Holmusk Interview

Cost of Care Prediction

Cathey Wang

06/26/2018

**Exploration of data:**

There are 4 csv files detailing diff ends of the care. Bill\_id & bill\_amount stores information about the costs. Clinical\_data stores information about the clinical visits, and the patient’s medical information. Demographics lists each patient’s basic info (sex, race, dob, etc).

It is noted that demographics is 3000 in length, clinical\_data is 3400 in length, and bill\_id & bill\_amount are each 13600 in length. This means that there are repeated visits for most patients, and for each visit, there are usually multiple bills associated. The sum of these bills would be the total cost of care for each visit.

I listed the unique values for each parameter. It was revealed that medical history, preop medication, and symptoms were all binary, while there were missing values for medical history 2 & 5. In demographics, there were 4 classes for race (Indian, Chinese, Malay, others), and 3 classes for residency (Singaporean, PR, foreigners). Apart from those 2 terms, age (derived from date of admission – date of birth), height, weight, & lab results were the only terms that were non-binary.

**Exploratory first model:**

Even though it’s not a great idea to make regression models with binary parameters – it’d just be shifting the curve based on the binary values – it’s a good to conduct a first step with linear regression based on all 27 parameters. I forced all nan’s from medical history 2 & 5 to 0.5 to minimize error.

In this model:

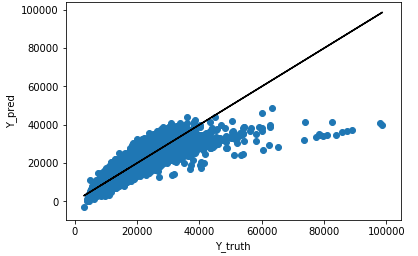
This is unlikely to be the best model, since many terms might not be linear. Eg. Probability of disease usually grows exponentially with age (Frammingham CVD Risk & Heart Age Formula: https://www.jmir.org/article/downloadSuppFile/3146/15304), so cost = w\*age + (everything else) is unlikely. But it may give us some insights.

I randomly chosen 90% of the clinical cases as the training set, and 10% for testing. For each clinical case, the corresponding patient info were found from demographics and the total cost was calculated from corresponding bills.

X columns: gender, age, height, weight, race, residence, history 1-7, medication 1-6, symptoms 1-5, lab results 1-3

Results:

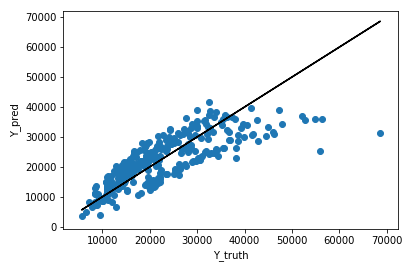
Training: (The black line marks the 1:1 ratio)



Mean squared error: 42545627.88

R2 score: 0.59

Testing:



Mean squared error: 33848380.08

R2 score: 0.63

Coeffs =

array([[ 2.00688149e+02, 2.21987768e+02, -1.59392954e+01,

1.51588710e+02, -2.03954187e-10, 2.11735378e+03,

6.07520748e+03, 5.15241249e+02, 5.34117935e+02,

4.28080100e+01, 9.42936650e+02, 3.24010357e+03,

1.22817516e+03, 2.25713481e+02, 2.99959296e+02,

5.23991561e+02, 3.64249468e+02, -2.92709458e+01,

5.78131693e+02, 2.41128717e+03, 4.04658769e+03,

3.84439903e+03, 2.99279015e+03, 1.05121620e+04,

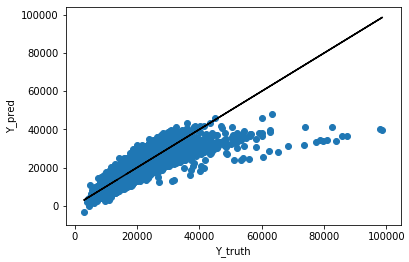
-7.31395973e+01, 7.58534805e+01, 2.42870202e+00]])

Takeaways:

1. The data performs reasonably well, given how little I expected of it. The overall trend was captured well, even though deviation was clear (R2 could be higher).
2. There was little overfitting. Test set performance did not differ from training set too much.
3. The predicted cost plateaued at higher values. This probably confirmed my previous concern that linear relationships weren’t optimal. Square terms & exponential terms may be able to generalize trends better.
4. The 5th term had very low coeffs, suggesting race isn’t strongly correlated to cost. Or probably more that using 0, 1, 2, 3 to denote Indians, Chinese, Malay & other would not be great for regression tasks. Consider changing to one-hot vectors instead
5. It is possible to predict cost with these 27 parameters. This is good news.

I fixed the problem raised in 4**: ternary & quaternary classes**, and retrained the system using linear regression:

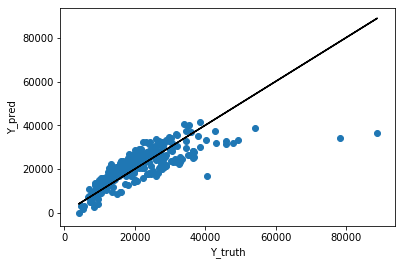
Training:



Mean squared error: 41488097.93

R2 score: 0.60

Testing:



Testing: Mean squared error: 38816846.48

Testing: R2 score: 0.59

Coeffs =

array([[ 1.83443491e+02, 2.29725669e+02, -3.14499689e+01,

1.50105917e+02, -1.47470250e+14, 1.67037314e+14,

-1.05420999e+13, 1.86430132e+14, 1.85379706e+14,

1.85379706e+14, 1.85379706e+14, 5.96947540e+03,

5.13659659e+02, 4.18666179e+02, 1.36022458e+01,

1.15516503e+03, 3.24438747e+03, 1.23306716e+03,

2.09260387e+02, 2.85110646e+02, 5.63599537e+02,

1.88030974e+02, -5.43872403e+01, 5.34350083e+02,

2.27702412e+03, 3.77763969e+03, 3.76407686e+03,

2.83579696e+03, 1.05681781e+04, -8.35301626e+01,

8.28494623e+01, 5.28558886e+00]])

The results indicate that race is not a non-factor anymore. Also testing set R2\_score went down. Although it does worry me that the terms seem too large (e14 power level). I might consider L2-regularization to fix that a bit in future versions, if it does cause problems.

**Additional parameters:**

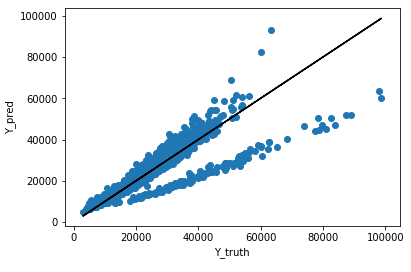
The previous version of the model listed all the values in the tables, and tried to make a linear correlation. But additional features may be useful. I added BMI (= weight/height^2) in this case. I have not removed height & weight respectively, but they might be removed in later versions to reduce overfitting.

**Improved relations:**

From observing the trends, I noticed that Y\_pred = WX + b wasn’t really linear with Y\_truth. The slope diminished at higher values, fitting more closely to a log curve.

Therefore, instead of mapping X to Y, we could map X to log(Y), and then calculate Y\_pred with exp(Y\_pred)

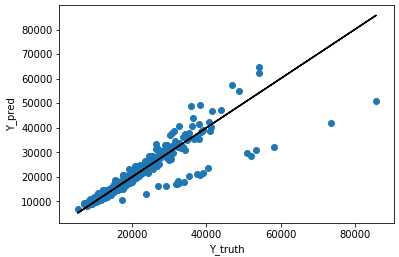
Training:

****

Mean squared error: 21794465.23

R2 score: 0.79

Testing:



Mean squared error: 26582502.95

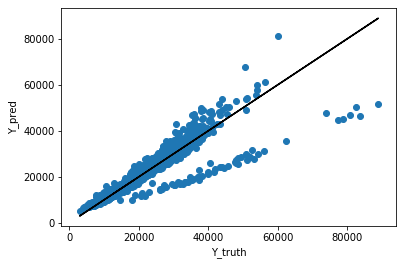
R2 score: 0.75

We see that the trend becomes more linear if we forced Y = exp(WX + b), implying costs grow exponentially with the growth in some parameters. This does sound like the tendency of today’s medical costs. However, we notice that there’s clear deviation. I suspect that this indicates that different cohorts may not subscribe to the same coefficients. We could consider training different models with respect to the binary terms such as gender.

**Separate binary groups and train separately:**

I separated X into X\_f and X\_m, based on gender (there were 1702 females and 1698 males), and removed the index for gender, since it would all be the same value in respective matrices.

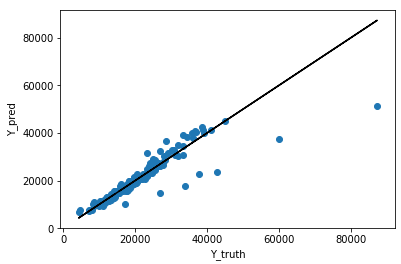
Female, train:



Mean squared error: 20684767.47

R2 score: 0.79

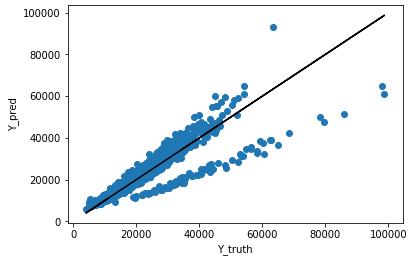
Female, test:



Mean squared error: 19731794.51

R2 score: 0.80

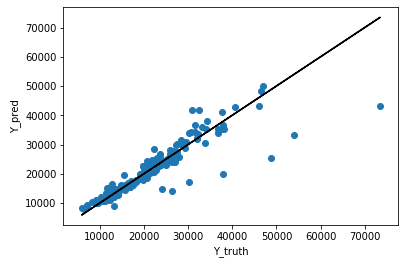
Male, train:



Mean squared error: 19182046.50

R2 score: 0.80

Male, test:



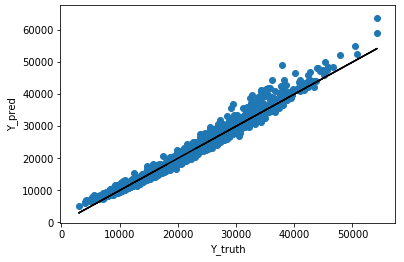
Mean squared error: 50884734.42

R2 score: 0.63

So this means that gender is not the factor causing the split in the results. We need to look at other classes.

I noticed in residence:

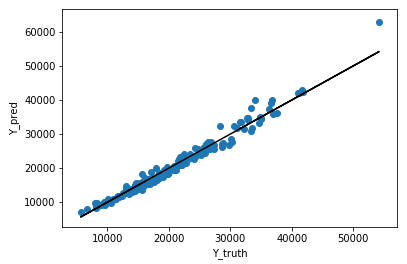
Singporean, train:



Mean squared error: 1992710.94

R2 score: 0.97

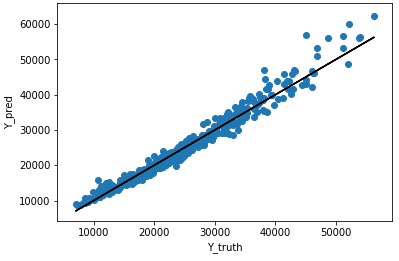
Sinaporean, test:



Mean squared error: 1824547.46

R2 score: 0.97

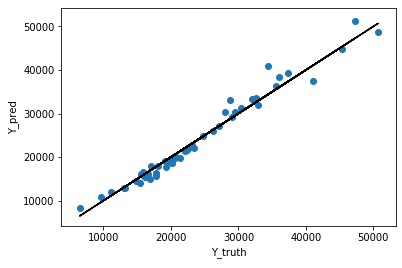
PR, train:



Mean squared error: 3115352.00

R2 score: 0.97

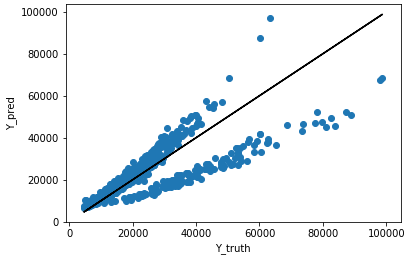
PR, test:



Mean squared error: 2900478.02

R2 score: 0.97

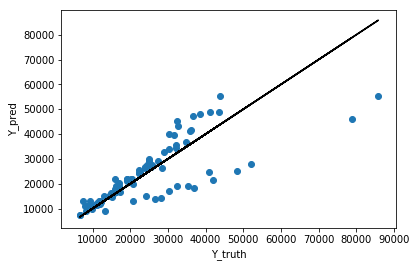
Foreigner, train:



Mean squared error: 77832808.98

R2 score: 0.58

Foreigner, test:



Mean squared error: 75705235.14

R2 score: 0.61

As shown we can predict cost to a high accuracy for Singaporeans and PRs. The foreigners are having splitting results though. And it might make sense. Since there might be some very different people in this mix. It’s possible that some foreigners, without proper insurance, chooses to not treat their conditions properly, which contributes to the discount in price.

I tried to go through all the binary classes in the dataset and find a certain class that was able to provide more differentiation, but there weren’t a class that was able to achieve that. So with the current system, we could write an algorithm that first checks the person’s residence status. If the person’s Singaporean or PR, then we can predict their cost to a fairly accurate degree. If foreigner, then the results are less confident. More information may be needed from the patients to make a better prediction. (eg. Insurance status)

**Just an idea: Use a neural network for regressions**

Deep learning is usually used for classifications tasks, instead of regression. However, if we replace softmax activation with linear/relu in the final layer, and use MSE as the loss function, it is possible to use neural networks for regression, even though it might be “overkill”.

The problem is, with this given task, a relu-activated neural net is not expected to perform much better than a linear regression model. Most linear transform outputs are expected to be in the > 0 range, which means the relu-activation output is not very different from its input. Which means the deep layers don’t really do much to the performance, W1W2…WnXis essentially just WX, making it not much different from linear regression.

What might improve performance here is adjusting the non-linear activation function. Eg. Use keras.backend.exp() for activation. Common nonlinear activation functions sigmoid & tanh have diminishing slopes, which is the opposite of what we want, hence excluded. If only one layer is used, that essentially becomes the linear regression to log(Y) model that I discussed in the previous sections.