CSCI 183: Final Project

Project Title: Salary Dynamics - Exploring the Impact of Experience on Earnings

Team Members: Lydia Myla and Parneet Kaur

Project Overview

Our project centers on analyzing two Kaggle datasets, detailing the experience and corresponding salaries of thirty employees. Motivated by research into human capital's impact on economic prosperity, the general objective is to characterize the correlation between work experience (in years) and salary compensation (in USD). Leveraging a Linear Regression Model, we seek to understand the priority of employee development and career mobility, which both continue to be derived from professional experiences. Here, we design our model to predict salaries (dependent variable) based on years of experience (independent variable). Given that we are using a simple linear regression model, we will most likely be able to minimize overfitting by deleting any possible outliers and reducing the complexities of the model. Although we assume that it is unlikely overfitting will occur, we continued to prioritize robust validation procedures during the 'Model Evaluation' phase.

Dataset: Project Dataset

For our project analysis, we accessed two distinct datasets from the Kaggle platform, each focused solely on one independent variable—years of experience—and one dependent variable—salaries. Recognizing the significance of amalgamating diverse perspectives, we merged these datasets to form a unified dataset that encompasses about 60 observations. Nonetheless, it is important to recognize that the datasets we retrieved only included the variables we had desired and did not undergo further manipulation or alterations. For example, during our research, we retrieved a dataset with more than one independent variable (e.g. years of experiences, state, country, age, etc.). However, in Data Science, including extraneous variables that are not directly relevant to the objective variables can introduce noise and potentially bias the results.

Previous Attempts

There have been a few prior works using Linear Regression to predict salary data and results. However, many of these projects deal with Multiple Regression Models where the Data Scientist(s) used multiple predictor values/independent variables, such as job title, years of experience, location, education level, etc. Most notably, a Data Scientist at a Japanese Fashion Company in Tokyo, Japan collected original, past data from the NBA leagues to predict NBA salaries. The Data Scientists utilize data cleaning (e.g. filter(), mutate(), etc.), correlation checks to evaluate the strength and direction of the relationship between two quantitative variables, data visualization (e.g. interactive plots), etc. Although the Data Scientist did not provide specific information on training accuracy, they did explain their regression analysis, which, in the chosen scenario, helps to understand how different basketball performance metrics (such as minutes played per game, points per game, assists per game, etc.) relate to player salaries. For example, the Data Scientist notes how for every additional point per game (PPG), the salary is estimated to increase by $686,815. Finally, the Data Scientist adopted a real life scenario (i.e. taking basketball Reference to be J.J. Redick) for testing purposes, achieving an output of $13,959,120 as J.J. Redick's salary for a particular season. Although we did not use such a complex Multiple Linear Regression Model, we acknowledge these previous reports/code to inform our understanding and approach to predicting salary data.

Methodology: Design, Process, Implementation, and Results

- **Importing Library**

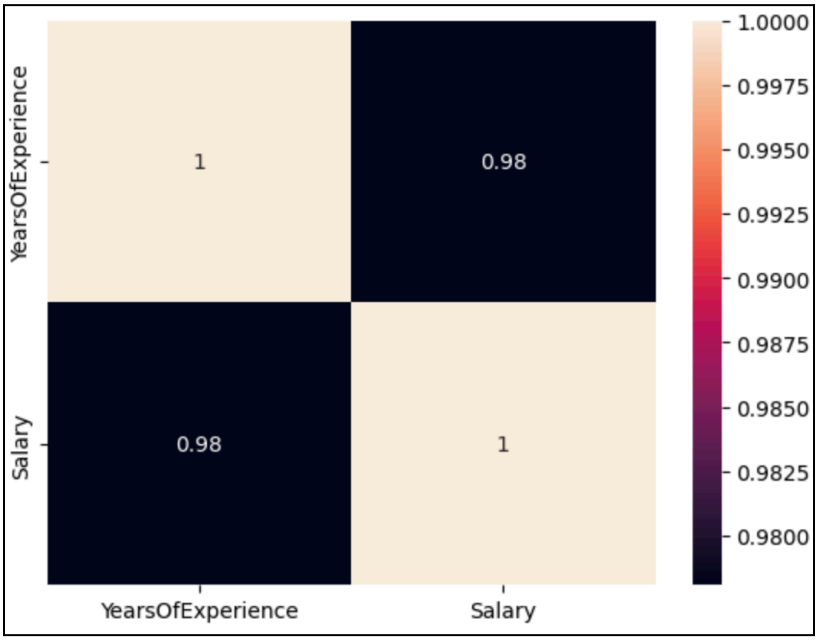
  In this initial step of our data analysis process, we lay the groundwork by importing essential libraries and loading our dataset. We imported critical libraries and modules tailored to our needs, including NumPy, pandas, Matplotlib, Plotly Express, and Seaborn. These libraries provide comprehensive support for numerical computation, data manipulation, and a diverse range of plotting options. Subsequently, we loaded our dataset, "salaries.csv," utilizing pandas' read_csv() function, which facilitated the creation of a DataFrame named data. From this initial DataFrame, we extracted pertinent columns, namely "YearsOfExperience" and "Salary," to construct a refined DataFrame. This selective extraction ensured our focus remained on the essential variables relevant to our analysis. Finally, by utilizing the head() function, we previewed the first few rows of

the DataFrame, enabling us to gain an initial understanding of the dataset's structure and content.

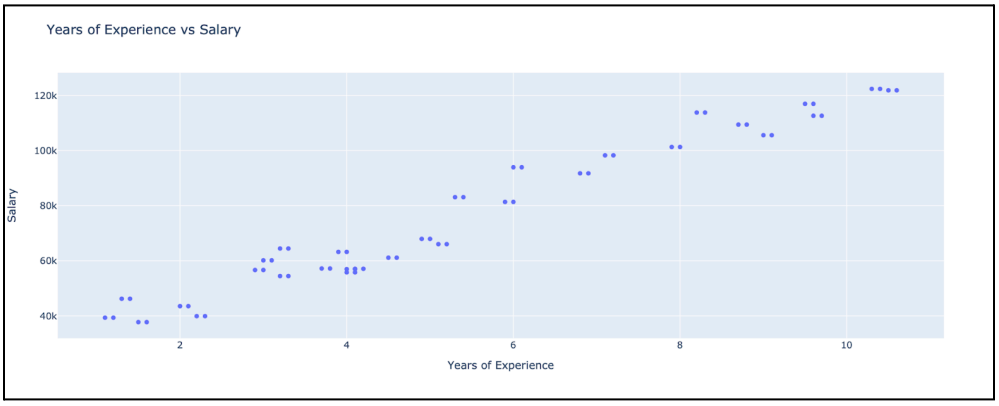
- **Data Information/Visualization**

  In this process, we performed exploratory data analysis (EDA) on a dataset represented by the DataFrame. Firstly, we inspected the shape of the DataFrame using df.shape() to understand its dimensions. Following that, we computed the correlation matrix using df.corr() to quantify the linear relationship between our independent and dependent variables. Both variables had a correlation coefficient of approximately 0.978, which is close to 1. This suggests that as years of experience increases, salary also tends to increase almost perfectly linearly. To visually represent the correlations, we created a heatmap using Seaborn's sns.heatmap() function, annotating the heatmap with correlation values. Additionally, we used Plotly Express (px.scatter()) to generate a scatter plot, depicting the relationship between years of experience and salary, with appropriately labeled axes. While heatmaps summarize correlations between multiple variables, scatter plots allow for the exploration of specific relationships in greater detail. Combining both types of plots enables us to validate patterns observed in the heatmap at a more specific level.

  - **Heat Map**



- **Scatter Plot**



- **Features**

  This feature step involves transforming specific columns from a pandas DataFrame `df` into NumPy arrays. The first line of code extracts the values from the 'YearsOfExperience' column, converting them into a one-dimensional array. The subsequent `.reshape(-1, 1)` operation reshapes this array into a two-dimensional format, with a single column and as many rows as necessary to accommodate all the elements. This process effectively prepares the data for machine learning tasks, particularly for regression analysis or predictive modeling, where the input features and target variable need to be organized in this manner for many machine learning algorithms to function optimally. The resulting arrays, `x` and `y`, are to be utilized for training and testing predictive models.

- **Train and Test**

  In this segment, we divided the dataset into separate training and testing sets, a common practice in machine learning. This function takes the predictor variables and the target variable as inputs, along with the optional parameter `random_state`, ensuring consistent and reproducible results across different runs. Upon execution, it generates four arrays: `x_train`, `x_test`, `y_train`, and `y_test`, representing the training and testing subsets for both predictor and target variables. The `x_train` and `x_test` arrays contain the respective sets of predictor variable values, while `y_train` and `y_test` store corresponding target variable values. This splitting strategy allowed for training the

model on one subset and validating its efficacy on unseen data from the other subset, aiding in assessing generalization capabilities and avoiding overfitting. We recognize that adjusting the `random_state` parameter can influence the partitioning of data, facilitating experimentation for optimal model performance.

- **Linear Regression Model**

  Linear Regression model is implemented and applied to predict salary values based on the predictor variable, years of experience. We named an instance of the Linear Regression model named 'lr'. The lr.fit(x_train, y_train) command trains the model using the training data, where x_train represents the predictor variable values and y_train indicates the corresponding target variable, salary. Following the model's training, the lr.predict(x_test) function generated predictions for the target variable based on the testing data. These predicted salary values were stored in the y_pred array. This process encapsulates the essence of linear regression modeling, where relationships between predictor variables and target variables are modeled through linear equations, enabling predictions for unseen data based on learned patterns from the training set.

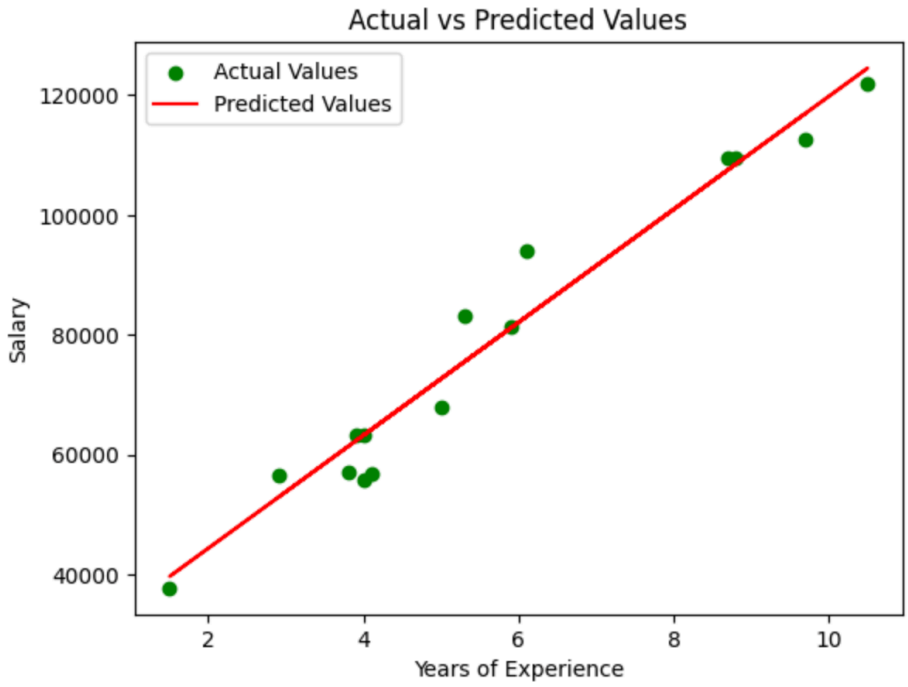
- **Model Evaluation**

  First and foremost, we performed a comprehensive evaluation of your machine learning model's performance by visualizing the relationship between actual and predicted values. Through the combination of a scatter plot depicting actual salary values against years of experience and a line plot representing the model's predicted salaries for the same set of data points, we gained insights into the model's effectiveness. By overlaying these two plots and labeling the axes appropriately, we provided a clear comparison between observed and forecasted outcomes. This visualization not only facilitated a qualitative assessment of the model's predictive capabilities but also aided us in identifying any discrepancies or trends between actual and predicted values.

  In addition to the graphical evaluation, we employed qualitative evaluation metrics to provide a more comprehensive analysis. The Mean Absolute Error (MAE) was calculated to quantify the average absolute difference between the actual and predicted salary values. This metric was useful as it offered straightforward interpretation of the model's

accuracy and provided a clear understanding of the magnitude of errors made by the model. Next, the Mean Squared Error (MSE) emphasizes larger errors more than smaller ones due to the squaring operation, making it beneficial for understanding the overall model performance and identifying any outliers. The Root Mean Squared Error (RMSE) was derived from MSE to provide a more interpretable measure of the average magnitude of errors in the same units as the target variable. Lastly, the R-squared (R2) score was calculated to assess the proportion of variance in the dependent variable (salary) that is explained by the independent variable (years of experience). R2 score offered insights into the goodness of fit of the model to the data, with higher values indicating a better fit.

- **Actual vs Predicted Values**



- **Metrics Analysis**

  The MAE of approximately 3916.94 indicated that, on average, the model's predictions deviate by about $3916.94 from the actual salaries. Meanwhile, the MSE of approximately 24,886,025.02 quantified the average squared difference between predicted and actual salaries, with larger emphasis on larger errors. The RMSE of around 4988.59 offered a more interpretable measure of average error magnitude, showing that, on average, the model's predictions are off by about $4988.59 from the actual salaries.

Additionally, the R-squared (R2) Score of approximately 0.96 indicates that the model explains about 96% of the variability in salaries based on years of experience. This suggests a strong fit of the model to the data and highlights its predictive power.

- Metrics Data

```
Mean Absolute Error:  3916.9382786566907
Mean Squared Error:  24886025.024355166
Root Mean Squared Error:  4988.589482444428
R2 Score:  0.9601886301941647
```

Conclusion

Our linear regression model is fairly accurate since it has an R2 score that is greater than 0.9 (90%). This high accuracy demonstrates the effectiveness of our Simple Linear Regression Approach in capturing the relationship between experience and earnings. Additionally, the simplicity of the Linear Regression model makes it widely applicable across various domains and industries (i.e. if there is to be further research). Our current analysis focused on a relatively small dataset, but we strongly believe that these methodologies can be scaled to handle larger datasets with minimal adjustments necessary.

Code File: Final Project Code

Resources and Links

- Kaggle
  - We retrieved two datasets from Kaggle in .csv files and combined them to create a larger dataset.
    - Dataset 1
    - Dataset 2
  - Cloud Convert: We used this website to convert files from xls to csv.
- Google Sheets
  - We imported the data (.csv) into a google sheet to organize, store it, and then download a full .csv file to use in our code.
    - Full Dataset
- Jupyter Notebooks

    - We implemented our project code on a shared notebook using Google Colab.
        - Notebook: Google Colab Code
        - IPYNB File: Final Project Code
- Python
    - We used Python and its libraries to write our code and implement our linear regression model.
- MATLAB
    - We visualized our data with graphs by using this tool.