Predicting Salaries from Candidate and Role Features

Alejandro Terrazas, PhD

Overview

The purpose of this exercise is to predict salaries based on seven features (job type, years experience, degree status, major, distance from major metropolitan area, and industry. Except for years experience and distance from major metropolitan area, the features were categorical. The Company Id feature was not used because, it had a small loading on salary and because it did not appear to make sense as a unique company identifier. For example, many company ids were associated with multiple CEOs and a range of distances to metropolitan area.

Exploratory Data Analysis

Data science projects require exploratory data analysis to identify missing data and to see if the data overall 'makes sense'. This project used Python, Pandas, and Seaborn to explore relationships between variables and to look for missing or 'suspicious' data that might need further examined and potentially eliminated. There were no missing data.

The data were surprisingly balanced (most likely, by design) with *nearly* equal counts across all variables. Usually, great effort is required to ensure training and test data are balanced and that under represented categories are over sampled. In addition, there were 1,000,000 records provided. These were split into Train (600,000 records), Validation (200,000 records), and Test (200,000 records) datasets. Only a subset of the 600,000 records were used for training. The entire 200,000 records from the test dataset were used with a script RunTest.py.

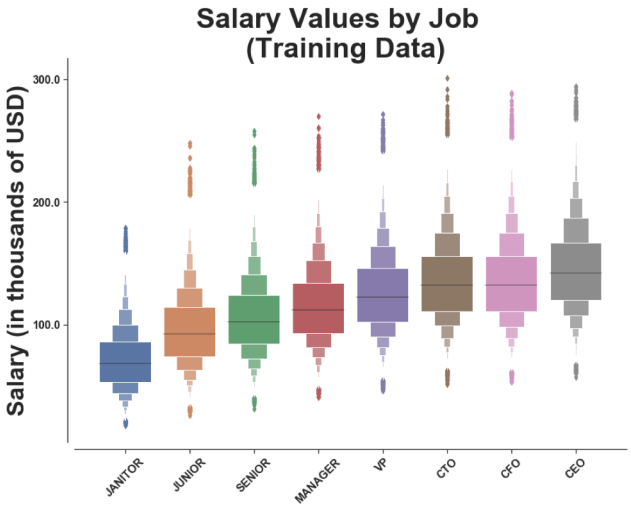


Figure 1: Presence of outliers by job. *The graph shows the training set data by job type. Each box represents the density of observations in quantiles with the black horizontal line in the central box representing the median value of the data. The dots at the top and bottom of each boxen plot represent extreme values that are potential outliers. The upper values are indeed extreme and add a small bimodality in the upper range. Some of these extreme values were suspect. For example, there are a cluster of janitors making above $150,000 per year, which seems unlikely (although some special cases, e.g., hazardous materials specialty, may exist.) Using the current Indeed.com Salary Estimator as a reference, revealed janitors tend to make $20-40 per hour (or $40-80k annually.) Also surprising, some CEOs had salaries under the median value of janitors. Again, these values may be special cases in which a CEO is paid with equity not accounted for in the salary value. An effort was made to transform the data; however, the transformation did not meaningfully improve the results.*

The first pattern that became obvious is the presence of a distinct group of high-value outliers for every job type. There were a small number of low-value outliers. The outliers were apparent in the plots of other variables, as well. **Figure 1** below shows these outliers in a series of Boxen plots by Job Type. Selecting these outliers and examining them directly indicated that they tended to have maximum (or near maximum) values across all other variables (e.g, years experience, distance, etc.)

The outliers resulted in a skew to an otherwise Gaussian-looking distribution. Different methods were attempted in an effort transform the target distribution to Gaussian; however, these transformation did not affect the model results significantly. **Figure 2** shows the distribution of salaries before and after quantile transformation.

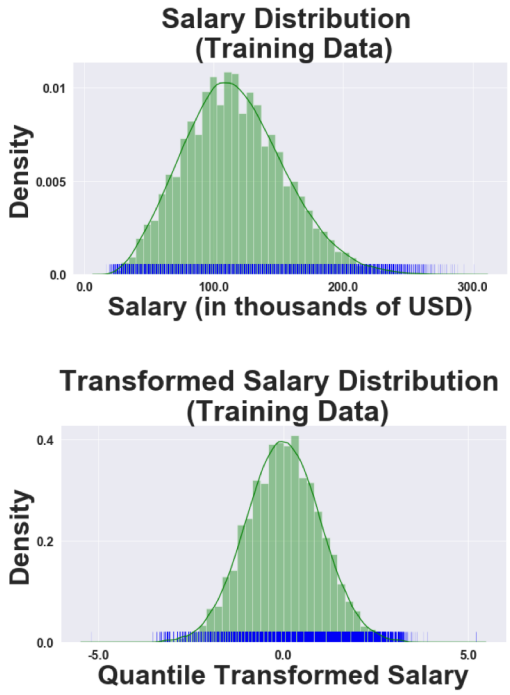


***Figure 2: Skewed distribution of salaries. Top panel**: Distrubtion and 'rug' plot of grouped salary for training data. Similar results were seen with the validation and test data sets. The blue 'rug' at the bottom shows the skew in the distribution extending to high upper values. Each small line represents a single observation in the data. **Lower panel**: Distribution of salary values after quantile transformation. After the transformation, the distribution more closely resembles a gaussian distribution. Moreover, the rug plot is more centered in the distrubtion. Because the skew is present for each job type, the quantile transformation does not completely eliminate the skew at each job level. Regardless, the number of potential outliers was fairly small compared to the overall number of observations. Furthermore, the predictions after transformaton of the target variable were not substantially different than those obtained with the the non-transformed data.. Transformation within job categories may be a way to improve the model going forward. Another approach that may be useful is Gaussian mixture modeling.*

Above and beyond the clusters of extreme salaries, some other unusual relationships were identified. For example, nearly 40% of CEOs has degree status of None. Internet searches indicated a number closer to 10% in public surveys. Moreover, many individuals with advanced degrees (Doctoral or Master's degrees) had None listed for major. While it is possible to 'clean up' these data, there was no compelling reason to do so. Other than the suspicious degree and major data, no missing values were identified.

Modeling Approaches

Ordinalized Variables

The categorical variables fell into a natural ranking that was re-expressed in ordinal values. For example, degree values of None, High School, Bachelors, Masters and Doctorate

became 1, 2, 3, 4, 5. Likewise jobType (Janitor, Junior, Senior, Manager, Vice President, CTO, CFO, and CEO) became 1, 2, 3, 4, 5, 6, 7, and 8 respectively.

The continuous variables (years of experience and distance to major metropolitan area) were also discretized to ordinal values. For example, years of experience (range 0-24 years) was converted into ten levels. Likewise, distance to major metropolitan area (range 0-99 miles) was converted into 10 levels. In addition, distance was reversed to 'nearness' (i.e., a distance of 0 corresponding to 10 in 'nearness'). The negative relationship between distance to major metropolis could also have been accounted for with a negative term in the prediction equation to yield the same result.

Different combination of the ordinal values were tested. For example, combining CTO, CFO, and CEO into a single category (EXECUTIVE) did not affect the error values. Degree status was binarized to DEGREE and NO-DEGREE, again with little effect on the prediction accuracy.

Choice of Linear Modeling Approach

Exploratory data analysis pointed to basic linear modeling as the best initial approach. Each feature was correlated with the salary target variable (r=0.58 for job type to r=0.26 for Industry). Only two features were substantially correlated with each other (Major and Degree, r=0.72), which makes sense. Major turned out to be the least important feature.

The intercepts were clearly different by several variables, most notably jobType. Having different intercepts for job type make logical sense, as This relationship can be seen in **Figure 1**, above.

Category Embeddings

Some experiments were conducted to use categorical embeddings but the complexity of the categorical variables were pretty simple. One idea that was tested and not ultimately used was to create embedding for Industry-by-Degree. This resulted in an 'appropriateness' score of the degree for each industry. So, for example, Math and Finance has a strong loading while Literature and Oil had a smaller loading.

Models Used

A total of 7 different models were tried with the data. All provided similar results. The methods tried were a) Bayes Maximum a Posteriori (MAP; written in Pystan), b) Decision Tree Regression, c) Random Forest Regression, and d) K Nearest Neighbors Regression (all in scikit-learn), f) a multilevel neural network written in Keras and TensorFlow, and g) simple averaging in Pandas, where the features in the test example are used to extract all matching examples from the training data then the salary averaged.

Hyperparameters for all method were adjusted until the validation results were optimized. Most of these adjustments were made based on the investigator's experience with the methods. In most cases, the default configurations were sufficient for decent performance on the problem set.

The original data were shuffled and then split into separate train, validation, and test data (60%, 20%, 20% respectively. The test set was only used at the very end. The validation data were used for adjusting hyperparameters until good performance was achieved.

Feature Importance

A simple way to tell which features are most important was the output of scikit-learn's RegressionTree feature_importance_ variable. In order of importance the features are: a) jobType (score = 0.41), b) years of experience (score=0.20), c) industry (score=0.14), d) degree (score=0.13), and e) major (score = 0.09). CompanyId was not included in the analysis but did not show a strong relationship with salary.

Bayes Model

The Bayes regression model utilized separate fits of the intercepts for the different eight different job types and for the degree category. Otherwise, slopes were estimated for the other ordinal variables. Once the posterior distributions for each of these parameters is estimated, random samples from each distribution are created and a prediction is made for each set of features. The maximum of the posterior for each prediction can be use

Scikit-learn Models

The scikit-learn models are fairly easy to set up, train and make predictions. Three scikit-learn models (DecisionTreeRegression, RandomForestRegression and KNearestNeighborsRegression models were trained using 80000 records from the training set.

Neural Network Models

The neural network model consisted of 5 layers (1 input, 3 hidden with 120, 80, 20 neurons, 1 output layer with a single neuron and linear activation function). Ten training epochs with a batch size of 128 were used. The cost function stabilized after a few epochs. For the neural network model, scikit-learn RobustScaler was used to transform the features and the target variable (salary) prior to network training and testing. Batch normalization was used at each layer.

Running the Models

To models were developed and trained using Jupyter Notebooks and Python 3.74. The required libraries are: 1) Keras and TensorFlow, 2) Numpy, 3) Seaborn, 4) scikit-learn, 5) Pandas, 6) Pickle for saving models and 7) h5.py, which is used to export/import the Keras model.

The main Juypter Notebook is Indeed-modeling.ipynb. EDA.ipynb shows some basic statistics on the datasets. The data were split using SpltiTrainCSV that produced 3 .csv files (TRAIN, VALIDATION, and TEST). These files are included in the .zip and therefore, do not need to be regenerated.

Finally, a script (RunTest.py) was made using the neural network model. The script imports the neural net model (trained in Indeed-modeling.ipynb), and runs the forward pass on the TEST dataset. The results of the forward pass are stored in test_salaries.csv.

Error Metrics

The error metrics chosen were a) adjusted r-squared, b) mean absolute error, c) mean % error root mean square error, d) percent error ([y_truth-y_hat]/y_truth), e) scatter index and f) root mean square error. In addition, the correlation between prediction and ground truth was used. The scatter index divides the RMSE value by the average of the values and thus, overcomes the tendency for RMSE to increases at high levels of the variable. **Table 1** provides a summary of error metrics for all methods. Again, the correspondence between methods is quite close.

Method	r-square	Mean abs error	Mean % error	Scatter Index	Correlation y_truth, y_hat	RMSE
Bayes	0.68	17.63	1.99	0.19	0.83	21.92
KNN	0.68	17.38	2.07	0.19	0.83	21.90
DecisionTree	0.63	18.27	2.66	0.20	0.81	23.30
RandomForest	0.68	17.66	2.71	0.19	0.82	22.34
Simple	0.73	16.40	2.73	0.18	0.85	20.43
Neural Net	0.72	16.43	1.69	0.17	0.85	20.28
Ensemble Ave	0.72	16.48	2.45	0.18	0.85	20.49

Table 1. Errors metrics for methods used in the exercise.

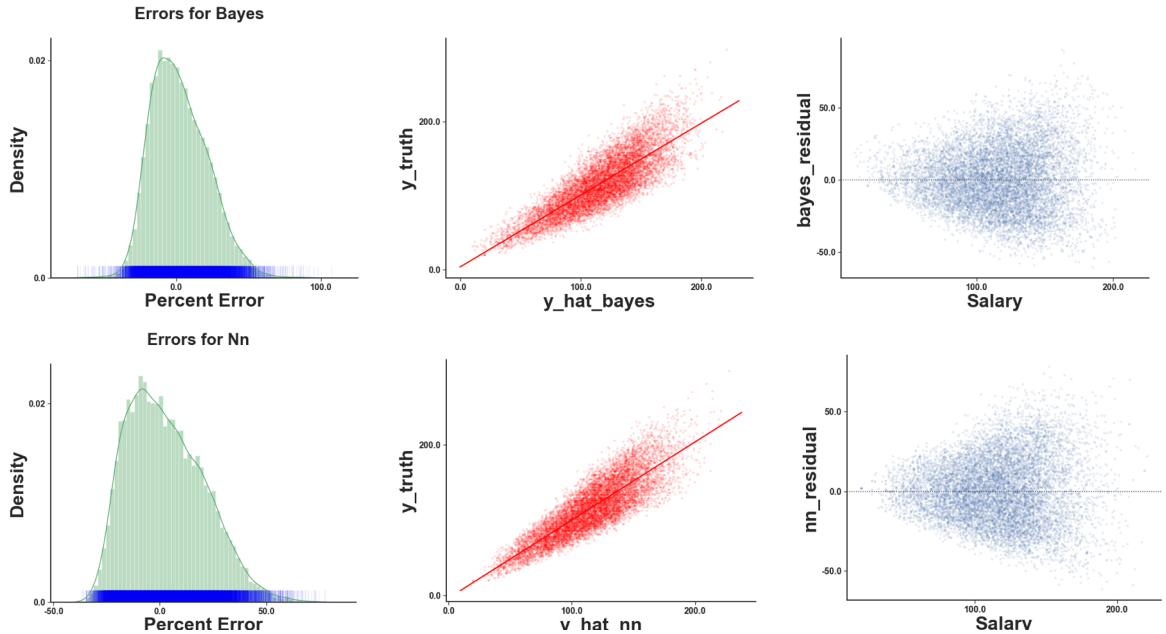
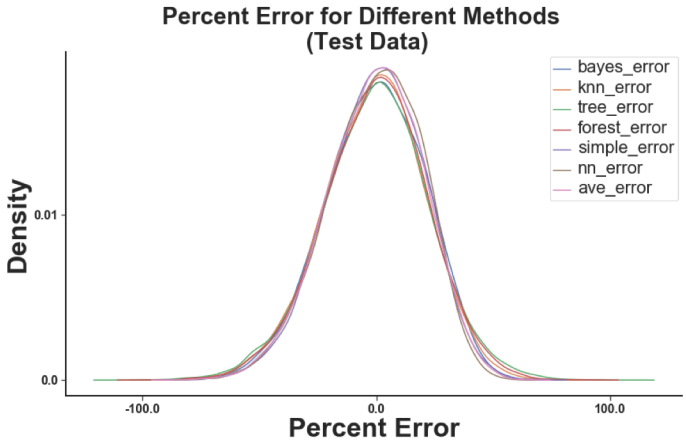


Figure 3. Error distributions, correlation and residual plots for two methods (Bayes and neural nets). *Left) distribution of percent errors with 'rug' in blue. Most of the errors are within the range [-50:50%]. Middle panel) scatter plot showing relationship between salary (y_truth) and predicted salary (y_hat). Both methods show a straight line plot indicating a close correspondence between the predicted and actual salaries. The correlation coefficients for the two methods are 0.83 and 0.85 for the Bayes model and the neural network model, respectively, demonstrating a good match (perfect would be 1.0). Right) Residual error plots. The residual errors become larger at larger salaries however, there is no discernable slope in the relationship.*

Figure 3, above, shows error distributions, correlation and residual error plots for the neural network model and the Bayesian model. **Figure 4**, below, shows the plots for all methods were very similar, confirming the old adage that data are more important than methods.

Figure 4. Percent error distribution for seven different models. *The methods show a close correspondence .*
Data are from 20000 runs of the test dataset. The vast majority of the errors are within -50%:+50% with a fairly sharp peak centered near 0. Any of the methods would provide the same basic results. The methods differed slightly in the preponderance of outliers.



Bayesian Credible Intervals

One nice feature of Bayesian modeling is the ability to produce posterior distributions and credible intervals. Credible intervals are equivalent to what many practitioners confuse with confidence intervals. For example, a 90% credible interval indicates that 90% of the possible values for the parameter of interest lie within the interval.

Figure 5 shows individual predictions with the posterior distribution and 90% Bayesian credible intervals. These intervals can be used for outlier detection.

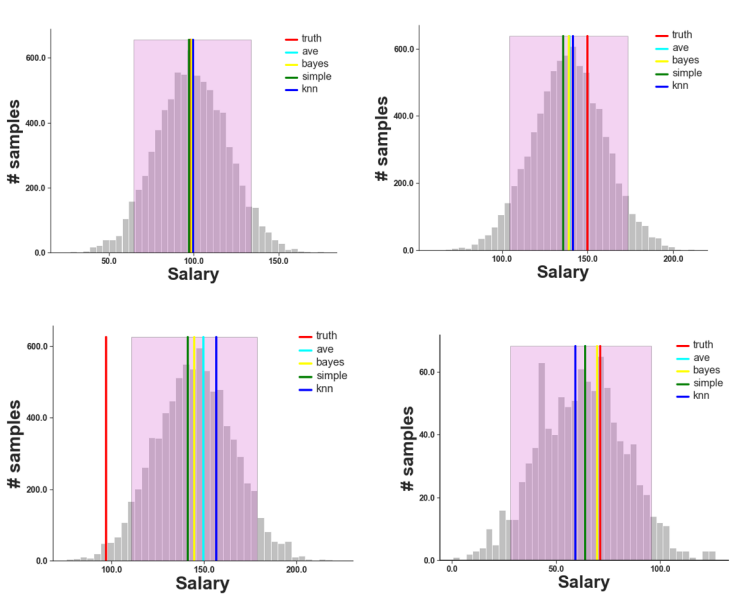


Figure 5. Results plotted over Bayesian posterior and 95% credible region. *Each of the three plots is a prediction for an individual example of the test dataset. These outputs could be simplified and provided in a web page. The red vertical line represents the ground truth value from the data. The area in pink is the Bayesian 90% credible region (5%-95%). Each method has separate color. The predictions are quite close to each other; however, the ground truth values sometime deviate beyond the credible region. Indeed, approximately 9% of the ground truth salaries were outside the region. Examples outside the credible interval can be considered over- or under-paid. Or, alternatively, could be considered 'superstars' or 'underachievers'. The lower left panel shows an underpaid individual, as the red ground truth line is below the credible interval.*

Computing Times

Each method was evaluated on a MacBook Air. Timings were made used the timer() function. Total time for 20000 predictions was used.

Overall, the Keras and Scikit learn models were substantially faster than the Bayes method and the Simple method. The Bayes and simple methods required 31.37 and 319.35 seconds for 20,000 predictions respectively.. The Keras forward pass took 0.50 seconds (over 60 faster than Bayes). The three Scikit-learn methods combined use 0.95 second (approximately 0.32 seconds per method).

Summary and Answer to Challenge Questions

1. **How long did it take you to solve the problem?** I worked on the problem over 60 hours. I kept trying different methods and kept getting the same answer from each. I wanted to significantly beat RMSE of 17 but it did not matter what method I tried. I spent quite a bit of time adjusting hyperparameters. The greatest amount of time was spent with the Bayesian model. I also spent time on graphics output. All of this was enjoyable to me and I view it as an opportunity to improve my skills.

2. **What software language and libraries did you use to solve the problem? Why did you choose these languages/libraries?** I used Python, Pandas, Seaborn, Scikit-learn, Keras, and Pystan for the variety of approaches I tried. I used these because they are convenient and I have experience with them.

3. **What steps did you take to prepare the data for the project? Was any cleaning necessary?** I was surprised to find that no real cleaning was necessary. I did not find missing values. Other irregularities are described in the paper above.

4. **a) What machine learning method did you apply? b) Why did you choose this method? c) What other methods did you consider?** I first tried Bayesian inference. After getting an RMSE around 17, I moved on to KNN Regression, Decision Tree Regression, and Random Forest Regression. I finally tried neural nets. All provided the same answers.

5. **Describe how the machine learning algorithm that you chose works.** The Bayesian approach allows one to model intercepts and slopes as model parameters as distributions of values. These distributions are estimated using Markov Chain Monte Carlo methods to estimate the complex multivariate probability spaces.

6. **Was any encoding or transformation of features necessary? If so, what encoding/transformation did you use?** For the neural net approach, transformation was necessary; however, the other methods did not require transformation.

7. **Which features had the greatest impact on salary? How did you identify these to be most significant? Which features had the least impact on salary? How did you identify these?** As noted in the above paper, In order of importance the features are: a) jobType (score = 0.41), b) years of experience (score=0.20), c) industry (score=0.14), d) degree (score=0.13), and e) major (score = 0.09). I used the feature importance method of decision tree and random forest to obtain the feature importance. These were clear from the correlation matrices, as well.

8. **How did you train your model? During training, what issues concerned you?** For the scikit-learn models, I used the .fit method. I used epochs and batches for the neural nets and statistical learning of the distributions for the Bayes model. I was concerned with over training on all of the models, except the Bayes model. Over training or under training are, of course, always a concern. The cost functions stabilized quickly. I was concerned about the possibility of outliers.

9. **a) Please estimate the RMSE that your model will achieve on the test dataset. b) How did you create this estimate?** I would estimate RMSE of around 20 for the model. I ran the model on 20000 records of the test dataset to obtain the numbers.

10. **What metrics, other than RMSE, would be useful for assessing the accuracy of salary estimates? Why?** Mean absolute error, scatter index and percent error are all good alternatives to RMSE. Number of estimates outside the credible intervals is another. All of these error metrics capture a different aspect of the error. RMSE is inflated because it increases with increasing values. Percent error and scatter index indicate better how a method pertains at all levels of the target variable.