AI SIMILARITIES USING LINUX BASH SCRIPT

Michael Chrisco

PROJECT IN A NUTSHELL

- Compare by classification on data from different origins.

PROJECT IN A NUTSHELL

- Compare by classification on data from different origins.
  - Linux processes

```
1782 ?        Ss     0:00 /usr/lib/gnome-settings-daemon/gnome-settings-daemon
1784 ?        S      0:00 /usr/bin/metacity
1787 ?        S      0:00 /usr/lib/gvfs/gvfs-gdu-volume-monitor
1788 ?        S      0:01 gnome-panel
1793 ?        Sl     0:00 /usr/lib/gvfs/gvfs-afc-volume-monitor
1796 ?        S      0:00 /usr/lib/gvfs/gvfs-gphoto2-volume-monitor
1797 ?        S      0:02 nautilus
1799 ?        Ssl    0:00 /usr/lib/bonobo-activation/bonobo-activation-server -
1807 ?        S      0:00 kerneloops-applet
1808 ?        S      0:00 python /usr/bin/system-config-printer-applet
1809 ?        S      0:00 nm-applet --sm-disable
1810 ?        S      0:00 /usr/lib/policykit-1-gnome/polkit-gnome-authenticatio
1818 ?        S      0:00 bluetooth-applet
1819 ?        Sl     0:00 /usr/lib/gnome-applets/mixer_applet2 --oaf-activate-i
1823 ?        S      0:00 /usr/lib/gnome-disk-utility/gdu-notification-daemon
1852 ?        Sl     0:00 /usr/bin/VBoxClient --clipboard
1857 ?        S      0:00 /usr/lib/evolution/2.30/evolution-alarm-notify
1862 ?        Sl     0:00 /usr/bin/VBoxClient --display
1863 ?        S      0:00 update-notifier
1866 ?        Sl     0:00 /usr/bin/VBoxClient --seamless
1869 ?        Ss     0:00 gnome-screensaver
1878 ?        S      0:00 /usr/lib/gvfs/gvfsd-trash --spawner :1.2 /org/gtk/gvf
1881 ?        S      0:00 /usr/lib/gvfs/gvfsd-burn --spawner :1.2 /org/gtk/gvfs
1888 ?        S      0:00 /usr/lib/gvfs/gvfsd-metadata
```

PROJECT IN A NUTSHELL

- Compare by classification on data from different origins.

  Linux processes

  Text documents (books)

```
Project Gutenberg's Armenian Legends and Festivals, by Louis A. Boettiger

This eBook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever.  You may copy it, give it away or
re-use it under the terms of the Project Gutenberg License included
with this eBook or online at www.gutenberg.net


Title: Armenian Legends and Festivals

Author: Louis A. Boettiger

Release Date: November 25, 2011 [EBook #38129]

Language: English


*** START OF THIS PROJECT GUTENBERG EBOOK ARMENIAN LEGENDS AND FESTIVALS ***
```

PROJECT IN A NUTSHELL

- Compare by classification on data from different origins.
- Linux processes
- Text documents (books)
- How to categorize different data by graphing the result.

BACKGROUND

- AI: to create and replicate actions that humans excel at.

- Given a set of documents, how similar are the documents and what kind of preset data needs to be generated.

- What can we learn with given data?

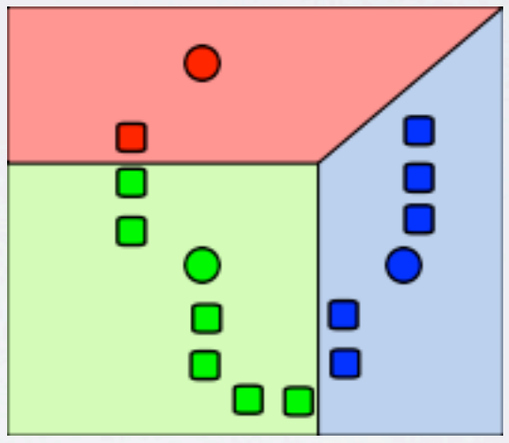
PROJECT #1
CREATE CONCEPT SCRIPT

- given pre-made (human generated) data, make the computer come up with categorizations by itself.

- Use k-means algorithm for the categorizations.

K-MEANS CLUSTERING ALGORITHM

Given a set of observations, partition the points into sets based on the centriod (the center point of the observations).



K-MEANS CLUSTERING ALGORITHM

Given a set of observations, partition the points into sets based on the **centriod** (the center point of the observations).

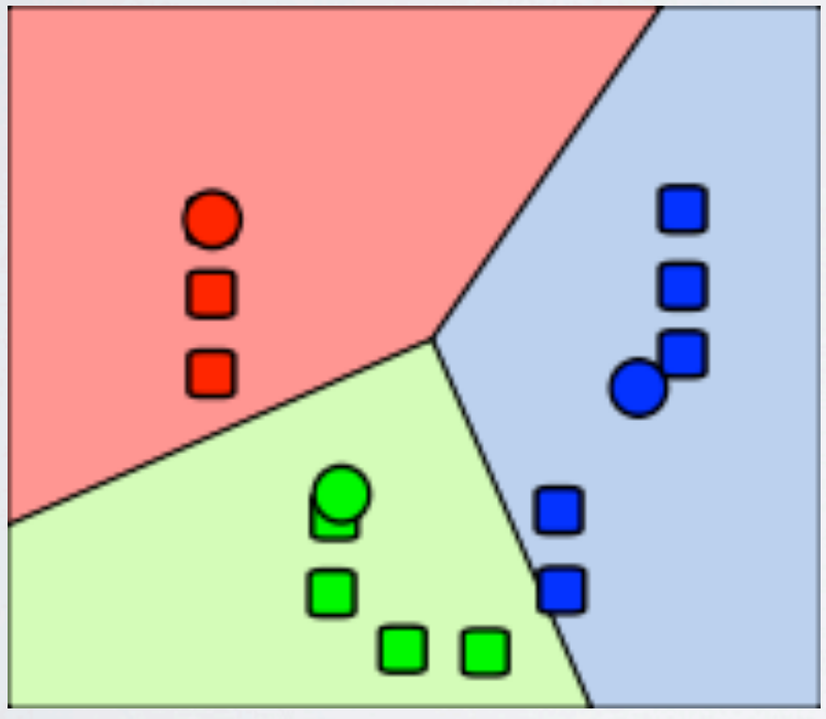
Given set of points

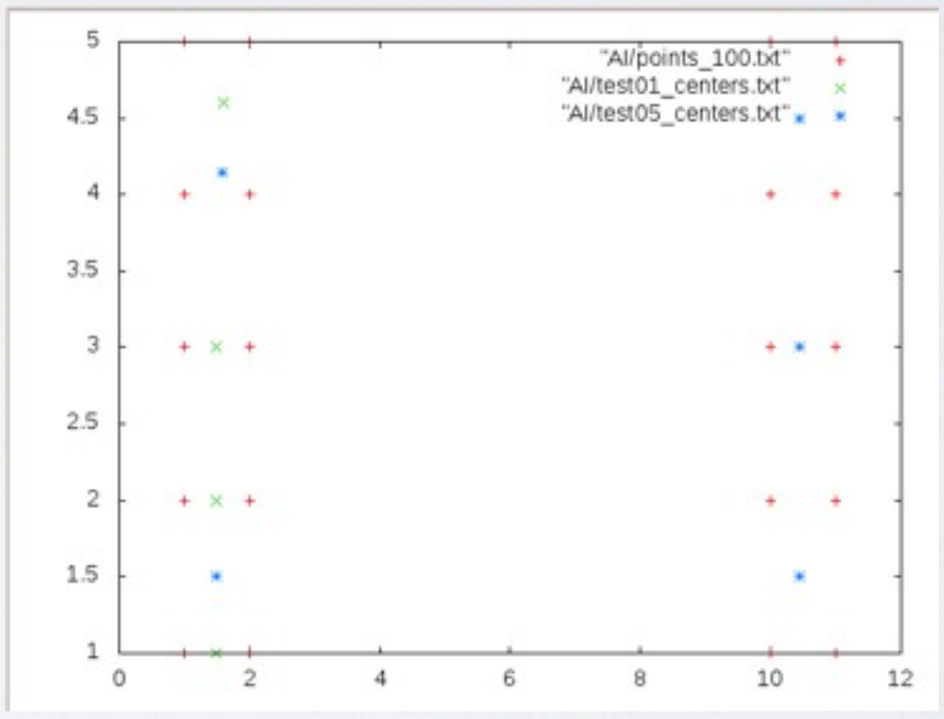
Partitioning given means of points

Create centriods and redo algorithm on each partition

RESULT OF K-MEANS

- 1) Centriods that represent the data
- 2) Partitioning of the data





CODE DESCRIPTION FOR SAMPLE DATA

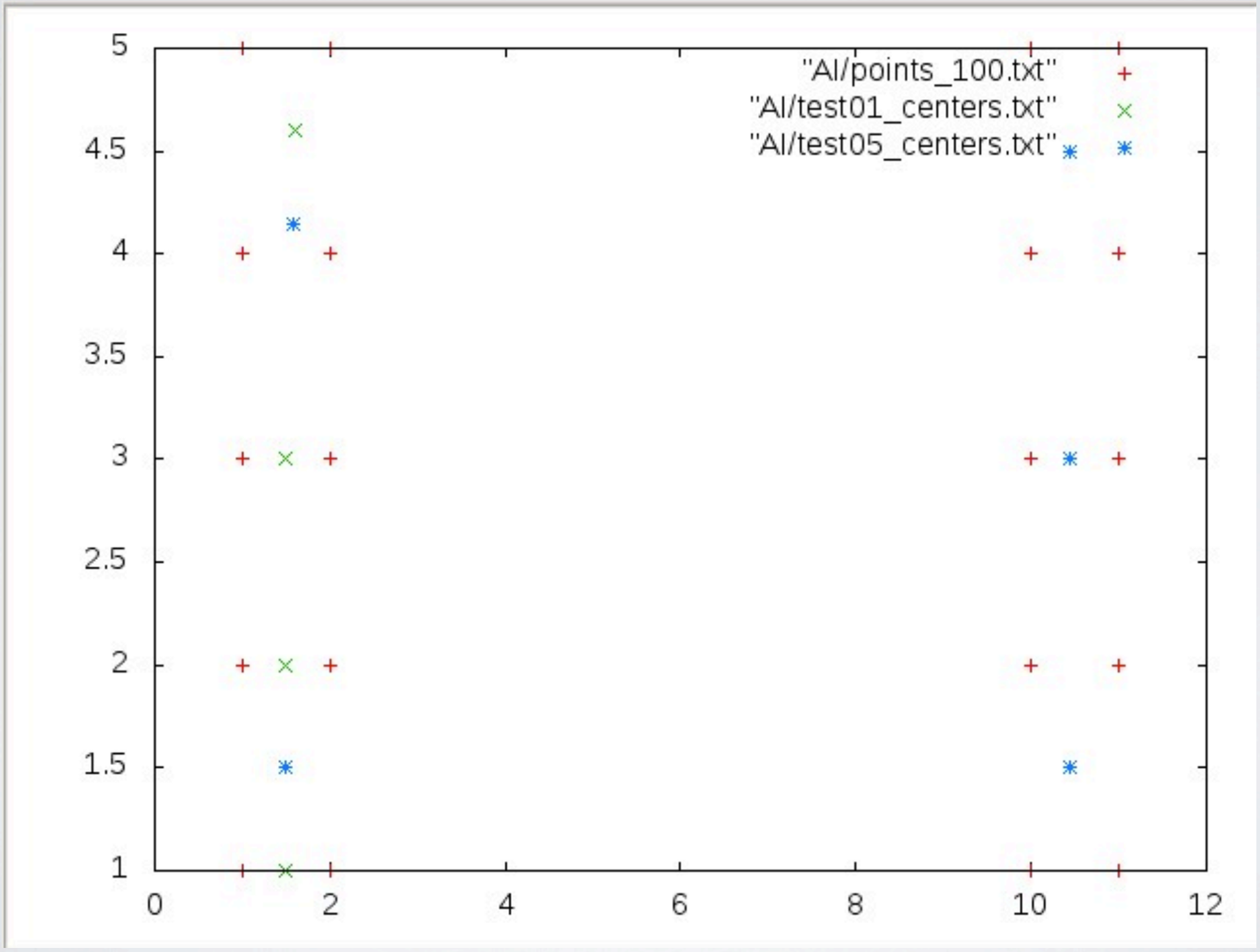
- Run the Main.bash script which will execute the k-means.bash script and the plot.p script.

- k-means script runs the k-means executable (C++ program) and plot the results using the linux tool gnuplot.

- Result will be in AI/output.png

- Given that the data itself is pre-made, the clusters should be around the datapoints to the left and to the right. 5 centroids will be displayed from the first run of the algorithm and 5 centroids from the last run.

RESULTS


"AI/points_100.txt"
"AI/test01_centers.txt"
"AI/test05_centers.txt"
5
4.5
4
3.5
3
2.5
2
1.5
1
0
2
4
6
8
10
12

APPLYING TO LINUX PROCESSES

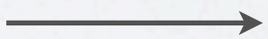
- 1) Given "normal" operations in linux, what kind of information can we learn from the processes themselves?

- 2) What processes are not in the set of "normal" operations for the system?

PRE-PROCESSING DATA

Data (IE process names) are extracted using the ps -x command in linux 100 times. These are snapshots of the running system.

```
1782 ?        Ss     0:00 /usr/lib/gnome-settings-daemon/gnome-settings-daemon
1784 ?        S      0:00 /usr/bin/metacity
1787 ?        S      0:00 /usr/lib/gvfs/gvfs-gdu-volume-monitor
1788 ?        S      0:01 gnome-panel
1793 ?        Sl     0:00 /usr/lib/gvfs/gvfs-afc-volume-monitor
1796 ?        S      0:00 /usr/lib/gvfs/gvfs-gphoto2-volume-monitor
1797 ?        S      0:02 nautilus
1799 ?        Ssl    0:00 /usr/lib/bonobo-activation/bonobo-activation-server -
1807 ?        S      0:00 kerneloops-applet
1808 ?        S      0:00 python /usr/bin/system-config-printer-applet
1809 ?        S      0:00 nm-applet --sm-disable
1810 ?        S      0:00 /usr/lib/policykit-1-gnome/polkit-gnome-authenticatio
1818 ?        S      0:00 bluetooth-applet
1819 ?        Sl     0:00 /usr/lib/gnome-applets/mixer_applet2 --oaf-activate-i
1823 ?        S      0:00 /usr/lib/gnome-disk-utility/gdu-notification-daemon
1852 ?        Sl     0:00 /usr/bin/VBoxClient --clipboard
1857 ?        S      0:00 /usr/lib/evolution/2.30/evolution-alarm-notify
1862 ?        Sl     0:00 /usr/bin/VBoxClient --display
1863 ?        S      0:00 update-notifier
1866 ?        Sl     0:00 /usr/bin/VBoxClient --seamless
1869 ?        Ss     0:00 gnome-screensaver
1878 ?        S      0:00 /usr/lib/gvfs/gvfsd-trash --spawner :1.2 /org/gtk/gvf
1881 ?        S      0:00 /usr/lib/gvfs/gvfsd-burn --spawner :1.2 /org/gtk/gvfs
1888 ?        S      0:00 /usr/lib/gvfs/gvfsd-metadata
```



```
/usr/lib/gnome-settings-daemon/gnome-settings-daemon
/usr/bin/metacity
/usr/lib/gvfs/gvfs-gdu-volume-monitor
gnome-panel
/usr/lib/gvfs/gvfs-afc-volume-monitor
/usr/lib/gvfs/gvfs-gphoto2-volume-monitor
nautilus
/usr/lib/bonobo-activation/bonobo-activation-server -
kerneloops-applet
python /usr/bin/system-config-printer-applet
nm-applet --sm-disable
/usr/lib/policykit-1-gnome/polkit-gnome-authenticatio
bluetooth-applet
/usr/lib/gnome-applets/mixer_applet2 --oaf-activate-i
/usr/lib/gnome-disk-utility/gdu-notification-daemon
/usr/bin/VBoxClient --clipboard
/usr/lib/evolution/2.30/evolution-alarm-notify
/usr/bin/VBoxClient --display
update-notifier
/usr/bin/VBoxClient --seamless
gnome-screensaver
/usr/lib/gvfs/gvfsd-trash --spawner :1.2 /org/gtk/gvf
/usr/lib/gvfs/gvfsd-burn --spawner :1.2 /org/gtk/gvfs
/usr/lib/gvfs/gvfsd-metadata
```

BAG OF WORDS

- Using the 100 process names, one can create a "bag-of-words" model of the data.

- "Bag-of-words" algorithm creates a list of unique words and word counts of each word.

```
100 xsessionmanager
100 usrlibgvfsgvfsgduvolumemonitor
100 usrlibgvfsgvfsdmetadata
100 usrlibgvfsgvfsd
100 usrlibgvfsgvfsafcvolumemonitor
100 usrlibgnomesettingsdaemongnomesettingsdaemon
100 usrlibgnomediskutilitygdunotificationdaemon
100 usrbinvboxclientseamless
100 usrbinvboxclientdisplay
100 usrbinvboxclientclipboard
100 usrbinsshagentusrbindbuslaunchexitwithsessionusrbinseahorseagentexecutexsessionmanager
100 usrbinseahorseagentexecutexsessionmanager
100 usrbinmetacity
100 usrbingnomekeyringdaemondaemonizelogin
100 usrbindbuslaunchexitwithsessionusrbinseahorseagentexecutexsessionmanager
100 updatenotifier
100 shoutputprocessnamesbash
100 shmainbash
100 pythonusrbinsystemconfigprinterapplet
100 psx
100 nmappletsmdisable
```

USING SEPARATE SNAPSHOT OF SYSTEM

- In order to see what processes are happening now on the system, create a new snapshot of the system (ps -x), and compare the bag-of-words of the new snapshot and the other "normal" processes. Use output of linux command diff to see what is different.

- Can also tell what processes are not running now that should be running.

RESULTS (PART 1)

```
17d16
< shsimple_AI.bash
```

In this system, the only difference is simple_AI.bash was not run on the 100 other snapshots of the system.

LINUX PROCESSES CONTINUED

- What information might we gather by running a k-means Algorithm on the processes themselves?
  - 1) What do system processes vs occasional processes look like?
  - 2) Normal operations vs Strange operations that can be flagged by the system.

CODE DESCRIPTION FOR SNAPSHOTS

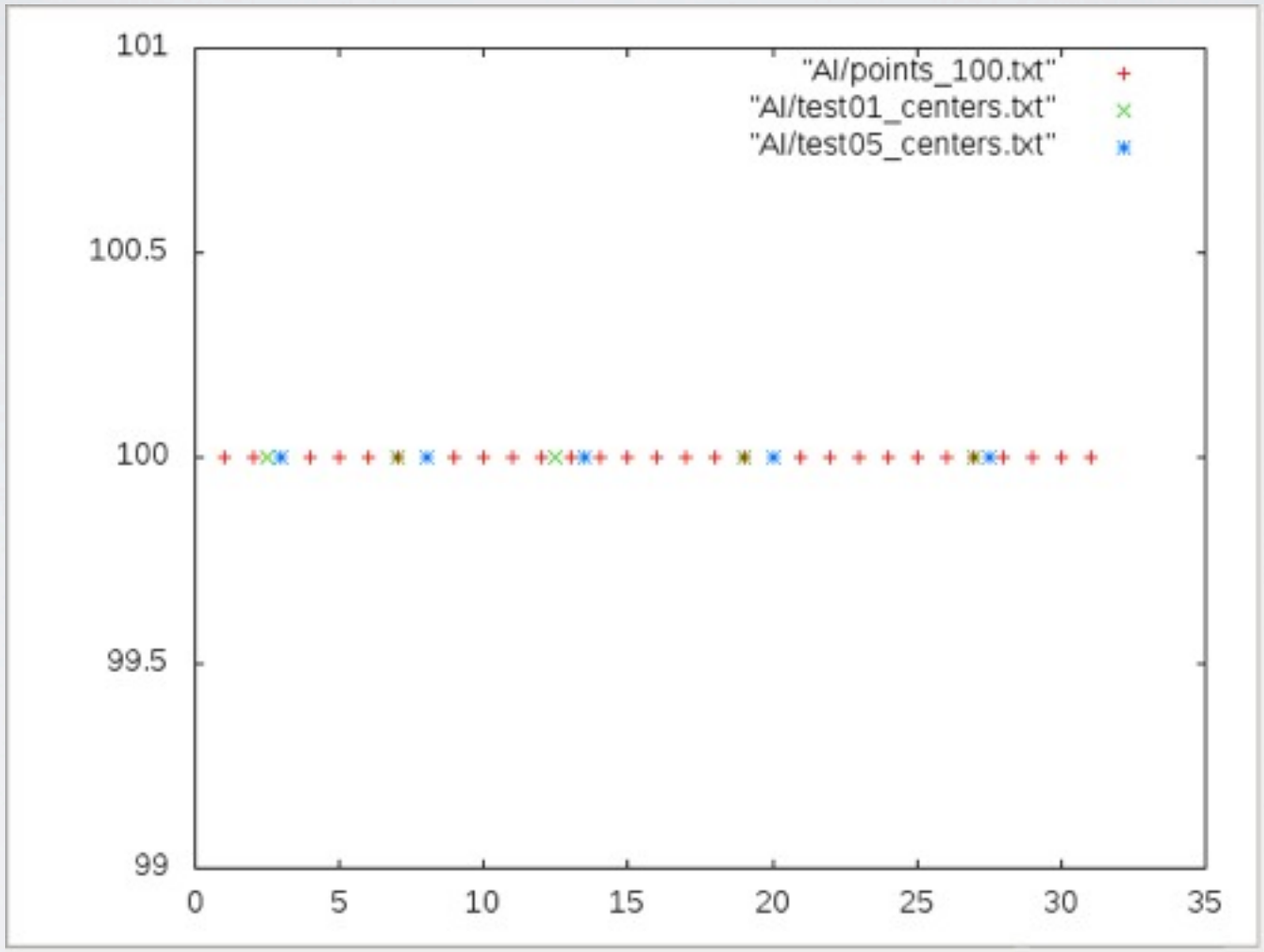
- Preprocess data by using Main.bash to generate the 100 process snapshots in /documents folder.

- Main.bash calls bag_of_words.bash which creates a bag of words from the 100 documents and puts the results in /Documents_BOW

- Concatenate all Documents_BOW and run the bag_of_words.bash again on the set of data. This will create a master document MASTER_BOW.txt in /MASTER_BOW

- Main.bash calls k_means.bash: Sets each process as the x value and the occurrence of each process in the 100 snapshots as the y value. Run the k_means executable on the data. Output goes to AI/output.png

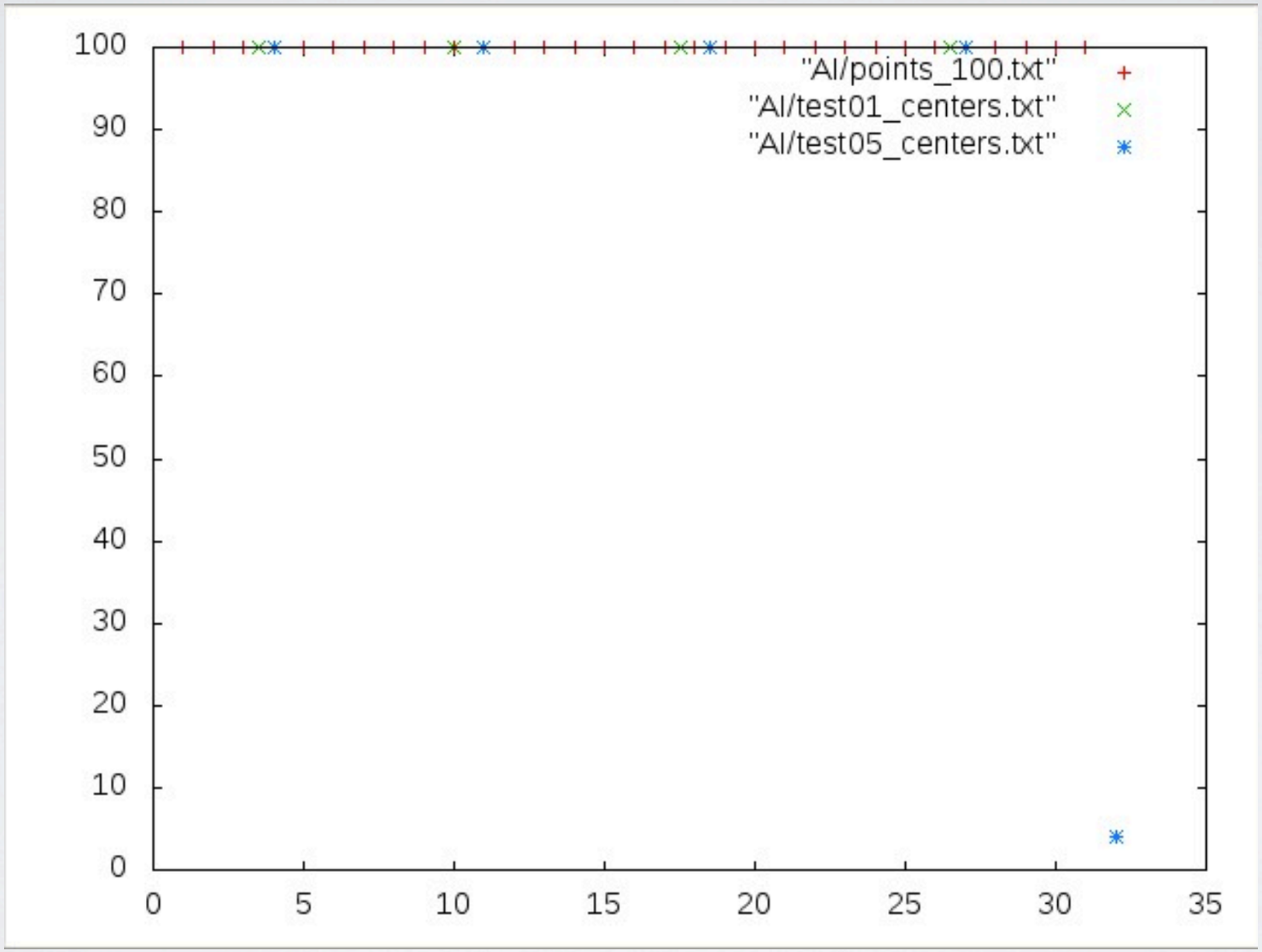
RESULTS FROM NORMAL OPERATION (ONLY SYSTEM PROCESSES)



All operations have 100 instances for all 100 snapshots.

All are system-necessary processes.

OCCASIONAL PROCESS RESULT



Few instance program on the bottom, it has its own category separate from the system-necessary processes. Has its own centroid. Might need to be flagged if system is server.

FINAL PROJECT: DOCUMENT WORD CLASSIFICATION

- Given a set of books by authors, what information can be categorized by word occurrences?

- Note: The bag of words now gets rid of capitalizations, punctuations, and special characters not associated with the words to make for more accurate data.

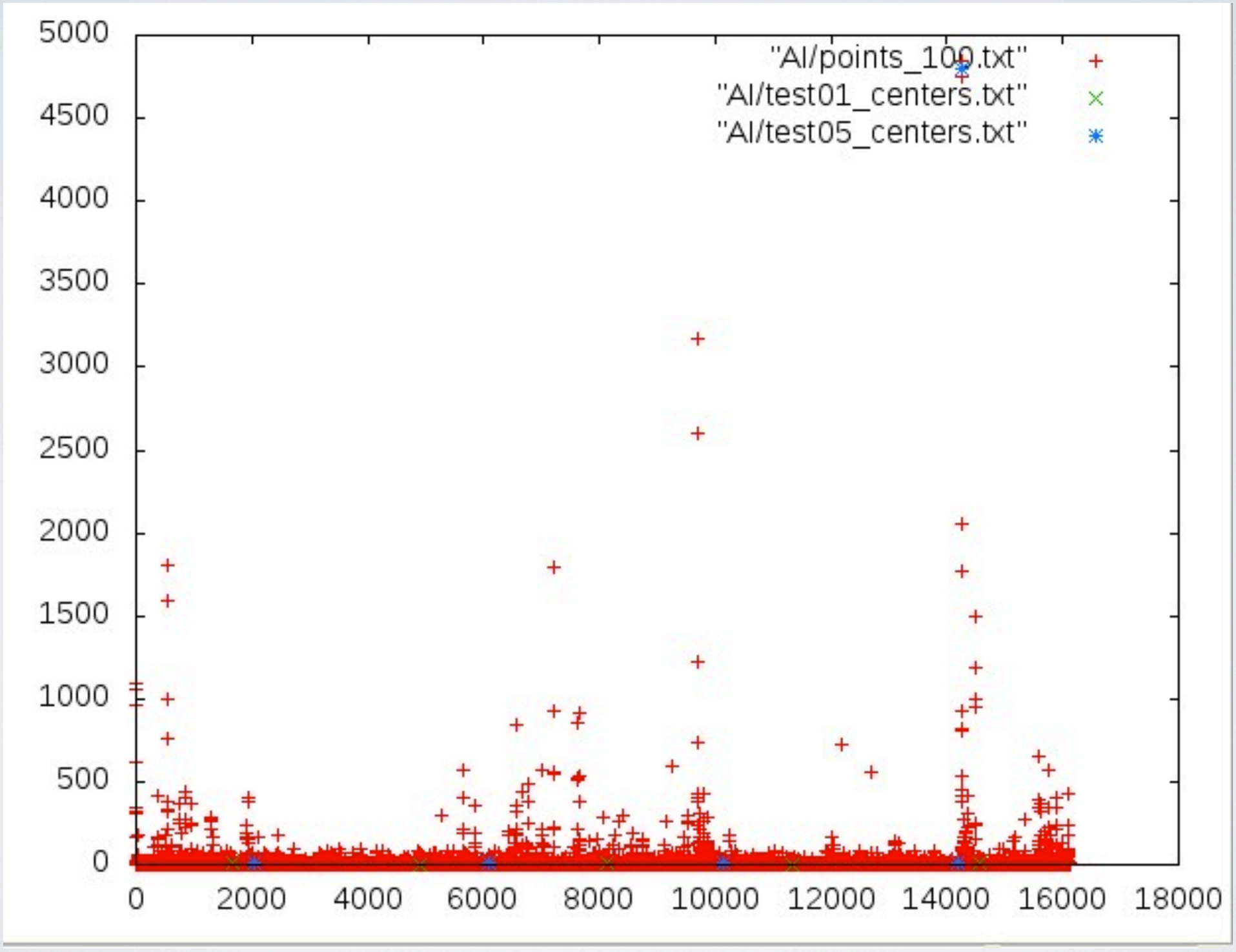
CODE DESCRIPTION FOR DOCUMENT WORD CLASSIFICATION

- Main.bash calls bag_of_words.bash which creates a bag of words from the books in /documents folder and puts the results in /Documents_BOW

- Concatenate all Documents_BOW and run the bag_of_words.bash again on the set of data. This will create a master document MASTER_BOW.txt in /MASTER_BOW

- Main.bash calls k_means.bash: Sets each word as the x value and the occurrence of each word as the y value. Run the k_means executable on the data. Output goes to AI/output.png

FINAL RESULTS



FUTURE DIRECTIONS

- Use the k-means algorithm to classify documents by preprocessing document differentiations (using Bayesian Filtering) instead of classification using word/word occurrences.

- Flagging system based on normal linux operations.

- Use data and bag of words on other AI algorithms such as LSA and PLSA.

WHAT I LEARNED

- Learned to use awk, bash, and C++ to format, process, and create visual data in the Linux operating system.

- Creating bash scripts that integrate other processes and commands to look into the system as it is running.

- Additional information on the k-means algorithm and where it is applicable.

SOURCES

- Pictures and description of k-means: http://en.wikipedia.org/wiki/K-means_clustering
- k-means C++ implementation and information: http://people.sc.fsu.edu/~jburkardt/cpp_src/kmeans/kmeans.html
- Bag of words description applied to the PLSA: http://people.csail.mit.edu/fergus/iccv2005/bagwords.html

QUESTIONS?