**( 1 ) Bayesian Belief Network**

**Objective : Construct a Bayesian Belief Network model using the given user knowledge dataset using “pomegranate” python package/library to analyse the User knowledge data for decision making.**

**Exploratory Data Analysis:**

* **Used “pandas\_profiling” library to understand the data and correlation between them.**

****

**This creates the html file containing reports on the type of data and various other statistics around the data. It also contains different correlation matrices (Pearson, Spearman’s, Kendall’s, Phik) on the dataset.**

* **Basic check on data for NULL values, describe() stats were generated for the data to check on any “outlier”.**

**Data Preprocessing:**

* **Replaced “extra space” in the column names.**
* **Converted categorical values of target variable “UNS” to numerical as needed.**

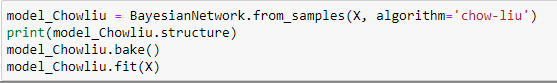
****

1. **Constructing the Bayesian Belief Network model:**

* **Assuming the edges between variables (nodes).**

****

* **Model constructed using “from\_samples” method and “chow-liu” algorithm**

****

**We used all 4 algorithms (greedy, exact, exact-dp and chow-liu) and compared for different input values and response value. The “chow-liu” algorithm gave us optimal result for the below probability prediction and inference.**

1. **Predicting the probability of the user having the following characteristics : 0.0**

**STG - Study time for Goal Objective : 0.09**

**SCG - repetitions for Goal Objective : 0.15**

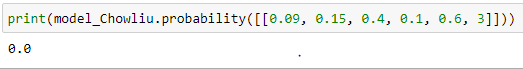
**STR - study time for relative objective : 0.4**

**LPR - Performance of user for relative objective : 0.1**

**PEG - Performace of user in goal objective : 0.6**

**UNS - 'middle' level performer**

**Probability is 0.0 for the given input.**

****

**3) Predict the type of the user’s knowledge with the following characteristics:**

**STG - Study time for Goal Objective : 0**

**SCG - repetitions for Goal Objective : 0**

**STR - study time for relative objective : 0.5**

**LPR - Performance of user for relative objective : 0.2**

**PEG - Performace of user in goal objective : 0.85**

**Correctly predicted the class as ‘1’ which is ‘High’**

****

**4) Infer the probability for the data :**

**STG - Study time for Goal Objective : 0.8**

**SCG - repetitions for Goal Objective : 0**

**STR - study time for relative objective : 0.78**

**STR - study time for relative objective : 0.8**

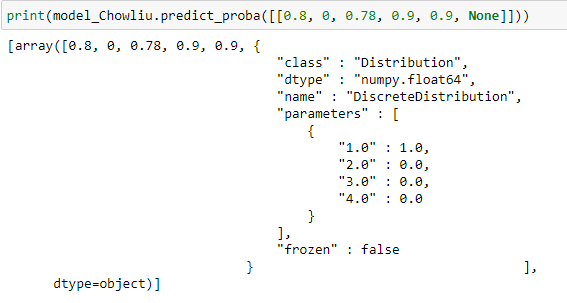
**LPR - Performance of user for relative objective : 0.9**

**PEG - Performace of user in goal objective : 0.9**

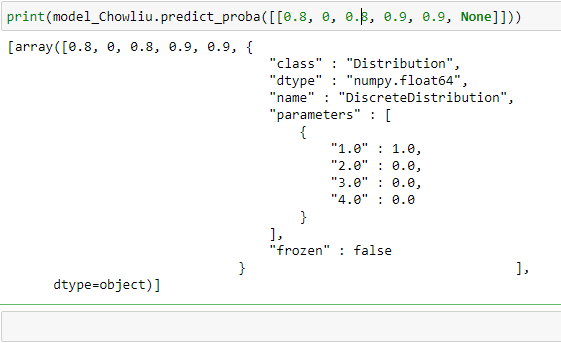
**UNS - 'middle' level performer**

**Probability for the given data with Target “3.0 - middle level” is 0.0 (given below)**

**STR - study time for relative objective : 0.78**

****

**STR - study time for relative objective : 0.8**

**Probability for the given data with Target “3.0 - middle level” is 0.0 (given below)**

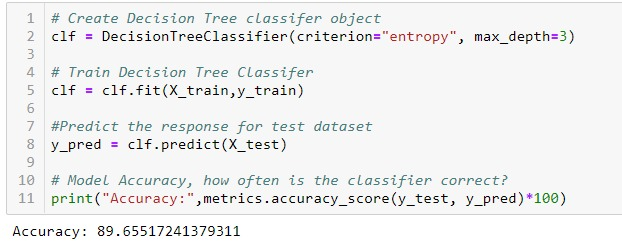
**( 2 ) Prolog**

**Created rules using the Decision Tree :**

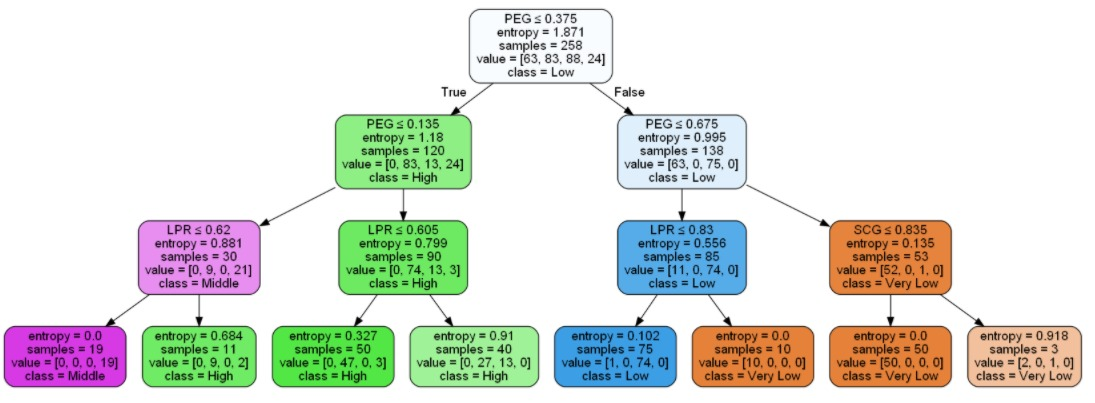
**We used sklearn’s DecisionTreeClassifier library to generate the decision tree for the given data set.**

**Decision Tree :**

* **Selected “entropy” as the criteria to create the tree with the max\_depth = 3.**

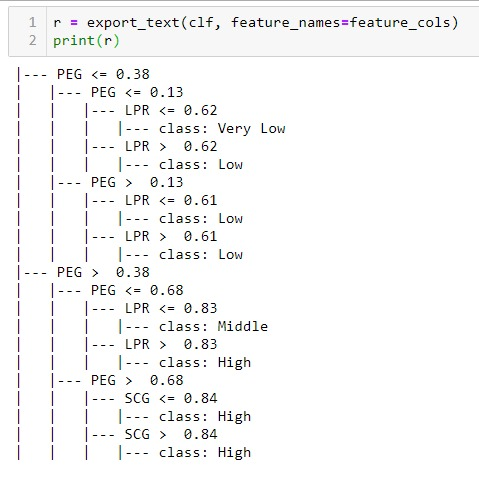
****

**Decision Tree**

****

**Rules:**

* **Used “export\_tree” library of sklearn to generate the “rules” from the above decision tree**

****

**Prolog Knowledge Base:**

***user\_know\_level(PEG,LPR,\_,\_,\_,'Very Low'):-not(PEG>0.38),not(PEG>0.13),not(LPR>0.62).***

***user\_know\_level(PEG,LPR,\_,\_,\_,'Low'):-not(PEG>0.38),not(PEG>0.13),(LPR>0.62).***

***user\_know\_level(PEG,\_,\_,\_,\_,'Low'):-not(PEG>0.38),(PEG>0.13).***

***user\_know\_level(PEG,LPR,\_,\_,\_,'MiddLe'):-(PEG>0.38),not(PEG>0.68),not(LPR>0.83).***

***user\_know\_level(PEG,LPR,\_,\_,\_,'High'):-(PEG>0.38),not(PEG>0.68),(LPR>0.83).***

***user\_know\_level(PEG,\_,\_,\_,\_,'High'):-(PEG>0.38),(PEG>0.68).***

**Type of knowledge level for the user with below Characteristics :**

**As per the prolog KB, the input is in the format :**

**user\_know\_level(PEG,LPR,SCG,STG,STR,CLASS).**

**Input : STG : 0.1**

**SCG : 0.1**

**STR : 0.7**

**LPR : 0.15**

**PEG : 0.9**

**UNS : ?**

****

****

**user\_know\_level(0.9,0.15,0.1,0.1,0.7,C).**

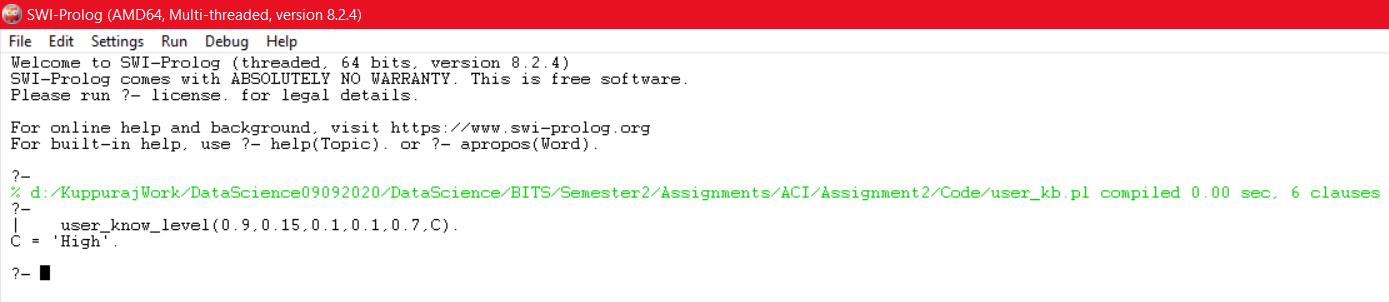
**Output :**

**?-**

**| user\_know\_level(0.9,0.15,0.1,0.1,0.7,C).**

**C = 'High'.**

**?-**

****