Final Report

Team members.

1) Parikshit Joshi (210895H) - Leader
2) Jia Jun (210897S) - Group Member

Introduction/Background information.

A startup is a company typically in the early stages of its development. These entrepreneurial ventures are typically started by 1-3 founders who focus on capitalizing upon a perceived market demand by developing a viable product, service, or platform. Startups face high uncertainty and have high rates of failure, but a minority of them do go on to be successful and influential. Startups play a major role in economic growth. They bring new ideas, spur innovation, create employment thereby moving the economy. There has been an exponential growth in startups over the past few years. Predicting the success of a startup allows investors to find companies that have the potential for rapid growth, thereby allowing them to be one step ahead of the competition.

Seed capital can be used by startups to fund research and the development of their business strategies. A Large portion of a company's seed funding may originate from sources close to its creators, such as family, friends, and other contacts with a high risk of failure in the start-up.

Machine learning uses algorithms to create models that reveal patterns from data, allowing businesses to gain understanding and make predictions to enhance their business model. We can use the help of machine learning

to predict the outcomes of a startup whether it will be successful or not. There are several algorithms to choose from and our goal is to make the most accurate result possible, which is called predictive modeling. We will be using Random Forest, Decision Tree, Support-Vector Machine (SVM), Logistic Regression to provide our forecast result on whether a startup will turn into a success or a failure.

Business objectives identified.

The business objective is to identify success startups with success criteria of us being able to identify by 25%.

Main and sub tasks identified.

Collection of datasets using techniques that comply with data protection and privacy ethics. - By using knime, remove/encrypt any sensitive information like name, age.

Use visualizations to get better insights about data

1) Check for imbalanced data to see if the dataset is in the 80 and 20 range by using bar graphs
2) Check for missing data - see for any null values - location
3) Identify trends by using scatter plot - eg. higher rounds of valuation result in higher success rate .
4) Using correlation matrix to check for multicollinearity like zip code, latitude and longitude
5) Remove any outliers using RapidMiner
6) See which factors link to high success and which factors lead to lower success using a Sankey diagram
7) Find differences between success and failed business by analyzing the features.
8) Transforming data to make the model understand data better - eg latitude and longitude categorized to state

Cleaning and modifying data & prepare data for modeling

1. Find out the possible causes of missing data - why is the location missing
2. Figure out the possible approaches and the best approach to fix the missing data - Use RapidMiner to input the data using k-nearest neighbor

Build multiple models using various machine learning algorithm

In the decision tree, we make many models by choosing a different column at every iteration for e.g. bootstrap.

1.Use predictive modeling techniques by using misclassification, AOC, ROC on the processed data to compare models

2. Use unsupervised learning method - k means clustering, to understand the data better

3. Assess the performance of trained models - Accuracy, misclassification rate

4. Tune the parameters to achieve the best possible outcome - testing by increasing more features, does the model get better

- Increase/ Decrease Penalty value for better model
- Kernel function change from linear to quadratic
- Change the number of trees and also the bootstrap value

Feature selection

1. Use feature importance analysis to see which feature affects the results the most and which feature make the model worse

2. By using visualizations -parallel coordinates technique, it will give a deeper look and impacts of the features across the columns.

Delegation of tasks/responsibility.

Collection of datasets:
JiaJun -kaggle datasets
Parikshit - UCL,google,universities

Make visualizations:
Jia Jun
Use bar chart to find out number of null values and input them
For time series data, change the values into dd/mm/yyyy
If data is unbalanced, use RapidMiner to balance the dataset

Using knime to analyze and input the null values. Modify the time series data so that we can use it in the modeling. To balance the dataset, we can use RapidMiner to balance our dataset by having the same number of records from both success and failure. Some columns are unnecessary or irrelevant to our desired result or deciding factor which we must remove.
When doing the train-test split, it is important to make sure that the distribution of the data between the training set and testing set is similar: The range distribution should be from 80:20 ratio. Must test out different numbers of trees and Decrease penalty value for a better model.

Parikshit

1) Remove any columns that are not useful or empty
2) Build graphs to analyze feature selection
3) using correlation matrix to check for multicollinearity

It is necessary to do some data cleaning such as from the data before visualizing it. Use feature selection to remove those features where the p value is more than 0.05, and which columns are not improving the model. There are columns such as zip code, latitude and longitude which have high multicollinearity. Do data transformation if needed such as converting and categorising the latitude and longitude into city. For

modelling. Changing the Kernel function from linear to quadratic to see the model get better.

Scoring metrics:

1) Accuracy
2) Precision
3) Recall

Training Model using SAS Viya:

1) Support Vector Machine, Decision Tree - Parikshit
2) Logistic regression, Random Forest - Jia Jun

Data Collection: Jia Jun

The dataset we collected is from an online source found in kaggle, https://www.kaggle.com/datasets/manishkc06/startup-success-prediction

We used Knime to view our data, and used a statistical node to see through the data, number of unique values and majority data. By clicking on the occurrence table, we can see all the occurrences of data and how many times each occurrence occurs.

Occurrences Table - 0:26 - Statistics (Statistics)

File Edit Hilite Navigation View

Table "default" - Rows: 922 Spec - Columns: 123 Properties Flow Variables

Row ID	S founded_at	I Count (founded_at)	D Relativ...	S closed_at	I Count (closed_at)
Row0	1/1/2003	55	0.06	?	588
Row1	1/1/2002	54	0.059	6/1/2013	25
Row2	1/1/2005	54	0.059	1/1/2012	24
Row3	1/1/2006	54	0.059	7/1/2013	15
Row4	1/1/2000	53	0.057	5/1/2013	12
Row5	1/1/2004	50	0.054	1/1/2011	8
Row6	1/1/2008	50	0.054	1/1/2013	7
Row7	1/1/2007	45	0.049	10/1/2012	6
Row8	1/1/2001	38	0.041	8/1/2012	5
Row9	1/1/2009	34	0.037	1/1/2010	4
Row10	1/1/2010	24	0.026	1/1/2009	3
Row11	1/1/2011	13	0.014	11/1/2012	3
Row12	10/1/2006	9	0.01	8/1/2013	3
Row13	6/1/2006	9	0.01	3/1/2013	3
Row14	1/1/1999	8	0.009	2/1/2013	3
Row15	5/1/2010	8	0.009	11/1/2011	3
Row16	6/1/2005	7	0.008	7/1/2012	3

What each column means and information's about each column
Like if the column is category, binary- 1 or 0.
If binary 1 means yes , 0 means no

Columns

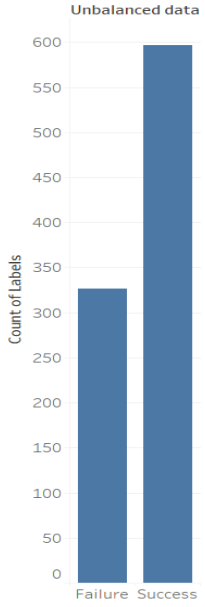
Unnamed: 0 - useless
State_code - state code of the company, with 35 unique values with majority is CA. CA for california, NY for New York etc There are 0 missing values for this column . 53% is CA and 11% is NY. The rest of the 36% are others.
Latitude - Latitude of the location of the company **Longitude** - Longitude of the location of the company
Zip_code - A ZIP Code is a postal code used by the United States Postal Service with data majority in zip_code is 94107 and 382 unique values.
Id - unique id of each company
City - this column shows where the company is based. San Francisco has the highest occurrence, with 128 occurrences in the dataset followed by New York, 91 occurrence. This shows that most of the startup companies are from these two cities
Unnamed: 6 - the city location including the postal code with data majority is San Francisco.
Name - the name of the company
Labels - it is to show if the company has achieved success or failure by binary values 1 - success - there are 597 companies succeeded, which is more than the failures in our dataset 0 - failed - there are 326 companies failed
founded_at - the date when the company was founded with 216 unique values with 1/1/2003 as data majority. The founded_at datas are mostly from 2000 to 2010, which shows that our datasets are mostly companies that are founded from 2000 to 2010
closed_at - the date when the company was closed with 202 unique values with 6/1/2013 as data majority. Most of the datas in closed_at are after 2010, which more companies are closed after 2010
first_funding_at - the date of the company since it got first funding with 585 unique values with 1/1/2008 as data majority. As noticed from the occurrences table, most of the companies got their first funding between 2005 to 2010.

last_funding_at - the date of the company since it got last funding with 680 unique values with 1/1/2008 as data majority.As noticed from the occurrences table, most of the companies got their last funding between 2008 to 2012.
age_first_funding_year - the age of the company since it got first funding, column **first_funding_at** minus **founded_at**. Majority of the occurrence is age 0, with 56 occurrences. This shows that most of the companies got their first funding on the day the company got started.
age_last_funding_year - the age of the company since it got last funding, with the majority of occurrences, 0 with 17 occurrences.
age_first_milestone_year - the age of the company since it got the first milestone, with majority of occurrences, 0 with 45 occurrences. This shows the company got their milestones when the company just started.
age_last_milestone_year - the age of the company since it got the last milestone, with majority of occurrences, 4 with 21 occurrences.
relationships - says how many relationships a startup has. For example a start up can have relationships with accountants, investors, vendors, mentors, etc. Majority of the occurrences are between 1 to 10, which means most of the companies got between 1 to 10 relationships.
funding_rounds - the amount of funding rounds that was given to the company. The majority of the occurrences are between 1 to 3, which means most of the companies are given 1 to 3 rounds of funding
funding_total_usd - the total amount of money funded to the company in usd
milestones - the amount of milestones the company has passed. A milestone in startups tracks progress as a startup grows and implements their plan. The majority of the occurrences are between 0 to 3, which means most of the companies have gotten 0 to 3 milestones
state_code.1 - the state code of the company, same as state_code
is_CA - whether the company is located in the state, CA, in binary format, 1 representing yes and 0 representing no. There are 436 companies that are in the state, CA, which is where most of the companies are located.

is_NY - whether the company is located in the state, NY, in binary format, 1 representing yes and 0 representing no. There are 106 companies that are in the state, NY
is_MA - whether the company is located in the state, MA, in binary format, 1 representing yes and 0 representing no. There are 83 companies that are in the state, MA
is_TX - whether the company is located in the state, TX, in binary format, 1 representing yes and 0 representing no. There are 42 companies that are in the state, TX
is_otherstate - whether the company is located in another state, in binary format, 1 representing yes and 0 representing no. There are 204 companies that are in othe state
category_code - the code of the category that the company is in. For example, music, enterprise, software etc. It has 35 unique values with software as data majority. The highest occurrence is software, with 153 occurrences followed by web, with 144 occurrences. This shows that software and web are the top popular category in our dataset
is_software - whether the company is in the category, software, in binary format, 1 representing yes and 0 representing no.
is_web - whether the company is in the category, web, in binary format, 1 representing yes and 0 representing no.
is_mobile - whether the company is in the category, mobile, in binary format, 1 representing yes and 0 representing no.
is_enterprise - whether the company is in the category, enterprise, in binary format, 1 representing yes and 0 representing no.
is_advertising - whether the company is in the category, advertising, in binary format, 1 representing yes and 0 representing no.
is_gamesvideo - whether the company is in the category, gamesvideo, in binary format, 1 representing yes and 0 representing no.

is_ecommerce - whether the company is in the category, ecommerce, in binary format, 1 representing yes and 0 representing no.
is_biotech - whether the company is in the category, biotech, in binary format, 1 representing yes and 0 representing no.
is_consulting - whether the company is in the category, consulting, in binary format, 1 representing yes and 0 representing no.
is_othercategory - whether the company is in another category, in binary format, 1 representing yes and 0 representing no.
object_id : Unique id of each company, same value as ID
has_VC - has been funded by venture capitalist, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 622, which is more than double the occurrence of 1, with 301 occurrences. This shows that most of the companies in our data was not funded by venture capitalist
has_angel - has been funded by angel investors, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 688, which is more than double the occurrence of 1, with 235 occurrences. This shows that most of the companies in our data was not funded by angel investor
has_roundA - has gone through round A funding, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 469 and the occurrence of 1, with 454 occurrences. This shows that the companies in our dataset that has gone through round A funding are quite even
has_roundB - has gone through round B funding, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 561, and occurrence of 1, with 362 occurrence. This shows that most of the companies in our dataset has not gone through round B funding
has_roundC - has gone through round C funding, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 708, which is more than double the occurrence of 1, with 215 occurrences. This shows most of the companies in our dataset has not gone through round C funding
has_roundD - has gone through round D funding, in binary format, 1 representing yes and 0 representing no. The occurrence of 0 is 831, which is more than 9 times the occurrence of 1, with 92 occurrences. This shows very less of the companies in our dataset has gone through round D funding
avg_participants - average number of participants. Most of the number of participants are between 1 to 3

Unbalanced Data



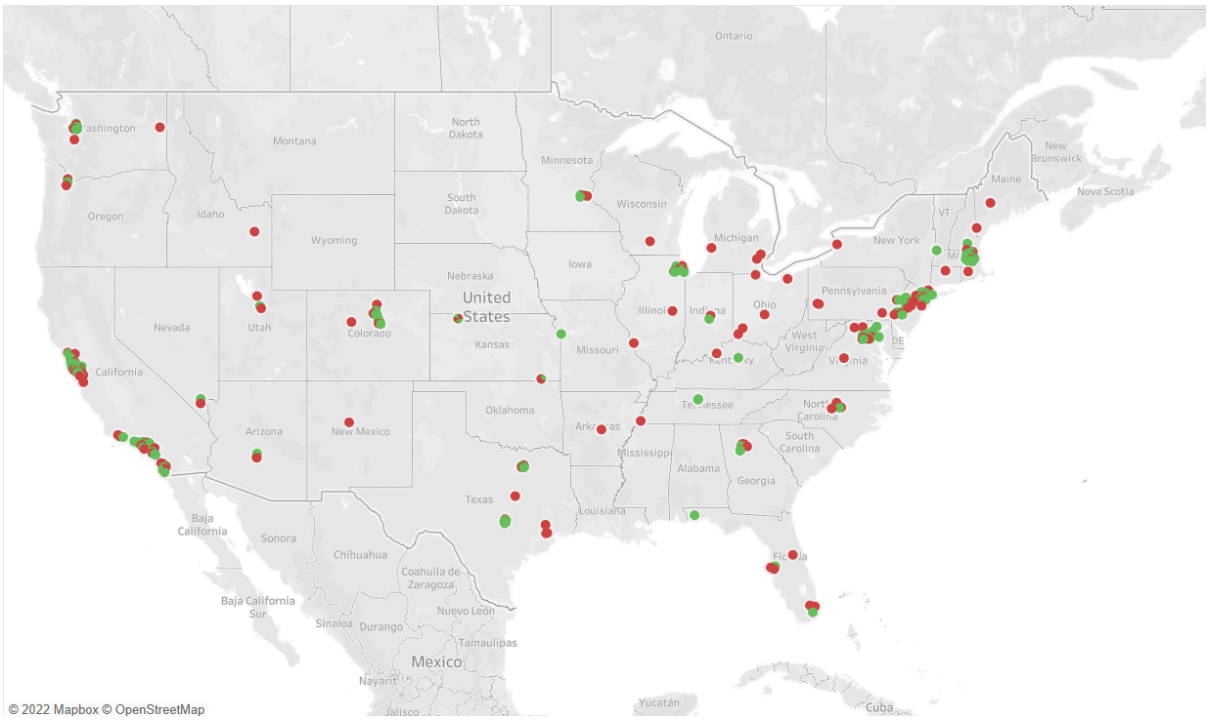
	is_top500 - is in the list of top 500 companies, in binary format, 1 representing yes and 0 representing no. The occurrence of 1 is 747 and occurrence of 0 is 176. Which shows that most of the companies in our dataset are in the top500 companies
	status - has 2 unique values with acquired and closed, acquired meaning the company was acquired and was a success, closed meaning the company was closed and was a failure. The occurrence of acquired is 597 and occurrence of closed is 326, which shows most of the companies in our dataset are acquired

Data Visualization: Parikshit

I used Tableau to check for the data balanced if it is in the 20/80 ratio. The data is balanced at the ratio of 65/35 ratio.

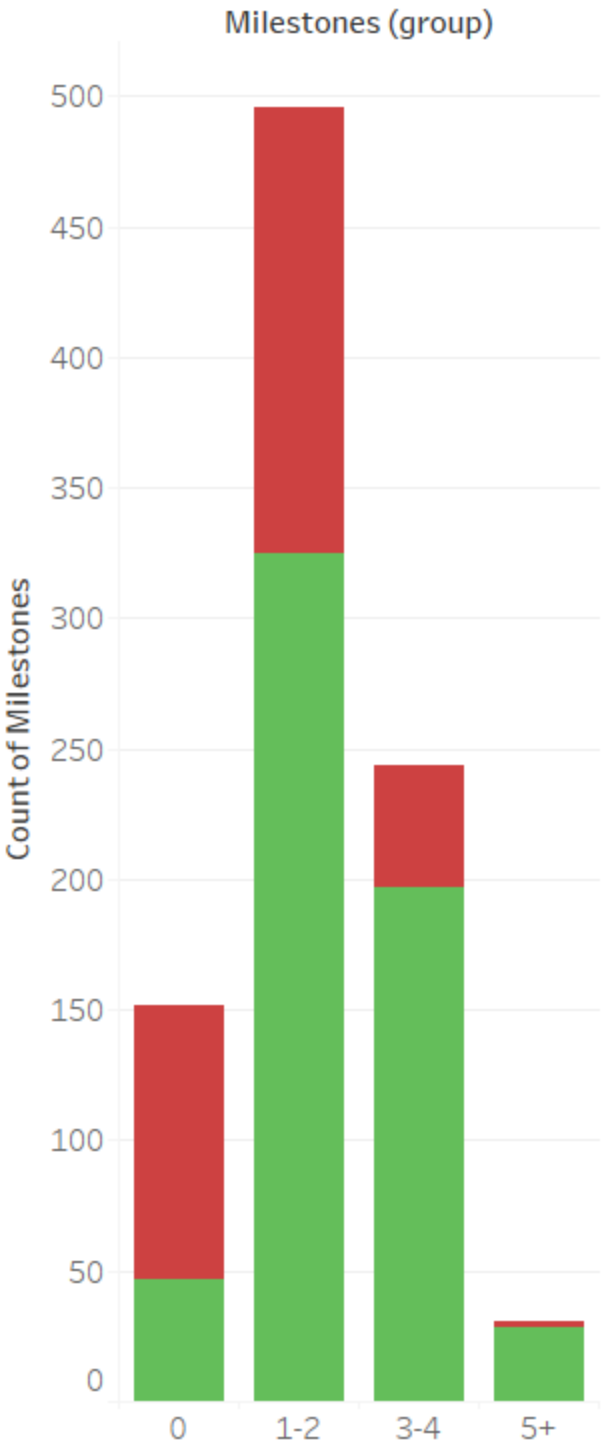
The reason i checked for data balance is when the records of a certain class are much more than the other class, our classifier may get biased towards the prediction. It doesn't represent the true distribution of the data

With imbalanced data, it doesn't represent the true distribution of the data. With unbalanced data our accuracy will be very high and our misclassification rate will be very low too. To keep our classifier honest, it was important for me to check about the data balance.



I used the map chart to show which state produces the most success and most failure companies. Most of the business startups from our data come from California, Pennsylvania and New York. A Lot of companies based in Pennsylvania have a higher chance of success. By doing this, we can analyze whether if the state is a factor for the company's success or not.

Below is another Bar Chart which is now showing the correlation between the number of Relationships to its success/failure. Here we can see that companies with more Relationships are more likely to succeed as compared to companies with fewer Relationships.

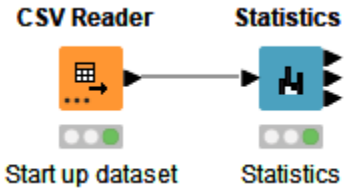


Below is another Bar Chart which is now showing the correlation between the number of company milestones to its success/failure. Here we can see that companies with more number milestones are more likely to succeed as compared to companies with fewer milestones.

Data Preparation

Jia Jun: Steps we did to check for missing data

After removing the unwanted columns, we used Knime to find out the statistics of the data, by using a csv reader node to read our dataset and then connecting it to a statistic node to find out the statistics of our dataset.



Missing data

There are 2 columns that have a missing value

1. age_first_milestone_year
2. age_last_milestone_year

Row ID	Column	No. missings	Histogram
age_first_mile...	age_first_mi...	152	
age_last_mile...	age_last_mil...	152	

- age_first_milestone_year as numerical data has 16.46% (152 rows) missing value
- age_last_milestone_year as numerical data has 16.46% (152 rows) missing value

Parikshit: How do we deal with missing data?

The missing value pattern is Missing at Random (MAR)

For the age_first_milestone_year column and age_last_milestone_year column, we filled the missing data with KNN with the neighbour value of 5..

We did this by using the Jupyter Notebook by using KNN Imputer which inputs the missing value and maintains the value and variability of your datasets and yet it is more precise and efficient than using the average values.

We did not replace the missing values with mean, median or mode values because using these methods to input can waste valuable data or lessen the variability of your data.

How does knn logic work?

In kNN methods, the goal is to detect 'k' samples in the dataset that are similar or close in space to each other. Using these 'k' samples, we estimate the value of the missing data points. Each sample's missing values are imputed using the value of the 'k'-neighbours found in the dataset. The predicted label is the one that has the most votes among the K neighbours, so it's the most common in the neighbourhood.

Why did I choose k = 5?

In classification, the decision boundaries are unstable when K is small for eg. k=1. That's why a substantial K value leads to smoother decision boundaries,thus I chose K to be 5. Another reason is since my data is not very big and only contains more than 900 rows I tried to keep my k value low.

How does the KNN imputer work ?

Code:

```
imputer = KNNImputer(n_neighbors=5)

data1 = pd.DataFrame(imputer.fit_transform(data),columns = data.columns)
```

n_neighbors: *number of data points to include closer to the missing value.*

fit_transform - In the fit() method, where we use the required formula and perform the calculation on the feature values of input data and fit this calculation to the transformer.

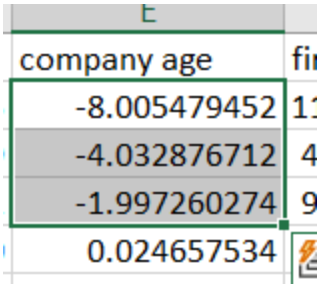
Parikshit: Implementing new column - Company's age

A date-time contains a lot of information that can be difficult for a model to take advantage of in its native form. Thus, I made a new column labelled as company's age to calculate how long the company has existed. This is because it is a better feature to implement instead of using founded_at and started_at as they are date values and since company_age is a numerical value, it is easier for the model to understand and will improve the model.

Parikshit: Remove errors from Company's age column

3/18/2009	
1/1/2002	2/15/2009
11/15/2000	
1/1/2004	4/27/2012
11/26/2007	
1/1/1999	3/28/2010
1/1/2003	9/22/2012
1/1/2004	9/28/2011
1/1/2002	7/25/2008
5/24/2005	
1/1/2007	11/14/2012
10/1/2006	2/17/2012
6/1/2009	5/27/2012
1/1/2007	1/28/2011

Fix error: To calculate the age, I had to format the founded_at and closed_at column for some of the rows. As some of the date formats were mm/dd/yyyy which was not allowing me to get the value. By changing the format of 149 rows from mm/dd/yyyy to dd/mm/yyyy, I could get the company's age.



Reason: Delete 3 columns because the company's age is negative. By removing these errors, it is making the data better for the other processes like better accuracy to detect outliers and also for inputting more accurate values for inputting the values.

Parikshit: Transformation of data

I had to find the absolute value of these four columns

Reason: There were negative values in these 4 columns. This means that these companies were first funded and last funded before their company started - prefunding. This also means that these companies were able to hit their first milestone and last milestone before their company started.

To transform this data, I made 8 columns. 2 of each feature.

1) **age_first_funding_year**
   - age_first_funding_year before startup
   - age_first_funding_year after startup
2) **age_last_funding_year**
   - age_last_funding_year before startup
   - age_last_funding_year after startup
3) **age_first_milestone_year**
   - age_first_milestone_year before startup
   - age_first_milestone_year after startup
4) **age_last_milestone_year**
   - age_last_milestone_year before startup
   - age_last_milestone_year after startup

Explanation

-age_first_funding_year
-age_last_funding_year
-age_first_milestone_year
-age_last_milestone_year

For those columns with the "**before startup**" tag, it changed all those values that were **negative** in these 4 columns above into **absolute values** and the positive values to 0.

For those columns with the "**after startup**" tag, it **kept** all the positive values in these 4 columns above positive and the negative values to 0.

By transforming the 4 columns, we felt that we are able to analyze the model better as we will be able to tell for Eg whether being funded early before or being funded later leads to more success. By making these models, we are able to do better analysis with more features.

Jia Jun: Removing unnecessary columns

Removal of columns & Reasoning

Unnamed: 0 - This column remains unknown
Id - The id has no impact on the possibility of the success rate of the company, therefore removing it **Object_id** - same value as id, which is unnecessary
Geographical location of the companies are represented by columns, is_CA, is_NY, is_MA, is_TX and is_otherstate. Therefore these columns are removed as it causes multicollinearity. **Latitude & Longitude** - The latitude and longitude of the company is unnecessary as it can be represented by the other columns **Zip_code** - the zip code of the company can be represented by the other columns **Unnamed: 6** - The city location including the postal code is not necessary **State_code** - The state code of the company, which is unuseful as it can be represented by the other columns

State_code.1 - The state code is already shown on the column, state_code. This column is duplicated which is unnecessary
Status - The status of the company being acquired or closed is already represented by the column, label, in binary format which makes this column useless
Category_code - there are binary columns already so there is no need for category code
Founded_at & Closed_at - the date of when the company started and closed, which can be implemented into one column, **'Company_age'.** After the implementation of the new column, these two should be removed
First_funding_at & Last_funding_at - the date when the company got its first funding at and last funding at, which can be represented by other columns, age_first_funding_year and age_last_funding_year. These columns should be removed due to multicollinearity
Age_first_funding_year & Age_last_funding_year - the age when the company got first funded and last funded. We decided to implement new columns, **'age_first_funding_year_before'** and **'age_first_funding_year_after'** stating the years when the company got funded before they started and after as there are negative values in the previous columns which might affect the machine learning algorithm.
Age_first_milestone_year & Age_last_milestone_year - the age when the company got first milestone and last milestone. We decided to implement new columns, **'age_first_milestone_year_before'** and **'age_first_milestone_year_after'** stating the years when the company got their first milestone before they started and after as there are negative values in the previous columns which might affect the machine learning algorithm.
Name - names of the companies, which are not useful to the analysis

Jia Jun : Outliers detection

What are outliers and why do we have to deal with them?

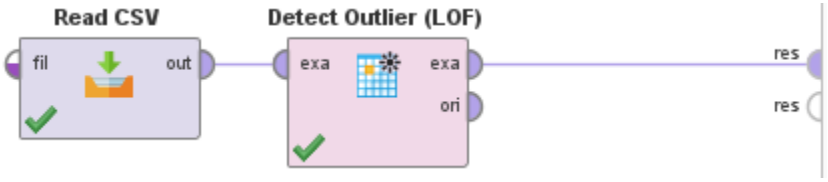
When we are dealing with data, we may have records that are odd or unusual when compared to the bulk of the rest of the records. We call such cases outliers or anomalies. In clustering results, the data point that is far away from the main clusters is an outlier.

Outliers are pretty rare and hence usually will not cause problems to most algorithms used for prediction. This is especially so when we have a large number of records. On the other hand, if

we have a small number of data records, anomalous or outlier records can be a concern because they can distort the outcomes of a modelling process.
For example, in classical statistics, regression algorithms can be strongly affected by such unusual cases

Steps we took to check for outliers

We use Rapidminer to check for outliers, by using a Read Excel node to read our dataset then using a outlier Detect Outlier (Distance) node to detect the outliers. What is Detect Outlier (Distance)? It is the simplest outlier detection operator. It works by calculating the distance to the k- nearest neighbours of each example.

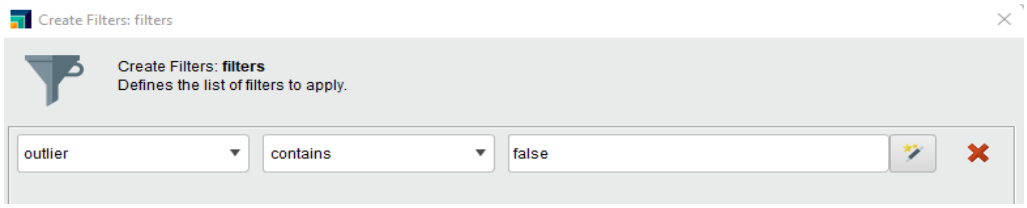


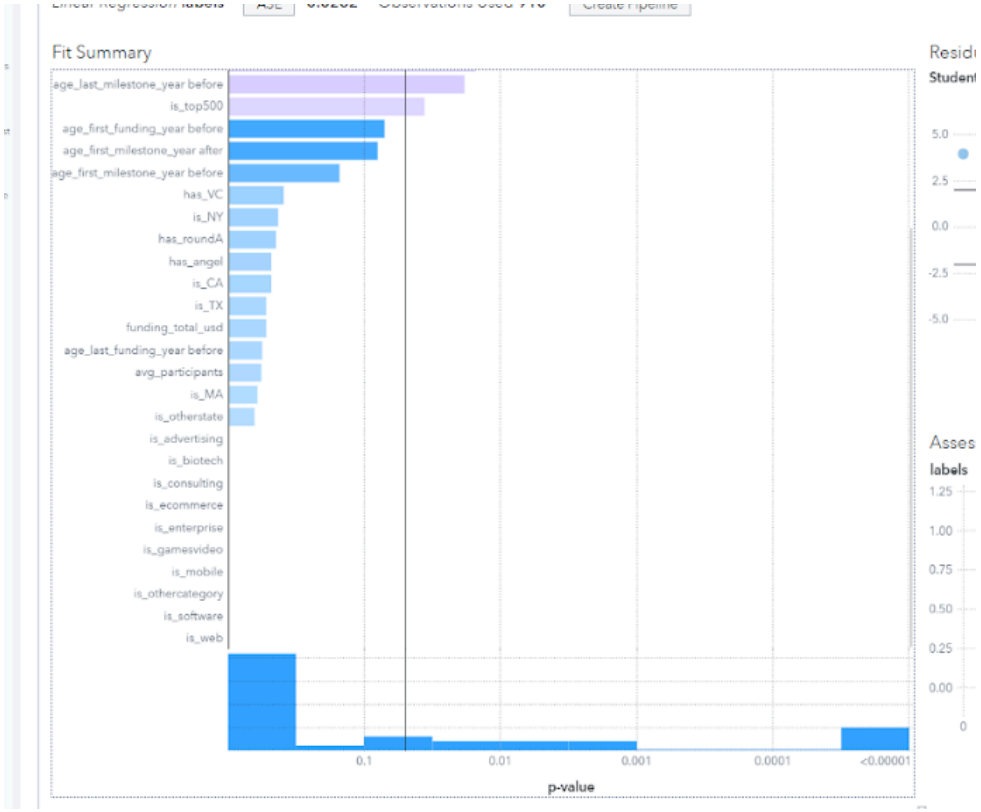
By looking at the result generated by the Detect Outlier (Distance) node, we can see that these are the 10 outliers being detected by our Detect Outlier (Distance) node.

Row No.	outlier ↓	labels	Company age	relationships	funding_rou...	funding_tota...	milestones
416	true	0	15.425	4	7	162264126	1
504	true	0	5.838	5	2	510000000	3
520	true	0	7.173	6	8	211403000	2
556	true	1	13.910	38	5	142000000	3
606	true	1	15.274	37	5	299500000	2
678	true	0	12.701	1	3	148000000	0
692	true	1	16.603	4	6	232000100	0
735	true	1	16.603	25	8	129677153	3
869	true	1	18.858	19	4	5700000000	2
910	true	1	21.690	29	7	238209999	3

How do we deal with outliers?

We chose to only remove 10 rows of outliers as we only have 921 rows of data in total which is quite a small dataset.
We used Filter Examples node to filter out the outliers and only show the results of outliers is equal to ‘false’. Then use the Write Excel node to create an excel file to our local repository.





Parikshit: Feature Selection using SAS Viya

Our dataset has 36 features in the dataset. All the features we find in the dataset might not be useful in building a machine learning model to make the necessary prediction. It is expensive and takes a lot of computational power and it is a time-consuming process. Using some of the features might even make the predictions worse. So, feature selection plays a huge role in building a machine learning model. The columns where the color of the bar is very light or have no bar have p value of lesser than 0.05.

Info | Selection Summary | Assessment | Assessment Statistics

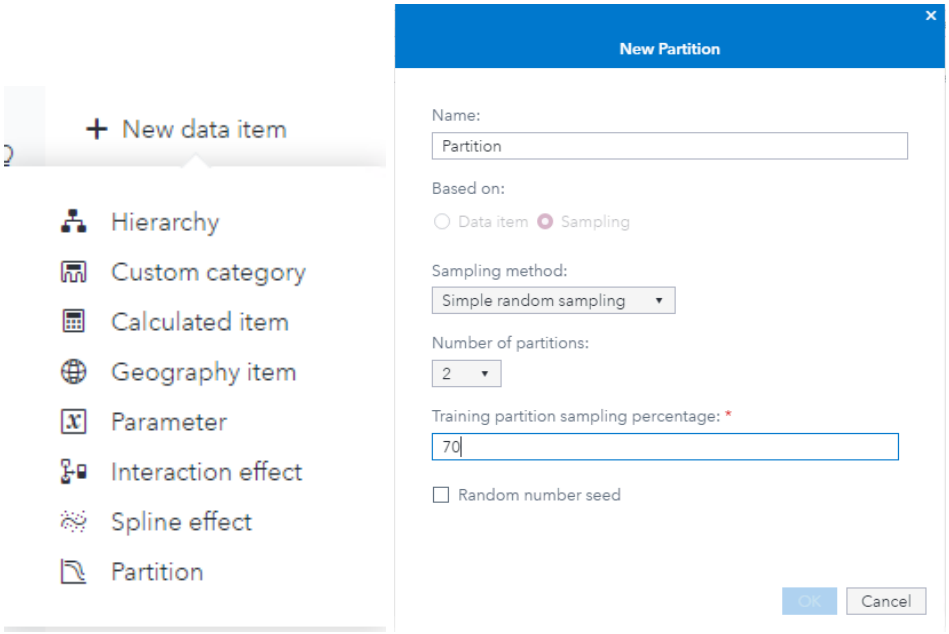
Step	Effect Removed	Number Of Effects
0		37
0	is_web	36
1	is_mobile	35
2	is_software	34
3	is_advertising	33
4	is_otherstate	32
5	is_MA	31
6	avg_participants	30
7	age_last_funding_year before	29
8	funding_total_usd	28
9	has_angel	27
10	is_gamesvideo	26
11	is_consulting	25
12	is_TX	24
13	has_roundA	23
14	has_VC	22
15	age_first_milestone_year before	21
16	age_first_milestone_year after	20
17	is_ecommerce	19
18	is_othercategory	18
19	age_last_milestone_year before	17
20	is_top500	16
21	has_roundD	15
22	age_first_funding_year before	14
23	relationships	13
24	has_roundB	12
25	funding_rounds	11

By using backward selection, these are the features that are removed by SAS viya. Thus, we removed all these columns. We will be using the remaining columns to analyze our features. By removing these features, and by setting our significance level to 0.05 our R squared and adjusted R square improved from 0.8673 to 0.8695. The mean square error improved also from 0.0303 to 0.0298.

Partitioning: Jia Jun

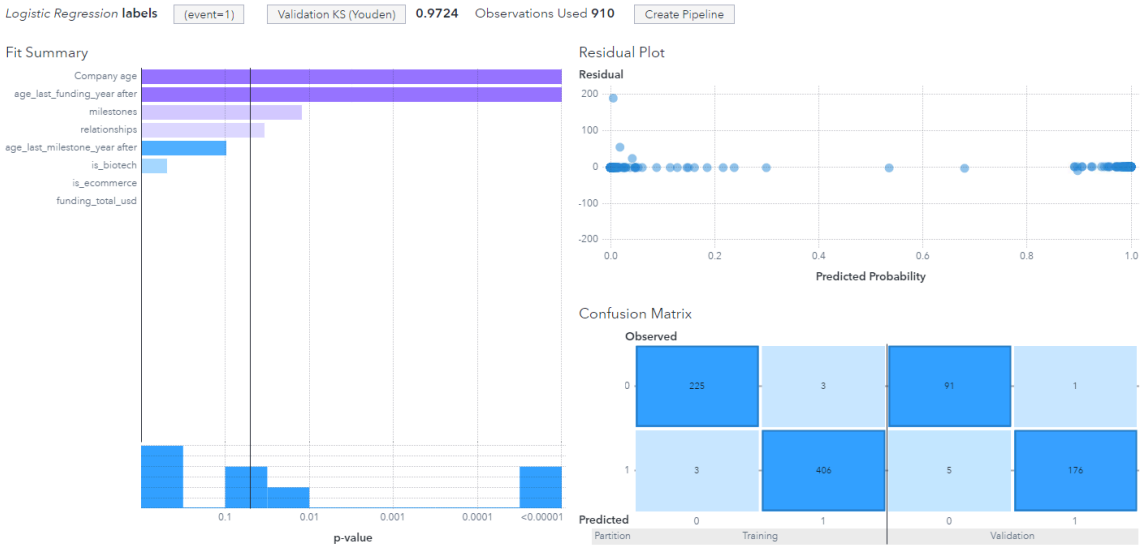
Partitioning is needed to partition the dataset to training and testing to detect the machine learning model behavior. It is necessary because if observations used in the training process are used, the evaluation of the model would be biased.

We partition our dataset in SAS Viya by adding new data item then put 70% of the dataset to training as our dataset is small.



Modeling

Logistic Regression: Jia Jun



Maximum iterations: 50

Number of bins: 20

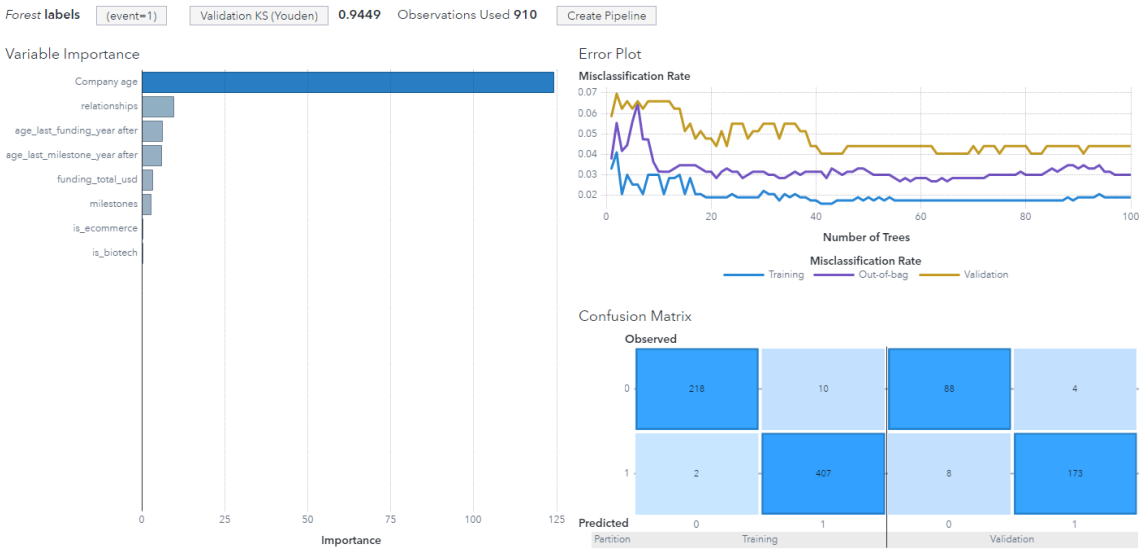
KS Youden: 0.9724

Misclassification rate: 0.0220

No tuning was needed as the model is giving the best possible result.

Random Forest: Jia Jun

Before tuning Random Forest



Number of Trees: 100

Bootstrap: 0.6

Maximum Levels: 6

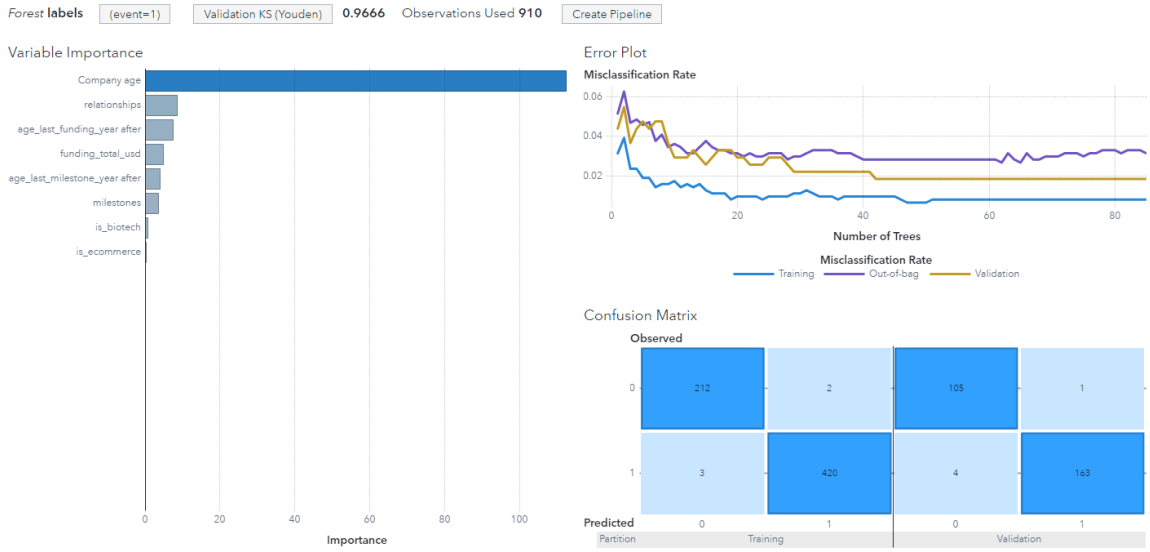
Leaf Size: 5

Predictor Bins: 50

Validation KS(Youden): 0.9449

Misclassification Rate: 0.0440

After tuning Random Forest



Number of Trees: Reduced from 100 to 85

Bootstrap: Reduced from 0.6 to 0.5

Maximum Levels: Increasing the levels from 6 to 16

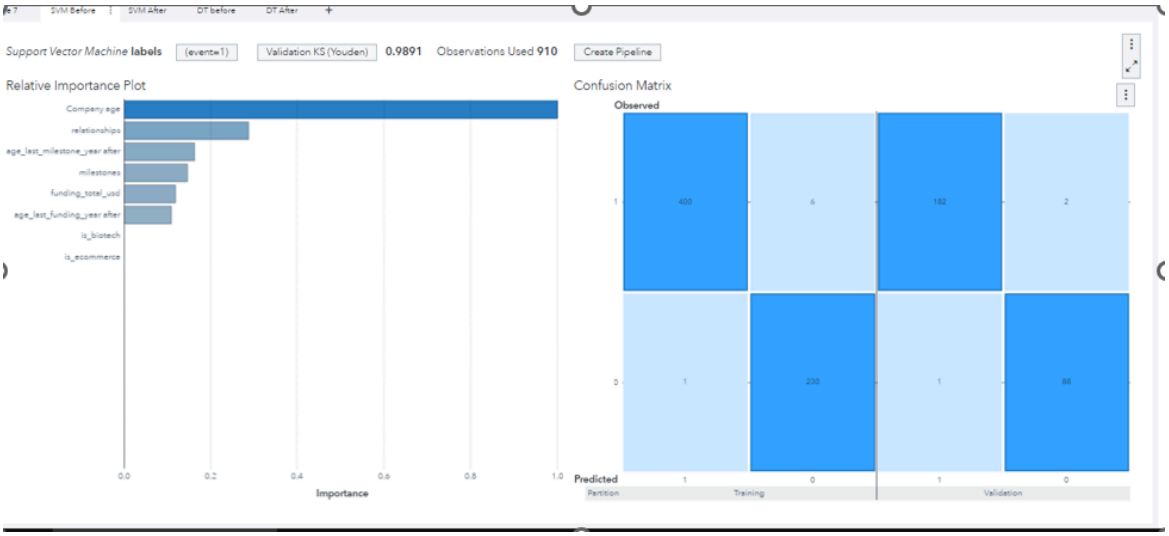
Leaf Size: Reduce the leaf size from 5 to 1

Predictor Bins: Reduce the bins from 50 to 40

Validation KS(Youden) : Increased from 0.9449 to 0.9666

Misclassification Rate: Reduced from 0.0440 to 0.0183

Support Vector Machine: Parik

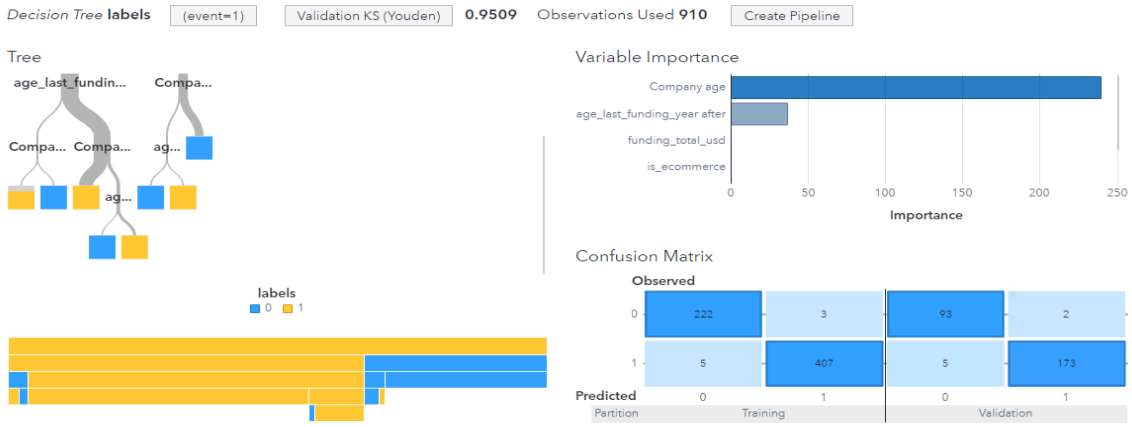


Validation (KS Youden) : 0.9891

Misclassification Rate: 0.0110

Decision Tree: Parik

Before tuning Decision Tree



Maximum Levels: 6

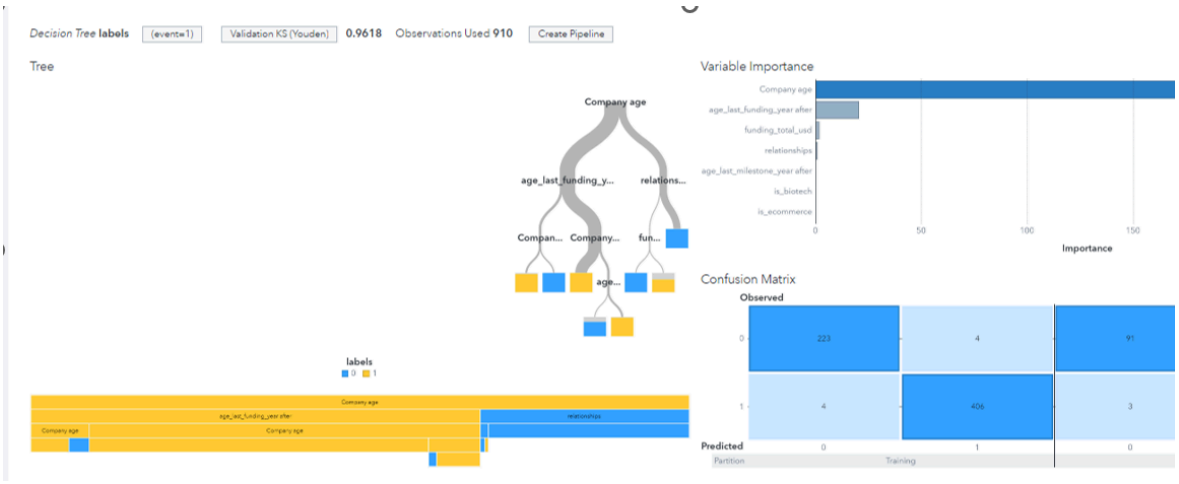
Leaf Size: 6

Predictor Bins: 50

Validation KS(Youden) : 0.9509

Misclassification Rate: 0.0330

After tuning Decision Tree



Maximum Levels: Increasing the levels from 6 to 5

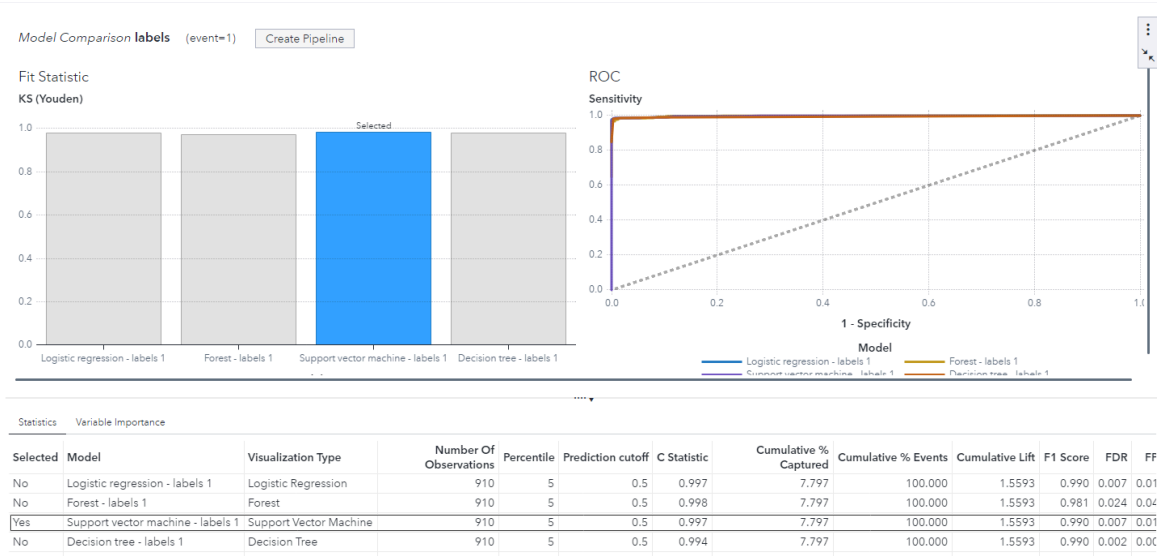
Leaf Size: Reduce the leaf size from 6 to 3.

Predictor Bins: Reduce the bins from 50 to 22.

Validation KS(Youden) : 0.9618

Misclassification Rate : 0.0275

Model Comparison: Jia Jun

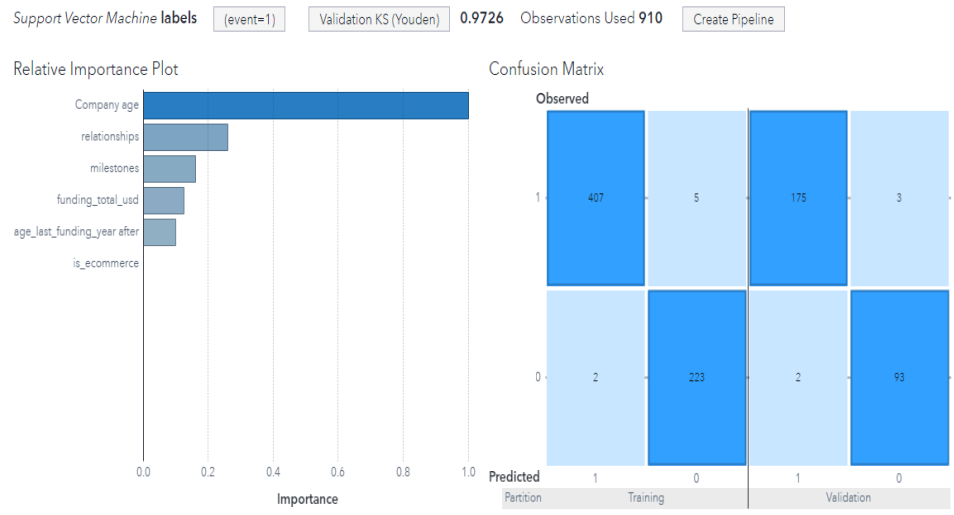


Statistics | Variable Importance

Selected	Model	Visualization Type	Number Of Observations	Percentile	Prediction cutoff	C Statistic	Cumulative % Captured	Cumulative % Events	Cumulative Lift	F1 Score	FDR	FF
No	Logistic regression - labels 1	Logistic Regression	910	5	0.5	0.997	7.797	100.000	1.5593	0.990	0.007	0.01
No	Forest - labels 1	Forest	910	5	0.5	0.998	7.797	100.000	1.5593	0.981	0.024	0.04
Yes	Support vector machine - labels 1	Support Vector Machine	910	5	0.5	0.997	7.797	100.000	1.5593	0.990	0.007	0.01
No	Decision tree - labels 1	Decision Tree	910	5	0.5	0.994	7.797	100.000	1.5593	0.990	0.002	0.00

Looking at the model comparison, the Support Vector Machine gives the most accuracy to determine the success of predicting whether the startup company will be a success or not.

In conclusion



Support Vector Machine is the best performing model as it gives the highest KS Youden.

Company age is the most important variable seen from the relative

importance plot,

As seen from the confusion matrix, the number of predicted values that are predicted wrongly were very few, with 3 out of 178 True in Validation were predicted wrongly and 2 out of 95 False were predicted wrongly