**Section 1: Warm Up**

**1. Which companies applied for the largest number of H­1B visas where the job opening was located in NYC? Please describe any issues you may encounter summarizing the data by employer name.**

The full list of employers with the number of employees for whom they submitted applications is located in the counts\_and\_wages\_by\_employer.csv document. There were two difficulties in aggregating this data:

* Location data was inconsistent. There were various spellings of “New York”, alternate designations (“Bronx” or “Brooklyn”), as well as mismatches between city and state. I included all locations that had a the words “New York” in the city designation and “NY” in the state designation. I included both primary and secondary location fields in my search.
* Employer names were messy. Messiness included the inclusion of annotations in parentheses such as “(Guam)”, inconsistent punctuation (“US” vs. “U.S.”), extraneous white space, and inconsistent inclusion of acronyms such as “LLC”. I removed all of that messiness (counting any word of three letters or less as a acronym) before aggregating.

**2. Calculate the mean and standard deviation of wages proposed for workers located in New York City and Mountain View. Are the average wages in these two locations statistically different? What factors could explain the results?**

|  |  |  |
| --- | --- | --- |
|  | **NYC** | **Mountain View** |
| **Mean** | $88,801 | $120,091 |
| **Standard Deviation** | $34,680 | $29,512 |
| **Probability of earning more than in the other city** | 23% | 77% |

I calculated statistical difference by bootstrapping 1000 sample means from each of the two data sets, and then comparing those distributions of means over 1000 permutations. The mountain view means were greater than the NYC means 100% of the time. I didn’t compare standard deviations because, while the Mountain View data appears normal (mean of $120,213, median of $120,245), the NYC data does not appear to be (mean of $89,064, median of $79,627). Because of the questions about normality, I slightly adjusted the question to ask what the chances were of pulling in a higher wage in one city than in the other. To do this, I bootstrapped 10,000 samples from each city and calculated the percentage of instances where a random draw from one city beat the random draw from the other city. I repeated the simulations 1000 times. On average, Mountain View wages beat NYC wages 77% of the time.

The standard deviations suggest that NYC has a greater spread in wages than Mountain View does. A simple explanation for this, as well as the lower mean in NYC, is that the Mountain View data set contains a higher density of technical hires, which often command higher salaries. However, the differences could also be due to different hiring norms in the two locations, or even among specific employers within the two locations.

**3. For NYC, what is the relationship between the total number of H­1B visas requested by an employer and the average wages proposed? Visually represent this relationship if appropriate. Is the relationship statistically significant? What might explain this relationship?**

The relationship between H1B visas requested and average wages proposed can be viewed by opening the applcations\_vs\_wages\_graph.html file in the browser. There does not appear to be any relationship between the two variables in terms of a central tendency. However, there is a much greater spread in average wages for employers who submit fewer applications. This could because the spreads among the higher-applicant employers is being hidden behind the average, or because employers who go through this process more often have standardized their overall hiring process, including wages offered, to a greater extent.

**Section 2: Brainstorming**

The following are some of the ways the data set could be parsed or visualized to help people better understand the landscape of H1B visa applications:

* **Give an in-depth description of locations that are bringing in more foreign workers.** Something like a filterable heatmap could be appropriate here: geolocate employment locations, plot a point for each applicant at each location, then bin the points when the points are too dense to be seen individually. Coordinates could be easily geolocated from city and state information. For the product to be really useful, the points on the heatmap would need to be filterable (so only visas for particular job titles, or particular wage ranges, etc. were shown). Additional datasets could be brought in if we wanted to filter on more than just application data.
* **Describe the variation in expected wages for different job types and locations.** Take averages within job type and location and then calculate the distribution of differences between actual applications and those central tendencies. This would probably be a good variation on the heatmap described above (showing wages instead of number of workers).
* **Explore the differences in employer location and expected work location of applicants.** Employer zip code already exists and expected work location can be geolocated easily. This could end up being yet another layer on the heatmap described above (showing distance rather than workers or wages).

The following are some specific questions about H1B visa applications, answers to which could help employers make better hiring decisions:

* **What characteristics of an application predict the success or failure of that application?** An employer invests time and money in trying to get an applicant a visa. Rejected applications equal wasted resources (and frustration for both employer and employee). It is possible there are hidden “red flags” within an application that make it more likely to rejected. The challenges in this project would be cleaning the data set to the point that it could be trusted in a model. Location and employer data are very messy. Several variables that might be important (for example, amount of time between application and expected start date) are implicit would need to be explicitly created. Because this would be an exploratory analysis, it would be best to select an algorithm that requires a minimum of assumptions and can handle large numbers of variables. The model-fitting stage of the project would need to include some checks to guard against overfitting.
* **What characteristics of an application predict the length of time it takes to get a decision on the application?** This project is similar to the previous one. Hiring an applicant means making plans about the company and/or teams on which that applicant is expected to work. Companie usually hire after they’ve identified a need, so waiting for an application to got through necessarily involves stop-gap measures to hold the team over until the new hire can fill in their capacity gaps. It helps to know how long you’ll need to continue those stopgaps. I wouldn’t recommend doing this project before first doing the one listed above. It’s very possible that there are different timeframes involved in acceptance or rejection of applications. We’d want to understand those dynamics first.

**Section 3: Exploration**

I took a first shot at predicting application certification for NYC applications.

**Dataset:**

I kept only records for full-time positions with “New York” in the workplace city name and “NY” as the workplace state name, and with a status of “certified” or “denied” applications. I also removed the top and bottom 1% of wages because of some unrealistic values (for example, a yearly salary of hundreds of millions of dollars), and converted proposed wage and prevailing wage to from hourly, weekly, bi-weekly, and monthly amounts to yearly equivalents. Finally, I cleaned employer name to remove acronyms, punctuation, and spaces, and consolidated SOC names and employer names that occurred 10 or fewer times into “other” categories.

**Predictors:**

* total\_workers: number of workers an application was submitted for
* prevailing\_wage: prevailing wage for the work location
* average\_wage\_rate: average and lower and upper (if provided) proposed wages
* has\_upper\_wage\_value: flag showing whether upper range of posed wages was provided
* diff\_from\_prevailing\_wage: difference between average\_wage\_rate and prevailing\_wage
* days\_from\_submit\_to\_start: days from application submission to proposed start date
* days\_from\_start\_to\_end: days from proposed start date to proposed end date
* Dummy variables for each employer state, employer name, and SOC name

**Procedure:**

I fit a 100-tree random forest through 5-fold cross-validation. I selected a random forest because the algorithm has a reputation for high accuracy, performs well with a lot of variables and relatively few training cases, has an internal method for avoiding overfitting, and can accommodate interactions among variables without needing those interactions to be explicitly specified. Random forests can underperform when categories are disproportionately represented in the training data: I corrected for this with balanced sampling. The procedure can also be very computationally expensive with large data sets (not relevant on this data set, but on larger data the implementation could be parallelized).

**Results:**

Overall, the model was quite accurate - over 90% accurate depending on how we decide to partition the probability estimates. However, given that the grand majority of applications are approved, overall accuracy means little. The following table shows how the false positive and false negative rates change as we partition each application’s probability estimate:

|  |  |  |
| --- | --- | --- |
| **cut** | **false\_positives** | **false\_negatives** |
| 0.1 | 0.03 | 0 |
| 0.2 | 0.03 | 0 |
| 0.3 | 0.03 | 0 |
| 0.4 | 0.03 | 0.01 |
| 0.5 | 0.03 | 0.02 |
| 0.6 | 0.03 | 0.03 |
| 0.7 | 0.02 | 0.04 |
| 0.8 | 0.02 | 0.05 |
| 0.9 | 0.02 | 0.1 |

Our choice of partition would depend upon the needs of a customer. A false positive (submitting the application only find that it is rejected) is a waste of resources. A false negative (not submitting was would be a successful application in the belief that is won’t be approved anyway) is a missed opportunity. One may be more damaging than the other based on the customer’s needs. If false positives were more damaging that false negatives, a cutoff between 0.6 and 0.7 would be more appropriate. If the opposite was the case, a cutoff between 0.3 and 0.4 would be better.

Very few predictors has high importance scores. The top five predictors, representing all variables that had an importance score of 0.04, were “diff\_from\_prevailing\_wage”, “days\_from\_submit\_to\_start”, “average\_wage\_rate”, “prevailing\_wage”, and “days\_from\_start\_to\_end”.

**Next steps:**

All of this was just a surface-level exploration of the issue of predicting application outcomes. Next steps would depend on the customer. For example, the most important predictor was the difference between the proposed wage and the prevailing wage. If the proposed wage was negotiable for the customer, then we would want to more fully explore the relationship between that variable the predicted probabilities. However, if that issues was non-negotiable (if, for example, a reduced proposed wage would make the job opportunity no longer attractive to the desired applicants), then there would be no urgent need to get a granular understanding of that particular variable’s relationship.