Table of Contents

1. Project Description

This project is an analysis of the job market in the US for different jobs in the fields of software engineering, data science, computer science, and machine learning. It comprises two parts:

- Data collection (performed locally)
- Data analysis and model training (performed on Google Colab)
  Data cleaning and preprocessing are done in both stages.

The scripts used for data collection and initial preprocessing are available as `.py` files on this repo, and the collected data is in the `data` subfolder. Access the model preprocessing, training, evaluation, and other analysis at this Colab link: https://colab.research.google.com/drive/1NRiq1IHsgNwIU-4p_yUjPHQjp4dEpS7q?usp=sharing

2. How to Use

Training and Inferencing

1. Run the import statements at the top of the Colab document
2. Upload the .csv file (`combined_output_CLEANED.csv` , located in the `data` subfolder) when prompted
3. Run all cells below (`CTRL + F10`) to perform training and inference

3. Data Collection

Tools Used

Python libraries:

- Selenium
- Chrome Webdriver Manager
- Beautiful Soup

Data Sources

Glassdoor:

- https://www.glassdoor.com/Job/united-states-software-engineer-jobs-SRCH_IL.0,13_KO14,31.htm
- https://www.glassdoor.com/Job/united-states-software-developer-jobs-SRCH_IL.0,13_IN1_KO14,32.htm
- https://www.glassdoor.com/Job/united-states-data-scientist-jobs-SRCH_IL.0,13_IN1_KO14,28.htm
- https://www.glassdoor.com/Job/united-states-computer-scientist-jobs-SRCH_IL.0,13_IN1_KO14,32.htm

SimplyHired:

- https://www.simplyhired.com/search?q=software+engineer&l=
- https://www.simplyhired.com/search?q=software+developer&l=
- https://www.simplyhired.com/search?q=data+scientist&l=
- https://www.simplyhired.com/search?q=computer+scientist&l=

Collected Attributes

Attributes scraped directly:

- `Job Title, Company, Location, Skills, Education, Experience, Job Level, Salary`

Attributes constructed:

- `salary_min, salary_max,salary_avg, salary_type, salary_standardized, state, standardized_title, education_level, experience_level, job_level`

Number of Data Samples

A total of 3900 job entries were scraped and processed, with about equal numbers coming from Glassdoor and SimplyHired.

4. Data Preprocessing

Data Cleaning

Data cleaning was performed both locally and on Colab.
Locally:

- We filter out 4 features for each job— `Skills, Education, Experience, Job Level` — based on keywords found in the job listings
    - This is much simpler for SimplyHired than Glassdoor, as all these attributes are plainly listed in a `Skills` list, and we categorize them accordingly. However, for Glassdoor, the `Education, Experience, Job Level` fields were retrieved from reading the job descriptions. This was a very time-consuming process, taking over 30 minutes to scrape 600 jobs.
    - Salary standardization into various different features (`salary_standardized` becomes our target variable)
    - Standardization of location (into US states, Remote, or general United States)
    - Extraction and standardization of job title from job position name
    - Extraction and ordinal encoding of education level, experience level, and job level from the presence of certain keywords in the directly scraped fields
- On Colab:
    - Removal of rows which:
        - Are duplicates of other rows
        - Contain missing salary information
        - Contain salary outlier values (only a handful of these)
    - Correlation Analysis
    - Feature Selection
        - Separate for `Skills, state, standardized_title`
    - One-hot encoding of `Skills, state, standardized_title` fields

Data Integration

Each URL scraped produced a different output .csv file, saved locally.

Data Ingestion

The output .csv files (8 of them) were then combined using the `combine-csv-files.py` script in this repo. This output file was then cleaned all at once.

Data Description + Metadata Specification

Eight features were initially scraped, with many more created during preprocessing and encoding. Here is a sample of these first eight:

Job Title	Company	Location	Skills	Education	Experience	Job Level	Salar
Cryptologic Computer Scientist 2	Leidos	Odenton, MD	Computer science, Operating systems, Calculus, Data structures, Research	bachelor, associate, master, education, degree, doctor	2 years, 5 years, 3 years, 7 years		$126I $228I (Emp est.)

5. Feature Engineering

How Data are Processed + Prepared

- Data cleaning, as well as ordinal encoding, is performed both locally and on Colab
    - Relevant functions written include:
    - `standardize_salary(df)`
    - `standardize_location(df, use_unabbreviated_name=False)`
    - `extract_job_title(title)`
    - `standardize_job_title(df)`
    - `extract_education_level(text)`
    - `extract_experience_level(text)`
    - `extract_job_level(text)`
    - `standardize_3_experience_features(df)`
- Correlation analysis, feature selection, and one-hot encoding is performed on Colab
    - Relevant functions written include:
    - `clean_dataframe(df)`
    - `count_unique_values(df, column_name, top_n)`
    - `one_hot_encode_top_values(df, column_name, top_n)`
    - `analyze_skill_importance_by_role(df)`
    - `plot_top_skills_for_role(skill_importance_by_role, role, top_n=10)`
    - `rfe(df, num_features=5, select_only_skills=False)`
    - `create_encoded_dataframe(df, initial_features, categorical_features)`

Though most are self-explanatory, see function docstrings on a few of them for more detailed descriptions.

6. Model Development and Evaluation

Train + Test Data Partition

Train and test data were split at 70% and 30%, respectively, using the set seed here: `X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=12345)`

Salary Prediction (Model 1)

Machine learning model

sk-learn Random Forest.

Input to model

Various standardized features on salary, education/experience/job level, and one-hot encoded values for skills.

Size of training data

3900 job entries (before preprocessing) with 68 columns (you can see how it performs with fewer features by more strictly selecting skills).

Attributes to model

Number of estimarors: `100`
Random state: `12345`

Performance on training data

Metric	Value
MAE	12,938.34
RMSE	19,664.41
R²	0.8195

Performance on test data

Metric	Value
MAE	28,842.41
RMSE	38,288.94
R²	0.3488

As can be seen from the test data, this initial Random Forest model heavily overfits. Although the training accuracy could be significantly decreased through feature reduction, the test accuracy could not be improved this way.

Salary Prediction (Model 2)

Machine learning model

sk-learn Gradient Boosting (including finetuned hyperparameters), which was selected as the best pre-tuned model after comparison with many others (Random Forest, Linear Regression, Ridge, Lasso, Decision Tree, SVR)

Input to model

Everything sent to model #1, but additionally, one-hot encoded features for location (state) and standardized job title.

Size of training data

3900 job entries (before preprocessing and imputation) with 137 columns (large reductions did not help much).

Attributes to model

Final learning parameters found:

- `learning_rate: 0.11934708601565772`
- `max_depth: 4`
- `max_features: sqrt`
- `min_samples_leaf: 1`
- `min_samples_split: 18`
- `n_estimators: 175`
- `subsample: 0.9210407653155763`

Performance on training data

Baseline performance:

Metric	Value	CV Mean	CV Std
MAE	23,045.13	26,271.29	1,081.32
RMSE	30,685.95	35,215.32	1,874.56
R^2	0.5604	0.4281	0.0469

Performance on test data

Tuned vs. baseline performance:

Model	RMSE	Improvement
Baseline GB	35,215.32	-
Tuned GB	33,786.71	1,428.62 (4.06%)

Final test set metrics:

Metric	Value
RMSE	33,786.71
R^2	0.4930
MAE	26,027.20

Skill Importance

Description of exploited feature importance techniques

- Correlation Analysis: Pearson correlation coefficients are used to measure the strength of the relationship between each skill (binary features) and salary.
- Statistical Significance Testing: P-values are calculated to determine if the observed correlations are statistically significant, with a threshold of $p < 0.05$.
- Model-based Feature Importance: The RandomForest and GradientBoosting models extract feature importance scores directly from the trained models using the built-in `feature_importances_` attribute.
- Coefficient Analysis: For linear models (LinearRegression, Ridge, Lasso), the absolute values of the coefficients are used as measures of feature importance.
- Comparative Analysis: The code enables comparing feature importance across different models and job roles, providing a multi-faceted view of which features most strongly influence salary predictions.

Most important skills by job role

Note: Most of these skills are NEGATIVELY correlated with average salary! Nevertheless, they are the most important predictors. See Visualization section for full bar chart.

- Data Scientist: Machine learning, Azure, Git, C, Python

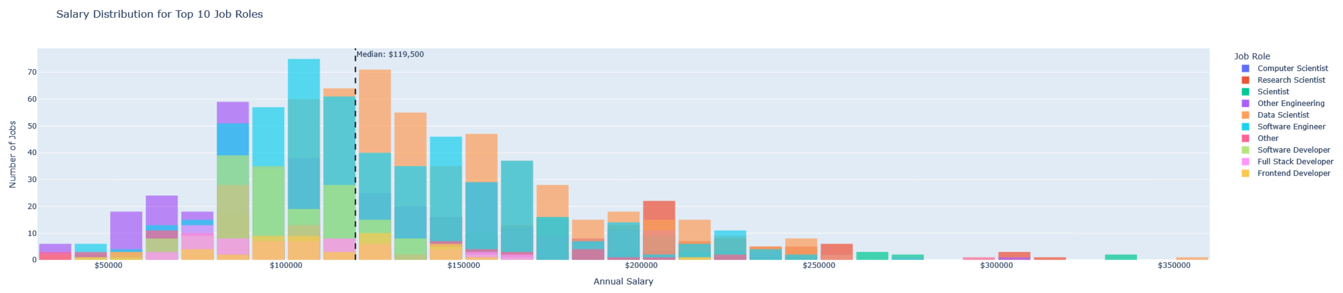
- Software Engineer: C#, SQL, English, writing skills, ASP.NET
- Other Engineering: SQL, English, CSS, AI, computer vision
- Software Developer: SQL, CSS, Oracle, HTML, Go
- Research Scientist: C, English, computer vision, data structures

7. Visualization

Histograms

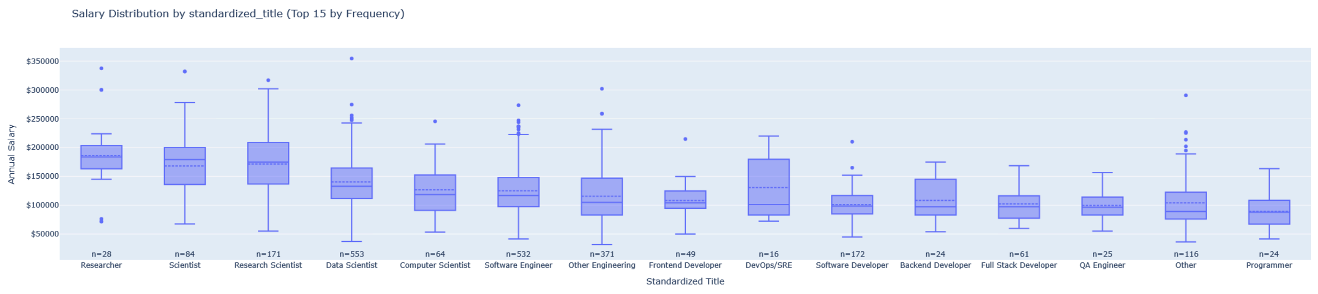
Salary distribution by job role (**see Colab link for interactive version**):
Some observations:

- The "Scientist" roles are almost universally above the median wage.
- The "Other Engineering" and "Software Developer" roles are almost entirely below the median wage.



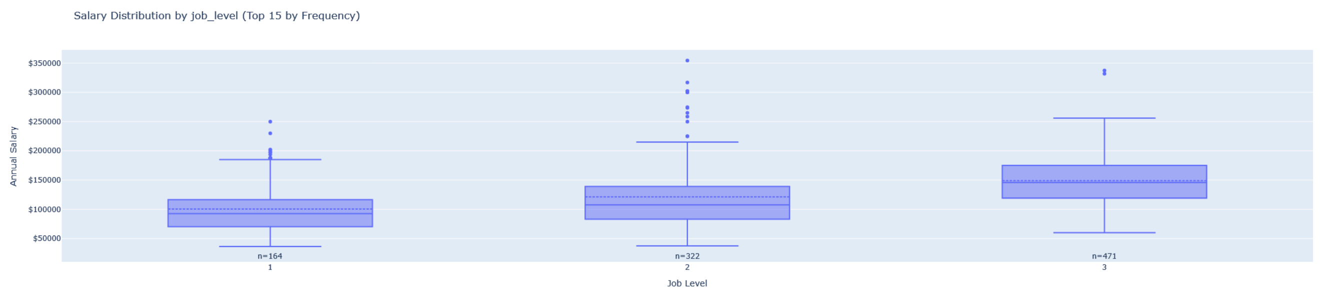
Boxplots

Salary distribution by standardized job title (**see Colab link for interactive version**):



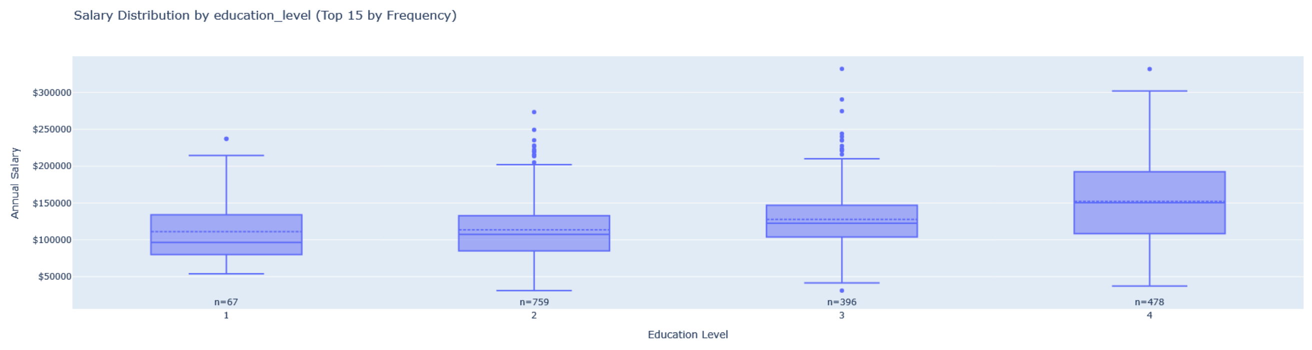
Salary distribution by job level (**see Colab link for interactive version**):

- 1 = Junior/entry level
- 2 = Mid-level
- 3 = Senior level



Salary distribution by education level (**see Colab link for interactive version**):

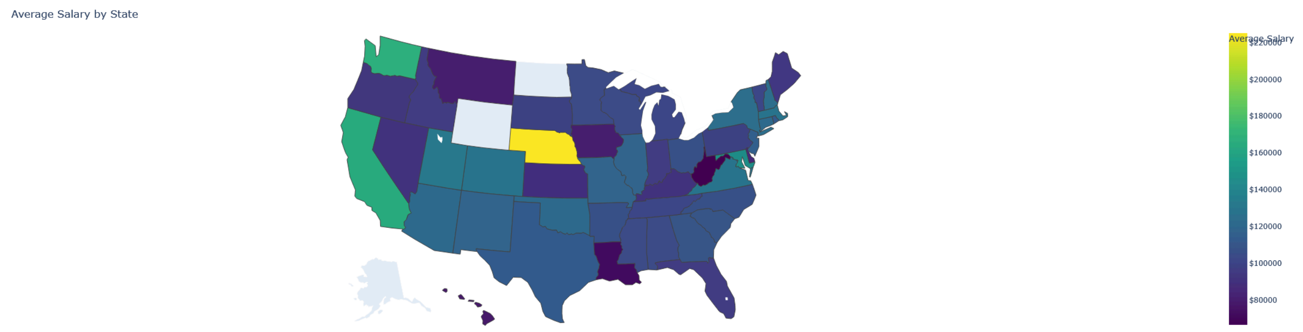
- 1 = Associate's
- 2 = Bachelor's
- 3 = Master's
- 4 = PhD



Map

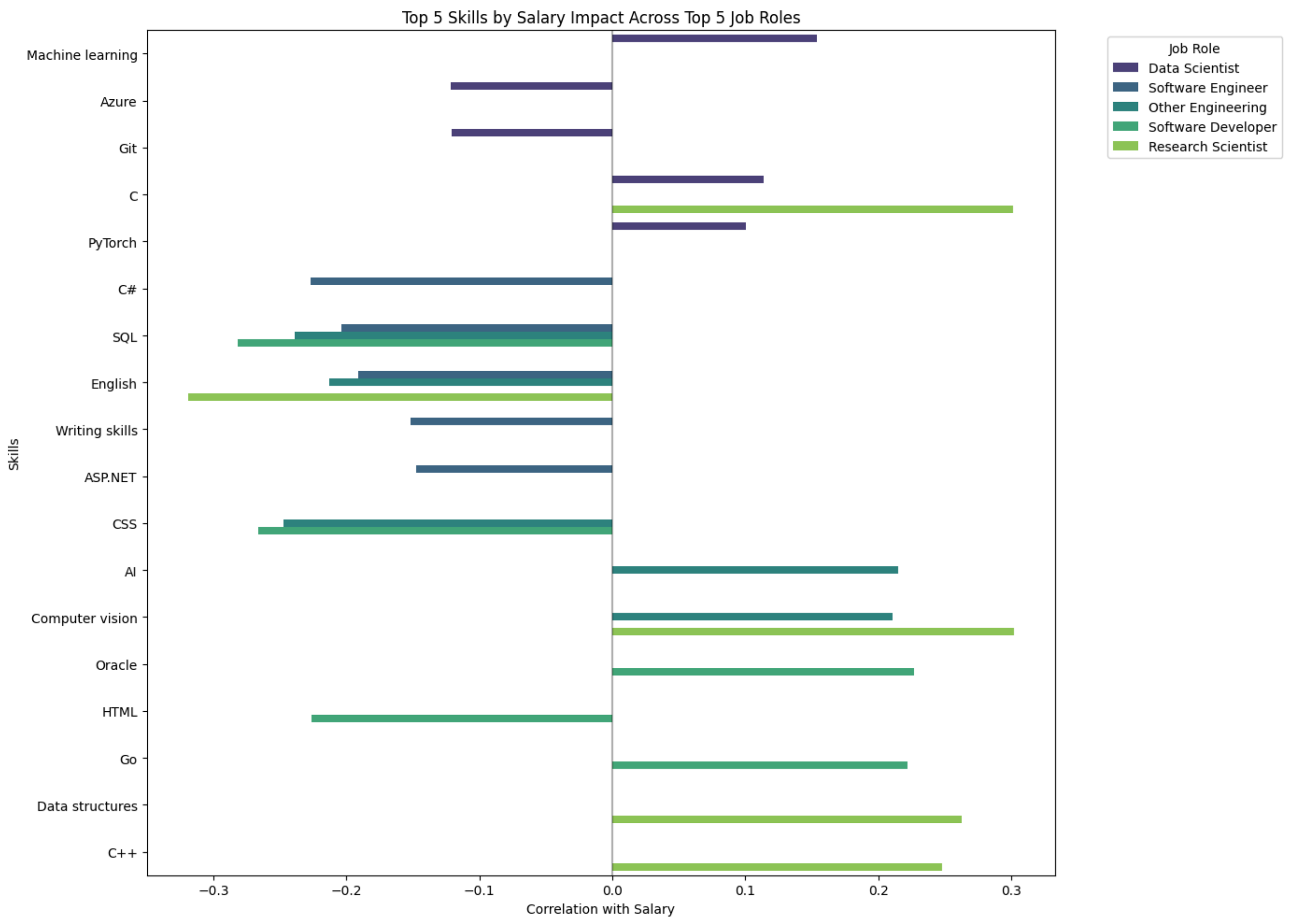
Average Salary By State (**see Colab link for interactive version**):

- California, Washington, and Maryland have the highest average salaries
    - (Nebraska has only a few entries that skew the salary significantly)
- West Virginia and Louisiana have the average lowest salaries

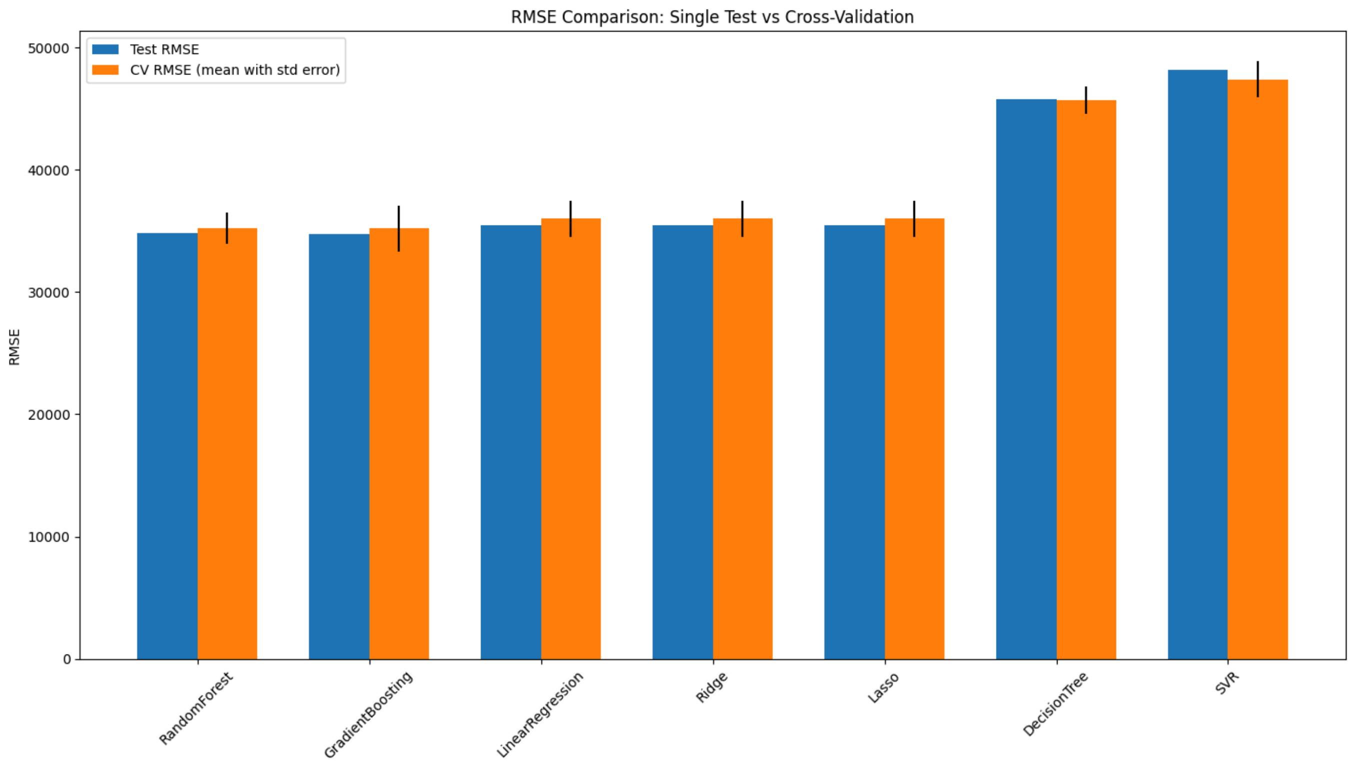


Bar Plots

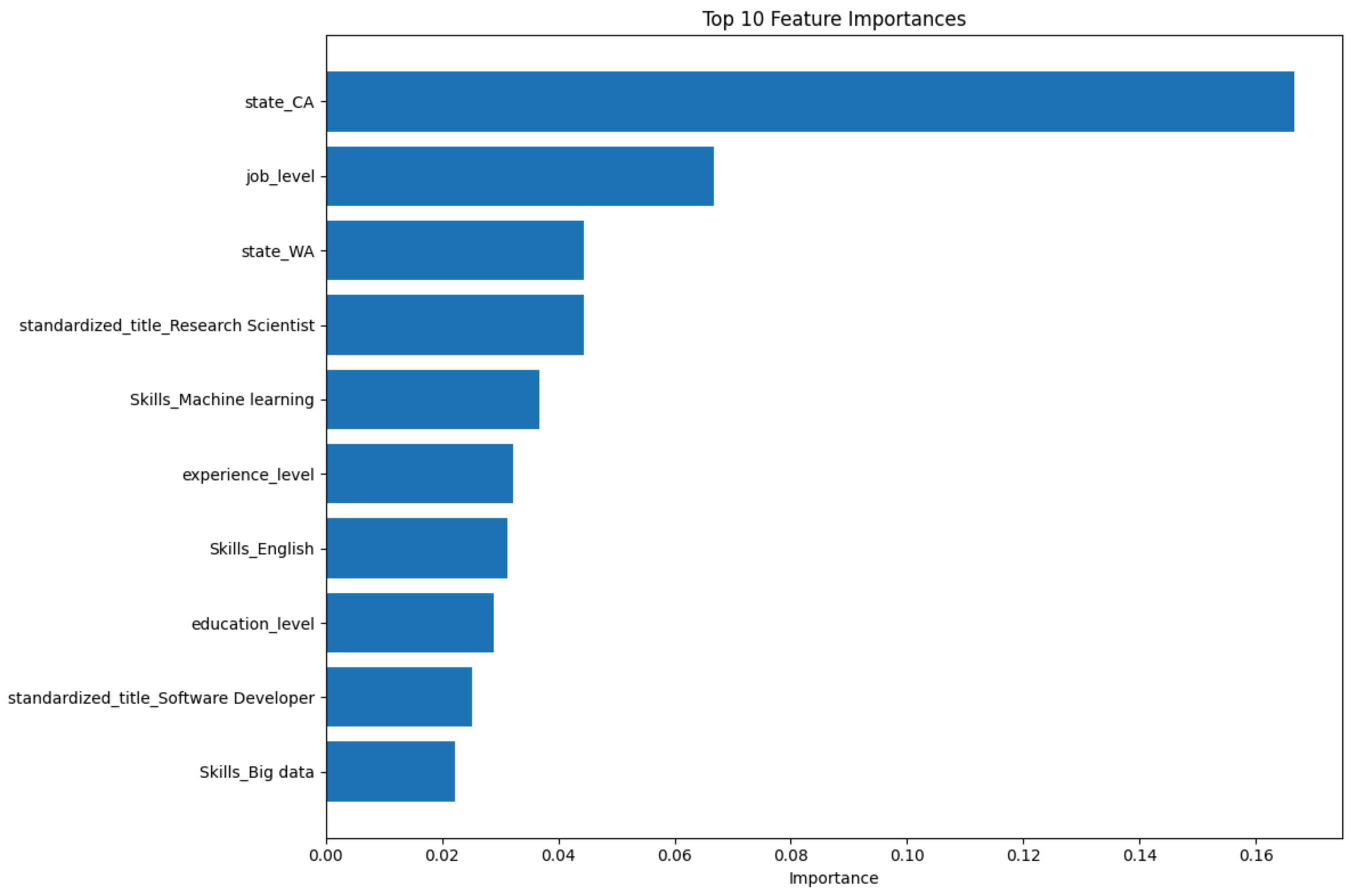
Top 5 skills by salary impact across 5 different job roles:



RMSE comparison of different models (final dataset, pre-hyperparameter tuning):

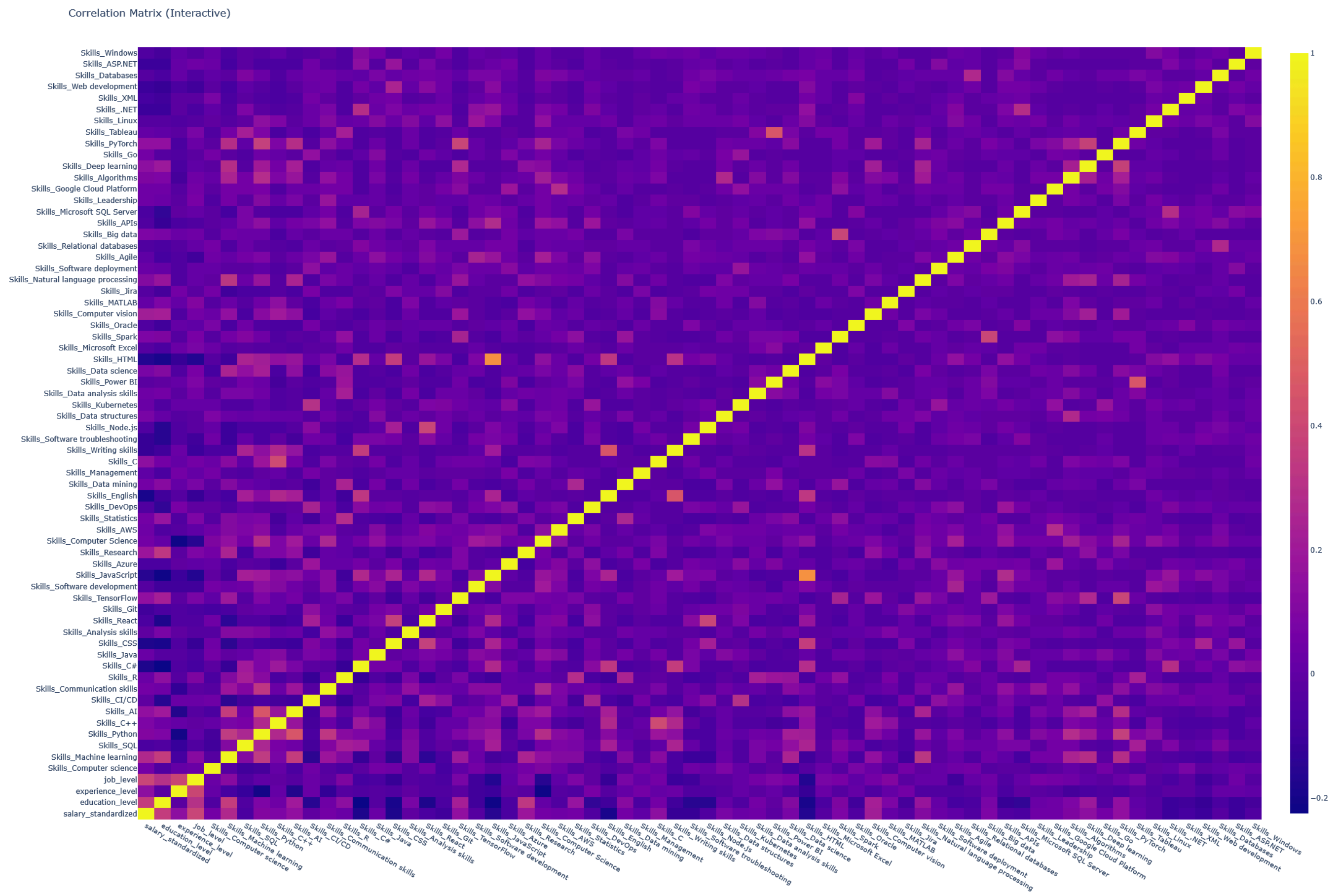


Final, top 10 most important features:

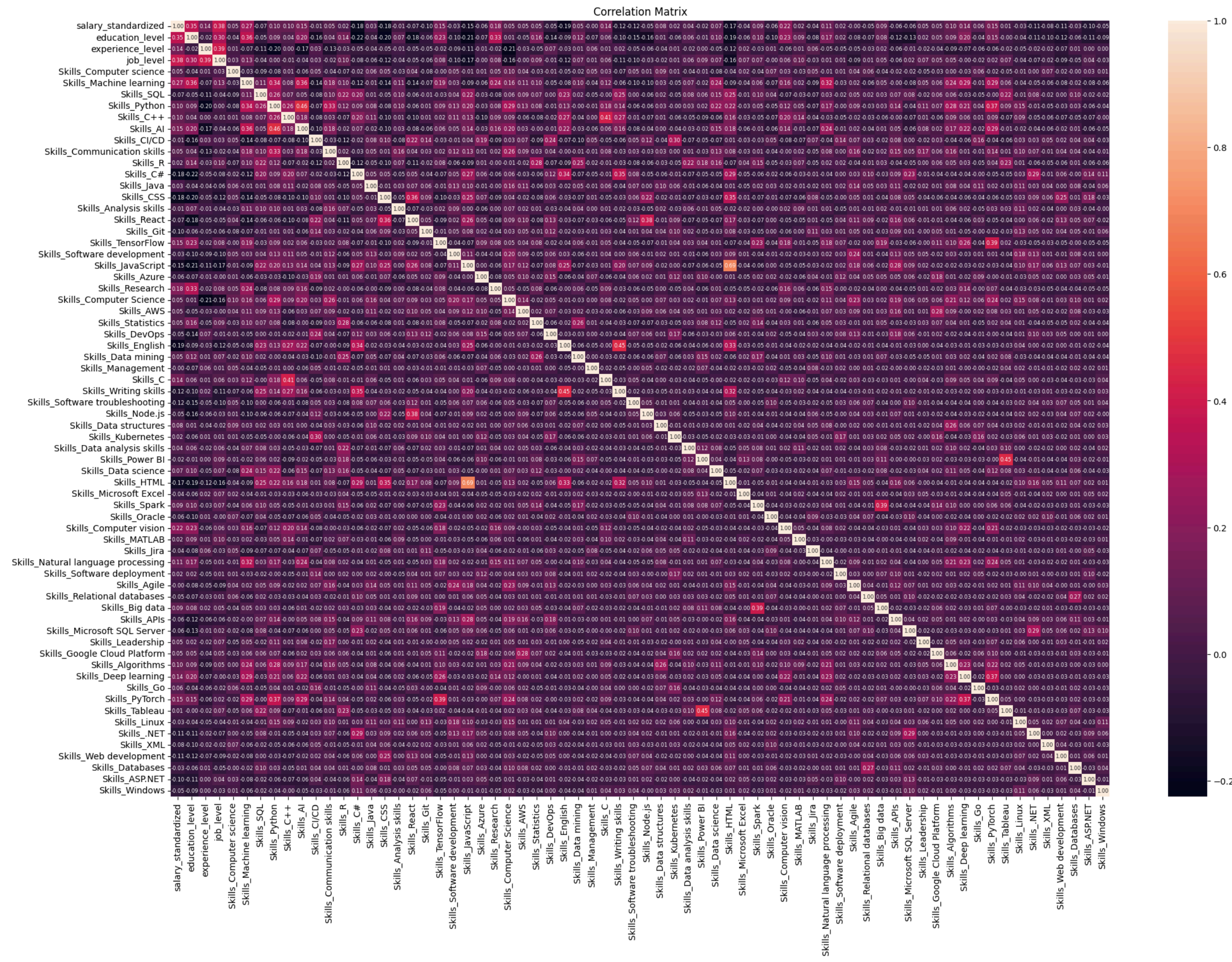


Heatmaps

Plotly feature correlation matrix (**see Colab link for interactive version**):



Seaborn feature correlation matrix:



8. Discussion and Conclusion

Project Findings

Ultimately, the top 10 most important features for salary were:

- `state_CA, job_level, state_WA, standardized_title_Research Scientist, Skills_Machine Learning, experience_level, Skills_English, education level, standardized_title_Software Developer, Skills_Big data`

These findings emphasize the importance of ordinal encoding in producing valid predictors: `job_level`, `experience_level`, and `education_level` were all inferred using keywords from each posting, and assigned a corresponding integer value where available. These were instrumental in getting model performance as high as it was.

Challenges

The majority of time spent on this project was during the scraping and preprocessing phase. Scraping from both SimplyHired and Glassdoor proved hugely challenging because the website structures were entirely different, no not only were the two scraping scripts separately adapted to the site layouts, but the methods of data ingestion and data integration were sometimes different (info on this detailed in the Data Cleaning section.)

Training the models to predict standardized salary was also difficult; although there are some things I did not attempt, I was not able to get the mean absolute error below $25,000 USD. Random Forest heavily overfit, and was easily able to attain over 90% accuracy on training data, but only about 40% on test data. Gradient Boosting barely overfit at all, but its test accuracy was comparable to Random Forest (before hyperparameter tuning, which boosted performance by about 4%).

Future Recommendations

Future avenues for improving model performance include:

- Review of feature engineering techniques, including implementation of PCA
    - Reduction in state/job title features
- Wider training/test dataset