HarvardX PH125.9x STEM Salaries

Daniel Yinanc

2/15/2022

1. Introduction

Data Science and STEM salaries dataset is a data set created in Kaggle via scrapping data from levels.fyi, a premier website for salary and total comparisons for data scientists and other technology workers across companies, locations, titles and levels.

Level.fyi dataset contains over 62,000 anonymous salaries collected through the time of the website. It is data cleaned a very tiny bit but there are a lot of data quality issues as we will illustrate below.

Our goal here is to build a predictive model that can predict total annual compensation of a tech worker from other variables, hence creating a tool that can aid tech company management, human resources and candidates themselves in salary negotiations. A high accuracy model that can predict the total annual compensation of a candidates part position or a target position will be of great benefit in the opaque labour market we live in, where actual values of compensation are a closely guarded secret.

Machine learning models we will progress from the baseline linear regression to models of greater sophistication utilizing different predictors from the dataset, culminating with gradient boosted model as the best model. We will present all models in terms of their RMSE to demonstrate the principles of model development.

As data-size is extremely large, I used parallels library for CPU level parallelization and had to find out how to use GPU accelerated XGBoost algorithm for model development. Even with all optimization work I undertook, model training takes many hours end to end. A lot of these long-running code snippets are NOT being evaluated within the RMarkdown. Solution code submitted contains same code and can be utilized for reproduction.

2. Data Exploration

2.1 Data Exploration

Below are variables in the Levels.fyi dataset.

- timeStamp: When the data was recorded. (numeric)
- company: Name of the company (character)
- level: Company specific levels (character)
- title: Role specific title (character)
- totalyearlycompensation: Total paid out compensation annually. (numeric)
- location: Job location. (char)
- yearsofexperience: Years of experience (numeric)
- yearsatcompany: Years of experience at the reported company (numeric)
- tag: Search tags for classifying job categories (character)
- basesalary: Base cash component of compensation (numeric)
- stockgrantvalue: Stock grant component of compensation (numeric)
- bonus: Annual cash bonus component of compensation (numeric)
- gender: Gender (numeric)

- otherdetails: Comments and other meta-data provided by submitters (character)
- cityid: Levels.fyi's internal location identifier (numeric)
- dmaid: Levels.fyi's internal identifier (numeric)
- rowNumber: Row number in data source (numeric)
- Masters_Degree: Boolean for master degree (numeric)
- Bachelors_Degree: Boolean for bachelor degree (numeric)
- Doctorate_Degree: Boolean for doctorate degree (numeric)
- Highschool: Boolean for highschool degree (numeric)
- Some_College: Boolean for associate or taking some courses (numeric)
- Race_Asian: Boolean for being asian (numeric)
- Race_White: Boolean for being asian (numeric)
- Race_Two_Or_More: Boolean for having parents in two or more races (numeric)
- Race_Black: Boolean for being black (numeric)
- Race_Hispanic: Boolean for being hispanic (numeric)
- Race: Race of the submitter (character)
- Education: Education level of the submitter (character)

2.1.2 Levels.fyi dataset

Download locations can be found for result replication purposes

- https://www.kaggle.com/jackogozaly/data-science-and-stem-salaries
- https://www.levels.fyi/

2.1.3 Data Load

In order to predict the ratings as a recommendation engine, I need to load initial variables to dataframes and transform them to a format that can be used later on.

```
## Rows: 62324 Columns: 29

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): timestamp, company, level, title, location, tag, gender, otherdeta...
## dbl (19): totalyearlycompensation, yearsofexperience, yearsatcompany, basesa...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2.2 Basic Summary Statistics

2.2.0 First few entries

As we can see from the very first entries, there are a lot of missing data entries in categorical variables such as gender and education related variables.

Also we can immediately see that variable scales do not match, indicating data itself is not normalized and/or scaled in any way.

```
## Rows: 62,324
## Columns: 23
## $ timestamp               <chr> "6/7/2017 11:33:27", "6/10/2017 17:11:29", "6/~
## $ company                 <chr> "Oracle", "eBay", "Amazon", "Apple", "Microsof~
## $ level                   <chr> "L3", "SE 2", "L7", "M1", "60", "63", "65", "6~
## $ title                   <chr> "Product Manager", "Software Engineer", "Produ~
## $ totalyearlycompensation <dbl> 127000, 100000, 310000, 372000, 157000, 208000~
## $ location                <chr> "Redwood City, CA", "San Francisco, CA", "Seat~
```

```
## $ yearsofexperience       <dbl> 1.5, 5.0, 8.0, 7.0, 5.0, 8.5, 15.0, 4.0, 3.0, ~
## $ yearsatcompany          <dbl> 1.5, 3.0, 0.0, 5.0, 3.0, 8.5, 11.0, 4.0, 1.0, ~
## $ tag                     <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ gender                  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ otherdetails            <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ masters_degree          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ bachelors_degree        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ doctorate_degree        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ highschool              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ some_college            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race_asian              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race_white              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race_two_or_more        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race_black              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race_hispanic           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ race                    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ education               <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
```

3. Dependent Variable Analysis

3.1 Totalyearlycompensation

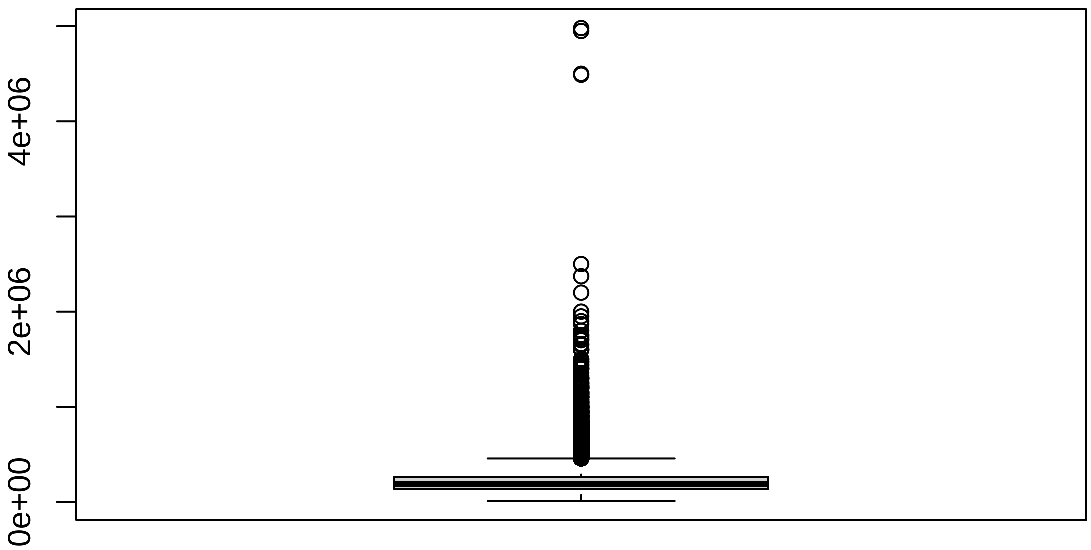
Our models will target to predict Total Yearly Compensation (totalyearlycompensation) using all other variables. Let's take a peek at the salient features of this variable for us to understand what this variable is and any important properties we can glean.

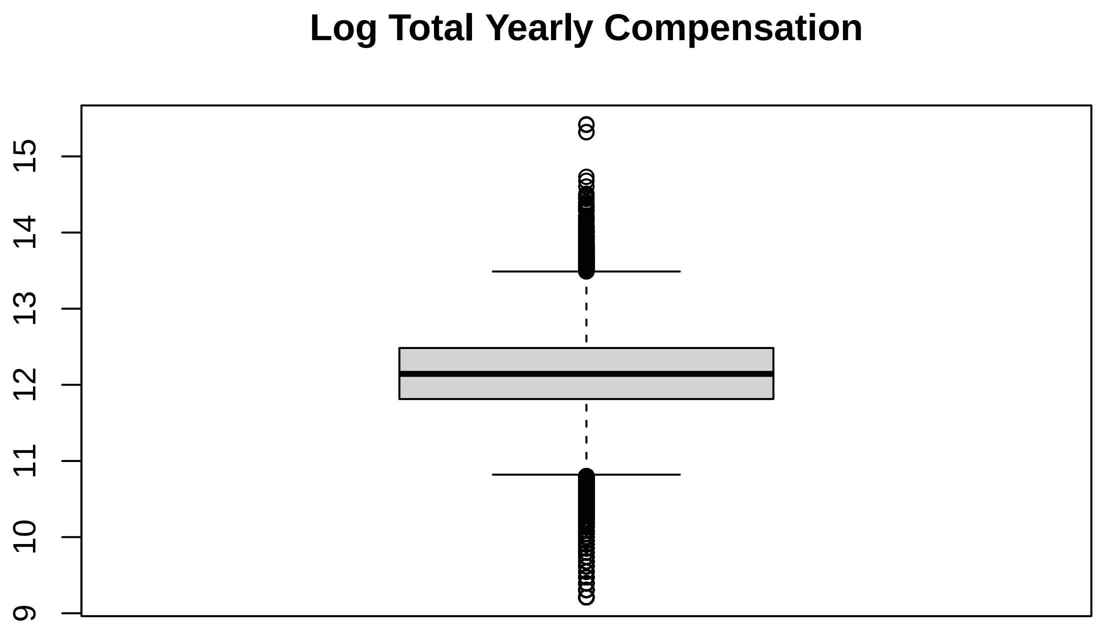
```
## # A tibble: 1 x 5
##       n    mean      sd   min     max
##   <int>   <dbl>   <dbl> <dbl>   <dbl>
## 1 62324 216120. 137977. 10000 4980000
```

We can quickly see that maximum value for our dependent variable is $4,680,000 while the minimum value is $10,000. These numbers with a mean of $238,000 indicate a very skewed distribution, which can thwart a lot of analytical approaches.

Let's visually see how skewed it is via a candlestick plot, indicating a major skewed variable with very large outliers on the top range.

Total Yearly Compensation





As we can see, even a logarithmic transformation did not fix the issue of skew in a meaningful manner as we can easily see substantial outliers on top and bottom of the logarithmic range now.

There does not seem to exist an easily discovered transformation that can lead to predicted variable to start to have some desirable distribution characteristics.

4. Independent Variables Analysis

This section covers our analysis of all other variables to assess if we should include them in our predictive models or not.

4.1 Categorical Variables

These variables are of the form of character strings and without exception contain missing entries as well as potentially typos. These data issues are typical of human input via web forms.

4.1.1 Company

Company is the name of the company for the submitted compensation package. Vast majority of submitted salaries and profiles belong to a handful of companies. This independent variable suffers from very strong imbalance, reducing its overall predictive power substantially.

Out of 1623 companies, top 10 firms represent over 40 percent of all entries, another clear indication of labour market power of Amazon, Microsoft, Google, Facebook and Apple with a few notable others such as Oracle, Intel, Cisco and IBM rounding out the top 10.

```
## # A tibble: 1,632 x 2
## # Groups:   company [1,632]
```

```
##    company        n
##    <chr>      <int>
##  1 Amazon      8082
##  2 Microsoft   5140
##  3 Google      4289
##  4 Facebook    2970
##  5 Apple       2017
##  6 Oracle      1127
##  7 Salesforce  1049
##  8 Intel        944
##  9 IBM          907
## 10 Cisco        902
## # ... with 1,622 more rows
```

We can also see that company average annual yearly compensation is different from each other, strongly indicating predictive ability.

```
## # A tibble: 1,632 x 2
##    company        mean_comp
##    <chr>              <dbl>
##  1 Coupa software  1483000
##  2 Cloudkitchens    700000
##  3 amplitude        680000
##  4 Doordash         593500
##  5 synaptics        539000
##  6 zillow group     520000
##  7 UBER             506667.
##  8 PDT Partners     500000
##  9 snapchat         485000
## 10 Netflix          481377.
## # ... with 1,622 more rows
```

4.1.2 Level

Levels are company specific employee compensation categorizations such as ICT3-6 for Apple, L3-8 for Google that determine salary bands, allowed bonus ratios and maximum RSU grants.

We can see below from grouped means, levels themselves are actually a by-product of company variable, as they are inventions of companies themselves to determine where in reporting and compensation hierarchy people are.

```
## `summarise()` has grouped output by 'company'. You can override using the `.groups` argument.
```

```
## # A tibble: 13,509 x 3
## # Groups:   company [1,632]
##    company   level       mean_comp
##    <chr>     <chr>           <dbl>
##  1 Microsoft 80            4950000
##  2 Facebook  E9            4735000
##  3 SoFi      EVP           2000000
##  4 Google    L10           1903500
##  5 Uber      Sr Director  1900000
##  6 Facebook  D2            1730000
##  7 Facebook  Director      1716000
##  8 Uber      L8            1700000
##  9 Snap      L8            1635667.
## 10 Zapier    L8            1605000
```

```
## # ... with 13,499 more rows
```

As we are utilizing company as an independent variable due to its clear predictive power, another highly correlated variable like levels we won't be introducing.

Another problem we are facing with levels is illustrated below, entries are incorrect, Facebook does not have a level called "L8 Director Product Management" or "L7 Product Management", it most likely is a title/function and level combined, reducing variable's utility very substantially. Resolving these data inconsistencies is way beyond the scope of this assignment.

Self-provided data like this has these natural weak points that we must take into consideration when selecting variables to use as part of our model.

```
## # A tibble: 68 x 3
##    level                          mean_comp   cnt
##    <chr>                              <dbl> <int>
##  1 E9                               4735000     2
##  2 D2                               1730000     2
##  3 Director                         1716000     1
##  4 D1                               1347250     8
##  5 E8                               1020250     4
##  6 L8 Director Product Management  1000000     1
##  7 Senior Engineering Manager       998000     1
##  8 E7                               865243.    37
##  9 M2                               801745.    55
## 10 L7 Product Manager               731800     10
## # ... with 58 more rows
```

4.1.3 Title

Title variable represents job titles on submitted compensation data by end users. Due to various historical factors as well as heterogeneity within the industry, titles themselves are not transferrable between technology firms. Vice-President title on an quantitative tech firm indicate a mid-level programmer while same title indicates a near top level executive on Google or Facebook. This has obvious implications on the predictive power of title alone, however this does not mean that it has no relevance to compensation.

Considering these factors, let us see if there is a discernible difference in average salaries for different titles

```
## # A tibble: 15 x 4
##    title                        mean_comp   cnt   stdev
##    <chr>                            <dbl> <int>   <dbl>
##  1 Software Engineering Manager   354563.  3557 228573.
##  2 Product Manager                257125   4632 181666.
##  3 Technical Program Manager      237100.  1381 108235.
##  4 Sales                          214273.   461 128206.
##  5 Hardware Engineer              213655   2200 108319.
##  6 Solution Architect             212576.  1152  96318.
##  7 Product Designer               207661.  1515 109772.
##  8 Software Engineer              205241. 40974 122089.
##  9 Data Scientist                 203481.  2576 109134.
## 10 Marketing                      198972.   710 115845.
## 11 Human Resources                178712.   364 101163.
## 12 Management Consultant          162795.   976  98474.
## 13 Mechanical Engineer            158443.   490  87964.
## 14 Recruiter                      155581.   451  69747.
## 15 Business Analyst               129728.   885  67272.
```

As we can see above, titles certainly have an impact on average salaries, this is a strong finding with number of data points backing it. However as we anticipated standard deviation is very high within title compensation levels indicating the heterogeneity of what an actual title means to a company and within the industry.

This level of standard deviation decreases predictive power of the variable, introducing potential overfitting. As a result, we won't be using it for building our models.

4.1.4 Location

Location variable is the physical location where the compensation data pertains to. From domain knowledge of tech industry compensations, we would expect to see a strong location dependency here, favoring Silicon Valley.

Let us see if this prediction of ours have the data backing needed to support this hunch to the level of us elevating location as a strong predictor variable.

```
## # A tibble: 1,049 x 4
##    location                            mean_comp   cnt   stdev
##    <chr>                                   <dbl> <int>   <dbl>
##  1 Aspen, CO                             650000      1      NA
##  2 Chapel Hill, NC                       605000      2 217789.
##  3 San Mateo, FL                         486000      2 299813.
##  4 Highland Park, NJ                     480000      1      NA
##  5 Los Gatos, CA                         479186.   226 131294.
##  6 Wimborne Minster, EN, United Kingdom   444000      1      NA
##  7 Los Altos, CA                         436000      6 244164.
##  8 Londonderry, OH                       405000      1      NA
##  9 Nazareth Illit, HZ, Israel            390000      1      NA
## 10 Sammamish, WA                         388000      1      NA
## # ... with 1,039 more rows
```

If we clean-up one-off locations by asking for at least 25 data points on a single location, real picture emerges far more clearly indicating California as the main location for high tech salaries. This is in congruence with our domain knowledge hinting at Silicon Valley effect on salaries.

```
## # A tibble: 149 x 4
##    location          mean_comp   cnt   stdev
##    <chr>                 <dbl> <int>   <dbl>
##  1 Los Gatos, CA       479186.   226 131294.
##  2 Menlo Park, CA      366161.  1430 266225.
##  3 Mountain View, CA   296674.  2260 149654.
##  4 San Mateo, CA       289982.   171 207981.
##  5 Cupertino, CA       287863.  1424 122706.
##  6 San Francisco, CA   285609.  6755 156005.
##  7 Kirkland, WA        282933.   163 113355.
##  8 Sunnyvale, CA       275002.  2228 127186.
##  9 San Bruno, CA       273934.   122 115128.
## 10 Santa Monica, CA    262395.   129 173548.
## # ... with 139 more rows
```

Except a single entry from state of Washington, all top entries are in continental United States and are all in California. Not only that but all these cities and towns are within the definition of Silicon Valley.

We will be using this variable in our predictive model as a result.

4.1.5 Gender

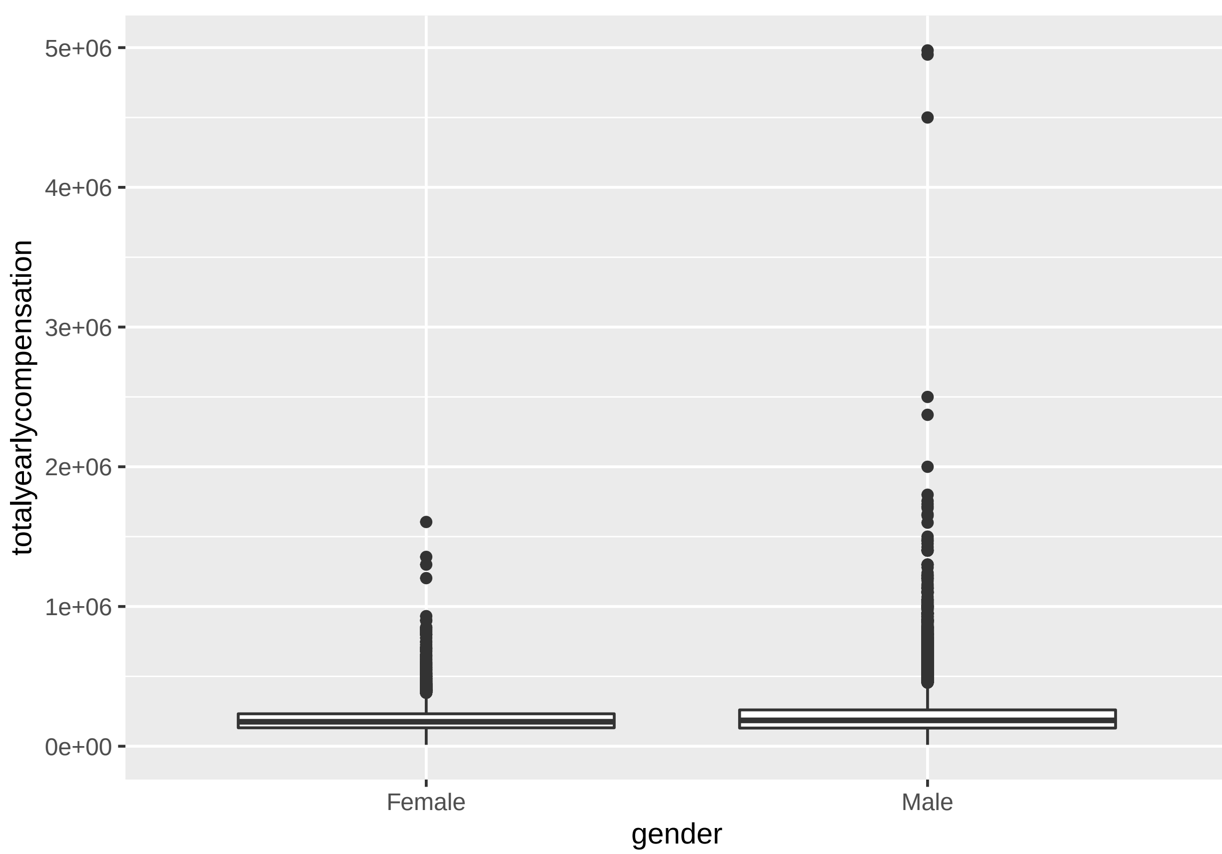
Gender is not a mandatory field on levels.fyi submissions, it is the gender of the submitted recipient. We would expect gender has an impact on ICT jobs as illustrated by domestic and international research [1,2]. Female compensation we expect to have a negative impact on total annual compensation.

Let's see if data provides support for our domain knowledge and academic research regarding gender impact on compensation. Considering the fact that these values are all self-reported with no verification, not validating or invalidating external research.

```
## # A tibble: 5 x 4
##   gender                          mean_comp   cnt   stdev
##   <chr>                               <dbl> <int>   <dbl>
## 1 Other                             232348.   400 141282.
## 2 <NA>                              230441. 19413 143962.
## 3 Male                              212250. 35524 139522.
## 4 Title: Senior Software Engineer   205000      1     NA
## 5 Female                            195076.  6986 104983.
```

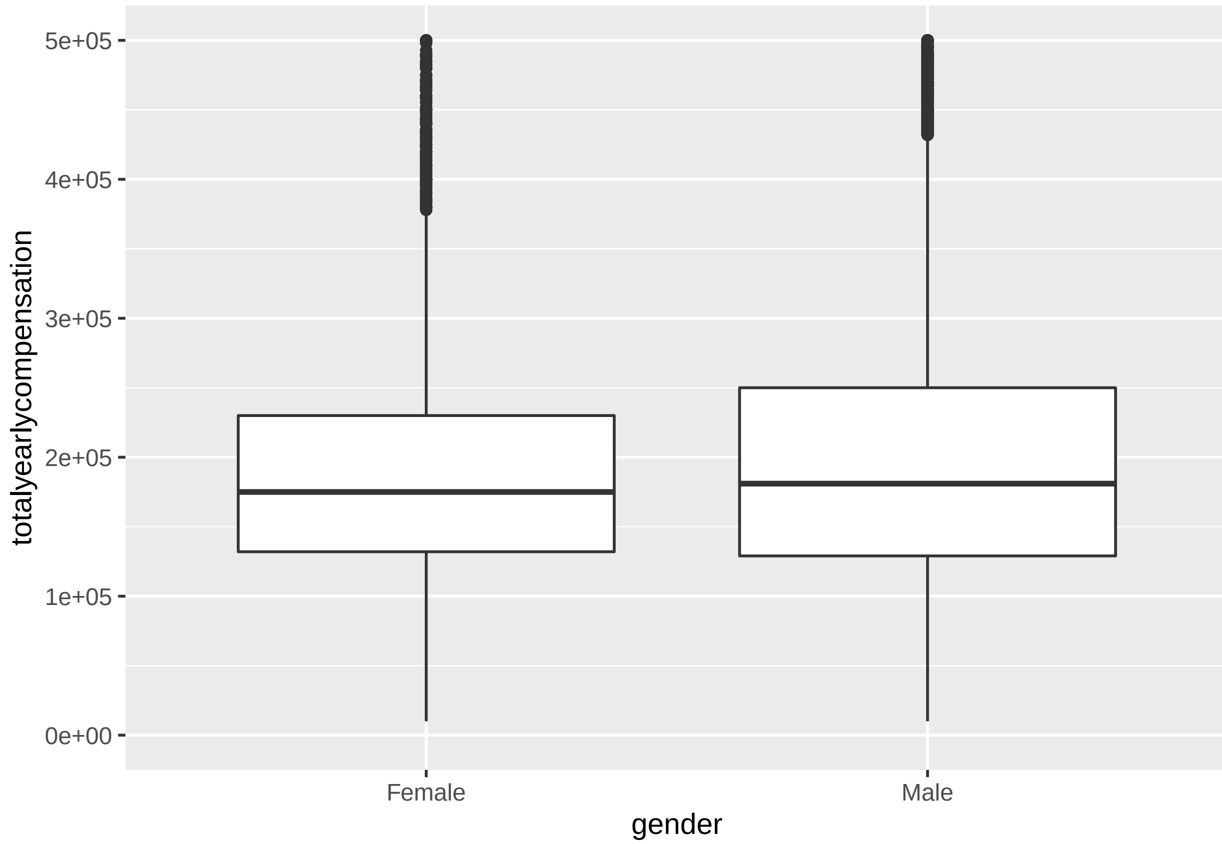
There are clear data issues with Gender variable, with a single entry of gender "Title: Senior Software Engineer" and 19540 NAs and 400 Others, a full 1/3 of entries are unavailable.

Having said that, a simple outliers analysis below indicate a small but perceptible skew towards higher salaries in Males in comparison to Females.



Difference between averages are too small to derive a meaningful conclusion, if there is an impact it is not visible on elementary statistics nor on a plot.

```
## Warning: Removed 1296 rows containing non-finite values (stat_boxplot).
```



We won't be using this variable as part of our predictive model as univariate analysis did not lead us to conclude it can have an impact.

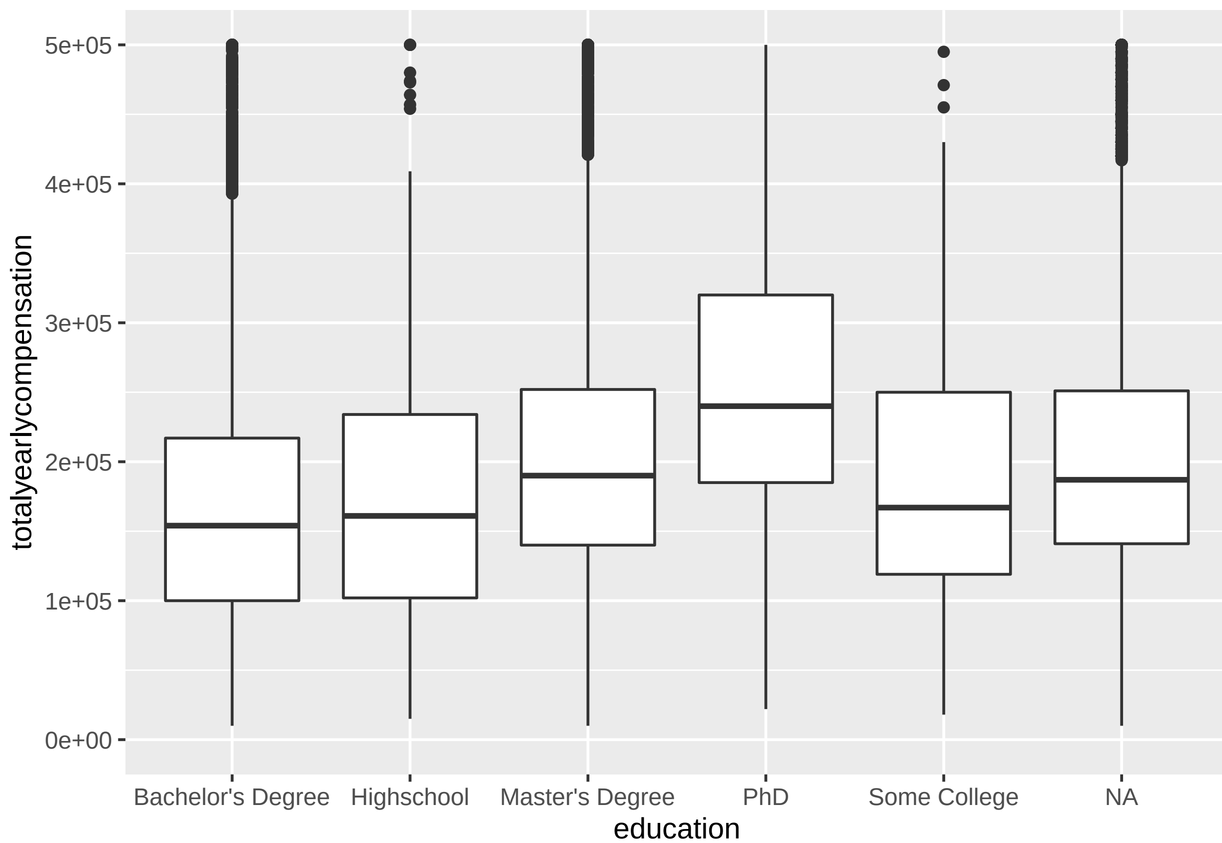
4.1.6 Education

Education vairables are masters_degree, bachelors_degree, doctorate_degree, highschool, some_college, and education.

```
## # A tibble: 6 x 4
##   education         mean_comp   cnt   stdev
##   <chr>                 <dbl> <int>   <dbl>
## 1 PhD                 291866.  1703 150202.
## 2 <NA>                225307. 31954 141072.
## 3 Master's Degree     220731. 15391 138376.
## 4 Some College        210121.   355 129397.
## 5 Highschool          187731.   320 121208.
## 6 Bachelor's Degree   177845. 12601 117969.
```

About 50 percent of all compensation entries education variables are missing as NA entries above. We will not be using education as a predictive variable as significant lack of available data reduced its usefulness.

```
## Warning: Removed 1296 rows containing non-finite values (stat_boxplot).
```



We won't be using this variable as part of our predictive model as univariate analysis did not lead us to conclude it can have an impact.

4.1.7 Race

Race variables which are race_asian, race_white, race_two_or_more, race_black, race_hispanic and race are to capture voluntary disclosures of the submitters race. From limited academic research [3,4] there is outstanding research questions regarding race to have an impact on total annual compensation for tech workers. As field is rapidly evolving, conclusions are not fully fleshed out and supported yet.

```
## # A tibble: 6 x 4
##   race        mean_comp   cnt   stdev
##   <chr>           <dbl> <int>   <dbl>
## 1 <NA>          226405. 39897 139596.
## 2 White         206294.  8032 132138.
## 3 Two Or More   204652.   804 127166.
## 4 Asian         193325. 11772 136264.
## 5 Hispanic      189702.  1129 106816.
## 6 Black         181325.   690 129440.
```

However 2/3 of data is missing with about 40,000 NAs which reduces the usability of this variable for us. We won't be using it as part of our predictive model due to majority of it being unreported.

4.1.8 Tag

Tags are variables created by levels.fyi to perform Search Engine Optimization (SEO), these synthetic variables are not part of original submitted data and will not be used for prediction purposes. Their exact definitions and methodology of generation is unknown and unreliable.

4.1.9 Otherdetails

Otherdetails is another categorical variable created by levels.fyi to capture comments by submitters, without a clear definition nor methodology of creation. We will not be using Otherdetails as part of our prediction model.

4.2 Numerical Variables

There are 12 numerical independent variables, out of which only two of them are numerical variables (yearsofexperience and yearsatcompany) of interest to us.

Remaining ten only contain 0,1 or NA as variables, they are an artifact of CSV conversion process. They were meant to indicate if submitter response are affirmative or negative for a particular data column. They are already dealt with as part of race, gender and education aggregated categorical variables above. These categorical variables parse Boolean responses in those fields and aggregate them to final combined variables.

4.2.1 Yearsofexperience

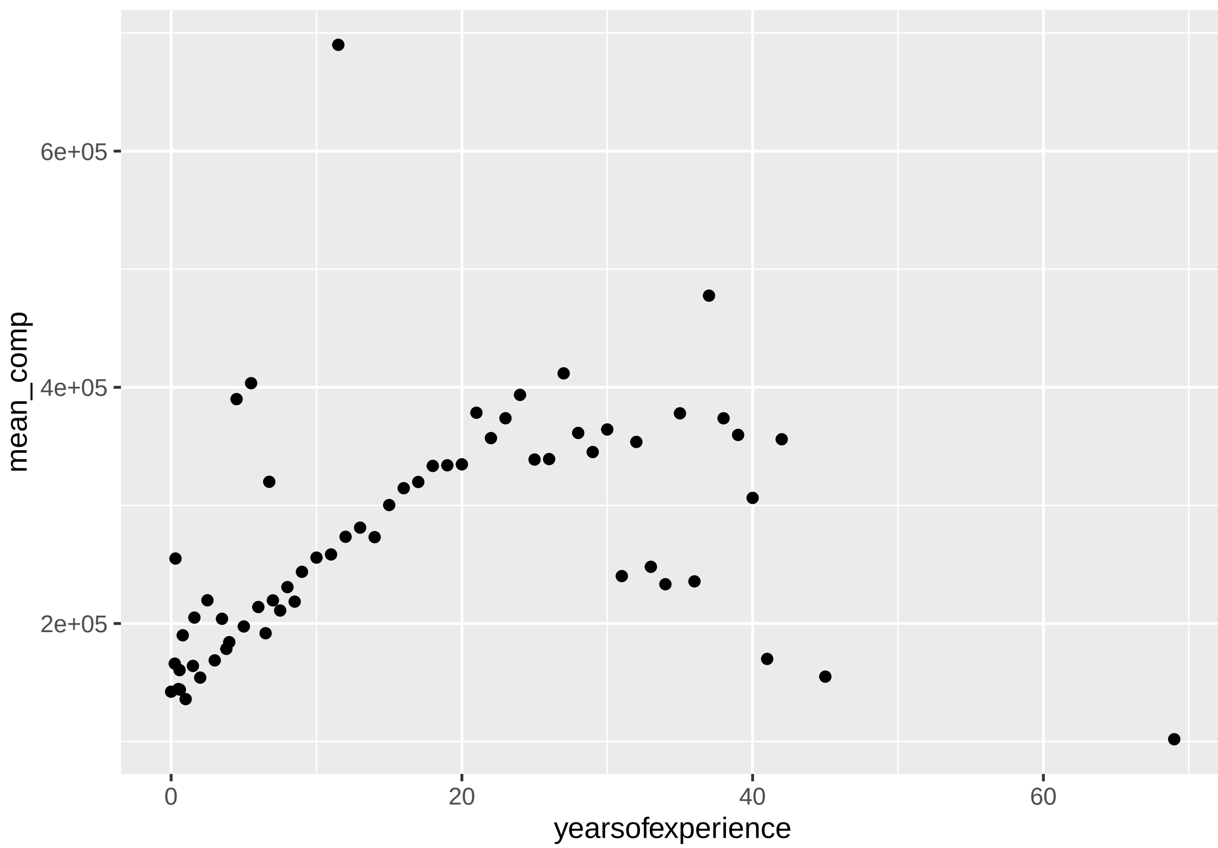
Yearsofexperience is the variable indicating experience candidate has at the time of compensation submission. It is a numeric variable containing fractional values such as 1.5 as well as full integers.

Let us take a look if we can see a relationship between yearsofexperience and totalyearlycompensation as below:

```
## # A tibble: 63 x 4
##    yearsofexperience mean_comp   cnt   stdev
##                <dbl>     <dbl> <int>   <dbl>
##  1              11.5    690000     1      NA
##  2              37      477600     5 380469.
##  3              27      411795.   39 299307.
##  4               5.5    403500     2 231224.
##  5              24      393578.  116 495148.
##  6               4.5    390000     1      NA
##  7              21      378462.  186 268544.
##  8              35      377967.   30 250645.
##  9              23      373786.  145 266405.
## 10              38      373750     4 196612.
## # ... with 53 more rows
```

There seems to be a relationship between but it is not clear in tabular format how it actually it is working, does compensation increase with years of experience or other way around?

Let us take a look at an x-y graph as below:



There are some outliers but we are able to see an almost linear relationship between 0 to 20 years is easy to spot on the graph above. Total compensation seems to plateau at/around 20 years of experience, in line with some outstanding research regarding age-ism in tech sector [5].

We will be using yearsofexperience as a predictor variable.

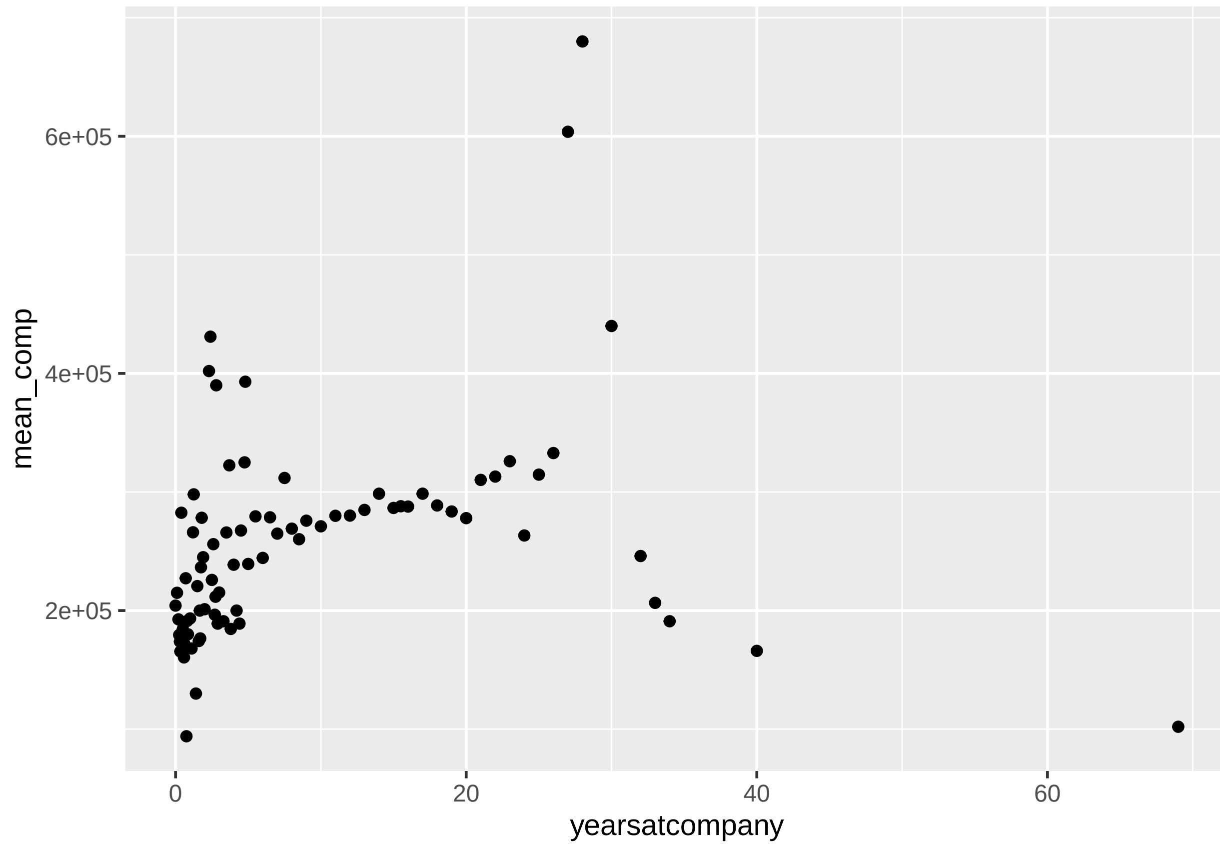
4.2.2 Yearsatcompany

Yearsatcompany is the variable indicating the years of experience submitter has in the firm compensation data relates to.

Is there a relationship between years spent at a company and total annual compensation, let's see in tabular firm if we can glean an easy to spot relationship:

```
## # A tibble: 81 x 4
##    yearsatcompany mean_comp   cnt   stdev
##             <dbl>     <dbl> <int>   <dbl>
##  1          28      680000      2 502046.
##  2          27      603800      5 465798.
##  3          30      440000      2 169706.
##  4           2.4    431000      2 288500.
##  5           2.3    402000      2 209304.
##  6           4.8    393000      1     NA
##  7           2.8    390000      1     NA
##  8          26      332833.     6 165022.
##  9          23      325950     20 220516.
## 10           4.75   325000      1     NA
## # ... with 71 more rows
```

If we just take a look at the average values, there appears to be a very small and linear relationship between years of time one spends in a company and the total compensation one receives in the end. However we can also easily observe that number of data points for 25+ years experience is very sparse.



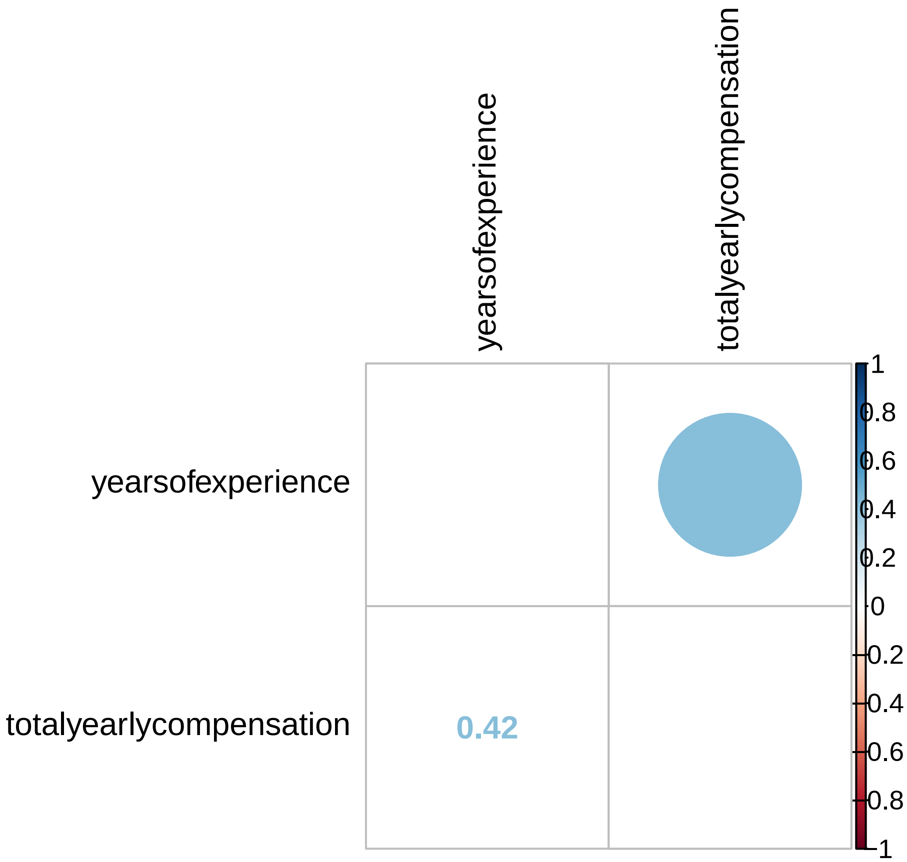
There appears to be a low magnitude linear relationship starting from 5 years of work experience to 20 years, impact seems to be minor. As a result, we won't be using it as part of our predictive model.

4.2.3 Numerical Variable Correlations

Numerical variable correlations are important considerations when we are looking into impact of predictive power of independent variables.

```
only_num_vars <- raw_data_sanitized %>% select(yearsofexperience, yearsatcompany,
                                               totalyearlycompensation)
corr_values <- cor(only_num_vars, use = "pairwise.complete.obs")
corr_matrix <- as.matrix(corr_values[,"totalyearlycompensation"])

corr_high <- names(which(apply(corr_matrix,1, function(x) abs(x) > 0.25)))
cor_num_var <- corr_values[corr_high, corr_high]
corrplot.mixed(cor_num_var, tl.col = "black", tl.pos = "lt")
```



As we can see above, yearsofexperience is the only variable demonstrating high correlation with totalyearlycompensation, we will be using it in our predictive model.

This finding supports our exploratory analysis regarding yearsatcompany, supporting our preliminary conclusion to leave it out of our predictive model.

5. Model training

We will be developing machine learning algorithm for predicting totalyearlycompensation variable utilizing yearsofexperience, company and location variables.

Using linear regression as the base model, we will build two additional machine learning models of greater complexity: ElasticNet and XGBoost to see which one will have the lowest RMSE score as our comparison metric.

We will NOT be using test data for training purposes, our RMSE comparison to determine which model performs the best will be strictly conducted on test data set predictions while using models trained on training data as industry best practice.

We will be using cross-validation and caret package's tuning ability to optimize the final model to its full capabilities.

5.1 Downsampling

Even with doParallels library achieving CPU threading and GPU acceleration on XGBoost, full training cycle takes many hours and requires over 50GB of memory. Recognizing these limitations, I used randomly sampled results which lead to some interesting insights about data size and algorithmic efficiency, which will be discussed fully in conclusion section.

5.2 Training and Test Set Creation

We will divide data into training and test sets, 90 percent will be in training and 10 percent will be in test dataframes respectively.

In order to ensure that categorical variables of interest (company and location) do not pose a mismatch issue when trained model is predicting on test-set, we are using a semi-join below to ensure test dataset only contains the same categorical variables that exist in training set.

```
# Create train and test datasets -- 0.9 train, 0.1 test
# By changing size variable to 1000, 10000 and 30000
# Report results can be replicated

set.seed(1,sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
size <- nrow(raw_data_sanitized)

relevant_data <- raw_data_sanitized %>%
  select(totalyearlycompensation, yearsofexperience, company, location)
complete_data <- relevant_data[sample(nrow(raw_data_sanitized), size),]

test_index <- createDataPartition(y=complete_data$totalyearlycompensation,
                                  times=1, p=0.1, list=FALSE)

train_set <- complete_data[-test_index,]
temp <- complete_data[test_index,]

test_set <- temp %>%
  semi_join(train_set, by = "company") %>%
  semi_join(train_set, by = "location")

# Add rows removed from test set back into train set
removed <- anti_join(temp, test_set)
```

```
## Joining, by = c("totalyearlycompensation", "yearsofexperience", "company", "location")
```

```
train_set <- rbind(train_set, removed)
```

5.3 Base Model: Linear regression

Considering we are looking for a numerical answer, linear regression is the most obvious place to start building our base model where we compare all other models to.

Following code produces our base model.

```
base_lm <- lm(totalyearlycompensation ~ yearsofexperience  +
                company + location,  data=train_set)
lm_predictions <- predict(base_lm, test_set, type="response")
lm_rmse <- RMSE(lm_predictions,test_set$totalyearlycompensation)
```

RMSE for the linear regression base model is $101558.60.

5.4 Optimized Models

We will be using optimizing via caret package's train function, all of which will use the following training control that ensures there will be two cross validations conducted in each train run.

This is here parallelization comes into play, as with this large datasize, running a long running training multiple times is averted due to multi-threading.

However as pointed below, parallelization is known to introduce reproducibility concerns, if facing it, please run as sequential. Unfortunately this problem originates from upstream package authors and caret offers no complete solution to ensure this problem can be solved at trainControl level or at any configurable caret parameter level [8].

```
train_control_default <- caret::trainControl(
  method = "cv",
  number=2,
  verboseIter = FALSE, # no training log
  allowParallel = TRUE, # FALSE for reproducible results
  savePredictions="final"
)
```

5.4.1 Optimized Linear Regression

We will optimize linear regression model via running multiple cross-validation sets to ensure we are obtaining the best possible weights mimizing RMSE.

As there are no tunable hyper-parameters except the intercept, we anticipate it to match the base linear regression case.

Code below will accomplish this task:

```
lm_optimized <- train(totalyearlycompensation ~ yearsofexperience  + company + location,
                      data=train_set,
                      method="lm",
                      na.action  = na.pass,
                      trControl = train_control_default)

lm_optimized_predictions <- predict(lm_optimized, test_set, type="raw")
lm_optimized_rmse <- RMSE(lm_optimized_predictions,test_set$totalyearlycompensation)
```

As predicted, optimized linear regression result matches the base linear case in line with theory.

Same RMSE score as the linear regression case, $101558.60.

5.4.2 Elastic Net Regression

Elastic net model is a regularized regression model combining both Lasso and Ridge regression's L1 and L2 penalties.

Considering we have a very large number of categorical variables, L1 penalty we expect to help us reduce feature-space, while L2 helps us improve accuracy by bringing results closer to a mean value.

Using default optimization parameters of Caret as below:

```
glmnet_optimized <- train(totalyearlycompensation ~ yearsofexperience + company + location,
                          data=train_set,
                          method="glmnet",
                          na.action  = na.pass,
                          trControl = train_control_default)
```

```
glmnet_optimized_predictions <- predict(glmnet_optimized, test_set, type="raw")
glmnet_optimized_rmse <-
  RMSE(glmnet_optimized_predictions,test_set$totalyearlycompensation)
```

Optimized model results are about the same as the base case with an RMSE of $101565.60.

5.4.3 Gradient Boost Tree

Gradient boosting is a technique that is designed to take advantage of ensembling a large number of weak predictors (usually decision trees).

On default optimization characteristics with same cross-validation, Gradient Boosted Tree model had the lowest RMSE score (100386.64) in comparison to other models.

In order to optimize the model via customized tuning, I focused on three key tuning parameters:

- nrounds: Variable representing how many boosted trees are added as part of cross-validation.We are using a sequence from 200, to 1000 by increments of 50.
- eta: Controls learning rate, we are changing it from 0.025 to 0.3.
- max_depth: This controls the depth of tree, larger the value greater chance of overfitting due to model complexity.

All other tuning parameters are the defaults selected by XGBoost package.

```
xg_tune_grid <- expand.grid(
  nrounds = seq(from = 200, to = 1000, by = 50),
  eta = c(0.025, 0.05, 0.1, 0.3),
  max_depth = c(2, 3, 4, 5, 6),
  gamma = 0,
  colsample_bytree = 1,
  min_child_weight = 1,
  subsample = 1
)

xg_optimized_model <- train(totalyearlycompensation ~ yearsofexperience  +
                              company + location,
                          data=train_set,
                          trControl = train_control_default,
                          tuneGrid = xg_tune_grid,
                          method = "xgbTree",
                          na.action  = na.pass,
                          tree_method="gpu_hist",
                          verbose = TRUE)

xgboost_optimized_predictions <- predict(xg_optimized_model, test_set, type="raw")
xgboost_optimized_rmse <- RMSE(xgboost_optimized_predictions,test_set$totalyearlycompensation)
```

Optimized XGBoost model's RMSE is $98605.52, lowest in comparison to other models. It is also the fastest to train with such a large dataset, thanks to GPU acceleration.

6. Results

Utilizing caret and its training control and tuning capabilities to optimize all three models below.

When possible utilized doParallel package for CPU intensive models (lm and glmnet) and GPU optimized version of XGBoost package in order to be able to process this large dataset utilizing a modest home computer.

In order to build the case for best model, I developed my machine learning models with progressively using more data starting with 1,000 data entries to culminating with the entire data set (62,000+).

I have looked into comparing results of three machine learning models: - Baseline: Linear regression, using lm package. - Generalized Regression: Lasso and Ridge regression combined, using glmnet package. - Gradient Boosting: Gradient boosted decision trees, using xgboost package.

6.1 Model Results

I found best model to be XGBoost with an RMSE of $98605.51, considering our maximum yearly compensation is $4,680,000+, with a standard deviation of $137977.1, our optimized XGBoost model's RMSE is about 71% of the standard deviation of complete data set.

Below, we can find the comparison between all three models trained:

```
knitr::kable(model_results, caption="All models and results")
```

Table 1: All models and results

method	rmse
Linear Regression	101558.60
ElasticNet Regression	101565.60
XGBoost Default Tuning	100386.64
XGBoost Custom Tuned	98605.51

6.2 Important Variables

As we look into variables we utilized in the model through varImp command, we can obtain a list of critical list of variables that shed a very strong light on total compensation for tech employees.

20 top variables are as following:

- yearsofexperience
- company
  - Facebook
  - Google
  - Netflix
  - Snap
  - Apple
  - Uber
  - LinkedIn
- location
  - San Francisco,CA
  - Seattle, WA
  - Bengalore, India
  - Mountain View, CA
  - Menlo Park, CA
  - Sunnyvale, CA
  - Cupertino, CA
  - New York, NY
  - Hyderabad, India
  - Palo Alto, CA
  - San Jose, CA
  - Bengaluru, India

This finding aligns very well with domain research about concentrated tech cities and their impact on STEM salaries, demonstrating how our gradient boosted method gleaned the importance of top tech cities and top tech firms as core ingredients of high compensation with years of experience driving as the most important variable.

7. Conclusion

We can see from above analysis and modelling results that yearsofexperience, company and location are the primary drivers of total compensation of a tech worker.

From our modelling efforts, we can easily conclude that XGBoost is the clear winner in algorithmic comparison held between ElasticNet, Linear Regression and Gradient Boosted Trees on our full levels.fyi dataset.

We must consider the limitations of the study such as, data quality concerns on many categorical variables such as titles, levels and others made direct use of them quite difficult to achieve. Also missing data on several other categorical variables such as gender, race and education also limited their utility for our modelling approach.

Another key concern we must take into account is the self-reported nature of levels.fyi dataset, potentially introducing bias that we are unable to account for as compensation data is one of the most heavily guarded and protected secret of individuals and companies alike. As far as my research goes, I could not locate another data source to use as a comparison to be able to validate levels.fyi dataset.

Next step in enhancing accuracy of our predictive model is to focus on data clean-up efforts to clean some of the categorical variables and develop mechanisms to deal with missing values in other categorical values. Both of these efforts require domain knowledge such as actual levels within the firms as well as context specific methodologies to account for missing values for education and/or race and/or gender variables. Extant academic research imply relationship between these variables and total annual compensation, which means if we are able to overcome data quality concerns, they should aid in building an even more accurate predictive model.

8. References

[1] Segovia-Pérez, M., Castro Núñez, R.B., Santero Sánchez, R. and Laguna Sánchez, P. (2020), Being a woman in an ICT job: an analysis of the gender pay gap and discrimination in Spain. New Technology, Work and Employment, 35: 20-39. https://doi.org/10.1111/ntwe.12145

[2] Goldin, Claudia, Sari Pekkala Kerr, Claudia Olivetti, and Erling Barth. 2017. "The Expanding Gender Earnings Gap: Evidence from the LEHD-2000 Census." American Economic Review, 107 (5): 110-14. https://doi.org/10.1257/aer.p20171065

[3] Jennifer C. Lee, Employment and Earnings in High-Tech Ethnic Niches, Social Forces, Volume 91, Issue 3, March 2013, Pages 747–784, https://doi.org/10.1093/sf/sos199

[4] Dreher GF, Lee J-Y, Clerkin TA. Mobility and Cash Compensation: The Moderating Effects of Gender, Race, and Executive Search Firms. Journal of Management. 2011;37(3):651-681. https://doi.org/10.1177/0149206310365728

[5] S. Baltes, G. Park and A. Serebrenik, "Is 40 the New 60? How Popular Media Portrays the Employability of Older Software Developers," in IEEE Software, vol. 37, no. 6, pp. 26-31, Nov.-Dec. 2020, doi: 10.1109/MS.2020.3014178.

[6] Juba, B. and H. S. Le, Precision-Recall Versus Accuracy and the Role of Large Data Sets, Association for the Advancement of Artificial Intelligence, 2018.

[7] "How much data is required for machine learning?" https://www.quora.com/How-much-data-is-required-for-machine-learning

[8] "Caret documentation on reproducibility" https://topepo.github.io/caret/model-training-and-tuning.html#notes-on-reproducibility