International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Design of Data Retrieval System for providing Live Feed of Enrollments in Universities using Data Visualization

Anish H. Narkhede[1]

[1]Department of Computer Engineering, P.V.P.I.T, Savitribai Phule Pune University(formerly University of Pune), Pune

Abstract: ***Data Retrieval Systems are playing an ever increasing role in several industries in today' s world. Information that is gathered by such Systems, allow organizations to expand their outreach in a more effective way. Traditionally, universities provide a statistics of enrollments in a particular branch of study with the help of reports. However, these reports are generated annually or bi-annually. Changes are bound to occur in the admissions in between this period. The System proposed in this paper, aims at gathering admissions data, and providing a graphical representation of the current status of enrollments, as and when they occur in the form of a Live Feed. This would allow the administration department at a particular university to have a better understanding of the current enrollment status, as data visualization provides a much better understanding as compared to numerical data.***

Keywords: Data analysis, data visualization, graphs, live feed

1. Introduction

In recent years, analysis of data, may it be data regarding the weather forecast, statistics related to the performance of an athlete, or huge data dumps that show trends of current fan following of a particular movie franchise, have become vital to the respective fields. These trends that are obtained from analyzing huge sets of data, allow businesses to expand their operations in a way that would optimize their impact in their respective fields, and at the same time, be of huge help to the general public as well. This paper proposes a system, which could be used by any Institute/University that would allow analysis of the variations in admissions over a fixed period of time. Instead of using traditional report based statistics, the System aims at providing Visual Information of the admissions data. It can be used to analyze the trends of admissions into a particular branch of study, thus allowing the Institute to have a fair idea of the increase and decrease of enrollments.

2. Working

The proposed system initially involves collection of data from the users. This would be done using a Python Tkinter GUI application, that would consist of general information fields regarding the student. It would also have a field to collect information regarding the particular branch of study in which the student is enrolling into. Further, it would also fetch information regarding the year in which a particular student is taking admission in the University. This would allow the system to monitor admissions ratio on a yearly basis. All this data will be then stored in a MySQL Database. In order to make the data retrieval faster and optimized, the system will access database only at the time of data insertion. At the time of representing the data in a graphical manner, the system will not query the Databases every single time. Instead, it would use Data Persistence (Pickling in Python). A primary List will be maintained, which would have the admission years of all the students. Similarly, a List will be maintained for the admissions in a particular branch of study, that would correspond to the Years List. These Lists will then be 'Pickled'and accessed at an interval of 1000milliseconds, in order to represent them in a graphical form, using Python Matplotlib library, so as to provide a Live Feed.

3. Literature Review

3.1 Introduction

Data visualization is regarded by many disciplines as a modern equivalent of visual communication, which involves study of visual representation of data[1]. The primary goal of data visualization is to communicate information clearly and efficiently to the end users by using graphs, plots and charts. Such type of information allows the users to analyze and reason the data and evidence. Usage of large amounts of data is not something new. However, data created by internet activity, expanding number of sensors in the world that we live in, have resulted into creation of these huge data sets.Thus, with rate at which data is being generated these days, it has become important that these enormous data sets be analyzed and be represented in a visual form.

3.2 Need for Data Visualization

There is a human side to data visualization. Cognition refers to processes in human beings like perception, learning, concept formation and problem solving. As a picture can speak a thousand words, data displayed graphically allows for a much easier comprehension of the information that is to be conveyed.

Table 1: Enrollment details of students at Universities

1st Year	1110	1144	2254	139	149	288	2542
2nd Year	783	851	1634	20	26	46	1680
3rd Year	967	956	1923	38	35	73	1996
4th Year	1285	1279	2564	72	53	125	2689
Graduate	3748	3315	7063	2241	1377	3618	10681
Professional	795	665	1460	4	3	7	1467
Special	39	19	58	111	175	286	344
Grand Total	8727	8229	16956	2627	1820	4447	21403

As mentioned earlier, Universities/Institutes typically provide information regarding the student admissions in different fields of study in the form of reports.These static reports, do provide insight into the school's enrollment, however, cannot be easily deciphered into information that is intuitive. The statistics and data provided by schools should be in a format that allows any person to quickly glance at it, and provide him/her a general overview of the current situation. Provided information should have the following characteristics:

- Present many numbers in a small space
- Make large data sets coherent
- Encourage the eye to compare different pieces of data
- Reveal the data at several levels of detail, from a broad overview to the fine structure

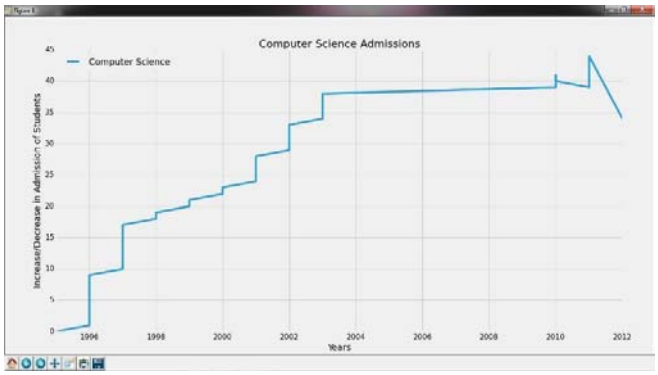
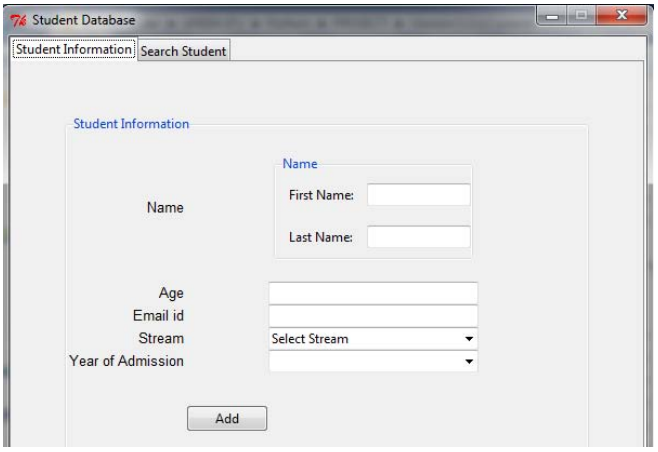


Figure 1: Computer Science admissions data represented using the proposed System (using Test data)

4. Methodology

4.1 Tkinter

Tkinter is an integral part of Python, which provides robust and platform independent windowing toolkit, which can be accessed in Python using the 'Tkinter' module. It is a set of wrappers that implement TK widgets as Python classes. Its chief virtues would be that it is fast and comes bundled along with Python. The system mainly uses Tkinter GUI in order to interact with the Human, for the purpose of data gathering, by using typically used forms.



4.2 DataFrame

DataFrame is a 2D labelled data structure with columns of different types. A DataFrame acts as spreadsheet or a SQL Table. It is a heterogeneous collection set which accepts different types of data inputs:

- Dict or ID narrays, lists, dictionaries
- Structured or record ndarray
- Another DataFrame

Optionally the DataFrame also accepts Index for rows and Column Labels for Columns as arguments. The system makes use of a DataFrame in order to store the admission count of a particular branch of study, and also to store the corresponding year of admission. This DataFrame is then plotted using a Matplotlib Line graph[2]

pickle.dump(obj,file) :

This function is used to write a pickled representation of the object that is to be serialized. 'file' represents the name of the file which would contain the byte stream.

pickle.load(file) :

Reads a string from the 'file', interprets it as a pickle data stream, and reconstructs the original object hierarchy.

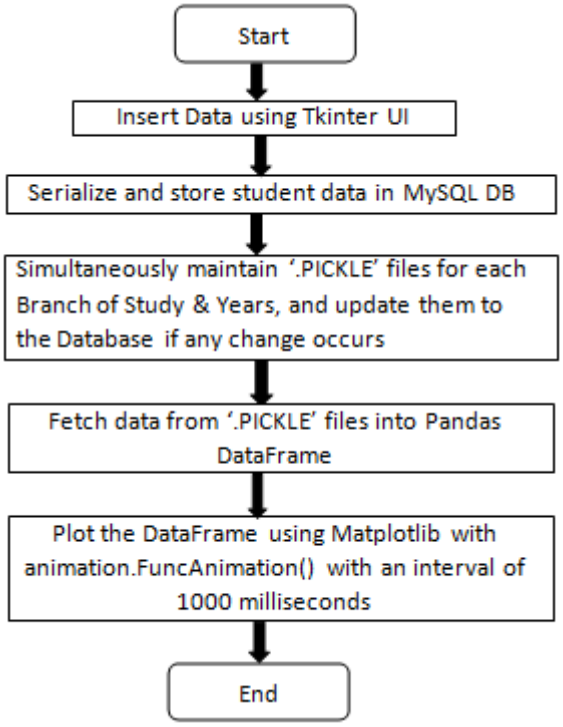
4.3Python Pickle Module

The Pickle module in Python is used for storing Python data in a persistent form on the disk. The 'pickle' module is capable of serializing various Python data types into streams of bytes[3]. This stream can later be used to recreate the same objects. Perhaps the most important feature of pickling is that the 'pickle' module can transform any sort of data item into a stream of bytes, and can later transform the byte stream into data objects, with the same internal structure. Pickling would be implemented in this system, in order to create persistent byte streams of Lists that are used to store Admissions data and also the Years in which the admissions happened. This byte stream would then be stored in a Relational Database. Another advantage of using Pickle would be, that a non-Python program would not be able to de-serialize the byte stream into a data object, thus protecting the data, as serialization is done in a Python-specific way.

4.4 Matplotlib

Matplotlib.pyplot is a collection of functions that allows creation figures, and plot various types of graphs in those figures. Matplotlib consists of standard functions that are used in order to plot data in a meaningful way[4]. Matplotlib is used in order to plot Years List along with the Admissions data.

5. System Design and Implementation



5.1 Data Insertion

The Tkinter GUI provides form fields that allow the user to input his/her personal data, which includes all relevant and important details such as, the Branch of Study and Year of Admission. This data is taken into a Python class and then serialized using the 'pickle' module.

5.2 Data Persistence/Serialization

This user data that is collected, is now serialized and added to a MySQL database, where it is stored. Simultaneously, the application also maintains '.PICKLE' files of the Branch of Study list, as well as the Year of Admissions list. Instead of querying the Database frequently, the pickle files provide faster access to user data, which can easily be reconstructed. The following code shows how a list which contains the count of Computer Science admissions is serialized:

```
def _pickleComp(self):
tempYear = self.yearLabel.get()

p_out=open('compCount.pickle','wb')
pickle.dump(self.compCount,p_out)
p_out.close()

self.COMP.append(self.compCount)
p_out = open('COMP.pickle','wb')
pickle.dump(self.COMP,p_out)
p_out.close()

self.COMPyear.append(int(tempYear))
p_out = open('COMPyear.pickle','wb')
pickle.dump(self.COMPyear,p_out)
p_out.close()
```

The system uses the Python 'Pickle' module. Using the standard functions provided by it, it first opens the already pickled file, and then reconstructs it. After reconstruction, it then appends the newly added data to the admissions data list, and then re-pickles it. A similar technique is used at the time of deletion of data. If a student de-enrolls from the School, then the list is manipulated in a way such that all the entries, except the last one, are retained. The last entry is taken into a variable, and then its count is decreased by 1. This decrease in count is done so as to represent either the student's successful completion/graduation from the University, or his de-enrollment from the University due any other reason. The following code shows this manipulation for the 'Computer Science' admissions data:

```
if stream == 'Computer Science':
newCount = []
string_count = 'COMP.pickle'

tempCount = open(string_count,'rb')
Count = pickle.load(tempCount)

last_Count = Count[-1]
tempLen = len(Count)

for i in range(0,tempLen-1):
newCount += [Count[i]]

if last_Count>0:
newCount.append(last_Count-1)

saveFile = open(string_count,'wb')
pickle.dump(newCount,saveFile)
saveFile.close()

saveFile = open('compCount.pickle','wb')
pickle.dump(last_Count-1,saveFile)
saveFile.close()
```

5.3 Creating Pandas DataFrame

The data from the pickle files, is taken into a Dictionary, in order to create a sort of a 'key/value' pair association that allows efficient plotting of data on a x-y axis Graph. A DataFrame is created for each Dictionary. This DataFrame is then plotted using Matplotlib.

The following code shows how Computer Science admissions list data is first put into a Python Dictionary, and then into a Pandas DataFrame:

```
comp = open('COMPyear.pickle','rb')
compList = pickle.load(comp)
compCount = open('COMP.pickle','rb')
compCountList = pickle.load(compCount)
comp_data = {'Years':compList,
'Computer Science': compCountList}
compDF = pd.DataFrame(comp_data)
compDF.set_index(compDF['Years'],inplace
=True)
```

The `comp_data`is a python dictionary, which essentially consists of two lists, viz. the 'Years' list, and the 'Computer Science' list. This Dictionary is then converted into a Pandas DataFrame, and the index of the DataFrame is set to 'Years', so that the plotting of this DataFrame would be Years against Admissions.

5.4 Plotting DataFrame using FuncAnimation

The DataFrame which contains data regarding every student's admission information, and also the year of admission, is plotted using FuncAnimation function, provided by Matplotlib[4].
class matplotlib.animation.**FuncAnimation**(fig, func, frames=None, init_func=None,
fargs=None, save_count=None, **kwargs)

This function makes an animation by repeatedly calling the function *func*. The argument **kwargs includes interval delays. Depending upon the interval specified, the function refreshes the content that is plotted. It is typically represented in milliseconds.

The following code shows how the System using FuncAnimation to provide Live Feed of the data:

```
import matplotlib.animation as animation
fig = plt.figure()
ax = fig.add_subplot(1,1,1)
ax.clear()
def _plot(i):
compDF['Computer
Science'].plot(linewidth=2,color='b')
ax.set_title('Computer Science
Admissions')
plt.ylabel('Increase/Decrease in
Admission of Students')
plt.legend(loc='upper left')
ani =
animation.FuncAnimation(fig,_plot,interv
al=1000)
plt.legend()
plt.show()
```

In this code, 'ax' is the main subplot which is to be plotted.The function `def_plot(i)`plots the `compDF` DataFrame that is mentioned in the earlier section. The primary purpose of this function is to create an effect that would allow the System to generate a visual output, that is Live. In order to achieve this, it uses a `FuncAnimation` function, which takes the following arguments: The Data to be plotted, the Plot function, and the interval at which the Data is to be refreshed. The system refreshes data at an interval rate of 1000 milliseconds.

6. Conclusion

1) The proposed system makes it much easier for the user to get anoverview of the present admissions scenario at any particular University/Institute.
2) The system will ensure an effective performance with respect to storing Student data in database.
3) The System also generates a CSV(Excel file) which gives a detailed information of the admissions data, if required.
4) Multiple fields of studies can be monitored individually.
5) The updates that are displayed on the graph provide a Live Feed. So as soon as a new student is admitted to a particular course, the data represented by the System is updated at that very instant.
6) It is up to the University to decide whether it wants to make the admissions data public. This would allow prospective students to get an idea of the current admission trends for every Department in a particular University.

7. Future Scope

Currently, the proposed system works as separate entity. It is not a part of any existing admissions system. However, this system can be integrated into the University's admissions portal, so that the data that is received from the Admission Details form, can be used, instead of having a separate GUI application for taking data entry into the database. Further, as the number of admissions increase, it would be of utmost importance to manage this data more effectively. In order to do so, Huffman Coding can be used to compress the Admissions data and also to store, manipulate and represent it on a graph. The Years Lists and the admission count for every individual branch of study, can be encoded using Huffman Coding. Thus the frequency of admissions in a particular year can be calculated, and then be converted into binary form. The same thing can be done with admissions data of every particular branch of study. At the time of displaying this information on the graph, the encoded data (represented using Huffman's Tree), can then be decoded.

References

[1] http://en.wikipedia.org/wiki/Data_analysis
[2] Pandas documentation: powerful Python data analysis toolkit Release 0.17.0-Wes McKinney & PyData Development Team
[3] The Python Library Reference Release 2.7.11 - Guido van Rossum and the Python development team
[4] Matplotlib documentation Release 1.5.1 - John Hunter, Darren Dale, Eric Firing, Michael Droettboom

Author Profile

Anish Narkhede has recently completed his Bachelor of Engineering in Computer Engineering from Padmabhooshan Vasantdata Patil Institute of Technology, Pune (affiliated to University of Pune) with First Class.