R-ising to the Challenge:

Understanding R Adoption Trends Through Advanced Modeling Techniques

December 2024
Analysis by Dan Powell
Completed as Part of the University of Utah's MSBA Program

Executive Summary

R has been losing popularity in the last several years due to challenges from competitors and artificial intelligence. While there is still a vibrant community online supporting this language, it is slowly beginning to dwindle in size. To ensure its longevity, the R community needs to understand its competitive advantage to position itself aside competing languages. It is the goal of this analysis to determine who among the developer community is still interested in learning R so that R's developers and community can improve the overall applicability of R before it fades further into obscurity.

While R has been losing popularity, my analysis shows that there are key characteristics among developers that can be capitalized upon to improve R's longevity.

- Younger Developers (age < 34) are less likely to be interested in R than older audiences
- R is popular in academic settings
    - Developers with higher levels of education (Master's degree or higher) are more likely to be interested in R
    - Developers who work in an academic professions (Scientist, Professor, Student) are more likely to want to learn R
- Those with an interest in data science language (Python, R, MATLAB) or database query languages (SQL) are more likely to want to learn R
    - Developers using R want to continue learning more about it
    - Developers who want to learn R also show interest in functional and scripting languages such as Crystal, VBA, and Cobol
    - However, those currently using MATLAB and Crystal are less likely to want to use R
- Developers in Fintech and software industries are less likely to want to learn R. Whereas developers in government and healthcare are more likely to want to learn R.
- Developers with higher compensations are less likely to have an interest in R

With this in mind, R's community can take these actions to improve the languages longevity:

- Promote R usage among those in healthcare and government industries. Continue to promote R as a language to be used in schools and academic professions.
- Demonstrate R's capabilities to those in the fintech industry for modeling, forecasting and other financial uses
- Promote R as a free alternative to MATLAB and other paid-for languages used in statistics
- Build packages to improve R's compatibility with other scripting and functional programming languages
- Use predictive modeling insights offered by the XGBoost and Linear Regression models to find those who may be interested in using R but have not encountered it yet

Additionally, despite the analyses' findings, there are still some limitations to be considered:

- **Data availability:** Survey results for the Stack Overflow Developer survey are annual and anonymous, requiring additional data collection methods to be developed outside of the Stack Overflow survey
- **Potential for Overfitting:** Models had high performances, but this could potentially be due to overfitting, despite the steps taken to avoid it in the analysis
- **Performance Concerns:** Dimensionality had to be reduced for the neural network and XGBoost model, potentially leading to information loss, especially in the case of the Neural Network which required the training dataset to be lowered to only 500 observations.

Introduction

R is falling in popularity. The rise of artificial intelligence's capabilities of completing complex statistical tasks and the increasing capabilities of languages like python in the field of data science have caused R to lose some of its audience. The goal of this analysis is to identify who among the developer community is interested in learning R to create models that can be used for targeting audiences that may be interested in R but may not have heard of it before.

Data Description

My dataset comes from Kaggle.com but was originally pulled from the results of the 2024 Developer Survey conducted by Stack Overflow. This survey collects data about the users of stack overflow, a popular forum site for developers to ask and receive answers on questions relating to programming.

I utilized feature selection and data cleaning to develop our final dataset which will serve as our predictors in our model. These methods included one-hot encoding, grouping certain categorical characteristics together into binary columns and scaling our numerical variables to prevent them from over influencing the models. A description of each of the features and its datatype can be found in the appendix of this report (Appendix 1.1).

I also have the binary target variable, want_to_work_with_R. This will serve as a representation of those who reported on the survey that they were interested in learning R. It also dictates that our models must be suited for classification tasks going forward.

To ensure that compensation totals were reported in US dollars, I limited our data to just those who reported residing in the United States. This lowered our total sample size to 11,078 observations from 65, 437 observations. Additionally, I imputed nulls in our numeric columns with medians. The binary columns did not contain null values, as missing responses were categorized as 0. The numerical features were also scaled to have a mean of 0 to ensure that their magnitude did not impact the models in the analysis. Finally, I eliminated extreme outliers in CompTotal that did not seem realistic in our data, which included observations over a million dollars.

When complete, we have a dataset of 127 variables with 11,078, many of which come from the one-hot encoded categorical variables for want_to_work_with and currently_work_with for the various programming languages available in the survey.

Methodology

To achieve the goals of understanding the data and producing effective predictive models, I used 5 different analytical techniques which combined supervised and unsupervised learning to investigate the factors that influence an individual's interest in R. To begin, I first ran PCA (Principal component analysis) on the cleaned data to get the principal components. This serves to isolate the key features of our data to reduce its dimensionality. I then utilized hierarchical clustering to create dendrograms using average, ward, and complete linkage methods that utilize Euclidean distance. Hierarchical clustering was utilized since I did not know how many clusters should be created and suspect there to be hierarchical relationships within our data (e.g. experienced developers may then split into experienced developers who work in different industries). The intention here is to discover if one of the larger clusters will have a

higher proportion of individuals interested in learning R. By analyzing the characteristics of this and other clusters, we can gain valuable insights into the overall survey data.

Next, we begin our predictive modeling with a lasso regression model. The lasso regression model was chosen due to its interpretability while also selecting the most impactful features. Prior to running the lasso regression model, I corrected class imbalances using SMOTE (Synthetic Minority Over-sampling Technique). This ensures bias from the majority class does not influence the model. Additionally, I used a 5-fold cross validation to select the best lambda value for the model to optimize its performance. This model will also display the magnitude and direction of the variables it deemed to be most valuable and offer us insight into a potential target audience's characteristics.

Before continuing, to mitigate potential performance and generalization issues in subsequent analyses, I reduced the dimensionality of the data by combining the 'want_to_work_with' and 'currently_work_with' columns into a single column representing an interest in specific language types (e.g., data science languages, functional languages, etc.). This combined dataset was then utilized in the subsequent XGBoost and SoftMax Neural Network models.

The first model to use this dataset is the XGBoost model. For context, the XGBoost model leverages an ensemble of smaller decision trees to identify important features and their impact on the target variable. Prior to fitting the model, I corrected for class imbalance using SMOTE on a randomly selected training sample of 7,000 observations due to performance concerns. I also set up a recipe for the ensemble method to tune the learning rate, number of trees, and depth through a grid search on two levels. This was used on top of a five-fold cross validation to improve the model's robustness. The primary outputs used from this model, besides its predictive power, are the PDP and feature importance graphs, which will display which variables contributed the most to the model and their magnitude.

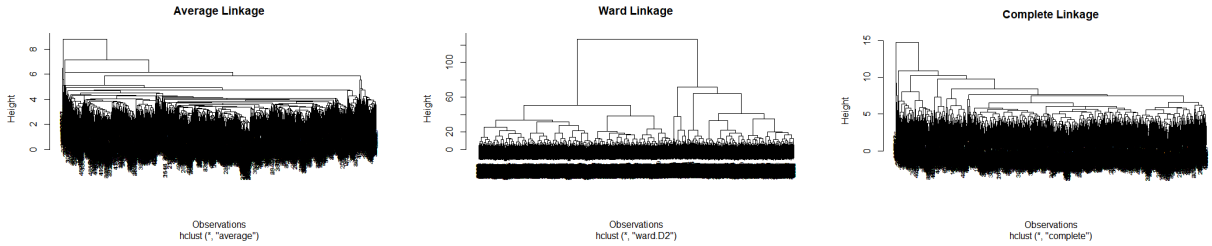
The final model used was a SoftMax Neural Net. This model was primarily designed for predictive power, as neural networks are inherently difficult to interpret due to their black-box nature. Even so, neural nets have a strong ability to handle many independent variables and identify which features are most important, which is crucial for our high-dimensional data. Additionally, the neural network algorithm can identify non-linear decision boundaries well, which are suspected to be in our data. Prior to fitting our Neural Network model, I utilized a train-test split (80-20), SMOTE for class imbalance of the training data, and 5-fold cross validation that fitted the model on 42 different hyperparameters for decay and size to minimize the error. Finally, the AVNnet method hyperparameter was chosen to reduce overfitting.

Results and Findings

PCA and Hierarchical Clustering

The 5 models I produced each showed their own strengths and weaknesses. To begin with the unsupervised hierarchical clustering algorithm which I utilized primarily for data discovery, I began by first extracting principal components from the data. To do this, I extracted 5,000 samples from our dataset and calculated the eigenvalues and eigenvectors from the variance-covariance mix. I then utilized these attributes to get the cumulative variance explained by the principal components. This revealed that the top 80% of variance is explained by the first 29 principal components out of 127 total components generated. I then used matrix multiplication to check for correlation in the eigenvector, which revealed that the data was uncorrelated in the new variance-covariance mix.

Now that I had the principal components, I moved into hierarchical clustering. The first step was to decide on a linkage method to use with our principal components. For this, I tested average, complete, and ward linkages using Euclidean distance. Below displayed the dendrograms produced by each of the linkage methods:



While the ward linkage appears to have clearer splits, I still gathered the cophenetic correlation coefficients for each linkage method to compare the two. When compared on this metric, the average linkage method performed much better than the other two methods, with a cophenetic correlation coefficient of 0.739. A comparison of each of the cophenetic correlation coefficients can be viewed in the HTML under the section “Trying different Linkage Methods.”

The next step was to understand where to split the linkage method. Since I just wanted a general idea of the landscape of the data, I asked the algorithm to find the optimal number of splits between 2 and 10 clusters. Here, it produced 2 clusters as the optimal split - one with 4973 observations and the other with 27 observations. Out of curiosity, I also cut the tree at 10 clusters to see the number of units per cluster. Here presents a bit more of a divided group, with 3783 clusters in one cluster and around 100-300 observations in the other top five largest clusters. The remaining clusters show very small numbers of observations.

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7	Cluster 8	Cluster 9	Cluster 10
348	3783	311	196	251	30	62	6	9	4

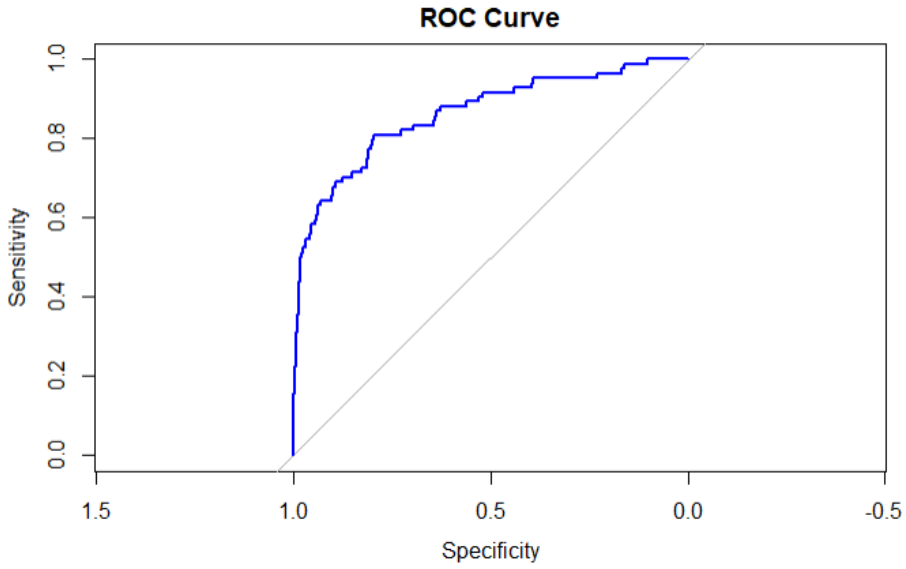
Next, I assigned each of our observations to their respective cluster utilizing the 10-cluster split to see the composition of each. For the sake of simplicity, we will just be looking at the 5 largest clusters; clusters 1 through 5 in the chart below.

Cluster	**Cluster 1**	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
Observations in Cluster	348	3783	311	196	251
CompTotal_mean	-0.019	-0.205	0.193	2.257	0.061
Software Development Industry mean	0.229	0.127	0.019	0.291	0.251
academic_prof_mean	0.011	0.126	0.023	0.015	0.020
isyounger mean	0.011	0.542	0.000	0.500	0.000
Want To Work With R Mean	0.046	0.046	0.045	0.015	0.044

In sum, we can see that want_to_work_with_R was likely not the main divider for the clustering algorithm. However, the cluster with the lowest average interest in R was characterized by a higher proportion of younger developers, higher compensation levels, and a greater prevalence in software development roles. This offers some evidence to support the claim that younger audiences may not have an interest in R. Additionally, those who work with software and who are paid more may have less of an interest in learning R as well. This could be a potential obstacle to overcome, as hopes of higher compensation is usually a significant motivator for learning a new programming language. Now that we have a better view of the data we are working with, we now move into our first predictive model: Lasso regression.

Lasso Regression

As stated before, lasso regression was chosen due to its optimized performance, interpretability, and ability to conduct dimensionality reduction, which is likely needed due to our large number of predictors. After conducting the preprocessing for SMOTE, train-test splits, and selecting the best lambda for our lasso model through cross validation, I was able to produce a model that reduced dimensionality significantly, cutting out 69 of the original predictors and leaving 56 that the model decided were most impactful. The model itself was high performing, producing an F1 score of 0.982 and an accuracy of 0.965 with an ROC curve that appeared to hug the top left of the chart between specificity and sensitivity, which is ideal.



The top 10 variables and their impact can be seen below - but for further analysis, please see the HTML document in the section “Optimal Lambda Value and Coefficient Estimates” or in the appendix (Appendix 2.1).

Variable	Coefficient	Odds Ratio
have_worked_withR	3.386	29.5509961
want_to_work_withNA	-1.249	0.287
want_to_work_withCobol	1.064	2.897
want_to_work_withCrystal	0.949	2.583
want_to_work_withVBA	0.924	2.521
want_to_work_withZephyr	0.910	2.485
have_worked_withCrystal	-0.859	0.423
want_to_work_withMATLAB	0.768	2.155
want_to_work_withPython	0.676	1.965
want_to_work_withClojure	0.610	1.841

The analysis reveals a strong positive association between R usage and the desire to continue working with it, indicating a promising future for R as existing users maintain interest in the language. Additionally, individuals interested in working with R also seemed to express interests in a wide range of languages, including Cobol, Crystal, VBA, Zephyr, Python, Clojure, and MATLAB. Conversely, those who had no interest in learning a new language (or who didn't fill out that part of the survey) and those who currently use MATLAB and Crystal had a negative relationship with wanting to work with R.

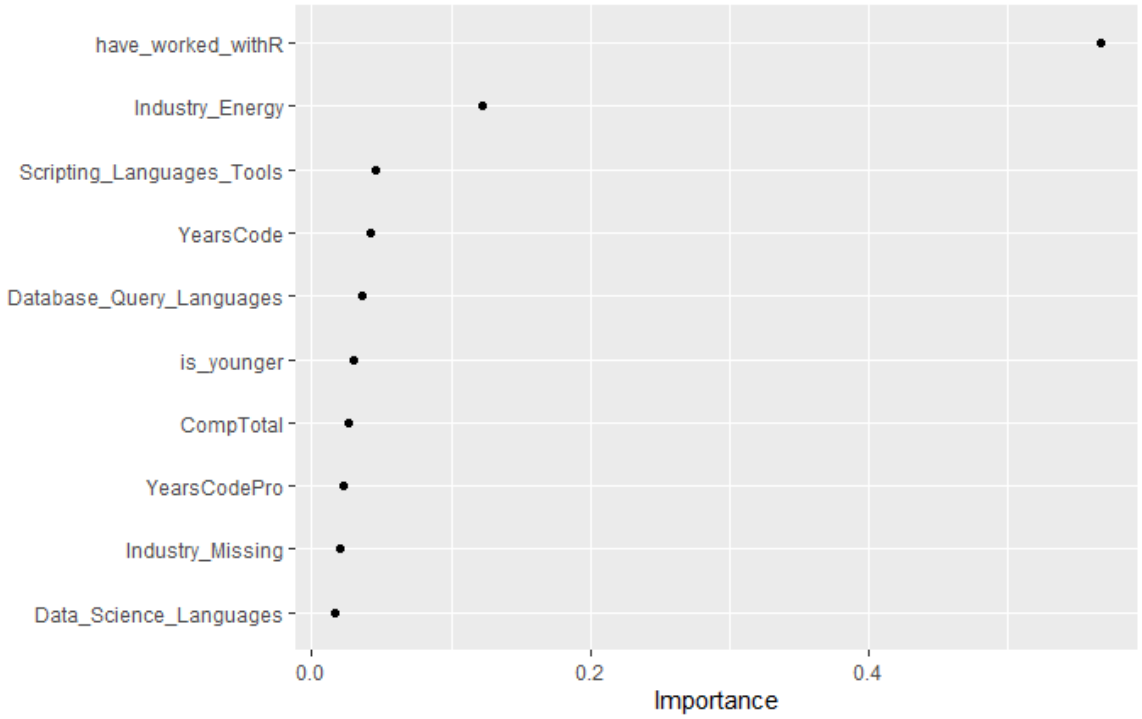
Further into the coefficients, we can find some of the other variables of interest. Individuals working in academia (e.g., professors, scientists, and researchers) and those with advanced degrees (e.g. Masters or Ph.D), showed a strong positive association with R usage, with the odds of interest in R increasing by a factor of 1.52 and 1.48 respectively. R was also popular with those employed in a governmental or healthcare industry, with the odds of interest in R increasing by 1.15 and 1.02 respectively. Conversely, individuals in the fintech and software development industries were less likely to be interested in R, with odds ratios of 0.72 and 0.82, respectively. Similarly, younger participants and those interested in functional languages like Ruby and Rust were less likely to express interest in R, with odds ratios of 0.91, 0.82, and 0.85, respectively. These findings suggest that interest in R may be more prevalent in certain academic and research-oriented fields while facing challenges within the rapidly evolving and potentially younger audiences of software development and fintech sectors. Next, we will continue to expand our analysis' robustness by continuing with our XGBoost model.

Boosted Trees: XGBoost Model

The XGBoost model also performed well on our test data. Prior to fitting the model, I simplified our data to reduce its dimensionality, conducted SMOTE on the training data, and tuned the number of trees, learning rate, and depth through grid search on top of a five-fold cross validation. After fitting the model with the best hyperparameters based off of F-measure, the model achieved an accuracy of 0.947 and an F1 score of 0.972 - which is slightly lower than the performance from the Lasso Regression. Still, the high F1 score does continue to sound alarm bells for the potential for overfitting, despite our efforts to prevent it.

One of the benefits of XGBoost models is that they are able to provide feature importance and PDP charts for the user to analyze each of the model's predictors and which ones contributed the most to the model and to what magnitude. For reference, the PDP charts can be viewed in the HTML document

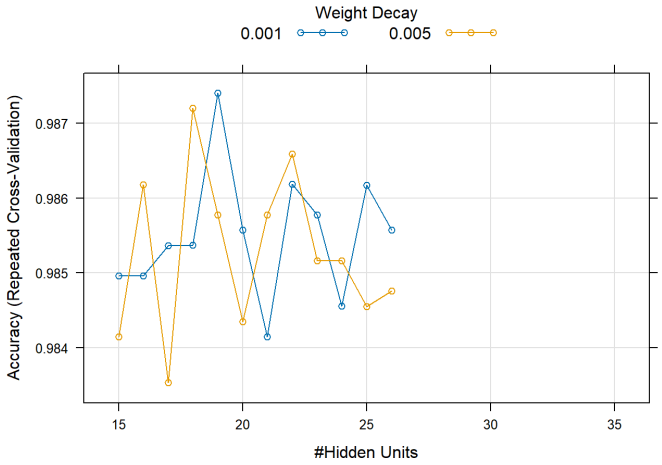
under the section “PDP Charts” and in the appendix (Appendix 3.1), as we will only be displaying the feature importance charts in this report and discussing some of the more notable attributes’ magnitude.



From the feature importance chart and PDP graph, it is evident that prior R experience is strongly correlated with interest in working with R. Additionally, interest in database query languages like SQL also increased the likelihood of an interest in working with R. While the 'Energy_Industries' variable was deemed important, it had a minimal impact on R interest. Similarly, interest in scripting languages had a slight positive correlation with R interest. These relationships might be influenced by complex interactions or non-linear effects not fully captured by the PDP charts. This model also demonstrated a positive relationship between being younger and wanting to learn R, which differed from the findings of the lasso regression. However, this could be due to the limitations stated earlier regarding a PDP plots ability to display complex nonlinear relationships. Finally, we can also see in the PDP charts that having interest in a data science language greatly improves the chances that the user would also be interested in learning R.

Neural Net

The final model ran for this analysis is a neural net. The neural net ended up being the weakest of our predictive models with an F1-score of 0.962 and an accuracy of 0.9282. While still a strong performance, the model did not perform as well as some of its counterparts. The model seems to have chosen the hyperparameters of 0.001 weight decay and 19 hidden units (size), as demonstrated by the chart below. Unfortunately, due to the black box nature of Neural Networks, I cannot make many more interpretations about how the model formed. However, due to the small sample size it was fit on and the robust hyperparameter tuning process over a 5-level and 5-fold cross validation, it may prove to be useful if the other models show signs of overfitting in real-world application.



Conclusions

Now that I have run each of the 3 predictive models, I will compare them to make a determination on which model is best suited for prediction. Below displays our 3 predictive models' performance across the metrics of F1-Score and Accuracy.

Model Name	F1-Score	Accuracy
Lasso Regression	0.982	0.965
XgBoost	0.972	0.947
Neural Network (SoftMax)	0.962	0.928

From the chart, we can see that the lasso regression performed the best out of all the models for predictive power, with an exceptional F1-score of 0.982 and an accuracy of 0.965. From the coefficient chart, we could see that the primary variables contributing to its predictions were having worked with R before, and a wide range of interests - specifically for other commonly used languages in data analysis such as VBA, Python, and MATLAB, which aligned with our findings in the XGBoost model. Interestingly, currently working with Crystal or MATLAB seemed to demonstrate a negative relationship with an interest in working with R, perhaps due to the performance differences between the two languages in the case of Crystal (Crystal is known for its performance and scalability whereas R can be slower for large-scale data processing).

However, while our Lasso regression performed well, it may prove challenging to gather preferences at the granularity offered by the lasso regression. To remedy this, we fit the XGBoost model on a simplified dataset which used "interest" in a particular category of programming languages instead of specific languages and the users relationship with them. Since it performed very similarly to the Lasso Regression in both F1 and Accuracy, it could be utilized as an effective alternative for less granular data.

In conclusion, each model produced findings that either aligned or conflicted with that of the other models. For similarities, both the XGBoost and Lasso Regression indicated that interest in data

science, scripting languages, and database query languages were positively correlated with an interest in working with R. However, the Lasso regression and XGBoost models presented conflicting findings regarding the relationship between age and interest in R, while the latter may have been influenced by interaction effects or data complexity. While the Lasso regression suggested a negative association, the XGBoost model indicated a slight positive relationship between younger age and R interest. Even so, the clustering analysis aligned more closely with the Lasso regression, indicating that the group with the least interest in R was also the youngest, providing more evidence to conclude that younger audiences may be less interested in learning R.

Limitations and Future Implementations

While each model applied different forms of robustness checks to enhance generalizability, there are still limitations to our models that may have impacted the performance. First, null observations for several of the variables were included in the dataset as want_to_learn_NA or Industry_Missing values. While it could mean that the user does not have an interest in working with or does not currently work with any of the programming languages, it could also mean that they skipped that question on the survey, as it was optional according to the meta data. The high importance of the "Industry_Missing" variable in the XGBoost model suggests that missing industry information may have influenced our findings and potentially masked the impact of other variables. Furthermore, our data may have been influenced by interaction effects among our variables. Future Analyses would do well to clean the data further to determine whether responses were skipped and further study potential interaction effects in our data.

Additionally, for the hierarchical clustering, XGBoost, and Neural Network models, we had to reduce either the dimensionality or the number of samples to train our model. While in some cases this could help our generalizability, it could also lead to information loss which could impact our models' performances. This could be improved upon by using resources with higher compute that can handle the complexity of the models with more data or higher dimensionality. Lastly, if we implement the predictive models to determine who may have an interest in learning R, we may encounter issues with data availability. Unfortunately for our purposes, the Stack Overflow survey doesn't provide user IDs, making direct targeting impossible. Potential solutions include data collection from Stack Overflow or GitHub, or using Google Ads to target users who frequently search R-related topics that we discovered in our analysis (e.g. data science languages, resources for academic professionals, etc.).

Finally, those interested in continuing this analysis should consider the following. First, I suggest implementing additional model types. Although I employed both supervised and unsupervised models, yielding consistent findings, future research could further strengthen the analysis by incorporating ensemble methods like AdaBoost or Random Forest for classification. Additionally, other classification models more robust to overfitting, such as Support Vector Machine, could be utilized to expand the generalizability of the analysis. Moreover, the hierarchical clustering analysis might benefit from different dimensionality reduction techniques or clustering algorithms like partition or density-based methods. I also believe that longitudinal data could significantly enhance the models by providing insights into R usage trends. For example, longitudinal data could help determine whether individuals in the fintech sector have never shown interest in R or if their interest has declined over time, leading to the observed negative relationship. This trend analysis could provide valuable insights into how to target this potential opportunity area. In sum, future research should explore alternative models (e.g., Adaboost, Random Forest, SVM), investigate different dimensionality reduction and clustering techniques, and incorporate longitudinal data to gain a deeper understanding of evolving trends in R usage.

Appendix

Appendix 1.1: Description of Variables

Variable Name	Type	Description
want_to_work_with_[language Name]	Binary	One Hot encoded categorical variable. Takes a value of 1 if

		the developer answering the survey wants to learn that language, 0 if the language was not selected as a language they want to learn.
currently_work_with_[Language Name]	Binary	One Hot encoded categorical variable. Takes a value of 1 if the developer currently works with that language and a value of 0 if they do not currently work with that language
industry_[Industry name]	Binary	One Hot encoded variable. Takes a value of 1 if the developer reported working in that industry.
is_younger	Binary	Takes a value of 1 if the user is less than 34 years old.
Advanced_ed	Binary	Takes value of 1 if the developer has a master's degree or higher
Higher_ed	Binary	Takes a value of 1 if the developer has an associate or bachelor's degree
lower_than_higher_ed	Binary	Takes a value of 1 if the developer does not have a college degree of any type (some college or lower)
Academic_prof	Binary	1 if the developer works in an academic profession (example: Scientist, Academic researcher, Student)
Use_AI	Binary	1 if the developer currently uses or wants to use AI in their daily workflows
Years Code	Numerical	Number of years the developer has coded for
Years Code Pro	Numerical	Number of years the developer has coded professionally for
Comp Total	Numerical	The amount the developer reports to make in USD
Work Experience	Numerical	The number of years of work experience the developer has.

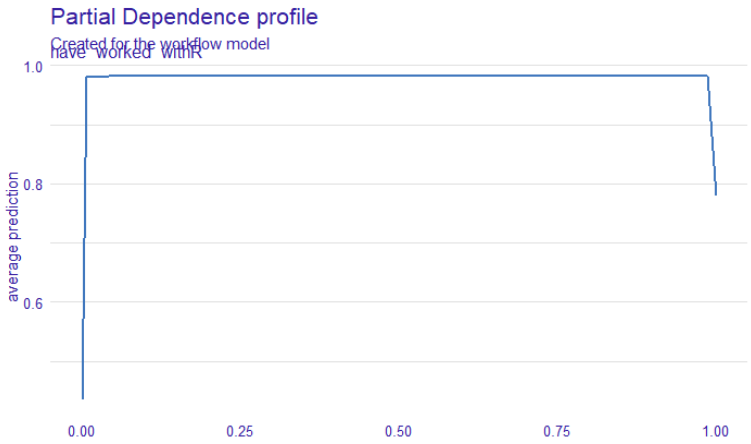
Appendix 2.1: Coefficients and Odds ratios from Lasso Regression

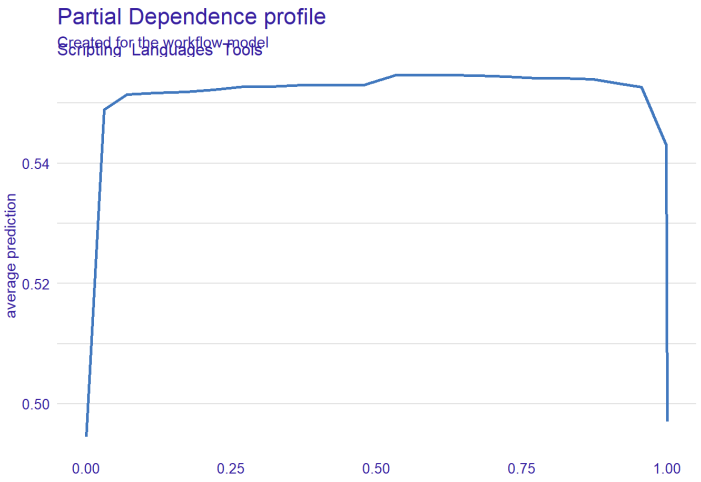
Variable	Coefficient	OddsRatio

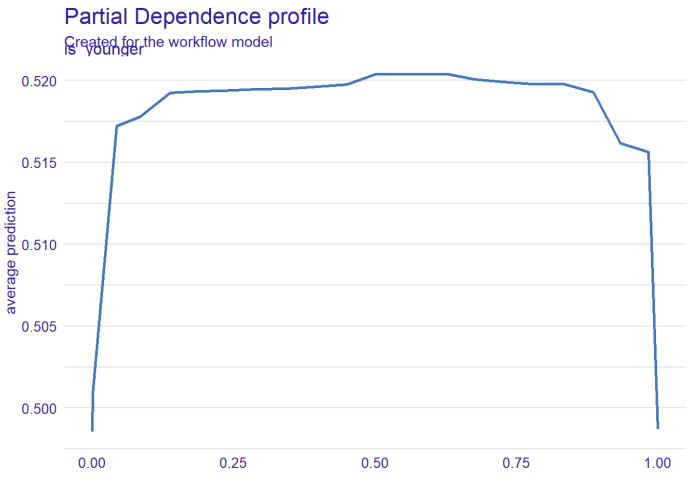
have_worked_withR	3.386	29.551
want_to_work_withNA	-1.249	0.287
want_to_work_withCobol	1.064	2.898
want_to_work_withCrystal	0.949	2.583
want_to_work_withVBA	0.924	2.521
want_to_work_withZephyr	0.910	2.485
have_worked_withCrystal	-0.859	0.423
want_to_work_withMATLAB	0.768	2.155
want_to_work_withPython	0.676	1.965
want_to_work_withClojure	0.610	1.841
have_worked_withZephyr	0.583	1.792
have_worked_withAda	-0.560	0.571
want_to_work_withLisp	0.554	1.739
want_to_work_withSQL	0.535	1.708
want_to_work_withF.	0.515	1.673
academic_prof	0.423	1.527
want_to_work_withErlang	0.416	1.516
want_to_work_withObjective.C	0.397	1.487
advanced_ed	0.396	1.485
want_to_work_withPerl	0.352	1.422
Industry_Fintech	-0.319	0.727
have_worked_withGo	-0.294	0.745
have_worked_withScala	-0.289	0.749
want_to_work_withRuby	0.284	1.329
want_to_work_withAssembly	-0.273	0.761
have_worked_withVBA	0.253	1.288
have_worked_withTypeScript	-0.247	0.781
want_to_work_withScala	0.203	1.226
have_worked_withGDScript	0.197	1.217
have_worked_withRuby	-0.194	0.824
want_to_work_withKotlin	0.192	1.212
Industry_Software.Development	-0.188	0.829
Industry_Other.	-0.170	0.844
have_worked_withElixir	-0.163	0.850
have_worked_withRust	-0.160	0.852

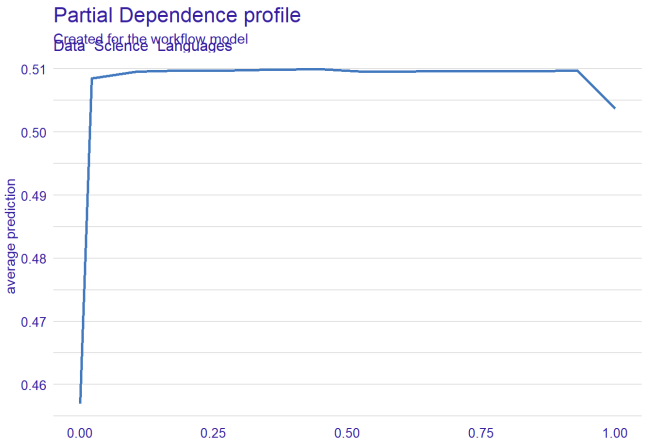
want_to_work_withJava	0.150	1.161
Industry_Government	0.139	1.150
want_to_work_withLua	0.138	1.148
have_worked_withCobol	-0.136	0.873
have_worked_withLua	-0.128	0.880
want_to_work_withFortran	0.125	1.133
have_worked_withAssembly	-0.123	0.884
is_younger	-0.090	0.914
higher_ed	-0.088	0.915
want_to_work_withApex	-0.071	0.931
want_to_work_withMicroPython	0.067	1.069
have_worked_withC..	-0.063	0.939
want_to_work_withZig	0.060	1.061
want_to_work_withJulia	0.044	1.045
have_worked_withMATLAB	-0.040	0.961
want_to_work_withBash.Shell..all.shells.	0.038	1.039
have_worked_withSQL	0.029	1.030
Industry_Healthcare	0.025	1.026
WorkExp	0.016	1.016
CompTotal	-0.016	0.985
want_to_work_withPowerShell	0.001	1.001

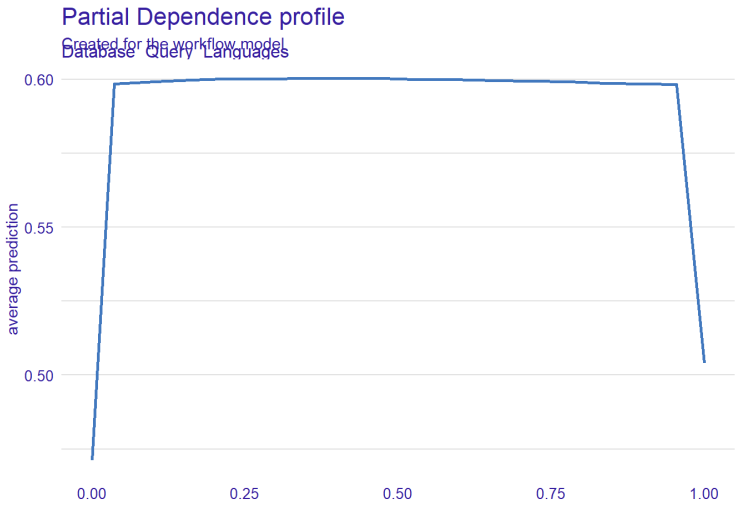
Appendix 3.1: PDP Charts








Partial Dependence profile
Created for the workflow model
Data_Science_Languages
0.51
0.50
0.49
0.48
0.47
0.46
average prediction
0.00
0.25
0.50
0.75
1.00


Partial Dependence profile
Created for the workflow model
Database_Query_Languages
0.60
0.55
0.50
average prediction
0.00
0.25
0.50
0.75
1.00