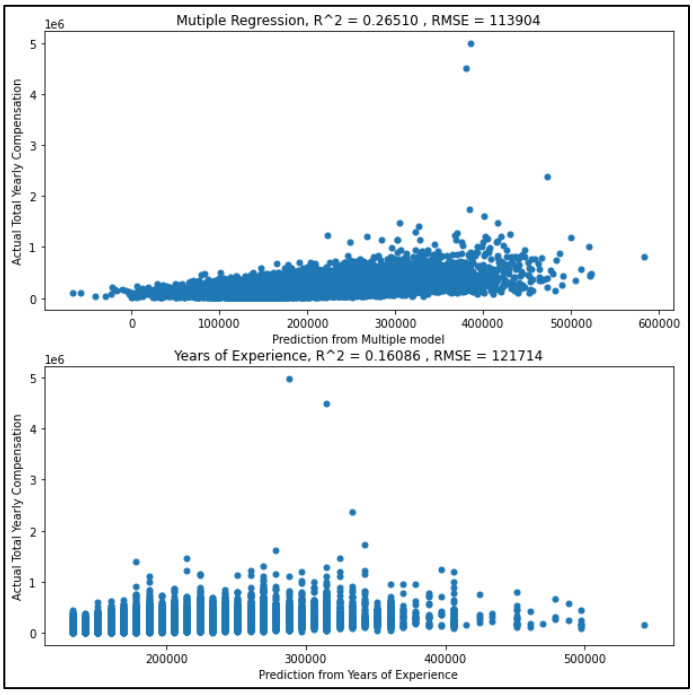
Tomisin Adeyemi – FML HW 2

Preamble:

For questions 2 to 5, I used a seed value of 1234 (set using `np.random.seed(state)` and `random_state` in `train_test_split()`), and a `test_size` of 0.4 (40%). Choosing different seeds and test sizes also resulted in different optimal lambdas for both question 2 & 3, I ended up sticking with seed and test size combo that gave lower RMSE scores for both questions and shrunk lambdas to zero in question 3. For questions 1 to 4, the dataset used was derived as follows: the first 3 features of the dataset were dropped because of their qualitative nature, and variables 5-7 were also dropped because of extremely high correlations to annual compensation. *All* the NaN values remaining after were dropped, leaving ~20k data points. The `Education` and `Race` columns were removed because of the encoded variables, and from the encoded variables, `Highschool` and `Race_Two_Or_More` were removed to make sure the data is not overdetermined. Lastly the "other" values in "Gender" were dropped, and the Male/Female values were encoded as follows: Male – 0, Female – 1.

1. Using multiple linear regression: What is the best predictor of total annual compensation, how much variance is explained by this predictor vs. the full multiple regression model?

I extracted the X and y values from the processed dataset described in the Preamble. For both the simple and multiple linear regression, I used the scikit-learn function LinearRegression(), to create models and evaluate them. The R^2 value was computed using the score() function in scikit-learn as well. To do the simple linear regression, I looped over all the feature variables, create models, computed and printed their R^2 value. The best predictor was determined by visually looking at the values that were printed.



The question asks to explain the *variance* explained by predictors, this is described by the R^2 value, which described the proportion of the variance in the outcome variable that a predictor explains. For the purpose of this question, the data is not split into train and test sets.

The best single predictor of total annual compensation is **years of experience**, accounting for **16.09%** of the variance in total annual compensation. However, the full multiple regression model accounts for **26.51%** of the variance in total annual compensation, performing *better* than years of experience. The figure to the right shows plots of Actual vs. Predicted years of experience for both predictors.

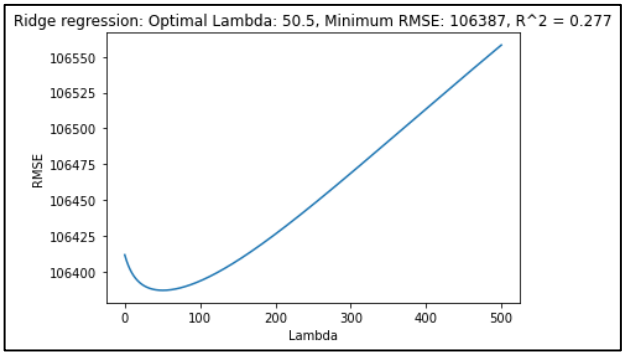
It is not surprising that the full multiple regression model explains more of the variance in total annual compensation than a single predictor, as there are other variables in the dataset that could perhaps have a link to total annual compensation.

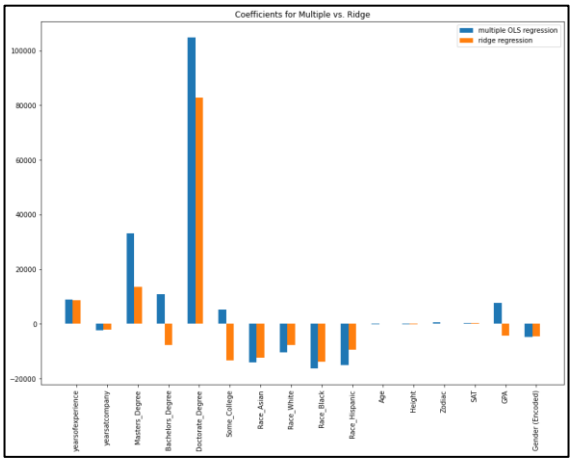
2. Using ridge regression to do the same as in 1): How does the model change or improve compared to OLS? What is the optimal lambda?

I first split the data into train and test sets, then tested lambda values from 0 to 500, recording the root mean square error for each lambda. The Ridge regression was implemented using the Ridge() function in scikit-learn. The optimal lambda was chosen by picking the lambda with the smallest root mean square error. To compare the Ridge Regression to OLS, I compared the R^2 and RMSE scores, as well as the coefficients for each variable.

Another way to perform validation could have been Cross Fold Validation, but I wanted to stick to simpler regular train test split. I also decided to minimize RMSE rather than R^2, because changes in R^2 would not be as significant as changes in RMSE.

The optimal lambda was found to be **50.5**, resulting in an RMSE of 106387 and R^2 of 0.277, performing way better than OLS in terms of RMSE, but only a little better in terms of R^2. The Ridge regularization shrunk the *magnitude* of most of the coefficients from OLS. The illustration for both the lambda and coefficients is shown below.





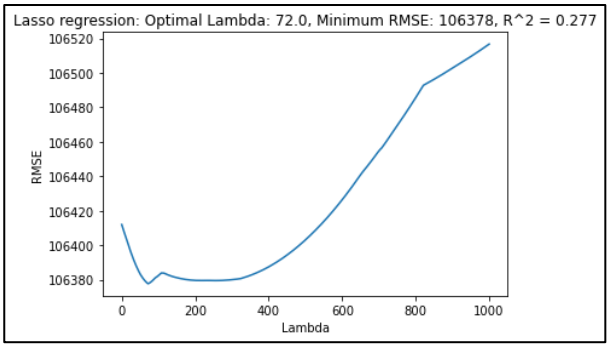
First, I assumed that the surge in performance in RMSE and slightly better R^2 could be because the OLS was not split into train and test sets, and I was right. On redoing the OLS with train and test sets, the RMSE was 106412 and the R^2 0.2764, which (are still smaller), BUT, very similar to the values obtained with Ridge regression. Thus, one can argue that the Ridge regularization does not do too much to improve the model, in an ideal world I would expect an optimal lambda closer to 0.

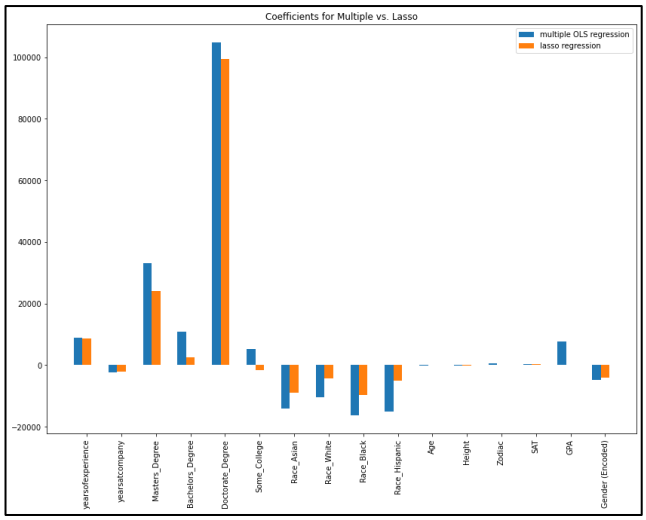
3. Using Lasso regression to do the same as in 1): How does the model change now? How many of the predictor betas are shrunk to exactly 0? What is the optimal lambda now?

A similar approach was used to the Ridge Regression in Part 2. I first split the data into train and test sets, then tested lambda values from 0 to 500, recording the root mean square error for each lambda. The Lasso regression was implemented using the Lasso() function in scikit-learn. The optimal lambda was chosen by picking the lambda with the smallest root mean square error. To compare the Lasso Regression to OLS, I compared the R^2 and RMSE scores, as well as the coefficients for each variable.

The possible alternatives are the same as Part 2 above.

The optimal lambda was found to be 72, resulting in an RMSE of 106378 and R^2 of 0.277, again performing better than OLS in terms of RMSE, and slightly better in terms of R^2. The Lasso regularization shrunk the magnitude of *all* of the coefficients from OLS except for SAT. The coefficients of GPA and Age were both shrunk to exactly 0. The illustration for both the lambda and coefficients is shown below:

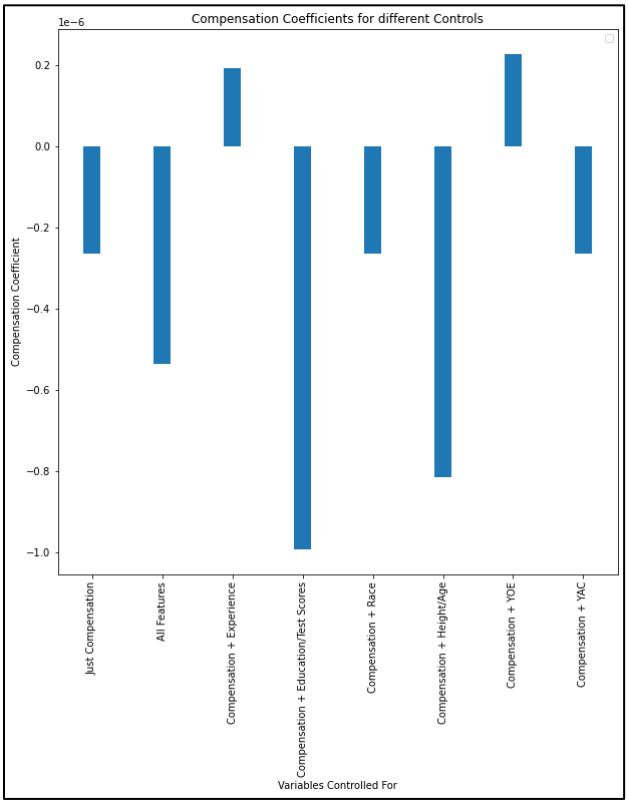




Like Part 2 above, the drastic difference in RMSE could be attributed to the lack of train and test sets in part 1, but on comparison to the train and test version of OLS, the R^2 is not that much different and neither is the RMSE. (In fact, the R^2 value is the exact same as in Part 2 above!) However, the shrinking of SAT and Age to 0 could mean they don't really have too much effect on predicting total annual compensation. Although, I would expected more variables to shrink to 0, especially Zodiac, Race and some education columns, as they had low R^2 values in part 1. Finally, the increase in the coefficient magnitude for SAT could be to cover up for the fact that GPA was dropped to 0, as these 2 variables are highly correlated.

4. There is controversy as to the existence of a male/female gender pay gap in tech job compensation. Build a logistic regression model (with gender as the outcome variable) to see if there is an appreciable beta associated with total annual compensation with and without controlling for other factors.

First, I looked at the number of Female & Male data points, and as expected, there was high class imbalance with approximately 20% women and 80% men. Because of this, I made sure to indicate that the class weights should be balanced in the LogisticRegression function. To control for different facors, I tried a mix of different predictors, and recorded the coefficient for total annual compensation in the logistic regression model. Finally, I plotted the coefficients for compensation for each variable that I controlled for.



As a side note, I made sure to encode the female variables as the positive (1) class as it was the minority. This way I was able to get a better idea of how good my Logistic Regression model was doing at classifying women.

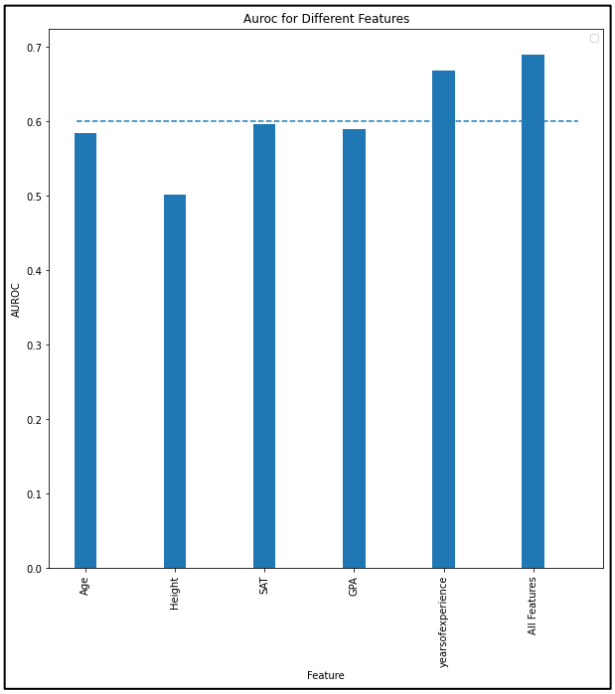
Given the orders of magnitude shown in the graph above, there is no particularly appreciable beta associated with total annual compensation.

The betas associated with total annual compensation don't change much, but their values can still be interpreted as follows. Coefficients represent the log of the odds ratio, so increasing total yearly compensation by $1, multiplies the log odds of being paid the same pay as a woman by $e\hat{}\beta$, where β is the coefficient of total annual compensation. Taking the beta of Compensation + Experience, which has the highest order of magnitude, the beta is -9.927201529176497e-07. This means that an increase in salary by $1 multiplies the odds of being a woman by $e^{-9.927201529176497*10\hat{}-7} \approx 1.00$, meaning the odds stay the same. This applies to all the betas.

5. Build a logistic regression model to see if you can predict high and low pay from years of relevant experience, age, height, SAT score and GPA, respectively.

The dataset I used for this question is different from the one used in the previous 4 questions. I used the entire original dataset (no nan values dropped), as there were no Nan values for the variables used in this question. Then, I used a median split to determine if a given salary was high or low pay. Then for each, variable, I performed a logistic regression, and kept note of the area under the ROC curve. At the end, I plotted the values for the AUROC for each variable used.

I decided to use all the data because the number of features were more restricted, so there was a possibility of less nan values (and it turned out to be true!). In addition, I decided to use the area under the ROC curve as my metric as the classes being predicted (high vs low pay) were properly balanced, so the AUROC would tell us how properly each predictor can classify high/low pay. I also decided to use a threshold of 0.6 to determine if a predictor to correctly classify or not, i.e. AUROC >0.6 means good predictor. I chose this because a value of 0.5 meant the model is as good as randomly guessing, and I did not want to pick a threshold that was too high given that this is real (not synthetic) data.

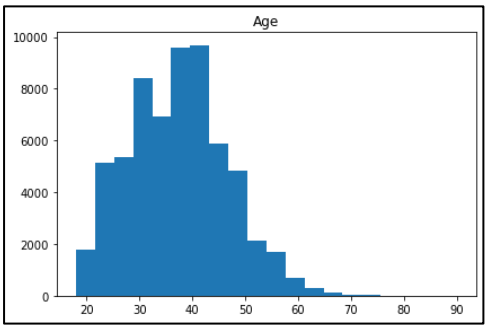


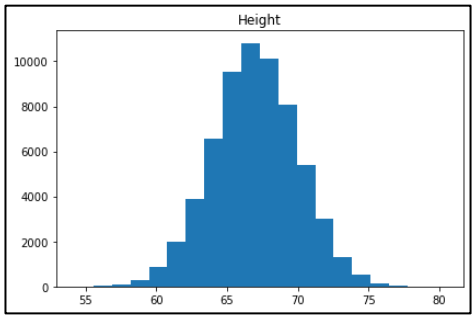
The graph of the AUROC for each predictor is shown on the right. Most predictors fell under the threshold, but years of experience seems like it would be a good predictor for high/low pay. (An ensemble of all the features is also a good predictor)

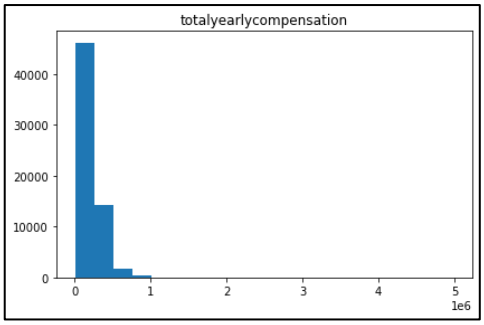
It makes sense that years of experience would be a good indicator of pay, as the longer someone works, the more likely they are to acquire more skill, get promotions, etc.

Extra credit: a) Is salary, height or age normally distributed? Does this surprise you? Why or why not?

Using the same dataset in part 5, I plotted histograms for each of the variables mentioned in the question. This is because a histogram helps visualize how data is distributed. The histograms are below:







From the histograms, we can see that only height is normally distributed. This makes some sense: there is no benefit/restrictions on certain heights (that I know of??) in the tech industry, thus it makes sense for the height to be distributed normally. The age is skewed to the left, which makes sense because older people retire, so there'd be less of them in the industry. Finally, yearly compensation is also skewed left, meaning that it is more common for people in the tech industry to earn below a million dollars. This also makes sense as mostly people in very high positions would be earning more than a million dollars.