**OPIM 5671**

**Data Mining and Business Intelligence**

**Resume Screening**

Logo, company name

Description automatically generated

University of Connecticut

Spring, 2022

*Submitted By:*

Anisha Biswal

**Table of Contents**

1. **Introduction**
2. **Dataset Description**
3. **Preprocessing**
   1. **Removing unwanted spaces using Python**
   2. **Combining similar categories**
   3. **Text Import**
   4. **Changing the variable role**
   5. **Data Partition**
   6. **Text parsing**
   7. **Text Filter**
   8. **Text Clustering**
   9. **Text rule builder**
4. **Modeling**
5. **Best Model**
6. **Insights**
7. **Conclusion**
8. **References**
9. **Appendix**

**Introduction:**

Resume screening is one of the most time-consuming aspects of recruiting: It is estimated that screening resumes might take up to 23 hours for just one hire. It's no surprise that the majority of talent acquisition directors still think the most difficult component of recruitment is filtering the proper people from a huge applicant pool when a job vacancy receives 250 resumes on average and 75 percent to 88 percent of them are unqualified. To make matters worse, according to a recent survey of talent acquisition leaders, 56 percent want to grow their hiring volume next year, while 66 percent expect their recruiting teams to remain the same size or shrink.

There are thousands of resumes for each job posting and there are dedicated screening officers to screen qualified candidates to each position. There is also a high volume of applicants if the business is labor-intensive, growing, and facing high attrition rates. Therefore, the task of selecting the best talent among many others becomes very hard. For large companies, it consumes a lot of time, labor and money and they do not have time to open every resume. To avoid this, they screen each resume and select resumes based on certain keywords. These keywords are specific to the qualifications and position. This is known as Resume Screening. In this project, a test mining analysis is done on the resume screening that assists the recruiting team to tackle the biggest bottleneck in recruiting. The text mining analysis is done using SAS Enterprise Miner.

**Dataset Description:**

The dataset that has been used here is taken from the Kaggle website. The link of the dataset is given below.

<https://www.kaggle.com/datasets/gauravduttakiit/resume-dataset>

There are 2 variables. They are: Category, Resume

**Category:**  This field describes the position or category in which the resume would fall. This is a nominal categorical variable

**Resume:** Each record in this is a resume which falls into the respective category

The total number of documents that are present in this dataset is **963**

There are numerous categories in this dataset. The categories are provided below.

**Resume Categories:**

Data Science, HR, Advocate, Arts, Web Designing, Mechanical Engineering, Sales, Health & Fitness, Civil Engineer, Java Developer, Business Analyst, SAP Developer, Automation Testing, Electrical Engineering, Operations Manager, Python Developer, DevOps Engineer, Network Security Engineer, PMO, Database, Hadoop,ETL Developer, DotNet Developer, Blockchain, Testing

The resume column will be of type text and the Resume variable will be of type Target.

**Graphical user interface, application

Description automatically generated**

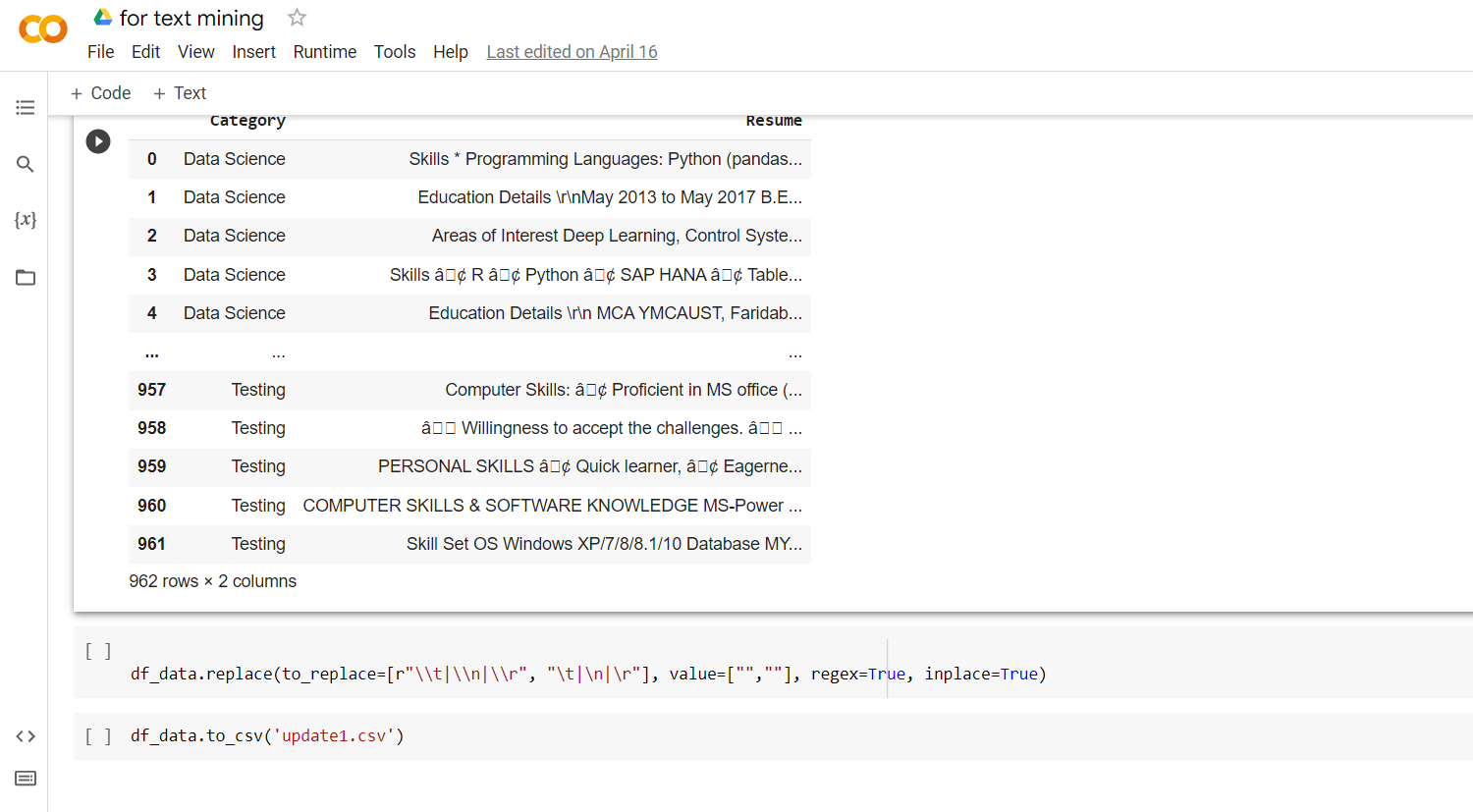
*Figure 1: The variables along with their text properties*

**Preprocessing:**

There are various steps in the preprocessing. The various steps are:

1. **Removing unwanted spaces using Python**

Initially, there were many unwanted spaces and hence the document was not imported into the SAS enterprise miner properly. The dataset that we imported had a lot of line breaks and spacing in the text that we had to analyze. Hence, we used Python code to remove the unwanted spaces and line breaks from the dataset.

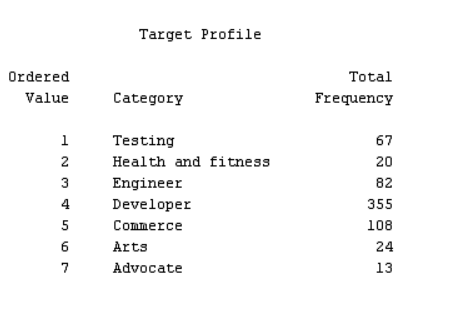


*Figure 2: Snapshot of the python code along with results*

The entire python documentation is uploaded along with other files.

1. **Combining similar categories**

The dataset had more than 10 categories. Since the dataset was not big enough to contain all the categories, a few of these above categories were combined to run a better model. The combined categories along with their counts in the dataset is provided below.



*Figure 3: Various target categories along with their counts*

1. **Text Import**

After removing the spaces and line break, we used **Text Import** node in our diagram to import the dataset.

1. **Changing the variable role**

Changing the role of variable - Our dataset had 2 columns, one being the target variable and other being the text variable that we had to analyse. We change the variable role.

Graphical user interface, application

Description automatically generated

*Figure 4: The variables along with their text properties*

1. **Data Partition**

After careful analysis the dataset was partitioned in the proportion of 70% for training 20% for validation and 10% for testing

Table

Description automatically generated

*Figure 5: The property window of the Data partition node displaying the partition proportion*

1. **Text Parsing:**

Text parsing is used to convert unstructured text into the structured format. And so we have converted the unstructured text into the structured format using a text parsing node attached to the data partition node.

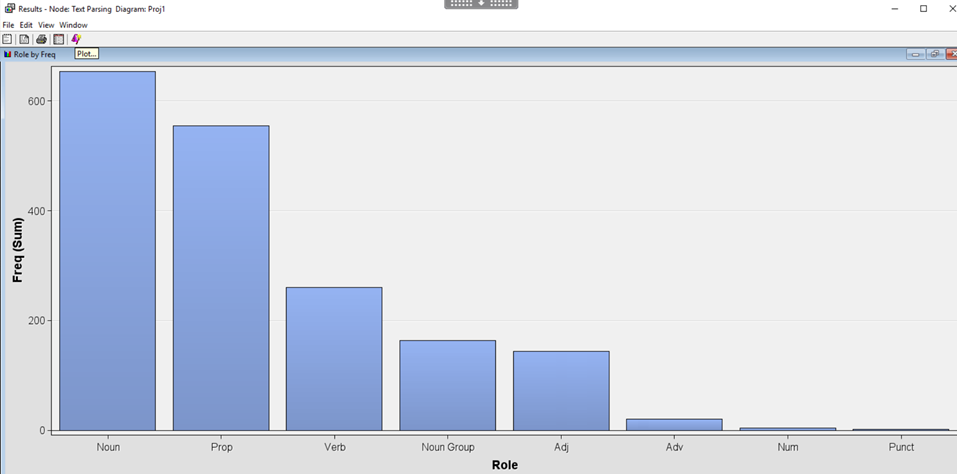
We have created a stop list to stop the non-english greek symbols which were not parsed through the text parsing and weren't useful for the modeling purpose.

Graphical user interface, table

Description automatically generated

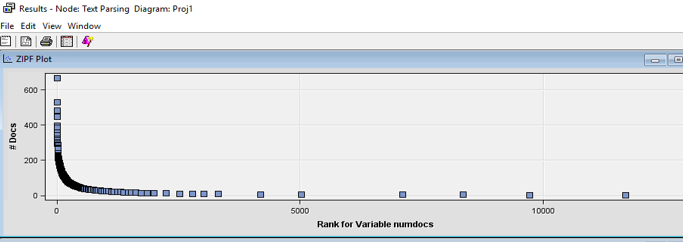
*Figure 6: Stop list that was created*

The start list is default since every word of the resume is important . Also we are not checking any spellings throughout our project since we are assuming every resume is refined. Through the text parsing results we are able to get that there are a large number of nouns in our text greater than 600 as it was expected since we are doing text mining for resumes.

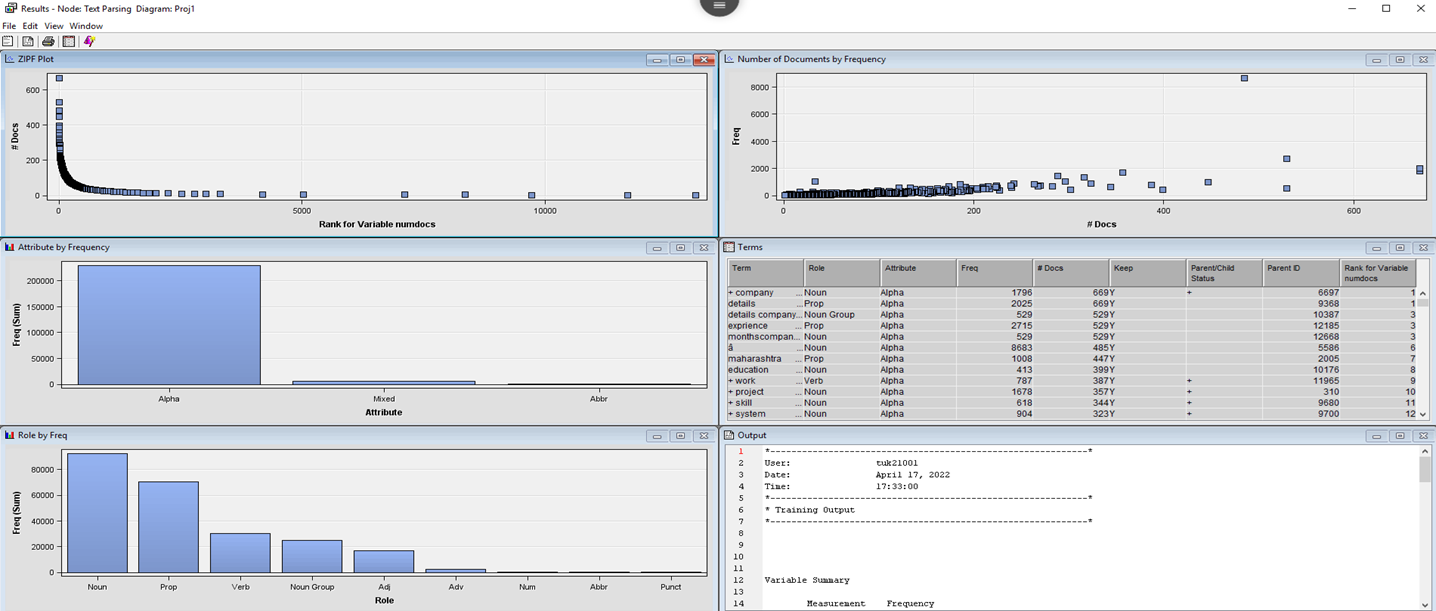


*Figure 7: Text Parsing result*

The zipf plot states that their docs and rank of variables share an inverse relationship .



*Figure 8: zipf plot*



*Figure 9: Text Parsing results*

1. **Text Filter**

To the text parsing node we have added three text filters as in which we have selected the options with varying frequency weights and term weights as they are described below

The frequency weight is set as LOG because the target variable is categorical variable.

* Text filter 1: frequency weight as LOG and term weight as Mutual information
* Text filter 2: frequency weight as LOG and term weight as IDF(Inverse document frequency)
* Text Filter 3 : frequency weight as LOG and term weight as Entropy

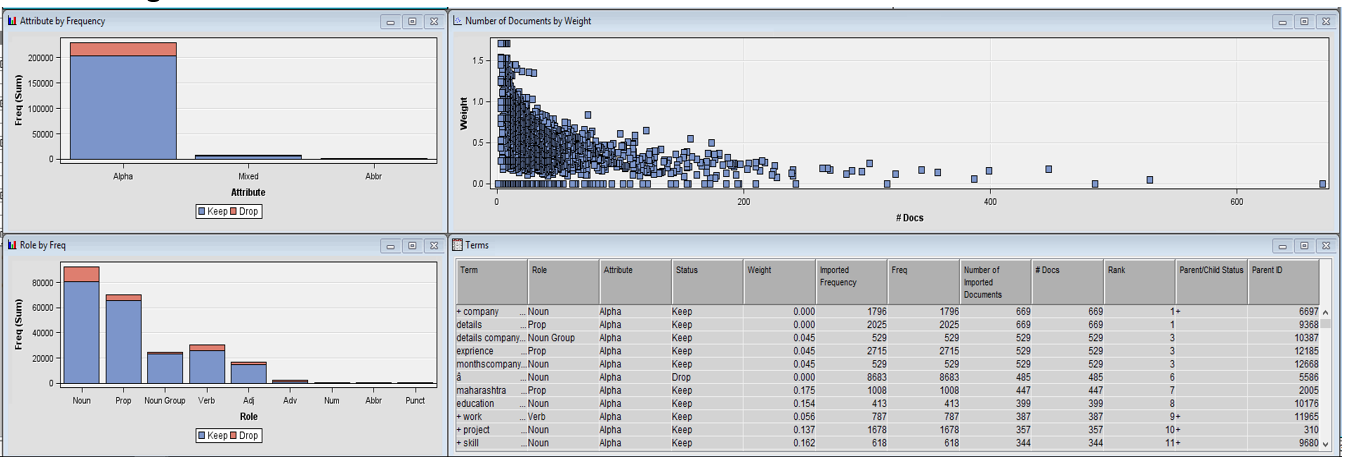
As we want less data loss so we are choosing the minimum number of documents to be 1. Since the target is to maximize data

* Graphical user interface, application

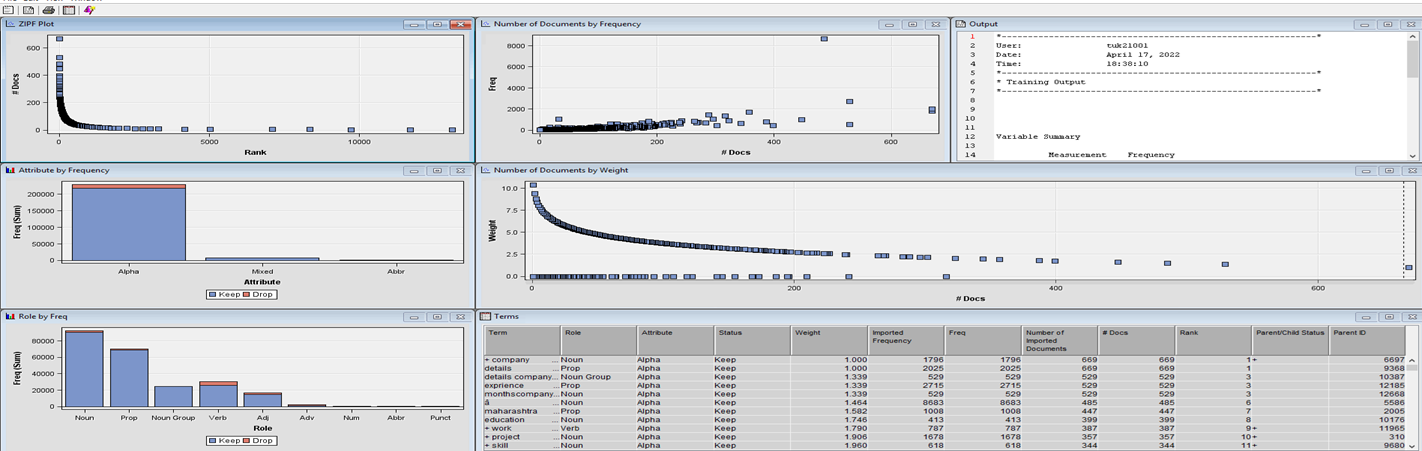
  Description automatically generated

*Figure 10: Property window of text parsing*

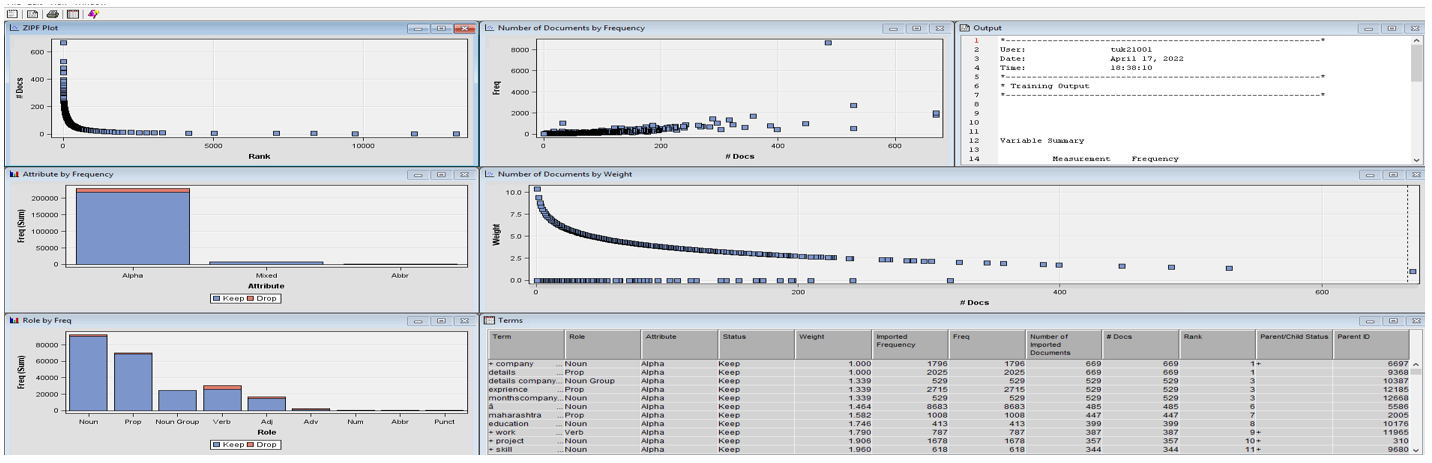
The results of various text filters are provided below:



*Figure 11 : Result of text filter- mutual information*



*Figure 12: Result of text filter -Entropy*

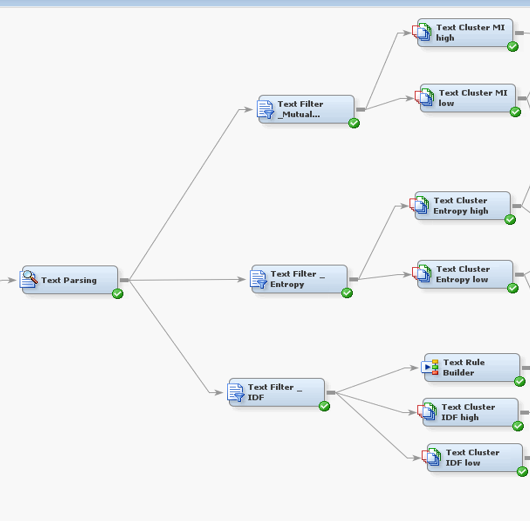


*Figure 13: Result of text filter - IDF*

The mutual information option text filter is giving the best results as it is able to drop more of the unwanted data as compared to other two filters and it is set when the target variable (in our case is category ) is modeled.

1. **Text Clustering**

To each of the three filters node ,two text clusters are joined The text clustering algorithms examine the text to see if there are any natural clusters (groups) in the data. Two clusters at each node of the text filter are being set for SVD as high and low with numbers of clusters being 7 with maximum numbers and the max svd dimensions being set to 100. The goal of SVD is to find the optimal set of factors that best predict the outcome



*Figure 14: Position of Clusters in the diagram*

1. ***Text Rule Builder***

The text rule builder is connected to the text filter IDF directly to the model comparison model as it creates Boolean rules from small subsets of terms to predict a categorical target variable. The m odeling is done after that for each of the text clusters. We are focusing on making 7 clusters for the categories described earlier and trying to fit the best model for all of them.

**Modeling:**

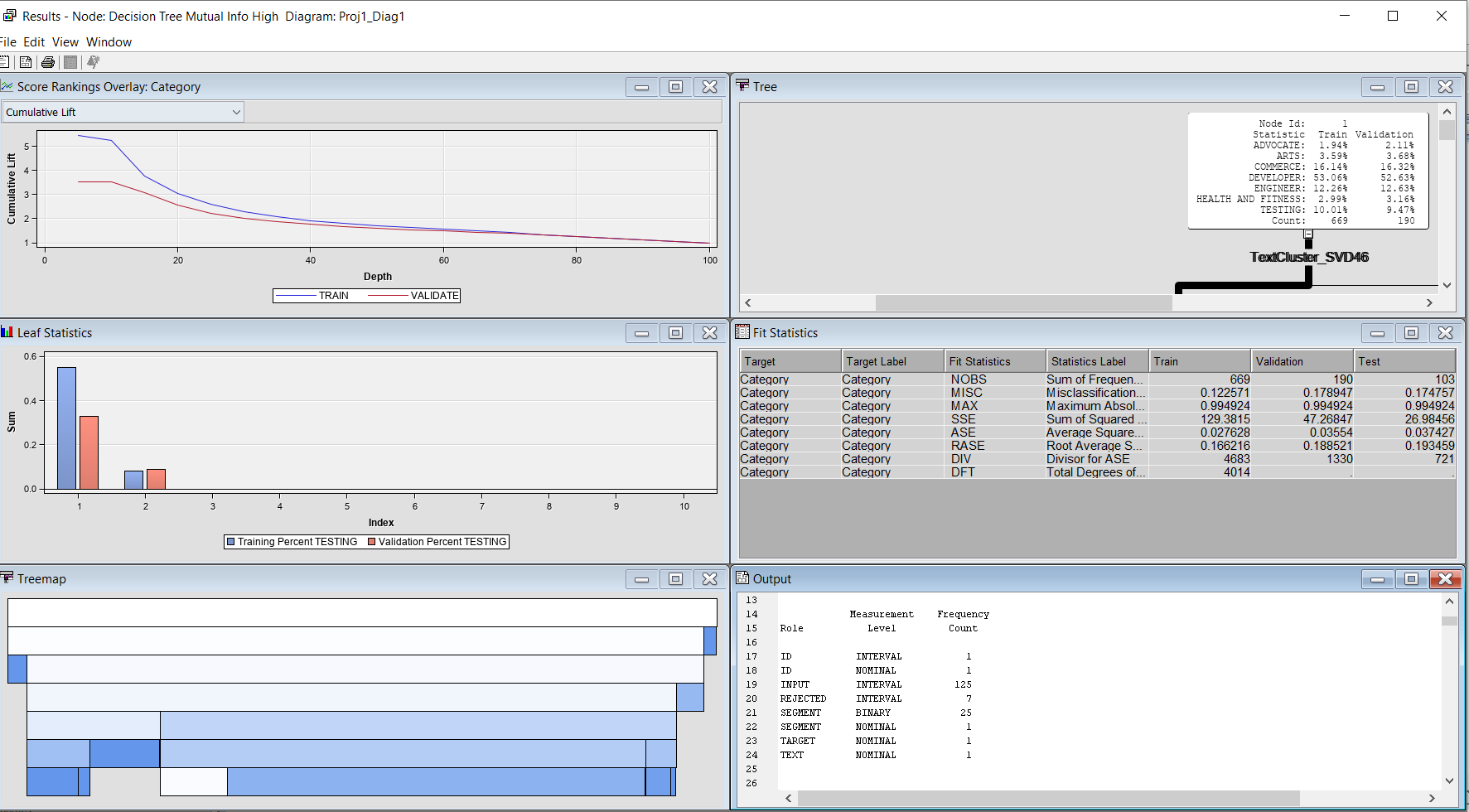
We ran various models on different iterations to conclude the final model. Models help us to visualize the structured data properly and help us to take decisions. For this project our variable is categorical. So we took three different models with different combinations of Text cluster and text filter.

**Model 1:**

The model used here is the decision tree model.

The term weight of this model is Mutual Information

The text cluster of SVD is High and the number of SVD is 7



*Figure 15: Result of Model 1.*

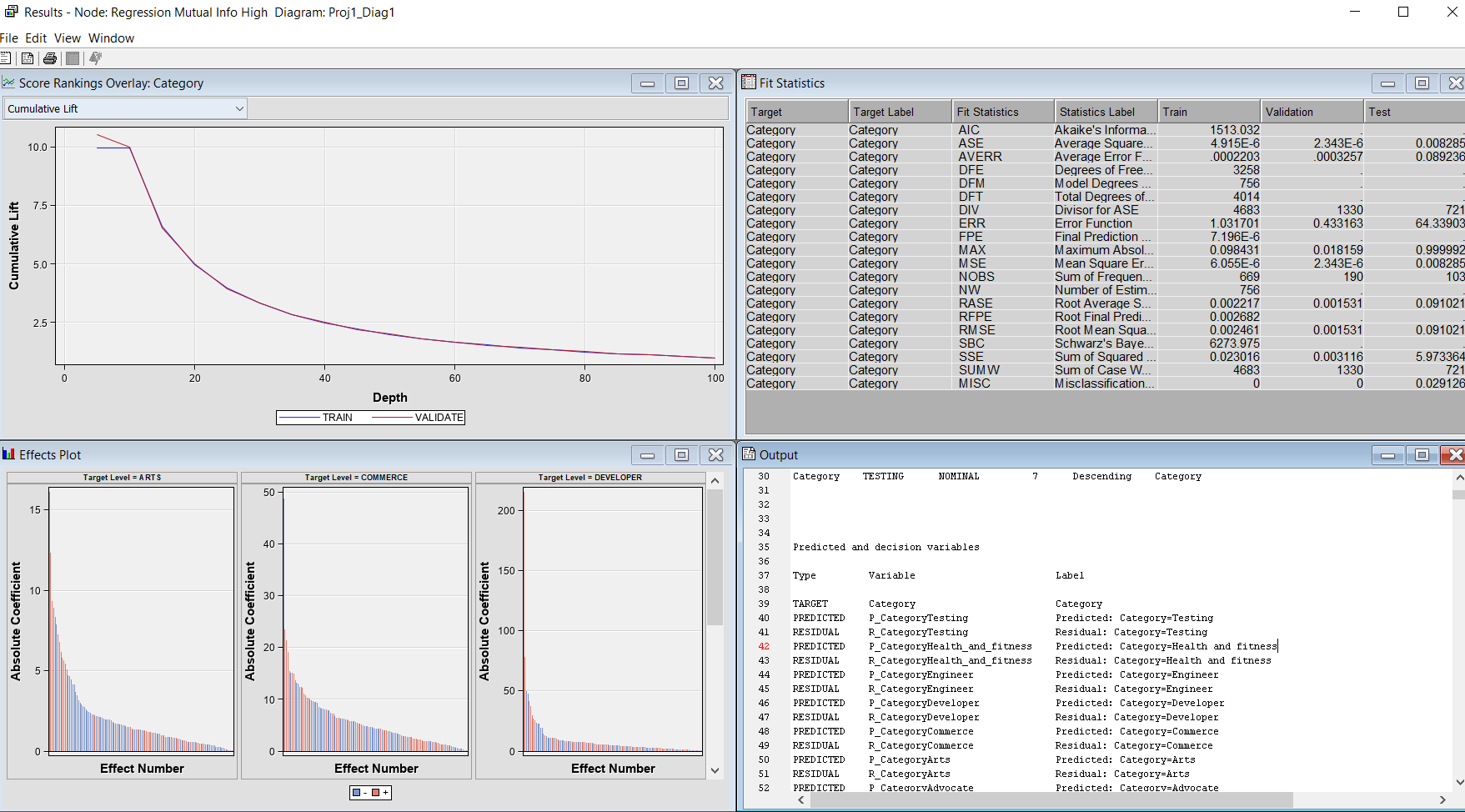
From the results it can be seen that the training sample performs better than the Validation sample

**Model 2:**

The model used here is the regression model.

The term weight of this model is Mutual Information

The text cluster of SVD is High and the number of SVD is 7



*Figure 16: Result of Model 2.*

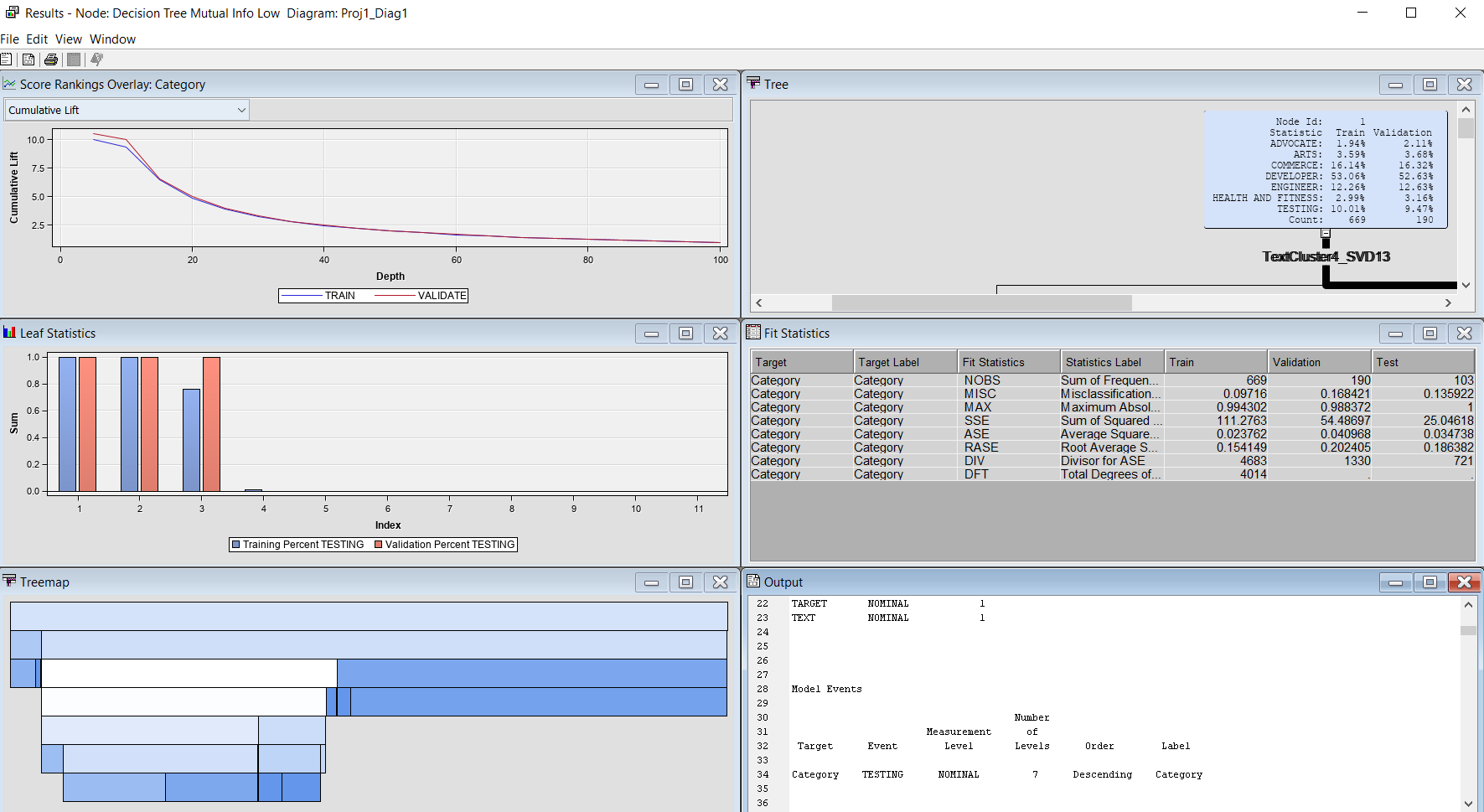
Here the train and the validation sample performs the same.

**Model 3:**

The model used here is the decision tree model.

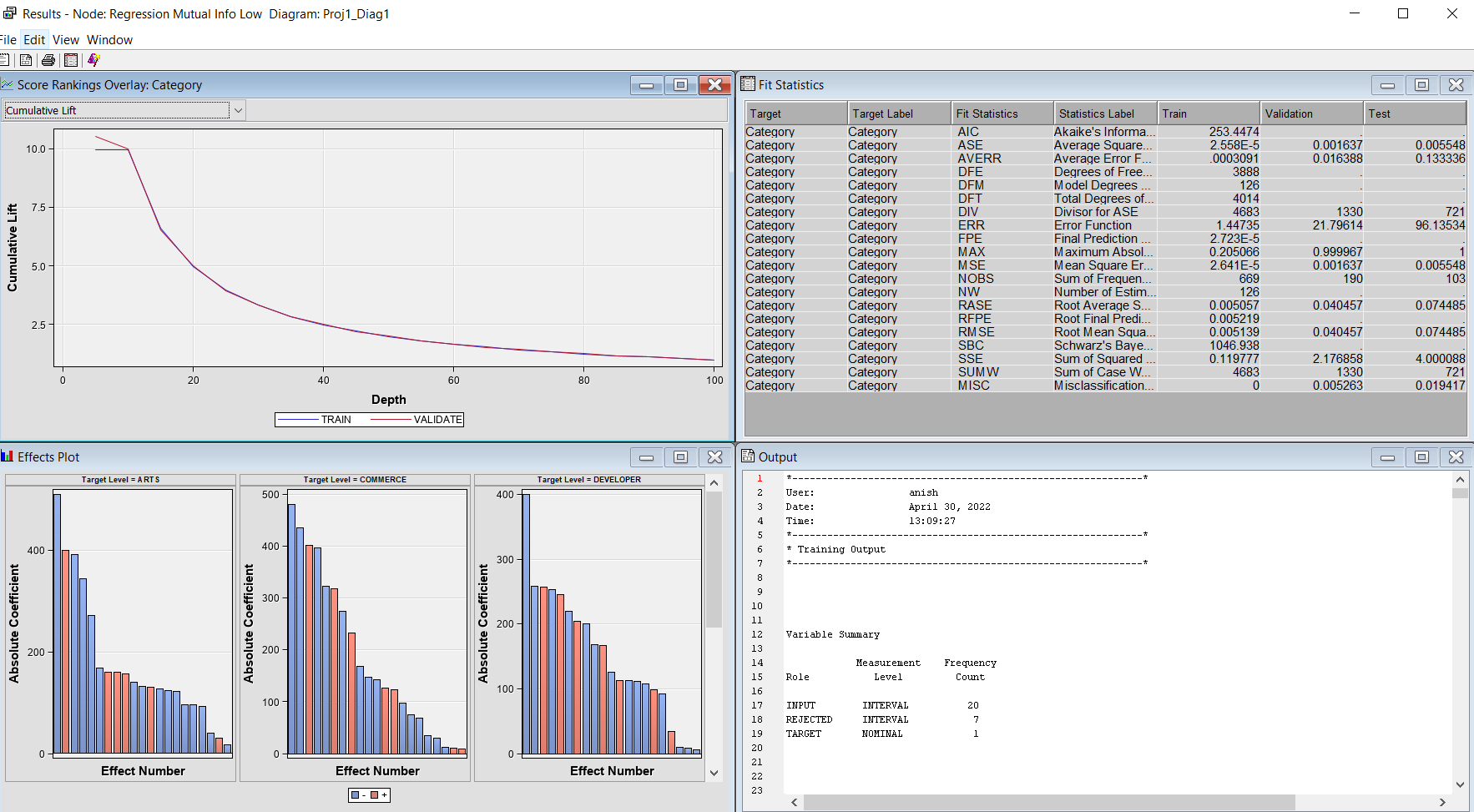
The term weight of this model is Mutual Information

The text cluster of SVD is Low and the maximum number of SVD is 100



*Figure 17: Result of Model 3*

**Model 4:**



*Figure 18: Result of Model 4*

The model used here is the regression model.

The term weight of this model is Mutual Information

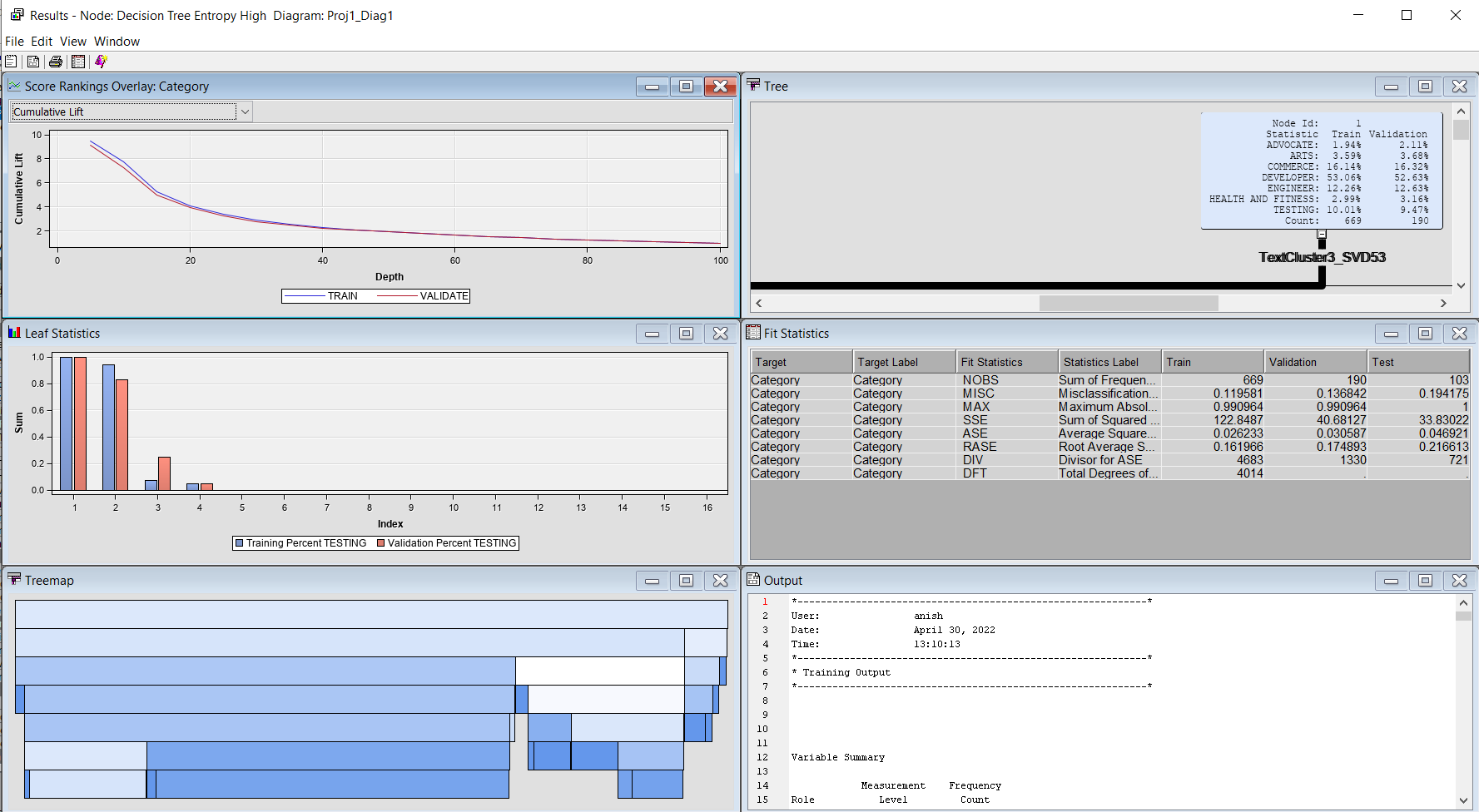
The text cluster of SVD is Low and the maximum number of SVD is 100

**Model 5:**

The model used here is the decision tree model.

The term weight of this model is entropy

The text cluster of SVD is High and the maximum number of SVD is 7



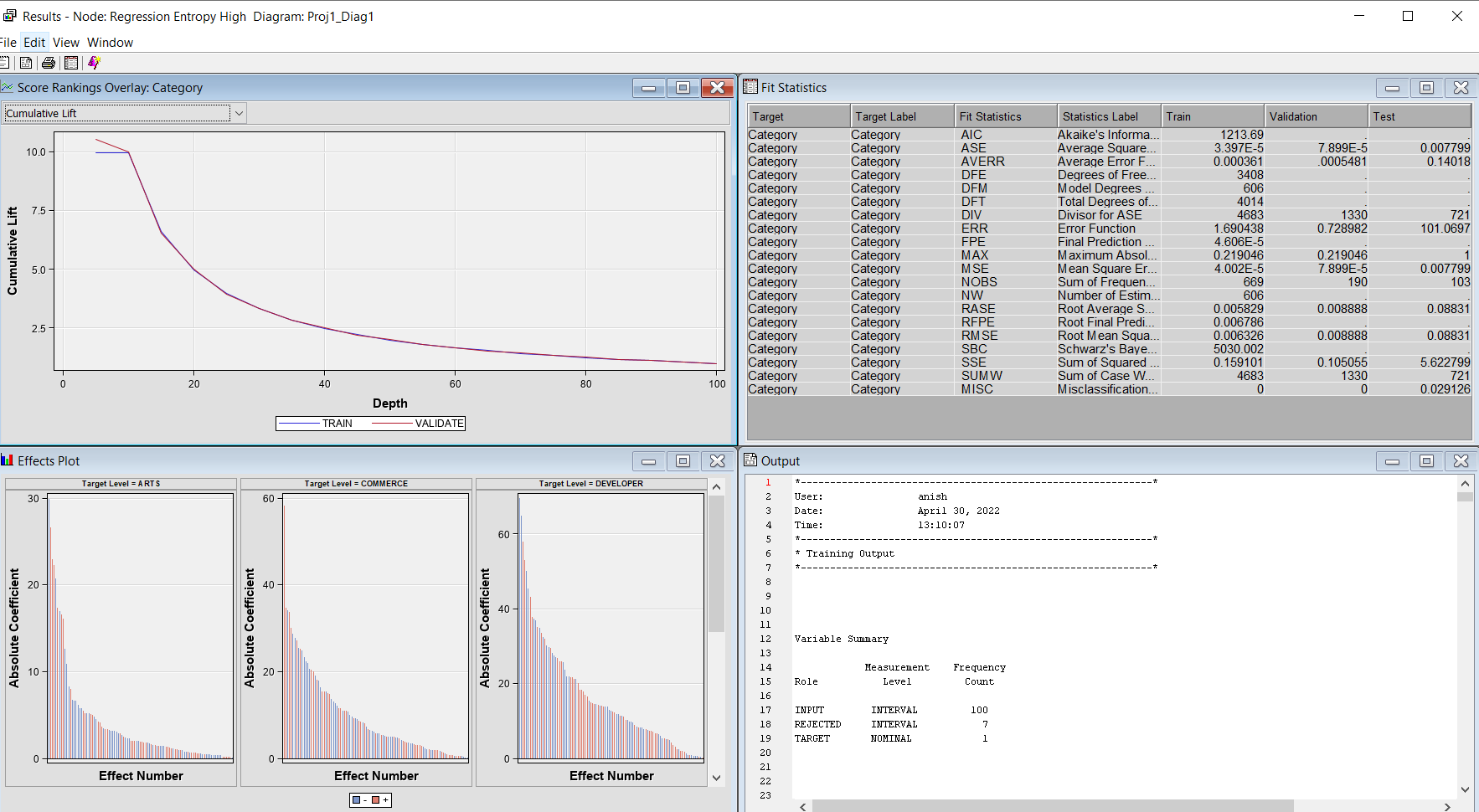
*Figure 19: Result of Model 5*

**Model 6:**

The model used here is the regression model.

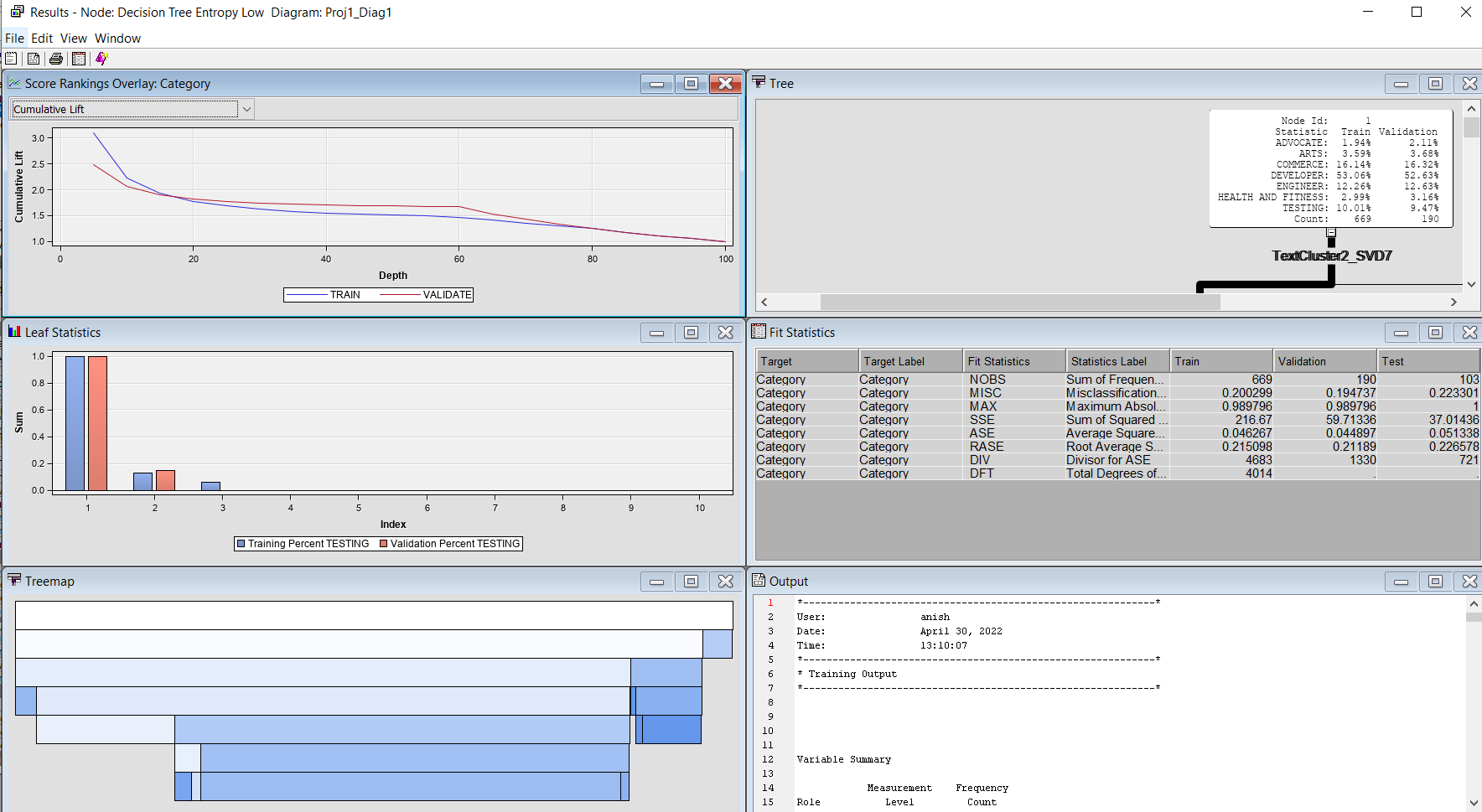
The term weight of this model is entropy

The text cluster of SVD is High and the maximum number of SVD is 7



*Figure 20: Result of Model 6*

**Model 7:**



*Figure 21: Result of Model 7*

The model used here is the decision tree model.

The term weight of this model is entropy

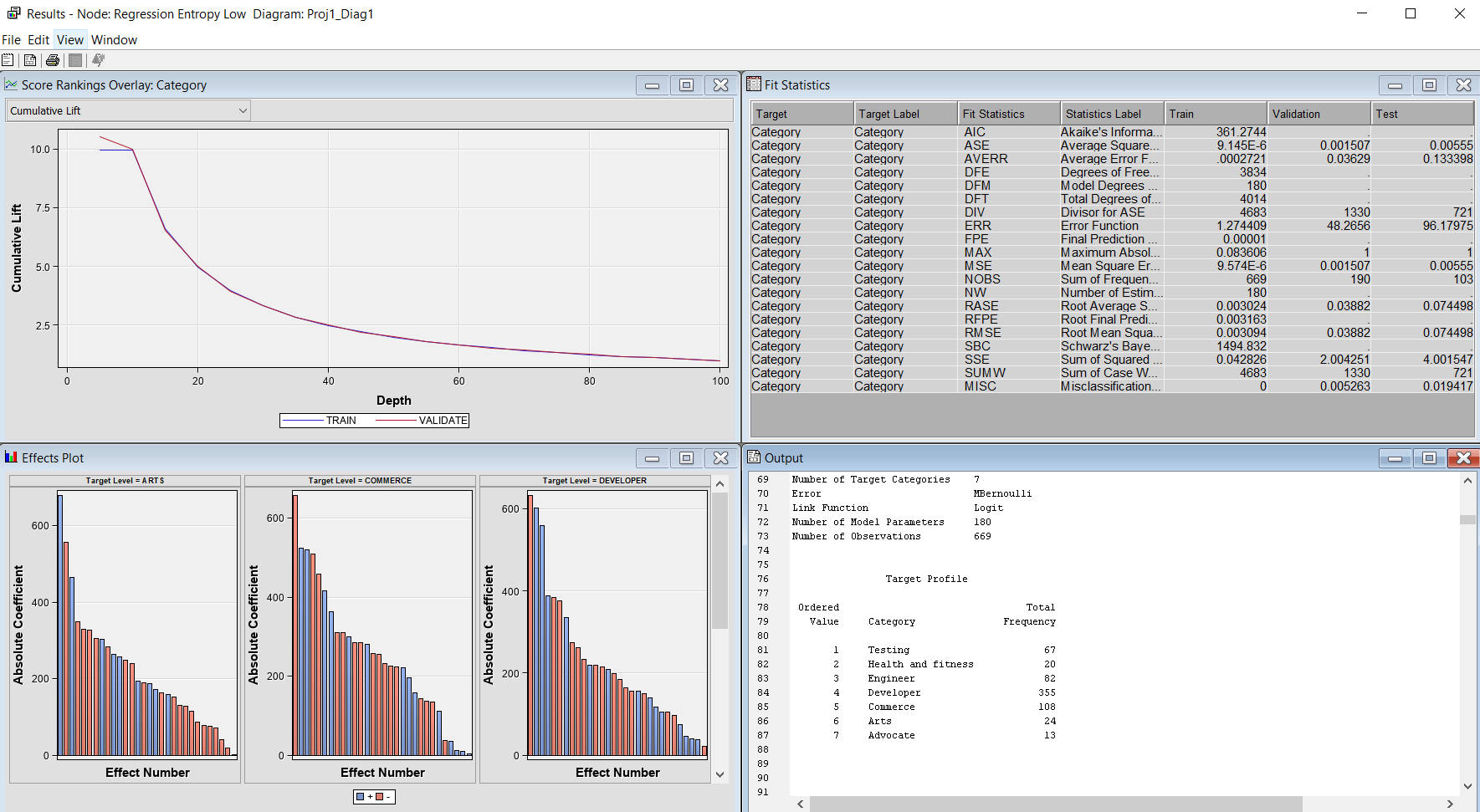
The text cluster of SVD is low and the maximum number of SVD is 100

**Model 8:**

The model used here is the regression model.

The term weight of this model is entropy

The text cluster of SVD is low and the maximum number of SVD is 100

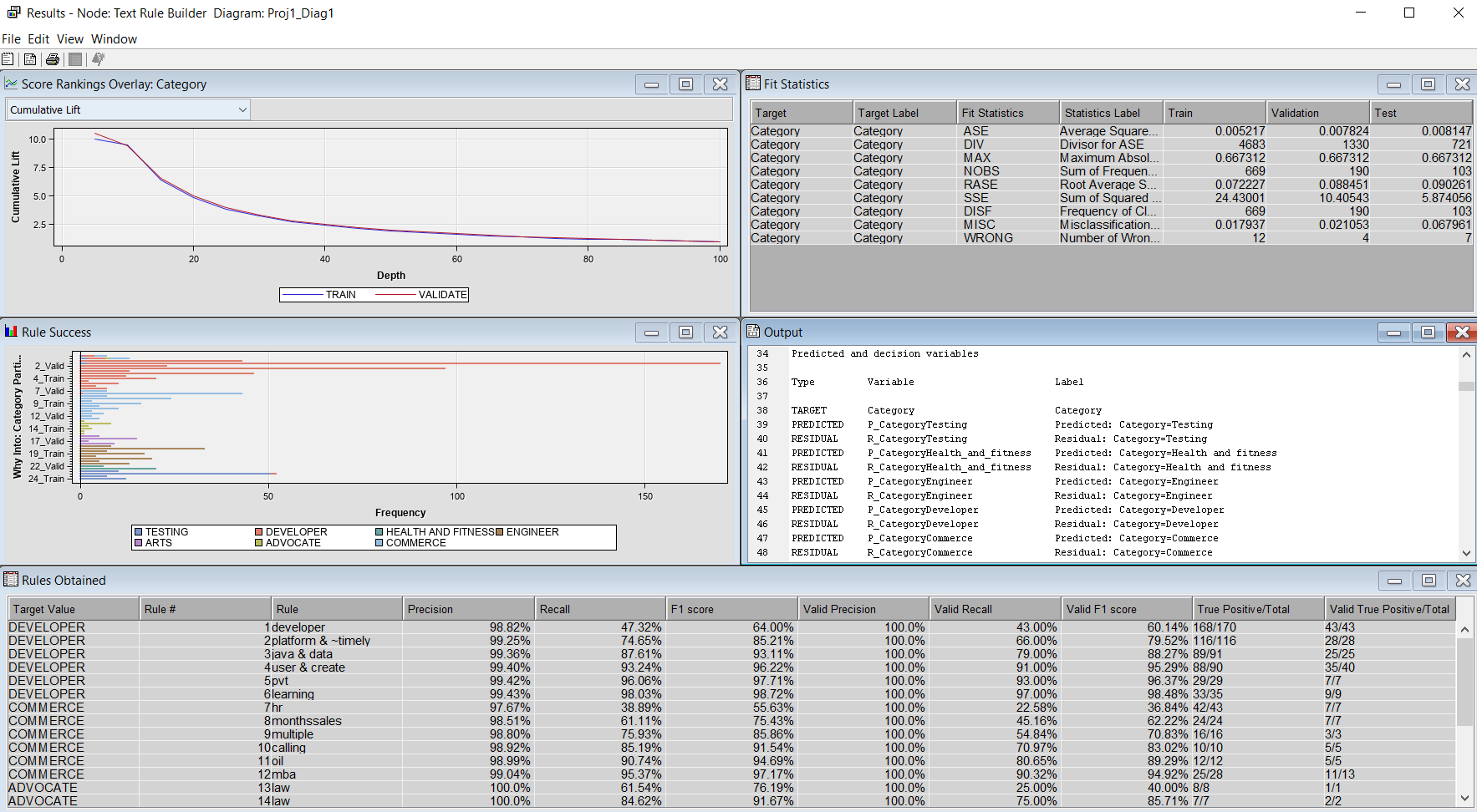


*Figure 22: Result of Model 8*

**Model 9:**

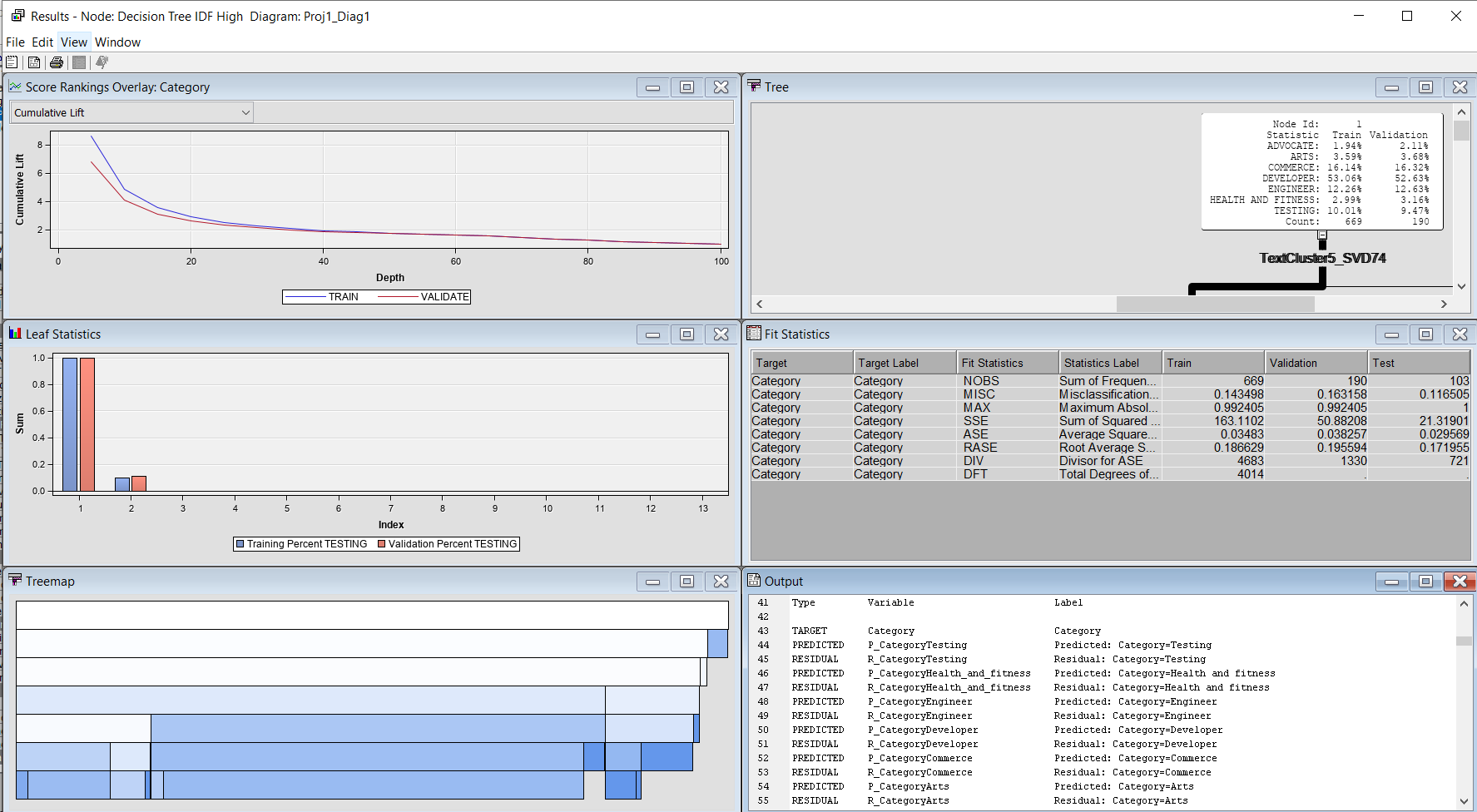
This node comes from a text rule builder

The term weight of this model is IDF



*Figure 23: Result of Model 9*

**Model 10:**



*Figure 24: Result of Model 10*

The model used here is the decision tree model.

The term weight of this model is IDF

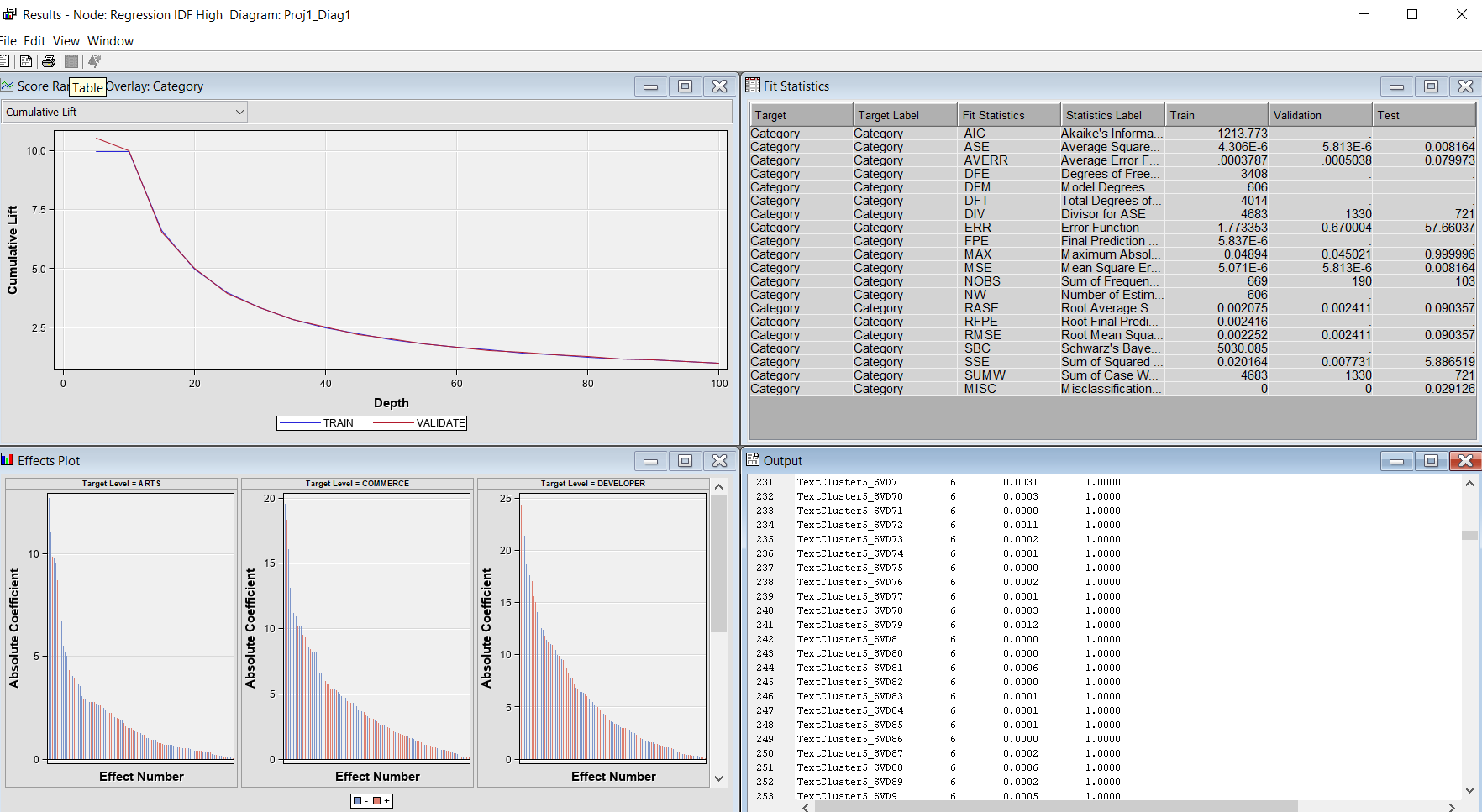
The text cluster of SVD is High and the number of SVD is 7

**Model 11:**

The model used here is the regression model.

The term weight of this model is IDF

The text cluster of SVD is High and the number of SVD is 7



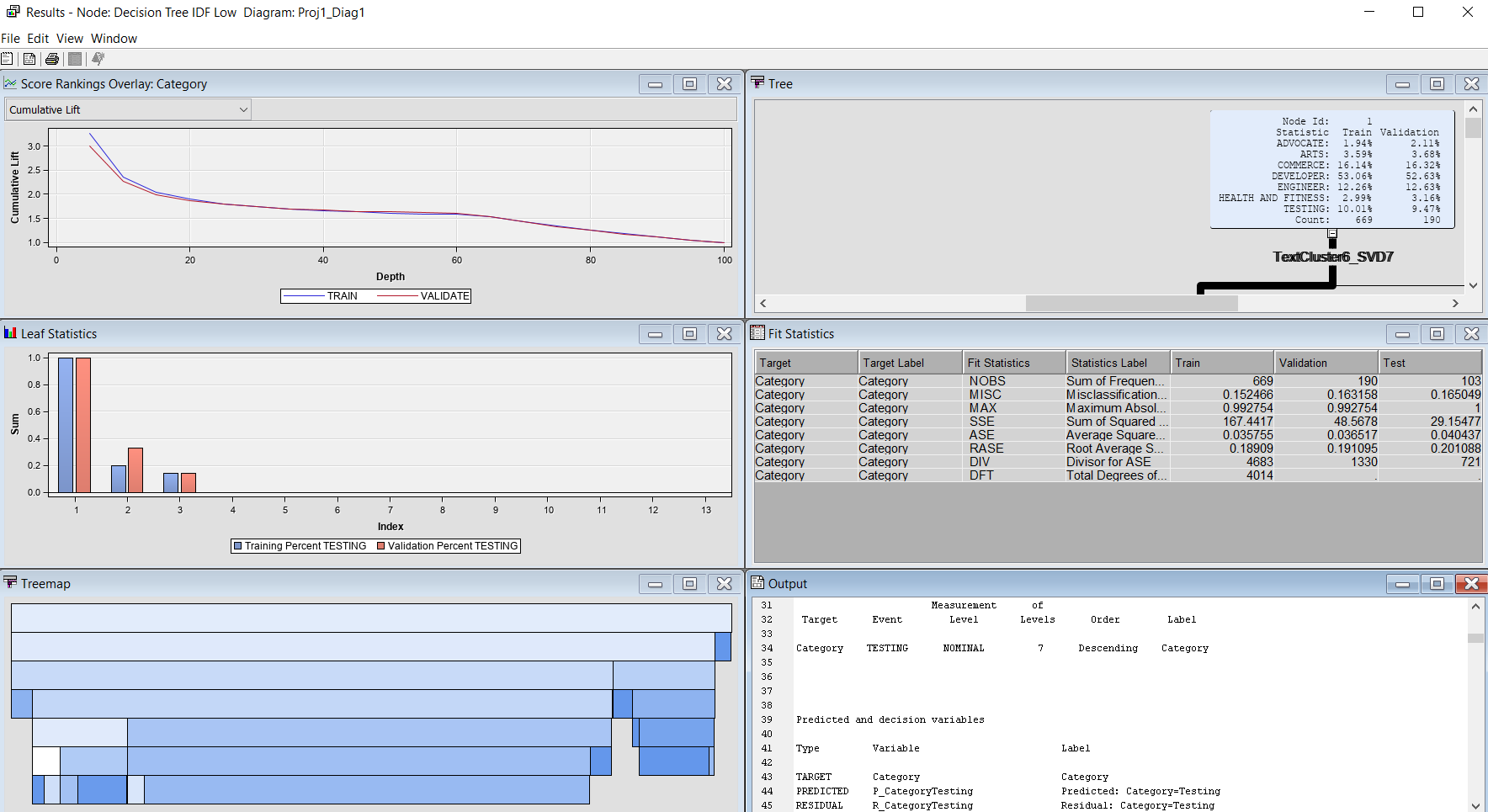
*Figure 25: Result of Model 11*

**Model 12:**

The model used here is the decision tree model.

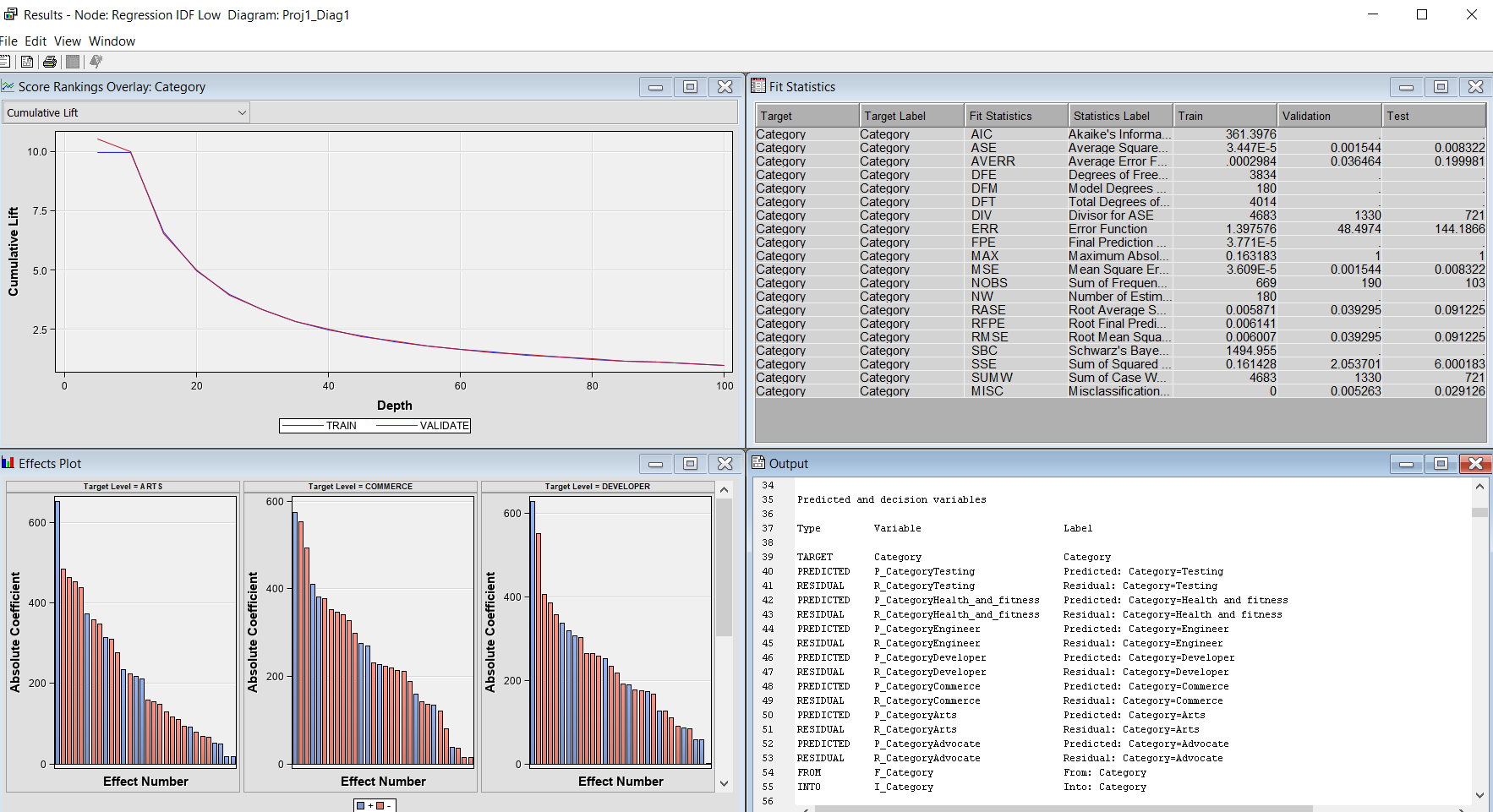
The term weight of this model is IDF

The text cluster of SVD is low and the maximum number of SVD is 100



*Figure 26: Result of Model 12*

**Model 13:**



*Figure 27: Result of Model 13*

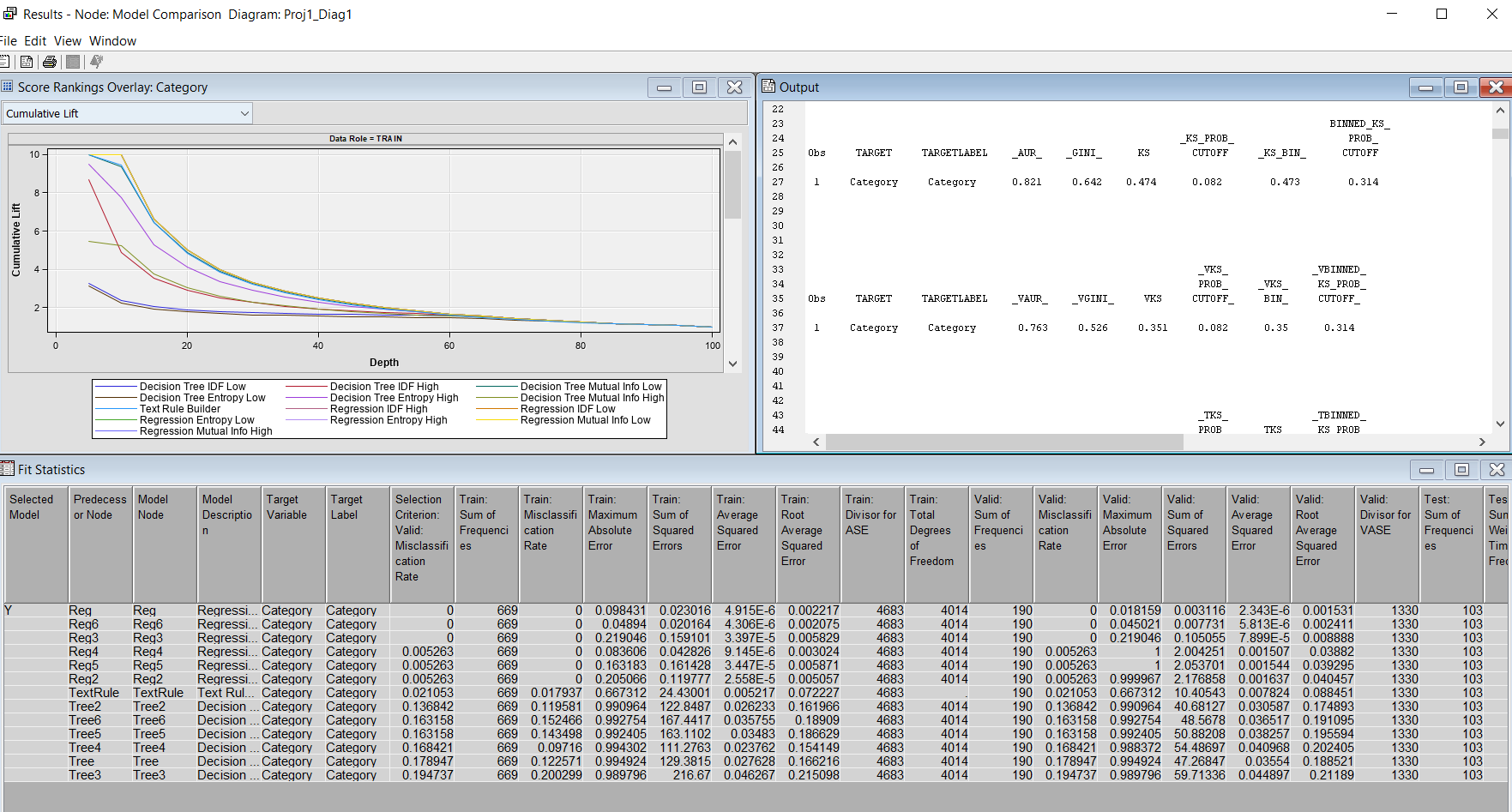
The model used here is the regression model.

The term weight of this model is IDF

The text cluster of SVD is low and the maximum number of SVD is 100

**Model Comparison:**

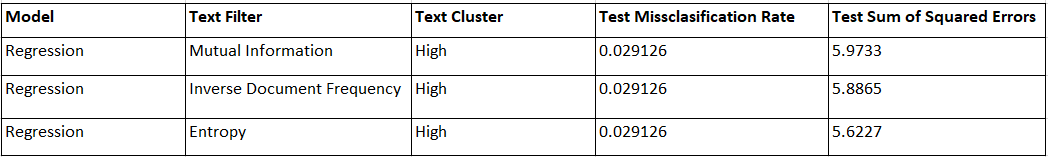
All the 13 models were combined and a model comparison was carried out. The results are given below.



*Figure 28: Results of model comparison*

The ROC, Lift curve and fit statistics are provided in the Appendix section (*Figure 29, Figure 30, Figure 31)* In the Cumulative Lift curve for all the Model’s Test, Train and validation sample is given. In Model Comparison it is clearly shown that out of three types of models, Regression Model performs better, followed by Text Rule Builder and then at the end Decision Tree Models.

To check the best model, we compare factors like misclassification rate of test sample and Error Metrics. As shown in the above screenshots, three models has same Test misclassification rate as below:



After Comparing the Misclassification Rate and Error Metrics, The best model for our data set is **Regression model where the term weight is Entropy and SVD is High.**

**Insights/Recommendations:**

1. If a resume contains these words, it has a high chance to pass the screening for the Developer/ Analyst positions.

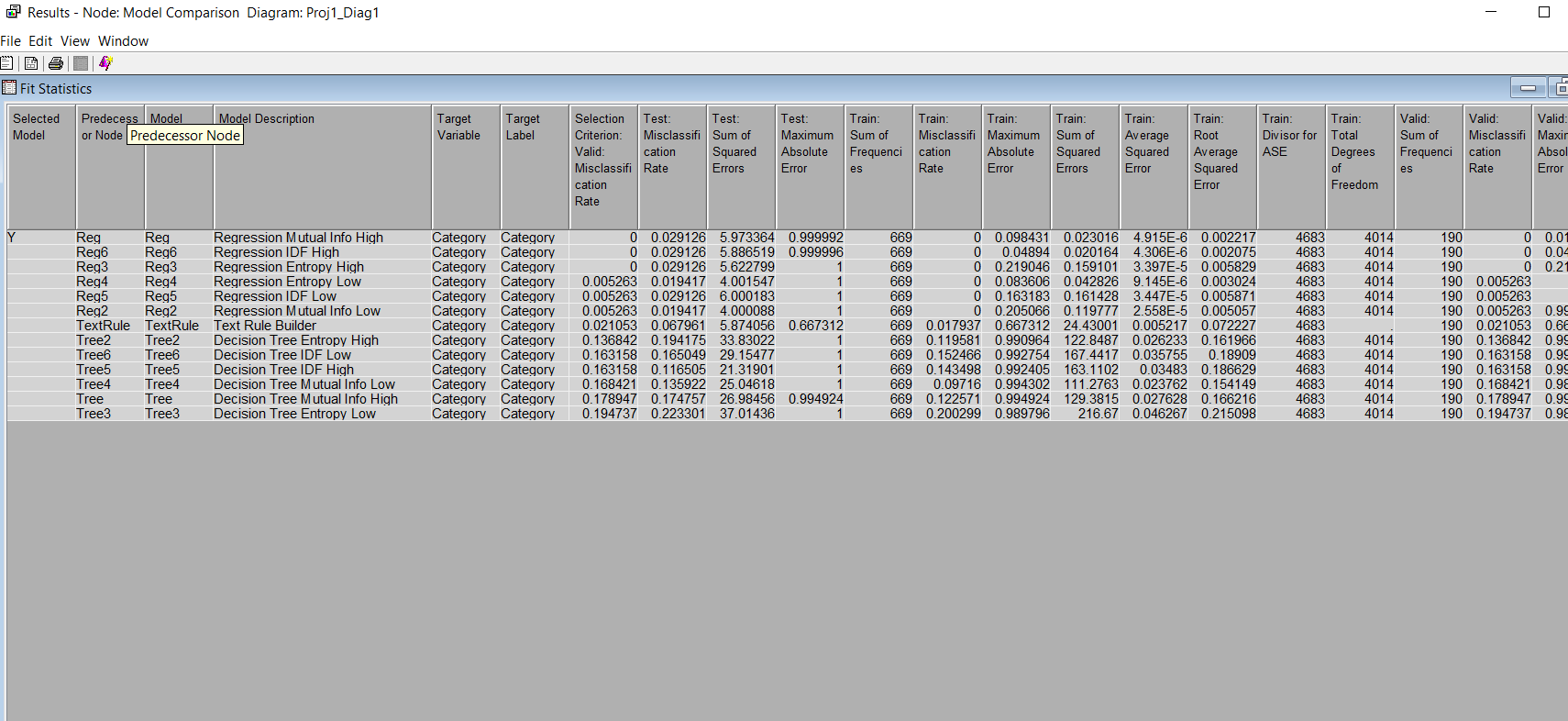
The keywords picked up for developers and analysts are development , data, web, sql +application +database +create +develop java windows developer +test +design +responsibility +project (Figure 32)

1. This will help in resume shortlisting for Tester positions. The keywords for testing positions are +switch +network troubleshoot +upgrade troubleshooting +configuration firewall network security +device +firewall +lan routers networking +switch +network troubleshoot +upgrade (Figure 32)
2. The keywords for commerce positions are +review monthly +manager +plan responsible +schedule +document +office +site management +report +process +good people +service +review monthly +manager +plan responsible (Figure 32)
3. The keywords for engineering profile are power, company, electrical, people , fitness, +site personal engineer maintenance engineering college handling +experience skill diploma power company electrical people fitness (Figure 32)
4. This can be used for shortlisting profiles from portals like Linkedin Engineering profile.

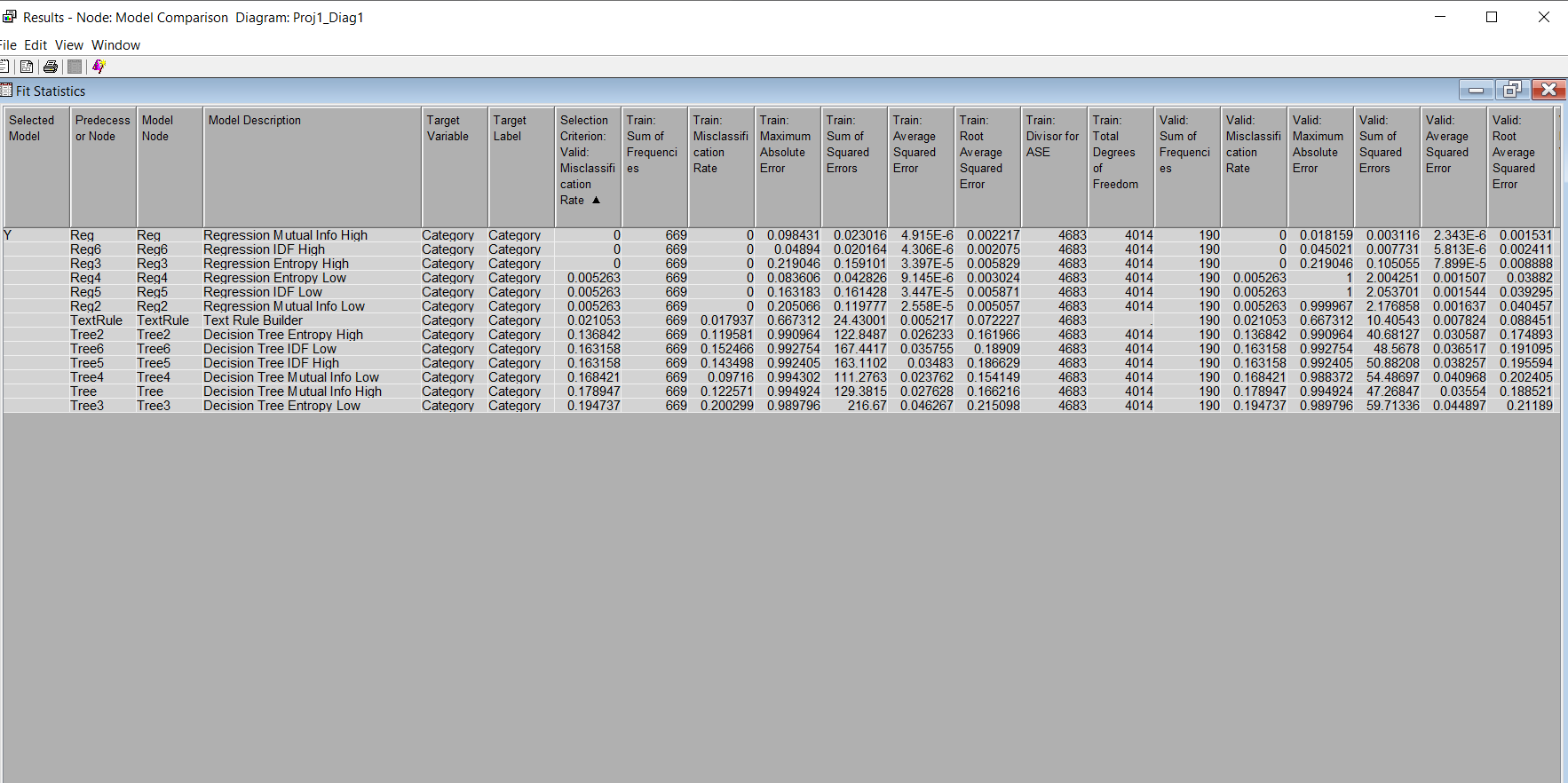
**References:**

1. <https://www.kaggle.com/datasets/gauravduttakiit/resume-dataset>
2. <https://ideal.com/resume-screening/>

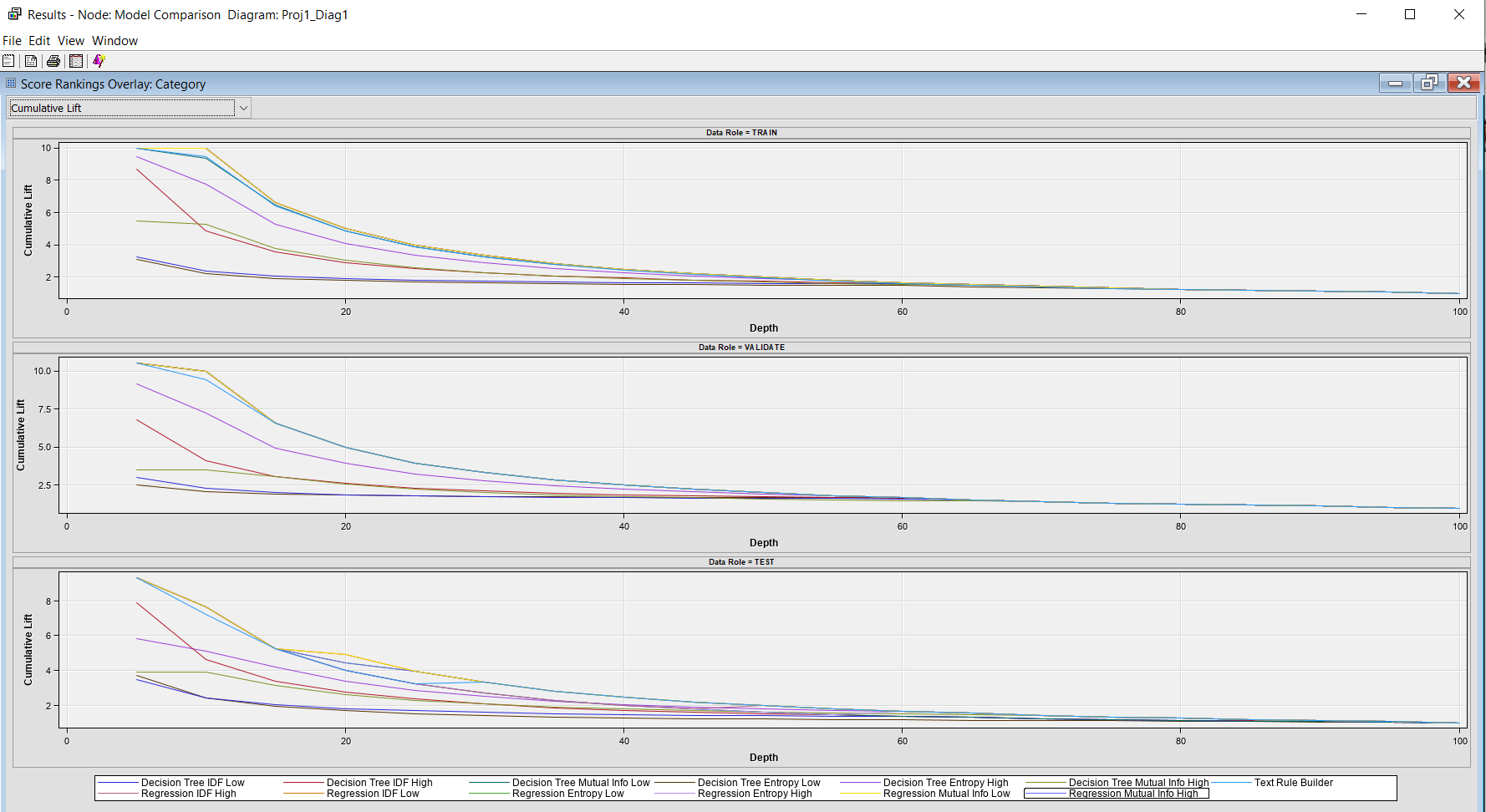
**Appendix:**



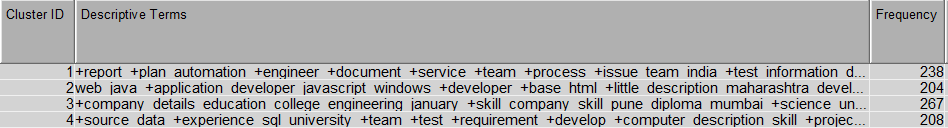
*Figure: 29*



*Figure: 30*



*Figure: 31*



*Figure: 32*