A picture containing text, logo, graphics, font

Description automatically generated

**Project of high school results 2022**

***Name:*** Ahmed Amr Mahmoud.

***ID:*** 42010624.

***Name:*** Ebada Hamdy.

***ID:*** 42010494.

***Name:*** Ahmed Mohamed.

***ID:*** 42010497.

***Name:*** Kareem Gaber.

***ID:*** 42010549.

***Name:*** Sadek Ashraf Khattab.

***ID:*** 42010750.

***Name:*** Seif Farid Elnemr.

***ID:*** 42010551.

***Instructor:***

***Eng- Ahmed Nousir.***

**A picture containing clothing, person, learning, education

Description automatically generatedA picture containing text, screenshot, circle, watch

Description automatically generated**

First, we summon panda’s library to read in a CSV file named "High\_School\_Public\_Results\_2022\_EG\_both\_attempts.csv" and store it in a variable called df.

* In this line(df.loc[df['branch'] == 'أدبي')

We will write "غير مقرر" for material that is not accepted by the column “أدبي.”

The condition on the 'branch' column of the Data Frame, where only rows that have a value of 'أدبي' in this column will be selected.

The fillna method is used to replace any missing values in the selected columns with the string 'غير مقرر'. The result of this operation is then assigned back to the same columns in the Data Frame using the loc method.

* df.loc[df['branch'] == 'علمي علوم'

We will write "غير مقرر" for material that is not accepted by the column “علمي علوم.”

The fillna method is used to replace any missing values in the selected columns with the string 'غير مقرر'. The result of this operation is then assigned back to the same columns in the Data Frame using the loc method.

* df.loc[df['branch'] == 'علمي رياضة

We will do the same thing that we did with the previous two columns, but with "علمي رياضه."

* df['status'] =df['status']. replace ('دور ثاني','راسب')

This code updates the 'status' column of the df Data Frame by replacing any occurrences of the string 'دور ثاني' with the string 'راسب'.

* df = df. dropna(subset=['status'])
* nan\_rows = df[df['status']. isna()]

-drop any missing value in “desk\_no” column

* df.loc[df['Percentage'] == 0

-we will fill all missing values in all materials with “0” on the 'Percentage' column

* df.loc[df['status'] == 'راسب'
* . fillna(0)

in the “status” column if we find any value with “راسب” or if we find missing values, we will replace them with “0”.

* df.loc[df['status'] == 'ناجح'
* . fillna('غير مدرج')

the 'status' column of the Data Frame, where only rows that have a value of 'ناجح' in this column will be selected.

The fillna method is used to replace any missing values in the selected columns with the string 'غير مدرج'.

-in the column “gender” if we find a missing value, we will replace it with ‘F’

* dw = df
* for column in df.columns:
* dw[column] = df[column].replace('غير مدرج', 0)

This code creates a new Data Frame dw with the same columns as df. For each column in df, the code replaces any occurrence of the string 'غير مدرج' with the integer “0”, and assigns the result to the corresponding column in dw.

* for column in df.columns:
* dw[column] = df[column].replace('غير مقرر',0)

For each column in df, the code replaces any occurrence of the string 'غير مقرر' with the integer “0”, and assigns the result to the corresponding column in dw.

* x1 = df[df['gender'].isna()]

-in this line we return the number of missing values in each column.

-we do this command in all columns to determine rate of missing value in each column.

**A diagram of data science process

Description automatically generated with medium confidence### analysis**

(1) We count the number of students in each city in the dw Data Frame. The resulting Data Frame is sorted in descending order by the number of students, and then the column name 'index' is changed to 'city' using the rename method.

(2) we calculate the percentage of students in each city in the dw Data Frame. The resulting Series is sorted in descending order by the percentage of students, and then the column name 'index' is changed to ‘city\_student\_rate’ using the reset\_index method and the name parameter.

(3) We count the number of students in each administration in the dw Data Frame. The resulting Data Frame is sorted in descending order by the number of students, and then the column name 'index' is changed to 'administration' using the rename method.

(4) here we calculate the percentage of students in each administration in the dw Data Frame. The resulting Series is sorted in descending order by the percentage of students, and then the column name 'index' is changed to 'admin\_student\_rate'

(5) counts number of gender if it “male” or “female”, then calculates the percentage of students in each gender in the dw Data Frame. The resulting Series is sorted in descending order by the percentage of students, and then the column name 'index' is changed to 'gender\_student\_rate'

(6) in “status” column we add status\_student\_num that counts the number of students in each status in the dw Data Frame. The resulting Data Frame is sorted in descending order by the number of students. It also calculates the percentage of students in each status in the dw Data Frame and generates a new Data Frame status\_student\_rate. The resulting Data Frame is sorted in descending order by the percentage of students.

(7) After this we make a list of top 10 students depend on “percentage” column and “branch” column, we need top 10 in all three majors “أدبي”"علمي رياضه" "علمي علوم"

(8) then we count number of student in (adaby) ,(math),(science)

(9) Count of passing student in each City depend on “city” and “status”

Columns, then add then in 'city\_passing\_students\_num' column

(10) Count the failed students in each city with “city” and” status” columns , then add them in 'city\_failed\_students\_num'

--we generate a new Data Frame admin\_passing\_students\_num that counts the number of passing students in each administration in the dw Data Frame. This is achieved by first selecting only the rows where the status\_column equals 'ناجح', grouping the results by the admin\_column, counting the number of rows for each group, and renaming the resulting column to 'admin\_passing\_students\_num'. The resulting Data Frame is sorted in descending order by the number of passing students.

Also, we generate a new DataFrame admin\_falling\_students\_num that counts the number of falling students in each administration in the dw Data Frame. This is achieved by first selecting only the rows where the status\_column equals 'راسب', grouping the results by the admin\_column, counting the number of rows for each group, and renaming the resulting column to 'admin\_falling\_students\_num'.

We generate a new Data Frame branch\_passing\_students\_num that counts the number of passing students in each branch in the dw Data Frame. This is achieved by first selecting only the rows where the status\_column equals 'ناجح', grouping the results by the branch\_column, counting the number of rows for each group, and renaming the resulting column to 'branch\_passing\_students\_num'. The resulting DataFrame is sorted in descending order by the number of passing students.

Then, the code adds two new columns to the DataFrame: 'branch\_students\_num' and 'rate\_passing\_students'. The 'branch\_students\_num' column contains the total number of students in each branch, and the values are manually assigned based on the given data. The 'rate\_passing\_students' column contains the percentage of passing students in each branch, calculated as the ratio of 'branch\_passing\_students\_num' to 'branch\_students\_num' multiplied by 100.

---as well as we do in the last example, we do it again with failed students in each branch.

# We have 6 percentages of which 5 are for success and one for failure

(1) generates student\_up\_90\_num that contains only the rows where the 'Percentage' column in the dw DataFrame is greater than or equal to 90. The resulting DataFrame includes all columns from dw.

Then, the code calculates the percentage of students with a 'Percentage' value of 90 or higher in the dw DataFrame. This is achieved by counting the number of rows in the student\_up\_90\_num DataFrame and dividing it by the total number of rows in the dw DataFrame. The resulting value is multiplied by 100 to obtain the percentage.

(2) as the same idea of previous ex, we need number of students between 80% to 90%

Also, from 70% to 80%

And, from 60% to 70%

And, from 50% to 60%

Finally, we calculate number of students that have less than 50% and they will be failed students.

--we need to know abroad\_student\_num that contains only the rows where the “school\_name” column in the dw DataFrame contains the string 'ابناؤنا فى الخارج'

We calculate the correlation matrix of the dw DataFrame and saves it to the correlation variable. The sns.heatmap() function is then used to create a heatmap plot of the correlation matrix with annotations, line widths and colors.

We create two histograms of the 'gender' and 'city' columns of the dw DataFrame using the plt.hist() function.

Then, we create a list of column names in the data variable and selects those columns from the dw DataFrame and stores them in the x variable. The 'Percentage' column is selected and stored in the y variable. The train\_testsplit () function is then used to split x and y into training and testing sets with a test size of 30%, and a random state of 42.

A picture containing text, screenshot, person, technology

Description automatically generated

#**machine** \\a Linear Regression () object is created and fitted to the training data using the fit () method. The model is then used to predict the values of y for the test data using the predict () method and the predicted values are stored in the prediction variable. The mean\_squared\_error () and mean\_absolute\_error () functions are then used to calculate the mean squared error and mean absolute error between the predicted values and the actual values of y for the test data. Finally, the score () method of the Linear Regression () object is used to calculate the R-squared value of the model for the test data and stored in the squared variable.