

Learning Analytics for MOOC

Data Understanding, Quality and Preparation Document

Group No. : DS707-2017-10
ChellaPriyadharshini M (**MT2016041**)
Daminee Sao (**MT2016045**)
Jyotsana (**MT2016068**)
Kanika Narang (**MT2016069**)
Tehreem Ansari (**MT2016145**)

Initial Data Collection Report

Data Source:

1. Online Data from edX:- This open source data is taken from Kaggle. It provides data on 290 Harvard and MIT online courses, 250 thousand certifications, 4.5 million participants, and 28 million participant hours on the edX platform since 2012.

Dataset contains 23 columns, the important ones are:

Column Name	Description	Data Type
Institution	online course holders	String
Course Number	the unique id of each course	String
Launch Date	the launch date of each course	DateTime
Course Title	the title of each course	String
Instructors	the instructors of each course	String
Course Subject	the subject of each course	String
Year	the last time of each course	Numeric
Participants	the number of participants who have accessed the course	Numeric
Audited	the number of participants who have audited	Numeric

	more than 50% of the course	
Certified	the number of participants who have been certified	Numeric
Median Age	median age of the participants	Numeric
% Male	percentage of male students	Numeric
% Female	percentage of female students	Numeric
%Bachelor's Degree or higher	the percent of bachelor's degree of higher	Numeric

2. De-identified student-level data of HarvardX and MITx courses:- This release is comprised of de-identified data from the first year (Academic Year 2013: Fall 2012, Spring 2013, and Summer 2013) of MITx and HarvardX courses on the edX platform along with related documentation. These data are aggregate records, and each record represents one individual's activity in one edX course.

Column Name	Description	Data Type
course_id	Identifies institution, course name and semester ("HarvardX/CB22x/2013_Spring")	String
userid_DI	First portion identifies dataset, second portion is a random ID ("MHxPC130442623")	String
registered	Registered for course	0/1
viewed	Anyone who accessed course materials from the "Courseware" tab	0/1
explored	Anyone who accessed at least half of the chapters in the courseware	0/1
certified	Anyone who earned a certificate	0/1
final_cc_cname_DI	Country name	String

LoE	Highest level of education completed	String
YoB	Year of Birth	Numeric
gender	Gender	m/f/o
grade	Final grade in the course	0 to 1
start_time_DI	Date of course registration	Date
last_event_DI	Date of last interaction with course	Date
nevents	Number of interactions with the course (from tracking logs)	Numeric
ndays_act	Number of unique days students interacted with the course	Numeric
nplay_video	Number of play video events within the course	Numeric
nchapters	Number of chapters (within the courseware) with which the student interacted	Numeric
nforum_posts	Number of posts to the Discussion Forum	Numeric
roles	Identifies staff and instructors (but blank as staff and instructors were removed from this release)	String
inconsistent_flag	Identifies records that are internally inconsistent*	0/1

*Due to a variety of data issues, including missing tracking logs, a portion of the records have NULL values for *nevents*, but not-null values for *ndays_act*, *nforum_posts* or *nchapters*. The source for the *nevents* and *last_event_DI* is the tracking logs whereas the others come from a data source known as “Courseware Student Module”. Due to the 2 different sources if something is wrong with the tracking logs for a class or a student then the records can be internally inconsistent and have a value ‘1’ in this column.

Data Description Report

Quantity Of Data: Both the data files are in .csv format. Online data from edx contains 290 rows and 23 attributes while de-identified student level data contains 64,1138 rows and 20 attributes field. Most of the columns are in numeric form. They also contain Date type of attributes.

Merging Dataset: Initially, we thought of merging them together. We found that joining keys Course_id [from De-identified dataset] and Course_number, Year [from online data from edx] were represented differently in both of the dataset. We applied basic preprocessing on these attribute to make them same and then performed join operation on them. The resultant data set had huge amount of NULL values and we found it unuseful. So, we decided on not merging the dataset and do analysis on both separately.

Data Exploration Report

Name: Chellapriyadharshini M

Roll No: MT2016041

The dataset had to be visualized for better understanding and to isolate data quality issues that are not apparent. We used Tableau for creating the reports.

Initial Understanding:

- The data was obtained from two different sources: Harvard Dataverse and Kaggle. Hereafter we'll refer to these as: Dataverse dataset and Kaggle dataset.
- On visual inspection, it was found that the two datasets were at different granularity: the Dataverse dataset was at user level, giving one individual user's activity per edX course, while the Kaggle dataset was at the course level aggregating all users' activity per course.
- Also, Dataverse dataset comprises de-identified data for AY2013 (Fall 2012, Spring 2013 and Summer 2013) whereas Kaggle dataset contains aggregated data for the years 2012-2015.
- And, the Kaggle dataset has summary attributes like %Certified, %Audited, %Male/Female, Median Age, etc.
- Other notable and important attributes that are found in the Kaggle dataset but missing from the Dataverse dataset are:
 - Topics names like: STEM, Government-Health-Social Sciences, Humanities-History-Design-Religion-Education and Computer Science.
 - Instructor names for each course.

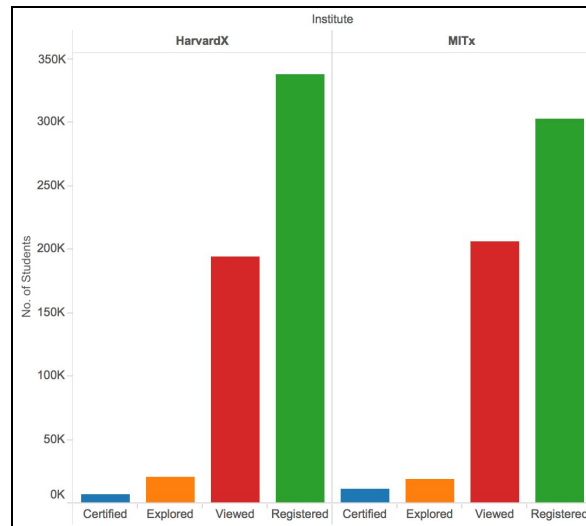
Reports and Issue Identification:

Dataverse Dataset:

- You can see the entire set of reports created [here](#), of which, we have listed below the most important ones. In all these reports, we show the measures for HarvardX and MITx separately.
- The `course_id` attribute is a combination of institute, course code, year and semester. In order to do the initial reports, this merged attribute was split into individual attributes.
- For brevity, the **course-description** is not included in the dataset but provided as a separate table in one of the documents that come along with the data. This gives useful information for interpreting the courses. The table is as given below:

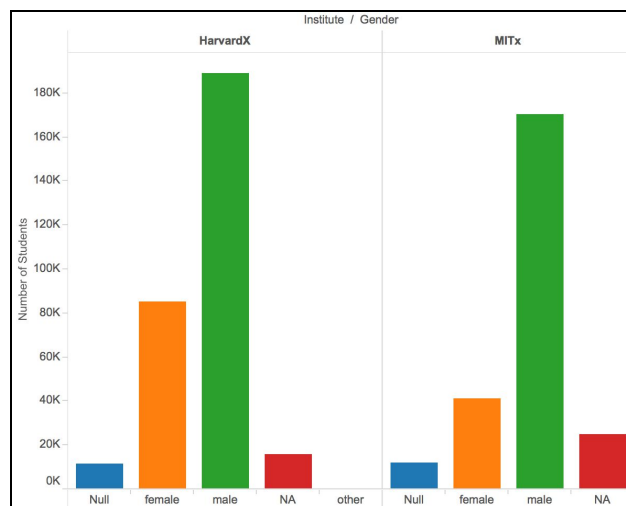
Institution	Course Code	Short Title	Full Title	Semester
HarvardX	CB22x	HeroesX	The Ancient Greek Hero	Spring-Summer 2013
HarvardX	CS50x	-	Introduction to Computer Science I	Fall 2012 – Spring 2013
HarvardX	ER22x	JusticeX	Justice	Spring-Summer 2013
HarvardX	PH207x	HealthStat	Health in Numbers: Quantitative Methods in Clinical & Public Health Research	Fall 2012
HarvardX	PH278x	HealthEnv	Human Health and Global Environmental Change	Summer 2013
MITx	14.73x	Poverty	The Challenges of Global Poverty	Spring 2013
MITx	2.01x	Structures	Elements of Structures	Spring-Summer 2013
MITx	3.091x	SSChem	Introduction to Solid State Chemistry	Offered twice: Fall 2012 and Spring 2013
MITx	6.002x	Circuits	Circuits and Electronics	Offered twice: Fall 2012 and Spring 2013
MITx	6.00x	CS	Introduction to Computer Science and Programming	Offered twice: Fall 2012 and Spring 2013
MITx	7.00x	Biology	Introduction to Biology – The Secret of Life	Spring 2013
MITx	8.02x	E&M	Electricity and Magnetism	Spring 2013
MITx	8.MReV	MechRev	Mechanics Review	Summer 2013

- **Student Categories:**
 - This report categorizes the students based on their interaction with the course as: Registered, Viewed (videos), Explored (course contents) and Certified (completed) and shows the number of students in each category, institute-wise.



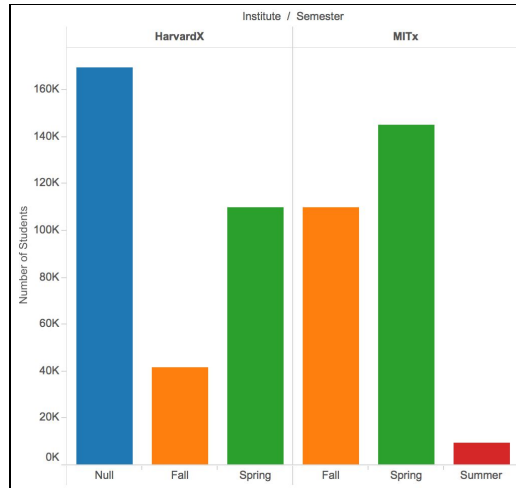
- **Gender Distribution:**

- This report shows the gender distribution in the courses offered by Harvard and MIT.
- We can see that the gender attribute has **null** and **NA** values apart from the allowed values of male, female and other. These null and NA values need to be handled.



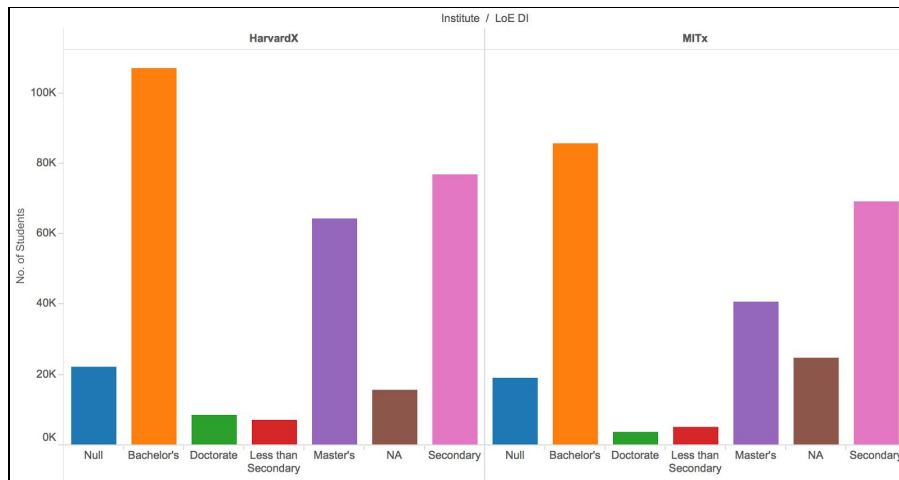
- **Semester-wise Distribution:**

- This report shows the number of enrolled students per semester.
- We see that while there are courses from HarvardX that do not have the semester attribute listed and hence we see null values. These should be handled by matching them with the **course-description table** in the documentation provided along with the dataset using the **Semester** column.



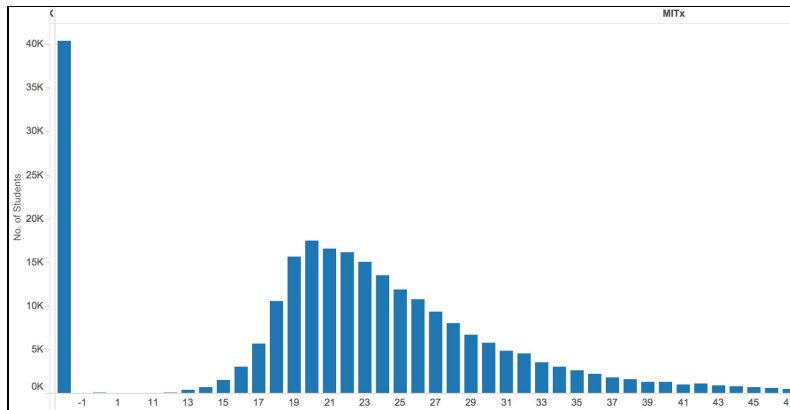
- **Level of Education:**

- This report shows the highest level of education of the students registered for the online courses.
- This attribute has **null** and **NA** values which needs to be treated.



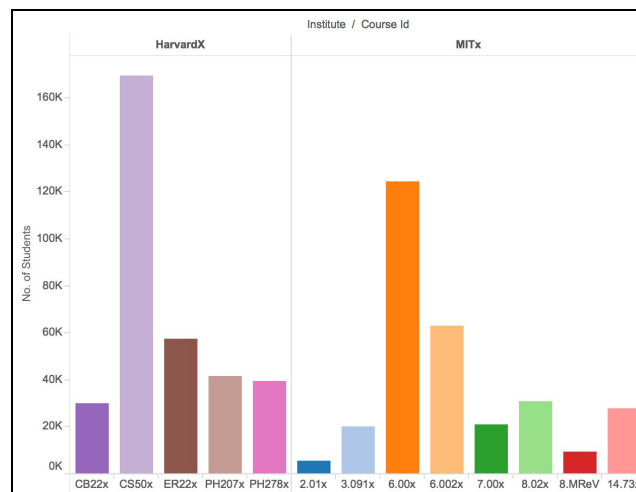
- **Age Distribution:**

- This report shows the age distribution across the courses of the two institutes.
- The dataset has **YoB** attribute which gives the user's Year of Birth. Age was derived from this attribute.
- We can see we have values like null, -1, 0 for Age because of the discrepancies in the underlying YoB attribute.



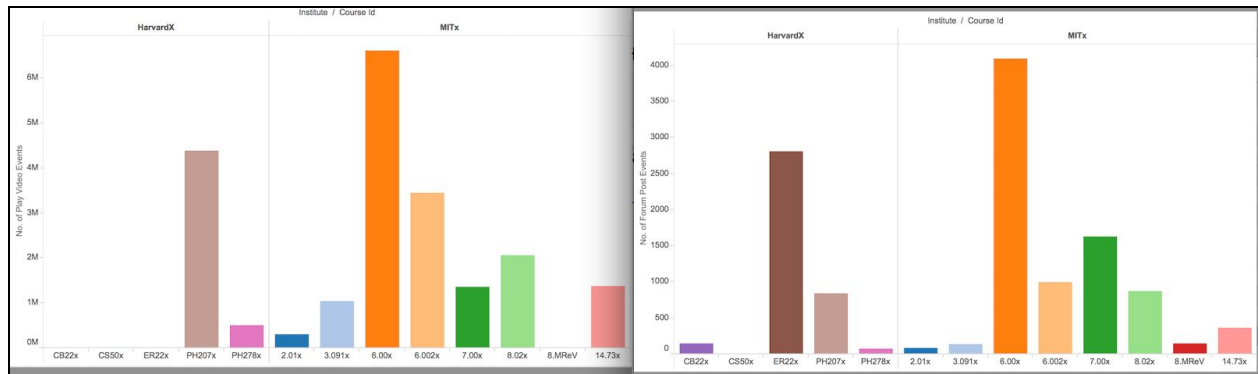
- **Course Popularity:**

- This report shows the number of registered students for each course during AY2013. We have the course codes and not the actual name of the course. This can be looked up from the **course-description table** provided in the documentation. If necessary the course name can be included as a separate attribute in the original dataset.



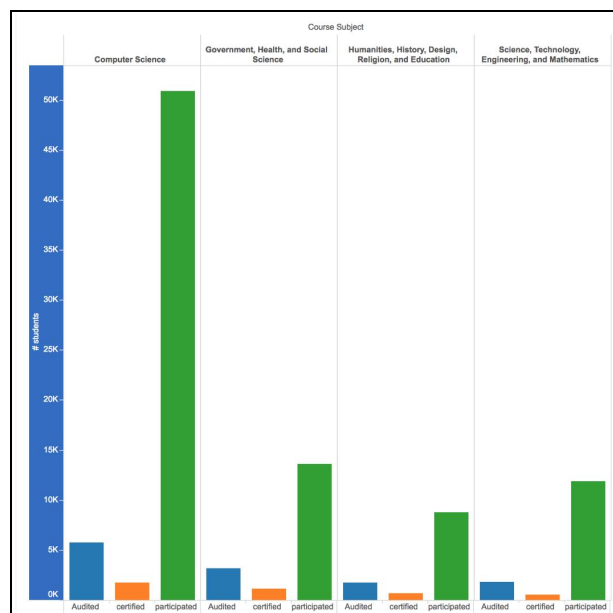
- **Play-Video Events and Forum Post Events:**

- The two reports shows the number of play video events and number of forum post events for the courses.
- We can see that courses like CS50x, CB22x, etc. have zero activity with respect to play video and forum posts. From the earlier report we saw that these are popular courses.
- The reason being the events are recorded from 2 different tracking logs, which if internally inconsistent gives rise to such issues. The inconsistency in event tracking is indicated by a flag attribute *incomplete_flag*, which we should use for handling this.



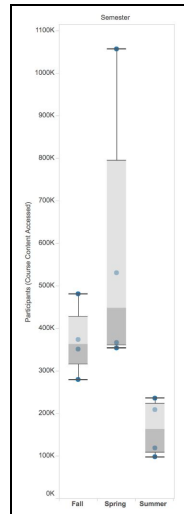
Kaggle Dataset:

- You can see the entire set of reports created [here](#), of which, we have listed below the most important ones.
- In all these reports, we show the measures segregated by the Topics: Science-Technology-Engineering-Mathematics, Government-Health-Social Sciences, Humanities-History-Design-Religion-Education and Computer Science.
- Another observation is that: since this dataset is already at a summarized granularity, we did not find any issues of NULL or missing values.
- Student Distribution:**
 - This report shows the number of students registered, certified and audited the courses under the mentioned topics.



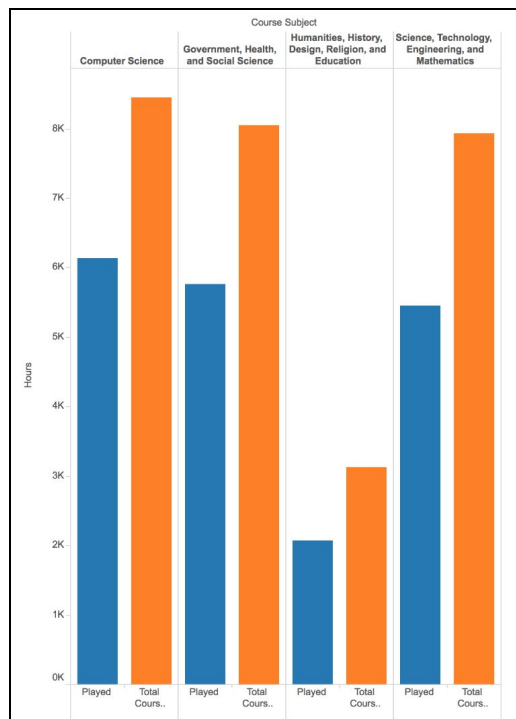
- Semester-wise Distribution:**
 - This report shows the semester-wise breakup of the students for the courses under the topics mentioned above.

- The original dataset did not have a Semester attribute. This was derived from the *Launch Date* attribute. If the month of the Launch Date is between 1 and 4 - Spring; if it is between 5 and 7 - Summer and if it is between 8 and 12 - Fall.
- From the report we can see that the median participation is higher for Spring courses compared to courses taught in Fall and Summer.



● Hours Video Played:

- This report shows the Total Video Hours and Played Video Hours for courses that fall under the respective Topics. We can see that the Computer Science courses top the chart closely followed by Government-Health-Social Sciences and STEM courses, while Humanities courses are the last.



Data Quality Report

Data Quality Issues

Points 1-2 and attribute wise analyses of data by Jyotsana, MT2016068

Points 3-6 done by Tehreem Ansari, MT2016145

1. Dummy Value

This de-identified data is taken from edx platform. Since edx offers platform for courses Harvard and MIT we can safely assume this data does not contain any dummy values and are from valid sources.

2. Absence of data

We wrote a R script to find number of missing values in each column of the dataset. The R script we wrote was:-

```
for (i in 1:20)
{
  print(paste(i,"th column",colnames( data )[i] ," has",sum(is.na(data[i])), " missing values"))
}
```

The output is:-

```
[1] "1 th column course_id has 0 missing values"
[1] "2 th column userid_DI has 0 missing values"
[1] "3 th column registered has 0 missing values"
[1] "4 th column viewed has 0 missing values"
[1] "5 th column explored has 0 missing values"
[1] "6 th column certified has 0 missing values"
[1] "7 th column final_cc_cname_DI has 0 missing values"
[1] "8 th column LoE_DI has 58132 missing values"
[1] "9 th column YoB has 96605 missing values"
[1] "10 th column gender has 58132 missing values"
[1] "11 th column grade has 57400 missing values"
[1] "12 th column start_time_DI has 0 missing values"
[1] "13 th column last_event_DI has 0 missing values"
[1] "14 th column nevents has 199151 missing values"
[1] "15 th column ndays_act has 162743 missing values"
[1] "16 th column nplay_video has 457530 missing values"
[1] "17 th column nchapters has 258753 missing values"
[1] "18 th column nforum_posts has 0 missing values"
[1] "19 th column roles has 641138 missing values"
[1] "20 th column incomplete_flag has 540977 missing values"
```

Similarly for Percentage:-

R Script:-

```
for (i in 1:20)
{
  print(paste(i,"th column",colnames( data )[i] ," has",(sum(is.na(data[i]))/NROW(data))*100,"% missing values"))
}
```

Output:-

```

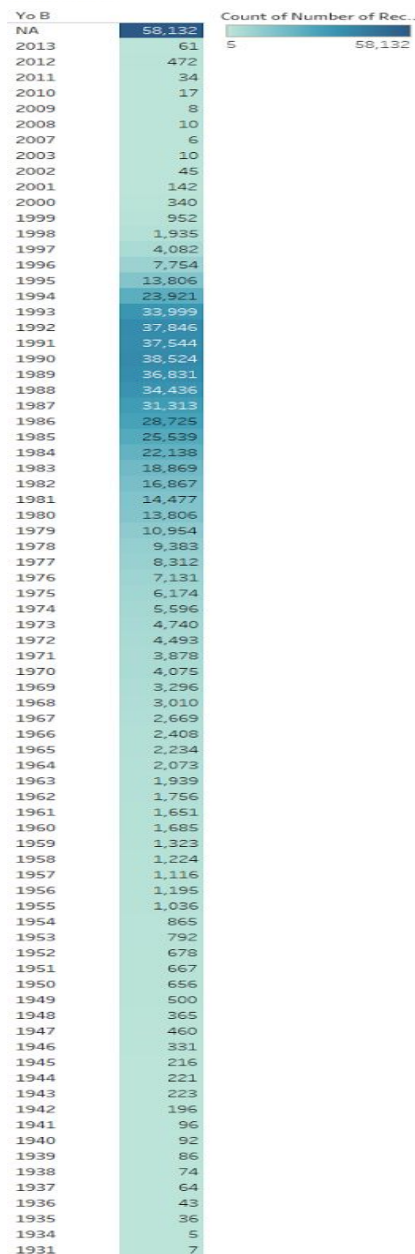
[1] "1 th column course_id has 0 % missing values"
[1] "2 th column userid_DI has 0 % missing values"
[1] "3 th column registered has 0 % missing values"
[1] "4 th column viewed has 0 % missing values"
[1] "5 th column explored has 0 % missing values"
[1] "6 th column certified has 0 % missing values"
[1] "7 th column final_cc_cname_DI has 0 % missing values"
[1] "8 th column LoE_DI has 16.5343498591567 % missing values"
[1] "9 th column YoB has 15.0677389267209 % missing values"
[1] "10 th column gender has 13.5393628204848 % missing values"
[1] "11 th column grade has 8.95283074782652 % missing values"
[1] "12 th column start_time_DI has 0 % missing values"
[1] "13 th column last_event_DI has 27.9119315966297 % missing values"
[1] "14 th column nevents has 31.062111433108 % missing values"
[1] "15 th column ndays_act has 25.3834587873438 % missing values"
[1] "16 th column nplay_video has 71.3621716385552 % missing values"
[1] "17 th column nchapters has 40.3583939807031 % missing values"
[1] "18 th column nforum_posts has 0 % missing values"
[1] "19 th column roles has 100 % missing values"
[1] "20 th column incomplete_flag has 84.3776222903649 % missing values"

```

The attributes having no missing values do not need any treatment. But, other 10 attributes needed some kind of missing value treatment. Further analysis led to:-

Attribute Name	Analysis	Missing Value Treatment														
LoE_DI	<ul style="list-style-type: none">This is an important attribute for overall analysis which give highest level of education completed by an individual. So, this cannot be simply ignored.Upon further inspection we found out that 16.53% of total rows have NA for LoE_DI which amounts to 58132 rows.From the below frequency distribution we can see that 'Bachelor's' has maximum entries. <p>LoE_DI</p> <table><thead><tr><th>LoE DI</th><th></th></tr></thead><tbody><tr><td>Bachelor's</td><td>2,19,768</td></tr><tr><td>Doctorate</td><td>13,387</td></tr><tr><td>Less than Secondary</td><td>14,092</td></tr><tr><td>Master's</td><td>1,18,189</td></tr><tr><td>NA</td><td>58,132</td></tr><tr><td>Secondary</td><td>1,69,694</td></tr></tbody></table>	LoE DI		Bachelor's	2,19,768	Doctorate	13,387	Less than Secondary	14,092	Master's	1,18,189	NA	58,132	Secondary	1,69,694	<p>Approach 1</p> <p>We will fill missing values by putting mode grouped by age group of Students. As people belonging to similar age group will have similar highest level of education qualification. (Atleast for early age groups this should be true).</p> <p>Approach 2</p> <p>Use final_cc_cname_DI field which gives us the country a person belongs to. Underlying assumption is that each country have</p>
LoE DI																
Bachelor's	2,19,768															
Doctorate	13,387															
Less than Secondary	14,092															
Master's	1,18,189															
NA	58,132															
Secondary	1,69,694															

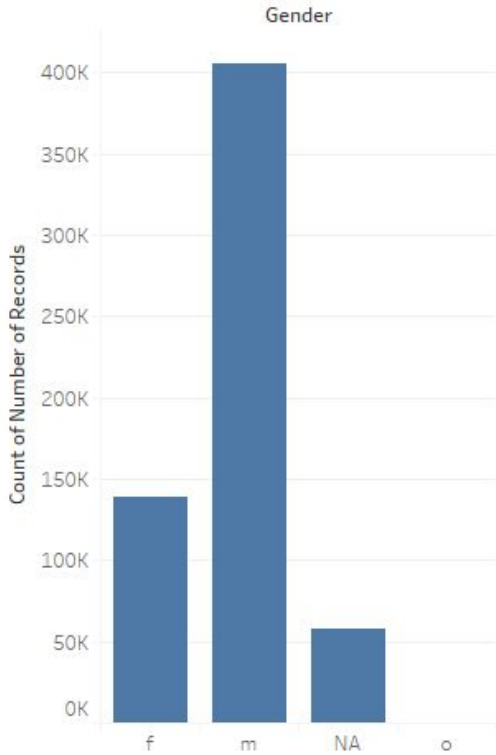
	<div><p>LoE DI</p><table><thead><tr><th>Education Level</th><th>Number of Records (Approx.)</th></tr></thead><tbody><tr><td>Bachelor's</td><td>220,000</td></tr><tr><td>Doctorate</td><td>10,000</td></tr><tr><td>Less than Secondary</td><td>10,000</td></tr><tr><td>Master's</td><td>120,000</td></tr><tr><td>NA</td><td>60,000</td></tr><tr><td>Secondary</td><td>170,000</td></tr></tbody></table></div> <div><ul style="list-style-type: none">Also using domain knowledge we can say that age and LoE_DI should be related. As people belonging to same age-group should have similar highest qualifying degree.</div>	Education Level	Number of Records (Approx.)	Bachelor's	220,000	Doctorate	10,000	Less than Secondary	10,000	Master's	120,000	NA	60,000	Secondary	170,000	<p>some minimum level of education.</p>
Education Level	Number of Records (Approx.)															
Bachelor's	220,000															
Doctorate	10,000															
Less than Secondary	10,000															
Master's	120,000															
NA	60,000															
Secondary	170,000															
YoB	<div><ul style="list-style-type: none">This attribute provides Year of Birth for an individual. Again this is very important attribute for our later analysis and hence can't be ignored.On inspection we find that there are 96605 missing values in this attribute which is 15% of the total entries.Most of the students were born between 1986-1994. This can be seen from below highlight table.</div>	<div><p><u>Approach 1</u></p><p>Use median age given in Kaggle dataset group by Course and the Year it was offered.</p><p><u>Approach 2</u></p><ol style="list-style-type: none">by filling mode/mean by grouping data on the basis of LoE_DI. Find the YoBThere will be 1 row which will both YoB and</div>														



LoE_DI
missing.
Delete that
row.

- Since, LoE_DI and YoB have some relation on the basis of age-group. We can use LoE_DI to estimate YoB values.
- But this solution is possible only if there are very few rows having missing values for both of them jointly because we are already using YoB to estimate LoE_DI. So, we ran a small R script to find this out.

```
print(paste("Number of records with both Year of Birth and highest degree missing is",
            sum(is.na(data['LoE_DI']) && is.na(data['YoB']))))
[1] "Number of records with both Year of Birth and highest degree missing is: 1"
```

	This means there is only one row which have both of them missing and this can record can be ignored for our analysis.											
gender	<ul style="list-style-type: none">13.53% of records have missing gender values which is 58132 rows.We do not have gender based dependent attributes available in our dataset.Frequency Distribution is:-<table><tr><th>Gender</th><th></th></tr><tr><td>f</td><td>1,39,240</td></tr><tr><td>m</td><td>4,05,288</td></tr><tr><td>NA</td><td>58,132</td></tr><tr><td>o</td><td>5</td></tr></table><p>‘m’ is the mode for this categorical attribute.</p><ul style="list-style-type: none">Since we don’t have major contribution of gender in overall analysis, we can ignore records having missing values for Gender based Analysis.	Gender		f	1,39,240	m	4,05,288	NA	58,132	o	5	<ol style="list-style-type: none">For gender - based analysis, simply ignore the records having missing values.In case, gender value is required for overall analysis, impute mode value i.e. ‘m’.
Gender												
f	1,39,240											
m	4,05,288											
NA	58,132											
o	5											
grade	Here the missing value NA specified for students are meaningful.	Ignore the missing values.										

	It implies that the course has not been completed by student yet. Hence, he was not awarded any grade.	
nevents	<ul style="list-style-type: none"> This attribute provides number of interactions with the course. On inspection we find that there are 31% of missing entries for this attribute. NA implies after registering the course students has never interacted with it. 	NA's should be replaced with 0.
ndays_act	<ul style="list-style-type: none"> This attribute provides number of unique days students interacted with the course. On inspection we find that there are 25.38% of missing entries for this attribute. NA implies after registering the course students has never interacted with it. 	Replace NA with 0's
nplay_video	<ul style="list-style-type: none"> This attribute provides number of videos student has played in the course. On inspection we find that there are 71.3% of missing entries for this attribute. We have very less data. So just ignore this attribute and use it only for special purpose. 	Ignore the records
nchapters	<ul style="list-style-type: none"> This attribute provides number of chapters student has interacted with in the course. On inspection we find that there are 40.35% of missing entries for this attribute. NA implies after registering the course students has never interacted with it. 	Replace NA with 0.
roles	It has 100% missing values and hence can be simply ignored.	Drop this attribute
incomplete_flag	<ul style="list-style-type: none"> It has 84.37% missing values. The missing values means that the data is not incomplete. So we fill 0 in place of NAs 	Replace NAs with 0

[R Script](#)
[Tableau](#)

3. Multipurpose Fields

An attribute which can be used for more than one purpose depending on the task and goal that is to be achieved.

In our dataset, we have 'final_cc_cname_DI' which is the 'Country Name' from which the user

has registered for the course. We can treat this field as the 'Country of Birth' too as we do not have a separate field for it. Hence it can serve two purpose: Country of Registration and Country of Birth.

4. Duplicate Data

To check if there are any data points that are duplicates. We will check if any two rows in the dataset are same.

Code:

```
nrow(data)
nrow(unique(data))
duplicates<-nrow(data)-nrow(unique(data))
cat("Number of duplicate records: ", duplicates)
```

Output:

```
641138
```

```
641138
```

```
Number of duplicate records: 0
```

Hence, there are no two rows which are completely identical to each other. Therefore we can conclude that all our data points are unique.

5. Contradicting Data

We can have contradicting data for:

1. If YoB is greater the year of start_time_DI.
2. If registered is false(0) but certified is true(1).
3. If YoB is greater the year of last_event_DI.

1. For checking "If YoB is greater the year of start_time_DI"

First we have to consider only those YoB and those start_time_DI that are not NA. Creating a subset of only those data points that have YoB and start_time_DI as not NA.

Code:

```
Subs1<-subset(data, (!is.na(data["YoB"])) & (!is.na(data["start_time_DI"])))
```

After removing the rows, we get 544533 rows that do not have YoB or start_time_DI as NA.

Code:

```
nrow(Subs1)
```

Output:

```
544533
```

Now, we extract just the year from start_time_DI for comparison with YoB.

Code:

```
Subs1$start_year <- substr(Subs1$start_time_DI, 1,4)
```

Subs1\$start_year is a new attribute that contains just the start year.

Now, we have to find those data points where YoB is greater than start year. This would be a contradiction as a person cannot have start_year less than his/her year of birth.

Code:

```
count<-0
for(i in 1:544533)
{
  if(Subs1$start_year[i]<Subs1$YoB[i])
  {
    count<-count+1
  }
}
cat("Number of contradictions:",count)
```

Output:

```
Number of contradictions: 0
```

For this part, we can see that there are no contradictions. Now checking for “If YoB is greater the year of last_event_DI”

Similar to above, we will consider only those YoB and last_event_DI that are not NA.

Code:

```
Subs2<-subset(data, (!is.na(data["YoB"])) & (!is.na(data["last_event_DI"])))
```

However, on printing we see that even though last_event_DI is not NA, many of them are blank “”.

Code:

```
print(Subs2)
```

Output:

	YoB	gender	grade	start_time_DI	last_event_DI	nevents	ndays_act
19330	2012	m	0.00	2013-08-30		NA	NA
19331	1987	m	0.00	2012-07-24		NA	NA
19332	1968	f	0.00	2012-07-24		NA	NA
19333	1989	m	0.00	2012-07-24	2013-07-27	6	3
19334	1978	m	0.00	2012-07-24		NA	NA
19335	1993	m	0.00	2012-07-24	2012-12-24	107	8

19336	1988	m	0.00	2012-07-24	2013-03-28	8	1	
19337	1993	f	0.00	2012-07-24			NA	NA
19338	1981	m	0.00	2012-07-24			NA	NA
19339	1980	f	0.00	2012-12-19			NA	25
19340	1980	f	0.00	2012-12-19			NA	25

Code:

```
Subs2$start_year <- substr(Subs2$last_event_DI, 1,4)
print(Subs2$start_year)
```

Output:

```
[191] "" "2013" "2013" "2013" "" "" "2013" "" "" "2013"
[201] "2013" "2013" "2013" "" "2013" "2013" "" "2013" "2013" "2012"
[211] "2013" "2013" "2013" "" "2013" "" "2012" "2012" "" ""
[221] "" "2013" "" "" "2013" "" "2013" "" "" ""
[231] "" "2013" "2013" "" "2012" "2013" "2013" "" "" ""
[241] "" "" "" "" "" "2012" "2013" "2013" "2013" "2012"
[251] "2012" "" "" "2013" "2012" "" "2013" "" "" "2013"
```

Hence, we need to first clean the blank spaces. We will ignore all those rows that have last_event_DI as "".

Code:

```
Subs2<-subset(Subs2, Subs2$start_year!="")
print(Subs2$start_year)
```

Output:

```
[11] "2013" "2013" "2012" "2012" "2013" "2013" "2013" "2013" "2013" "2013"
[21] "2013" "2013" "2013" "2013" "2012" "2013" "2013" "2013" "2013" "2012"
[31] "2013" "2012" "2013" "2013" "2012" "2012" "2013" "2013" "2013" "2013"
[41] "2013" "2012" "2013" "2013" "2012" "2013" "2013" "2013" "2013" "2012"
[51] "2012" "2013" "2013" "2012" "2013" "2013" "2013" "2012" "2013" "2013"
[61] "2013" "2013" "2013" "2013" "2013" "2013" "2013" "2013" "2012" "2013"
[71] "2013" "2012" "2013" "2013" "2013" "2013" "2013" "2013" "2013" "2013"
[81] "2013" "2013" "2013" "2013" "2013" "2013" "2012" "2013" "2013" "2013"
[91] "2013" "2012" "2012" "2013" "2013" "2013" "2013" "2013" "2013" "2012"
```

Now, the number of rows that do not have YoB as NA or last_event_DI as NA or blank are:

Code:

```
nrow(Subs2)
```

Output:

386573

Checking for contradictions such that YoB is greater than last_event_DI.

Code:

```
count<-0
for(i in 1:386573)
{
  if(Subs2$start_year[i]<Subs2$YoB[i])
  {
    count<-count+1
  }
}
cat("Number of contradictions:",count)
```

Output:

Number of contradictions: 0

Finally, we check if registered is false(0) but certified is true(1).
First, removing all rows that have registered as NA or certified as NA.

Code:

```
Subs3<-subset(data, (!is.na(data['registered'])) & (!is.na(data['certified'])))
nrow(Subs3)
```

Output:

641138

Thus, we can notice that we do not have any NA's with regards to registered or certified. Each of the values in registered and certified is filled with a boolean value of 0 or 1, 0 representing false.

We will have a contradiction if we have registered as false ie 0, but certified as true ie 1. This would mean that even though the student did not register for the course he/she received a certificate, which is not possible.

To check this we have:

Code:

```
count<-0
for(i in 1:641138)
{
  if(Subs3$registered[i]<Subs3$certified[i])
  {
    count<-count+1
  }
}
```

```
}
}
cat("Number of contradictions:",count)
```

Output:
Number of contradictions: 0

Here, we are doing `Subs3$registered[i]<Subs3$certified[i]` because this condition will be true only when registered is 0 but certified is 1.

In all our above test cases, we noticed we do not have any contradictions with regards to our data.

6. Data Integration Problems

In this the "course_id" has the format "Institute/CourseNumber/SemesterTime"
eg: "HarvardX/CB22x/2013_Spring".

For this, we need to split the integrated data such that we have a separate attribute for Institute, separate for course number and separate for the time of semester in which the course took place.

Also, due to the already cleaned data from "Kaggle" from the original dataset, we cannot do much analysis using the combined dataset from Kaggle. Hence, we are not integrating our dataset with any other dataset from outside as for now.

We have evaluated our data sets for following Data Quality Indicators:-
by Tehreem Ansari, MT2016145

Data Quality Indicator	Valid/Invalid	Steps to be taken
Accuracy	Not applicable	NA
Domain Integrity	<p>Some values in grade are above 100% which is not a valid grade.</p> <div> <p>Code: <code>length(which(data\$grade > 1))</code></p> <p>Output: 6</p> </div>	<p>Converting the values which are above 100% to 100%.</p> <div> <p>Code: <code>data\$grade[data\$grade>1] =1</code></p> <p>Wherever, the grade is greater than 1, or more than</p> </div>

	Hence, there are 6 values which have more than 100% which is not possible.	100% we make it 1.
Data Type	Data Types in our set are: Date, String, Integer. Flag data is represented as an integer 0 for false, and 1 for true. Date currently is represented as string.	Converting the string format for "YoB", "LoE_DI", "start_time_DI", "last_event_DI" to date format. <pre>> # converting the last_event_DI to date format > last_event_DI<- as.Date(big_student_data\$last_event_DI,format="%d/%m/%y") > big_student_data[["last_event_DI"]]=last_event_DI</pre>
Consistency	Valid	NA
Completeness	Some of the data values are NA's or blank.	Depending on the attribute, doing the following: 1. Ignore the row. 2. Take the mean/mode and fill the empty values with mean/mode. 3. Filling with default value.

[R script for data quality](#)

Data Preparation

Done by: Kanika Narang (MT2016069)

On the Data from dataverse

1. We need to split the column called '**course_id**' into four columns 'institute' , 'course_id', 'year' and 'semester'. Since this column 'course_id' contains all the four merge in one.

```
> # splitting column called course_id into 4 columns called Institute, course_id, semester, year.
> library(tidyr)
> big_student_data<-separate(data = big_student_data, col = course_id, into = c("institute", "course_id", "year_sem"), sep = "/")
> #Now sepearting year_sem into year and semester
> big_student_data<-separate(data = big_student_data, col = year_sem, into = c("year", "semester"), sep = "_")
```

2. Dropping the column '**registered**' because this column contains 1 for all the rows signifying that the student has registered and hence provides no useful information.

```
> #removing the column registered because it contains 1 in each row.  
> big_student_data<- big_student_data[,-6]
```

3. Dropping the column '**roles**' because it is empty for all the rows.

```
> #removing the column roles because it is completely blank  
> big_student_data<- big_student_data[,-22]
```

4. Converting '**start_time_DI**' and '**last_event_DI**' into date type column , since they represent date.

```
> # converting the last_event_DI to date format  
> last_event_DI<- as.Date(big_student_data$last_event_DI,format="%d/%m/%y")  
> big_student_data[["last_event_DI"]]=last_event_DI  
  
> # converting the start_time_DI to date format  
> start_time_DI<- as.Date(big_student_data$start_time_DI,format="%d/%m/%y")  
> big_student_data[["start_time_DI"]]=start_time_DI  
> str(big_student_data$start_time_DI)  
Date[1:641138], format: "2012-12-19" "2012-10-15" "2013-02-08" "2012-09-17" "2012-12-19" "201  
2-09-17" ...  
> |
```

5. There is only one course called 'CS50x which has no data fro the semester

```
> course_id_with_no_semester<- unique(x$course_id)  
> course_id_with_no_semester  
[1] CS50x  
13 Levels: 14.73x 2.01x 3.091x 6.002x 6.00x 7.00x 8.02x ... PH278x  
~ |
```

We are filling this missing data for the '**semester**' using the column 'start_time_DI' it tells the data of course registration. semester column is a factor with 4 levels 'Fall','Spring','Summer' and ''.

We need to replace the '' values of semester column with the value corresponding to month in the column 'start_time_DI'.

Spring is considered to belong to months March, April and May , Fall is considered in month September, October and November and summer is considered for the month June. July and August. Since our data does not have any data regarding winter we will be considering months December, January and February in Fall.

```

> # adding column month containing month from the column start_time_DI
> month=format(big_student_data$start_time_DI,"%m")
> big_student_data<-cbind(big_student_data,month)
> big_student_data$month <- as.character(big_student_data$month)

```

```

> # function to fill the value of semester
> sem2<-function(ele)
+ {
+   if(!(is.na(ele["semester"])) & (ele["semester"]==""))
+   {
+     if(as.character(ele["month"]) %in% c("03","04","05"))
+     {
+       ele["semester"]<- "Spring"
+     }
+     else if(as.character(ele["month"]) %in% c("06","07","08"))
+     {
+       ele["semester"]<- "Summer"
+     }else {
+       ele["semester"]<- "Fall"
+     }
+   }else
+   {
+     ele["semester"]<- (ele["semester"])
+   }
+ }
>
>
> #filling the value of semester
> big_student_data$semester<-apply(big_student_data,1,sem2)
- |
> # removing column month
> big_student_data<- big_student_data[-24]
- |

```

6. Removing Missing values in column '**nevents**' . This column tells about number of interactions with the course, recorded in the tracking logs, its NA if no interactions beyond registration. Thus NA's are replaced with 0, here 0 signifies no interaction.

```

> # replacing NA with 0 in column nevents
> big_student_data$nevents<- ifelse(is.na(big_student_data$nevents),0,big_student_data$nevents)

```

8. Column '**ndays_act**' tells about the number of unique days student interacted with the course. NA shows that the students did not interacted with the course after registration. Thus NA's are replaced with 0.


```
> # replacing Na's in ndays_act with 0 since they signifies that student
> # had no interaction with the course after registration
> big_student_data$ndays_act<- ifelse(is.na(big_student_data$ndays_act),0,big_student_data$ndays_act)
```

9. Column '**incomplete_flag**' identifies records that are internally inconsistent, due to a variety of data issues, including missing tracking logs and one course (CS50x) which has virtually no logs because most of the course content is hosted outside of the edX platform. Here value 1 shows inconsistent column and NA shows consistent column, so we are replacing NA with 0.

```
> # replacing Na's in incomplete_flag with 0
> big_student_data$incomplete_flag<- ifelse(is.na(big_student_data$incomplete_flag),0,big_student_data$incomplete_flag)
```

10. Column '**nchapters**' gives number of chapters (within the Courseware) with which the student interacted, here also NA's are replaced with 0.

```
> # replacing Na's in nchapters with 0
> big_student_data$nchapters<- ifelse(is.na(big_student_data$nchapters),0,big_student_data$nchapters)
```

11. '**nplay_video**' column shows the number of play video events within the course. NA's here shows missing values, arising due to different sources of data (tracking log and Courseware Student Module). Taking the assumption that student whenever interacts with the course, he/she views the video thus I am replacing NA with maximum of the three values from the column 'nchapters', 'ndays_act' and 'nevents'.

```
> #replacing NA's from the column nplay_video with the maximum of 'nevents', 'nchapters'
> #and 'ndays_act'
> big_student_data$nplay_video<- ifelse(is.na(big_student_data$nplay_video),max(big_student_data$nevents,big_student_data$nchapters,big_student_data$ndays_act),big_student_data$nplay_video)
```

12. A new column called "age" can be derived using column "YoB". We just need to subtract the Current Year from YoB. Before the subtraction we need to convert "YoB" into numeric.

```
> # Deriving new column age from yob
> age<- big_student_data$year - big_student_data$YoB
> big_student_data<-cbind(big_student_data,age)
```

Data Preparation On the Data from kaggle

1. This data has no missing value and no NA's. It has a column called 'Instructors' which is a multi-valued columns, thus we need to split it, and then add the new rows for each instructor with remaining data as same.

```
> library(splitstackshape)
> new_student<-cSplit(student_data, "Instructors", ",", "long")
> student_data<-new_student
```

Now the table has 516 rows.

2. We can add a derived column 'dropouts' this will tell the number of students who dropped the course. It is calculated as:

$\text{drop_outs} = \text{participants} - \text{certified} - \text{audited}$

Here we have taken the assumption that the students who certified are not counted in audited.

The data has the number of students who audited more than 50% of the course been considered as audited.

```
> # get drop_outs
> drop_outs<- student_data$Participants..Course.Content.Accessed.- student
_data$Certified - student_data$Audited....50..Course.Content.Accessed.
> student_data<- cbind(student_data,drop_outs)
```

Merging of DATA

Merging of both the dataset is not possible because both the dataset are in different grain. Data from the kaggle is course specific data, they have given aggregated values for all the metrics. Data from dataverse is student level data which provides the data regarding how student interacts with the course he/she is registered.

Splitting the Data into training, test and validation sets

1. We will split data in 9:1 ratio for training/test and validation sets. The validation set will be used to control generalization error.
2. We will use k-fold cross validation on remaining 90% data which will comprise of train-test sets. This will reduce overfitting of the model on train data.

Feature Engineering

New Feature	Description	How to get
Age	This is an approximate of an user	Use Course Year-YoB to

	since we don't have date of birth.	generate this attribute.
Ttal_Day_enroll	This gives total number of days student performed last activity to day of registration	

Done by : Daminee (MT2016045)

Done on the dataverse data.

1. The column 'age' has NA's, I thought of following ways to replace these NA values:
 - a. Data from kaggle has a column called 'Median_Age' this could be used to fill NA values in age column by matching course_id and year.
 - b. Using the column 'LoE' we will be filling the age by the mode of the age of students belonging 'LoE' value of that row.
2. The 'column' called 'LoE' have some NA values, we have following possible ways to replace NA:
 - a. Replace it with minimum education level, with the assumption that the every student should have minimum education level.
 - b. Replace it using the assumption based on the country, since different country could have different minimum level of education.
 - c. Replace it by fixing some age groups to that degree level.
3. The value of the column 'grade' must be in between 0 to 1. I found out that some rows have grade of 1.01 so we are converting it to 1.

```
> length(which(big_student_data$grade>1))
[1] 6
> # replacing grade value to 1 where it is greater than 1
> big_student_data$grade[which(big_student_data$grade>1)]<-1
> length(which(big_student_data$grade>1))
[1] 0
```