Salary Prediction

Mattia Fedeli

Automation Engineering

Student ID: 0001148008

Optimization & Machine Learning M Exam

June 23, 2025

Contents

List of Figures

List of Tables

0 Introduction

The goal of this project is to perform Exploratory Data Analysis on a salary dataset and to train a machine learning model to predict the salary based on some relevant features.

This report follows step-by-step the Python notebook structure, so code snippets will be omitted to avoid repetition. The focus of this report is the rationale behind each operation and the results obtained.

Before starting, the virtual environment has been set up according to the specific library versions reported at the beginning of the notebook (to repeat the notebook execution).

Some auxiliary functions have initially been defined to print tables in a comfortable way.

1 Data Exploration and Preprocessing

1.1 Loading the dataset

The first step of the project consists of importing the dataset from the given .csv file. Exploring the information regarding the loaded pandas dataframe, it can be seen how the file contains a total of **6704 rows** (with some missing values that will be addressed later) and 6 features:

- **Age** (numerical);
- **Gender** (categorical);
- **Education Level** (categorical);
- **Job Title** (categorical);
- **Years of Experience** (numerical);
- **Salary** (numerical);

The following are the first 5 rows of the dataframe:

Table 1: Imported Dataset

Age	Gender	Education Level	Job Title	Years of Experience	Salary
32	Male	Bachelor’s	Software Engineer	5	90000
28	Female	Master’s	Data Analyst	3	65000
45	Male	PhD	Senior Manager	15	150000
36	Female	Bachelor’s	Sales Associate	7	60000
52	Male	Master’s	Director	20	200000

1.2 Handling missing values and duplicates

By searching for **missing values**, 2 rows appear completely empty, and other 4 rows present some empty spaces. The empty holes could be filled with median or average values, but it is decided to remove them for simplicity. The total samples are now **6698**.

When searching for duplicates, most of the data samples appear to be present more than once. In particular, only **1787** rows appear just once, while the remaining **4911** rows are duplicated. The following are the 5 most appearing rows:

Table 2: Most duplicated rows

Age	Gender	Education Level	Job Title	Years of Experience	Salary	Repetitions
24	Female	High School	Receptionist	0	25000	45
32	Male	Bachelor's Degree	Product Manager	7	120000	45
27	Male	Bachelor's Degree	Software Engineer	3	80000	45
32	Male	Bachelor's	Software Engineer	8	190000	39
33	Female	Master's	Product Manager	11	198000	38

Although some rows could reasonably appear more than once, since not many features are present in the dataset, the probability of having 45 product managers with the same exact age and experience seems very low. Since the data source is unknown, and to prevent overfitting, **all duplicated rows will be dropped**.

1.3 Visualizing the distribution of the variables

Initially, the histograms of the numerical values in the dataframe are plotted (Fig. 1).

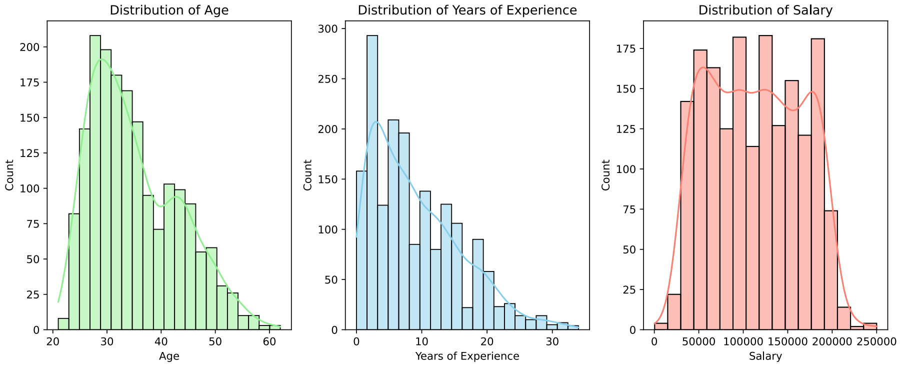

Figure 1: Histograms of the numerical features of the dataset

The first 2 subplots have a similar distribution, skewed towards lower values of age and experience. The salary subplot instead is more uniform with some very high and very low values.

In the following table, a statistical description of the numerical data is performed.

Table 3: Descriptive Statistics of the numerical features

Statistic	Age	Years of Experience	Salary
count	1787.00	1787.00	1787.00
mean	35.14	9.16	113184.66
std	8.21	6.84	51596.54
min	21.00	0.00	350.00
25%	29.00	3.00	70000.00
50%	33.00	8.00	110000.00
75%	41.00	13.00	160000.00
max	62.00	34.00	250000.00
median	33.00	8.00	110000.00
mode	29.00	2.00	120000.00

From the table, it can be noted that there are some values that could be considered outliers with high number of years of experience and for high salaries. Since the source of the dataset is unknown, this values in this case can be considered reasonable.

However, the low value of the minimum salary is wrong and is not useful. By printing the salaries by ascending order, one obtains the following data:

Table 4: Top 10 Lowest Salaries

Age	Gender	Education Level	Job Title	Years of Experience	Salary
29	Male	Bachelor's	Junior Business Operations Analyst	1.5	350
31	Female	Bachelor's Degree	Junior HR Coordinator	4	500
25	Female	Bachelor's Degree	Front end Developer	1	550
23	Male	PhD	Software Engineer Manager	1	579
22	Female	High School	Sales Associate	0	25000
22	Female	High School	Receptionist	0	25000
25	Female	High School	Sales Associate	0	25000
24	Female	High School	Sales Associate	0	25000
30	Female	High School	Junior Sales Associate	1	25000
33	Male	High School	Junior Sales Associate	1	25000

The first 4 samples are certainly outliers, so they are eliminated.

Next, in the following figure (Fig. 2) the salaries with respect to age and years of experience are plotted.

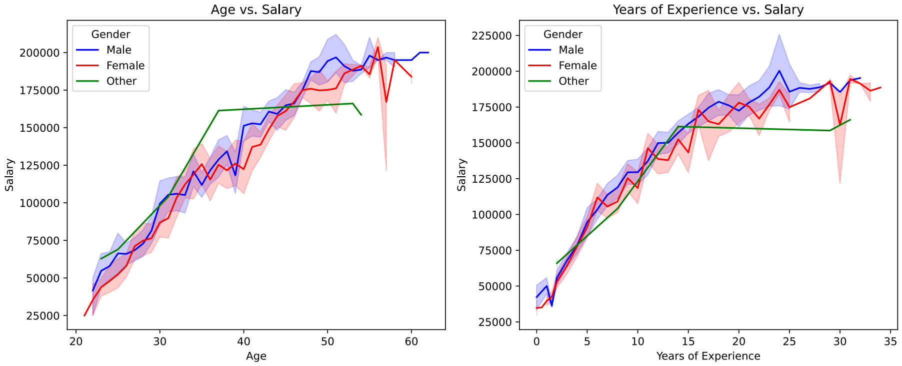

Figure 2: Line plots of the salaries with respect to age (left) and years of experience (right) for all the genders

There appears to be a strong correlation between both variables and the salaries, so the correlation matrix is computed (Fig. 3).

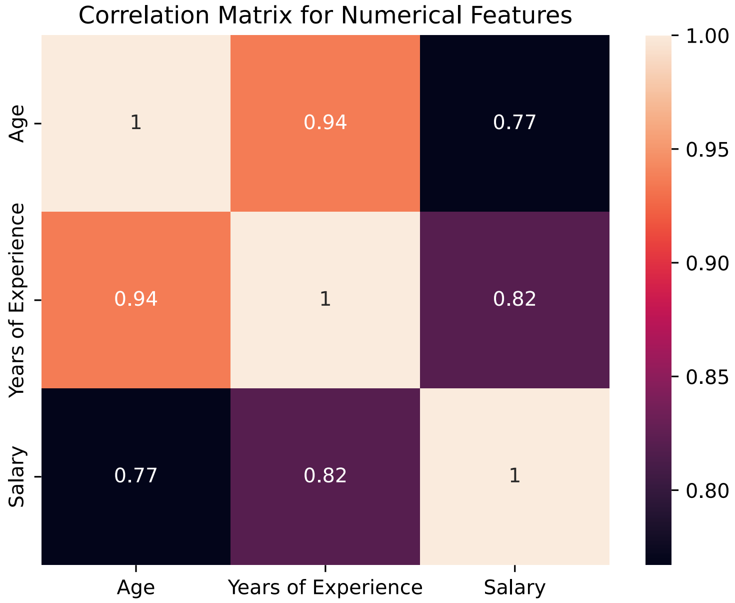

Figure 3: Correlation Matrix

Obviously, there is a very high correlation between age and experience (trivial), and also a high correlation between both quantities and the salary.

2 Features Selection and Engineering

2.1 Identifying relevant features

All the features in the dataframe can be considered as very relevant to predict the salary, in particular, each one is analyzed in detail in the following paragraphs.

2.1.1 Age and Years of Experience

These two features are strongly correlated, as previously observed. However, since there is no strict one-to-one relationship between them, both will be considered in the final dataset to preserve potentially distinct information.

2.1.2 Gender

For the gender feature, there are 3 possible values:

- **Male**: 964 samples (54.07%);
- **Female**: 812 samples (45.54%);
- **Other**: 7 samples (0.39 %);

From an ethical point of view, all minorities and diversities should be included and considered. However, in this didactic context, in order to have better results, the ”Other” value can be neglected, since it has very low representation in the dataset.

For the other 2 values, the following table shows the differences in salaries between Male and Female classes.

Table 5: Salary stats by Gender

Statistic	Female	Male
count	812.00	964.00
mean	107557.78	118300.16
std	50547.29	51629.89
min	25000.00	25000.00
25%	60750.00	75000.00
50%	105000.00	120000.00
75%	150000.00	165000.00
max	220000.00	250000.00
median	105000.00	120000.00
mode	120000.00	120000.00

It can be seen that male salaries are significantly higher in the mean value, the median value, and in the maximum value. Of course, unfortunately, this feature is relevant for the prediction of the salary.

2.1.3 Education Level

For what concerns the education level, the following are the unique strings present in the dataset:

- **Bachelor’s Degree**: 504 samples (28.27 %);
- **Master’s Degree**: 446 samples (25.01%);
- **PhD**: 339 samples (19.01%);
- **Bachelor’s**: 261 samples (14.64%);
- **Master’s**: 122 samples (6.84%);
- **High School**: 110 samples (6.17%);
- **phD**: 1 sample (0.06%);

In this case, the features can be merged in just four classes, since, for example, ”Bachelor’s Degree” and ”Bachelor’s” represent the same thing.

This feature is very relevant, as reported in the following graph, where the relation between higher salaries and higher education level can be observed (Fig. 4).

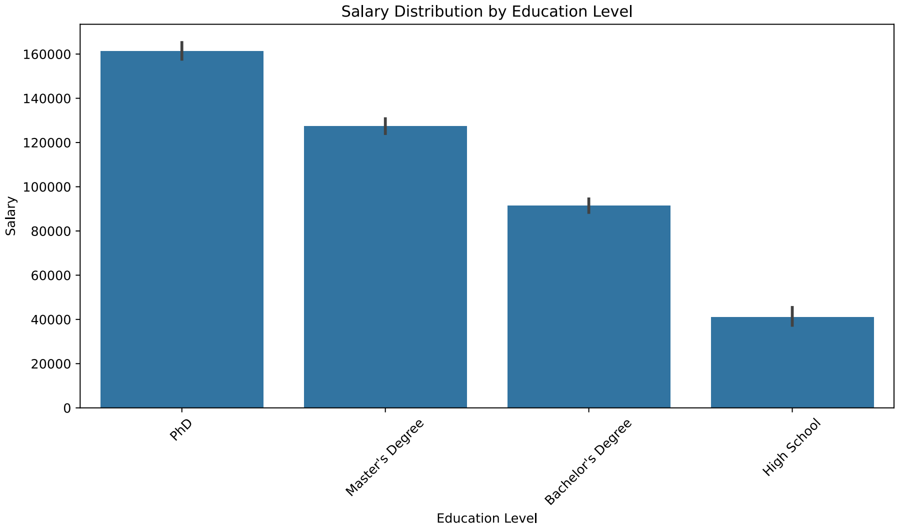

Figure 4: Relation between education and Salary

2.1.4 Job Titles

By searching for unique job titles (lowercase strings to avoid duplicates), there are a total of **190 unique jobs**. In the following table, the 10 most frequently appearing job titles are reported.

Table 6: Top 10 Most Frequent Job Titles

Job Title	Frequency
software engineer manager	126
full stack engineer	120
senior software engineer	94
senior project engineer	94
back end developer	80
data scientist	80
software engineer	78
front end developer	73
marketing manager	55
product manager	53

There are many different job titles, which are categorical features, making them more challenging to use directly for training a model. However, job titles remain relevant for predicting the salary, as shown in the following plot (Fig. 5).

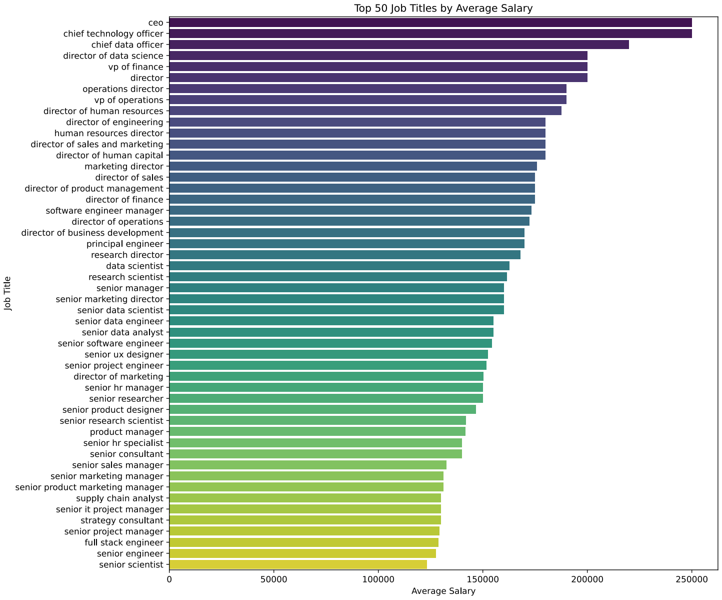

Figure 5: Average salary of the first 50 job types in descending order

2.2 Feature Engineering

The only feature that still needs to be addressed is the job titles. Its relevance was addressed earlier, now it is necessary to make it utilizable.

2.2.1 Seniority

The first step is to extract any reference to ”Senior” or ”Junior” (including misspellings like ”Juniour”) from the job titles and add it as a new column in the dataframe. There are **363 Seniors** and **176 Juniors** (plus 4 wrongly spelled as ”Juniours”). If a job title contains neither of the two, it is labeled as None. Seniority is clearly relevant, as senior employees are expected to have more experience than juniors for the same profession.

2.2.2 Management

Looking at Fig. 5, the highest paying jobs are management positions, which include terms such as ”Director”, ”Chief”, ”Manager”, ”Principal”, and similar. A list of similar keywords was created and used to identify whether a job title corresponds to a management position, resulting in a new binary feature.

This process is somewhat manual but effective for this simple study-case. In a more general setting, an unsupervised model could be exploited to automatically identify higher-profile positions. For this project, however, it is sufficient to check the presence of some keywords in the title. There are **615** individuals in this dataset who have management positions.

2.2.3 Aggregation by Similar Words

At this point, there are still 115 unique job titles, which is a very large number of classes for this feature. Encoding them as a one-hot vector is still unpractical due to the dimensions.

The next idea is to use an unsupervised algorithm to cluster the remaining job titles. Although it was reasonable to manually group management positions by extracting specific terms, grouping the rest by similarity is more challenging. Therefore, the **KMeans clustering algorithm** is applied after mapping the job titles into a vector space using a **TfidfVectorizer**.

This approach is more of a black box and less interpretable than the previous manual step. Although more complex models could be exploited, the TfidfVectorizer is efficient and effective. It transforms each job title into a high-dimensional vector, where each dimension corresponds to a unique word in the job list.

The optimal number of KMeans clusters is evaluated using the Silhouette Score, testing between 2 and 30 clusters, as shown in Fig. 6.

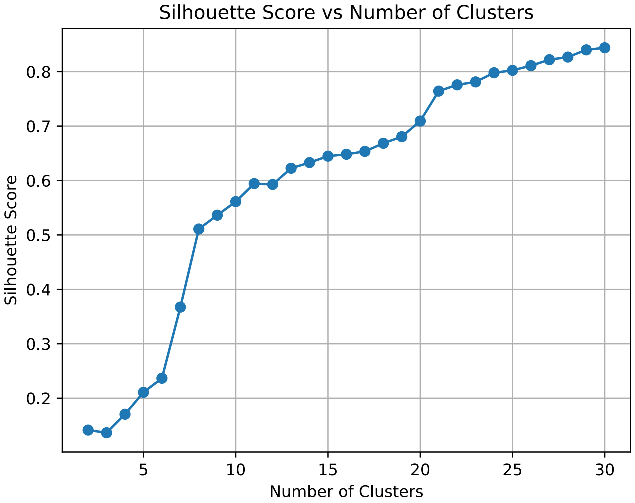

Figure 6: Evaluating clustering quality

It can be observed how the score keeps increasing as the number of clusters increases, but to keep the number of features small, it seems reasonable to choose only **8 clusters**, where a big leap takes place with a satisfying 0.5 score in the Silhouette metric.

In the following list, at most 3 samples per cluster are printed to understand how the algorithm performed.

- **Cluster 0**: 120 individuals
  - full stack engineer
- **Cluster 1**: 209 individuals
  - marketing analyst
  - marketing coordinator
  - marketing
- **Cluster 2**: 118 individuals
  - project
  - project engineer
  - project coordinator
- **Cluster 3**: 136 individuals
  - sales associate
  - sales

  - sales representative

- **Cluster 4**: 310 individuals
  - software engineer
  - engineer
  - software
- **Cluster 5**: 624 individuals
  - manager
  - director
  - software developer
- **Cluster 6**: 166 individuals
  - data analyst
  - scientist
  - data scientist
- **Cluster 7**: 93 individuals
  - product
  - product designer
  - product marketing

The clustering seems mostly reasonable. However, the software developer role in cluster 5 might fit better in cluster 4. Cluster 5 seems to act as an “other” group, containing jobs that are less similar to each other, which could explain its wider distribution.

2.2.4 Final Features Mapping

Once all columns are ready, the ”Job Title” column is dropped, and the categorical columns are mapped to numbers to make them compatible with scikit-learn models.

For example, the ”Education level” is mapped from ”High school” to ”PhD” as values from 0 to 3.

The following table shows the first 5 samples of the final dataset.

Table 7: Final dataset

Age	Gender	Education Level	Years of Experience	Seniority	Is Management	Job Clustered	Salary
32	0	1	5	0	0	4	90000
28	1	2	3	0	0	6	65000
45	0	3	15	2	1	5	150000
36	1	1	7	0	0	3	60000
52	0	2	20	0	1	5	200000

3 Model Development and Evaluation

3.1 Train test split

The dataset is split into a train set and a test set, with an 80-20% split, which is the ”standard”, with a fixed seed for repeatability.

3.2 Model Selection

This is a regression problem, so many different types of models can be used. After obtaining unsatisfactory results with linear regressors, regularized linear regressors, and decision tree regressors, it is chosen to use a **random forest regressor**. The results of the other models are not reported, only the best one.

The random forest regressor works by creating multiple decision trees, each trained on a random subset of the data, and averaging their predictions to produce a continuous output.

3.3 Model Training

Initially, the model is trained with the train set, with the standard procedure, as reported in the notebook. The performance of the model on the same dataset is evaluated using the three following metrics.

The R^2 **Score** measures how well a regression model explains the variance of the prediction variable. Its value is contained in the interval $(-\infty, 1]$, and it is obtained as

$$R^2 = 1 - \frac{SSres}{SStot}$$

where $SSres$ is the sum of the squares of the residuals (prediction errors at each sample), while $SStot$ is the variance from the mean. The closer it is to 1, the better the regression model fits the dataset.

The **Root Mean Squared Error (RMSE)** is a commonly used metric to evaluate the performance of a regression model. It represents the square root of the average squared differences between predicted and real values. It is obtained as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

where y_i is the true value, $\hat{y}_i$ is the predicted value, and n is the number of samples. Lower RMSE values indicate a better fit, and since RMSE is in the same unit as the target variable, it is easily interpretable.

The **Mean Absolute Error (MAE)** measures the average norm of the errors in a set of predictions, without considering their direction. It is computed as

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

where y_i and $\hat{y}_i$ are the true and predicted values, respectively, and n is the number of samples. MAE is more robust to outliers than RMSE and provides a clear interpretation of the average prediction error in the same unit as the target.

The results of the model on the training set are the following:

- R^2 **Score**: 0.97
- **RMSE**: 8089.99
- **MAE**: 5119.92

The following figure (Fig. 7) shows a graphic interpretation of the training results.

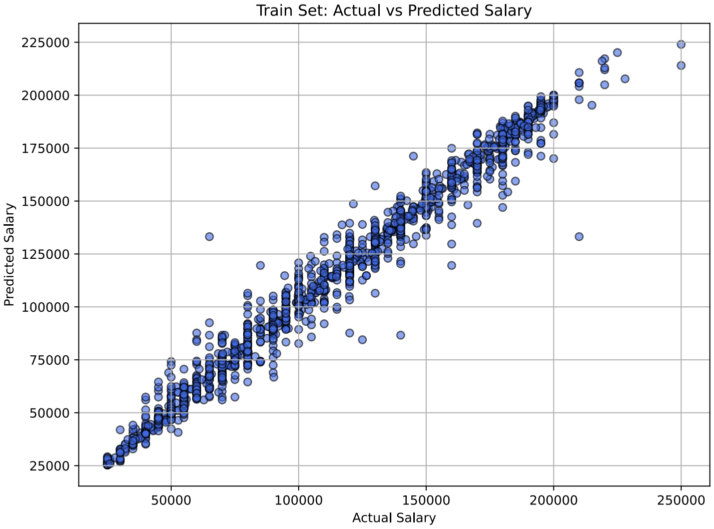

Figure 7: Evaluation of the training performance

3.4 Hyperparameters Fine Tuning

The model shows a very satisfying performance on the training set with a high R^2 score and low errors. However, this may also indicate overfitting, and further improvements on the training metrics could reduce the model's ability to generalize. To prevent this, the model complexity should not be increased. It could be useful to investigate in reducing the complexity without decreasing the performance (such as decreasing the number of estimators or limiting the max tree depth), but for this example the standard version is considered valid.

4 Prediction and Interpretation

4.1 Test Set Predictions

After having chosen and trained the model, its performance needs to be evaluated. The remaining part of the dataset was never seen during training, so this step is crucial for assessing the model's generalization ability. If properly trained, the model should achieve a performance on the test set that is comparable to that obtained on the training set.

In this case, the final model was evaluated on the test set using the same metrics used for the training set for comparing the two phases. The following are the results:

- R^2 **Score**: 0.89
- **RMSE**: 17582.63
- **MAE**: 11435.85

The results confirm that the model maintains satisfactory predictive accuracy. The reduced performance compared to the training set may be due to a slight overfitting, and the gap between the two could be reduced by lowering the model complexity.

4.2 Prediction Comparison

The following figure (Fig. 8) visually shows the predictive capabilities of the model, by comparing the real salary data of the test set with the inference predictions of the model.

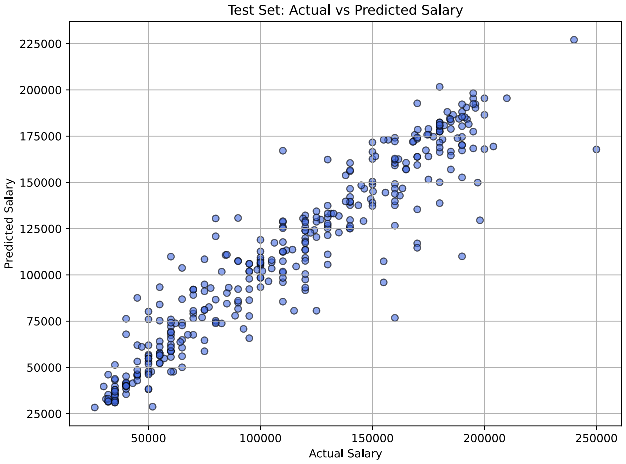

Figure 8: Evaluation of the test performance

In this case, unlike the previous one (Fig. 7), the predictions show much more variance. There are also some notable outliers, such as the point on the far right with the highest actual salary, which is predicted quite poorly. However, the overall relationship remains mostly linear, as expected, with a satisfactory R^2 score of nearly 90

4.3 Results interpretation

In general, the model has been able to identify relationships between the features and the target. In particular, according to the following figure (Fig. 9), years of experience is the most important feature, consistently with the previous correlation matrix (Fig. 3)).

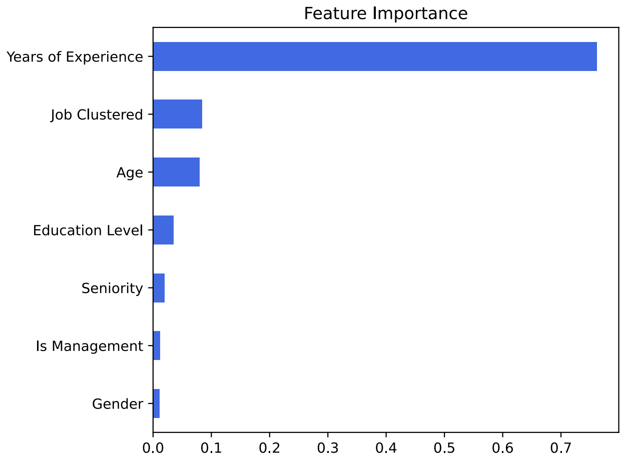

Figure 9: Percentage of importance of each feature on the prediction

The job area and age features (which are strongly related to experience) also play a role, though not as crucial as experience itself. All the other features, like education level and gender, seem to have very little impact on the final prediction of this model.

5 Conclusions

This project aimed at developing a regression model capable of predicting annual salaries starting from a messy dataset. The main focus was on cleaning and refining the features to obtain usable data. The dataset initially contained various types of features, many duplicates, missing values, and other kinds of annoying issues.

The chosen model is simple, general, and performs well. Although more complex than basic regressors, it delivers good results, trading off some interpretability for better accuracy.

The key insight from the data is the strong dependence of salary on years of experience, as obtained from both the correlation matrix and from the model feature importance.

For future developments, it is clear that more data is needed, ideally already cleaned. Additionally, the model could be further improved by performing a proper hyperparameter tuning, for example through a grid search, carefully aimed at maximizing performance while also avoiding overfitting.