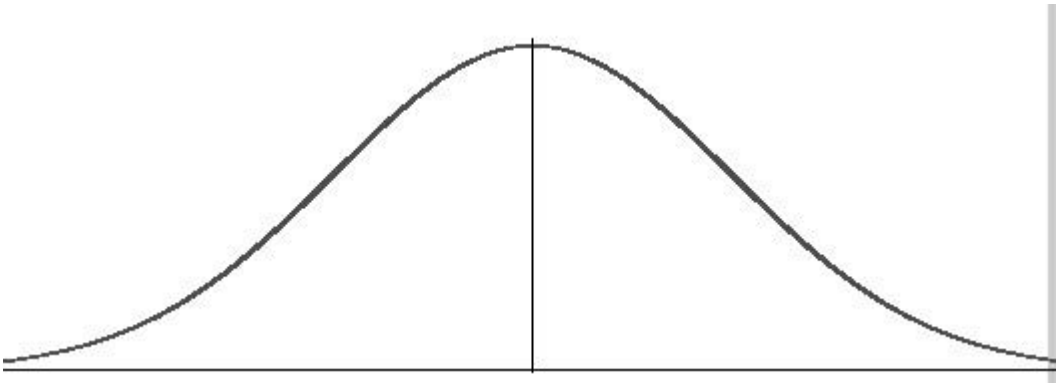
Challenge 1: Analysing complex data: Interesting trends on exam grades

Background

CBSE (Central Board of Secondary Education) is a large education board in India, and millions of students study under that board. The aim of this challenge is to analyse a dataset on the grades obtained in their final year examination (Class 12; equivalent to the A-Levels in the UK) and generate interesting insights out of it.

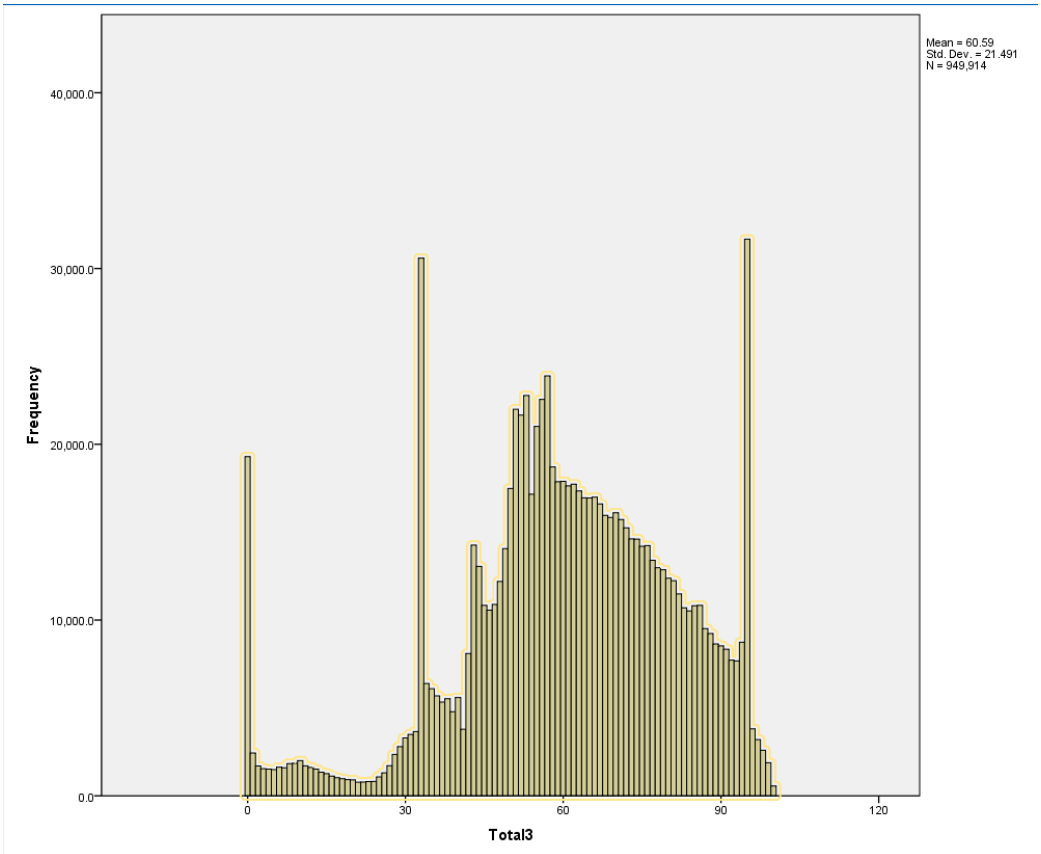
Motivation

The marks of students should roughly follow a normal or Gaussian distribution. This would mean that most students should be concentrated right on the middle, with students doing well, and poorly, falling on either sides of the curve. This is also called as a *bell* curve as the shape of the graph:



(source: https://www.statisticshowto.com/probability-and-statistics/normal-distributions/)

However, perhaps worryingly, this is *not* the graph that would be generated when analysing the CBSE results. Instead, we get one that resembles a multimodal distribution:



(source: own)

This suggests that an external force of some kind is responsible, and does make us wonder – what exactly is going on?

Tasks

Analyse the dataset, at the least generating graphs of the students' performance in various subject groups. Also find and document other interesting insights about this data. For instance,

1. What are the standard parameters of each subject? For instance, standard deviation for Mathematics? Can we get a corelation between them and the resulting graphs for each subject?
2. The names of every candidate are included in this dataset. Is there a potential corelation between that and their overall score?
3. What are the percentiles – and how well does it correlate to the overall score?

Note that all of these above tasks are independent of the background knowledge you have about that board, and that is expected because we have students from all over the world. That being said, you can try to, for instance, explain using the dataset why the graphs resemble what they are (for instance, the multiple modes)?

Also the way the dataset is designed could mean that it is difficult to analyse data per-subject, so it would be fine to analyse them by subject number instead (for instance, nearly all candidates would take English as Subject 1, most take Maths as Subject 2, and so on).

Items provided

Raw dataset for the year 2014 in .csv format: https://universityofstandrews907-my.sharepoint.com/:x:/g/personal/dm282_st-andrews_ac_uk/EUSm5UfCA3dBm6e83WXa2LsBwJrySOPqIpF_38-BtkyBNA?e=Om7EDf

Please do not use any of the data against the interests of this challenge.

Strategies

Use any tool or language you like. You may wish to skip this section if you already know what to do, but otherwise read on, as it gives helpful pointers on where to start.

1. The first step is to convert the CSV file that contains the dataset, into an application or language from which you can perform further analysis. Notice that the *Foundations of Data* learn path uses Python: if you choose to go down that route, you will need to find a way to import the CSV data into Python. Turns out that it does have a module to handle CSV files: https://www.guru99.com/python-csv.html#:~:text=Python%20provides%20a%20CSV%20module,get%20data%20from%20specified%20columns. If you want to use Java, you will most likely have to write your own function, though this isn't difficult considering that every column in that CSV file is delimited.

   Another solution would be to convert to Excel: go to **Data>From Text/CSV** and import the CSV file.

2. The next step is to try to get insights into the data. For instance, converting to a chart. This is something where Excel would be the *wrong* tool – do not try to create a chart using it – it will simply

freeze. Here, Python is a better choice (see the *Design and plot graphs in Python* module), or alternatively, if you can access it, IBM SPSS is a powerful and easy way to quickly import and generate graphs. Alternatively, consider using the programming language R, which is particularly suited for statistical applications like this.

3. Finally, you might want to go further and get even further information about the data, like the percentiles (like what mark is required to be in the top 5% in a subject group, or find the overall percentile for every student). This will likely require use of computational functions like calculating percentile. Again, do *not* use Excel for it, because it is ridiculously slow and could take hours. Python or R are good choices, or alternatively SQL is also possible (by importing the data using the *Import and Export Data* wizard). That being said, it is somewhat tricky to coerce it to calculate the percentiles, so the following piece of (partially commented) code may be of use and may be used without reference (the below code is T-SQL and works on Microsoft SQL Server 2017). Once that data has been computed, you may wish to export it back to Excel.

```
Update dbo.cbse2014 set Percentile1 = NULL
Update dbo.cbse2014 set Percentile2 = NULL
Update dbo.cbse2014 set Percentile3 = NULL
Update dbo.cbse2014 set Percentile4 = NULL
Update dbo.cbse2014 set Percentile5 = NULL
--SELECT count(*)
--(
--     select distinct

--     [Percentile1], [Percentile2], [Percentile3],
--     [Percentile4], [Percentile5]

--     from dbo.cbse2014
--) distinct_fields
-- calculate the percentile for each subject group
UPDATE p1
SET p1.Percentile1 = p1.S1P
FROM (
      SELECT Percentile1, ROUND(100*PERCENT_RANK() OVER (ORDER BY Total1),2) as S1P
      FROM dbo.cbse2014 WHERE Total1 IS NOT NULL
      ) p1
UPDATE p2
SET p2.Percentile2 = p2.S1P
FROM (
      SELECT Percentile2, ROUND(100*PERCENT_RANK() OVER (ORDER BY Total2),2) as S1P
      FROM dbo.cbse2014 WHERE Total2 IS NOT NULL
      ) p2
UPDATE p3
SET p3.Percentile3 = p3.S1P
FROM (
      SELECT Percentile3, ROUND(100*PERCENT_RANK() OVER (ORDER BY Total3),2) as S1P
      FROM dbo.cbse2014 WHERE Total3 IS NOT NULL
      ) p3
UPDATE p4
SET p4.Percentile4 = p4.S1P
FROM (
      SELECT Percentile4, ROUND(100*PERCENT_RANK() OVER (ORDER BY Total4),2) as S1P
      FROM dbo.cbse2014 WHERE Total4 IS NOT NULL
      ) p4
UPDATE p5
SET p5.Percentile5 = p5.S1P
FROM (
      SELECT Percentile5, ROUND(100*PERCENT_RANK() OVER (ORDER BY Total5),2) as S1P
      FROM dbo.cbse2014 WHERE Total5 IS NOT NULL
      ) p5
```

```
-- find the percentile average. Some students might have not taken all subjects and hence we need to account for that. This works since PercAvg is null if any of the percentile data is undefined
UPDATE dbo.cbse2014
SET PercAvg = (Percentile1 + Percentile2 + Percentile3 + Percentile4 + Percentile5)/5
UPDATE dbo.cbse2014
SET PercAvg = (Percentile1 + Percentile2 + Percentile3 + Percentile4)/4 WHERE PercAvg IS NULL
UPDATE dbo.cbse2014
SET PercAvg = (Percentile1 + Percentile2 + Percentile3)/3 WHERE PercAvg IS NULL
UPDATE dbo.cbse2014
SET PercAvg = (Percentile1 + Percentile2)/2 WHERE PercAvg IS NULL
UPDATE dbo.cbse2014
SET PercAvg = Percentile1 WHERE PercAvg IS NULL
-- find percentiles again
UPDATE pperc
SET pperc.PercPercentile = pperc.S1P
FROM (
      SELECT PercPercentile, ROUND(100*PERCENT_RANK() OVER (ORDER BY PercAvg),2) as S1P
      FROM dbo.cbse2014
      ) pperc
UPDATE perc
SET perc.Percentile = perc.S1P
FROM (
      SELECT Percentile, ROUND(100*PERCENT_RANK() OVER (ORDER BY Average),2) as S1P
      FROM dbo.cbse2014
      ) perc
-- return results
SELECT * from dbo.cbse2014 ORDER BY Total2 desc
```

Version 1.0, November 6, 2020