Midterm Project Report

Author: Artem Starodubtsev

Goal: Predict annual medical insurance charges using linear-regression models, comparing performance with and without the income feature, and gender-specific approach.

EDA

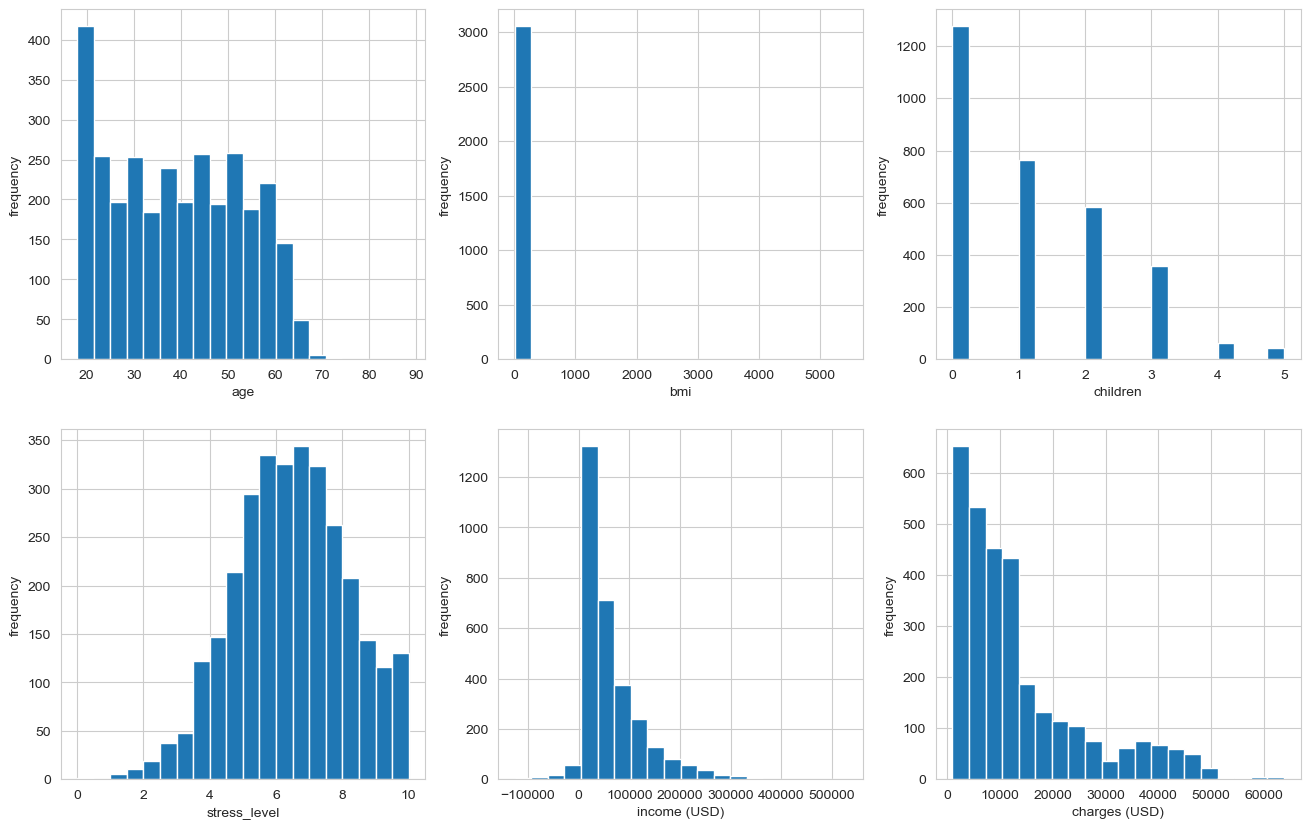
We have a raw dataset. First, we want to understand what data we are dealing with.

For this we use .describe() for numerical and .unique() for categorical.

Dropping abnormal values (like negative children, strange work\_sectors, etc. ).

Alongside starting to format data for future (str to int/float transformations).

Plot data to gather some insights from visuals.



Here we clearly see skewness of some features and is they are uniform/distributed

A group of blue and white graphs

AI-generated content may be incorrect.

Data transformations:

* bmi: dropped outlier (values, not rows)
* income: dropped negative values (but created a feature-flag) + log1p transform
* charges: lop1p transform

Age is looking uniform, so we might want to bin it. Let’s look at corr heatmap

|  |  |
| --- | --- |
|  | 1. strong income and charges correlation 2. age to charges correlation 3. low children to stress\_level correlation   As of (2) I decided not to bin age feature |

Missing values were held mostly with simple mode/medium imputation (as they all had <5% of missing vals). For some numerical features, I used StandardScaler (good for L1/L2 later) + used kNN imputation with JS loss for keeping distribution as much as possible (also N of neighbors was tuned with n=2).

After imputation categorical values were One-Hot encoded.

Dataset was split with 70/30 ratio + shuffle with set seed (seeds were “tuned” too)

As a feature selection part, I decided to create and sum 3 masks from each method (AVOVA, RF, RFE (RF)), number of n\_features was manually tuned.

Model building

At first let’s use sklearn LR implementation:

|  |  |  |
| --- | --- | --- |
| set | RMSE | R^2 |
| train\_reduced | 0.2015 | 0.7733 |
| test\_reduced | 0.2198 | 0.7652 |

We’ll use them as sanity checks for our implementation.

Batch Gradient Descent.

I’ve created a general function to run learning procedure with given params and export logs.

For BGD best learning rate = 0.299, but as seen from plot It’s not the best value (but line is flat, so there is no point searching for better eta)

|  |  |
| --- | --- |
|  |  |

N.B. I’ll present table with all metrics by the end of report

We can see that It took 500 epochs to achieve “best” loss result, but it’s still convergences (thus it’s super slow, so we interrupt training)

Mini-Batch Gradient Descent.

For MBGD I tried different learning rates and batch sizes, best\_eta = 0.151 with batch\_size = 32,

Convergence at epoch 273

|  |  |
| --- | --- |
|  |  |

Here we see minor win in final MSE loss, lower learning speed and lower number of epochs before convergence. Learning metrics aren’t smooth. But it’s totally fine, as with small batches we have not 1, but N steps inside each epoch. It also answers why it’s not smooth, as each step takes gradient of different batches, each step thus have different span towards convergence point, but length of step stayed the same, that’s why we have less epochs and smaller learning\_rate.

Batch Gradient Descent (polynomial + interaction features)

We preprocess dataset\_reduced (in my case X\_train\_reduced and X\_test\_reduced) and chose numerical features to “multiply”/”expand” (I don’t know how to properly call it)

best\_eta = 0.36, epoch 183

|  |  |
| --- | --- |
|  |  |

Polynomial features helped us! We’ve Improved model accuracy.

Also, we need a big learning rate, as we have more features, but it still convergences pretty fast

BGD Lasso (L1)

I’ll use learning rate from previous model (0.36) and tuning end up with alpha = 0.005, epoch 149

A screenshot of a graph

AI-generated content may be incorrect.

Loss didn’t change (1e-3 change, +-), but we found features that can be dropped.

We can drop 7 features and keep accuracy.

BGD Ridge (L2)

learning rate from previous model (0.36) and tuning end up with alpha = 0, epoch 191.

tuning alpha resulted in 0 => L2 regularization didn’t improve our model.

(Image skipped)

Gender specific model

We will modify \*\_reduced to \*\_reduced\_f (female) and \*\_reduced\_m (male)

N.B. In code I first concat train and test datasets in tmp\_df then do gender split -> train/test split

Let’s run MBGD with batch\_size = 64.

Male:

best\_eta = 0.071, epoch 30

|  |  |
| --- | --- |
|  |  |

As we can see performance is worse (compared to general model \*\_reduced), but this might not be the case for female.

Female.

best\_eta = 0.036, epoch 115

|  |  |
| --- | --- |
|  |  |

A little bit better then general model (for female samples)

**Conclusion about gender-specific models: none of models showed improvement for gender-specific query => thus general model would be better**

Dataset without income

I’ve decided to “without income” preamble in the end (for this I’ve copied notebook and dropped income and adjusted some EDA steps)

As expected from correlation analysis we will lose in accuracy as “income” feature had 0.83 correlation with target feature.

Polynomial + interaction features didn’t help increasing accuracy a lot.

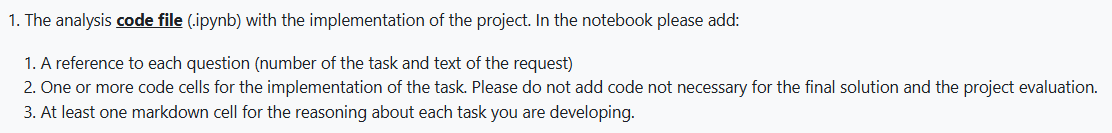
Also, both regulations didn’t improve (tuning got us alpha = 0 for L1/L2), makes sense as without “income” we want to gather as much info from other features as possible.

Performance metrics

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| preamble | specific | dataset+model | MSE | RMSE | MAE | R^2 |
| with income | general | reduced | 0.2207 | 0.4698 | 0.297 | 0.7644 |
| reduced (batch) | 0.2178 | 0.4667 | 0.2897 | 0.7674 |
| **augmented** | **0.1855** | **0.4307** | **0.2493** | **0.8019** |
| augmented (L1) | 0.1856 | 0.4308 | 0.252 | 0.8018 |
| augmented (L2) | 0.1869 | 0.4323 | 0.2508 | 0.8004 |
| male | reduced (batch) | 0.2559 | 0.5059 | 0.3098 | 0.7556 |
| female | reduced (batch) | 0.2152 | 0.4639 | 0.2852 | 0.7362 |
| without income | general | reduced | 0.2786 | 0.5276 | 0.3641 | 0.7025 |
| reduced (batch) | 0.2759 | 0.5252 | 0.3465 | 0.7054 |
| **augmented** | **0.2691** | **0.5188** | **0.3581** | **0.7126** |
| **augmented (L1)** | **0.2691** | **0.5188** | **0.3581** | **0.7126** |
| **augmented (L2)** | **0.2691** | **0.5188** | **0.3581** | **0.7126** |
| male | reduced (batch) | 0.315 | 0.5613 | 0.3742 | 0.6992 |
| female | reduced (batch) | 0.2764 | 0.5258 | 0.3477 | 0.6612 |

N.B. Therefore general augmented is the best model in “with income” category, L1 regularization would be a better choice.

P.S. It was very intense research as I’ve been trying all possible hyper values + seeds [1, 2, 3, 4, 5, 42, 52, 89, 123, 228, 666, 777], a lot of those seeds had test\_loss < train\_loss (trivial split), also batch\_size was affecting this. As a result my notebooks aren’t clean…



1. I’ve added guide-marks only to the first notebook “main.ipynb”, “main\_wo\_income.ipynb” was a copy of earlier versions, so there are no marks. Sorry.
2. Done
3. I’ve moved reasoning to “report.docx”, as after EDA phase I’ve been redoing code a LOT, and syncing it with markdowns was tough