Project: Gyan CRM Mentorship Platform**-Suite (GYANCRM-S)**

Subject Area: Gyan CRM Mentorship Platform Tools (GYANCRM-T)

Authored on: 27/01/2022, and still work in progress as we at AOEC focused on the solution finding than coding the Gyan CRM Map Component

Team: AOEC and Senior Faculties of Sri Sharada Tutorials and DR's Academy

Team members for the Gyan CRM Map Component and Team Suite:

1. K.S.Venkatram 2. Abhiram and 3. Aakkash K V

For: Hackers Earth CTR-ALT-DEBT

Submission: Gyan CRM Mentorship Platform

Problem solving: Help connect students (or mentees) to volunteering tutors (or mentors) via a Gyan CRM Map Component and Mentorship Platform that designs methodologies to address the problem statement of Gyandaan and addresses the Classroom-Teacher-Student connect dynamics problem due to the pandemic, post pandemic and social distancing scenarios.

About AOEC

AOEC stands for Akaash Open Enterprise Centre (a Gap analysis and problem solving consultancy) with a team comprising of myself (K.S.Venkatram), Abhiram (Technical consultant and Operations Advisor) and Aakkash K V (BTECH Automotive Engineering).

We start with reviewing the Gyandaan problem statement.

Gyandaan Problem Statement

CTRL ALT DEBT

Started: **Dec 15, 2021 06:00 AM IST**

Ends on: Jan 27, 2022 11:59 PM IST

Gyandaan is analogous to Shramdaan, which means voluntary contribution of knowledge. Gyandaan is made of two words, 'Gyan' means knowledge and 'daan' means donation. It means a voluntary contribution of citizens towards community education involving teaching, mentoring and helping someone in academics. It is a way of helping our society and contributing for the overall upliftment of the students specially from the underprivileged and poor backgrounds who can't afford to enrol themselves into coaching classes and other courses.

Design and develop a platform where students enrol to learn and obtain mentorship and volunteers enrol themselves to help / mentor those students.

Students mention what subjects / career goals they need help with or need mentorships. The topics can be anything from a specific topic like "trigonometry" to a broader subject like "Class 10th CBSE Math". Students can ask help for more than one topics with their preferences like time & day of the week.

Volunteers mention the areas of their expertise and their availability in a week. They can specify maximum hours they are willing to volunteer in a week.

The platform does the matching of students to volunteers and the sessions happen over electronic means like GMeet or Zoom or any similar online platform.

Use your creativity on how you make the platform engaging, safe and efficient (student to volunteer ratios).

Table of contents	**Page No**

1. Background

As per tradition, the past and today, the teaching methodologies and mapping of curriculums to classroom sessions form the science behind the timely supply of education. Today and since 2019, classroom environments have changed.

With the availability of virtual reality or online technology, we have Google classrooms, the G-Suite for Education helping deliver classroom sessions remotely.

We also have NPTEL that delivers technology enhanced learning to students in engineering, basic sciences and selected humanities and social sciences subjects.

NPETL: National Programme on Technology Enhanced Learning (NPTEL) was initiated by seven Indian Institutes of Technology

We do know about online web portals but this does not mean timely supply of formal syllabus/curriculum/course outcome based education via educational institutions or different categories of service providers, in times of natural or man-made adversity or for the need to manage dynamics in time availability.

The Ministry could also air cognitive educational content via T.V. and other educational or institutional frameworks. The multimedia content for this is emerging. With all this mind, what we think is missing is a CRM bridge for any dynamic teacher-student connect or adaptive classroom environment.

Today the need for Social Accountability is a solution based on standpoints. We need to translate this to processes for Sustainable resource management, Responsive Educational System Development & Capacity Management, where educational institutions, or the facilitating businesses and/or the educational system and its links are profiled to understand the FMCEA (Failure Mode Cause & Effects Analysis) that is critical to ensure student welfare.

Our CRM based Mentorship Platform addresses what is affecting the educational system today. It looks at social accountability to address the changing classroom environment and teacher-student connect. Our idea for mentorship innovatively helps both the poor, underprivileged and regular students.

2. Problem solving (background)

Our Gyan CRM bridge uses a Design and Deliver framework & platform that helps a mentor (or volunteer) commit a level of learning assistance to the mentee. Our bridge includes specific learning functions such as

1. Level of mentorship (or tutoring) that is Basic, Intermediate, Advanced

2. Domain of mentorship (or tutoring) that is School and Pre-university level based or Undergraduate level based…

3. Gyan components (Gyan CRM Products or Gyan CRM Services)

4. A K-Choreograph that includes a Mentor-Mentee cycle and mentorship platform specific cycles such as a Mentor-Resource Allocation cycle, a Mentor-Process Management cycle etc

5. A Platform specific Network such as the Internet/Mobile /new CRM specific connectivity services etc

6. A Scalability factor (such as long duration Mentor-Mentee cycles and additional Mentees cycles)

7. A Pairing/Un-pairing Factor that uses 2-way ANOVA to determine the pairing in the Mentor-Mentee cycle, the Choreograph-Fast Track cycle

Our Sense and Respond solution finding

Our in-time problem solving via the bridge identifies that, what is needed, is the use of practices like Customer Relationship Management

With CRM, the projected insight is that we will be soon able to

(i) Bridge Quality of Service gaps for learning assistance, outcome management and volunteered or prescriptive mentorship

(ii) Scale up for choreographed accountability ensuring we can develop platforms, products and services as part of a management framework that can be fast tracked or adaptive (like all the Classroom-Teacher-Student e-Connect solutions available today), but Is more conformant and sustainable for the educational system, which we state is evolving to keep up with remotely teaching students.

    This potential management framework will include a choreographed mentor-mentee connect that can be leveraged by any institution, business or faculty expecting to address the issues of less conformant classroom-teacher to student (end to end) facilitation or volunteered learning guidance.

3. What it does (Solution and Approach)

The GYANCRM-S and its tools implement a Mentorship platform Map based Platform-Mentor-Mentee Search, Recommend and Connect solution via

[a] Recommendation Engine/System solutions

[d] Search, Recommend and Connect solutions (with Classification or Supervised Learning) for Degrees of freedom, a Degree of Social Accountability and a Degree of Risk mitigation based on Post Pandemic and Environment factors

The Gyan CRM Map Component and its tools will help a mentee or student search for mentorship recommendations based on

1. Popularity of the Mentor
2. Content such as Domain, Area of mentorship, Subjects, Topics, Lessons, Course objective learning assistance
3. Classifications such as

a. Availability

b. Pre-requisite criteria

c. Topic modeling

d. Regular / Fast Track Schedules

e. Channels for online lessons

f. Case studies

g. Feedback (Positive, Negative, Comments (using the TFIDF practice)

h. Non-parametric criteria such as Nearest neighbor

4. Collaborative Filtering such as User-User collaborative filtering (that is mentee-mentee collaborative filtering) and Item-Item collaborative filtering (that is Subject-Subject, Topic-Topic, Lesson-Lesson)

5. Model categorization or Mentorship Platform categorization such as

a. Window functions (both Rolling and Expanding window functions)
b. Correlation (such as Environmental Factors Management, Risk Management, Time Interaction Performance (TIP) theory, Touch Point performance, Priority Area guidance, Schedules, Costs)
c. Time Series Forecasting based on the Pairing factor principle

4. Inference

We infer that a Gyan CRM Machine Learning Process is needed where it should involve the following steps

1. Define the mentorship platform problem, like the mainline classroom-teacher-student connect dynamics or the Gyandaan specific call for machine learning to help students enrol to learn and obtain mentorship and volunteers enrol themselves to help / mentor those students
2. Describe the problem based on Task, Experience and Performance
3. Ask / Assess the need for a solution based on the R2E CRM model and its different degrees of freedom, degree of social accountability and degree of CCMA / risk mitigation
4. Data collection using Enrolment/ Analytics / Surveys / Feedback
5. Prepare the data for machine learning via

a. Cleaning

b. Formatting

c. Sampling

d. Decomposition

e. Scaling

6. Select the algorithm based on

a. Classification, Regression, Clustering

b. Recommendation Systems.

7. Train the algorithm based on Pairing factors
8. Evaluate the performance based on Test data, use this to tune the pairing factors or parameters such as (1) Gyan Product / Service components, (2) Gyan CRM Choreographs (3) Levels of mentorship
9. Start using the model to plan, implement, manage and improve mentorship

   For the prototype/model possible, you need to refer to the Code excerpts we have included as a proof of concept

5. Methodology

In the solution,

1. The Mentorship Platform codifications in the repository are clustered using a combination of

(a) **Text-analytics** of "text fields" with select choreographed mentor-mentee connect descriptions,

(b) **"trainable qualified-choreographed connect-experiences",**

(c) "trainable qualified-choreographed connect-information" and

(d) **a categorization variable** that categorizes the nature of choreographed learning assistance, that is whether the recommendation is based on new Pairing factors, Data collection, Responsiveness**.**

2. The Text-analytics technique is based on **Word2Vector**

3. The clustering technique is based on **DBSCAN**

4. The **Cosine similarity algorithm** is used to classify choreographed learning assistance to fit within one of the buckets created (where this is based on text categorization)

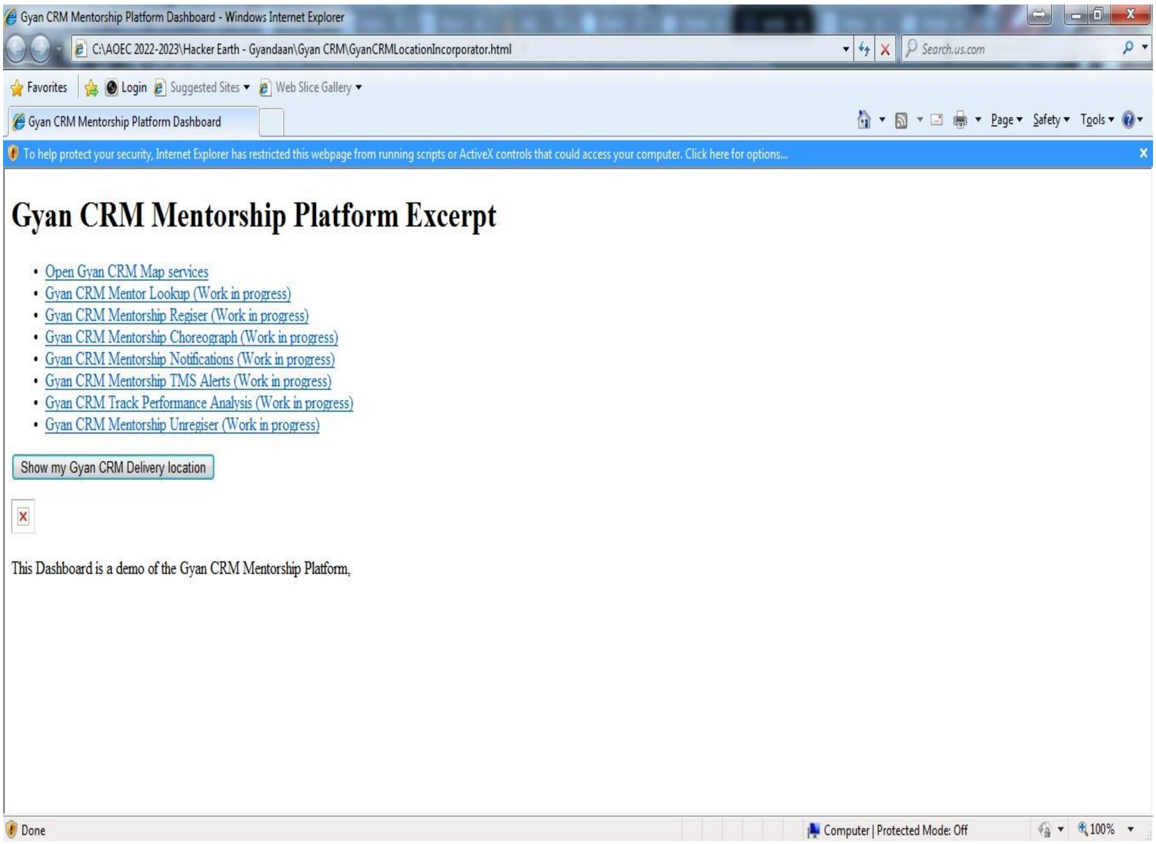
5. sklearn.neighbors , sklearn.linear_model, sklearn. model_selection and sklearn. metrics – to import Logistic Regression, train_test_split and accuracy_score for choreographed mentor-mentee connects or choreographed learning assistance

6. How we will build it

We at AOEC are developing the idea using the Python & Anaconda framework and different libraries for Recommendation systems, data analysis, array processing, Natural language processing, Text-analytics & clustering, visualizing of clusters, **choreographed learning assistance/mentor-mentee connect** description similarity

<u>Our solution's Basis of meeting the evolving dimension of Tutoring</u>

Our Gyan CRM Map will actively change to help mentors register themselves (when they are ready to involve themselves as Gyan CRM delivery channels) , where their contacts are thereon available to any student (or mentee) seeking educational assistance (or tutoring).



An enrolment form as part of the Registration could include data such as

1. Name

2. Type of enrolment: Tick as applicable

Student / Mentor / Service provider

3. Location

4. Address

5. Domain

6. Area of mentorship

7. Subjects

8. Topics

9. Lessons

10. Course objective learning assistance

<u>11. Timing: Tick as applicable</u>

a. Anytime/ During [From ____ To ____]

b. Anywhere / Neighboring location/ Suitable location

c. Anyhow mode/Classroom mode/Online mode/Combo

d. Learning assistance/mentor-mentee connect preferences: Microsoft Teams/ Google classroom/ Zoom/ Volunteered solutions for Virtual classrooms

e. Planned sessions / Request and respond sessions / Subscription based sessions

12. At individual level, Health / Ability parameterization: Tick as applicable

(1) **Physical ability** like Normal/Afflicted/Sick/Differently able

(2) **Mental ability** Normal/Afflicted/Sick/Differently able

(3) **Acclimatized ability** for this platform, like Knowledgeable/ Partially knowledgeable/Not acclimatized

(4) **Accountability to assist like** Experienced/have Intermediate level Experience/New/Cannot help currently)

(5) **Health parameterization like** (Physical help needed for mobility/Companionship needed/Handicapped/Use different wear-ons that accentuate behavioral or stress vulnerability (Artificial limbs or prosthetics; Aids for hearing or speaking; Pacemaker for the heart)

13. Agree to Participate in Lexio Ontology and the **data (key metrics and drivers): Yes/No/Not sure now**

For our promo, we recommend Supervised Learning for the Mentorship Platform experience to report data (key metrics and drivers) that can be converted into a data story via Lexio and a concept called Supervised Learning Ontology (URL: https://narrativescience.com/data-storytelling)

The implementation can be planned in the following manner:

1. Define a GYANCRM cloud data warehouse that is integrated to a GYANCRM Mentorship Platform

2. Let Lexio connect to the GYANCRM cloud data warehouse

3. Understand the Context by mapping to the Lexio Ontology and the **data (key metrics and drivers), thereon the Authoring engine determines what is to be written as Mentorship Platform Experience based on the related, adaptive or fixed questions for assistance, Lexio then runs analytics using Natural Language Processing & Generation to come up with a data story**

4. Lexio can then deliver briefs or what is to be read next insights

5. Lexio can empower action by recipients or subscribers to comment, further share within Mentorship Platform groups and request for notification

This can create a data driven culture for Mentorship Platform Experiences and at the next level for *visual, tactile, auditory experiences via Gyan CRM Products/Services/Allied innovation. This is done by empowering every stakeholder, responder, interested party to report data that can then be a data story that can be read and experienced*

The Supervised Learning Ontology could use

- Process-oriented factors
- Performance factors
- Gyan Product / Service components
- Gyan CRM Choreographs
- Levels of mentorship

The details of the libraries follow:

Specific libraries to load data, perform computation and display output are

(a) Pandas – Data acquisition library

(b) numpy – Array processing library

(c) nltk.data and nltk.corpus – Natural language processing library

(d) gensim and gensim.models – for text analytics and clustering, where the Word2Vector function is used

(e) gensim.models.keyedvectors – to import keyed vectors

(f) matplotlib – for visualizing clusters

(g) sklearn.cluster – to import DBSCAN for clustering

(h) sklearn.metrics.pairwise – to import cosine-similarity to find out sense and respond assistance description similarity

(i) keras.datasets – to import the CIFAR-10 dataset

(j) keras – to create a Convolutional Neural Network

(k) scipy.misc - to import image functions

(l) sklearn.neighbors – to import Nearest neighbors

(m) statsmodels.tsa.stattools – to import adfuller

(n) statsmodels.tsa.arima_model – to import ARIMA

(o) sklearn.linear_model – to import Logistic Regression

(p) sklearn. model_selection – to import train_test_split

(q) sklearn. metrics – to import accuracy_score

(r) imblearn. over_sampling – to import SMOTE

Work in progress

Code snippets in the basic proof of concept for a Mentorship Platform tool that clusters / trains learning assistance

(1) To import libraries and functions

(2) To load data

(3) For filtering of requests based on mentorship platform groups for "learning assistance categorization" (where there are multiple groups and a Gyan CRM Mentorship Platform (GYANCRM_MP) category, it is noted that the Gyan CRM Mentorship Platform category is a proof of concept that helps improved mentor-mentee connects, problem solving for classroom-teacher-student-connect (dynamics) and emergent solution finding for **Mentorship Platform Experiences for the crisis seen in educational platforms in the post pandemic or social distancing scenario.**

(4) Text analytics to create the training data for the machine learning algorithm

(5) Running of the clustering function

(6) Assigning of a new learning assistance request to a correct bucket based on the cosine-similarity function

A.1 To import libraries and functions

```
import os

import pandas as pd

import numpy as np

from numpy import array

from IPython.display import display

#For natural language processing ability

import nltk.data
```

```
from nltk.corpus import stopwords

#gensim libraries

import gensim

from gensim.models import word2vec

from gensim.models.keyedvectors import KeyedVectors


#to visualize the clusters

import matplotlib.pyplot as plt

import matplotlib.cm as cm


#for clustering

from sklearn.cluster import DBSCAN

import sklearn.metrics as metrics


#to compute service request description similarity

from sklearn.metrics.pairwise import cosine_similarity
```

A.2 To load data

The algorithm will be based on the source of data for the GYANCRM-TOOLS / GYANCRM-SUITE, where this can be a spreadsheet, a .CSV dump of learning assistance, or direct customized access to a GYANCRM-SUITE database.

The interest is to load the data into the program as an array of strings.

The structure of the .CSV data file for example is

SRA_number	**learning_assistance**	**GYANCRM_group**
SRA30000	*School level*	*MP_LEVEL*
SRA32000	*Pre-university level*	*MP_LEVEL_2*

SRA34000	*Under-graduate level*	*MP_LEVEL_3*
SRA36000	*Post-graduate level*	*MP_LEVEL_4*
SRA40000	Course level	*MP_LEVEL_5*
SRA41000	CRM Project level	*MP_LEVEL_6*
SRA42000	Core subjects	SUBJECT_LEVEL_1
SRA50000	Associated subjects	SUBJECT_LEVEL_2
SRA51000	Specialization subjects	SUBJECT_LEVEL_3
SRA52000	Elective subjects	SUBJECT_LEVEL_4
...	...	...

Code snippet

#data file that contains old learning assistance details

f = 'old_learning_assistance.xlsx'

data_1 = pd.read_excel (f, sheet_name='SRA', converters={'learning_assistance':str})

A.3 For filtering of requests based on groups for "learning assistance"

The algorithm will be based on identifying the information from the data set to make learning assistance categorization or clustering easy. For this example, we will use the GYANCRM_MP group or department to be the driver element for the clustering. The details are as follows

{

'*MP_LEVEL*',

'SUBJECT_LEVEL_1',

'SUBJECT_LEVEL_2',

'SUBJECT_LEVEL_3',

'SUBJECT_LEVEL_4'

}

Code snippet

assignment_group_subset = {

'*MP_LEVEL*',

'SUBJECT_LEVEL_1',

'SUBJECT_LEVEL_2',

'SUBJECT_LEVEL_3',

'SUBJECT_LEVEL_4'}

}

data_1 = data_1[data_1.assignment_group.isin(assignment_group_subset)]

A.4 Text analytics to create the training data for the machine learning algorithm

The algorithm

1. Create training data by averaging vectors for the words in the learning assistance (SRA)

2. Calculate the average feature vector for each element and return a 2D numpy array

3. This array is the training data for running cluster functions

Code snippet

```
#Load Google's pre-trained Word2Vec model known to contain 300 dimensioned vectors for
# 3 million words and phrases, this is still in a point of (work in progress) evaluation

model_google3M = gensim.models.KeyedVectors.load_word2vec_format('./GoogleNews-
vectors-negative300.bin', binary=True)

#Create training data by averaging vectors for words in the learning_assistance #column

def createFeatureVec(words, model, num_features):


  #convert Index2word list to a set for speedy execution

  index2word_set = set (model.wv.index2word)
```

```
    #loop over each word in the learning_assistance

    #if it is in the model's vocabulary, add its feature vector to the total

    for word in words:

        if word in index2word_set:

            nwords  = nwords + 1.

            featureVec = np.add ( feastureVec, model[word])


    #divide the result by the number of words to get the average

    featureVec = np.divide(featureVec, nwords)

    return featureVec


def getAvgFeatureVecs(vShortDescription_s, model, num_features):

    #for the given set of vShortDescription calculate the average feature vector for each list of
#words and return a 2D numpy array

    counter = 0

    #preallocate a 2D numpy array for speed in execution

    vShortDescriptionVecs = np.zeros((len(vShortDescription_s), num_features), dbtpe =
'float32')


    for vShortDescription in vShortDescription_s:

            vShortDescriptionVecs[int(counter)] = createFeatureVec(vShortDescription, model,
num_features)

            counter = counter + 1.


    return vShortDescriptionVecs

 clustering_vec = getAvgFeatureVecs(data_1['learning_assistance'], model_google3M, 300)
```

A.5 Running of the clustering function

The algorithm uses DBSCAN for clustering, which uses a high-density clustering approach. The positions of the vectors created in 12.4 are checked and high-density areas are taken as a new cluster, where low density areas separate clusters

Code snippets

```
#clustering using DBSCAN

db = DBSCAN(eps=0.3, min_samples = 10).fit(clustering_vec)

core_samples_mask = np.zeros_like(db.labels_, dtype=bool)

core_samples_mask[db.core_sample_indices_] = True

labels = db.labels_
```

Visualizing the Clustering output

```
#plot result

unique_labels = set (labels)

colors = [plt.cm.Spectral(each)

        for each in np.linpace(0,1,len(unique_labels))]

for k, col in zip (unique_labels, colors):

        if k == -1:

        #use black for noise aspect

                col = [0,0,0,1]

class_member_mask = (labels == k)

xy = clustering_vec[class_member_mask & core_samples_mask]

plt.plot(xy.iloc[:,0], xy.iloc[:,1], ‘o’,  markerfacecolor = tuple(col), markeredgecolor = ‘k’,
makersize = 14)

xy = clustering_vec[class_member_mask & core_samples_mask]

plt.plot(xy.iloc[:,0], xy.iloc[:,1], ‘o’,  markerfacecolor = tuple(col), markeredgecolor = ‘k’,
makersize = 1)

plt.title(‘GYANCRM-S Estimated number of clusters: %d’ %n_clusters_)
```

plt.show()

A.6 Assigning of a new learning assistance to a correct bucket based on the cosine-similarity function

<u>The Algorithm used</u>

1. Create the vector from the description text of the learning assistance (SRA) using the Word2Vec function

2. Calculate the similarity score for the vector using the cosine_similarity function

3. Find the cluster that the learning assistance is assigned to where this is done based on the maximum similarity score and averaged across all Sense and respond assistance in the cluster

4. If no matching cluster is found for a learning assistance vector then the learning assistance has no training detail in the repository and hence is unassigned for any clustering

<u>Code snippets</u>

<u>#Function assigns a new learning assistance to previously grouped learning assistance clusters</u>

```
def newSRAClusterer(newSRAText):

#vectorize SRAText

newSRAVector = getAvgFeatureVecs (newSRAText, model_google3M, 300)

#Build the data frame with learning assistance meta data and similarity scores

data_8 = pd.concat ([pd.DataFrame(labels), data_6_1[['SRA_number'learning_assistance',

'MP_LEVEL']]], axis=1)

data_8.rename (columns = {0:'Cluster'},inplace = True)

data_8['similarityScore'] = cosine_similarity(data_6_1.iloc[:,26:326], newSRAVector)

# Find the cluster that the learning assistance is assigned to where this is done

# based on the maximum similarity score and averaged across all learning

# assistance in the cluster

similarityScoreMean = data_8.groupby ('Cluster')['similarityScore'].mean().max()

newSRACluster = data_8.groupby ('Cluster')['similarityScore'].mean().idxmax()
```

if similarityScoreMean >= 0.7

#this threshold needs to be tuned to ensure noise element is not incorrectly assigned a clustered bucket

print (‘The learning assistance is assigned to the cluster’, newSRACluster)

print (‘The learning assistance similarity to the assigned cluster:’, round(similarityScoreMean,2))

else:

print (‘This learning assistance is unlike any detail in the training repository and is not assigned to any cluster’)

return similarityScoreMean, newSRACluster

Work in progress

Code snippets in the basic proof of concept for a GYANCRM tool that uses Classification based Recommendation / Learning for the Pairing Factors control needed for learning assistance

(1) To import libraries and functions

(2) Read **Pairing Factors Control Level needed for mentorship or learning assistance**

(3) Show all the columns

(4) Check the head of the dataframe

(5) Use relevant attributes to build the classification model

(6) Convert values to numeric if needed

(7) Check the shape of the data

(8) Extract features from the dataset

(9) Extract class

(10) Split the data in training and test set

(11) Create object of logistic regression

(12) Use SMOTE algorithm for over sampling

(13) Fit the model

(14) Predict using the logistic regression model

(15) Calculate accuracy for equity level

Work in progress

(1) To import libraries and functions

import pandas

import sklearn

from sklearn.linear_model import Logistic Regression

from sklearn. model_selection import train_test_split

from sklearn. metrics import accuracy_score

from imblearn. over_sampling import SMOTE

(2) Read **Pairing Factors Control Level needed for mentorship or learning assistance**

pf_level_needed_data = pandas.read_csv (‘./data/pf_level_needed_data.csv’, sep=‘;’)

(3) Show all the columns

Pandas.set_option (‘display.max_columns’, None)

(4) Check the head of the dataframe

pf_level_needed_data.head()

(5) Use relevant attributes to build the classification model

pf_level_needed_data = pf_level_needed_data[[‘Degrees of freedom’, ‘Degree of social accountability’, ’Degree of risk mitigation’, ‘Scalability factor’, ‘Gyan CRM Store’, ‘Gyan CRM Mentorship Platform’’]]

(6) Convert values to numeric if needed

(7) Check the shape of the data

pf_level_needed_data.shape()

(8) Extract features from the dataset

X = pf_level_needed_data.iLoc[:, :6]

X = pandas.get_dummies (pf_level_needed_data.iLoc[:, :6]).values

(9) Extract class

Y = pf_level_needed_data.iLoc[:, :7].values

(10) Split the data in training and test set

X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.10)

(11) Create object of logistic regression

logistic_regression = LogisticRegression()

(12) Use SMOTE algorithm for over sampling

smote_object = SMOTE (random_state=4)

X_res, y_res = smote_object.fit_sample (X_train, Y_train)

(13) Fit the model

logistic_regression.fit (X_res, y_res)

(14) Predict using the logistic regression model

Y_pred = logistic_regression.predict (X_test)

(15) Calculate accuracy for pairing factor control level

accuracy_score (Y_test, Y_pred)

Code snippets in the basic proof of concept for a GYANCRM tool that uses Immersive & Perceptive Time Series Forecasting for the Real Time Score, Interactive factors (step wise)

(*) To do the analysis first plot the Real time score data, Interactive factors data for the “Domain, Area of mentorship, Level of mentorship’ Choreograph

(1) To import libraries and functions

(2) To get and print test result

(3) To apply test on Real time score data, Interactive factors data for the “Domain, Area of mentorship, Level of mentorship’ Choreograph

(4) If the Test Statistics is higher than critical value meaning raw time series is not stationary, so apply transformations and again check results

(6) Regenerate **plot of the transformed** Real time score data, Interactive factors data for the “Domain, Area of mentorship, Level of mentorship’ Choreograph

(7) Get a new time series with a difference of consecutive values

(8) Plot the new data

(9) If any record has a NaN remove that value and apply Dickey-Fuller test, check Test Statistics to see how close is it to the critical value to report confidence about the time series being stationary

(10) Apply aggregation, smoothing and regression-fitting to make Real time score data, Interactive factors data more stationary

Apply moving average technique of drills or evacuations or assistance for the past 12 values (meaning 12 months or 1 year) representing the yearly average at that point. If we feel that new values are more important than the old ones, we can use the weighted moving average technique.

(11) Generate a difference series

(12) Check the result of the test

Check Test Statistics to see how close is it to the critical value to report confidence about the time series being stationary

(13) Apply ARIMA model for time series forecasting

Note on the ARIMA model

AR (Number of Auto regressive terms): This represents the lag of the dependent variable

I (Number of differences): Number of non-seasonal differences

MA (Moving average): Lagged errors

(13.1) Import the library needed for the ARIMA model

(13.2) Make the model using AR only at first

(13.3) Calculate RSS for the model

(13.4) Make the model using MA

(13.5) Check the error

(13.6) Check the RSS value to identify the difference, if needed consider both AR and MA

(13.7) Check the error

(13.8) Check the RSS value of the above three models

(13.9) Check the best model, take it to the original form to get the current output predictions

(13.9.1)To do this create a new time series of values generated from the model

(13.9.2) Convert differencing to the log scale

(13.9.3) Add differences to the base numbers by using the cumsum() function

(13.9.4) Add values to the first base number

(13.9.5) As we are working on log, take the exponent value

(13.9.6) Plot the result

(13.9.7) Check the overall error

Thereon create models with different settings and compare RMSE

C Code snippets in the basic proof of concept for a GYANCRM tool that uses a recommendation engine to recommend a Domain, Area of mentorship, Level of mentoring Choreograph

C.1 To import libraries and functions

```
import pandas

import numpy

import sklearn

from sklearn.neighbors import NearestNeighbors
```

C.2 To read data for the choreograph

```
choreograph_data = pandas.read_csv ('./data/choreograph_data.csv')
```

C.3 To check initial data

```
choreograph_data.head()
```

C.4 To apply any algorithm to convert categorical variable to numeric values

Convert prerequisites

choreograph_data.code[choreograph_data ['prerequisites'] == 'School level', 'prerequisites'] = 0

choreograph_data.code[choreograph_data ['prerequisites'] == 'Pre-university level', 'prerequisites'] = 1

choreograph_data.code[choreograph_data ['prerequisites'] == 'Under graduate level, 'prerequisites'] = 2

choreograph_data.code choreograph_data ['prerequisites'] == 'Post-graduate level'', 'prerequisites'] = 4

C.5 To do the same for all other attributs

Similarly we need to convert **Core Subject level**

Similarly we need to convert **Associated Subject level**

Similarly we need to convert **Specialization Subject level**

Similarly we need to convert **Elective Subject level**

C.6 To create nearest neighbour object

nn1 = NearestNeighbors (n_neighbors=1)

C.7 To fit model

nn1.fit (choreograph_data.code[:, 'prerequisites':' Application'])

C.8 To define learning assistance requirement in terms of Choreograph cycles.

requirement = [4,1,16,1]

C.9 To check the most suitable additions to learning assistance, like **Choreograph cycles**

print (nn1.kneighbors (requirement))

##.D Code snippets in the basic proof of concept for a GYANCRM tool that uses Immersive & Perceptive Time Series Forecasting for the Real Time Score, Interactive factors (step wise)

(*) To do the analysis first plot the Real time score data, Interactive factors data for the Domain, Area of mentorship, Level of mentorship Choreograph

For our promo

We will use the **Choreopraph Variations reduction** column of the Real time score for Visual CERC art/art form/art work/allied innovation

choreograph_realtimescore = pandas.read_csv ('./data/ choreograph_ realtimescore.csv', parse_dates=['Month'], index_col=1)

choreograph_realtimescore.head()

choreograph_realtimescore.plot()

(1) To import libraries and functions

from statsmodels.tsa.stattools import adfuller

(2) To get and print test result

def test_timeseries_stationary (ts) :

ts = ts ('Real time score')

print ('Dickey-Fuller Test:')

dickey_fuller_test = adfuller (ts, autolag = 'AIC')

dickey_fuller_output = pandas.Series (dickey_fuller_test[0:4], index = ['Test Statistic', 'p-value', 'Number of Lags Used', 'Observations Used'])

for key,value in dickey_fuller_test[4].items():dickey_fuller_output['Critical Value (%s)'%key] = value

print (dickey_fuller_output)

(3) To apply test on Real time score data, Interactive factors data for the choreograph

test_timeseries_stationary (choreograph_realtimescore)

(4) If the Test Statistics is higher than critical value meaning raw time series is not stationary, so apply transformations and again check results. Take the log of the existing data

For this promo, the column we are detailing code for is **"**Choreograph variations **reduction for learning assistance" where the evaluation could result in any of the following values**

[1] Excellent experience (0001) (numeric value = 1)

[2] Good experience (0010) (numeric value = 2)

[3] Poor experience (0100) (numeric value = 4)

[4] Not applicable **(1000)** (numeric value = 8)

choreograph_realtimescore_log_value = numpy.log (choreograph_realtimescore)

(6) Regenerate **plot of the transformed** Real time score data, Interactive factors data for the choreograph

choreograph_realtimescore_log_value.plot()

(7) Get a new time series with a difference of consecutive values

choreograph_realtimescore_log_value_diff = choreograph _realtimescore_log_value - choreograph_realtimescore_log_value.shift()

(8) Plot the new data

choreograph_realtimescore_log_value_diff.plot()

(9) If any record has a NaN remove that value and apply Dickey-Fuller test, check Test Statistics to see how close is it to the critical value to report confidence about the time series being stationary

choreograph_realtimescore_log_value_diff.dropna (inplace=True)

test_timeseries_stationary (choreograph_realtimescore_log_value_diff)

(10) Apply aggregation, smoothing and regression-fitting to make Real time score data, Interactive factors data more stationary

Apply moving average technique of learning assistance for the past 12 values (meaning 12 months or 1 year) representing the yearly average at that point. If we feel that new values are more important than the old ones, we can use the weighted moving average technique.

exponential_weighted_average = pandas.ewma (choreograph_realtimescore_log_value, halflife=12)

plt.plot (choreograph_realtimescore_log_value)

plt.plot (exponential_weighted_average, color ='Green')

(11) Generate a difference series

exponential_weighted_average_diff = choreograph_realtimescore_log_value - exponential_weighted_average

(12) Check the result of the test

test_timeseries_stationary (exponential_weighted_average_diff)

Check Test Statistics to see how close is it to the critical value to report confidence about the time series being stationary

(13) Apply ARIMA model for time series forecasting

Note on the ARIMA model

AR (Number of Auto regressive terms): This represents the lag of the dependent variable

I (Number of differences): Number of non-seasonal differences

MA (Moving average): Lagged errors

(13.1) Import the library needed for the ARIMA model

from statsmodels.tsa.arima_model import ARIMA

(13.2) Make the model using AR only at first

Auto Regression Model, MA = 0 in this promo

ARIMA_model_1 = ARIMA (choreograph_realtimescore_log_value, order (2,1,0))

results_AR_1 = ARIMA_model_1.fit (disp=-1)

plt.plot (choreograph_realtimescore_log_value_diff)

plt.plot (results_AR_1.fittedvalues, color =’Green’)

plt.title (‘AR Model’)

(13.3) Calculate RSS for the model

print (‘Residual Sum of the Square: %.6f’ %sum((results_AR_1.fittedvalues-choreograph_realtimescore_log_value_diff[‘Real time score’])**2))

(13.4) Make the model using MA

Moving Average Model, MA = 1 in this promo

ARIMA_model_2 = ARIMA (choreograph_realtimescore_log_value, order (0,1,1))

results_MA_2 = ARIMA_model_2.fit (disp=-1)

plt.plot (choreograph_realtimescore_log_value_diff)

plt.plot (results_MA_2.fittedvalues, color =’Green’)

plt.title (‘MA Model’)

(13.5) Check the error

print (‘Residual Sum of the Square: %.6f’ %sum((results_MA_2.fittedvalues-choreograph_realtimescore_log_value_diff[‘Real time score’])**2))

(13.6) Check the RSS value to identify the difference, if needed consider both AR and MA

ARIMA Model, AR+MA in this promo

ARIMA_model_3 = ARIMA (choreograph_realtimescore_log_value, order (2,1,1))

results_ARIMA_3 = ARIMA_model_3.fit (disp=-1)

plt.plot (choreograph_realtimescore_log_value_diff)

plt.plot (results_ARIMA_3.fittedvalues, color =’Green’)

plt.title (‘ARIMA Model (AR+MA)’)

(13.7) Check the error

print ('Residual Sum of the Square: %.6f' %sum((results_ARIMA_3.fittedvalues-choreograph_realtimescore_log_value_diff['Real time score'])**2))

(13.8) Check the RSS value of the above three models

(13.9) Check the best model, take it to the original form to get the current output predictions

(13.9.1)To do this create a new time series of values generated from the model

ARIMA_diff_values = pandas.Series (results_ARIMA_3.fittedvalues, copy=True)

(13.9.2) To convert differencing to the log scale, first (13.9.3)

(13.9.3) Add differences to the base numbers by using the cumsum() function

ARIMA_diff_values_cumsum = ARIMA_diff_values.cumsum()

(13.9.4) Add values to the first base number

ARIMA_log = pandas.Series (float choreograph_realtimescore_log_value.ix[0]), index = choreograph_realtimescore_log_value.index)

ARIMA_log = ARIMA_log.add (ARIMA_diff_values_cumsum, fill_value=0)

(13.9.5) As we are working on log, take the exponent value

ARIMA_Result = numpy.exp (ARIMA_log)

(13.9.6) Plot the result

plt.plot (choreograph_realtimescore_log_value_diff)

plt.plot (ARIMA_Result)

plt.title ('ARIMA Model')

(13.9.7) Check the overall error

print ('Root Mean Squared Error: %.6f' %numpy.sqrt(sum((ARIMA_Result - choreograph_realtimescore ['Real time score'])**2)/len(choreograph_realtimescore ['Real time score'])))

Thereon create models with different settings and compare RMSE

7. What we learned (Conclusion)

Machine Learning Algorithms help us use past understanding or today's details to ideate and enable solutions for corresponding or standardized resolution, where machine learning can quicken problem solving and solution finding. We have attempted to apply this concept for improving the need for tutoring / learning assistance by poor, underprivileged and regular students.