Exploring the Data Job Market: An Analysis and Prediction of Job Listings and Salary Trends

Milestone: Exploration of Candidate Data Mining Models, and Select the Final Model or Models

Group 7
Valli Meenaa Vellaiyan
Hrithik Sarda

857-832-0123
978-654-0445

vellaiyan.v@northeastern.edu
sarda.h@northeastern.edu

Percentage of effort contributed by student 1: _____50_____

Percentage of effort contributed by student 2: _____50_____

Signature of student 1: _____Valli Meenaa_____

Signature of student 2: _____Hrithik Sarda_____

Submission date: _____3rd March 2023_____

Problem Setting

The problem setting is as follows: analyze and understand the current job market for Data Scientists, Data Engineers, and Data Analysts, such as identifying:

- The most in-demand skills and qualifications
- The most common industries hiring data scientists/engineers/analysts etc.
- The regions or cities with the highest concentration of data job openings
- The average salary range for various positions
- The most common job titles
- The most common words and phrases used in job descriptions to identify specific tasks and responsibilities that are most sought after by employers.
- Most importantly, to use the job descriptions to develop a model that can predict the job's salary range, given the job description.
- Additionally, it can also be used to build a model that can predict the job industry, given the job description.

A variety of techniques such as data cleaning, data visualization, statistical analysis, and machine learning will be used to understand and provide necessary insights from the dataset.

Problem Definition

The project's goal is to solve the issue of unreported salaries for data scientists, data analysts, and data engineers. Companies can also use this methodology to determine the appropriate starting compensation for new hires. A model is developed based on various criteria to group firms for staff transfers. The projected pay from the model might help employees choose the next ideal position. The project uses job descriptions as input to forecast job attributes such as income range, industry, job title, and other aspects of the work to study and understand the employment market for data scientists, engineers, and analysts. In summary, this model uses machine learning techniques to analyze and understand the job market for data scientists/analysts/engineers by predicting prospective job characteristics.

Data Sources

The datasets for “Data Analyst Jobs”, “Data Scientist Jobs”, and “Data Engineer Jobs” have been taken from https://www.kaggle.com/, an open-source, secure online community, which allows users to browse through numerous datasets. The links for the three datasets are provided below:

i) https://www.kaggle.com/datasets/durgeshrao9993/data-analyst-jobs-datset

ii) https://www.kaggle.com/datasets/andrewmvd/data-scientist-jobs

iii) https://www.kaggle.com/datasets/andrewmvd/data-engineer-jobs

Data Description

After combining the three datasets, the final dataset contains 16 columns (15 attributes and 1 target variable - “Salary Estimate” and 8676 rows. The dataset contains job listings for data scientist roles, data analyst roles, and data engineer roles. It includes several fields such as job title, company name, location, salary, and job description. All the attributes and their descriptions are given below:

No.	Attribute	Description
1.	ID	To identify each record uniquely
2.	Job Title	The title of the job listing
3.	Salary Estimate	The salary range for the job, and this is our response variable

4.	Job Description	A text description of the job responsibilities and qualifications
5.	Rating	Job rating out of 5
6.	Company Name	The name of the company that posted the job listing
7.	Location	The location of the job
8.	Headquarters	Headquarters of the company providing the respective job role
9.	Size	Number of employees working at the company currently
10.	Founded	The year in which the company was founded
11.	Type of Ownership	Type of business [Public, Private, Sports, etc.]
12.	Industry	The industry in which the company operates
13.	Sector	The sector to which the industry belongs to
14.	Revenue	Total income of the company.
15.	Competitors	Other companies that are current competitors to the listed company
16.	Easy Apply	Tells if it is easy to apply or requires specific reference/connections

Table 1: Initial Column Names with their Descriptions

Data Collection (integration of all the three datasets)
To begin our analysis, we initially reviewed the three datasets (pertaining to data analysts, data scientists, and data engineers) independently, with the aim of exploring the distinct columns of each dataset individually. Upon analysing the three datasets, we identified an extraneous column in the Data Analysts dataframe that we dropped, as well as two unnecessary columns in the Data Scientists dataframe that were also removed. However, we did not find any such columns in the Data Engineers dataset.

Next, we merged the three individual dataframes into a consolidated dataframe named "df_comb". Following this, we removed any duplicated rows from the merged dataframe, resulting in the elimination of 16 rows. Ultimately, our final dataframe contained 8674 rows.

Data Cleaning
I. Dropping columns based on number of null values:
To begin with, we computed the number of missing values in each column of the combined dataset. Next, we visualized this data using a horizontal bar chart (Fig 1). The chart allows for an easy comparison of the number of missing values across the different columns.

The column with the highest number of null values is "Easy Apply", with 8286 null values (95.52%). This indicates that a vast majority of the job postings in the combined dataset do not have an easy apply option. Hence, we remove that column. Additionally, we chose to drop the "Competitors" column as well, as it had 71.84% null values (6232). The decision to remove the columns was made based on the understanding that null values may hinder our ability to analyze the data in the columns, and we could obtain useful insights without those columns.

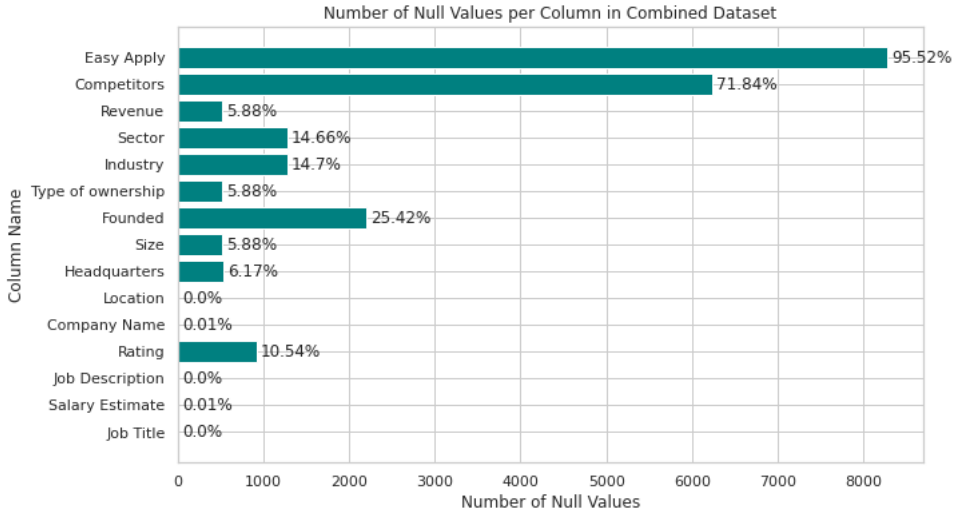

Fig 1: Horizontal Bar Chart to represent number of null values in each column

II. Cleaning "Company Name":

Moving on, we made use of various data cleaning techniques to ensure consistency and reduce redundancy in the columns. In the next step of our data cleaning process, we focused on cleaning the "Company Name" column. Fig 2 represents the state of the "Company Name" column before cleaning, while Fig 3 depicts the state of the column after cleaning.

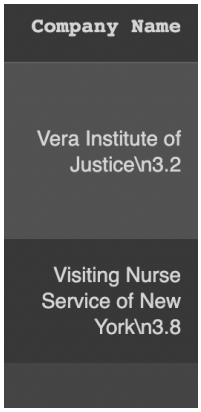

Fig 2: Company Name before cleaning

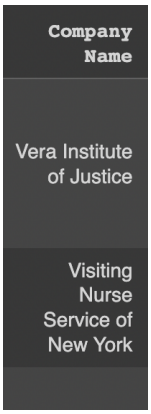

Fig 3: Company Name after cleaning

III. Cleaning "Salary Estimate":

We proceeded to split the "Salary Estimate" column into two separate columns, namely "Starting_Salary" and "Ending_Salary". We also cleaned the column by removing the dollar symbol and "K", multiplying the numerical values by 1000, and converting the values to float type for ease of future analysis. We noticed that in a dataset with nearly 9000 data points, there were only approximately 111 unique values for the starting and ending salaries. To address this, we added a bit of noise, or jitter, to the salary values to improve the diversity of the data. To accomplish this, we utilized the numpy.random.normal() function and generated random numbers from a normal distribution, adding the result to each salary value. The amount of jitter added was equivalent to 10% of the standard deviation of the salary data. By performing these steps, we were able to prepare the salary data for further analysis.

Salary Estimate
$37K-$66K
$37K-$66K

Fig 4: Salary Estimate before cleaning

Starting_Salary	Ending_Salary
39277.0	65008.0
37755.0	62535.0

Fig 5: Salary Estimate after cleaning

Therefore, we have two new Variables: 1) Starting_Salary 2) Ending_Salary. We combine these two columns into a single variable called **"average_salary"**, which is our **response/target variable**.

IV. Cleaning "Size" Column:

We moved on to cleaning the "Size" column. We started by splitting the column into two separate columns, namely "Min_Size" and "Max_Size." We did this by using the split() function and removing any strings or special characters that could affect the column's quality. Once the column was split, we converted both the columns to numeric datatype.

Size
201 to 500 employees
10000+ employees

Fig 6: Size before cleaning

Min_Size	Max_Size
201	500
10000	NaN

Fig 7: Size after cleaning

V. Cleaning "Revenue" Column:

Now, we clean the revenue column by removing "(USD)", dollar symbols, any special charactors, replacing "million" or "billion" with the respective number of zeroes, transforming the datatype to numeric, and splitting the column into two: "Minimum_Revenue" and "Maximum_Revenue".

Revenue
$100 to $500 million (USD)
$2 to $5 billion (USD)

Fig 8: Revenue before cleaning

Minimum_Revenue	Maximum_Revenue
100000000.0	500000000.0
2000000000.0	5000000000.0

Fig 9: Revenue after cleaning

VI. Creating a “job_state” column from “Location”:
We split the “Location” column on “,” and extract the state from it, and assign it to a new column called “job_state”.

VII. Finding Company’s Age from “Founded” column:
We perform a small mathematical calculation to compute the company’s age, based on the year provided in the “Founded” column.

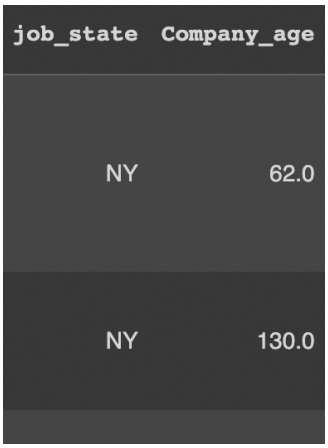

Fig 10: job_state and company_age columns

VIII. Working on the “Job Title” column:
Since “Job Title” is a categorical variable, we will be required to one-hot encode it, in order to be able to use them as predictors for our final model. However, this column has a very huge bracket of roles. Therefore, we first write a loop to classify the job titles into just nine classes and add them as additional features to our dataset (data engineer, analyst, data scientist, machine learning engineer, AI engineer, cloud engineer, manager, director, and others). We also create another column called “seniority” based on the “Job Title” column, and we obtain two classes for the same (senior or junior). We further convert the seniority column into an ordinal categorical variable by doing label encoding.

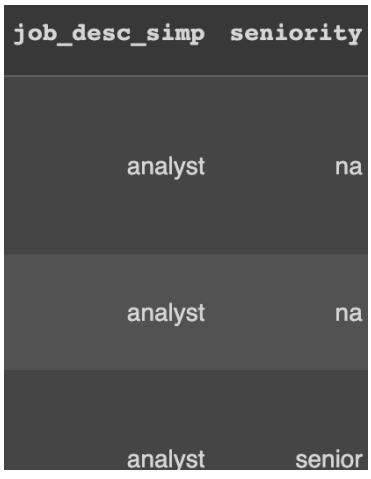

Fig 11: job_desc_simp and seniority columns

IX. Converting the “Competitors” column into a numerical variable:
We convert the “Competitors” column into a numerical variable by counting and assigning the number of competitors each company has.

```
df_comb['Competitors']

0       0
1       0
2       1
3       0
4       1
       ..
8159    0
8160    0
8161    0
8162    0
```

Fig 12: modified competitors column

X. Working on the "Ownership" column:

This column also requires us to do one-hot encoding to convert the categorical variables into additional features based on the number of unique values in this categorical feature. "Ownership" is converted into five different columns (private, public, educational_institution, government, and other organizations).

```
df_comb['Type of ownership']

0       other organisations
1       other organisations
2                   private
3       other organisations
4                   private
              ...
8159                private
8160                private
8161                private
8162                 public
```

Fig 13: modified type of ownership column

Data Visualization and Exploration

We've visualized the starting salaries and ending salaries through box plots. By creating box plots for the starting and ending salaries, we were able to gain insights into the salary ranges and how they vary across the different job titles in our dataset. The distribution is mostly uniform, with the exception of a couple outliers.

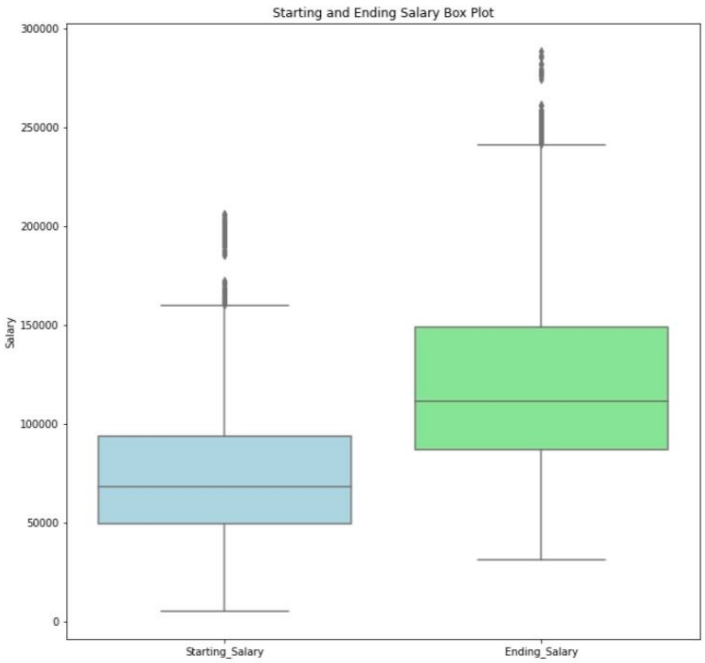

Fig 14: Box Plot visualization of Starting and Ending salaries

A distplot of the "Rating" column would give a graphical representation of the distribution of the "Rating" values in the dataset. It can be observed that the ratings of 1-5 are spread over all the different job postings in an almost normally distributed manner. The slightly left-skewed distribution implies that the median rating is likely higher than the mean rating. This can happen when there are a few companies with very high ratings that pull up the mean but not the median. The skewness can also indicate that there are more companies with lower ratings in the dataset than those with higher ratings. The outlier at rating 5 suggests that there may be some companies that are exceptional in terms of their ratings.

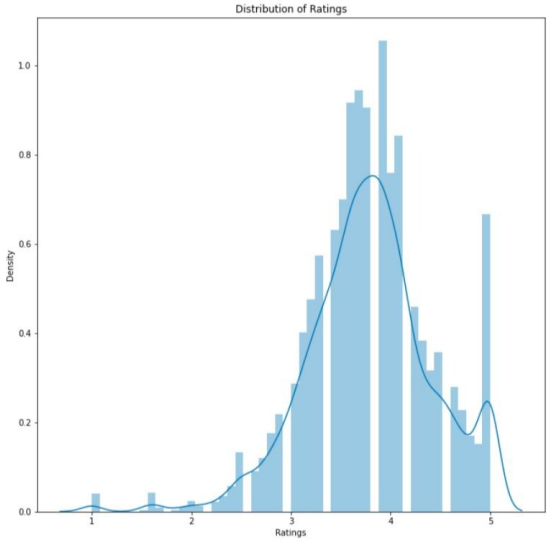

Fig 15: Distplot of Ratings

We've also created distplots for the minimum revenue and company age columns for filling the missing values:

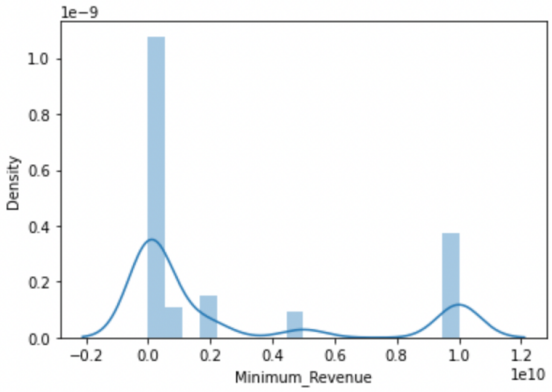

Fig 16: Distplot of minimum revenue

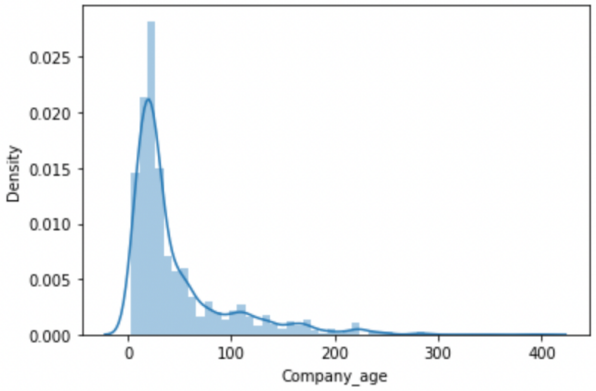

Fig 17: Displot of company age

Next, we've taken mean salary, average size of the company, rating, and average revenue generated by the company each year to create a pairplot, as these are the only numerical columns. We cannot observe any correlation between these variables as we have many overlapping values. We also observed that we would need to use jittering for better visualization.

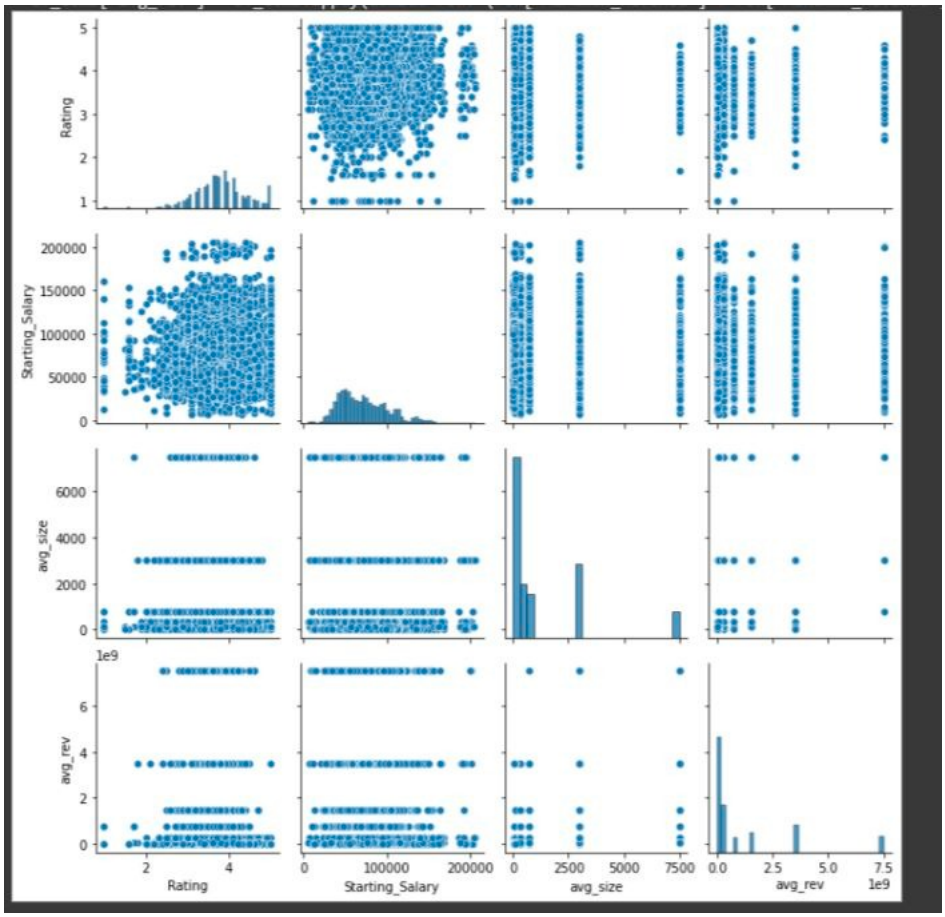

Fig 18: Pairplot among all numerical variables

We have three categories of jobs in our dataset. The below chart shows the names of the top 20 companies that provide these roles in bulk.

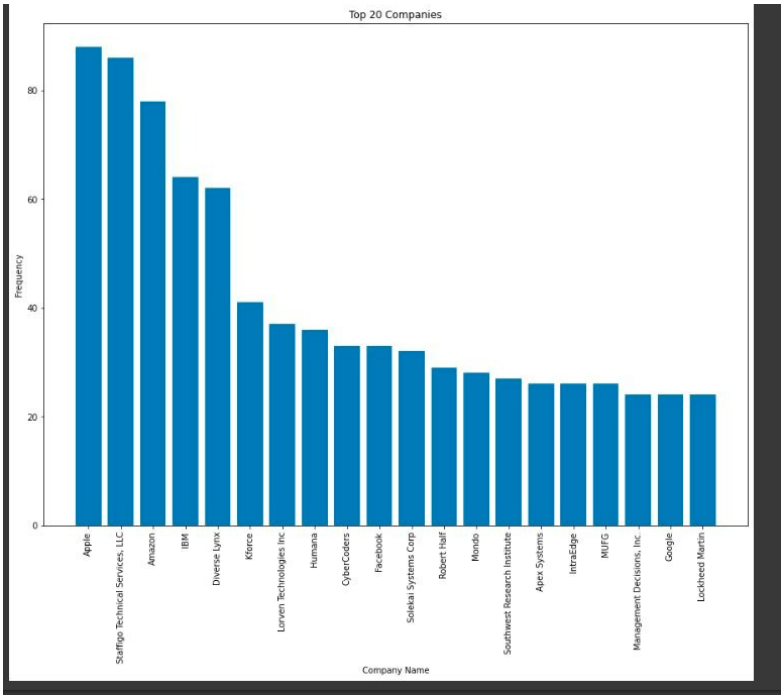

Fig 19: Bar plot for top 20 companies offering DA, DE, and DS roles

Data Processing

I. Sector vs. Mean Salary

We can use a statistical measure called Cramer's V, which is a measure of association between two nominal variables (mean salary and Sector). It ranges from 0 to 1, where 0 indicates no association and 1 indicates a perfect association. After performing a Cramer's V test on the "Sector" column and the mean salary, we obtained a low value of 0.0021. This value suggests that there is no significant association between the Sector that a specific company/job belongs to and our response variables. This means that the salary is independent of sector and vice-versa.

```
import researchpy as rp
import pandas as pd
df_comb["mean_salary"] = df_comb.apply(lambda row: (row["Starting_Salary"] + row["Ending_Salary"])
crosstab, test_results = rp.crosstab(df_comb['Sector'], df_comb['mean_salary'], test='chi-square')
cramers_v = np.sqrt(test_results.iloc[2,1] / (df_comb.shape[0] * (min(crosstab.shape) - 1)))
print(f"Cramer's V: {cramers_v}")

Cramer's V: 0.002140338012218702
```

Fig 14: Cramer's V Test on Sector and Mean Salary

II. Processing "Job Description"

We'll be performing Natural Language Processing (NLP) on the Job Description column. Job description column consists of a string of words in the form of a paragraph. Each paragraph contains information about the job posting, required qualifications and skills. We merged the entire column of Job Descriptions into one big text chunk, and found the frequency of words using NLP. This involves preprocessing the text data, tokenizing the text, converting it into a numerical representation using count vectorization or TF-IDF, and analyzing the frequency of words. We chose 200 such words and then manually filtered 60 skills ['microsoft office', 'oral', 'written','customer service', 'regression', 'classification', 'problem-solving','excel', 'reasoning', 'communication', 'charting', 'python', 'sql', 'management', 'creative', 'dashboards', 'machine learning', 'data visualisations', 'tableau', 'powerbi', 'r', 'etl', 'extract/transform/load', 'nlp', 'nltk', 'natural language processing', 'aws', 'predictive modelling', 'computer vision', 'cloud computing', 'software', 'azure', 'gcp', 'distributed system', 'data reporting', 'cleaning', 'modelling', 'big data', 'deep learning', 'neural network', 'statastical', 'analysis','modeling', 'databases', 'mongodb', 'network', 'cypher', 'hadoop', 'analytics', 'hadoop', 'spark', 'flask', 'florish', 'bachelor','master','data science','computer science','mathematics','statstics']. We then stored the skills in a list and created a data frame with these skills as columns. Now, we go over each row in our original dataframe and check if each job description contains any of the skills, and if it does, then we assign 1 to a particular skill, else 0. Finally, we merge the skills dataframe with our "df_comb".

	microsoft office	oral	written	customer service	regression	classification	problem-solving	excel	reasoning	communication	...	hadoop	spark	flask	florish	bachelor	master	data science	computer science	mathematics	statstics
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
669	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
670	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
671	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
672	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
673	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

Fig 15: Pivot form of skills

The number of years of experience in each job description is present in various formats, we're working on its extraction.

III. One-hot encoding of "Type of Ownership" column:

We have converted this categorical variable into a numerical variable using one-hot encoding. We will using the long pivot form of the dataframe for predicting salaries.

one_hot_encoded

	College / University	Company - Private	Company - Public	Contract	Franchise	Government	Hospital	Nonprofit Organization	Other Organization	Private Practice / Firm	School / School District	Self-employed	Subsidiary or Business Segment	Unknown
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4	0	1	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8669	0	1	0	0	0	0	0	0	0	0	0	0	0	0
8670	0	1	0	0	0	0	0	0	0	0	0	0	0	0
8671	0	1	0	0	0	0	0	0	0	0	0	0	0	0
8672	0	0	1	0	0	0	0	0	0	0	0	0	0	0
8673	0	1	0	0	0	0	0	0	0	0	0	0	0	0

8674 rows × 14 columns

df_comb['Starting_Salary']

Fig 16: One-hot-encoded form of "Type of Ownership" column

Similarly, we've dine one-hot encoding for 'Sector', 'Type of Ownership', 'Industry', 'job_state', and 'job_desc_simp' columns.

```
[87] df_mod=pd.concat([model,sect_encoded,owner_encoded,industry_encoded,sts_encoded,title_encoded,ski],axis=1)
```

df_mod

ry	job_state	Company_age	Min_Size	Max_Size	size_null	Minimum_Revenue	...	hadoop	spark	flask	florish	bachelor	master	data science	computer science	mathematics	statstics
2.0	NY	62.0	201.0	500.0	0	100000000.0	...	0	0	0	0	0	0	0	0	0	0
7.0	NY	130.0	10000.0	10000.0	0	2000000000.0	...	0	0	0	0	1	1	1	1	1	0

Fig 17: Final One-hot-encoded form

Data Partitioning

After finishing the process of feature engineering (data cleaning, processing, manipulation, and transformation), we use MinMaxScaler and fit_transform to scale the required columns. Next, we split the dataset into training and test sets in the ratio of 85:25. This implies that 75% of the dataset is used for training the prediction models and the remaining 25% of the dataset is used as the test set for evaluation of the prediction performance of those models. X_train contains 6123 records and 149 variables while X_test contains 2041 records and 149 variables respectively. Y_train contains 6123 records and 1 variables while Y_test contains 2041 records and 1 variables respectively.

Data Mining Models

Our response/dependent variable (average_salary) is a continuous, numeric variable. Therefore, our project comes under regression problem.

1. Linear regression

Linear regression is a statistical technique used to create a model that shows the relationship between a dependent variable and one or more independent variables. In the case of salary prediction, the objective is to estimate the "average_salary" of an employee by considering the independent variables discussed earlier. The effectiveness of our linear regression model in generating accurate predictions depends on the quality and significance of the data used to construct the model.

Advantages:

a) One of the main advantages of using linear regression is that it is a relatively simple and straightforward statistical method to implement.
b) The output coefficients of the model are easy to interpret, making it easier to understand the relationship between the independent and dependent variables.
c) Linear regression is also ideal when there is a clear linear relationship between the dependent and independent variables, as it has less model complexity.

Disadvantages:

a) In some cases, a significant amount of feature engineering may be required to ensure that the model performs well.
b) If the independent variables are highly correlated, the accuracy of the model may be negatively impacted.
c) Linear regression models can be vulnerable to noise and overfitting, which may result in inaccurate predictions.

Implementation:

Testing Score			
	Accuracy	R2 Score	RMSE
0	0.294631	0.294631	30131.90248

2. Decision tree regression

Decision tree regression is an ML algorithm that uses a tree-like structure to model a problem. The algorithm works by recursively dividing the dataset into smaller subsets based on the value of a particular attribute. Each split is designed to maximize the information gain, which is the difference between the entropy of the parent node and the sum of entropies of the child nodes. The goal of the algorithm is to create a model that can accurately predict a continuous output variable (in our case, average salary). At each node, the algorithm selects the attribute that provides the most information gain, meaning the attribute that best splits the data into subsets that are as different as possible from each other in terms of the output variable. The process continues until a stopping criterion is met, such as when a minimum number of instances are reached, or the maximum depth of the tree is achieved. At this point, the leaf nodes represent the predicted value for the output variable.

Advantages:

a) This is another algorithm which is easy to understand as well as interpret, even for a layman.
b) It can handle both numerical as well as categorical data.
c) It can capture non-linear relationships between the input variables and the target variable.
d) Decision trees can also handle missing values and outliers in an effective manner.

Disadvantages:

a) Decision trees are also prone to overfitting.
b) Sometimes, making small changes in the data can cause large changes in the tree structure, therefore, decision trees are quite unstable.
c) They could also be biased towards independent variables that have more number of levels or values, so decision trees are not always reliable.

d) They can be sensitive to the order of the input variables, which can affect the tree structure, and ultimately affect the prediction performance.

Implementation:

```
Testing Score
   Accuracy  R2 Score          RMSE
0  0.303292  0.303292  29946.332301
```

3. Random Forest regression

Random Forest regression is an ML algorithm that can also be used for the prediction of salaries. It combines ensemble learning with decision trees to create many random decision trees (which are used as base learning models) drawn from the dataset, averaging the results to give a new output that leads to stronger predictions. It uses a technique called "bootstrap and aggregation", otherwise known as "bagging".

Advantages:

a) This model can also handle both numerical and categorical data.
b) It captures the non-linearity between the input and output variables, and it also handles missing values and outliers effectively.
c) In comparison with decision tree regression, it reduces the risk of overfitting by using an ensemble of multiple decision trees and a random sampling of input variables and training observations.

Disadvantages:

a) RF regression is computationally expensive as well as memory expensive, especially for large datasets like the one we're using.
b) It is more difficult to understand in comparison with decision tree regression because of the use of multiple trees and feature important measures.
c) It can be biased towards certain variable(s), especially when there are strong correlations between them.
d) The curse of dimensionality also comes into play in case of RF regression, that is, the performance of the model might degrade as the number of input variables increases.

Implementation:

```
Testing Score for random forest regressor
   Accuracy  R2 Score          RMSE
0  0.210072  0.210072  31886.891814
```

4. AdaBoost regression

One of the first ever boosting algorithms, AdaBoost helps in combining multiple weak learners into a single strong learner. Adaptive boosting is used as an ensemble method. The most common estimator used with AdaBoost is decision trees with one level which means Decision trees with only 1 split. These trees are also called Decision Stumps. What this algorithm does is that it builds a model and gives equal weights to all the data points. It then assigns higher weights to points that are wrongly classified. Now all the points with higher weights are given more importance in the next model. It will keep training models until and unless a lower error is received.

Advantages:

a) AdaBoost works well with a large number of predictors and automatically detects interactions between them. This feature makes it particularly useful for predicting salaries, where numerous factors like education, experience, job type, and location may interact.
b) Despite being a complex algorithm, AdaBoost is fast and efficient, making it suitable for large datasets.
c) AdaBoost can reduce overfitting compared to decision tree regression by using an ensemble of multiple weak models and adaptive weighting of the training instances.
d) AdaBoost is a flexible algorithm that can be adapted to different types of data and classification problems.

Disadvantages:

a) AdaBoost is highly sensitive to noisy data and outliers, which can negatively affect the performance of the weak models and the ensemble as a whole. It is, therefore, crucial to clean and preprocess data before using AdaBoost.
b) AdaBoost can be computationally expensive for large datasets as it involves iterative training of multiple models.
c) AdaBoost can be difficult to interpret, making it challenging to explain its results to non-experts.
d) AdaBoost may be biased towards certain variables/predictors when strong interactions exist between them, which can result in unreliable predictions. This issue can be mitigated by using techniques like regularization or feature selection.

Implementation:

```
Testing Score for Adaboost regressor
   Accuracy  R2 Score          RMSE
0  0.202644  0.202644  32036.463473
```

5. Support Vector Regression

Support vector regression (SVR) is an ML algorithm that can be used for predicting the average salary. It is based on the Support Vector Machine (SVM) algorithm and it is a type of regression analysis where the goal is to predict the value of a continuous target variable (here, average salary) based on multiple input variables. In SVR, the algorithm tries to find the best-fitting line or hyperplane that can predict the response variable while minimising the error. It uses a kernel function to transform the input data into a higher dimensional space where a linear model can be used to predict the target variable.

Advantages:

a) It is a very powerful algorithm that can handle both non-linear data as well as high dimensional data. It is hence very useful for predicting salaries as the relationship between the average salary and the predictors can be quite complex or non-linear.
b) It is a robust algorithm that can handle outliers in the data as well. Outliers in salary data might arise due to factors like bonuses and incentives, and SVR can handle them pretty effectively.
c) SVR has good generalization performance, therefore, it can perform well on new data.
d) The kernel function used in SVR can be customized to different types of data, making it flexible for different applications.

Disadvantages:

a) It can be computationally expensive for a large dataset like ours, with high dimensional data. It requires more computing power and time compared to other regression algorithms.

b) The choice of kernel function can have a significant impact on the performance of the algorithm, and it may not always be clear which kernel function to use for a given dataset.
c) It can be sensitive to the choice of hyperparameters, such as the regularization parameter and the kernel function parameters. This sensitivity can make it challenging to find the optimal hyperparameters for a given dataset.
d) Interpretation of the results is difficult, as it may be difficult to explain the relationship between the input variables and the predicted salary.

Implementation:

```
Testing Score for Support vector regressor
   Accuracy  R2 Score         RMSE
0  0.017193  0.017193  35567.42367
```

Inference from the above models:

Although we used 149 predictor variables for our models, their accuracy scores were observed to be low. This is likely due to the high variance in our data caused by the large number of variables. However, we plan to improve our data processing and manipulation techniques to reduce the number of predictor variables and increase the accuracy of the models. We will also perform hyperparameter tuning to optimize the performance of the models, ultimately leading us to select the best model for our project.