

Report on Salary range prediction

CAPSTONE PROJECT



August 15, 2024

bY: SARA MEMON

ADVISOR: MUSTAFA AHMED

PROGRAM ADVISOR: DAVID PARENT

**Table of Contents**

1. **Executive Summary**
   * 1.1 Executive Introduction
   * 1.2 Executive Objective
   * 1.3 Executive Model Description
   * 1.4 Executive Recommendations
2. **Introduction**
   * 2.1 Background
   * 2.2 Problem Statement
   * 2.3 Objectives
   * 2.4 Scope
3. **Literature Review**
   * 3.1 Overview of Previous Work
   * 3.2 Gaps in Literature
   * 3.3 Theoretical Framework
4. **Methodology**
   * 4.1 Data Collection
   * 4.2 Exploratory Data Analysis
     + 4.2.1 Null Values
     + 4.2.2 Statistics
     + 4.2.3 Near Zero Variance
   * 4.3 Data Preprocessing
     + 4.3.1 Handling Missing Values
     + 4.3.2 Feature Engineering
       - 4.3.2.1 Company Profile
       - 4.3.2.2 Experience
       - 4.3.2.3 Job Title
       - 4.3.2.4 Role
       - 4.3.2.5 Skills
       - 4.3.2.6 Benefits
       - 4.3.2.7 Salary Range
     + 4.3.3 Dropping Columns
     + 4.3.4 Visualizations
5. **Modeling**
   * 5.1 Modeling Process and Challenges Encountered
   * 5.2 Modeling Step Overview
   * 5.3 Model Comparison and Performance Metrics
6. **Understanding the Correlation Matrix**
7. **Class Imbalance Consideration**
8. **Conclusion and Recommendations**
   * 8.1 Impacts on Business Problem (Scope of the Recommended Model)
   * 8.2 Recommended Next Steps
9. **References**

**Executive Summary**

This capstone project explores the development of a predictive model to estimate salary ranges based on job descriptions, leveraging a dataset containing over 1.6 million records. The primary objective is to help individuals and organizations better understand the factors that influence salary, such as qualifications, experience, job title, role, sector, industry, and type of work.

**Executive Introduction**

The project begins with a thorough exploration of the dataset, including data cleaning, preprocessing, and exploratory data analysis (EDA) to identify key patterns and correlations. This foundational work sets the stage for building a robust predictive model.

**Executive Objective**

The primary goal of this project is to create a predictive model that can estimate salary ranges with a high degree of accuracy. By identifying and understanding the key factors that influence salary, the model aims to assist both job seekers and employers in making informed decisions related to compensation.

**Executive Model Description**

Various machine learning models were considered, including Gradient Boosting, decision trees, and random forests, to predict salary ranges. After evaluating these models based on performance metrics such as F1 score and Accuracy, the final model was selected for its strong predictive capability, ensuring both accuracy and reliability.

**Executive Recommendations**

Key findings reveal that factors like Qualifications, Skills and experience significantly impact salary outcomes. The model provides valuable insights that can guide job seekers in negotiating salaries and help employers establish competitive compensation packages. The implications of this study suggest that a data-driven approach to understanding salary trends can benefit both individuals and organizations. Future research could further refine the model and explore additional factors influencing salary, enhancing its applicability across different sectors.

**4. Introduction**

**Background**

In the modern job market, understanding the factors that influence salary is crucial for both job seekers and employers. With the increasing availability of job-related data, it has become possible to analyze and predict salary ranges based on various factors, such as qualifications, experience, job title, role, sector, industry, and type of work. Accurate salary predictions can empower individuals to make informed career decisions, negotiate salaries more effectively, and set realistic expectations. For employers, these predictions can aid in establishing competitive compensation packages that attract and retain top talent. As the job market continues to evolve, the ability to predict salary trends based on comprehensive data analysis is becoming increasingly valuable.

**Problem Statement**

The problem addressed in this project is the challenge of accurately predicting salary ranges based on a variety of job-related factors. While many job seekers and employers rely on anecdotal evidence or generic salary surveys, these methods often fail to account for the specific qualifications, experience, and industry factors that can significantly impact compensation. This project seeks to fill this gap by developing a predictive model that can estimate salary ranges using a large dataset of job descriptions, offering a more nuanced and data-driven approach to salary prediction.

**Objectives**

The primary objectives of this project are:

1. **Data Analysis:** To perform a thorough analysis of a large job description dataset, identifying key factors that influence salary.
2. **Model Development:** To develop a predictive model capable of estimating salary ranges based on job-related factors such as qualifications, experience, job title, role, sector, and industry.
3. **Model Evaluation:** To evaluate the performance of the model using relevant metrics and ensure its accuracy and reliability.
4. **Insights Generation:** To generate actionable insights from the model that can guide job seekers in their career planning and assist employers in setting competitive salaries.
5. **Tool Development:** To create a user-friendly tool or framework that can be utilized by both individuals and organizations for salary prediction.

**Scope**

The scope of this project is defined by several boundaries:

* **Data Source:** The project will focus on a dataset containing over 1.6 million job descriptions, which includes various factors such as qualifications, experience, job title, role, sector, and industry.
* **Predictive Focus:** The primary focus will be on predicting salary ranges. While other factors like job satisfaction or benefits could be explored, they are beyond the scope of this project.
* **Model Selection:** The project will consider and compare multiple machine learning models (e.g., linear regression, decision trees, random forests) to find the most suitable one for salary prediction.
* **Geographic Limitation:** The dataset may be limited to specific geographic regions, which will be considered when generalizing the findings.
* **Application:** The final model and insights will be applicable primarily to the dataset in question, with suggestions for how it could be adapted to other datasets or sectors.

**Literature Review**

**Overview of Previous Work**

Salary prediction and job market analysis have been extensively studied across various disciplines, including economics, human resources, and data science. Traditional approaches to salary prediction have often relied on statistical methods such as linear regression, which examines the relationship between salary and a set of independent variables like education, experience, and job title. For instance, studies have shown that factors such as education level, years of experience, and industry significantly influence salary levels. Additionally, surveys and reports from organizations like the Bureau of Labor Statistics (BLS) and Glassdoor have provided aggregated salary data, often used as benchmarks for compensation analysis.

In recent years, the advent of big data and machine learning has opened new avenues for salary prediction. Machine learning models, including decision trees, random forests, and neural networks, have been employed to capture complex, non-linear relationships between variables. For example, random forests have been effective in accounting for interactions between factors such as location and industry, while neural networks have shown promise in capturing intricate patterns in large, multidimensional datasets.

However, much of the existing work has focused on specific industries or geographic regions, limiting the generalizability of the findings. Moreover, many models do not account for the dynamic nature of the job market, where new roles and skills constantly emerge. This project aims to build upon these methods by applying them to a more comprehensive dataset that encompasses a wide range of sectors, job titles, and geographic locations.

**Gaps in Literature**

Despite the progress in salary prediction models, several gaps remain:

1. **Generalizability:** Many existing models are tailored to specific industries or regions, which limits their applicability to a broader job market. This project seeks to create a model that is more generalizable across different sectors and locations.
2. **Factor Interaction:** While some studies have explored interactions between variables, there is often a lack of focus on how multiple factors interact to influence salary. This project will explore these interactions more thoroughly using advanced machine learning techniques.
3. **Real-time Adaptation:** The job market is continuously evolving, yet many models do not account for the emergence of new roles or the shifting importance of certain skills. This project aims to address this by incorporating a diverse and up-to-date dataset.
4. **Holistic Approach:** Existing research often focuses on a limited set of variables (e.g., education and experience), neglecting other potentially important factors such as job sector, company size, and type of employment (e.g., full-time vs. part-time). This project will adopt a more holistic approach by including a wider array of factors.

**Theoretical Framework**

The theoretical framework underpinning this project is grounded in both economic theory and machine learning principles.

1. **Human Capital Theory:** This economic theory posits that individuals’ skills, knowledge, and experiences (collectively referred to as human capital) are key determinants of their productivity and, consequently, their earnings. According to this theory, higher education and more experience should correlate with higher salaries. This theory forms the basis for including qualifications and experience as key variables in the salary prediction model.
2. **Labor Market Segmentation Theory:** This theory suggests that the labor market is divided into distinct segments, each with different characteristics and wage-setting mechanisms. For example, the primary labor market may consist of well-paid, stable jobs, while the secondary market includes lower-paid, less secure jobs. This theory justifies the inclusion of factors like job sector and industry in the predictive model.
3. **Machine Learning Theory:** The project leverages supervised learning techniques, particularly regression-based models and ensemble methods like random forests, which are well-suited for predicting continuous outcomes such as salary. These models can handle complex relationships between variables and are robust to overfitting, making them ideal for large datasets with many features.
4. **Predictive Analytics Framework:** The project follows a predictive analytics framework, which involves data preprocessing, model selection, training, testing, and evaluation. This framework ensures that the model is not only accurate but also interpretable and applicable to real-world scenarios.

By integrating these theories, the project aims to develop a robust and comprehensive model for predicting salaries, contributing to both the academic literature and practical applications in the job market.

**Methodology**

**Data Collection**

The dataset used in this project comprises over 1.6 million job descriptions, sourced from Kaggle. It is a comprehensive database that aggregates job postings from various online platforms. This dataset includes a wide range of variables that are critical for predicting salary, such as:

1. **Job Id:** A unique identifier for each job posting.
2. **Experience:** The required or preferred years of experience for the job.
3. **Qualifications:** The educational qualifications needed for the job.
4. **Salary Range:** The range of salaries or compensation offered for the position.
5. **Location:** The city or area where the job is located.
6. **Country:** The country where the job is located.
7. **Latitude:** The latitude coordinate of the job location.
8. **Longitude:** The longitude coordinate of the job location.
9. **Work Type:** The type of employment (e.g., full-time, part-time, contract).
10. **Company Size:** The approximate size or scale of the hiring company.
11. **Job Posting Date:** The date when the job posting was made public,
12. **Preference:** Special preferences or requirements for applicants (e.g., Only Male or Only Female, or Both)
13. **Contact Person:** The name of the contact person or recruiter for the job.
14. **Contact:** Contact information for job inquiries.
15. **Job Title:** The job title or position being advertised.
16. **Role:** The role or category of the job (e.g., software developer, marketing manager).
17. **Job Portal:** The platform or website where the job was posted.
18. **Job Description:** A detailed description of the job responsibilities and requirements.
19. **Benefits:** Information about benefits offered with the job (e.g., health insurance, retirement plans).
20. **Skills:** The skills or qualifications required for the job.
21. **Responsibilities:** Specific responsibilities and duties associated with the job.
22. **Company Name:** The name of the hiring company.
23. **Company Profile:** A brief overview of the company's background and mission.

A close-up of numbers

Description automatically generatedA screenshot of a computer

Description automatically generated

The dataset is rich in detail and provides a robust foundation for developing a predictive model that considers a broad spectrum of factors influencing salary.

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a company profile

Description automatically generated

**Exploratory Data Analysis:**

**Null Values:**

**Data Preprocessing**

Data preprocessing is a critical step to ensure that the dataset is clean, consistent, and ready for analysis. The following steps were taken:

1. **Handling Missing Values:**
   * **Company Profile:** The "Company Profile" variable had 5,478 missing values. Since this variable might provide additional context but is not essential for the primary salary prediction task, these missing values were either left as-is or imputed with a placeholder indicating a missing profile, depending on the importance of this feature in model performance.
   * **Other Variables:** There were no missing values in the other variables, so no extensive imputation or removal was necessary.

A screenshot of a phone

Description automatically generated

A screenshot of a phone

Description automatically generated

**Statistics**

**. Job Id**

* **Count:** 1,615,940 — This indicates that there are 1,615,940 entries in the dataset for the Job Id column.
* **Mean:** 1.548935e+15 — The average Job Id is approximately 1.55 trillion. Since Job Id is likely an identifier, this number doesn't provide much analytical value beyond confirming that all jobs have unique identifiers.
* **Standard Deviation (std):** 8.946722e+14 — This value shows the spread of Job Id values around the mean.
* **Minimum (min):** 1.817948e+11 — The smallest Job Id is approximately 181.79 billion.
* **25th Percentile (25%):** 7.740508e+14 — 25% of Job Id values are below approximately 774.05 billion.
* **Median (50%):** 1.547858e+15 — The median Job Id is approximately 1.55 trillion, indicating that half of the jobs have Job Ids below this value.
* **75th Percentile (75%):** 2.323729e+15 — 75% of Job Id values are below approximately 2.32 trillion.
* **Maximum (max):** 3.099618e+15 — The largest Job Id is approximately 3.10 trillion.

**2. Latitude**

* **Count:** 1,615,940 — There are 1,615,940 entries in the latitude column, corresponding to the number of jobs.
* **Mean:** 1.937743e+01 — The average latitude is approximately 19.38 degrees. This value represents the central tendency of the job locations in terms of latitude.
* **Standard Deviation (std):** 2.355690e+01 — The latitude values vary by about 23.56 degrees around the mean.
* **Minimum (min):** -4.090060e+01 — The smallest latitude value is approximately -40.90 degrees, which could indicate a location in the Southern Hemisphere.
* **25th Percentile (25%):** 5.152100e+00 — 25% of the latitude values are below approximately 5.15 degrees.
* **Median (50%):** 1.807080e+01 — The median latitude is approximately 18.07 degrees.
* **75th Percentile (75%):** 3.907420e+01 — 75% of the latitude values are below approximately 39.07 degrees.
* **Maximum (max):** 7.170690e+01 — The largest latitude value is approximately 71.71 degrees, likely in the Northern Hemisphere.

**3. Longitude**

* **Count:** 1,615,940 — There are 1,615,940 entries in the longitude column.
* **Mean:** 1.639926e+01 — The average longitude is approximately 16.40 degrees.
* **Standard Deviation (std):** 7.066762e+01 — The longitude values have a standard deviation of about 70.67 degrees, indicating a wide distribution of job locations across the globe.
* **Minimum (min):** -1.751982e+02 — The smallest longitude value is approximately -175.20 degrees, likely indicating a location in the Western Hemisphere.
* **25th Percentile (25%):** -1.531010e+01 — 25% of the longitude values are below approximately -15.31 degrees.
* **Median (50%):** 1.914510e+01 — The median longitude is approximately 19.15 degrees.
* **75th Percentile (75%):** 4.757690e+01 — 75% of the longitude values are below approximately 47.58 degrees.
* **Maximum (max):** 1.780650e+02 — The largest longitude value is approximately 178.07 degrees, likely indicating a location in the Eastern Hemisphere.

**4. Company Size**

* **Count:** 1,615,940 — There are 1,615,940 entries in the Company Size column.
* **Mean:** 7.370467e+04 — The average company size is approximately 73,704 employees.
* **Standard Deviation (std):** 3.529886e+04 — The standard deviation is about 35,299 employees, indicating variability in the size of companies.
* **Minimum (min):** 1.264600e+04 — The smallest company size is approximately 12,646 employees.
* **25th Percentile (25%):** 4.311400e+04 — 25% of companies have fewer than approximately 43,114 employees.
* **Median (50%):** 7.363300e+04 — The median company size is approximately 73,633 employees.
* **75th Percentile (75%):** 1.043000e+05 — 75% of companies have fewer than approximately 104,300 employees.
* **Maximum (max):** 1.348340e+05 — The largest company size is approximately 134,834 employees.
* \A screenshot of a computer

  Description automatically generated

**Near Zero Variance:**

In the process of preparing the dataset for analysis, I conducted a check for near-zero variance (NZV) among the variables. Near-zero variance occurs when a variable has very little variability, meaning that almost all of its values are identical or fall into a very limited range of possible values. Variables with near-zero variance can be problematic in modeling because they provide little to no useful information for predicting the target variable and can even cause issues like multicollinearity.

To identify any such variables, I calculated the frequency of the most common value for each variable and examined the proportion of unique values relative to the total number of data points. Typically, a variable is considered to have near-zero variance if the most common value accounts for a large majority of the data and if there are very few unique values relative to the size of the dataset.

Upon conducting this analysis, I found that none of the variables in the dataset exhibited near-zero variance. This indicates that all the variables have sufficient variability to be useful in the predictive modeling process. As a result, no variables needed to be excluded based on this criterion, allowing the full set of variables to be retained for further analysis and model development. It means that none of the categorical variables in dataset have a near-zero variance based on the criterion we used (i.e., no single category constitutes more than 95% of the values in any column). This implies that all categorical variables have a diverse distribution of values.

**Feature Engineering:**

**Company Profile:**

As part of the data preprocessing and preparation process, I performed feature engineering to enhance the dataset by extracting additional relevant information. Specifically, I focused on the Company Profile column, which contained a variety of descriptive information about the companies.

Recognizing the potential value of understanding the industry and sector in which a company operates, I extracted these two features from the Company Profile column. Here's how this was done:

1. **Industry Extraction:**
   * The Company Profile column often included information about the specific industry the company operates in, such as "Healthcare," "Technology," "Finance," etc. I parsed the text within this column to identify and extract the industry associated with each company.
   * This extraction involved identifying common industry-related keywords and standardizing them across the dataset to ensure consistency.
2. **Sector Extraction:**
   * In addition to the industry, the broader economic sector to which a company belongs (e.g., "Public," "Private," "Non-profit") was also extracted from the Company Profile column.
   * Like the industry extraction, this process involved parsing the text to identify sector-related information and mapping it to standardized sector categories.

By extracting these two features—industry and sector—from the Company Profile column, I enriched the dataset with more granular and structured information. These newly created features are expected to play a significant role in predicting salary, as both the industry and sector of a company can significantly influence compensation levels.

This feature engineering step not only increased the amount of relevant data available for analysis but also helped in making the dataset more interpretable and useful for the modeling phase.

A screenshot of a website

Description automatically generated

**Experience:**

In the dataset, the Experience column provided the required or preferred years of experience for a job, but it was often presented as a range (e.g., "5-15 years," "2-12 years"). To make this information more useful for analysis and modeling, I separated the Experience column into two distinct features: Experience Min and Experience Max.

Here’s how this was done:

1. **Experience Min Extraction:**
   * For each entry in the Experience column, I extracted the minimum number of years specified in the range. For example, if the entry was "5-25 years," the value of Experience Min would be 5.
   * This feature captures the minimum required or expected experience for the job.
2. **Experience Max Extraction:**
   * Similarly, I extracted the maximum number of years specified in the range. For the entry "5-25 years," the value of Experience Max would be 25.
   * This feature represents the upper limit of the experience range, indicating the maximum amount of experience that is typically considered ideal for the position.

By creating these two separate features, Experience Min and Experience Max, I was able to provide more detailed and structured information about the experience requirements for each job. This allows the model to better understand the potential range of experience that employers are looking for, which can have a significant impact on the predicted salary.

Furthermore, these features can be used independently or combined (e.g., averaging them) to represent the experience requirement more flexibly, depending on the specific needs of the modeling process. This step-in feature engineering adds granularity and clarity to the dataset, making it more useful for predictive analysis.

A screenshot of a phone

Description automatically generated

**Job Title:**

To enhance the dataset's usability and improve the predictive power of the model, I performed an additional feature engineering step on the Job Title column. Originally, the Job Title column contained a wide variety of specific job titles, leading to a high number of unique values—147 distinct job titles in total. While this specificity can be valuable, it also increases the complexity of the model without necessarily adding significant predictive value, as many of these titles may have similar roles and responsibilities.

To address this, I created a new Generalized Job Title column, where specific job titles were grouped into broader, more standardized categories. This process involved the following steps:

1. **Analysis of Specific Job Titles:**
   * I carefully analyzed the 147 unique job titles to identify common themes and responsibilities. For example, titles like "Backend Developer," and "Frontend Developer" were all grouped under a more generalized title like "Developer."
2. **Creation of Generalized Categories:**
   * Based on this analysis, I created 46 generalized job titles that could effectively represent most of the specific titles in the dataset. This involved standardizing roles such as "Manager," "Analyst," "Consultant," "Developer," and so on, to capture the essence of the roles without unnecessary granularity.
3. **Mapping Specific Titles to Generalized Titles:**
   * Each specific job title in the original column was mapped to one of the 46 generalized titles. For example, all variations of managerial positions (e.g., "Project Manager," "Operations Manager") were grouped under the generalized title "Manager."

By reducing the number of unique job titles from 147 to 46, I was able to simplify the dataset while still retaining meaningful distinctions between different types of jobs. This not only reduces the dimensionality of the dataset, making it easier to model, but also helps to prevent overfitting by ensuring that the model focuses on significant differences rather than noise created by overly specific job titles.

This generalized approach makes the dataset more manageable and improves the model’s ability to learn from the data, ultimately leading to more robust and accurate salary predictions.

**Role:**

To further enhance the dataset and streamline the analysis process, I applied a similar approach to the Role column as I did with the Job Title column. Initially, the Role column contained 376 unique values, representing a wide array of specific roles. To make the dataset more manageable and to improve the effectiveness of the predictive model, I created a Generalized Role column by categorizing these specific roles into broader categories, reducing the unique values from 376 to 64.

Here’s how I achieved this:

1. **Role Categorization Function:**
   * I defined a function, categorize role, to systematically group specific roles into generalized categories. The function looks for key terms in each role name and maps them to a predefined category. For example, any role containing the word "manager" was categorized as "Manager," and roles containing "analyst" were categorized as "Analyst."
2. **Implementation:**
   * The categorize role function was applied to each entry in the original Role column. This automated process ensured that specific roles like "Senior Project Manager," "Business Analyst," and "Software Developer" were grouped into broader categories like "Manager," "Analyst," and "Developer."
3. **Resulting Generalized Role Column:**
   * The transformation resulted in a new Generalized Role column with only 64 unique values, down from the original 376. This reduction in the number of unique roles made the dataset more concise and improved the model's ability to learn meaningful patterns without being overwhelmed by an excessive number of categories.

The categories defined in the categorize role function include:

* **Manager:** For roles containing "manager."
* **Analyst:** For roles containing "analyst."
* **Developer:** For roles containing "developer."
* **Engineer:** For roles containing "engineer."
* **Consultant:** For roles containing "consultant."
* **Administrator:** For roles containing "administrator."
* **Specialist:** For roles containing "specialist."
* **Assistant:** For roles containing "assistant."
* **Coordinator:** For roles containing "coordinator."
* **Director:** For roles containing "director."
* **Executive:** For roles containing "executive."
* **Technician:** For roles containing "technician."
* **Architect:** For roles containing "architect."
* **Lead:** For roles containing "lead."
* **Teacher:** For roles containing "teacher."
* **Supervisor:** For roles containing "supervisor."
* **Generalist:** For roles containing "generalist."
* **Writer:** For roles containing "writer."
* **Nurse:** For roles containing "nurse."
* **Secretary:** For roles containing "secretary."
* **Accountant:** For roles containing "accountant."
* **Banker:** For roles containing "banker."
* **Counselor:** For roles containing "counselor."
* **Attorney:** For roles containing "attorney."
* **Clerk:** For roles containing "clerk."
* **Lawyer:** For roles containing "lawyer."
* **Representative:** For roles containing "representative."
* **Scientist:** For roles containing "scientist."
* **Advisor:** For roles containing "advisor."
* **Programmer:** For roles containing "programmer."
* **Designer:** For roles containing "designer."
* **Therapist:** For roles containing "therapist."
* **Ambassador:** For roles containing "ambassador."
* **Sales:** For roles containing "sales."
* **Marketing:** For roles containing "marketing."
* **Customer Service:** For roles containing "customer."
* **Human Resources:** For roles containing "human." And so on.

By grouping the specific roles into these broader categories, the dataset became more structured, and the model could focus on the most important distinctions, rather than being distracted by overly granular variations. This step significantly improved the efficiency and effectiveness of the predictive modeling process.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a table

Description automatically generated A screenshot of a computer

Description automatically generated A screenshot of a phone

Description automatically generated

**Skills:**

In the dataset, the Skills column originally contained 376 unique values that were highly unstructured and varied greatly in specificity. To make this information more manageable and useful for analysis, I performed a categorization process that grouped specific skills into broader, more standardized categories. This process aimed to reduce the complexity of the dataset and enhance the model's ability to interpret and leverage the skills data effectively.

Here’s how the transformation was achieved:

1. **Analysis of Skills:**
   * I analyzed the 376 unique skills listed in the Skills column. These skills ranged from specific programming languages like "Python" and "JavaScript" to broader competencies like "Database Management" or "Web Development."
2. **Categorization of Skills:**
   * I systematically grouped these specific skills into generalized categories that represent broader skill sets. For example, all programming languages such as "HTML," "Python," and "JavaScript" were categorized under a single generalized skill: Programming.
   * Similarly, skills related to database technologies like "SQL" and "MongoDB" were grouped under Database, and web development frameworks such as "Django" and "React" were categorized as Web Development.
3. **Creation of a Generalized Skills Column:**
   * I created a new column called Generalized Skills, where each skill was replaced by its corresponding category. Here are some examples of how the skills were grouped:
     + **Programming:** HTML, CSS, JavaScript, React, Angular, Vue, Python, Java, C++, C#, PHP, Ruby
     + **Database:** SQL, NoSQL, MongoDB, PostgreSQL, MySQL, Firebase
     + **Web Development:** Django, Flask, Spring, Express, Node.js, WordPress, Magento, Shopify
4. **Handling Unstructured or Unclassified Skills:**
   * For skills that did not fit neatly into one of the predefined categories, I grouped them under a catch-all category called Others. This ensured that no skill was excluded from the analysis, even if it was unique or uncommon.
5. **Reduction in Unique Values:**
   * Through this categorization process, I was able to reduce the number of unique values in the Skills column from 376 to just 31 in the new Generalized Skills column. This significant reduction in complexity makes the dataset more structured and easier to work with.

By categorizing the skills in this way, I made the dataset more manageable and improved the model's ability to analyze and interpret the data. This approach also helps in identifying broader trends and patterns related to skill sets that influence salary, without being overwhelmed by the variability of specific, niche skills.

This feature engineering step ensures that the skills data is both comprehensive and concise, making it a powerful component of the predictive modeling process.

A screenshot of a computer

Description automatically generatedTop of Form

A screenshot of a computer

Description automatically generated

A close-up of a list of words

Description automatically generated

**Benefits:**

To further improve the structure and usability of the dataset, I focused on the Benefits column, which, although it originally contained only 11 unique values, was highly unstructured. To make this column more meaningful and consistent, I categorized the benefits into broader, more standardized categories, reducing the number of unique values to 10. This reorganization was done to enhance the clarity and interpretability of the data, making it easier to analyze the impact of different types of benefits on salary and other job-related factors.

Here’s how the Benefits column was restructured:

1. **Analysis of Benefits:**
   * I carefully reviewed the specific benefits listed in the Benefits column. These benefits included a range of offerings from health insurance to professional development opportunities, each varying significantly in nature and impact.
2. **Categorization of Benefits:**
   * I grouped these benefits into broader categories based on their primary focus or purpose. For example, various health-related benefits such as "Health Insurance" and "Wellness Programs" were categorized under Health, while financial incentives like "Bonuses" and "Retirement Plans" were grouped under Financial.
   * This systematic categorization ensured that similar benefits were grouped together, providing a clearer understanding of what types of benefits a job offered.
3. **Creation of a Structured Benefits Column:**
   * A new, structured Generalized Benefits column was created, where each specific benefit was replaced by its corresponding category. Here are the main categories and examples of benefits grouped under each:
     + **Health:** Health Insurance, Flexible Spending Accounts (FSAs), Wellness Programs, Health and Wellness Facilities
     + **Financial:** Retirement Plans, Bonuses and Incentive Programs, Profit-Sharing, Stock Options or Equity Grants, Financial Counseling
     + **Time Off:** Paid Time Off (PTO), Parental Leave
     + **Flexibility:** Flexible Work Arrangements, Relocation Assistance
     + **Support:** Employee Assistance Programs (EAP), Legal Assistance
     + **Recognition:** Employee Recognition Programs, Employee Referral Programs
     + **Development:** Professional Development, Tuition Reimbursement
     + **Insurance:** Life and Disability Insurance
     + **Other:** Transportation Benefits, Employee Discounts, Casual Dress Code, Social and Recreational Activities
4. **Reduction in Unique Values:**
   * By reorganizing the Benefits column in this manner, I reduced the number of unique benefit types from 11 to 10. This modest reduction was not just about decreasing the number of unique values but also about imposing a more logical structure on the data. This categorization provided a clearer and more consistent framework for analyzing how different types of benefits influence salary and job satisfaction.
5. **Handling of Unstructured Benefits:**
   * Any benefits that did not clearly fit into one of the predefined categories were placed into an Other category. This ensured that all benefits were accounted for without forcing them into ill-fitting categories.

This restructuring of the Benefits column ensures that the data is both organized and informative, making it easier to draw meaningful insights. The new Generalized Benefits column provides a clearer view of the types of benefits offered across different jobs and allows for more effective analysis of how these benefits correlate with other factors such as salary and job role. This step-in feature engineering further enhances the overall quality and interpretability of the dataset.

A screenshot of a computer program

Description automatically generated

**Salary Range:**

To refine the dataset and focus on predicting salary ranges, I performed a detailed transformation on the Salary Range column. Initially, the Salary Range column contained 561 unique salary ranges, which made the data highly variable and complex. To streamline this and make the salary prediction more manageable, I undertook the following steps:

1. **Extraction of Min and Max Salary:**
   * For each salary range in the Salary Range column, I extracted the minimum and maximum salary values. This allowed me to break down the ranges into their constituent parts, providing a clearer understanding of the salary spectrum for each job.
2. **Calculation of Average Salary:**
   * Once the minimum and maximum salaries were extracted, I calculated the average salary for each range. This average salary provides a single value that represents the typical salary for that job title, simplifying the analysis.
3. **Grouping by Job Title:**
   * I then grouped the data by the Job Title column that I had previously created, using the average salaries calculated for each salary range. By aggregating the salary data at the level of the generalized job titles, I was able to derive a more standardized view of what different roles typically pay.
4. **Creation of New Salary Ranges:**
   * Based on the grouped average salaries, I created new salary ranges for each job title. These new salary ranges were designed to reduce the complexity and variability in the data, effectively bringing down the number of unique salary ranges from 561 to just 4. The four unique salary ranges were carefully defined to cover the spectrum of salaries across different job titles, while also ensuring that the ranges were broad enough to be meaningful but distinct enough to be predictive.
5. **Target for Prediction:**
   * The newly created salary ranges now serve as the target variable for my predictive modeling. By reducing the target to 4 unique salary ranges, the model can focus on predicting which range a particular job title’s salary is likely to fall into. This approach not only simplifies the prediction task but also enhances the model’s accuracy by focusing on broader, more generalizable salary categories.

This transformation of the Salary Range column is a critical step in preparing the dataset for predictive analysis. By distilling the original 561 unique salary ranges into 4 well-defined categories, I have made the prediction task more feasible and focused. This structured approach allows the model to efficiently predict salary ranges based on job titles and other relevant features, aligning perfectly with the overall goal of the project.

A screenshot of a phone

Description automatically generated A screenshot of a computer

Description automatically generated

**Dropping Columns:**

After completing the necessary feature engineering and data transformation steps, I proceeded to streamline the dataset by dropping several columns that were no longer needed for the predictive modeling process. These columns were either redundant, contained information that had already been incorporated into new features, or were irrelevant to the prediction task. The specific columns dropped were as follows:

**Columns Dropped:**

1. **Job Id:**
   * As a unique identifier for each job posting, this column was not useful for the prediction model and was therefore removed.
2. **Latitude & Longitude:**
   * These columns provided geographic coordinates, but since the location's impact on salary was already captured through other features (such as job title and sector), these columns were dropped.
3. **Company Size:**
   * The company size data was considered during feature engineering, but its direct use was deemed unnecessary after deriving other related features.
4. **Job Posting Date:**
   * The date a job was posted does not directly influence the salary range prediction, so this column was excluded.
5. **Contact Person & Contact:**
   * These columns contained personal or administrative contact information, irrelevant to salary prediction, and were thus removed.
6. **Job Portal:**
   * The source or platform where the job was posted was not considered to have a significant impact on salary prediction, leading to its exclusion.
7. **Job Description & Responsibilities:**
   * While these text-based columns might have provided some insights, they were deemed too unstructured for direct use in the model, especially after the extraction of more meaningful features.
8. **Company Profile:**
   * Information from the company profile was used to extract the industry and sector, after which the original column was no longer needed.
9. **Salary Range:**
   * After transforming this column into the new target variable, it was removed from the dataset to avoid redundancy.
10. **Benefits:**
    * After categorizing and simplifying the benefits into the new Generalized Benefits column, the original Benefits column was dropped.
11. **Skills:**
    * Like the benefits, the Skills column was dropped after creating the Generalized Skills feature.
12. **Role, Specific Job Title, Specialized Role:**
    * These columns were dropped after creating more generalized versions that better suited the modeling process.
13. **Min Salary & Max Salary:**
    * These columns were used to create the new salary range target variable and were no longer needed after this transformation.

**Rationale for Dropping Columns:**

The primary goal of dropping these columns was to reduce the dataset's complexity and focus on the most relevant features for salary prediction. By removing redundant or irrelevant data, I ensured that the model would be trained on the most meaningful and impactful variables, thereby improving its efficiency and accuracy.

This careful pruning of the dataset also helps to avoid overfitting by eliminating noise and ensures that the model generalizes well to new data. The resulting dataset is now streamlined and ready for the modeling phase, with only the essential features retained to predict salary ranges effectively.

**Null values:**

After dropping the unnecessary columns, I performed another check for null values to ensure the dataset was fully prepared for modeling. During this process, I discovered that the Sector and Industry columns had 29,240 null values.

A screenshot of a phone

Description automatically generated

To address this issue and maintain the integrity of the data, I decided to fill these null values using the mode, which is the most frequently occurring value in each column.Top of Form

**Why Mode Was Used:**

* **Sector and Industry:**
  + The mode is a suitable method for filling null values in categorical columns like Sector and Industry because it represents the most common category within the dataset. This approach assumes that the missing values are more likely to belong to the most frequent category, thus minimizing the introduction of bias.
  + By filling these null values with the mode, I ensured that the dataset remained complete and consistent, without losing any significant information that could impact the predictive modeling process.

**Impact of Filling Null Values:**

* This step helped to ensure that the Sector and Industry columns were fully populated, allowing the model to utilize these features effectively without being hindered by missing data.
* Using the mode for imputation is particularly effective when the majority category is strongly representative of the dataset, which was the case here.

This step of handling null values further refined the dataset, ensuring that all features were complete and ready for use in the salary range prediction model. The dataset is now clean, structured, and fully prepared for the next phase of analysis and model training.

A screenshot of a phone

Description automatically generated

**Visualizations:**

**Bar Plots:**

**A screenshot of a computer

Description automatically generated**

**A bar graph with numbers and text

Description automatically generated**

**A bar graph with blue and white lines

Description automatically generated**

**A blue and white bar chart

Description automatically generatedA bar graph with numbers and text

Description automatically generated**

**A bar graph with blue squares

Description automatically generated**

**A bar graph with numbers and lines

Description automatically generated**

**A bar graph with numbers and a bar chart

Description automatically generated**

**A bar graph with blue and white bars

Description automatically generated**

**A bar graph with numbers and numbers

Description automatically generatedA graph of a bar graph

Description automatically generated with medium confidence**

**Outliers in the Dataset**

During the analysis, we identified some notable outliers in the dataset that had a significant impact on certain features:

1. **Skills:**
   * Outliers in 'Other': The 'Skills' feature had a substantial number of entries categorized as "Other," with a value count of 774,772. This large count indicates that a wide variety of skills were grouped under this general category, potentially obscuring more specific and relevant skills. This grouping as "Other" can be considered an outlier because it disproportionately represents a significant portion of the data, making challenging to draw precise insights from this feature.
   * A graph with lines and dots

     Description automatically generated
2. **Sector:**
   * Financial Services: In the 'Sector' feature, the Financial Services category showed a value count of 163,346, which is significantly higher than other sectors. This high concentration indicates that the dataset is heavily skewed toward this sector, which may influence the model's predictions more strongly than intended.
   * Energy Sector: Similarly, the Energy sector had a value count of 100,367. While not as extreme as Financial Services, this also represents a considerable portion of the data, which could disproportionately affect the analysis.

A blue rectangular object with a black line

Description automatically generated

**Implications:**

* These outliers in 'Skills' and 'Sector' suggest that the dataset is not evenly distributed, with certain categories dominating the data. This can lead to biased model outcomes where predictions might be overly influenced by these dominant categories.

**Considerations:**

* In future analyses, it might be beneficial to further investigate these outliers. For example, breaking down the "Other" category in Skills into more specific subcategories could provide clearer insights. Similarly, rebalancing the dataset or applying techniques to handle these outliers might lead to more accurate and generalizable models.

**Understanding the Correlation Matrix**

The correlation matrix displays the correlation coefficients between pairs of variables in your dataset. The correlation coefficient is a measure that indicates the extent to which two variables change together. It ranges from -1 to 1:

* **1** indicates a perfect positive correlation, meaning as one variable increases, the other increases.
* **-1** indicates a perfect negative correlation, meaning as one variable increases, the other decreases.
* **0** indicates no linear correlation between the variables.

**Interpretation of the Correlation Matrix**

1. **Experience min vs. Other Variables:**
   * **Experience min & Experience max:** The correlation is almost zero (-0.000970), suggesting that there is no linear relationship between minimum and maximum experience in this dataset.
   * **Experience min & Min Salary:** The correlation is very close to zero (0.001025), indicating no meaningful linear relationship between minimum experience and minimum salary.
   * **Experience min & Max Salary:** Similarly, there is almost no correlation (0.001677) between minimum experience and maximum salary.
   * **Experience min & Salary Range Count:** The correlation is extremely low (0.000101), again indicating no significant linear relationship.
2. **Experience max vs. Other Variables:**
   * **Experience max & Min Salary:** The correlation is close to zero (-0.000789), suggesting no linear relationship between maximum experience and minimum salary.
   * **Experience max & Max Salary:** The correlation is very weak (0.000304), showing almost no linear relationship between maximum experience and maximum salary.
   * **Experience max & Salary Range Count:** The correlation is negligible (-0.000300), indicating no significant linear relationship.
3. **Min Salary vs. Other Variables:**
   * **Min Salary & Max Salary:** There is a very weak positive correlation (0.000995), suggesting a very slight linear relationship between minimum and maximum salary.
   * **Min Salary & Salary Range Count:** The correlation is slightly negative (-0.001388), indicating no meaningful linear relationship.
4. **Max Salary vs. Salary Range Count:**
   * **Max Salary & Salary Range Count:** The correlation is very weak and negative (-0.001982), indicating no significant linear relationship between maximum salary and the count of salary ranges.
5. **Salary Range Count:**
   * **Correlations with Other Variables:** The salary range count has very low correlations with all other variables, meaning it does not have a meaningful linear relationship with any of them.

**Overall Summary:**

* **Lack of Strong Correlations:** The matrix shows that all the correlation coefficients between the variables are very close to zero. This indicates that there is no strong linear relationship between any of the pairs of variables in this dataset.
* **Independence of Variables:** The lack of significant correlations suggests that these variables are relatively independent of each other in a linear sense, meaning changes in one variable do not predictably lead to changes in another.
* **Implications:** In the context of predictive modeling, these low correlations imply that linear models might not capture strong relationships between these variables, and other forms of analysis might be needed to uncover more complex patterns or interactions.

A screenshot of a computer code

Description automatically generated

**Class Imbalance Consideration**

During the analysis, it was observed that the target variable, representing salary ranges, exhibited a significant class imbalance. The majority of the data is concentrated within the $59K – $104K salary range, while the other salary ranges have considerably fewer instances.

This imbalance is important to note because it can potentially bias the model towards the more frequent salary ranges, leading to less accurate predictions for the less common ranges. Although resampling techniques such as oversampling the minority classes or under sampling the majority class were not applied in this project, the impact of the imbalance was carefully considered during model evaluation.

In future work, addressing this class imbalance could be an important step. Techniques like resampling, or adjusting the class weights within the model, could be explored to ensure that the model performs well across all salary ranges, not just the most frequent ones.

A graph of a bar graph

Description automatically generated with medium confidence

**Modeling:**

**Modeling Process and Challenges Encountered**

After preparing the dataset, I proceeded with the modeling phase using several machine learning algorithms, specifically Decision Tree, Random Forest, and Gradient Boosting. The objective was to leverage these algorithms to predict salary ranges based on the key features identified during data exploration, such as experience, qualifications, and skills.

**Initial Modeling Attempts:**

* **Decision Tree:** I began with a Decision Tree model due to its simplicity and interpretability. The model quickly returned an R² value of 1.0, indicating perfect accuracy in predicting the salary ranges.
* **Random Forest:** Next, I employed Random Forest, an ensemble method known for its robustness and ability to handle overfitting better than single decision trees. Surprisingly, this model also returned an R² value of 1.0.
* **Gradient Boosting:** Finally, I implemented Gradient Boosting, another ensemble method that often excels in performance by combining weak learners into a strong predictive model. However, like the previous models, it too yielded an R² of 1.0.

**Interpreting the R² of 1.0:**

* **Understanding the Perfect Score:** While an R² of 1.0 might initially appear to be an ideal outcome, it quickly raised red flags. An R² of 1.0 means that the model explains 100% of the variance in the target variable on the training data. In practical terms, this suggests that the model is making perfect predictions for every single instance in the training dataset.
* **Concerns About Overfitting:** The perfect R² score across all models strongly indicated that they were overfitting. Overfitting occurs when a model learns not just the underlying patterns in the data but also the noise and specific details that don’t generalize to new, unseen data. This leads to a model that performs exceptionally well on the training data but poorly on validation or test data.
* **Potential Causes of Overfitting:** Several factors could contribute to this overfitting:
  + **Complexity of the Models:** Decision trees, especially deep ones, can easily overfit by capturing too many details of the training data. Ensemble methods like Random Forest and Gradient Boosting, while more robust, can still overfit if not properly regularized or if the data itself has issues.
  + **Data Issues:** The presence of highly correlated features, outliers, or the class imbalance in the target variable could also lead to overfitting, as the models may latch onto these patterns too strongly.

A screenshot of a computer

Description automatically generatedBottom of Form

**Modeling Step Overview**

As part of the modeling process, the data was carefully prepared and transformed to ensure that the machine learning models could be trained effectively. Below is a detailed explanation of each step in the modeling process based on the provided configuration:

**Session Setup:**

* **Session ID (3098):** A unique identifier was assigned to this session, ensuring that the results and models could be easily tracked and reproduced.

**Target Variable:**

* **Target (New Salary Range):** The target variable for this modeling exercise was the New Salary Range, which represented the salary bands we aimed to predict.

**Target Type:**

* **Multiclass:** The New Salary Range was treated as a multiclass classification problem, where the goal was to classify the data into one of several discrete salary ranges.

**Target Mapping:**

* **Mapping Salary Ranges:** The salary ranges were mapped as follows:
  + $59K-$104K: 0
  + $59K-$105K: 1
  + $60K-$104K: 2
  + $60K-$105K: 3 This mapping allowed the model to distinguish between the different salary ranges and predict the appropriate category for each instance.

**Data Shape and Transformation:**

* **Original Data Shape (51899, 10):** The original dataset consisted of 51,899 observations and 10 features, which included both the predictors and the target variable.
* **Transformed Data Shape (51899, 34):** After preprocessing, the data was transformed into a shape of 51,899 observations and 34 features. This increase in the number of features was primarily due to the one-hot encoding of categorical variables, which expanded the dataset.
* **Train/Test Split:**
  + **Transformed Train Set Shape (36329, 34):** The training set comprised 36,329 observations with 34 features.
  + **Transformed Test Set Shape (15570, 34):** The test set consisted of 15,570 observations, also with 34 features. This split allowed for the model to be trained on the majority of the data while reserving a portion for validation and testing.

**Feature Types:**

* **Numeric Features (2):** There were 2 numeric features in the dataset after transformation. These features were standardized through normalization to ensure they contributed effectively to the model.
* **Categorical Features (7):** There were 7 categorical features that were one-hot encoded, resulting in multiple binary columns corresponding to each category. This process was essential for incorporating categorical data into the models.

**Data Preprocessing:**

* **Preprocessing (True):** Preprocessing was applied to ensure the data was clean, consistent, and ready for modeling. This included handling missing values, encoding categorical variables, and normalizing numeric data.
* **Imputation Type (Simple):** Simple imputation methods were used to handle missing values.
  + **Numeric Imputation (Mean):** Missing numeric values were imputed using the mean of the respective feature, ensuring that missing data did not skew the results.
  + **Categorical Imputation (Mode):** Missing categorical values were imputed using the mode (most frequent value), maintaining the integrity of the categorical data.

**Encoding and Normalization:**

* **Maximum One-Hot Encoding (25):** Categorical variables with more than 25 unique categories were one-hot encoded up to 25 categories to prevent an explosion in the number of features.
* **Encoding Method (None):** No additional encoding was applied beyond the one-hot encoding of categorical variables.
* **Normalization (True):** Normalization was applied to scale numeric features, ensuring they had a mean of 0 and a standard deviation of 1.
* **Normalization Method (zscore):** The z-score method was used for normalization, which standardizes features by subtracting the mean and dividing by the standard deviation.

**Cross-Validation Setup:**

* **Fold Generator (StratifiedKFold):** Stratified K-Fold cross-validation was used to ensure that each fold had a representative distribution of the target variable. This method helped to maintain the balance of the classes across training and validation sets.
* **Fold Number (10):** A 10-fold cross-validation approach was employed, where the dataset was divided into 10 parts, and the model was trained and validated 10 times, each time using a different part of the data for validation.

**Computational Resources:**

* **CPU Jobs (-1):** All available CPU cores were utilized during the training process, which optimized the computational efficiency.
* **Use GPU (False):** GPU acceleration was not used for this session, as the CPU was sufficient for handling the computations.

**Experiment Logging:**

* **Log Experiment (False):** Experiment logging was not enabled, meaning that the session’s details were not logged for future reference in an external system.
* **Experiment Name (clf-default-name):** The default name was assigned to the experiment, indicating that no specific name was given to this particular run.

**Unique Session Identifier:**

* **USI (d9d8):** A unique session identifier was generated, which helps in tracking and managing the session's results and configurations.

A screenshot of a computer

Description automatically generated

**Model Comparison and Performance Metrics**

The table summarizes the performance of several machine learning models that were trained and evaluated on the dataset. Here’s what each column and model indicates:

**Models:**

* **LightGBM (Light Gradient Boosting Machine):** A highly efficient and effective gradient boosting model optimized for speed and performance.
* **XGBoost (Extreme Gradient Boosting):** Another popular gradient boosting algorithm known for its robustness and accuracy.
* **GBC (Gradient Boosting Classifier):** A more traditional implementation of gradient boosting, often used for its flexibility in handling various types of data.
* **DT (Decision Tree Classifier):** A simple and interpretable model that splits data based on feature values.
* **RF (Random Forest Classifier):** An ensemble method that builds multiple decision trees and averages their predictions to improve accuracy and control overfitting.
* **ET (Extra Trees Classifier):** Similar to Random Forest, but uses random splits for decision trees, adding more randomness to the model.
* **AdaBoost (AdaBoost Classifier):** An adaptive boosting algorithm that combines weak learners into a strong model.
* **NB (Naive Bayes):** A probabilistic classifier based on Bayes’ theorem, assuming independence between features.
* **LR (Logistic Regression):** A linear model that estimates probabilities for binary classification tasks.
* **LDA (Linear Discriminant Analysis):** A technique that models the difference between classes using linear combinations of features.
* **SVM (Support Vector Machine - Linear Kernel):** A linear classifier that tries to maximize the margin between classes.
* **Ridge (Ridge Classifier):** A regularized version of linear regression that can reduce model complexity.
* **KNN (K Neighbors Classifier):** A non-parametric method that predicts the label based on the majority vote of the nearest neighbors.
* **QDA (Quadratic Discriminant Analysis):** Similar to LDA but allows for a quadratic decision boundary.
* **Dummy Classifier:** A baseline model that makes predictions without considering the input features, often used to set a lower bound on model performance.

**Performance Metrics:**

* **Accuracy:** The ratio of correctly predicted instances to the total instances. It measures how often the model makes the correct prediction overall.
  + **Best Performers:** LightGBM (82.05%), XGBoost (81.69%), and GBC (80.86%) had the highest accuracy, indicating that these models correctly predicted the salary range most of the time.
  + **Lowest Accuracy:** The Dummy Classifier had the lowest accuracy (42.18%), serving as a baseline for comparison.
* **AUC (Area Under the Curve):** Represents the area under the Receiver Operating Characteristic (ROC) curve, which plots true positive rate against false positive rate.
  + **Best AUC:** LightGBM (0.9539) and XGBoost (0.9522) had the highest AUC values, suggesting that they were very effective at distinguishing between different classes.
  + **Zero AUC:** Some models like GBC and others have an AUC of 0.0000, which likely indicates that the AUC was not calculated or available for these models.
* **Recall:** The ratio of true positives to the sum of true positives and false negatives. It indicates how well the model identifies all positive instances.
  + **Best Recall:** LightGBM (82.05%) and XGBoost (81.69%) excelled in recall, meaning they were good at identifying the relevant salary ranges.
  + **Lowest Recall:** QDA and Dummy Classifier had lower recall rates, indicating poor performance in identifying the correct salary ranges.
* **Precision (Prec.):** The ratio of true positives to the sum of true positives and false positives. It measures the accuracy of positive predictions.
  + **Best Precision:** LightGBM (82.48%) and XGBoost (82.17%) also performed well in precision, meaning that when they predicted a certain salary range, it was likely correct.
  + **Lowest Precision:** The Dummy Classifier had very low precision (17.79%), indicating that it frequently made incorrect positive predictions.
* **F1 Score (F1):** The harmonic mean of precision and recall. It provides a balance between the two, especially when you need to consider both false positives and false negatives.
  + **Best F1 Scores:** LightGBM (81.95%) and XGBoost (81.58%) showed the best balance between precision and recall, making them reliable choices.
  + **Lowest F1 Score:** The Dummy Classifier had a very low F1 score (25.03%), confirming its poor overall performance.
* **Kappa:** Cohen’s Kappa measures the agreement between the predicted and actual classifications, accounting for chance agreement.
  + **Best Kappa:** LightGBM (0.7327) and XGBoost (0.7270) had high Kappa values, suggesting strong agreement between their predictions and actual outcomes.
  + **Lowest Kappa:** The Dummy Classifier had a Kappa of 0.0000, indicating no agreement beyond chance.
* **MCC (Matthews Correlation Coefficient):** A measure that takes into account all four confusion matrix categories (true positives, false positives, true negatives, and false negatives). It’s a balanced metric that can be used even if the classes are of different sizes.
  + **Best MCC:** LightGBM (0.7347) and XGBoost (0.7292) had the highest MCC values, indicating a strong correlation between their predictions and actual outcomes.
  + **Lowest MCC:** The Dummy Classifier had an MCC of 0.0000, showing no meaningful correlation between its predictions and the actual data.
* **TT (Sec):** The training time in seconds indicates how long it took to train each model.
  + **Fastest Models:** The Dummy Classifier (0.9010 seconds) and Logistic Regression (1.2950 seconds) were among the fastest to train, but their performance was also lower.
  + **Slower Models:** KNN took significantly longer (2.4710 seconds), while LightGBM (13.1090 seconds) and GBC (16.9570 seconds) also took longer but provided higher accuracy and performance.

A screenshot of a data

Description automatically generated

**Overall Insights:**

* **Top Performers:** LightGBM and XGBoost emerged as the top-performing models across almost all metrics. They delivered high accuracy, strong AUC, and well-balanced precision and recall, making them robust choices for predicting salary ranges.
* **Trade-offs:** While models like LightGBM and GBC took longer to train, their superior performance justified the additional time. In contrast, simpler models like Decision Trees and Random Forests offered faster training times but slightly lower performance.
* **Baseline Comparison:** The Dummy Classifier, serving as a baseline, performed poorly across all metrics, reaffirming the effectiveness of the other models.

A screenshot of a computer

Description automatically generated

This table displays the results of cross-validation for the Light Gradient Boosting Machine (LightGBM) model, which is an ensemble method that combines multiple decision trees to make predictions. The metrics are calculated across 10 different folds of the dataset, allowing for a robust evaluation of the model’s performance. Let’s break down what each part of the table represents:

**Cross-Validation Results:**

1. **Folds (0-9):**
   * The dataset was split into 10 folds (0 through 9) for cross-validation. In each fold, the model was trained on 9 parts of the data and tested on the remaining part. This process was repeated 10 times, each time with a different part as the test set.
2. **Metrics Across Folds:**
   * **Accuracy:** The proportion of correctly classified instances out of all instances. Accuracy for each fold varies slightly, with the values ranging from 81.25% to 83.15%. The average accuracy across all folds is **82.05%**.
   * **AUC (Area Under the Curve):** Measures the model’s ability to distinguish between classes. AUC values are consistently high across all folds, with a mean AUC of **0.9539**. This suggests that the LightGBM model has a strong capability to differentiate between salary ranges.
   * **Recall:** The ratio of true positive predictions to the total actual positives. Recall is also consistent across folds, with an average of **82.05%**, indicating the model's reliability in identifying positive instances.
   * **Precision (Prec.):** The ratio of true positive predictions to the total positive predictions. The model's precision is strong across all folds, with a mean value of **82.48%**. This means that when the model predicts a certain salary range, it is correct about 82.48% of the time.
   * **F1 Score (F1):** The harmonic mean of precision and recall, balancing the two metrics. The F1 score averages at **81.95%**, showing that the model maintains a good balance between precision and recall.
   * **Kappa:** Cohen’s Kappa measures the agreement between the predicted and actual classifications, accounting for chance. The average Kappa score is **0.7327**, indicating a substantial agreement between the model’s predictions and the actual data.
   * **MCC (Matthews Correlation Coefficient):** A measure of the correlation between predicted and actual classifications, with an average of **0.7347**. This high value suggests that the model's predictions are well-correlated with the true labels.
3. **Mean and Standard Deviation (Std):**
   * **Mean:** The average value of each metric across all 10 folds. The mean values highlighted in yellow provide a summary of the model's overall performance:
     + **Accuracy:** 82.05%
     + **AUC:** 0.9539
     + **Recall:** 82.05%
     + **Precision:** 82.48%
     + **F1 Score:** 81.95%
     + **Kappa:** 0.7327
     + **MCC:** 0.7347
   * **Std (Standard Deviation):** The standard deviation measures the variation in the metric across the folds. For example, the standard deviation for accuracy is 0.0068, which indicates that the model's accuracy is fairly consistent across the different folds. The low standard deviations for all metrics suggest that the model's performance is stable and reliable.

**Summary of Insights:**

* **Consistent Performance:** The LightGBM model performs consistently across all 10 folds, with only slight variations in each metric. This consistency is reflected in the low standard deviations.
* **Strong Predictive Power:** The high AUC, Precision, Recall, F1 Score, Kappa, and MCC values indicate that the LightGBM model is very effective at predicting salary ranges, with strong generalization across the different folds of the dataset.
* **Reliability:** The model shows a high level of agreement between its predictions and the actual data, making it a reliable choice for predicting salary ranges in this context.

In conclusion, the cross-validation results suggest that the LightGBM model is well-suited for this task, offering strong, consistent performance across a range of important evaluation metrics.

**Conclusion and Recommendations**

**Impacts on Business Problem (Scope of the Recommended Model)**

The predictive model developed in this project offers significant value to both individuals and organizations by providing a data-driven approach to understanding salary trends. By accurately estimating salary ranges based on factors such as qualifications, experience, job title, role, sector, industry, and type of work, the model empowers job seekers to negotiate competitive salaries and enables employers to set equitable and market-aligned compensation packages.

The scope of the recommended model extends to various business applications:

* **Human Resources:** HR departments can use the model to benchmark salaries, ensuring that their compensation offers are competitive and fair.
* **Talent Acquisition:** Recruiters can leverage the model to align job offers with industry standards, improving their ability to attract top talent.
* **Career Development:** Employees and career counselors can use the insights from the model to make informed decisions about career paths and required qualifications or experience.
* **Market Analysis:** Organizations can analyze salary trends within specific sectors or industries, allowing them to make strategic decisions about entering new markets or expanding in existing ones.

Overall, the model addresses a critical business problem by offering a systematic way to analyze and predict salary outcomes, thereby supporting informed decision-making across multiple facets of an organization.

**Recommended Next Steps**

To further enhance the model’s applicability and accuracy, the following next steps are recommended:

1. **Expand Dataset:** Incorporate additional data sources to cover a broader range of industries, sectors, and geographic locations. This will improve the model’s generalizability and make it applicable across different contexts.
2. **Refine Feature Selection:** Explore the addition of new features, such as company size, education level, and geographic cost of living, which could provide deeper insights into salary predictions.
3. **Address Class Imbalance:** Implement techniques such as oversampling, under sampling, or class weighting to address the class imbalance in the target variable, ensuring that the model performs well across all salary ranges.
4. **Model Optimization:** Experiment with advanced machine learning algorithms, hyperparameter tuning, and ensemble methods to further improve the model’s accuracy and reduce prediction errors.
5. **Deploy the Model:** Develop a user-friendly interface or integrate the model into existing HR software to make it accessible to non-technical users within organizations.
6. **Continuous Monitoring:** Establish a process for continuously monitoring and updating the model with new data to maintain its relevance and accuracy over time.

By following these next steps, the model can be refined and expanded to provide even greater value to users, making it a vital tool for salary analysis and prediction.

**References**

* **Data Sources:**
  + [Provide the dataset sources, such as job boards, company databases, or publicly available salary surveys.]
* **Algorithms and Models:**
  + Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics.
  + Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794).
* **Feature Engineering:**
  + Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling*. Springer.
  + Brownlee, J. (2016). *Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python*. Machine Learning Mastery.
* **Model Evaluation Metrics:**
  + Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427-437.
* **Cross-Validation Techniques:**
  + Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (Vol. 2, pp. 1137-1143).
* **Ethics and Bias in Machine Learning:**
  + Barocas, S., Hardt, M., & Narayanan, A. (2019). *Fairness and Machine Learning: Limitations and Opportunities*. fairmlbook.org.

Bottom of Form