What Are The Technical Skills that Affect the Annual earnings in the US IT Industry, and how Machine Learning Algorithms is Ranking The Important Skills

Chenke Bai
University of Colorado Boulder
2021-2022

Thesis Advisor

Brian C. Keegan
Department of Information Science

Defense Committee

Lecia Barker Department of Information Science	Richard Mansfield Department of Economics	Brian C. Keegan Department of Information Science

1. Abstract

This study uses multiple different machine learning models to investigate how different skills are affecting the earnings in the IT field. To investigate that we decided to take 7 different machine learning approaches to help identify the factors that affect IT workers' earnings by applying feature importance analysis to the best performing machine learning model, based on the evidence we saw from exploring similar literature on the internet. The topic of study that researches skill-based salary analytics and prediction in the most important skill sets that affect people's salary in the US IT field is under-researched. The limited study in IT skill and respective salary rate provides us a great opportunity to apply training on selected machine learning models that can be then used to predict IT workers' salaries based on their skill sets. The data used in this study is StackOverflow 2021 Developer Survey data. This company conducts these surveys annually to gather users' information so StackOverflow can improve its service. Our choice of computational tool is Python 4.0, the machine learning model was built using

the Scikit learn library. By using Scikit learn we can adopt 7 machine learning(ML) algorithms and UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction technique into this study. After in-depth analysis, we got the highest testing accuracy of 80%, and around 50% of the salaries were predicted. At the end of this study, we discovered that even though technical IT skills are very important in affecting peoples' earnings, years of professional technical experience always outweigh a specific technical skill. Furthermore, if we wish to look at skills to see how they affect the salary, we need to look at multiple skills together as a whole. The findings of this study can be used for developing a skill-based salary calculator that allows future IT workers to project their potential salary. By entering different skill combinations and their experience they can see which skills are more likely to yield a higher salary.

2. Introduction

The growing population of college graduates along with the growing demands of the job market has set the bar even higher for many IT workers in the United States. Many industries have developed complex evaluation processes that involve rigorous evaluations with a focus on assisting in hiring the best candidates for the company. Thus many companies have adopted the machine learning(ML) technique in their recruitment processes. The application of the ML model for automated talent detection helps the company locate the most professional profiles in the field that the company lacks. This has created an opportunity for companies like Linkedin and Indeed to develop a set of analytical tools to help their users better plan and prepare themselves with regard to professional IT skills to gain better job opportunities and higher earnings.

In the following section, we discuss the previous studies that have been done in researching skill-based job classification using ML models like a decision tree to classify skills with related job positions. Some other researchers have used the Linear Regression model to predict salary based on a variety of skills. We also discuss research done by a group of researchers from Spain, who have shown us that by using a combination of ML algorithms like linear regression and SVM(Support Vector Machine) with feature importance analysis they were able to identify the programming languages that correlate to a high salary.

In the methods section, we first used UMAP analysis to see how different IT skills are clustered together to understand which groups of skills often occur together in this dataset. After gaining an understanding of how different skills are clustered together, we selected 7 different ML models for the study. In each round of the analysis we tested each of the six models and selected the one with the best test score. We then applied feature importance analysis to the ML model with the best test score to examine what are the top 25 most important skills. Upon the completion of the ML section, we applied

basic statistical analysis and statistical testing to 6 general skillsets to analyze how the understanding(experience) of each skill set affects people's salaries.

Last, in the result section, we discussed our analytical findings from each method, and in the discussion section, we discussed how valid these analysis methods are for our research, and what can be done in the future to improve the research in this field.

Previous research by Martín et al. has demonstrated ML models are useful for predicting salary based on skills. Moreover, feature importance analysis combined with ML can help us understand which skills have the most impact on salary prediction. Based on this understanding, we hypothesized that we could use these tools to identify which IT skills are the most important for predicting salary.

3. Literature Review

Below we analyze the BLS Working Paper On Job Requirements, Skill, and Wages, Predicting employee expertise for talent management in the enterprise, Applicability of clustering and classification algorithms for recruitment data mining, Challenge: Processing web texts for classifying job offers, in Semantic Computing, Predict the emergence: Application to competencies in job offers, in Tools with Artificial Intelligence, and Salary Prediction in the IT Job Market with Few High-Dimensional Samples: A Spanish Case Study.

3.1. Salary Prediction based on Skills and Skill Clustering

In the study published by the U.S. Bureau of Labor Statistics in March 2019, authors Matthew Dey and Mark A. Loewenstein conducted their study based on two data sets. One from O*Net contained information on job attributes like required skills with the occupational wages. Their O*Net data set was similar to ours, which also highlights survey respondents' skills and their reported salaries. Another dataset they used was from Occupational Employment Statistics(OES) and included employment information of a specific occupation in the states and its corresponding wages. From these two data sets, they were able to combine job attributes from the O*Net data set and occupational wage and employment information from OES.

During the first step of their research process, Dey and Loewenstein conducted factor analysis on the job categories provided by the O*Net data set, which allows them to connect between job categories and their corresponding job skill requirements and job attributes. In the second step, they used the regression tree model to aggregate detailed job occupations into broader aggregates. They did this aggregation because each aggregated group of occupations had similar information regarding skill requirements and wages. We believed we could apply

this technique of grouping to our study, in which we could group similar skills together and redefine them as broader skill sets, which would be a good way for us to understand what kinds of skill sets are associated with what kinds of wages.

From their regression analysis conducted on the O*Net data set, they discovered that the result of regressing occupational wages against the O*Net variables other than the educational features yielded an R-Square score of 0.933. This R-Square score is not a surprise since the O*Net data is highly correlated. According to Dey and Loewenstein, there are two reasons why O*NET variables are highly correlated. First, many variables O*Net measures are similar across the board. Second, skills, job activities, and working conditions may be distributed separately across jobs. But a single job may require a combination of a variety of skills. A particular skill may have significant value when combined with other sets of skills but have little value by itself. This discovery provided us with important insight when we conducted our research, in which we were more careful about making an assumption about one specific skill and how that skill influences people's salary. Dey and Loewenstein pointed out that a particular skill rarely has significant value on its own. From their research, we concluded that connecting job requirements and occupation information with wages was a good way to gain an understanding of the data set. (Day & Loewenstein, 2019) But our approach to connecting occupation skills and salary was different since we were using a different data set. In addition, we were planning on generating a visualization so we decided to use the UMAP technique for the regression model used by Dey and Loewenstein.

3.2. Research Done in Skill Classifications and Salary Predication in IT Fields

Sivaram and Ramar have pointed out that the construction of the decision tree model is generally simple and fast, and normally it has good accuracy. In addition, it is also very good for exploratory analysis. Since our research was an exploratory analysis, we thought it would be reasonable to adopt the decision tree model. (Sivaram & Ramar, 2010) Generally, the decision tree algorithm creates a node, then applies the attribute selection method to determine the best splitting standard and creates a node named by that attribute. In our research, we used a decision tree as a predictive model to predict the salary based on given skill sets. Generally, the decision tree takes the training data sets as input.

A similar study has also been done by F. Amato et al. in their paper: "Challenge: Processing Web Texts for Classifying Job Offers." In Amato et al.'s study, their goal was to apply and compare the methods of classifying online labor market data using explicit rules, machine learning, and LDA-based algorithms.

The data of "Web job offers is collected from 12 heterogeneous sources against a standard classification system of occupations."

In their analysis, they constructed automated algorithms that grouped job offers using supervised machine learning(ML) algorithms Linear SVM combined with expert labeling. During the application of SVM, Amato et al. turned each job title into a vector. So the whole set of job offer titles was turned into a matrix, the job title became a row and the columns became the occurrence of stemmed words extracted from each job offer. They then used this matrix data to train the SVM classifier. Although this research had a very different goal to ours, their approach of vectorizing the words(in our case words were skills) gave us a good idea of what we should do before training text data on a machine learning model. (Amato et al., 1970) Other researchers have done a similar study in finding the relationship between skills and jobs. In Yacine Abboud, Anne Boyer, and Armelle Brun's study, they reverted the pattern recognition to predict the skill emergence in the job market. The goal of their research was very similar to ours in essence, since we were all trying to predict our findings based on the skills people possess. Although they investigated a similar field, their approach to their goal was very different from ours, their methods were very different from those we planned to use, as we were planning on using the ML model to investigate the relationships between skills and salary. Theirs was to use job information and web text to predict emerging skills. But in their conclusion section, they highlighted the need for the type of research our project was working on (Abboud et al., 2016), "The emergence of new technology created a huge need for reactivity and anticipation"(Abboud et al., 2016). In our research, we explored which IT skills were more vital in dictating wages in the IT field. Our findings could be an important supporting argument for one of the conclusions Martín et al. proposed in their article(Martín et al., 2018).

To extend on previous research done by other analysts, Martín et al. analyzed 4000 job offers in the Spanish IT recruitment portal called Tecnoempleo. The data collection process used machine automated data collection methods which use Python based web crawlers to gather information from a recruitment website. Compared to our data collection, Martín et al. had more control over what kinds of raw data they wanted to access. When it came to feature selection Martín et al. did something very different compared to our approach. They used the automatic feature selection which used the filter method X-MIFS. This method was designed for selecting features based on the maximization of the mutual information (MI) between features and output variables. The advantage of this feature selection method is scalability to a high dimensional data set. It works well with high dimensional data and it does not limit the choice of machine learning(ML) model. (Brunato & Battiti, 1970) Compared to our feature selection

method their's was more robust and more machine-driven. Our feature selection was generally less work because the survey data set had already labeled many useful features for us. In Martín et al.'s method section, they used ML models like Linear models (LM), Logistic regression (LR), K-nearest neighbors (KNN), Multi-layer perceptrons (MLP), Support vector machines (SVM), Random forests (RF), and Adaptive boosting with decision trees (AB). In our research, we decided to use many common ML algorithms similar to what Martín et al. used. For example, we planned on using LR, SVM, and RF models for salary prediction. Martín et al.'s ML analysis results showed highly demanded IT skills are .NET which is 16.9%, and Java 16.7%. Other IT skills included SQL at 11.7% and JavaScript at 9.3% or PHP at 8.8%. We expected to see similar results from our analysis, which was that people who had .NET, Java, and PHP under their skill set were paid more, because in our data set, we also have features like SQL, Java, .NET, etc.

Martín et al.'s study highlighted that an IT worker's salary is often associated with education and experience as the two most determining factors for the total wages. In their Interpretable Linear Model section, they produced a table similar to our "feature importance" table. It shows that the most recent technology and back end technology contribute to generally higher pay. For example, new software technologies like AngularJs and Security each scored 3.23*10^-06 in p-value and 0.00041 in p-value, respectively. Thus, we decided feature importance analysis was an important method to include in our study to help us check if we were going to get a similar result to Martín et al. 's findings.

After the feature importance analysis, they also conducted the K-mean clustering method for previously mentioned features. They discovered that "salary and job conditions improve with experience." They also discovered that the longer the years of time people had under their experience the more it would positively influence employee salary and permanent contract capability. (Martín et al., 2018) Martín et al.'s discovery served as an important model for us when conducting our research and checking on our results to see if our study had provided a similar conclusion to theirs.

Having considered all the related research done in the past, we concluded that besides the paper "Salary Prediction in the IT Job Market with Few High-Dimensional Samples: A Spanish Case Study," other researchers had barely touched on using skills data to predict salary. This study shortage motivated us to look into how the combination of different IT skill sets influences people's salaries in the United States. In this research, we wanted to ask the question of how much do IT skills affect salary.

4. Methods

4.1. Data Set

The data set we decided to use is the Stack Overflow 2021 Developer Survey data. This data set was created by the Stack Overflow survey team. They have conducted this type of survey study since 2011. The purpose of conducting these developer surveys was to Stack Overflow understand their users' profiles and how they use the Stack Overflow platform. So Stack Overflow can help improve their platform and services. Generally, the survey respondents are all developers of some kind. Some are students still in college, some have been working in the industry as a business analyst or software engineer for 5 to 6 years. The data is directly acquired by downloading from the website of the StackOverflow Survey site. The data we used for our study is the 2021 data set. The dimension of the Stackoverflow 2021 Survey contains 46844 user responses and 47 individual questions(columns) except the user response id category. Stack Overflow did not include some respondents' entry into this study due to those present spending less than 10 minutes on the survey.One of the key reasons we decided to use this survey data is beaceue this data set contains survey respondents' self reported salaries. This feature is an important factor we wanted to study for our initial goal of doing this thesis.

4.2. Feature Selection

Before we start messing around with this data right away, we spend some time reading through the features that we think are going to impact IT workers' annual salaries. The features we finalized are: Country, Education level, Learn Code, Developer Type, Organization Size, LanguageHaveWorkedWith, DatabaseHaveWorkedWith, PlatformHaveWorkedWith, WebframeHaveWorkedWith, MiscTechHaveWorkedWith, ToolsTechHaveWorkedWith, NEWCollabToolsHaveWorkedWith, OpSys(Operation System), Age, Gender, Trans, Sexuality, and Ethnicity.

We choose to include these features into our study because we believe these features have influence on an employee's earnings. The features we did not include are features that would not help us understand the relationship between a person's skill and salary. For example, one feature that measures how long it took survey respondents to complete the survey. Another feature that asks if they found this survey difficult to complete. (The meaning of these variable names is in Table 1.)

Table 1

Variable Name	Description
Country	Current Country (Categorical)
US_State	US State names (Categorical)
EdLevel	Latest Education (Categorical)
LearnCode	How did interviewees learn Code
DevType	Developer Type(Categorical)
OrgSize	Organization Size
LanguageHaveWorkedWith	Programmer language have worked with
DatabaseHaveWorkedWith	Database Have Worked With
PlatformHaveWorkedWith	Cloud Platform Have Worked With
WebframeHaveWorkedWith	Web frame Have Worked With
MiscTechHaveWorkedWith	Programming Libraries and Packages
ToolsTechHaveWorkedWith	Development tools (e.g. Unity 3D, Deno, Docker, etc.)
NEWCollabToolsHaveWorkedWith	Collaboration Tools have worked with
OpSys	Operation system used (e.g. Windows, MacOX)
Age	Age (Numerical)
Gender	Gender (Categorical)
Trans	Transgender (Bool)
Sexuality	Sexuality (Categorical)
Ethnicity	Ethnicity (Categorical)
ConvertedCompYearly	current total compensation(salary, bonuses, and perks, before taxes and deductions) (numerical: Float)
LogCompYearly	Logged current total compensation (with application of .apply(np.log) function to all ConvertedCompYearly data points)
YearsCodePro	Years Code Professionally

YearsCode	Years Code (non-professional)

4.3. Data Preprocessing

Considering the first phase of the project we are trying to forecast users' salaries, and try to find how certain skills affect people's earnings. To do all these we want to adopt multiple different machine learning (ML) algorithms, to make the implementation of ML algorithms easier, we decided to dummy all the data points under each column the reason for doing so is because dummied variables enable us to use a single regression equation to represent multiple groups. After using "Pandas" builtin function ".get_dummies", we immediately received an error. We noticed under the column YearsCode(Total Coding Years) and YearsCodePro(Total years of coding professionally), these two columns have more than just numerical data. Somehow the designer of the survey decided to give the option for people to select options like "Less than 1 year" and "More than 50 years" as options for years of coding experience matching those two options. To make Python less confused about mixing variable types. We decided to replace "Less than 1 year" with 0.5 and "More than 50 years" with 50. After finishing this procedure, we converted all the data points under YearsCode and YearsCodePro into float data types, to compromise having "0.1" as the potential entry. Then we dummified all the data points in this data frame.

After all the data points are being finalized and dummified, we decided to visualize the distribution of the salary by applying the ".hist" function to the "ConvertedCompYearly" column, we then realized that due to some interviewees having entered values that are way too large. These outlier values are making visualization unable to display properly because we realized the panda's plot won't display other lower salaries. So we decided to create a new column in the salary_2021 data frame called "LogCompYearly" this column's value will be generated by copying the value directly from the "ConvertedCompYearly" and applying the function ".apply(np.log)" on all of them.

4.4. Implementation of UMAP Skill Analysis (Move this to the first section)

After exploring and analyzing the skills that have a large influence on employees' earnings we conclude that almost all the people who work in the industry possess more than just one specific skill. In addition, it is not one specific skill, for example, it is not because of knowledge in one specific programming language that makes a huge impact on people's earnings, it is a set of combinations of skills that could potentially make a huge impact. So we decided to use UMAP: Uniform Manifold Approximation and

Projection for Dimension Reduction. This method allows us to discover what skills are clustered together. For example, from the original data set, the survey asked the survey respondents “Which programming, scripting, and markup languages have you done extensive development work in over the past year, and which do you want to work in over the next year?” And the survey would ask the respondents to check all of the boxes that apply(see picture below).

Which **programming, scripting, and markup languages** have you done extensive development work in over the past year, and which do you want to work in over the next year? (If you both worked with the language and want to continue to do so, please check both boxes in that row.)

	Worked with in PAST year	Want to work with NEXT year
APL	☐	☐
Assembly	☐	☐
Bash/Shell	☐	☐
C	☐	☐
C#	☐	☐
C++	☐	☐
Clojure	☐	☐
COBOL	☐	☐
Crystal	☐	☐
Dart	☐	☐
Delphi	☐	☐
Elixir	☐	☐
Erlang	☐	☐
F#	☐	☐
Go	☐	☐

In order to do perform UMAP analysis, we first need to modify the original data frame, so we selected columns LanguageHaveWorkedWith, DatabaseHaveWorkedWith, PlatformHaveWorkedWith, WebframeHaveWorkedWith, MiscTechHaveWorkedWIth, ToolsTechHaveWorkedWith, we contacted all these columns that indicate “have” we then dummified these columns using the pandas building function “pandas.get_dummies” these will get dummified version of each column and we named this new data frame “have_concat” After the creation of the “have_concat” we moved on to create a different data frame called “wants_concat” by repeating the similar step we did in creating the “have_concat” data frame. We applied the “fit_transform” function to the have_concat and wants_concat data frames. This procedure is to standardize/scale the data set. Because each row in both “have_concat” and “wants_concat” is each response, so we added the response id to each row in those two columns. In the following step, we applied the “.T” function that helped transpose the index and columns in both data frames. So the index becomes the skills that respondents have and the column names become the

response id. Finally, in our last step, we applied the “reducer.fit_transform” function to those two data frames. We got an embedded data frame that has labels as indexes that indicate the skills processed and skills wanted from all the survey responses. (result in the image below)

	index	0	1
0	APL-have	5.137106	5.726786
1	Assembly-have	5.329269	4.357448
2	Bash/Shell-have	5.257352	4.510713
3	C-have	5.578898	4.170143
4	C#-have	23.137074	0.548474
...	...	...	...
127	React Native-have	7.496116	4.955647
128	TensorFlow-have	-7.286006	5.914949
129	TensorFlow-have	-7.438866	6.067808
130	Torch/PyTorch-have	7.208188	3.782141
131	Torch/PyTorch-have	7.257589	3.703567

132 rows × 3 columns

	index	0	1
0	APL-want	13.352165	4.605072
1	Assembly-want	13.659985	5.438484
2	Bash/Shell-want	13.691504	5.074054
3	C-want	13.595110	5.648702
4	C#-want	8.421185	6.961921
...	...	...	...
127	React Native-want	16.214493	5.626323
128	TensorFlow-want	4.190674	5.286184
129	TensorFlow-want	4.075215	5.320823
130	Torch/PyTorch-want	3.805952	5.292262
131	Torch/PyTorch-want	3.827378	5.210289

132 rows × 3 columns

These two data frames are the final form we need to visualize the UMAP visualization. To visualize the UMAP we decided to import the Plotly package and apply the scatter plot based on two data frames. The Plotly has interactive tooltips that can help us better understand what each dot on the scatter plot stands for as we move our mouse on the visualization.

4.5. People's Skill and Salary UMAP clustering Analysis

After exploring how different skills are clustered together, we want to extend our scope and explored how different people with different skill sets are clustered together, in addition, we want to add the salary into this analysis. The goal of this task is we want to see how we can use this approach to generate a visualization to see how people with different skill sets are clustered together and how is their salaries looking across the 'map.'

For this task, the process of generating UMAP is fairly similar to the process in the previous UMAP Skill Analysis Section. The only difference is that we replaced the index column of the data frame from the previous UMAP Skill Analysis Section with each individual person's skillsets, and this table below shows the look of the data frame.

	0	1	level_1
0	-11.493869	7.728032	C++, HTML/CSS, JavaScript, Objective-C, PHP, S...
1	-1.681480	5.350989	JavaScript, Python, PostgreSQL, Cordova
2	12.810015	3.818241	Assembly, C, Python, R, Rust, SQLite, Heroku, ...
3	-12.946157	6.342293	JavaScript, TypeScript
4	12.326033	5.723650	Bash/Shell, HTML/CSS, Python, SQL, Elasticsear...
...	...	...	...
83434	12.991014	21.408916	Clojure, Kotlin, SQL, Oracle, SQLite, AWS
83435	2.845593	7.152935	Firebase, MariaDB, MySQL, PostgreSQL, Redis, S...
83436	11.424229	6.022973	Groovy, Java, Python, DynamoDB, Elasticsearch,...
83437	12.764727	3.858016	Bash/Shell, JavaScript, Node.js, Python, Cassa...
83438	-0.908328	3.124102	Delphi, Elixir, HTML/CSS, Java, JavaScript, Or...

83439 rows × 3 columns

In this data frame, we can see that 0 and 1 are the coordinates, and the "level_1" represents the skill set people possess. The "ConvertedCompYearly" is the legitimized compensation data. We will use this data frame to generate a Plotly expression using dot plots.

4.6. Implementation of Machine Learning Algorithms and Feature Importance Analysis

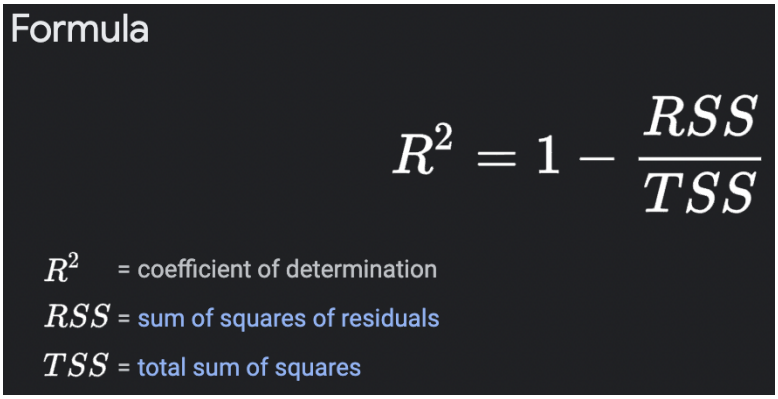
Right before we start to do the machine learning we need to split the dummified data frame into two different sets, the train and test set, which are represented by character 'X' which stands for the training set, and character 'y' stands for the testing set.

In the machine learning section, we decided to use 6 machine learning(ML) algorithms. They are LinearSVR, Decision Tree Regressor, GaussianNB, Random Forest

Regressor, KNeighborsRegressor, Linear Regression, Gradient Boosting, and model_selection package which we will need to use the function ".cross_validate" from it. In Jupyter Notebook we used Sklearn to import all aforementioned ML algorithms packages(function set). In addition we will use the 'r2' scoring method on each of these algorithms. We set the number of folds in cross validation to 5 folds.

- **SVM(Support Vector Machine):** "SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data are transformed in such a way that the separator could be drawn as a hyperplane."(IBM)
- **Decision Tree Regressor:** "A decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes."(Decision tree regression)
- **Gaussian Naive Bayes:** "supports continuous-valued features and models each conforming to a Gaussian (normal) distribution. An approach to creating a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions." (Majumder, 2020)
- **Random Forest Regressor:** "Random forest is a type of supervised learning algorithm that uses ensemble methods (bagging) to solve both regression and classification problems. The algorithm operates by constructing a multitude of decision trees at training time and outputting the mean/mode of prediction of the individual trees." (Raj, 2021)
- **KNN:** "works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression)." (Harrison, 2019)
- **Linear Regression:** "In Regression, we plot a graph between the variables which best fit the given data points. Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis). To calculate best-fit line linear regression uses a traditional slope-intercept form." (Linear regression algorithm to make predictions easily 2021)
- **Gradient Boosting Regressor:** "Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model in order to minimize the error." (Displayer 2020)
- **R2 Scoring Method:** "The R2 score is a very important metric that is used to evaluate the performance of a regression-based machine learning model. It is

pronounced as R squared and is also known as the coefficient of determination. It works by measuring the amount of variance in the predictions explained by the dataset." (Kharwal, 2021)



After importing all the packages we give each function a name of reference. They are being assigned with names in string data types, and we pack these ML function sets and their assigned names into a dictionary format, and we named those dictionary 'Models.' We then created an empty pandas data frame and a for loop. The empty pandas data frame has 5 columns. They are model(ML algorithm name), train_score(ML algorithm training Score), test_score(ML algorithm testing Score), fit_time(ML algorithm fitting time), score_time(ML algorithm scoring time). We named that data frame "cv_results_all." After the creation of the empty data frame, we constructed a for loop to iterate through each key in the models and apply each ML algorithm to the data sets using a cross-validate function. In the cross-validate function, the parameters are modes(ML model), X, y, return_training_score parameter is set to True, scoring method is set to be method "r2", the cv(number of fold during cross validation) parameter set to be 5, last but not least the n_jobs parameter is set to be '-1.' Normally, machine learning models will split data into parts, one part for training and another part for testing. ML model will train on the training set of data and test on the testing set of data. For k-folds cross validation we have talked about it briefly, here we want to provide an example. For example, on a random data set, if we choose 5 folds on a data set, in the first round, we will train using the folds 1, 2, 3, 4 but test on the 5th fold. In the second round we will train on the folds 1, 2, 3, 5 and test on the fold 4 and so on for 3 more iterations. The test score is generated through such processes.

- **Train _score:** how the model is fitting the data, it describes how the model generalizes the data.
- **Test_score:** measured how the model performed in the testing set of data.

After finding out the best performing ML algorithm, we decided to add an analysis method called — feature importance analysis. So we can see how the best performing ML algorithm is ranking the top 20 to 25 most important features. To do that we decided to create an empty data frame, with two columns. The left stands for feature, the right column stands for feature importance score. The feature importance score is

extracted by, first, creating a variable and assigning that variable equal to the Fitting the classifier to the input training data which are aforementioned data frames X, and y (looks like this: rf(variable name: Random Forest) = RandomForestRegressor().fit(X, y)). Then we apply ".feature_importance_" to the previously fitted variable 'rf'. Like this rf.feature_importance_. This code will allow Jupyter Notebook to return all the feature's correlated impotence scores. After that, we push all the feature importance scores(p-value) and their corresponding feature names into the empty data frame so we can future examine the most important features ranked by the best performing algorithms.

To improve the performance of 6 ML algorithms and further explore the most important features(skills), we decided to perform the first round of dimensionality reduction by removing features(columns) that are irrelevant to our focus of this study, those features are "Age", "Gender", "Trans", "Ethnicity", "Sexuality", and "Country". At the same time limit all the survey response data to just the response from the United States. After feature reduction and shifting focus data to just focus on the US. We repeated the previous machine learning and feature importance analysis.

We realized that simply removing the aforementioned features is not helping us to understand what are the most important skills. So we decided to continue to perform second round of dimensionality reduction by only using these features: LanguageHaveWorkedWith, DatabaseHaveWorkedWith, PlatformHaveWorkedWith, WebframeHaveWorkedWith, MiscTechHaveWorkedWIth, ToolsTechHaveWorkedWith, NewCollabToolsHaveWorkedWith, and OpSys. Then perform another round of ML analysis and feature importance examination.

4.7. Skill Want & Skill Have Against Salary Analysis

In this section we conducted the last part of our analysis by focusing on comparing the skills we want and skills we have against salary. We planned on generating two separate horizontal bar plots to visualize the data. So the y-axis is going to be all different kinds of skill sets, and for each tick on the y-axis there are going to be 2 bars, one bar representing "yes" and another bar representing "no." The x-axis is going to be the salary.

Before we started manipulating the data set, we realized that for the y-axis there will be more than 100 ticks if we do not reduce all different programming languages and development platforms into a summarized category. So what we did was we created 6 different variable names, they are 'lang', 'database', 'platform', 'web', 'misc_tech', and 'tools_tech'. Each of these represents all the skills, programming languages, platforms that interviewees have reported. In our Jupyter Notebook, we wrote code that read through all the data from columns: LanguageHaveWorkedWith, DatabaseHaveWorkedWith, PlatformHaveWorkedWith, WebframeHaveWorkedWith, MiscTechHaveWorkedWIth, ToolsTechHaveWorkedWith. For each specific sill under each of these columns, we collect both 'yes' and 'no' answers and push these response

data under the 6 different variable names we created previously, we then convert this data structure into a data frame so we can plot the result. One last thing we did before the plot was to integrate the salary data with the data frame, so the x-axis will display the annual salary information. By following the previous aforementioned methodology of previous methods we also created the data frame for skills wants data frame. The final form of these two different data frames is shown below.

Skill Want & Salary (top 5 rows of data frame)

	Response_id	Salary	Type	Skill	Present
0	0	62268.0	lang	APL-want	0.0
1	0	62268.0	lang	Assembly-want	0.0
2	0	62268.0	lang	Bash/Shell-want	0.0
3	0	62268.0	lang	C#-want	0.0
4	0	62268.0	lang	C++-want	0.0

Skill Have & Salary (top 5 rows of data frame)

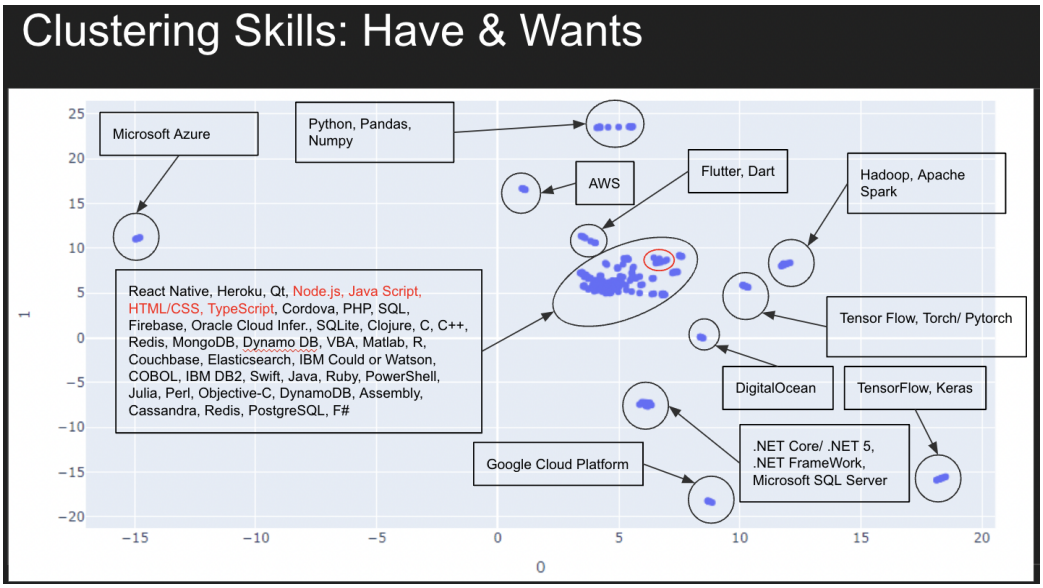
	Response_id	Salary	Type	Skill	Present
0	0	62268.0	lang	APL-have	0.0
1	0	62268.0	lang	Assembly-have	0.0
2	0	62268.0	lang	Bash/Shell-have	0.0
3	0	62268.0	lang	C#-have	0.0
4	0	62268.0	lang	C++-have	1.0

From these two data frames above, we have 5 different columns, the first column is Response_id, which is not going to be used in this sections analysis, the Salary column is going to be used in the y-axis, the Type is going to be used in the x-axis, the Skill column is not going to be used in the analysis, last but not least the Present column is used for labeling two different bars on each tick for the y-axis.

5. Results

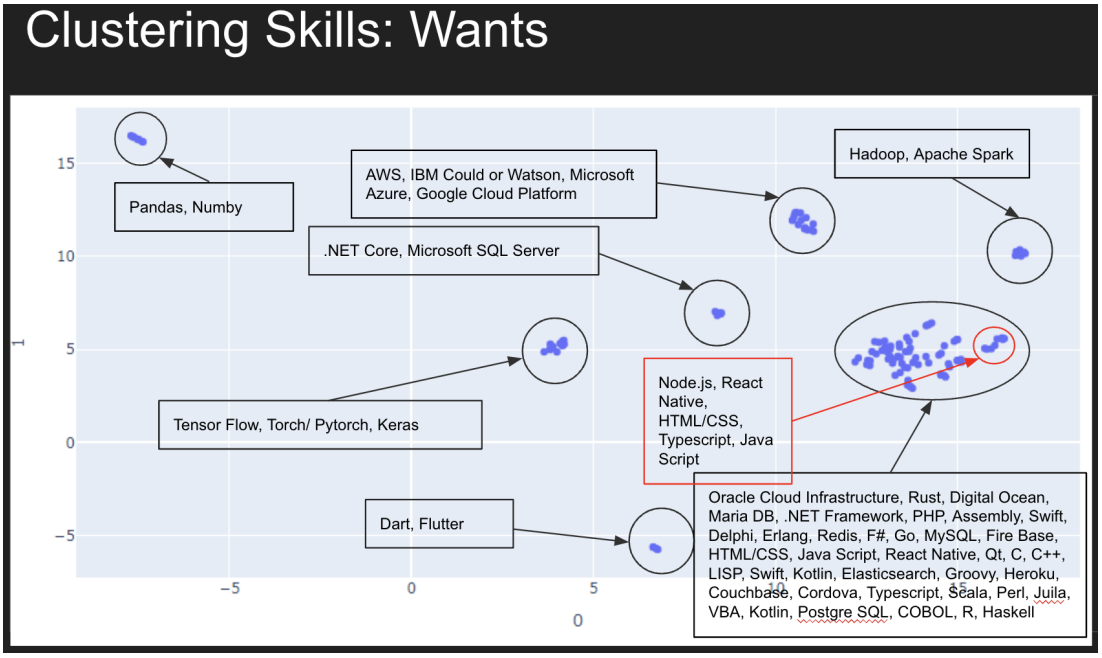
5.1. UMAP analysis and Cluster plot Interpretation

The first visualization contains both have and want skills. The purpose of this visualization is to see what skills that people have at the same time want to learn more. The visualization of skills have and wants is displayed below.



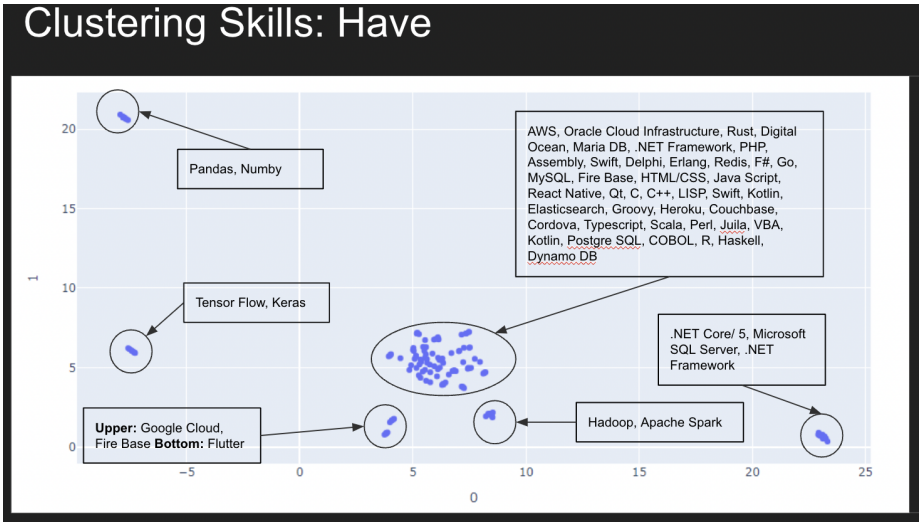
As you can see in this visualization that Python, Pandas, and Numpy are clustered together. When we were using the mouse on this visualization we noticed that there are four dots clustered together here. So that means 2 dots for every 3 skills. Because Pandas and Numpy are both libraries under the Python programming language so this explains why these dots are clustered together. A similar thing can be discovered about the Tensor Flow and Torch/ Pytorch, as you can see they are clustered together as well because they are both python libraries that are used for deep learning and ML development. One thing people might find interesting is that TensorFlow and Keras are clustered together but not clustered together with Pytorch. We think it is because although Pytorch is a deep learning library for python, it is a totally different library on its own. The Keras is the high-level API(Application Programming Interface) that is built based on the Tensorflow library; it's in a similar relationship just like the panda's library is built to use on Python. One thing we found is very interesting is that before we got this visualization we would expect that the web services like AWS, Microsoft Azure, and Google Cloud Platform are clustered together. However, they are not clustered together in this visualization. We think it’s probably because they are targeting different audience groups that have different needs for web service. For example, if a developer’s focus is building web infrastructure then they would likely choose AWS, if they are building Windows-based applications they may choose Microsoft Azure.

After visualizing the scatter plot for both have and wants UMAP visualization we went on conducting visualization for wants, which is shown below.



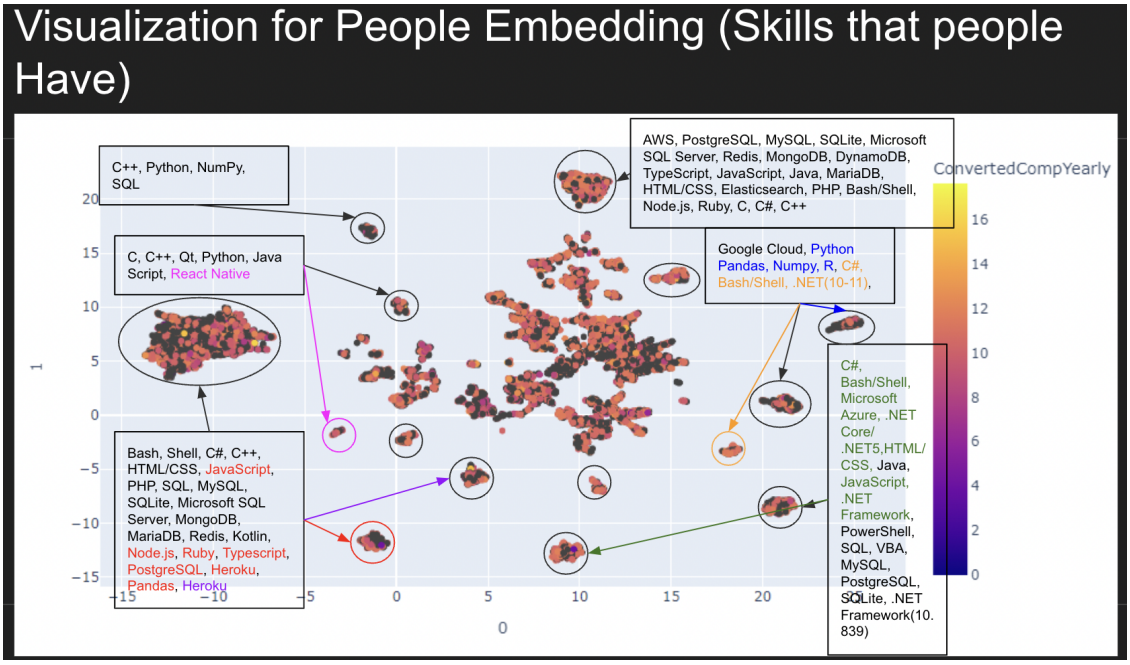
In this case, the clustering result is closer to our expectations. As you can see, people who want to learn about pandas also want to learn about Numpy. In reality, they are both libraries under the Python programming language. The Tensor Flow, Torch/ Pytorch, and Keras are clustered together because people who want to learn about deep learning want to learn how to use all these libraries. Furthermore, we see that all the web services applications and platforms are clustered together, as you can see AWS, IBM Cloud, Google Cloud, and Azure are all grouped very close together. Last but not least we can also see that all the web-development languages are also clustered together in the red circled area on the plot above. In this case, HTML is a basic web development language, another language like JavaScript is a web development language/script that is built on HTML that enables HTML to display more interactive features.

In the last section of the UMAP analysis, we looked at the skills that people have already possessed. This result is shown in the image below.



In this case, we can see that generally when people know about Pandas people will know how to use Numpy because they are very if not the most basic library and function in Python, so it make sense that when people are filling out the survey they would select both if they have already know how to use one of them. There is another pair of skills that also makes sense is the TensorFlow and Keras. As we mentioned earlier, Keras is an extension package that is designed for TensorFlow so when people check the box of Keras they will also check the box of TensorFlow. So most of the time, they are correlated. A similar conclusion can be drawn between Google Cloud and FireBase. FireBase is designed and developed by Google, and it is an extension of the Google Cloud Platform, that serves as a mobile development platform that is used to build mobile-based applications. So it makes sense that when people say they know and have used FireBase will also say they have used Google Cloud Platform. Hadoop and Apache Spark are very similar, they both allow the user to manage big data sets and solve vast data problems. Hadoop was developed prior to Apache Spark. As a newcomer, Apache Spark works faster. Because developers sometimes combine these two tools together to do their work, many developers that are working on Apache Spark had the experience of working with Hadoop or maybe still using Hadoop to this day. So this makes sense as to why we are seeing that Apache Spark and Hadoop are clustered together.

5.2. Result People's Skill and Salary UMAP clustering Analysis



This visualization above shows the dot plot of people and their salaries. To be more specific this visualization shows people's skill set and their salaries. The texts inside of the black box(tooltip) are the skills that this cluster of people has. And the color of the dot represents the salary they earn per year. We have to take note that the scale of their salary is logrizmized. The legend color is ranging from 'cold' represented at the bottom shown by the number 0 to warm at the top shown by the number 16. (note: the dark gray dots are people who did not report their salary.)

If we look at the plot closer we can see that in the top left corner the tooltip shows that a group of people have skills in C++, Python, Numpy, and SQL. So we can conclude that these people are more likely to be data scientists or data analysts. Because these skills are all data science-related programming languages and tools.

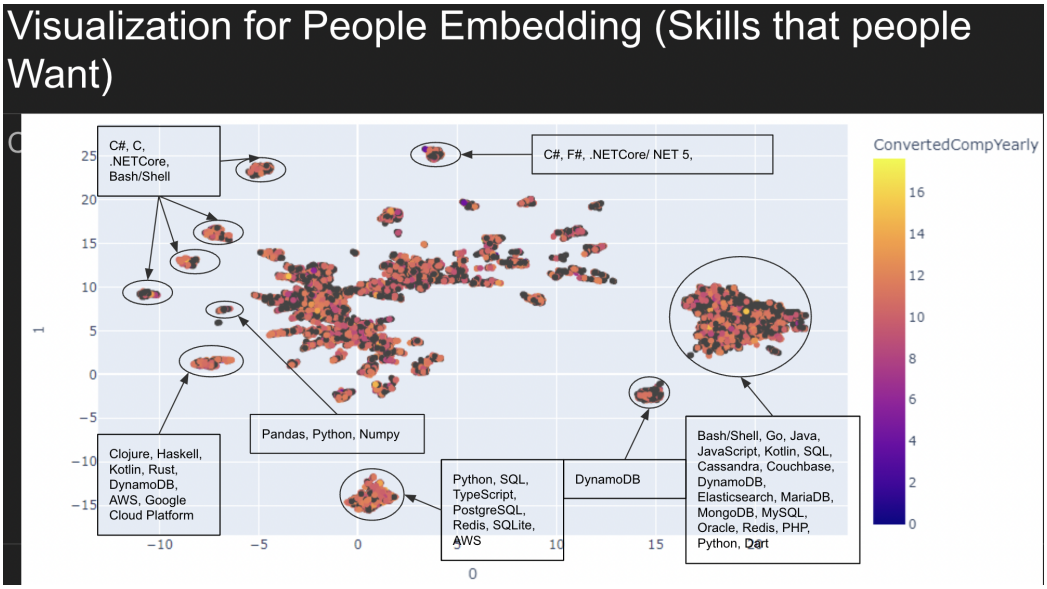
Similarly, a group of people also shares a similar skill set as the aforementioned groups of people. This group can be found over the cluster on the far right side of the visualization pointed by the blue arrow. We can see that this group of people also possess the skill of Python, Pandas, Numpy, and R. As we have previously explained in section 5.1 UMAP Analysis and Cluster Plot Interpretation. Numpy and Pandas are part of the Python library and the programming language R happens to be used by many data scientists, mostly favored by people who are working in a business or financial analyst role. One interesting point we noticed is that the Google Cloud Platform is also embedded in the tooltip. We included it because many people who possess these data analyst-related skills also seem to have worked with Google Cloud Platform in the

past. We think Google Cloud Platform may be one type of Cloud Computing platform that Business and Financial Analysts use. From this cluster, we can also see the yellow-colored texts, they are colored in yellow because they often occur together. After researching the design purpose of the C# and .NET we realized that these two types of development skills are all associated with computer application development. However, the difference is that the C# is a programming language, and the .NET is the framework on which the language is built. So .NET developers will use programming languages such as C#. From this cluster, we can see that the color of the dots is all in the range of 10 to 12 in terms of the ComvertedCompYearly salary range.

Next, we looked at the green-colored text in the tooltip pointing to the cluster located in the bottom middle of the visualization. We can see those green-colored texts are C#, Bash/Shell, Microsoft Azure, .NET Core/.NET5, HTML/CSS, JavaScript, and .NET Framework. As we mentioned in the previous paragraph. Texts in the same tooltips that use the same color are the highlights that mean those skills often occur together. This clue can also be used as a predicting factor for readers to guess what kind of developer group may be embedded in that cluster. In this case, the green-texts are suggesting to some developers in this cluster that they are web-based application developers. Because Microsoft Azure is a cloud computing platform that is used by people to manage cloud services, the .NET/Core /.NET5 is used to develop cloud-enabled, internet-connected apps, for example, UPS developers use .NET to develop internet-based shipping management apps to track packages. If we look at these dots' color we can see that most of these people are in the range of 12 to 14 in the ConvertedCompYearly salary range.

The result of this suggests that the "back end" and internet-based APP and service applications developer who uses .NET framework and Microsoft Azure cloud service tends to make more money than people who are in an analyst position who only knows how to use Pandas and R. Although we can't draw a conclusion on whether a person will make more money based on what kind of development platform and computer programming language they use, still, these skill sets gives us a better understanding of the nature of their job. With that information in mind, it will help us understand the level of difficulty of their job thus their salary.

After analyzing the Visualization for People Embedding (Skills that people Have) we did a second visualization focusing on skills that people want. This visualization looks very similar to the visualization we did for "Skills that people Have." The visualization of "Skills that people Want" is shown below.



From this visualization for Skills that people Want we can discover what kind of combination of skills that people want to learn to use. Just from skimming through the visualization, we noticed that we can see some major platforms and languages showing up. For example, in the top left corner of the visualization, we can see that there are four different clusters of dots, according to the tooltips we labeled we see that most dots(interviewees) have mentioned they wanted to learn more about C#, C, .NET Core, Bash/Shell. As we previously explained that C# and .NetCore is used for developing computer applications when C# is combined with .NetCore we can tell these developers have an interest to learn these skills to develop web-based applications and web backend services.

Moving downwards from the visualization we can see that people in one cluster want to learn more about Clojure, Haskell, Kotlin, Rust, DynamoDB, AWS, and Google Cloud Platform. We can tell that this group of people want to learn more about backend development revolving around data mining, security, APP development, and cloud storage management.

When moving rightward to the visualization we can see that a group of dots represents a group of people who want to learn more about Python, SQL, TypeScript, PostgreSQL, Redis, SQLite, and AWS. This group of people is the ones that want to learn more about data science skills combined with a web development focus. Because besides TypeScript the rest are all data management and data analysis-related tools and programming languages.

Considering the fact that most developers in the field are in a constant learning process. So in this survey, they will very likely select both options of “Worked within PAST year” and “Want to work with NEXT year.” So the reader will realize these two visualizations under this section look very similar.

5.3. Machine Learning Analysis with Dimensionality Reduction and Feature Importance Score Interpretation

In the first round of ML predicting results we get our test_score back which shows the ML algorithm's performance over the empty data frame created before. Now it is filled with the ML algorithm results. The ML algorithms are tested by using all the columns (feature variable name) in table 1.0. The best performing algorithm is Gradient Boost Regressor which scored 5.096657e-01 in test score. The second-best performing algorithm is SVM(SVR in Table below), which scored 5.182468e-01in test score.

Model	Train_score	Test_score	Fit_time	Score_time
SVR	0.516769	5.074175e-01	11.225948	0.033430
Decision Tree	1.000000	-2.709891e-02	3.526559	0.036698
RandomForest	0.931413	5.079685e-01	221.919190	0.0409014
KNeighbors	0.541370	3.040362e-01	0.205045	16.195362
GradientBoost Regressor	0.529342	5.096657e-01	25.916747	0.086701

After examining the result of the feature importance table generated from ML algorithm SVM Gradient Boost Regressor we noticed that the top 20 most important features ranked from SVM are all nationalities. Because Country is a complex factor to include in our study, due to each country has a very unique economic condition, they may pay their IT worker in a very different way, in addition, they are not the feature we are interested in, so we decided to see the feature importance score generated by the second-best performing ML algorithm — Gradient Boosted Regressor(GBR). While trying to implement the GBR to feature importance function we realize that GBR as a regressor does not have the ability to use the ".feature_importances_" function. So we have to examine the third-best performing algorithm which is the Random Forest (RF) algorithm's feature importance table.

```
# RandomForestRegressor
feature_importance = pd.DataFrame()
feature_importance["feature"] = X.columns
feature_importance["importance"] = rf.feature_importances_
feature_importance = feature_importance.sort_values("importance", ascending=False)
feature_importance.head(20)
```

	feature	importance
166	United States of America	0.175327
0	YearsCodePro	0.138750
478	White or of European descent	0.069874
1	YearsCode	0.031854
29	Canada	0.013174
310	PHP	0.012539
164	United Kingdom of Great Britain and Northern I...	0.011750
411	MacOS	0.011691
23	Brazil	0.009792
417	18-24 years old	0.008443
377	Docker	0.007822
57	Germany	0.006790
68	India	0.006450
283	Just me - I am a freelancer, sole proprietor, ...	0.005294
278	2 to 9 employees	0.004907
9	Australia	0.004789
276	10,000 or more employees	0.004778
74	Israel	0.004677
336	AWS	0.004628
415	Windows	0.004509

In the feature importance analysis from the RF algorithm, we noticed that the feature with the second-highest p-value(importance) is YearsCodePro which scored 0.138750 in p-value. The feature YearsCodePro is referring to the number of years that a person has been doing coding-related work professionally(i.e. For work). We noticed that features like YearsCodePro are reflecting a person's experience level in their professional occupations. At the same time, we realized that in Martín et al.'s study they have emphasized that the longer the years of time people have under their experience the more it will positively influence employees' salary and permanent contract capability. Our feature importance table result on features like YearCodePro is the second-most important feature further supports Martín et al.'s result from their study.

Other than YearsCodePro, features like the United States are ranked in No.1 most important feature by RF algorithm, ethnicity features like "White or European descent", and other personal information-related features are also taking many places and mixed in the top 25 most important features rankings ranked by RF feature importance ranking. Although these features do have a more influential role in influencing people's earnings in the IT field, it does not help us understand what kind of skill-related features are influencing people's earnings greatly. Thus we decided to perform the dimensionality reduction in the following steps to just focus our scope on surveys that come from the US, and remove features like country, ethnicity, sexuality, etc.

In the first round of dimensionality reduction, we limited the focus of our data to just the survey response data from the US. We also removed features "Age", "Gender", "Trans", "Ethnicity", "Sexuality", and "Country." After dimensionality reduction we

repeated the same ML code and we realized that the best performing algorithm is still GBR which scored 1.237436e-01 in test score. The second best-forming ML algorithm is RF which scored 9.135073e-02 in test score. We may also notice that after each time we remove features, the performance of the test_score drops. It is beaceue less features for the ML model to compute means less data for machine learning model to get trained on the accuracy therefore will reduce. The key benefit of doing feature reduction is that it helps with reducing ML algorithms' over fitting.

Model	Train Score	Test_score	Fit_time	Score_time
SVM	0.101927	8.19980e-02	1.436121	0.005951
RandomForest	0.873155	9.135073e-02	21.868160	0.059718
Gradient Boost Regressor	0.275169	1.247436e-01	3.140773	0.009722

Because Sklearn's GBR function does not have access to the ".feature_importances_" function. So we decided to use the RF to generate the feature importance table. After the feature importance table was generated(below) we noticed that the No.1 most important feature is "YearsCodePro" which scored 0.250848 in p-value, the No.2 most important feature is "10,000 or more employees"(survey responder reported company size) which scored significantly lower than the No.1 feature which only scored 0.056725 in p-value. The other three features behind the No.2 feature are "YearsCode"(Coding years before starts coding professionally), "2 to 9 employees"(company size), and "Just me - I am a freelancer…"(company size) From this, we can tell that the feature "YearsCodePro" advanced one place from the last feature importance testing. It shows that professional experience in the IT development career is still the most important factor in deciding people's earnings. The two features with regards to company sizes are not very helpful for us to interpret how they can have an effect on people's earnings, nor are they the features we are looking for in this study.

We were expecting that after the first round of dimensionality reduction we could see some education level, and developer type-related features being ranked among the top 20 most important features. After examining the table we realized that the feature– Student is ranked 8th place on the feature importance sheet which is higher than Data or business analyst. This does not make much sense considering students are less likely to also take on a full-time job and students who have full-time jobs are most likely part-time workers. So we suspect there is a chance that these people may be part time students.

```
feature_importance_gbr_us = pd.DataFrame()
feature_importance_gbr_us['feature'] = X_us_.columns
feature_importance_gbr_us['importance'] = gbr_us_ftest_.feature_importances_
feature_importance_gbr_us = feature_importance_gbr_us.sort_values('importance', ascending = False)
feature_importance_gbr_us.head(30)
```

	feature	importance
0	YearsCodePro	0.250848
51	10,000 or more employees	0.056725
1	YearsCode	0.055872
53	2 to 9 employees	0.046557
58	Just me - I am a freelancer, sole proprietor, ...	0.039248
190	Windows	0.038138
186	MacOS	0.022394
47	Student	0.021943
168	IntelliJ	0.021336
85	PHP	0.017668
23	Data or business analyst	0.017646
111	AWS	0.016645
46	Senior Executive (C-Suite, VP, etc.)	0.015945
49	1,000 to 4,999 employees	0.015669
73	Go	0.015608
8	Secondary school (e.g. American high school, G...	0.014765
95	TypeScript	0.012990
140	Flutter	0.012913
9	Some college/university study without earning ...	0.010057
113	Google Cloud Platform	0.010032

After seeing the feature importance table from the second time of dimensionality reduction we realize that there are still some unwanted features like company size. And feature that is hard for users to understand like developer type. So we decided that we wants to drop these features from our second time of dimensionality reduction and third round of ML and feature importance analysis. After implementing the reduced features to 6 ML algorithms we can see the best performing ML algorithm is Decision Tree, it scored 0.798395. The other ML algorithms are not so well performed across the board. They are all scored around ±0.500000 in test scores. This is surprising considering SVM, RF, and GBR were used as the top 3 performing ML algorithms in the previous data sets.

Model	Trian_score	Test_score	Fit_time	Score_time
SVM	0.476235	0.484877	0.207031	0.003075
Decision Tree	0.012615	0.798395	0.132315	0.002778
Random Forest	0.206444	0.538307	7.547244	0.047121
KNeighbors	0.475987	0.584362	0.013572	0.424179
Linear Regression	0.486051	0.494682	0.042655	0.002976

GradientBoost Regression	0.470741	0.493271	1.349812	0.005357

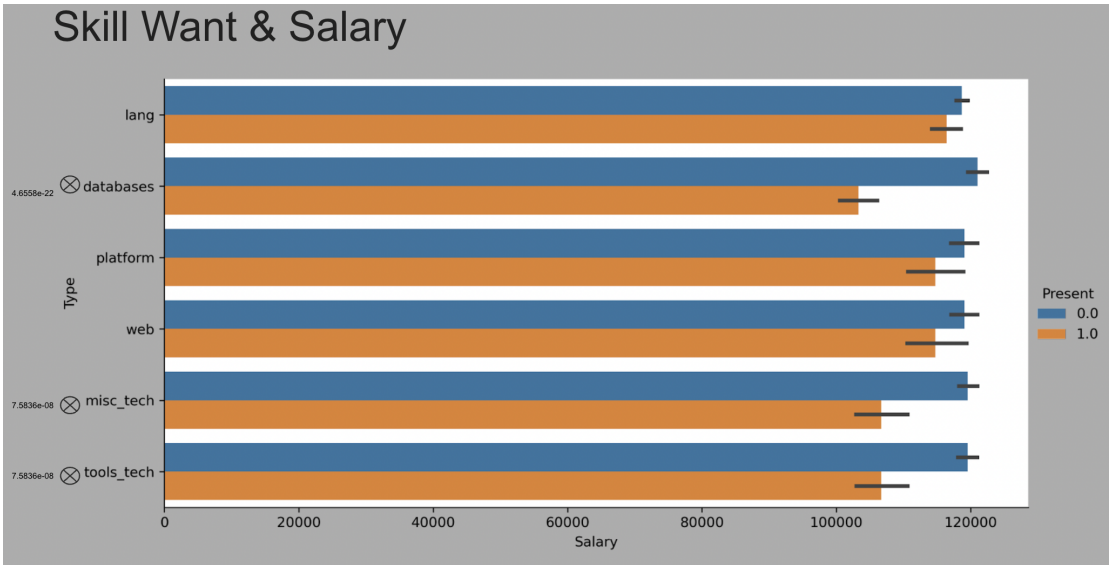
```
5 feature_importance_gbr_us.head(25)
```

Out[94]:

	feature	importance
14	Go	0.057397
52	AWS	0.054717
96	Kubernetes	0.053753
50	Redis	0.034934
16	HTML/CSS	0.032576
26	PHP	0.031742
40	DynamoDB	0.028216
19	JavaScript	0.028147
81	Flutter	0.026682
54	Google Cloud Platform	0.022829
36	TypeScript	0.022538
93	Docker	0.021720
41	Elasticsearch	0.021134
49	PostgreSQL	0.020841
37	VBA	0.018369
0	APL	0.017802
99	Terraform	0.016897
63	Django	0.014893
21	Kotlin	0.014347
25	Objective-C	0.014005
47	MySQL	0.013747
80	Cordova	0.013041
20	Julia	0.012842
18	Java	0.012785
57	Microsoft Azure	0.011968

From the second round of the feature importance table, we are expecting to see most of the high-ranking skills are all developer skills like a specific programming language, a developer tool, a developer platform, etc. To be more specific we are expecting to see the Cloud Platform have a higher ranking, the Misc Tech-related skills to have a higher ranking in the feature importance table. If we look at the feature importance table (above) we can see the No. 1 most important feature ranked by Decision Tree(DT) is Go, which scored 0.057397 in p-value, the AWS: Amazon Web Service scored 0.054717 in p-value and the Kubernetes scored 0.053753 in p-value. We noticed that Go is a programming language developed by Google initially that is used for backend development. Now people could use Go to develop cloud and network services, people use Go to create Command-Line Interfaces(CLI), Go could also be used to create Web Applications and help with Dev&Ops and site reliability. Considering the

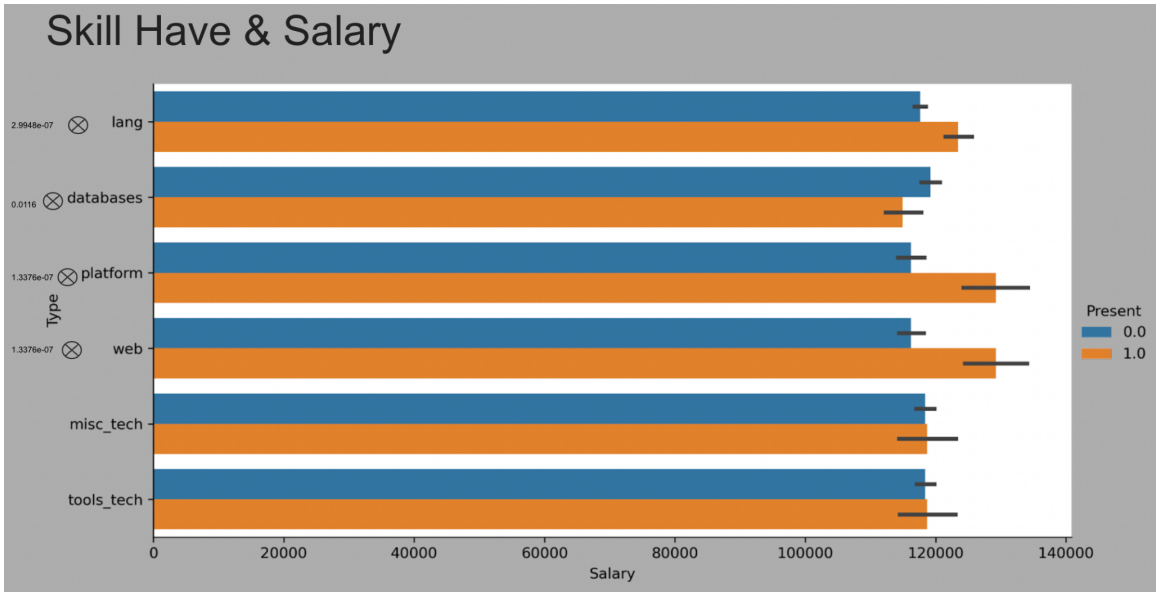
5.4. Results of Skill Want & Skill Have Against Salary Analysis



'Lang' in this visualization stands for languages we have worked with, the blue bar means no and the orange bar means yes. Each bar on the plot is the average value of the salary. We have also calculated the p-value for each category and we put a mark on the category that has a p-value less than 0.05. We noticed that categories like "databases", "misc_tech", and "tools_tech" have a p-value that is less than 0.05. That means "databases", "misc_tech", and "tools_tech" measurements are statistically significant, thus we should reject the null hypothesis, this means that the data we see from these three visualizations is not random.

From the category databases, we can see that people who don't want a base are earning more money than people who do. One cause of that may be people who have possessed databases skills are already working in the data science field, they have possessed enough knowledge thus they don't need to learn extra other skills within the databases. Another cause can be people who select they don't want databases skills are the people who are already working in a different discipline of IT development. For example, a rendering engineer makes a lot of money but they don't handle databases, they do not need to learn how to handle databases to make more money. So these people will likely drag the no response to a higher salary range.

The misc_tech and tools_tech both have the p-value of 7.536e-08 means both of these two categories are less than the threshold of 0.05. This means that both of these two categories' visualization does not occur on a random occasion.



From this Skill Have & Salary above we conducted the null hypothesis test, and we discovered that categories like ‘lang’, ‘database’, ‘platform’, ‘web’ are both having a p-value less than 0.05. The p-value for ‘lang’ is 0.9948e-07, the p-value for databases is 0.0116, the p-value of the platform is 1.3376e-07. The p-value for the web is 1.3376e-07. For these, for categories, we can conclude that because their p-value is less than 0.05 thus (reason)

Looking at the visualization we can see the people who reported yes under the lang(Programming Language) tend to earn more salary than those who reported no. The databases category has returned a very different result where people who reported they don’t have database knowledge are earning more salary than those who reported they have database knowledge. This Result is certainly surprising considering during the recent 3 years including 2022 data science-related skills have become highly demanded by IT industries, we are expecting to see people who have database skills make more on average than those who do not. However, we can see that the p-value of the databases is less than 0.05 thus the effects on databases salary from both reporting may be affected by some random cause. So our new hypothesis is that on average people who have database skills are making more than those who don't. This may be caused by some outlier developers who are working in other IT fields that make significantly more money than database workers’ salaries on average. Thus this may affect the result of this visualization.

The platform skill has a p-value of 1.3376e-07 which is significantly lower than the threshold of 0.05. We can see that people who have possessed platform skills are making way more than people who don’t. Logically this may sound right however, due to the fact that we have to reject the null hypothesis, we have to exclude the chance that this result from visualization is a random event. Consider the platform is referring to a cloud computing platform and the cloud computing platform is still in a fast development and

expansion phase. We can make a hypothesis that people who have platform-related skills are likely to make more money, and the visualization has confirmed that as well.

The web refers to web development skills. These skills include web development platforms from programming languages to web development platforms. The p-value for the web is also 1.3376e-07, which is also smaller than our threshold of 0.05. In this case, we have to accept the alternative hypothesis. This means the salary data from responses yes and no are not random. From the plot, it's reasonable to say that people who have possessed more web development skills are making more money in general.

The misc tech and the tools tech both have the same value returned for the visualization. To explain this we have to take note that misc tech stands for the development packages and libraries that can be used on different programming languages and development tools. Developers can not use misc tech on their own to develop projects. Thus misc tech has to be built on tools tech. This explains why the yes and no for both misc tech and tools tech are having the same value that reflects the salary. In addition, we can see that both misc teach and tools tech are having p-value higher than 0.05, this suggests that there is no direct cause as to why yes and no (have and have not possessed) from misc tach and tools tech are the same.

6. Discussion

From the first UMAP analysis, we saw how different skills are clustered together. However to be more scientific about our findings we believe for the future expansion of UMAP analysis we should incorporate a legend for each dot. The underlying reason for that is because all these different skills have their attribute, this attribute can be found in the survey pdf sample. For example, from Clustering Skill: Have & Wants we can see that Python, Pandas, and Numpy are clustered together. Those three are highly correlated, but they are different kinds of skills. According to the survey pdf documentation, Python is classified as a programming language, and Numpy and Pandas are considered libraries/frameworks. Sometimes different combinations of programming language and frameworks can yield a potential higher salary. As we found from our literature review section, it is not likely that signal skill dictates who gets the longer contract and higher annual pay, it is the combination of skills. For example under the section of Result People's Skill and Salary UMAP clustering Analysis, from visualization of Visualization for People Embedding (Skills that people Have) we see that an IT worker gets paid more than $100,000 a year when they have skill sets that are a combination of C#(programming language) with .Net Framework/.NetCore(framework). Because this type of combination suggests that this specific developer is a computer application developer, if they happen to have HTML/CSS and Microsoft Azure under the skill sets, they are more likely to be web application developers that develop the back end of web service. This type of developer tends to make even more money annually. From this

example, you can see that by adding more features to a visualization, like applying different-shaped dots to each survey responder we can not only see their skill, their salary but how different kinds of skill categories are overlapped with each other. Thus we can generate a more comprehensive result for our UMAP analysis.

In the machine learning section, we have noticed that regressors can not be applied with the .feature_importance function to it, so during multiple rounds of machine learning analysis we were unable to apply the feature importance to the GBR model. Thus, the feature importance score we got from the second or third beast perform model may not be the most accurate one. So this shortcoming may become one of the limitations of this ML section. To compare our first three rounds of feature importance analysis we noticed that the features “YearsCodePro” and “YearsCode” are always ranked among the top 5 of the most important features. So we can see that the higher the feature importance score the more likely that feature is going to affect the model on salary prediction. From this finding, we can connect back to the article we read from the Spanish Study that the longevity of technical experience is the top deciding factor of salary.

7. Conclusion

The growing population of college graduates along with the growing demand for the job market has pushed standards even higher for many IT workers in the United States. Many industries have developed a complex task that involves very rigorous evaluations with a focus on assisting in hiring the best candidates for the company. Thus many companies adopted the machine learning(ML) technique in their recruitment processes. In order for future candidates to be prepared for future job markets’ ML model-based candidate selection algorithms, it is important to understand how these algorithms work in classifying IT workers' skills.

By introducing 4 different approaches we are able to get a better understanding of how skills and skillsets are affecting the IT workers’ salaries. We learned that the UMAP analysis is a good way to see what kind of skills are popularly combined together. We have also discovered that what are the skill sets that get combined together have the greater potential of reaching a better pay bucket. Understanding this information can help future IT workers understand their interests and help them figure out the best way to prepare their skill sets. Looking at the machine learning section our goal was to focus on finding the best performing ML algorithms and then apply feature importance study to them. One important thing to keep in mind is that the feature importance score does not reflect the values of each individual skill. The feature importance score reflects how important that feature is to the machine learning model to accurately predict the actual salary. So, from the last round of the feature importance analysis, we can conclude that AWS and Google Cloud are two very important features for ML models to accurately

predict salaries. As to if having AWS or Google Cloud under your skillset is going to help you to make more money, we don't know. But one thing we can conclude is that considering feature importance score reflects if a certain feature is going to accurately predict the salary. We can reassure readers that "YearsCodePro" and "YearsCode" two features can be used as a predicting factor for salary. For example, people with 10 years of "YearsCodePro" under their resume are very more likely to get paid more than people who have 3 years of "YearsCodePro."

This feature's importance has also created a limitation, which is that other than measurable features like "YearsCodePro" which uses longitude of time as a measurement. Non-measurable things like "AWS "(a cloud platform) are just a category. It is not continuous or discrete; we can't use that to show if people who have a certain category of skills are going to make more money.

The limitation of feature importance that makes it unable to represent which category of skill is more important for people to have in order to get paid more could be resolved by using a different approach. This approach is what we have mentioned earlier in the discussion section. That is we could expand the Visualization for People Embedding (Skills that people Have) from the Result People's Skill and Salary UMAP clustering Analysis section. Where we could add options to the legend that use different shapes to represent 6 different kinds of categories we used on the y axis in Skill Have & Salary plot, and we make that dot plot interactive. So when you click on each dot that represents each survey respondent. That dot will expand into a new visualization which shows another UMAP plot that shows how each individual skill is clustered. If we could implement this on Linkedin, users can find specific developers who purported their earnings and see what kind of skills they have and how their IT skills are structured. This insight will provide new IT job seekers with guidance in developing similar skill sets and being successful as their predecessors.

8. Reference

Day, M., & Loewenstein, M. A. (2019). *On job requirements, skill, and wages*. U.S. Bureau of Labor Statistics. Retrieved April 3, 2022, from https://www.bls.gov/osmr/research-papers/2019/ec190030.htm

Sivaram, N., & Ramar, K. (2010). *Applicability of clustering and classification algorithms* ... Retrieved April 3, 2022, from https://www.ijcaonline.org/volume4/number5/pxc3871165.pdf

Amato, F., Boselli, R., Cesarini, M., Mercorio, F., Mezzanzanica, M., Moscato, V., Persia, F., & Picariello, A. (1970, January 1). *Figure 2 from Challenge: Processing web texts for classifying job offers: Semantic scholar*. IEEEXplore. Retrieved April 3, 2022, from

https://www.semanticscholar.org/paper/Challenge%3A-Processing-web-texts-for-classifying-job-Amato-Boselli/994967f9f89dceb6ede02a8e98b62369d3853ed9/figure/1

Abboud, Y., Boyer, A., & Brun, A. (2016, April 12). *Predict the emergence - application to competencies in job ...* HAL-inria. Retrieved April 3, 2022, from https://hal.inria.fr/hal-01254179/document

Brunato, M., & Battiti, R. (1970, January 1). *[PDF] X-MIFS: Exact mutual information for feature selection: Semantic scholar*. undefined. Retrieved April 3, 2022, from https://www.semanticscholar.org/paper/X-MIFS%3A-Exact-Mutual-Information-for-feature-Brunato-Battiti/364697d1e97a9c4838df75f6130e18f18ec46a92

Martín, I., Mariello, A., Battiti, R., & Hernández, J. A. (2018, July 5). *Salary prediction in the IT job market with few high-dimensional samples: A Spanish case study*. International Journal of Computational Intelligence Systems. Retrieved April 3, 2022, from https://www.atlantis-press.com/journals/ijcis/25899235

Decision tree regression. (n.d.). Retrieved April 17, 2022, from https://www.saedsayad.com/decision_tree_reg.htm

Majumder, P. (2020, February 23). *Gaussian naive Bayes*. OpenGenus IQ: Computing Expertise & Legacy. Retrieved April 17, 2022, from https://iq.opengenus.org/gaussian-naive-bayes/

Raj, A. (2021, June 11). *A quick and dirty guide to random forest regression*. Medium. Retrieved April 17, 2022, from https://towardsdatascience.com/a-quick-and-dirty-guide-to-random-forest-regression-52ca0af157f8

Harrison, O. (2019, July 14). *Machine learning basics with the K-nearest neighbors algorithm*. Medium. Retrieved April 17, 2022, from https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Linear regression algorithm to make predictions easily. Analytics Vidhya. (2021, June 10). Retrieved April 17, 2022, from https://www.analyticsvidhya.com/blog/2021/06/linear-regression-in-machine-learning/

Gradient boosting explained - the coolest kid on the Machine Learning Block. Displayr. (2020, December 7). Retrieved April 17, 2022, from https://www.displayr.com/gradient-boosting-the-coolest-kid-on-the-machine-learning-block/

Kharwal, A. (2021, June 22). *R2 score in machine learning*. Data Science | Machine Learning | Python | C++ | Coding | Programming | JavaScript. Retrieved April 17, 2022, from https://thecleverprogrammer.com/2021/06/22/r2-score-in-machine-learning/