Studying the Impact of Job Descriptions on Data Science Salaries

Team Humilty

CHLOE NEO TZE CHING   ENG JING KEAT   GOH WEI LUN GLENN
KANEKO YOSHIKI   TAN WEI KEONG

2022-11-17

Abstract

Millions of workers worldwide are no longer satisfied with just a fair paycheck, instead, they wish achieve job satisfaction through meaningful career opportunities. In this project, we aim to uncover key insights in data science salaries by comparing models that effectively associate keywords in job descriptions to high paying data science roles. The data is cleaned by preparing job description text for analysis, and filtering the keywords in job descriptions with a minimum occurrence of 40. Least Absolute Shrinkage and Selection Operator (LASSO) is then applied to select the most important variables to be used in the model. The models used are Multiple Linear Regression (MLR), Partial Least Squares (PLS), Random Forest (RF), XGBoost (XGB), and Multivariate Adaptive Regression Splines (MARS), where their performances will be assessed in relation to one another. Across all models, MLR has the lowest Root Mean Square Error (RMSE) of 16.2. In other words, MLR is the best model that accurately predicts salary, and we identified keywords used in this model.

Contents

1 Introduction to the problem

1.1 Literature review

A rapid shift towards digitalisation of businesses has radically changed the employment landscape in Singapore, which means Singaporeans need to keep up with the changes if they wish to stay competitive at work (My Skills Future, 2021). Employers in Singapore are starting to place an emphasis on skills rather than education (Tan, 2021). New hires today are assessed not just by their qualifications and work history, but also by their soft skills as there are a variety of soft skills in demand (The Straits Times, 2021).

According to a new report by Instant Offices, 73% of Singaporean workers are dissatisfied with their jobs (Arora, 2022). When asked if they planned to change jobs over the following six months, 31% of respondents responded “yes” (Chong, 2022). Millions of workers worldwide are no longer willing to return home with just a fair paycheck. They prefer to know how well they are progressing towards a meaningful career, which is a wellness, freedom, security, and experience at work, so as to achieve job satisfaction. As a result, job seekers should take all these factors into consideration when applying for a job.

These factors can be further broken down into keywords in job descriptions. Keywords are crucial to job adverts because they allow job seekers to narrow their search related to a role, skill, or industry for suitable employment (Alexander, 2019). Suitable individuals are more likely to find the job post when hirers and recruiters add key terms and phrases that are relevant to a particular role. This increases the percentage of successfully matching job seekers with their ideal jobs.

1.2 Objective

As most employers will be impressed when they notice that the resume is customised, highlighting relevant skills and using certain keywords suited to that particular company or position, job seekers can use these findings to narrow down their job search based on their own preferences such as salary range or skills set. As such, this project aims to predict data science salary based on the keywords in job descriptions.

2 Dataset

Dataset: https://www.kaggle.com/datasets/nikhilbhathi/data-scientist-salary-us-glassdoor

2.1 Description of dataset

Import relevant libraries:

Below is a sample of our dataset:

```
## Rows: 742 Columns: 42
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (17): Job Title, Salary Estimate, Job Description, Company Name, Locatio...
## dbl (25): index, Rating, Founded, Hourly, Employer provided, Lower Salary, U...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Table 1: A sample of our dataset.

Avg Salary(K)	company_txt	Job Location	Age	Python	spark
72.0	Tecolote Research	NM	48	1	0
87.5	University of Maryland Medical System	MD	37	1	0
85.0	KnowBe4	FL	11	1	1
76.5	PNNL	WA	56	1	0

Avg Salary(K)	company_txt	Job Location	Age	Python	spark
114.5	Affinity Solutions	NY	23	1	0
95.0	CyrusOne	TX	21	1	0

This dataset has 41 variables, consisting of numerical, categorical, and text. Some variables are derivations from others, and as such we will not be using all 41 variables.

2.2 Exploratory data analysis

2.2.1 Data cleaning

Note: the dataset downloaded has already been cleaned by the owner, but we will do some additional cleaning and data preparation so that it is suited for our needs.

- Removing "\n" from job descriptions, cleaning job descriptions text, and creating a new variable to store lengths of job description texts:

2.2.2 Feature selection

Our project's focus is on job descriptions and their relationship with data science salaries. As such, we will only keep these two variables

2.2.3 Data visualizations

1. Histograms of salaries in thousands:

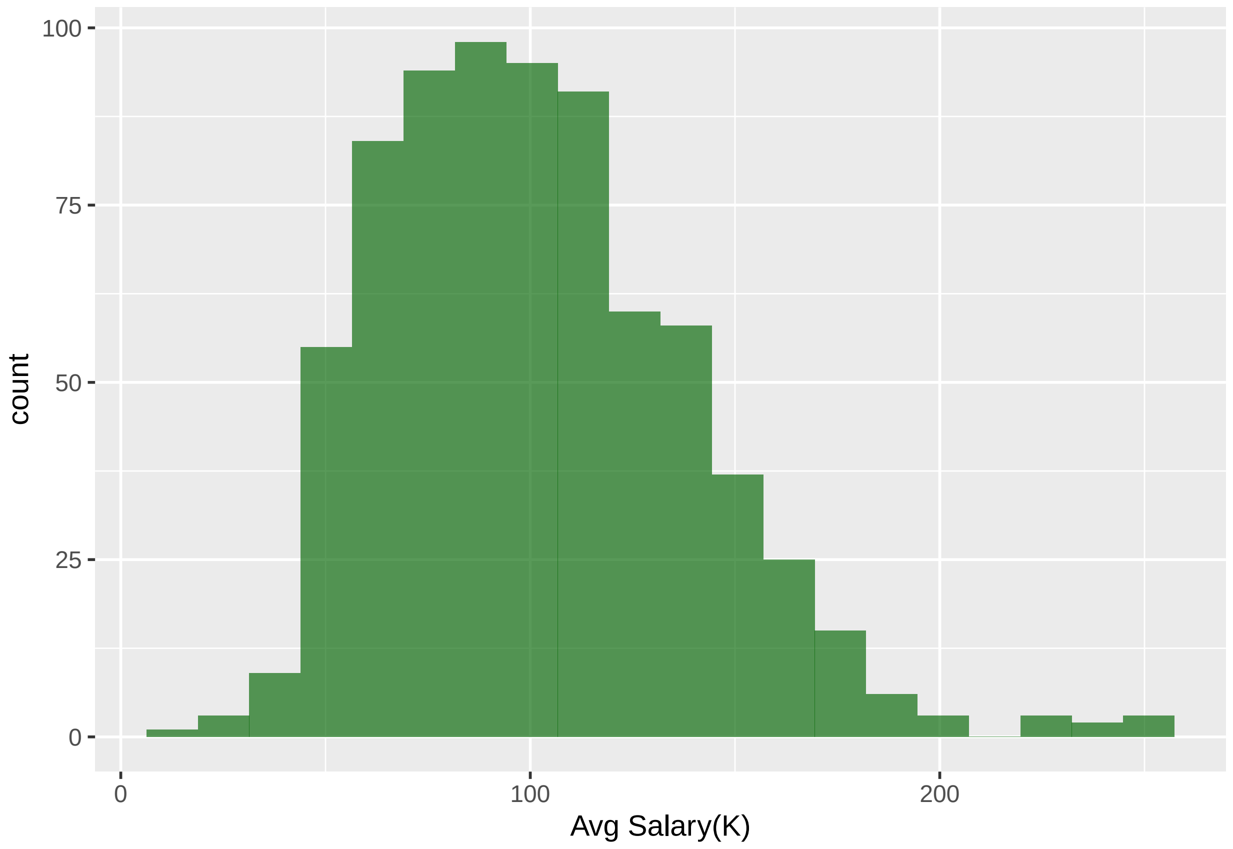

Figure 1: Histogram of salaries

From our histogram, we see that our average salary (in thousands) roughly follows a normal distribution. Hence we will predict the raw annual salary without further transformations.

2. Word cloud for job descriptions:

We visualise the job descriptions with a word cloud after removal of words whose total frequency is below 40, and stopwords such as "the", "he" etc. as these are common words that do not store any informational value in our analysis.

We set the minimum occurrence of words to 40, and show the 100 most common words above.

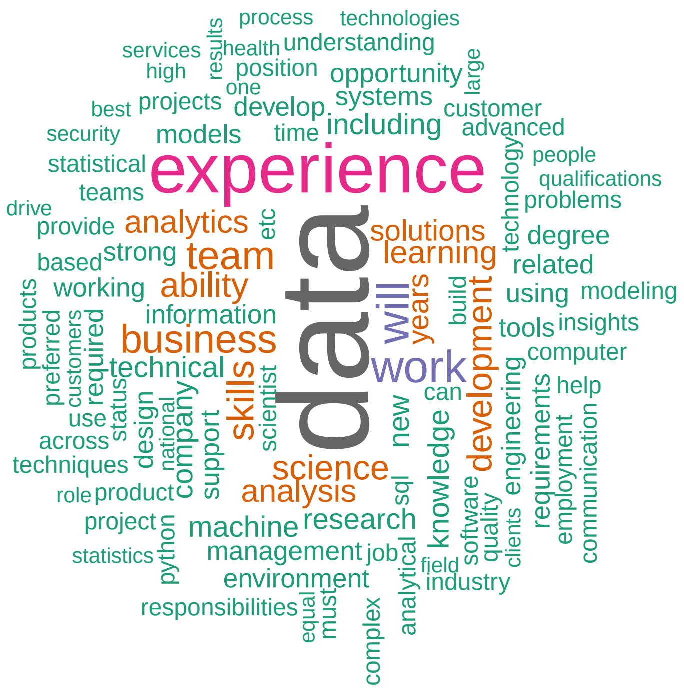

Figure 2: Job description word cloud

2.2.4 Feature engineering

Next we create a document-term matrix (DTM) for our job descriptions, setting minimum word frequency to 40.

```
Setting minimum word frequency to 40, we retain 1306 words out of the original 10314 words.
```

```
Our document-term matrix consists of 742 job descriptions against 1238 words present in our vocab after
```

Next, we create a new dataframe for our DTM.

Table 2: A sample of our DTM.

data	machine	python	analytics	science
11	1	1	0	1
7	2	2	7	1
18	2	1	3	1
6	4	1	0	7
8	2	1	1	2
16	1	1	1	0

3 Modelling

3.1 Feature selection using LASSO

We will use LASSO regularization to select features to prepare our data for three of our models: Multiple LInear Regression, Random Forest, and XGBoost. The last two models use their own methods of feature selection, so we will not use features selected by LASSO for those two models, but the entire dataset instead.

The LASSO procedure is as follows:

1. Use cross-validation to find the best value of lambda (approx. 0.359)
2. Store our variables selected by LASSO
3. Prepare our final dataset to be used for MLR, RF, and XGB

```
knitr::kable(head(D_final[, c('Y_salary', 'predictive', 'education', 'learn',
                              'machine', 'experience', 'analytics', 'support')]),
             caption = "A sample of our final dataset.")
```

Table 3: A sample of our final dataset.

Y_salary	predictive	education	learn	machine	experience	analytics	support
72.0	0	2	0	1	3	0	0
87.5	3	1	0	2	7	7	2
85.0	2	0	0	2	7	3	1
76.5	1	0	1	4	9	0	1
114.5	0	0	1	2	1	1	3
95.0	0	0	0	1	5	1	0

We also prepare our training and test data to be used, and the dimensions are as follows:

```
Dimensions of the training set are 585 355
```

```
Dimensions of the test set are 157 355
```

From this, we can see that what started with over 10,000 unique words in our job descriptions has been narrowed down to 354 words.

3.2 Models

3.2.1 Multiple Linear Regression

The first model we use is Multiple Linear Regression, using the 354 features selected by LASSO.

```
[1] 16.20561
```

```
[1] 9.322398
```

Our MLR model's RMSE value is 16.20561, the lowest we will achieve in this project.

Below is a bar plot of the top 20 variables with the highest absolute coefficients. The sign of their coefficients reflect a positive or negative relationship with the predicted salary.

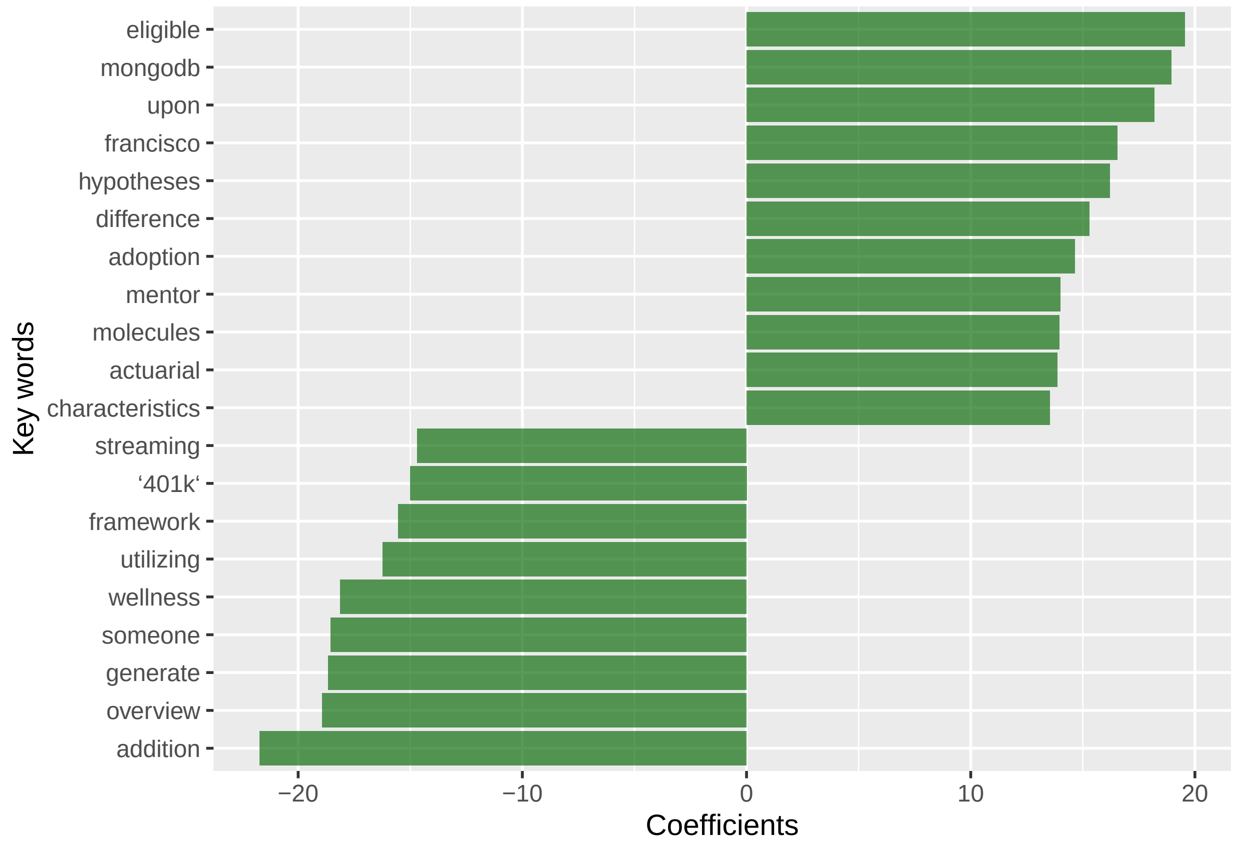

Figure 3: Top 20 variables according to absolute coefficients

This plot shows that the presence of words such as “eligible” and “molecules” in the job description will result in a higher predicted salary, whereas words such as “streaming” and “wellness” in the job description will lower the predicted salary.

3.2.2 Random Forest

The next step is to visualize our data using random forest decision trees. We first train the data to obtain its optimal hyperparameters. We then perform grid search for the optimal hyper-parameter values to minimize the out-of-bag error.

The optimal values of the hyper-parameters are:

	mtry	splitrule	min.node.size
5	178	extratrees	5

So we retrain the model with the selected hyper-parameters, and fit our Random Forest model on the training data set.

Table 5: A sample of our Random Forest prediction results.

predictions	Y_salary	education	learn	machine	experience	analytics	support
97.92145	73.5	0	0	2	0	0	0
96.87724	85.0	0	0	1	5	0	0
73.43357	47.5	1	2	0	2	4	3
105.40680	96.0	0	0	2	5	1	0
117.03311	121.0	0	1	4	4	5	0
83.21587	106.0	0	0	0	4	4	0

The graph below shows the data between Actual Average Salary vs. Predicted Average Salary

Our RMSE for the Random Forest model is shown below:

```
MAE: 15.39
```

```
RMSE: 21.59
```

The bar graph below shows the top 20 most important variables in this prediction. Based on the bar graph, the top 5 most important variables are “machine”, “give”, “expression”, “predictive”, and “groups”.

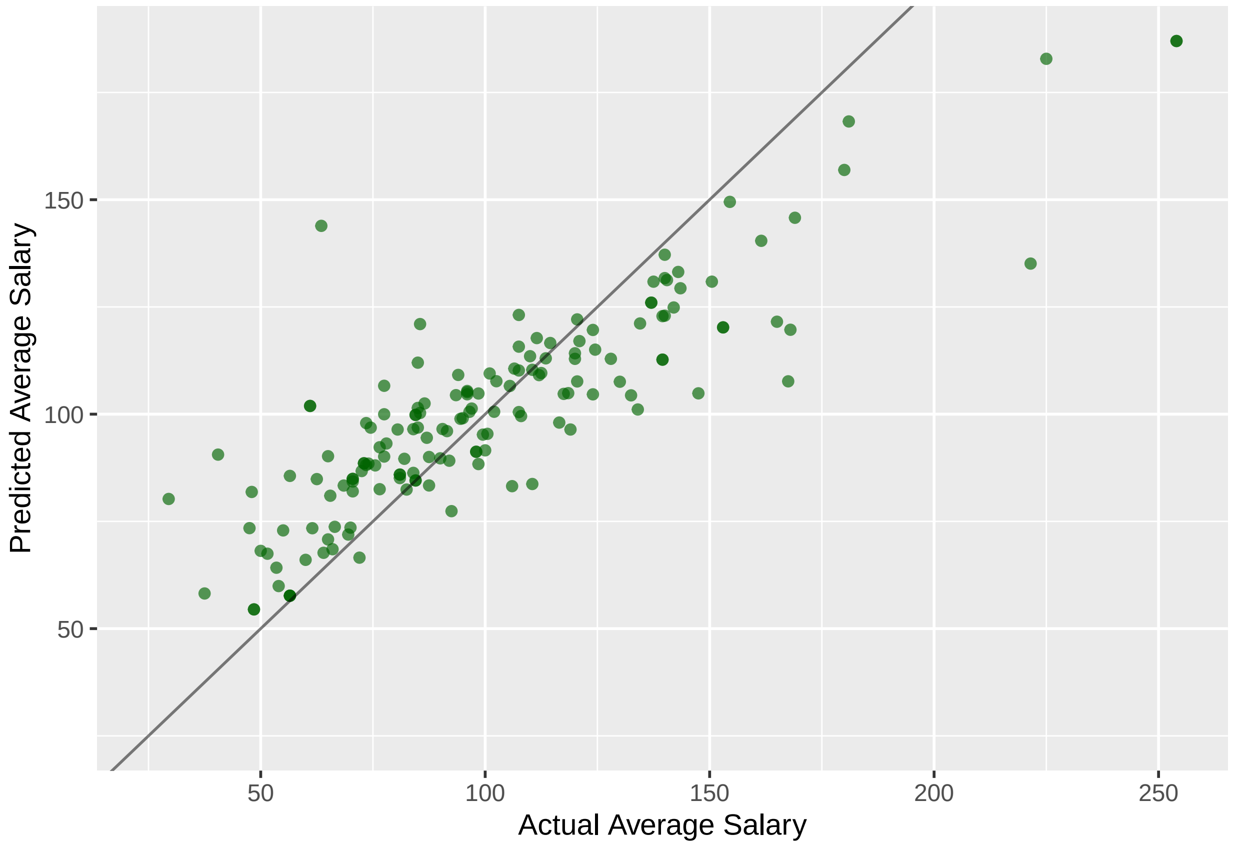

Figure 4: Actual average salary vs. predicted average salary

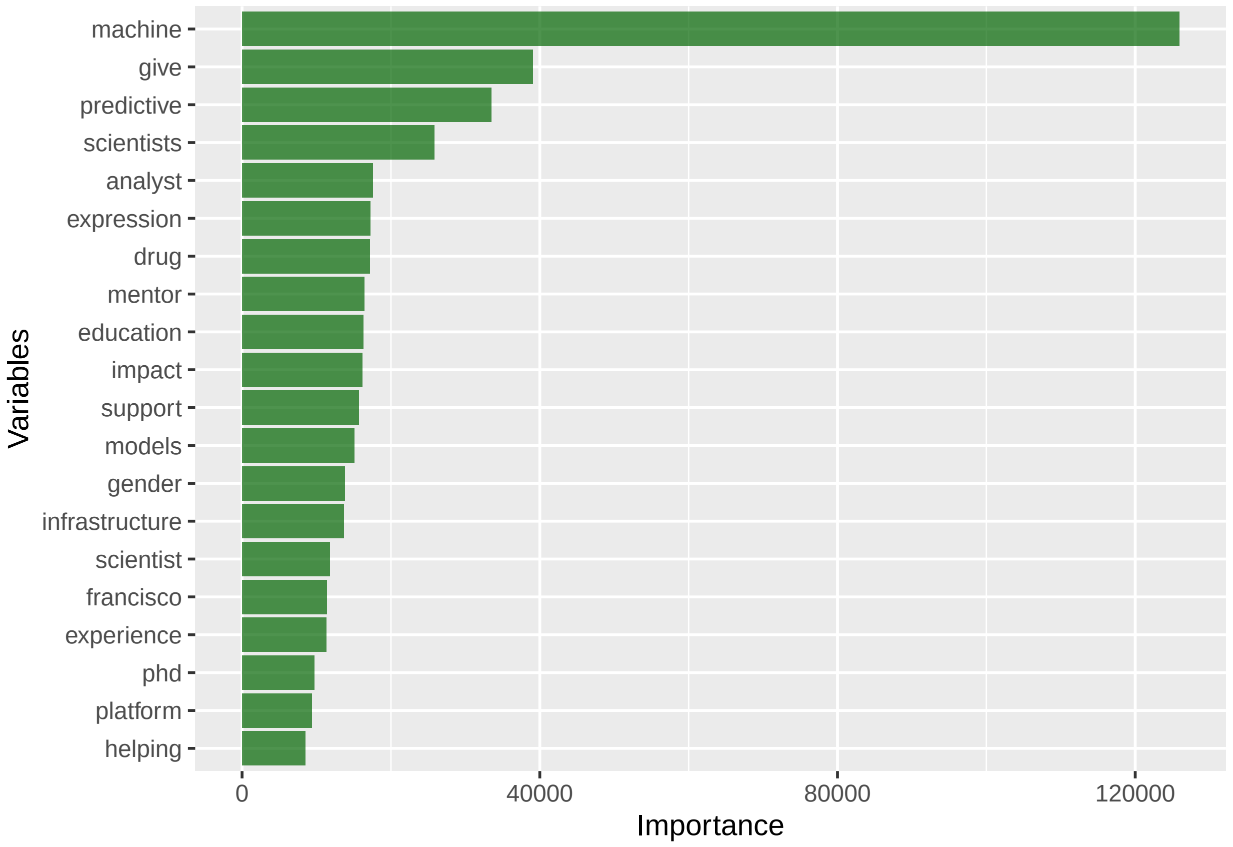

Figure 5: Random forest top 20 most important variables

3.2.3 XGBoost

Like random forests, gradient boosting machines does classification based on decision trees. However, while random forest builds an ensemble of deep (i.e complex) trees that are independent of one another, gradient boosting machines build shallow trees sequentially, where each tree learns and improves from the previous tree. This means that one would start with a weak model and sequentially boost its performance by allowing each new tree to focus on training data where the previous tree had the largest errors in prediction (or residuals). This is done by fitting each tree in the sequence according to the residuals of the previous tree.

Moreover, it computes the second-order gradients, i.e. second partial derivatives of the loss function, which provides more information about the direction of gradients and how to get to the minimum of our loss function while gradient boost uses the loss function of simple decision tree model as a proxy to minimize the error of the overall model. In addition, it uses advanced regularization (L1 and L2), which improves model generalization.

Each weight in all the trees would be multiplied by the learning rate in an XGBoost model, such that

$$w_j = \text{Learning Rate} \times \frac{\sum_{i \in I_j} \frac{\partial Loss}{\partial(\hat{y}=0)}}{\sum_{i \in I_j} \frac{\partial^2 Loss}{\partial(\hat{y}=0)^2} + \lambda}$$

where I_j is a set containing all the instances ((x, y) data points) at a leaf, and w_j is the weight at leaf j with regularization from the λ constant.

We create our XGBoost model from the caret library, using hyperparameters as shown below:

As shown above, we have used these hyperparameters:

1. *gamma*: Pseudo-regularisation hyperparameter that controls the complexity of each tree.
2. *nrounds*: Number of decision trees in the final model
3. *eta*: Learning rate; determines the contribution of each tree on the final outcome and also how quickly the algorithm goes down the gradient descent.
4. *max_depth*: Depth of each tree
5. *min_child_weight*: Minimum number of observations in terminal nodes; controls complexity of the trees
6. *colsample_bytree*: subsample of columns used for each tree (repeated for every tree)
7. *subsample*: subsampling ratio of training data for growing trees to prevent over-fitting

The hyperparameters were tuned using 5-fold cross validation and grid search to find the best model, and we arrived at the optimal values for the hyperparameters.

Below we compute our RMSE.

```
MAE: 8.04
```

```
RMSE: 16.59
```

Overall, XGBoost gave an RMSE value of 16.59.

Below we extracted the 20 most important features (words) from the XGBoost model.

```
Attaching package: 'xgboost'

The following object is masked from 'package:dplyr':

    slice
```

Table 6: Some of our 20 most important features.

Feature	Gain	Cover	Frequency
machine	0.1251407	0.0105748	0.0093227
give	0.0291924	0.0036677	0.0021936
education	0.0287696	0.0024835	0.0024678
analyst	0.0266963	0.0013853	0.0019194
phd	0.0261783	0.0075795	0.0054840
scientists	0.0218101	0.0076164	0.0049356

Plotting our top 20 most important variables:

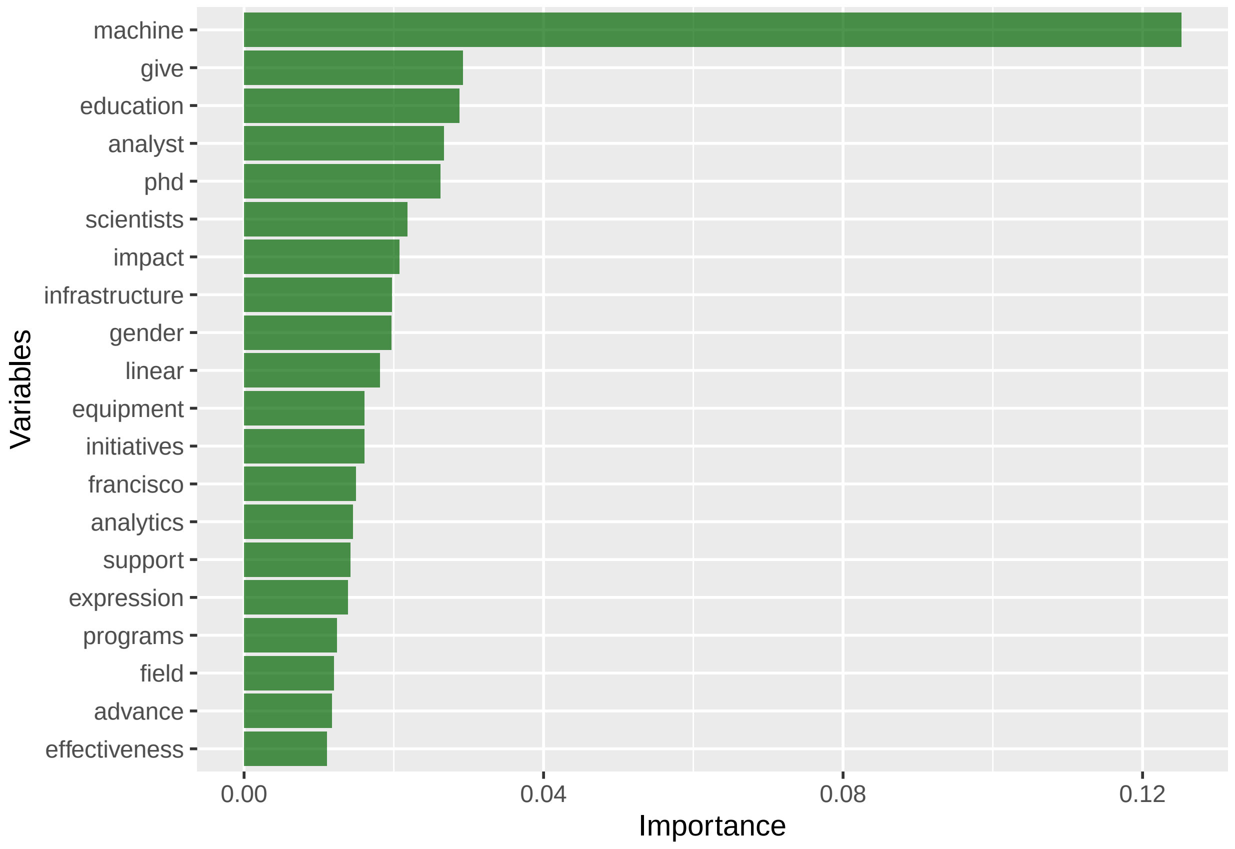

Figure 6: XGBoost top 20 most important variables

3.2.4 Multivariate Adaptive Regression Splines (MARS)

We used multivariate adaptive regression splines (MARS) (Friedman 1991) model here, it is an approach that automatically generates a piecewise linear model that serves as an understandable stepping stone into non-linearity after learning the notion of multiple linear regression.

By evaluating cutpoints (knots) similar to step functions, MARS offers a practical method to capture the nonlinear relationships in the data. This method evaluates every data point for every predictor as a knot and builds a linear regression model using the candidate feature(s).

Consider non-linear, non-monotonic data where $Y = f(X)$. The MARS method will initially search for a single point within a range of X values where two distinct linear relationships between Y and X provide the lowest loss. The outcome is referred to as a hinge function $h(x-a)$, where a is the cutpoint value.

For example, if $a = 1$, our hinge function is $h(x-1)$ such that the linear models for y are:

$$y = \begin{cases} \beta_0 + \beta_1(1-x)\,, & x < 1 \\ \beta_0 + \beta_1(x-1)\,, & x > 1 \end{cases}$$

After the first knot is identified, the search for a second one begins, and it is discovered at $x = 2$. Now the linear models for y are:

$$y = \begin{cases} \beta_0 + \beta_1(1-x)\,, & x < 1 \\ \beta_0 + \beta_1(x-1)\,, & 1 < x < 2 \\ \beta_0 + \beta_1(2-x)\,, & x > 2 \end{cases}$$

This process is repeated until several knots are identified, leading to the creation of a highly non-linear prediction equation. Even if using a lot of knots could help us fit a particularly excellent relationship to our training data, it might not perform well to unseen data. Once all of the knots have been found, we may systematically eliminate knots that do not significantly improve predictive accuracy. This is pruning process, and we may use cross-validation to determine the optimal number of knots.

We will use the following packages. First of all, we divided the dataset into training dataset and test dataset:

```
Dimensions of the MARS training dataset are 512 1239
```

```
Dimensions of the MARS test dataset are 230 1239
```

MARS model have two hyperparameters: the maximum degree of interactions and the number of terms retained in the final model. To achieve the optimal combination of these tuning parameters, we must conduct a grid search that minimize the error of prediction.

Here, we built up a grid with 30 different combinations of interaction complexity (degree) and the number of terms to include in the final model (nprune).

We performed required grid search by using 10-fold cross-validation:

- Our chosen parameters.

	nprune	degree
3	11	1

degree	nprune	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
1	11	31.87122	0.3140381	25.10864	2.810016	0.0945591	2.344299

The backwards elimination feature selection process used in MARS models seeks for reductions in the generalized cross-validation (GCV) estimate of error when each additional predictor is introduced to the model. The variable importance is based on this overall reduction. MARS effectively accomplishes automated feature selection since it will automatically include and remove variables throughout the pruning phase.

After pruning, a predictor's significance value is 0 if it was never used in any of the MARS basis functions in the final model. There are only 11 features have importance values greater than 0, whereas the other features all have importance values of zero since they were excluded from the final model.

We also kept track of how the residual sums of squares (RSS) change when terms are added. However, we noticed that there is no much difference between these two measures.

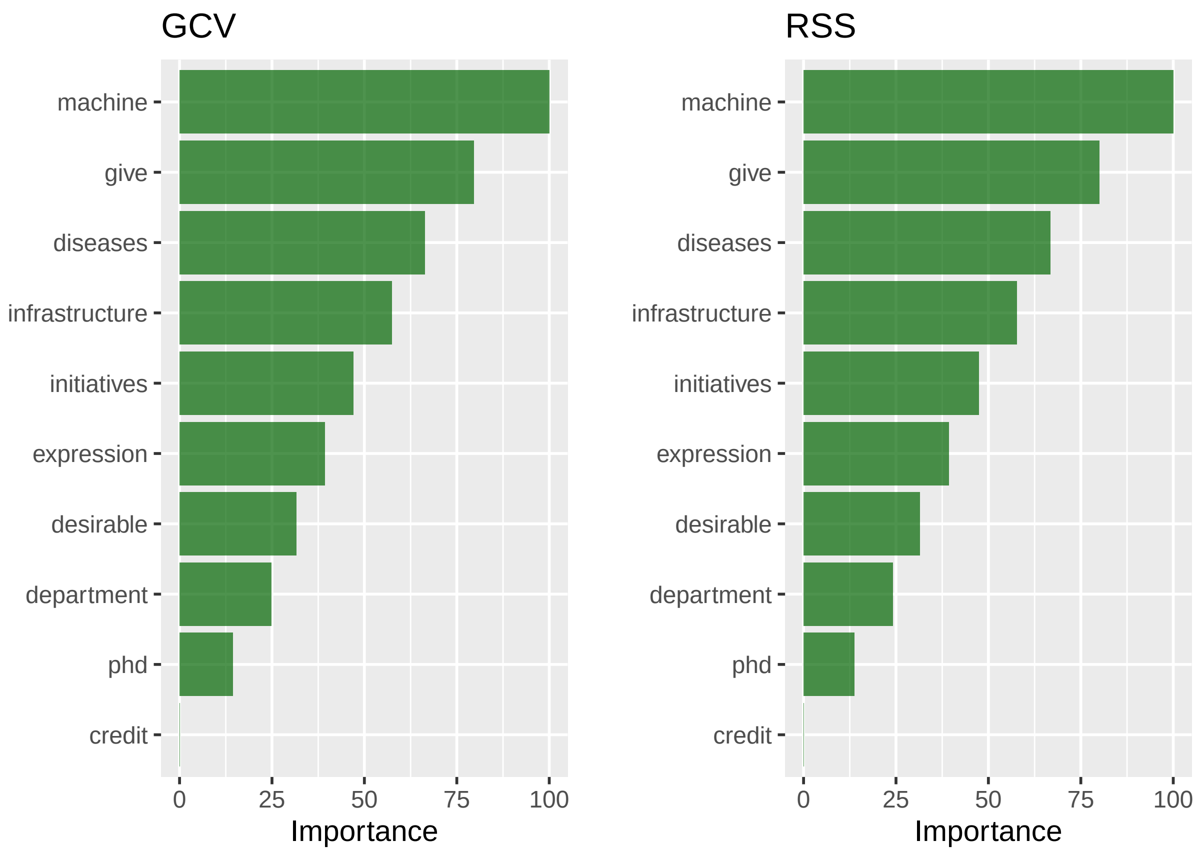

Figure 7: MARS top 20 most important variables

We used the optimal hyperparameters to train the model and then calculated the RMSE:

`MAE: 23.22`

`RMSE: 30`

3.2.5 Partial Least Squares (PLS)

Partial Least Squares (PLS) is a common technique to analyse relative importance when the data includes more predictors than observations. It is an useful dimension reduction method which is similar with principal component analysis (PCA).

We do a regression against the response variable inside the narrower space created by mapping the predictor variables to a smaller set of variables. The response variable is not taken into account during the dimension reduction process in PCA. PLS, on the other hand, seeks to select newly mapped factors that best describe the response variable.

Below are the required packages. We divided the dataset into training dataset and test dataset first:

The hyperparameter for PLS model is the number of components used in the model (ncomp) .We conduct a grid search that minimize the prediction error to achieve the optimal hyperparameter. The grid search was conducted by 10-fold cross-validation:

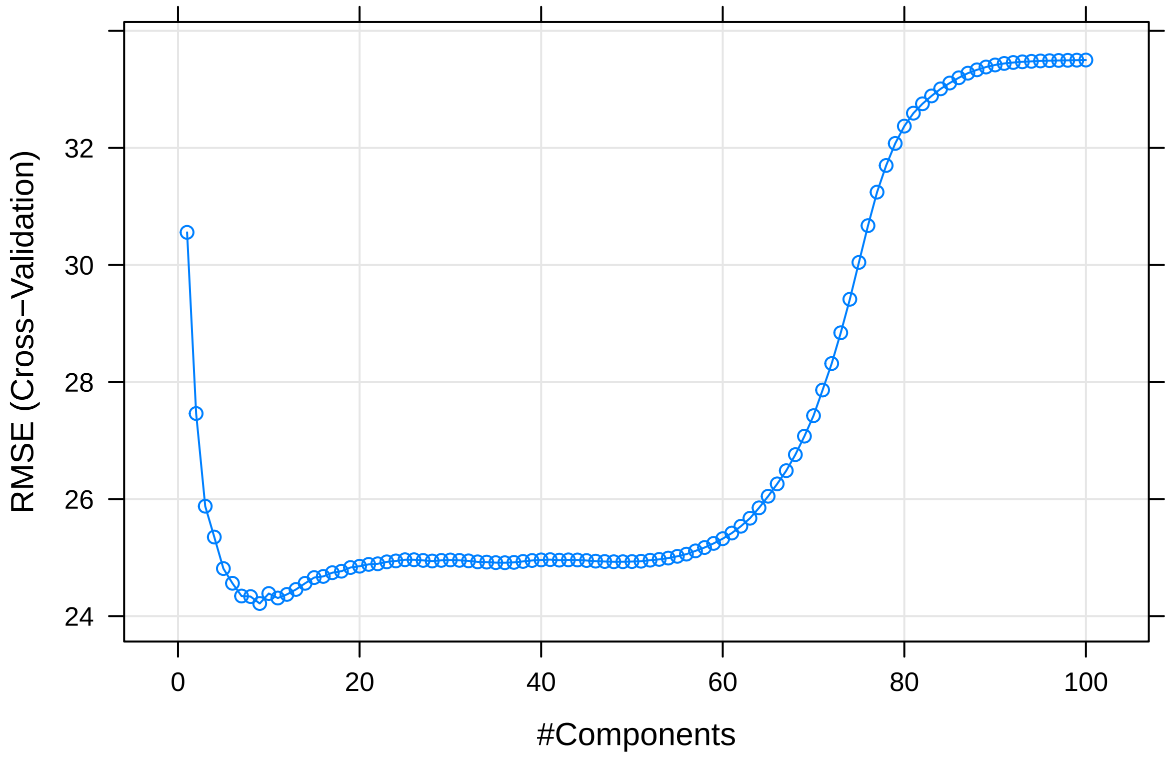

Figure 8: Number of components in PLS model vs. RMSE

We used the optimal hyperparameter to train the model and calculated the RMSE as well:

```
MAE: 16.63
```

```
RMSE: 23.88
```

The barplots below show that ‘addition’, ‘generate’, ‘hypotheses’, ‘framework’ and ‘characteristics’ are positive predictors, while ‘streaming’, ‘francisco’, ‘difference’, ‘mongodb’ are negative predictors:

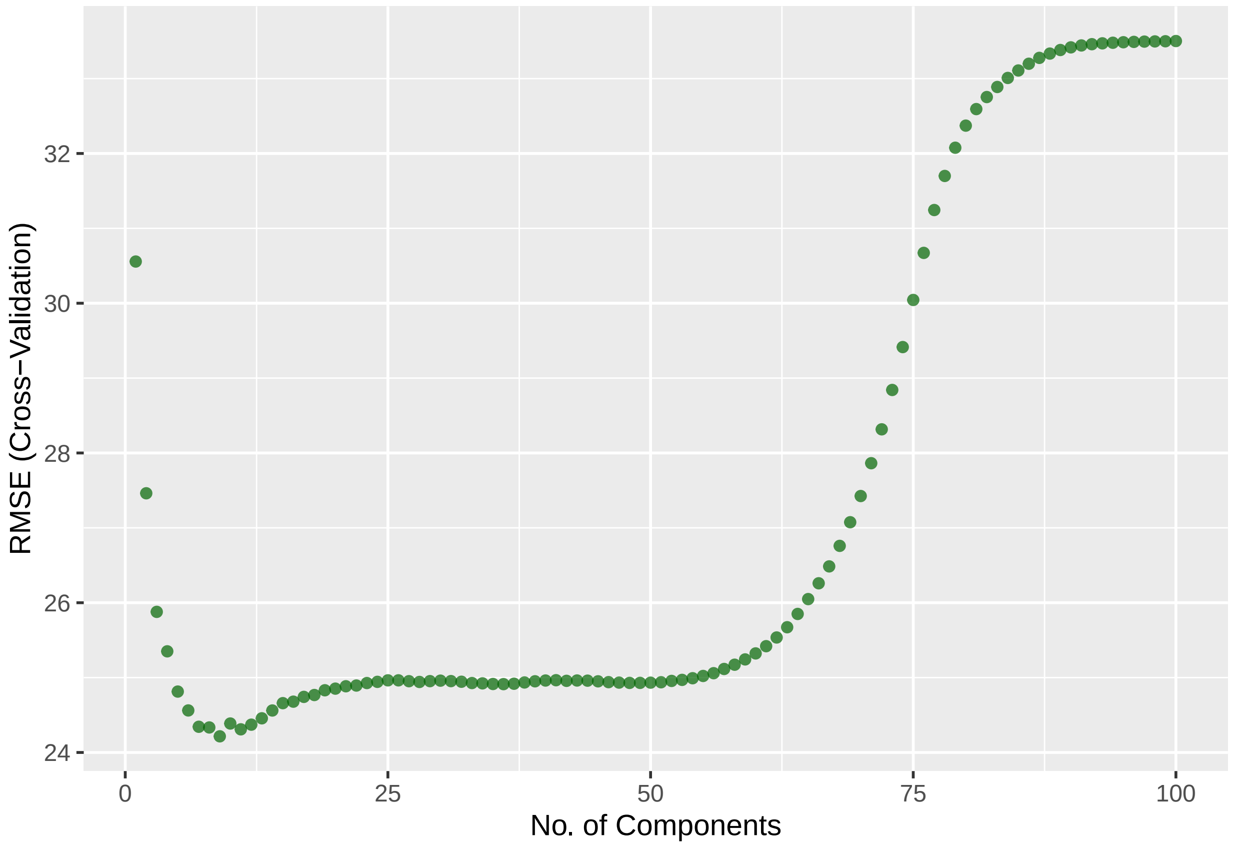

Figure 9: Number of components in PLS model vs. RMSE

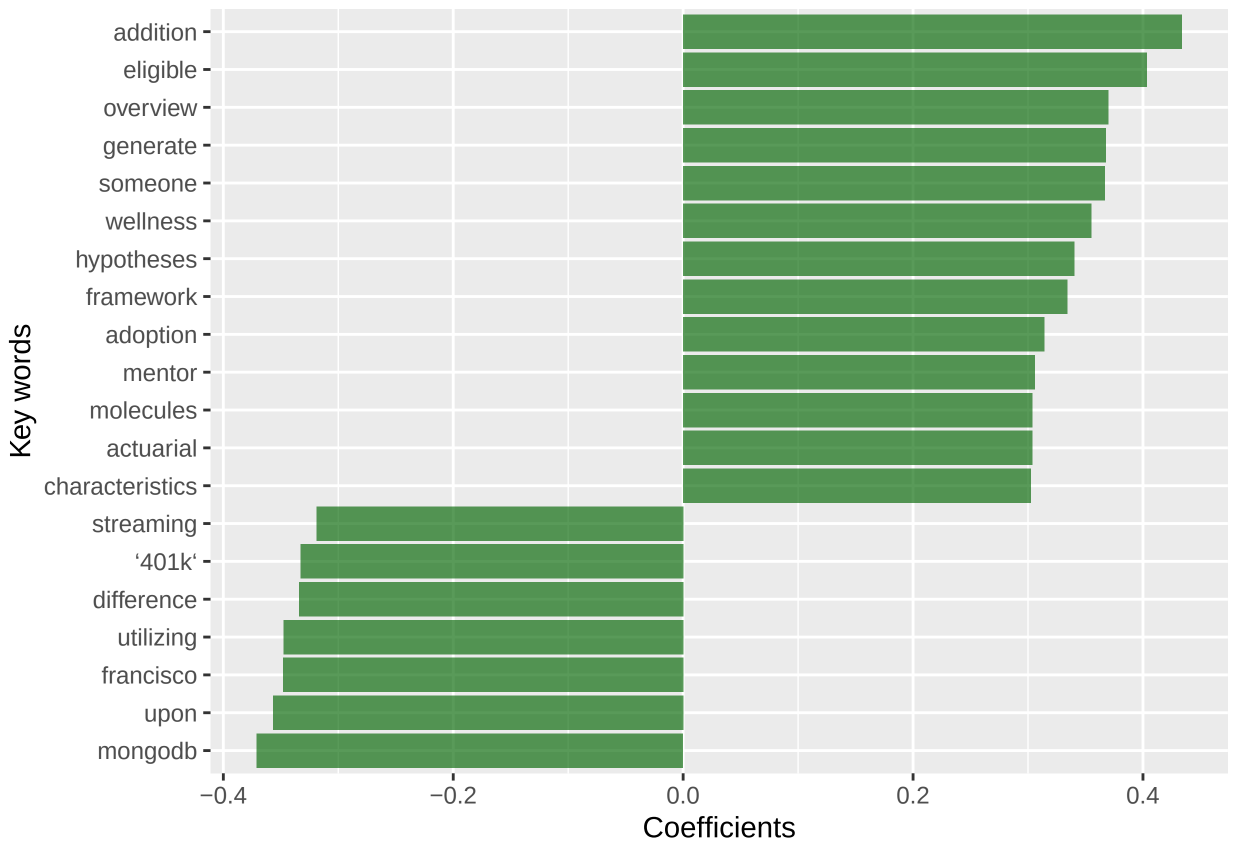

Figure 10: PLS top 20 most important variables

3.3 Summary of results

Table 9: Accuracy of models.

Model	RMSE
MLR	16.20561
Random Forest	21.62234
XGBoost	16.59140
MARS	30.00250
PLS	23.87552

4 Conclusion

Recommendation engines are a commonly found solution applied to job search portals, such as the Singapore government's national jobs portal, MyCareersFuture. However, more can be done to bridge the gap between job seekers and their desired careers. In this project, we have produced models that predict the expected average salary earned given a job description. Our best performing model, Multiple Linear Regression, can be used to help workers set expectations of salary based on keywords they value as important in search of a job. This will help job seekers focus on searching for their desired job role rather than focus on maximizing salary earned, which will hopefully increase job satisfaction. The model also identifies key terms such as "eligible", "mongodb", "upon", "francisco", and "hypotheses". Some of these keywords may not seem to make sense, and understanding the importance of these words is unclear. Some of these words, on the other hand, give insight into areas that job seekers can focus on, be it upskilling (for example, learning MongoDB), or narrowing their job search to sectors such as actuarial science or molecular chemistry in order to maximize their potential salary.

Based on RMSE, our best model uses Multiple Linear Regression, and it has identified key terms that have a significant impact on data science salary. However, Multiple Linear Regression does not identify the same important features (words) as our other models. Further research is required to better understand the difference between models and why they identify vastly different features as important.

5 References

1. Alexander, L. (2019). The importance of keywords in job ads. SEEK. Retrieved November 16, 2022, from https://www.seek.com.au/employer/hiring-advice/the-importance-of-keywords-in-job-ads
2. Arora, P. (2022, August 29). What's keeping Singapore employees unhappy at work? - ETHRWorldSEA. HR News Southeast Asia. Retrieved November 7, 2022, from https://hrsea.economictimes.indiatimes.com/news/employee-experience/whats-keeping-singapore-employees-unhappy-at-work/93833788
3. Chong, C. (2022, May 17). Nearly 1 in 3 workers in S'pore plans to change employers in first half of 2022: Survey. The Straits Times. Retrieved November 4, 2022, from https://www.straitstimes.com/singapore/jobs/nearly-1-in-3-workers-in-spore-plan-to-change-employers-in-first-half-of-2022-survey
4. My Skills Future. (2021, June 21). 5 Crucial Skills You Need to Remain Employable in the Wake of Covid-19 | Myskillsfuture.gov.sg. MySkillsFuture. Retrieved October 14, 2022, from https://www.myskillsfuture.gov.sg/content/portal/en/career-resources/career-resources/education-career-personal-development/5-crucial-skills-you-need-to-remain-employable-during-covid.html
5. The Straits Times. (2021, December 22). It's a match: How skills-based hiring fits in the future of work. The Straits Times. Retrieved October 14, 2022, from https://www.straitstimes.com/singapore/jobs/its-a-match-how-skills-based-hiring-fits-in-the-future-of-work
6. Tan, E. (2021, April 14). S'pore employers prioritise skills over education, experience: LinkedIn survey. The Straits Times. Retrieved October 14, 2022, from https://www.straitstimes.com/singapore/jobs/singapore-employers-prioritise-skills-over-education-experience-linkedin-survey