False  False
```

Good enough. Let's try a model

[33]:
```
model = '''charges ~ middle + old + bmi_log + children
        + smokes'''

result = models.bootstrap_linear_regression(model, data=insurance)

models.describe_bootstrap_lr(result)
```

[33]:

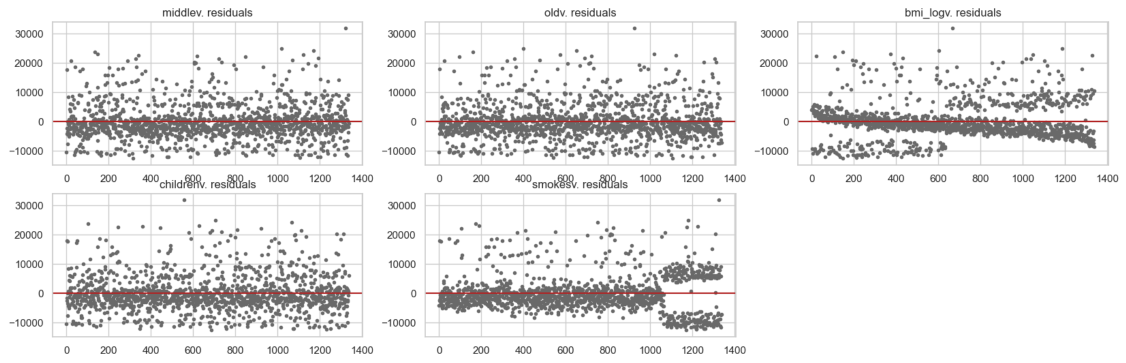
Model: charges $\sim$ middle + old + bmi_log + children + smokes

			95% BCI	
Coefficients		**Mean**	**Lo**	**Hi**
	β_0	-29385.91	-35000.20	-23363.56
middle	β_1	2840.17	2143.54	3469.84
old	β_2	8734.03	8119.87	9528.18
bmi_log	β_3	9823.00	8075.04	11538.62
children	β_4	624.48	369.56	943.05
smokes	β_5	23901.00	22789.37	24795.42

Metrics	**Mean**	**Lo**	**Hi**
σ	6156.67	5710.04	6508.19
R^2	0.74	0.72	0.78

Not much has changed from our intial model. We can look at the residuals as well.

[34]:
```
figure = plt.figure(figsize=(20,6))

variables = ['middle', 'old', 'bmi_log', 'children', 'smokes']

plots = len(variables)
rows = (plots // 3) + 1

for i, variable in enumerate(variables):
    axes = figure.add_subplot(rows, 3, i+1)

    keyed_values = sorted(zip(insurance[variable].values, result['residuals']),
                          key=lambda x: x[0])
    residuals = [x[1][0] for x in keyed_values]

    axes.plot(list(range(0, result['n'])), residuals, '.', color='dimgray')
    axes.axhline(y=0.0, xmin=0, xmax=result['n'], c='firebrick')
    axes.set_title(variable + 'v. residuals')

plt.show()
plt.close()
```



Again bmi_log and smokes have some patterns to them, but at this point I'm not sure how to properly transform these variables.

2.0.1 Conclusion

We know from EDA that smoker has the largest effect on charges. All other variables produce a much smaller effect, in terms of the regression model. We know that sex and region do not produce large effects, and don't reduce the model predictability, and so may not be worth including in the model.

We tried transforming bmi in order to make the residuals more symmetric, which did not help a ton. There are only so many transformations, but it doesn't seem like any would help. We could consider discretizing the variable similar to what we did with age.

Smokes variable also has a pattern in its residual plot, but since it is a binary categorical variable, we cannot easily perform mathematical transformations on this variable.

Comparing all the models we've seen, the 'best' one was the one with the bmi square root transformation, but really not by much, it has a slightly smaller σ but the same R^2 as what we've seen in the other models. At this point, we could simply perform a simple regression on just the smoking variable and that might be sufficient for our needs.

I look forward to viewing the solution and seeing how others approached this regression analysis!