Personalized Data Science Career Advice

How a data scientist can improve their expected salary

Hayden Sather

A research paper for the class of
DSCI-403

The Data Science Department of
Colorado School of Mines
March 19, 2020

I. Overview

A. Motivation

Data Science has been called the "Sexiest Job of the 21st century" by the Harvard Business Review (Brynjolfsson, 2017), and for good reason. The pay is exceptional, the benefits are unmatched, and the flexibility is phenomenal. Due to this, there has been a massive explosion in the number of people who want to pursue this career path. This unfortunately caused data science to be one of the most competitive jobs in our society. It seems as if there are endless and vastly different requirements for each of the jobs, so it is often difficult to know how qualified a candidate actually is. This makes it hard to know how much a candidate can expect to be paid, and even harder to find out what the next technology they should learn should be. This project aims to fix that by providing three insights to a data science candidate. First, it will tell them how much money they can expect to be paid depending on their current qualifications. Second, it will tell them what technology they should learn next to maximize their expected salary. Third, it will tell them what salary they can expect to make if they do decide to learn that next technology and also provide resources to learn it.

B. Dataset

The data set that is used is "Data Analyst/Scientist jobs" from Kaggle (Gutstadt, 2020). This dataset was obtained by scraped Glassdoor for all of the jobs related to Data Science and Data Analyst roles in the United States posted in 2020. Each row is a separate job posting that contains general information about the position.

C. Problem Statement

The purpose of this project is to provide advice to a new data scientist who wants to improve their chances at landing a job with a high salary. Although money is only part of what is important in a job, it is a large driving factor for the quality of life that one lives. The problem that will be solved is that it is very difficult for new data scientists to find out exactly what they need to do to be hired at a good job. Should they change locations? Should they pick up another certification? Should they apply for different roles? This project aims to provide a job-seeker with personalised advice for the best move that they could take to improve their chances of getting a high salary. It will first tell the user what their expected salary is, then it will tell them what their next step should be (Learning a specific technology, moving to a different city), then it will tell them what their expected salary is after taking that step.

II. Related Work

A. Predicting Salaries with Decision Trees

The research paper, "Predicting the Probability and Salary to Get Data Science Job in Top Companies" (Situ et al., 2017), is very similar to what this project achieves. Similarly to mine, they also scraped website data from hiring sources, in their case, Glassdoor and LinkedIn. They used this information to predict the probability that an applicant gets the position at a company given the company and salary range. They used gradient boosting decision trees and logistic regression to calculate these probabilities. This provides a great method to determine which companies one should apply for and at what salary ranges. This shows that machine learning techniques can be used to predict the salary of data science jobs with Glassdoor information as predictors. This project will expand on this by providing the applicant with the next step in advancing their career to increase their salary, along with the salary that they can expect to get given that they take that step.

B. Predicting Salaries with CNNs

Another project that is similar to mine is the chapter "Modelling and Predicting Individual Salaries in United Kingdom with Graph Convolutional Network" in the book "Hybrid Intelligent Systems" (Chen et al., 2019). Similarly to the dataset in this project, they also pulled job listings from online job-hunting websites in an attempt to predict the salary of the positions. This project was much more focused on creating an intricate representation for the job postings. They created a graph-like representation of all of the postings which could include much more detailed information about how each of the jobs were related. This allowed for them to have much more detailed predictors, which in turn gave them a very high accuracy of their model. They were only interested in regression of the salary and did not expand further after predicting it like this project does. Their project uses data from all job types while mine is just for data scientists, but the methodologies should all be similar. It was interesting that they decided to use a convolutional neural network as those are commonly only for computer vision tasks. This allowed for them to find relationships between different job postings and predictors more efficiently. A convolutional neural network because it is overkill for this project and sufficiently accurate results should be possible without one. Additionally, it would be very complicated to interpret. This project is a great indicator that predicting the salary of a job posting can be done with great accuracy with predictors that come from a job website using a neural network. This shows that the representations that are being used in this paper are sufficient and more factors will not have to be pulled in to get a good prediction.

C. Predicting Salaries With Traditional Methods

The article, "Fun with Numbers: Alternative Models for Predicting Salary Levels" (Johnson, 1987), is a fascinating application of machine learning in the early days. This was before many modern machine learning techniques became popular, so more traditional methods such as linear regression were used to create the models, instead. This research team created multiple models to predict the salary of a professor given their gender, tenure, and field of study at an undisclosed university. Their main focus was to discover if there is a pay gap in genders for equal roles as professors. The most interesting thing that they did was train three different

models: All professors, only male professors, and only female professors. By separating these datasets and training individual models on them, they were able to clearly show that given everything else equal, male professors still made more money than females. This provided insight on how to handle the problem of different cities providing different base salaries. A model will be trained on all of the data, then for each candidate, it will estimate their salary for every city and return that information along with the average of all of them. This will remove one of the biggest confounding variables and each estimated average salary for each of the cities will be displayed individually.

III. Data Aquisition

The data set that is used is ”Data Analyst/Scientist jobs” from Kaggle (Gutstadt, 2020). This dataset was obtained by scraped Glassdoor for all 26,980 jobs related to Data Science and Data Analyst roles in 2020. Each row is a separate job posting. The columns are Job Title, Salary, Description, Rating, Company Name, Location, Size of Company, and Year Founded. There is quite a lot of preprocessing that must occur, and this will be explained in the next section.

IV. Preprocessing

Since this data was acquired using scraping techniques, not all values will be filled for each of the job postings. The rows with information missing will be simply dropped because they will throw off the algorithm, there is no good way of predicting these values, and there is no shortage of complete data. The most important column will be the description, which often will outline the qualifications required for the job. Qualifications can include having knowledge of technologies, certain degrees, certifications, and various other skills. The description will have to be parsed to find each of the qualifications. This list of technologies was created by combining the results of two websites that state the most important skills for a data scientist to have (DataFlair, 2019), (AIMultiple, 2021). The list will be read in as a text file called qualifications.txt where every qualification will be on its own line, and includes 94 different qualifications that a data scientist can have. This list is very easy to expand for future use with new technologies and certifications. The description will be parsed and checked to see if the listed technologies are included in it. If the technology is present in the description, then it will be marked as required, if not, it will not be required. It is important to note that all possible ways to format each description is considered. First, a regex is used to ignore the case of the technology name. Second, if there is a multiple word name like ”TensorFlow”. All iterations are considered such as ”TensorFlow”, ”Tensor Flow”, and ”Tensor Flow”, due to the different formatting of each technology. This insures that none are missed. All of the technologies will be added as a binary value in a new column. Another step will be converting the Salary column to an int value. It currently has the format of ”$XXK-$XXK (Glassdoor est.)” where the two XX values are the minimum and maximum estimated salaries. The average of these two numbers will be computed after parsing and this result will become a new integer column after deleting the original salary. Also, since the location is in the format ”City, ST”, there are far too many cities for this to be a useful value. So, this will be parsed for just the state, then create dummy variables for the states as a binary value. Another change that is needed is the size of the company. There are 5 different possible sizes in the format ”XX - XX employees”. Finally, there are a handful of columns that will not be useful for predicting the salary with this method. This includes Job Title, Rating, Company Name, Headquarters, Founded, Type of Ownership, Industry, Sector, Revenue, Competitors, and Easy Apply. Although some of these columns may be helpful predictors for salary, for this purpose are excluded. This is because an applicant won’t be able to change any of these values about themselves. The data will be split by putting 80% into a training set and 20% into a testing set.

A. Predictors

1) Sizes: One surprising piece of information found was that the sizes of companies are much less diverse than previously thought. LinkedIn allows for 7 different categories of sizes of companies: However, the distribution of companies is U-shaped. This is probably because of the large number of small startups, and the large number of massive companies, with few in-between. The distribution can be seen in Figure 1. Note that the x-axis is the average of the two max and min bounds, which is helpful later with the regression task.

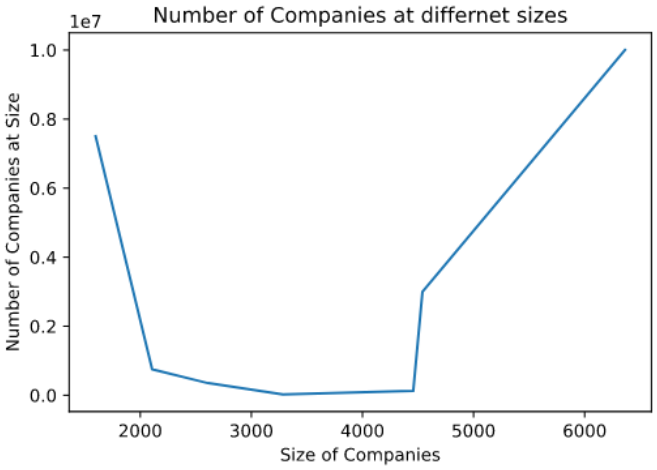

Fig. 1. Distribution of company sizes

2) States: As one could guess, the majority of the jobs come from just a few of the states that have the highest populations of people and also the biggest tech hub cities. Figure 2 shows the top 10 most represented states in this dataset along with the number of jobs that they are offering.

3) Technologies: The next thing that we wanted to see were the distributions of the technologies available to a data scientist, discussed above. As one would expect, some technologies were much more important than others. The distribution of the

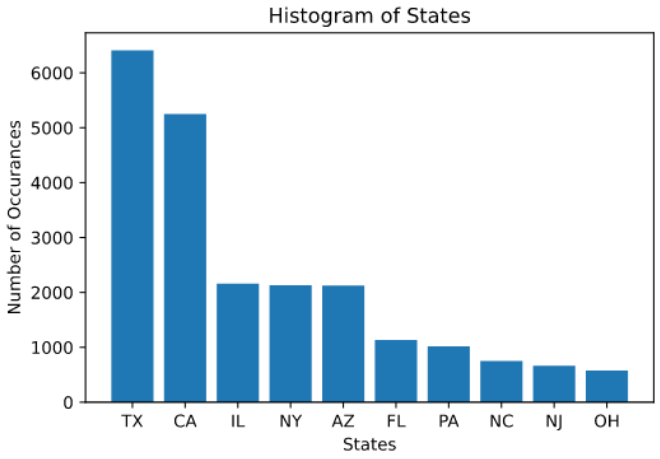

Fig. 2. Distribution of jobs offered from different states

number of times each of the technologies were mentioned is shown below in Figure 3. Only the top 11 technologies, which were the 25% most popular, are pictured in this graph.

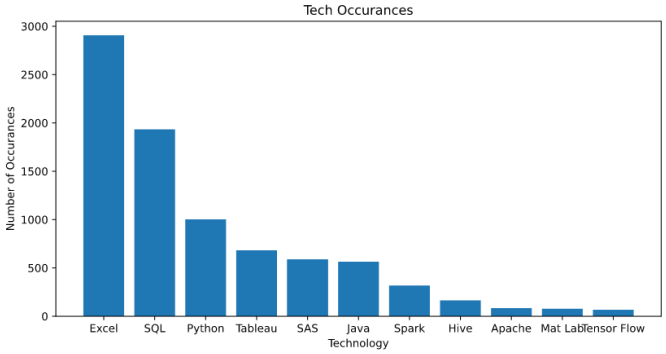

Fig. 3. Distribution of jobs offered from different states

B. Target

The target for this project is to predict the salary of the job postings. The distribution of these salaries is shown below in Figure 4. As can be seen, the salary is non-uniform. It is roughly normal, but is skewed positively.

C. Limitations of Data

The biggest limitation that this technique has is its technique on parsing the description for the technologies. It does not distinguish between required technologies or recommended technologies, which is very prevalent in today's job search. However, it would take a large amount of time to develop a more robust parsing system like that and intuition shows that simply scanning the input for the names is a close enough approximation, given the large number of samples that this data-set has. Also, if a technology, skill, or certification is present in a description for a job posting, it means that is at least of importance to the company. If an applicant has all of the required and recommended qualifications, then it is likely that they will get the job.

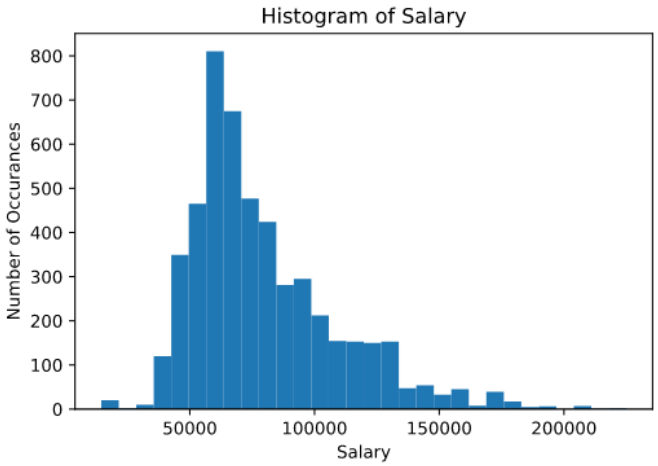

Fig. 4. Distribution of salaries for the posted data science jobs

V. Model Selection

A. Random Forest

The main purpose of this project is to provide a digestable way for a data science job applicant to improve their expected salary in the job market. Therefore, it is important that a model is interpretable so that they can easily understand why the algorithm made the choices that it did. So, the random forest model was chosen. These are easy to interpret, so if the applicant desired to check how the algorithm made its decision, they would be able to follow the logic. Since the predictors are all binary categorical, except for the size of the company which is a discrete number, decision trees excel at modeling these types of data. The model can use each possible value of the categorical data as its own splitting point of sub-trees. The model will be RandomForestRegressor from the Scikit-learn library. This model provides easy functions to fit on a dataframe and test it on new data.

B. Hyperparameter Selection

1) Max Depth: Since there are two main hyperparameters that define the random forest, max_depth and n_estimators, a grid search was used to find their optimal combination. To get an idea of the ranges of values to test, standard single-predictor cross-validation was used on each of the predictors separately. The first hyperparameter that we found was max_depth, which is the maximum depth that each tree can be. There are a large amount of predictors to consider since each possible technology is its own binary predictor, so it is likely that the maximum depth will be a large value to consider all of the predictors properly. Values from 1 to 55 were chosen as candidates for the maximum depth. A new random forest model was trained on the test dataset with 100 estimators for each of the candidate values. Then, it was tested on the test set with the explained variance and root mean squared error being the two evaluation metrics. The results of this cross-validation search are shown below. Figure 5 shows the explained variance and Figure 6 shows the root mean squared

error for the different max depths of each of the trees in the random forest. As can be seen, the results of both metrics indicate that a max depth of around 30 is sufficient in achieving high enough metrics that are very close to the optimal values. So, the grid search that we performed later considered depths from 30 to 40 with a step size of 1, to give some headroom.

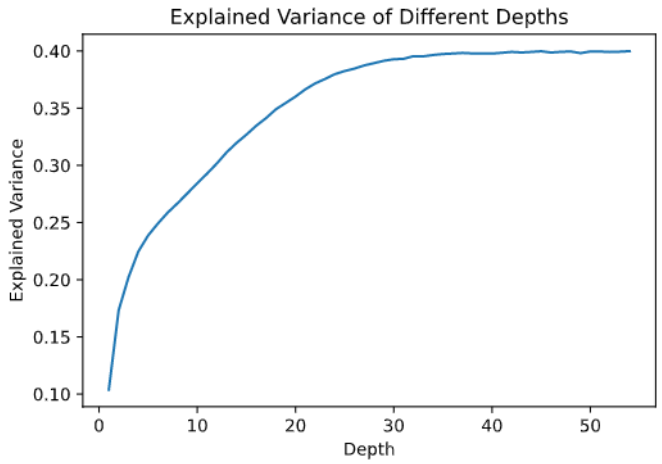

Fig. 5. Explained Variance for Different Max Depths of Trees

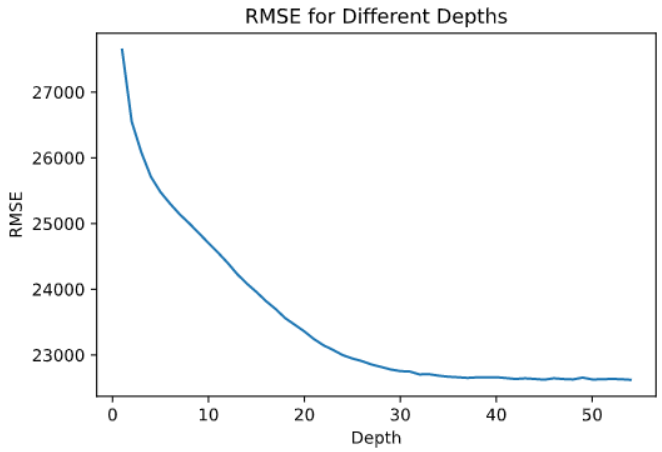

Fig. 6. RMSE for Different Max Depths of Trees

2) Number of Estimators: The other hyperparameter, n_estimators, was found in a similar way. The default value of the number of estimators is 100, which suggests that the optimal number will be around that value as well. So, the values of the number of estimators for consideration were chosen to be 1 to 251 with a step size of 10. This hyperparameter is in control of the regularization of the model, rather than in control of how well it can capture the complexity of the data. So, it is less important than the max_depth hyperparameter, which is why we used a larger step size and considered fewer candidate values. The results of this experiment are shown below. Figure 7 shows the explained variance and Figure 8 shows the root mean squared error for the different number of estimators in the random forest. As can be seen, the effectiveness of the model does not improve much after around the 40 estimators mark. So, in the grid search, the n_estimators hyperparameter will be searched from 40 to 50 with a step size of 1, to give some headroom.

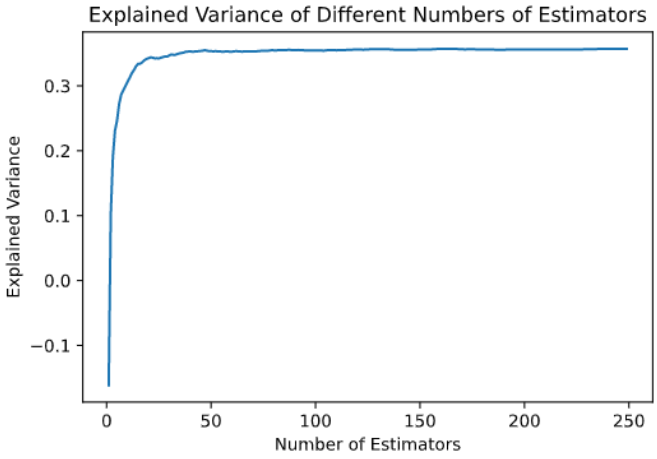

Fig. 7. Explained Variance for Different Number of Estimators

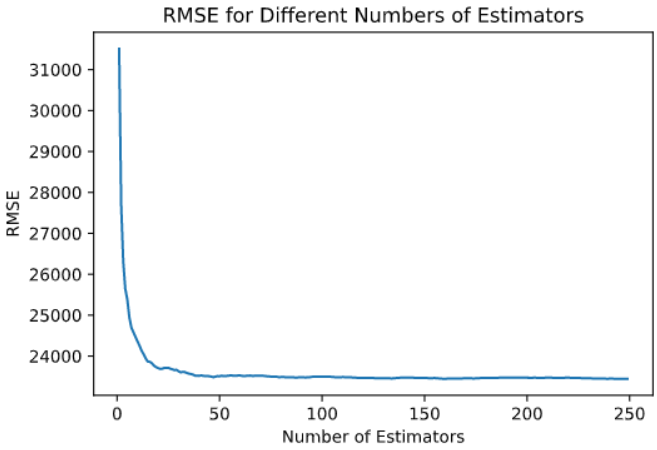

Fig. 8. RMSE for Different Max Depths Number of Estimators

3) Grid Search Results: After the ranges of the grid search were discovered in the previous section, we could finally execute the grid search to find the optimal values for the two hyperparameters. The grid search used a cross-validation fold of 5 for its training and testing. The evaluation metric used was the root mean squared error. The ranges of candidate values for the max depth was from 30 to 40 with a step size of 1. The ranges of candidate values for the number of estimators was from 40 to 50 with a step size of 1. So, a total number of 100 combinations of hyperparameters was tested and cross-validated. The result of the grid-search results is a random forest model with 45 estimators and a max depth of 39.

VI. Evaluation and Results

A. Evaluation

A random forest model was then trained with the optimal hyperparameters that the grid search returned. As can be seen in Table I, it has an explained variance of 36% and a RMSE of 23,345. Although these seem like relatively unimpressive values, this model is attempting to predict the salary of a job based on just its description, size, and the state that it is based in, which is quite a difficult task. Also, since the main goal of this project is to provide advice to improve the expected salary of a data science candidate, the precision of prediction is not very important. The important thing is that it can rank the next steps for the candidate to improve their salary. So the relative salary for different models is more important than the precision of the salary.

Explained Variance	0.363
RMSE	23,345

TABLE I
Evaluation Metrics of Model

B. Results

The data science job seeker will fill out a text file called applicant.txt with the state they are in, all the programming languages and libraries that they know, their degree level, and any certifications that they may have. Each qualification will be on a new line in this file and all of the qualifications must be included in the original text file with all the possible qualifications. The program then reads in all of these qualifications and prints out the expected salary of the applicant. Since the random forest model is relatively lightweight, it returns a prediction within a second. Next, the program will output useful information on how the applicant can improve their expected salary. When the author of this paper input tjeir own qualifications: Colorado, Python, TensorFlow, Linux, Agile, Networking, etc., it output that their expected salary should be $100,3535.

1) Different States: First, the program outputs information on which states would provide the best, and worst salaries for the applicant. It iterates through all of the states and trains a new model on that state with the same qualifications. It then provides the 10 best states that the applicant can move to and the 10 worst states that the applicant could move to, including the salaries in a bar plot. For me, the best states can be seen below in Figure 9 and the worst states can be seen in Figure 10.

2) Different Qualifications: Next, the program outputs information on which qualifications would provide the highest and lowest increase in salary for the applicant. It iterates through all of the qualifications that the applicant doesn't currently have and estimates their salary if they did have that qualification. Then, it returns a bar plot of the top 10 best and worst qualifications that the applicant could get. For the author's qualifications (minus a few to demonstrate the capabilities of the program), the output is below. Figure 11 shows the best qualifications that they could learn/earn while Figure 12 shows the worst qualifications that they could learn/earn. One of the most fascinating parts about this program is that there are qualifications that would actually lower the expected salary of the applicant. The explication for this is that the presence of some qualifications on a job description indicate that the job is for a lower-level entry position or for a roll that may have a low salary ceiling cap. The explination for the three worst qualifications are described, next. First, having a security clearance would mean that an expected salary is lower because government jobs tend to pay lower than industry jobs. Secondly, SQL usually means that the job is for a data analyst, not a data scientist, so it will likely pay lower if included. Third, it was originally unexpected to see that the inclusion of "finance" means a lower salary, but it may be because there are numerous paper pushing jobs that use low-level analysts at finance companies. As could be guessed, the inclusion of machine learning, languages such as C# and Java, and ML technologies such as PyTorch and Spark means

Fig. 9. Best States for Salary

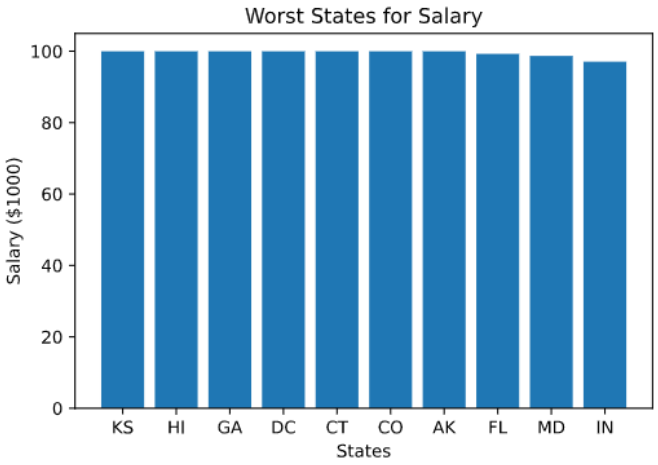

Fig. 10. Worst States for Salary

an increase in the salary because these skills are so sought after.

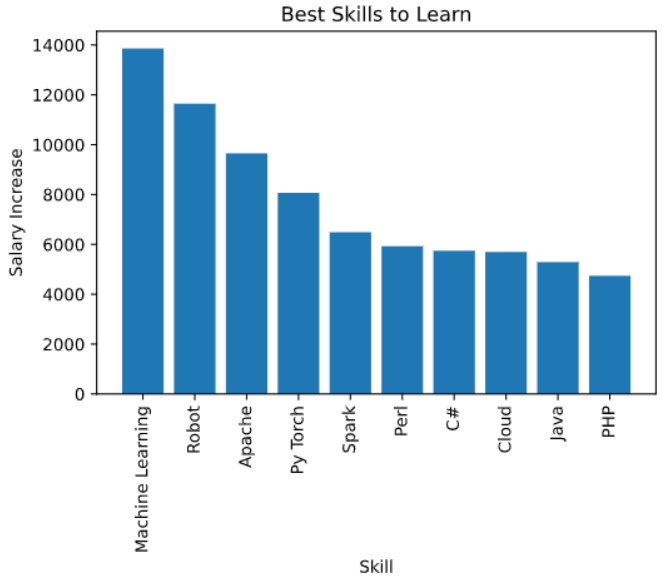

Fig. 11. Best Qualifications for Salary

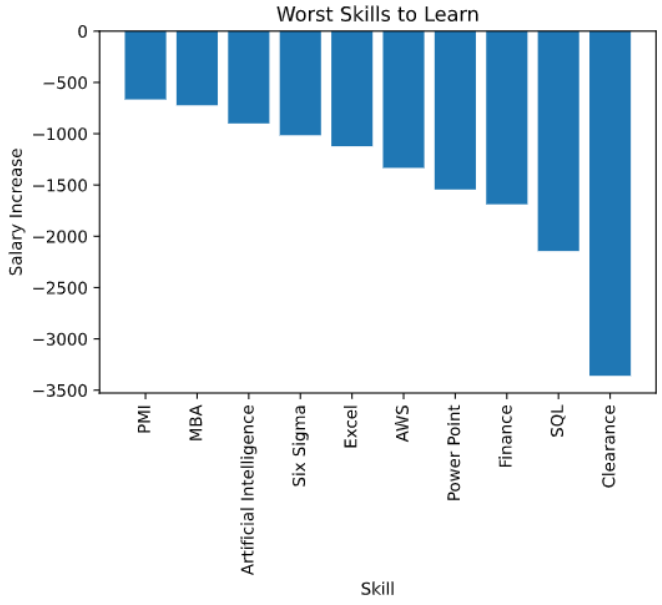

Fig. 12. Worst Qualifications for Salary

3) Discussion: The most surprising result of this project is how individualized that it is based on the applicants qualifications. At first, it may seem like the program may simply find that the qualifications have a very similar effect on all applicants. In other words, that for all applicants, qualification A would be the best to learn, then qualification B, then C, etc. This would mean that one could basically recreate this whole program by sorting a list of qualifications based on how effective they were at increasing the expected salary, then return the top 10 that the applicant doesn't currently have. However, this program gives very specialized and unique advice for every different combination of starting qualifications. It was a lot of fun to experiment with different applicants and see what the program would recommend for them to learn next.

VII. Ethics

A. Employers

Perhaps the biggest ethical concerns to consider when creating a model that can predict the salary of an individual is the ramifications if an employer uses this to decide on the salary they should offer a new applicant. With the current predictors that are being used now using now, Skills, Experience, State, this is not a concern because it does not discriminate anyone. However, if an employer also includes predictors such as gender, age, and race, this can quickly turn the model into one that is ageist, sexist, and racist. If the predictors of these models were heavily controlled, it could possibly be a suitable solution for employers. But regardless, it is important to be aware of this issue.

B. State Predictor

One predictor that this model uses that may have ethical concerns is the state predictor. Some states, such as California and New York, have a much higher salary for Data Scientists than others. For sake of argument, if this model and tool were to become very popular, it would almost certainly recommend people move to these states to increase their salary potential. This would drive high-salaried people to these states and worsen the problem of gentrification that they are already experiencing. The rapid growth of high paid people moving to California, particularly the Bay Area, has caused it to be one of the most expensive places to live in the world. This causes the existing citizens who may not have high paying jobs to lose significant amounts of money and pushes many to poverty or homelessness. If this tool were to be used on a mass-scale, it will be important to control for these problems in order to avoid worsening situations like this.

VIII. Future Work

A. Recommended vs. Required

The biggest improvement that can be made to this program is to use better parsing techniques on the description to pull out the information for the qualifications. As mentioned, currently the program simply checks the description for the presence of a qualification in the form of a string. If the decryption has that qualification, then it is considered required. This has the main issue of not distinguishing between recommended and required qualifications, which are very common on job postings. This would reduce the error of the predictor greatly, but there was not enough time to implement this change.

B. Years of Experience

Another thing that can be parsed for is the experience level of applicants. There are seemingly countless ways to represent the experience level desired for a job posting. Formats include ”5+ years”, ”5 or more years”, ”over 5 years”, etc. It would be difficult to create a parser to account for all of the possible formats that experience could take on. However, if this could be done, it would likely reduce the error of the predictor greatly.

C. Easier UI

Currently, to run this program, the user has to download the code as an interactive python notebook, input their qualifications in the text file, then run the program. It would be much easier if this were hosted as a web application or as an API. However, this would require both more time to set up, and a machine to host it on, so it was not able to be implemented. The functionality of the machine learning aspects were the most important parts for this project and those were demonstrated effectively without a better UI.

D. NLP Techniques

One method to solve the parsing problem is to use NLP techniques to parse for technologies automatically without having to self-define regex expressions and other tricks that were used in this project. This would be a whole other machine learning model to train, test, and tweak so it would take quite a bit of time to implement. However, it would have a great side-effect of abstracting away a lot of the complexities of this program and also has the potential to improve the effectiveness of the original machine learning model greatly.

E. Cost of Living

It is important to note that salary alone isn't a good indicator of how relatively wealthy somebody is. A $100,000 salary would make one rich living in a rural town in Kentucky, but the same salary goes much less far if one were living in San Francisco. The cost of living varies for every city and state and would be a very helpful feature to add to this project. It was not added to this project because the point of this project is to showcase the machine learning algorithms that the models use rather than create a robust application. This is a great starting point to a helpful and complete application like that in the future, though.

REFERENCES

[1] Brynjolfsson, Andrew McAfee and Erik. “Data Scientist: The Sexiest Job of the 21st Century.” Harvard Business Review, 26 May 2017, hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century.

[2] Gutstadt, Joseph. “data jobs.” Kaggle, 16 Aug. 2020, www.kaggle.com/josephgutstadt/data-jobs.

[3] “14 Most Used Data Science Tools for 2019 - Essential Data Science Ingredients.” DataFlair, 23 May 2019, data-flair.training/blogs/data-science-tools/.

[4] Dilmegani, Cem. “20 Top Data Science Tools: The Ultimate Guide [2021 Update].” AIMultiple, 7 Feb. 2021, research.aimultiple.com/data-science-tools/.

[5] Wangming Situ, Lei Zheng, Xiaozhou Yu, and Mahmoud Daneshmand, 2017, Predicting the Probability and Salary to Get Data Science Job in Top Companies: IIE Annual Conference. Proceedings, p. 933–939.

[6] Chen, L., Y. Sun, and P. Thakuriah, 2019, Modelling and Predicting Individual Salaries in United Kingdom with Graph Convolutional Network, in Hybrid Intelligent Systems: Cham, Springer International Publishing, 61-74 p., p. 61–74, doi:10.1007/978-3-030-14347-3_7.

[7] Catherine B. Johnson, Matt L. Riggs, and Ronald G. Downey, 1987, Fun with Numbers: Alternative Models for Predicting Salary Levels: Research in higher education, v. 27, no. 4, p. 349–362, doi:10.1007/BF00991663.

[8] “Sklearn.ensemble.RandomForestRegressor.” Scikit, 2020, scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.