EXPECTED SALARY PREDICTOR

Data Science Final Project

Members

Maham Tariq	17K-3658
Eisha Tir Raazia	17K-3730

Presented To

Dr. Atif Tahir

INTRODUCTION

A salary predictor is created using the job listings from an employment website, in this case, Glassdoor.com. A data mining technique is used to generate a model which will scrape the number of jobs from the employment website, clean it on the basis of a number of factors including the rival companies, revenue, and skill required thereby predicting the salary to be expected when applying for a job. Techniques like linear regression, lasso regression, random forest regressors are optimized using GridsearchCV to reach the best model. Another variation of modeling is applying the NN on JD to predict the salary. The model can be further extended to build a fast API thus can be deployed on the internet for public usage.

Methodology:

Methodology And Steps Involve:

- Setting the research goal
- Retrieving Data
- Data Cleaning and Preparation
- EDA and Visualizations
- Model Building and Evaluation
- Presentation/Model into Production

Setting the research goal

The research goal of the project was to predict salaries based on different variables
This will help the candidate to negotiate their income when they enter the job process.

Retrieving the data

Through the selenium-powered web-scraper, around 10000 job postings(against 14 different job titles) were being scraped from glassdoor.com. The data collected from the glassdoor included the following data points:

- Job Title
- Company Name
- Salary Estimate
- Job Description
- Rating
- Company Size
- Company Founded Date
- Type of Ownership
- Industry
- Sector
- Revenue

Size of valid scraped values: 7984

Data Cleaning and Preparation

Following changes were made in order to clean and prepare the data:

- Removed rows without salary
- Removed Duplicates
- Parsed numeric data out of salary
- Made columns for employer-provided salary and hourly wages
- Converted Hourly salary to annual to make column values consistent
- Column for simplified job title(job tag) and Seniority level
- Column for job description(JD) length
- Made a new column for company state
- Transformed founded date into the age of the company
- Made columns for of different skills were listed in the job description:
    - Python
    - R
    - Excel
    - AWS
    - Spark
    - SQL
    - Django
    - Java
    - DB
    - Tableau
    - Azure
    - Hadoop
- Replacing all numeric values with 'Unknown', in the ‘Type of ownership’ column.

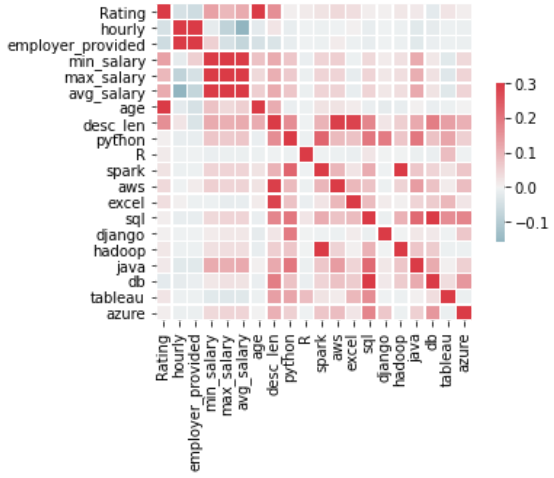
Exploratory Data Analysis (EDA), and Visualizations

In EDA, we looked at several data distributions and patterns in other to get an idea of how they correlate with the 'salary'.
Some of the main steps that EDA and visualization part included are:

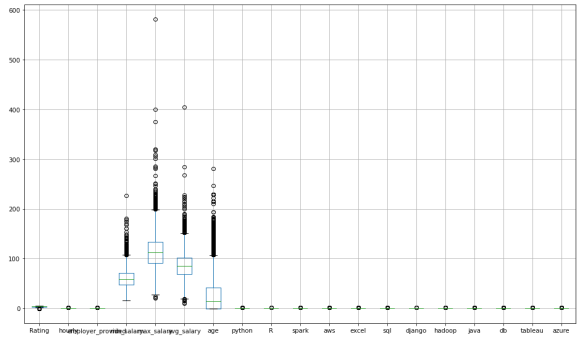
- Correlation heatmap
- Box plot
- Histogram and data distribution plots
- Scatter Plot with regression lines
- Pivot tables
- Word Cloud image

Following are some of the important visualizations:

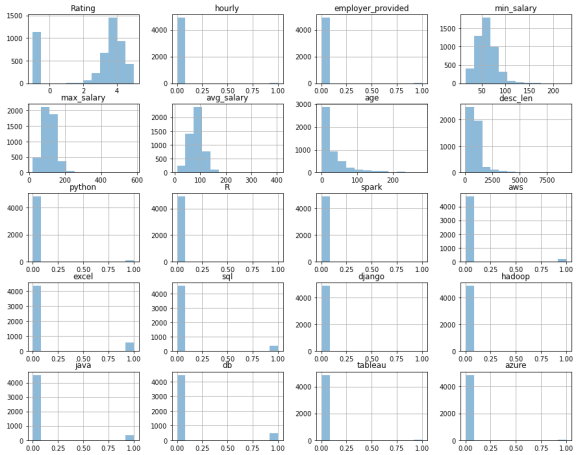
Correlation HeatMap



Box Plot



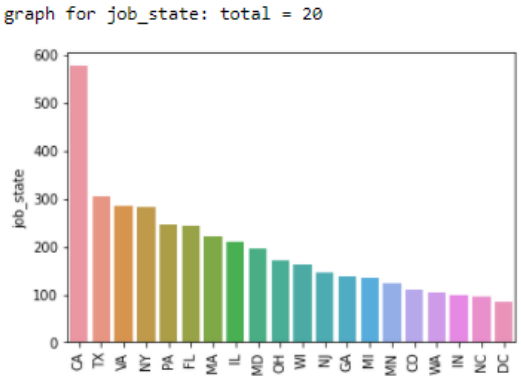
Histogram

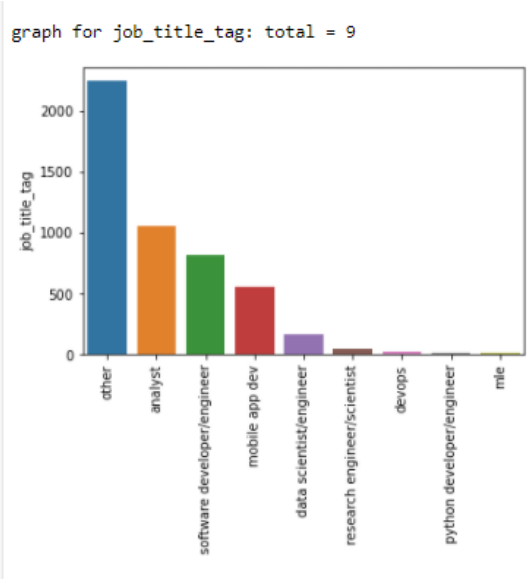


Pivot Table

job_title_tag	avg_salary
analyst	73.947947
data scientist/engineer	111.136905
devops	106.413043
mle	114.909091
mobile app dev	108.679091
other	79.769933
python developer/engineer	96.750000
research engineer/scientist	86.820513
software developer/engineer	99.061736

Other Visualizations





Model Building and Evaluation

Problem Category: Regression.

First, we transformed the categorical variables into dummy variables(one-hot data encodings of the categorical columns). We also split the data into train and test sets with a test size of 20%.

We implemented 2 types of models' use cases:

- NN-based model to predict Salary using JD embeddings.
  Results:
  - mae: 11.6152 (best possible minimum value), mse: 231.2114, and msle: 0.0337
- ML regression models to predict salary using other features i.e: company size, job title, etc
  Results:
  - Random Forest: MAE = 13.10 (best minimum)
  - Linear Regression: MAE = 01430.09
  - Lasso Regression: MAE = 14.93

Predictions and Microservice

After saving the best models to a directory or on the cloud, they are then loaded to make predictions on unseen data

The neural net predictions after loading the Keras model is presented below:

```
print('predicted:', y[0][0], ' ---> actual: 74K- 139K', )
```

```
predicted: 124.784546  ---> actual: 74K- 139K
```

On order to make predictions, an endpoint microservice through FAST-API is created which returns the predicted salary in the response body.

Response body


```
{
  "Predicted Salary": 124.7845458984375
}
```


CONCLUSION & FUTURE WORK

Looking at the current results, this can be concluded that for current data and model the JD-based prediction approach can be a better choice to predict the expected salary. One thing that can be done in the future is the addition of corpus for skills related to each stack and then parsing the JDs accordingly in order to predict for different job roles.

REFERENCES

1. https://towardsdatascience.com/selenium-tutorial-scraping-glassdoor-com-in-10-minutes-3d0915c6d905
2. https://towardsdatascience.com/the-most-in-demand-skills-for-data-scientists-in-2021-4b2a808f4005
3. https://pandas.pydata.org/docs/