Prediction of Salary in UK

Luoxiao Li
Computer Science and Engineering
University of California, San Diego
La Jolla, California

Xutong Liu
Computer Science and Engineering
University of California, San Diego
La Jolla, California

Yijun Zhou
Computer Science and Engineering
University of California, San Diego
La Jolla, California

ABSTRACT

Salary is one of the most important things to consider once you receive a job offer. However not all job advertisements display how much they are willing to pay their employees. If we can accurately predict the salary of the job posting based off of the advertisement alone, then people who are considering taking the position on the advertisement can get a rough idea of what they can expect their salary to be and if they have room to negotiate.

1. Dataset Characteristics

The data set that we used is a collection of job advertisements in the United Kingdom, provided by Adzuna. The collection contains 244,768 data points, which were formatted to provide the following information:

A List of categories for the job
The company name
The contract type (permanent or non-permanent)
The contract time (Part time/Full Time)
Description of the Job
Job ID: A unique identifier for each job ad
SalaryNormalized: The actual salary of the text
Location of the job
Title of the job

A. Basic stats:

Our dataset had a total of 244,768 job advertisements. Not every advertisement provides each of the listed information, because the data that we use are gathered from real data that are used in job ads and are subjected to real world noise, such as having ads that are not UK based, or incorrectly inputted salaries.

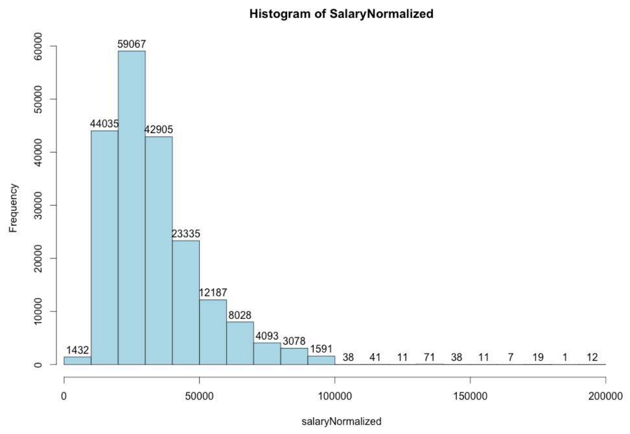
B. Data property details:

Below we have some interesting statistics of our data

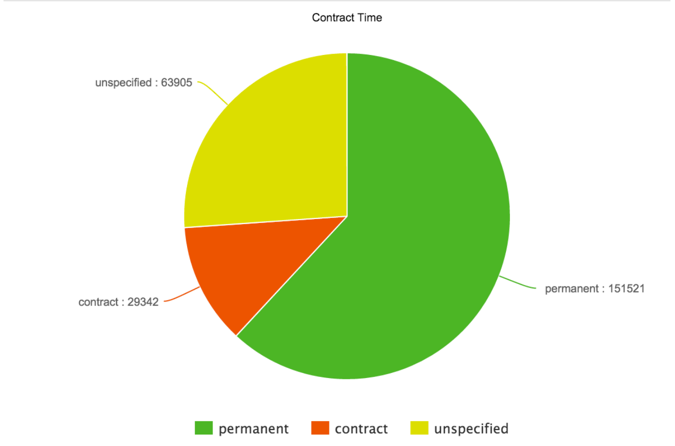
- The average salary in the UK is 34,122.58 pounds
- The highest paying position has the title 'Quantitative Researcher Required by Leading High Frequency Trading Prop Company London' at "NJF Search International', and pays 200,000 pounds
- The minimum paying position is an 'Accounts admin' at 'MPA Recruitment', and pays 5,000 pounds
- There are a total of 19,203 companies that have job advertisements in this dataset
- The jobs in our dataset are categorized into 29 types of work
- 2,580 locations have posted job advertisements

C. Basic analysis:

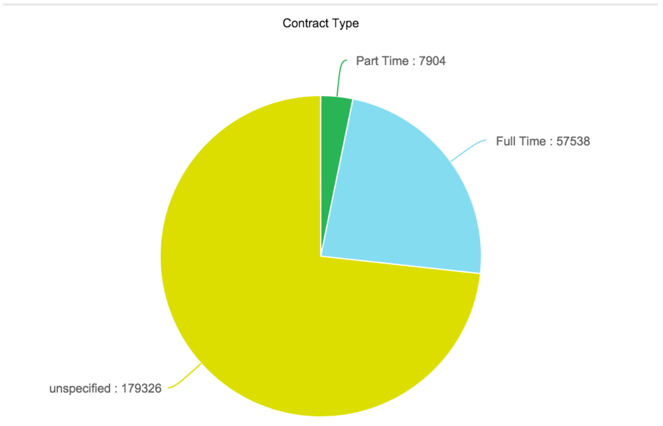
1) Distribution of Salary Normalized



We can see here that if we just use the salaries given to us in the data set, the salary distribution is very skewed to the left, meaning that there are a lot more low salaries on the low side than there are on the high side. The distribution of salaries like this lead to a standard deviation of 17640.5070886



In this diagram, we can also see that the data set primarily covers permanent contract times with 61.9% of the data being advertisements for permanent positions and only 12% for contract positions. However, we can also see that 26.1% of the advertisements do not specify whether or not they are looking for a part time or full time employee, which means that there's a lot of noise in our data.



Initially we thought that the contract would make for a really good feature, however, the problem is that most of the data (73.26%) does not specify whether or not it is for full time or part time, making this property of the dataset mostly noise, which is unfortunate, but realistic since not all real world job advertisements specify full time/part time. However for the ones that do, 23% are for Full Time and 3.2% are for Part Time.

Location	Number of Job advertisements
UK	38217
London	28487
South East London	11232
The City	6196
Manchester	3285

The table above lists the locations with the most job advertisements. One thing that we noticed was that the location with the most job advertisements was the UK. However, the problem with this is that most of these locations are cities in the UK. Therefore, the data is inconsistent, in that not all these locations are cities.

2. Predictive task

A. Task: Salary Prediction

In this project, we want to predict the salary of the person who accepts the job posted by the job advertisement. Using the dataset provided by Adzuna, we want to create a model that minimizes the mean absolute error when predicting the salary (One that predicts the salary as accurately as possible). To do this we chose to use a random forest regressor, which we used to fit our feature model of the data.

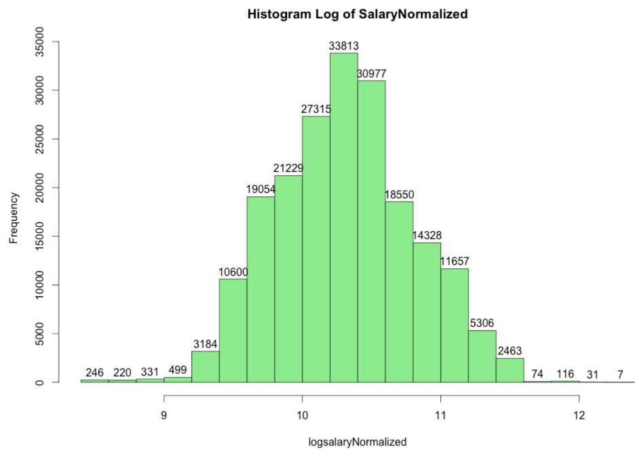
B. Validity of predictions

Since we are given 244,768 total job advertisements, we first randomize the order of the data, and then select the first 230,000 data points for training, and then the rest of the data was kept for testing. After training the model, we wanted to estimate the salaries and calculate the MAE on that data set, which is basically on average, how many pounds does our model predict off the actual salary. We also decided to calculate a normalized MAE, which calculated the percentage that our prediction was off by, because 1,000 pounds means much more to the person making 5,000 than the person making 200,000.

C. Data Preprocessing

1) Distribution of Salary Normalized

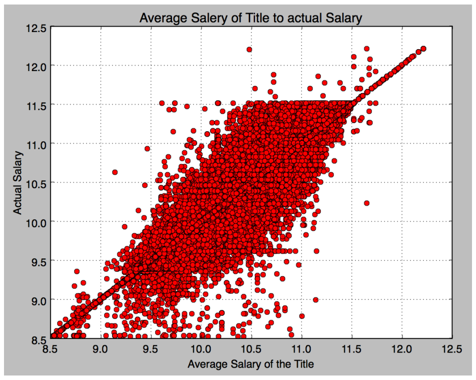
In the exploratory analysis section, we saw that the data was skewed to the left. This is a problem because linear regression models assume that distributions are normalized. Possible solutions to this problem are taking the square root or the log of the salaries. This leads to a more evenly distributed set of salaries. If we take the log of the salaries, we can see that the distribution closely resembles a bell curve, which is what we want.



D. Feature exploration

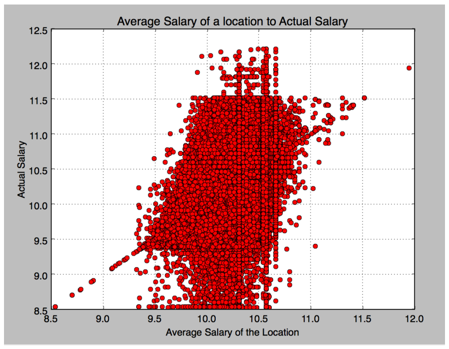
*For each of these features, when dealing with Salaries, we always use the log of the salaries and not the actual salary

a. *Average Salary of Salary of people with the same Title (Correlation 0.91135613)*



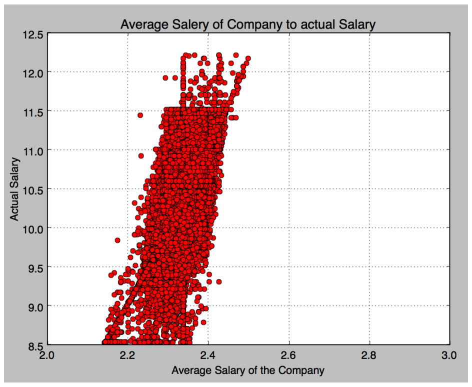
We gathered the average salary of people with the same title, and for each advertisement, we compared this average to the actual salary of the position. This led to a very strong correlation to the actual salary.

b. *Average salary of a location (0.41511654)*



We can see that although we can see a correlation, there is also a lot of noise due to the UK not being a city problem. This could explain why the correlation of the location's average is lower than the company average.

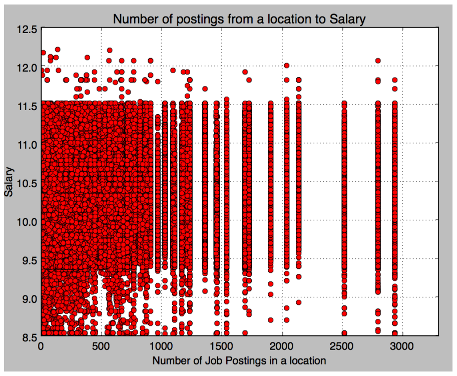
c. Average Salary of Companies (correlation 0.6679146)



We also computed the average salary of the whole company and found that while there was a correlation to the actual salary, it wasn't as strong as the correlation to the title.

d. Num of job postings in a location (correlation 0.1048831)

*In the graph below, we only took the locations with 0-3,000 job postings due to a few outliers in the dataset.



We can see that the number of postings in a location to Salary has little to no correlation to salary. This was surprising because we initially thought that more job postings meant that the company was more desperate and would be willing to pay more money. This could also be due to the fact that the location data is inconsistent. Even though it is not shown in the graph due to it being an outlier, the location with the most job postings is the UK, differs from the rest of the data set because it is a country and not a city.

e. Part Time/Full Time (correlation 0.12048831)

Because our dataset has a lot of noise when pertaining to this property, the correlation is also very low.

f. Text-mining: Unigram

Using 500 most common unigrams, and see which 5 unigrams with the most positive associate weight, and which 5 with the most negative associate weight. "note, may, live, ltd, over" are 5 unigrams with the most positive associate, and "require, recruiting, seeking, duties, originally" are 5 unigrams with the most negative weight.

3. Select/Design Models

A crucial step in this project is to select useful features and accurate models. From the correlation between the features and data, we were able to design several models.

A. Evaluation of the Models

To evaluate the accuracy of the model, we calculated the MAE (Mean Absolute Error) of the prediction in the validation set. We calculated both MAE Score (prediction - actual) and MAE Normalized ((prediction - actual) / actual) to evaluate the accuracy of the models.

B. Baseline Models

Based on the relevant features, we tried to start with the basic models using only one feature on linear regression.

a. **Model 1:** *Prediction = average Salary*

We started with the simplest model of prediction. Without using any techniques and features, we calculated the average salary of the training data. We predicted all the validation data to be the average salary of the training data. The MAE of this baseline model is: MAE Normalized = 0.781204005349 & MAE Score = 21838.716943. We can see that the accuracy of the baseline is extremely low.

b. **Model 2***: Feature Company*

We then used single feature company and linear regression to predict the salary. We categorized the salary into each company, then took the average of the salary in each company as the feature of the model. The MAE of this model is: MAE Normalizedb= 0.39400684388 & MAE SCORE = 13171.086606. We can see that there is already an improvement in this model.

Using similar technique, we computed several models using different features, the result is in the chart below:

Features	MAE Norm	MAE Score	Accuracy
AveSalary	0.781204005	21838.72	0.218795995
AveSalary InLocation	0.454725897	14840.90	0.545274103
LengthOf JobDescription	0.43105197	14764.18	0.56894803
AveSalary InCompany	0.39400684	13171.09	0.60599316
NumberOf JobsInLocation	0.40904916	15865.55	0.58095084
AveSalaryIn Title+ContractType	0.36224943	13337.20	0.63775057

C. Complex Models

We believe that more complex model of the features would make the prediction more accurate. Therefore, we started to combine the features together.

a. **Model 1:** *Combine all the features*

We first combined all the features together as one basic feature to predict. The MAE for this model is: MAE Normalized = 0.32913566735 & MAE SCORE = 10363.578615

b. **Model 2:** *Create 3 groups and categorize the data point into ""/"full_time"/"part_time", generate theta within the group*

Then we realized that jobs are categorized into 3 groups by contract type: full_time, part_time and not specified. We believed that if we grouped the data points into those three groups, and have individual thetas for each group, we might have a better model for the prediction. The MAE for this model is: MAE Normalized = 0.32813957995 & MAE SCORE = 10357.978534. This does not improve too much because there are too many data in the category of not specified, which makes the feature ambiguous.

c. **Model 3:** *Create 3 groups and categorize the data point into ""/"permanent"/"contract", generate theta within the group*

Similarly, we used contract time as a feature to categorize the data into permanent, contract and not specified. The MAE for this this model is: MAE Normalized = 0.30797619889 & MAE SCORE = 9310.459168

In addition, we used text mining. We used most 500 common unigrams and 500 common bigrams as the features, but neither of them have a better result.

D. **Dealing With Noise/Overfitting**

We can see that adding more features does not necessarily improve our model. One obstacle that prevents us from getting a better performance is the large amount of noise in the data. From the diagrams that we showed in part 1 and part 2, there is a correlation between the features and salary, but some of the correlations are very weak. Therefore, we need to delete those outliers from the training data to eliminate the noises. We first used the raw training data to get the parameters (thetas) for the model. Next, we run one round of the thetas on the training data, and store the data points that have difference between prediction and actual greater than 0.8. We assume that there is some sort of error in these data points due to human error, such as adding an extra 0 to the salary, or inputting wrong information. We then delete those data points from the training set and rerun the process to get new parameters. Finally run the model using validation set. From research, we realized that adding the square and cube of the features would also reduce the overfitting. Therefore, we also add the square and cube of the features to the basic feature.

The MAE for the model with regularization is: MAE Normalized = 0.275165830841 & MAE Score = 8583.914206

The MAE for the model eliminating the Noise is: MAE Normalized = 0.26378159615 & MAE Score = 7635.396851

4. Relevant Literature

The dataset comes from kaggle, called "predict the salary of any UK job as based on its contents", published by Adzuna, a job search engine based in UK. The dataset is used as a census of the people's income dataset. With the relevant features and exciting models we found, we started our research by predicting salary with classification or regression. If we want a binary answer, like (1 or 0), we can predict if a person's salary is below or above the average salary or threshold. We can also predict exact salary that a person will have if the person takes the job. To predict whether a person's salary is above or below the average, we have several options, classification tasks, such as logistic regression, support vector machine, etc.To predict the salary a person might have, we used regressions, such as linear regression, ridge regression, etc to build our model. Some other people chose to predict binary output of this dataset. We prefer to predict the real salary a person will have with contents of the dataset.

A. Linear Regression:

Linear regression is one of the simplest supervised learning approaches to learn the relationship between inputs and output. Inputs are features and output is prediction. For modeling the relationship between two variables, it fits the inputs and features into a linear equation (y = ax + b). In out prediction task, we used average salary per company, average salary per location, average salary per job category, and numbers of job advertisements per location published, as input features to linear equation.

$$y_i = \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

B. Tikhonov Regression: [3]

Tikhonov regression is also known as ridge regression. It is a supervised learning approach that learns the relationship between input vector and output. It addresses some problems with ordinary least square by penalizing size of coefficients. The difference between ridge regression and linear regression is that ridge regression penalizes on overfitting.

$$\min_{w} ||Xw - y||_2{}^2 + \alpha ||w||_2{}^2$$

C. Nearest Centroid: [3]

Nearest Centroid Classifier assigns samples in each class to be their centroid, and the method detects whose centroid is nearest to observations. Nearest centroid classifier is a simple algorithm that represents each class with the centroid of members of the class. This is a similar process to the k-means process.

D. Nearest Neighbor Regression: [3]

When data labels are continuous rather than discrete, nearest neighbor regression is a good choice. A label of data will be assigned to a query point, the nearest centroid of its nearest neighbors.

E. Support Vector Machine(SVM): [2]

Some people also predicted binary solutions with similar datasets. By predicting output 0 or 1, support vector machine is the method they use. Using Support Vector Machine can minimize the misclassification error in classification. Also, the method of Support Vector Classification can be used to solve regression problems. The method is called Support Vector Regression(SVR). Because SVR ignores any training data that close to the prediction when build the model, the SVR depends on subset of training data.

F. Random Forest: [3]

Random forest comes from a general technique called random decision forest. A random forest is an ensemble learning method for classification and regression, etc. The random forest can fit a number of classifying decision trees on various dataset and sub-samples. A single decision tree will lead overfitting, but random forest can improve test accuracy largely by reducing the overfitting.

The state-of-art algorithm is Extremely randomized trees[1]: The Extremely Randomized Trees contains ExtraTreeClassifer class and ExtraTreeRegressor class. Same as random forests, extremely randomized trees also use a random set of candidate features; however, thresholds are picked randomly on each candidate feature, and the splitting rule is selected by the best of those randomly-conducted thresholds. In this way, we can minimize the variance of the model. We tried extremely randomized tree on our data sets. The outcome is similar to the random forest regressor but not better.

Our method is similar to other regression methods implementation. Same features, like location, company, category, contract type have been used in the past studies[4] from census data and most used similar approaches to tackle their tasks.

5. Result/Conclusion

To improve our performance, we tried to apply different models that we have researched on to our data set. We used Linear Regression as our baseline classifier. We tried Support Vector Machine - Support Vector Regression, Nearest Centroid, Linear Regression, Logistic Repressor, K Neighbors Regressor, and Random Forest Regressor. The table below shows the MAE and accuracy to the corresponding classifier:

Methods	MAE Normalized	MAE Score	Accuracy
SVM.SVR	0.393471035	12714.720205	0.606528965
Nearest Centroid	0.348165830	12083.914206	0.651834170
Linear Regressor	0.263781596	7635.396851	0.736218404
Logistic Regressor	0.257162178	7082.585793	0.742837822
KNeighborsRegressor	0.189628491	6325.819339	0.810371509
RandomForestRegressor	0.184378163	6064.261172	0.815621837

The features we used are average salary per location, length of job description, average salary per company, number of jobs per location, average salary in each job title, contract time and contract type. Average salary in each job title + contract type work well on our data dataset. Average salary per location does not help much on the dataset. We think average salary per company may depends on contract type, because if a company offers more full time positions than part-time, the average salary might be higher. Furthermore, if there are more jobs posted in one location, there might be some big companies in this location and the salary will be higher for those companies.

With the results shown above, we realized Random Forest Regressor gives us the best result. At first, we thought K Neighbors Regressor is the best solution because merging each data point as its centroid may have a positive effect on eliminating the noises; however, the result is worse than Random Forest Regressor. The reason might be K Neighbors Regressor does not improve predictive accuracy with averaging and control the overfitting as good as Random Forest Regressor does. Additionally, we think Support Vector Regression gets the worst result because there are many data points around average and a few data points at two edges. Support Vector Regression just put more weights on the data points that near the decision boundary. In this data set, most points are close to the decision boundary; thus SVR would not help.

Overall, the most efficient method that we found was the random forest regressor.

References:

1. Geurts, Pierre, Damien Ernst, and Louis Wehenkel. "Extremely Randomized Trees." Springer Science Business Media, Inc, 2 Mar. 2006. Web. 1 Dec. 2015.

2. Smola, Alex, and Bernhard Scholkopf. "A Tutorial on Support Vector Regression." *Neurocolt.com*. Produced as Part of the ESPRIT Working Group in Neural and Computational Learning II NeuroCOLT. Web. 1 Dec. 2015.

3. "Scikit-learn." *: Machine Learning in Python — 0.17 Documentation*. Web. 1 Dec. 2015.

4. Lane, Terran, and Ronny Kohavi. *Census-Income (KDD) Data Set*. Print.