Description report of Azures' VM workloads traces.

Nahed Hassan Zuhier Jarrar, 0164172.

Hala Abd-Alrahman Ahmad Al-Azzam, 0178404.

Alaa Khaled Mohammad Al-Sabateen, 0172461.

Note:

Any compressed dataset can be read by pandas with any decompressing, pandas will automatically detect if the file is compressed or not and deal with accordingly.

Note:

Even though that subscription_id, deployment_id and VM_id are hashed, they are still unique and connected between tables. So, we can use the hashed ids as if they were not hashed at all.

Note:

The source code for each plot is published on the following github repository:

https://github.com/sabateen2github/PythonLab

Python lab.

Fall, 2020.

University of Jordan, CPE department.

Contents

Role of each team member.

Alaa Khaled Mohammad Al-Sabteen	Rest of the Plots, designed template of the report, assigned tasks to each member, and had put a fixed template for plots code so we have a common understanding in case some errors happened.
Hala Abd-Alrahman Ahmad Al-Azzam	Second 4 Plots code, designed template of the report and helped in debugging code.
Nahed Hassan Zuhier Jarrar	First 4 Plots code, designed template of the report and helped in debuggin code.

Introduction

This is a simple report desecribing a sanatized subset of Azure VMs workload traces collected during the year of 2020. These traces are published for research and are used by Azure (bigger dataset than what was published) to make better utillization of its' datacenters by depending on machine learning predictions.

😐(short introduction)

Schema.csv (Main File)

This a simple table describing the structure of other files in this report by specifying their columns and the corresponding format.

filepattern	field_number	content	format
vm_virtual_core_bucket_definition.	1	bucket	STRING
vm_virtual_core_bucket_definition.	2	definition	STRING
vm_memory_bucket_definition.csv	1	bucket	STRING
vm_memory_bucket_definition.csv	2	definition	STRING
subscriptions/subscriptions.csv.gz	1	subscription id	STRING_HASH
subscriptions/subscriptions.csv.gz	2	timestamp first vm created	INTEGER
subscriptions/subscriptions.csv.gz	3	count vms created	INTEGER
deployment/deployment.csv.gz	1	deployment id	STRING_HASH
deployment/deployment.csv.gz	2	deployment size	INTEGER
vmtable/vmtable.csv.gz	1	vm id	STRING_HASH
vmtable/vmtable.csv.gz	2	subscription id	STRING_HASH
vmtable/vmtable.csv.gz	3	deployment id	STRING_HASH
vmtable/vmtable.csv.gz	4	timestamp vm created	INTEGER
vmtable/vmtable.csv.gz	5	timestamp vm deleted	INTEGER
vmtable/vmtable.csv.gz	6	max cpu	DOUBLE
vmtable/vmtable.csv.gz	7	avg cpu	DOUBLE
vmtable/vmtable.csv.gz	8	p95 max cpu	DOUBLE
vmtable/vmtable.csv.gz	9	vm category	STRING
vmtable/vmtable.csv.gz	10	vm virtual core count bucket	STRING
vmtable/vmtable.csv.gz	11	vm memory (gb) bucket	STTRING
vm_cpu_readings/vm_cpu_readings-	1	timestamp	INTEGER
vm_cpu_readings/vm_cpu_readings-	2	vm id	STRING_HASH
vm_cpu_readings/vm_cpu_readings-	3	min cpu	DOUBLE
vm_cpu_readings/vm_cpu_readings-	4	max cpu	DOUBLE
vm_cpu_readings/vm_cpu_readings-	5	avg cpu	DOUBLE

Category.txt

This file shows the percentage of each vm-category (i.e vm class) wether it is an interactive, delay insensitive or unknown. The category is of each VM is determined easily if it was a first party VM or at-least a Paas VM (Platform as a service, a vm that is managed by the cloud provider). However, third party Iaas are difficult to categorize due to the lack of information about the specific job of the VM. So, Azure is solving (or trying to solve) this problem by performing an analysis of the (average) CPU utilization time series to infer wether a certain VMs' workload is likely to be interactive (i.e interactive VMs' workload is most active during the day and sleep at night, a diurnal cycle).

The algorithim to categorize the VMs' workload depends on FFT (Fast fourier transform); the usage can look like a sinusoidal so FFT fits well and we don't need to worry about the (the missing fundamental problem).

Workloads that show periodicity at a diurnal cycle are considered "potentially interactive" and those who don't are considered "delay insensitive". The algorithim builds a time series of cpu utilization by collecting data for 3 days, any VM that does not last more than 3 days are categorized as "UNKNOWN".

CPU usage utilization was chosen as the determining resource of the category due to its; usage being a proxy of every other resource usages.

Cores.txt

This is a simple table that summarize the percentages of vCores assignments (assigned by customers) to deployed VMs. The distribution of vCores is limited to the following buckets:

- 2 vCores
- 4 vCores
- 8 vCores
- 12 vCores
- 24 vCores
- 30 vCores

😐(nothing more to say)

Cpu.txt

This file shows the CDF (cumulative distribution function, in table form) of the average and the 95^{th} – percentile of vCPU utilization.

😐(nothing more to say)

Deployment.txt

This file shows the CDF (cumulative distribution function, in table form) of the percentages of deployment sizes, a deplyment size is defined by as a set of VMs that the customer groups and manages together. The CDF does not include all deployments, it only includes deployments up to 50 VMs, this is because deplyments do not have an upper bound and there are customers who deploy more than 50 VMs by a lot. However, they only constitute less than 3% of the entire deployments.

😐(nothing more to say)

Lifetime.txt

This file describes the lifetime of VMs (how long VMs last from creation to termination) by hours. Lifetimes are divided into bucket increments of 0.1 hours up to 100 hours, this range of lifetimes constitutes nearly 92% percent of the entire VMs deployed during the period of collecting the data (traces).

😐(nothing more to say)

Memory.txt

This is a simple table that summarize the percentages of vMemory assignments (assigned by customers) to deployed VMs. The distribution of vMemory is limited to the following buckets:

- 2 GB
- 4 GB
- 8 GB
- 32 GB
- 64 GB
- 70 GB

😐(nothing more to say)

Vmtable.csv.gz

The file contains the followingcolumns in this order:

1. VM ID
2. Subscription ID
3. Deployment ID
4. Time stamp Created
5. Time stamp Deleted
6. Max CPU Utilization
7. Avg CPU Utilization
8. P95 CPU Utilization
9. VM Category (interactive and delay-insensitive or other)
10. VM Virtual Core Count (2, 4, 8, ...)
11. VM Memory GB (2, 4, 8 ...)

The VM, Subscription and Deployment IDs represent actual users, but instead of using the real names and IDs, they have been hidden by using random strings for privacy reasons. There is not much usefulness in these columns.

The difference between the time stamp created and deleted represents thelifetime. The cumulative distribution function was derived from the Avg CPU Utilization, and the P95 was derived from Max CPU Utilization

Deployments.csv.gz

Like the CPU, this file represents a CDF (A cumulative distribution function). The bucket represents the number of virtual machines a VM creator used in their deployed application, while the "value"column represents the cumulative percentage. For example, when we say that at bucket 1, we have a value of 47.021, this means that 47.021% of all Azure Applications used only one virtual machine. When we say at bucket 10 that we have 87.884, this means 87.884%of VM creators used 10 or less virtual machines. If you subtract each value in the "value" column from the one before it, you can get the PDF (probability distribution function). The new value represents the probability at acertain number of interes t(without the less part). So,if you subtract the value at bucket 9 from bucket 10 (87.884-86.553= 1.331), this means that 1.331% of applications used exactly 10 virtual machines

Subscriptions.csv.gz

This file looks from customer perspective, it contains the following columns:

- Subscription ID (Hashed).
- Timestamp of first VM created.
- Number of VMs created under this subscription.

Subscriptions are basically the accounts of customers of the cloud service provider.

vm_cpu_readings-file-45-of-195.csv.gz

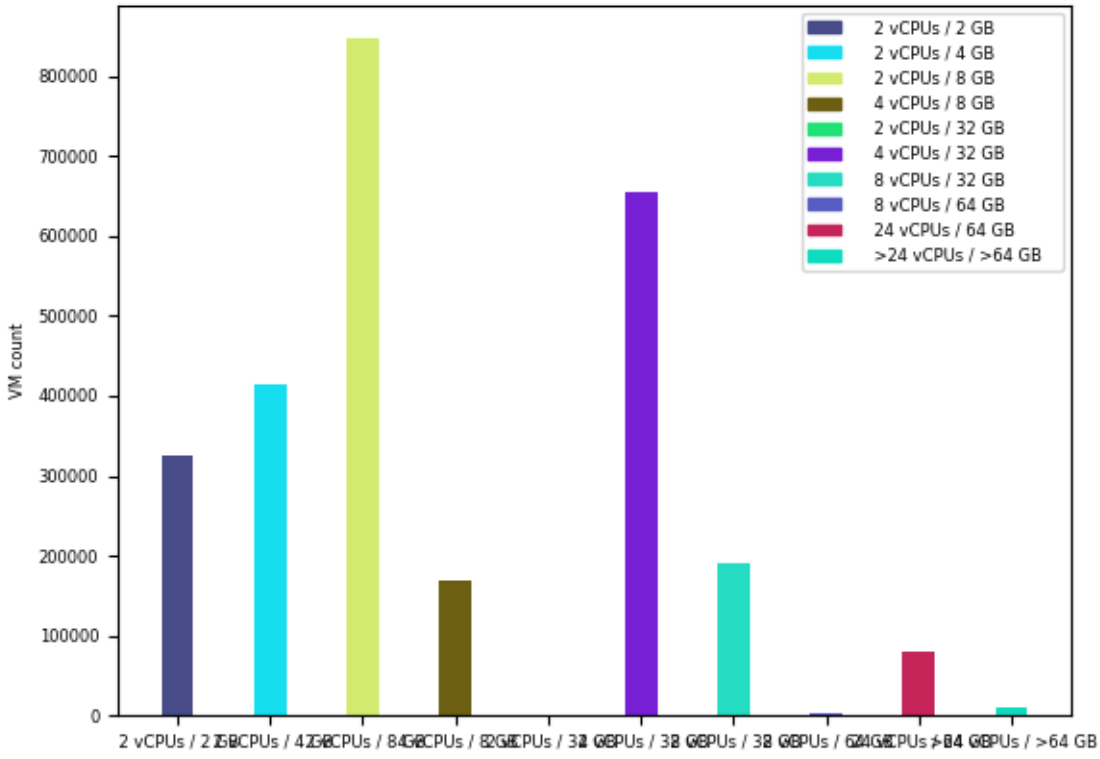
The file contains the followingcolumns in this order:

1. Time stamp
2. VM ID (Hashed)
3. Minimum CPU Usage
4. Maximum CPU Usage
5. Average CPU Usage

Plots

vMemory bucket and vCPU bucket combinations versus VM distrubutions.

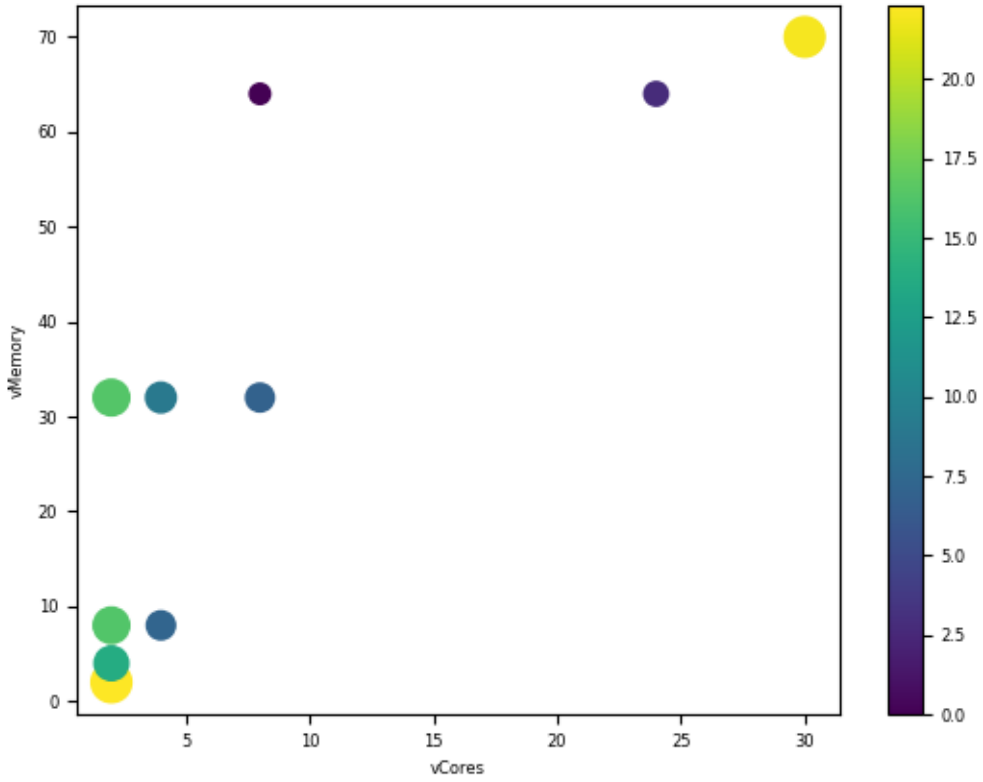
vCPU_vMemory_bucket.py



This plot shows the distribution of VMs between the different VMs classes (i.e., 2 vCores/2 GB, 2vCores/8GB, 12 vCore/64 GB …. Etc.). As we can see, the 2 vCore/8GB and the 4 vCore/8GB are the most popular, while the 2 vCore/32Gb and 8vCore/64GB are the least popular.

vCore bucket and vMemory bucket vs average-aggregated vCPU utilization.

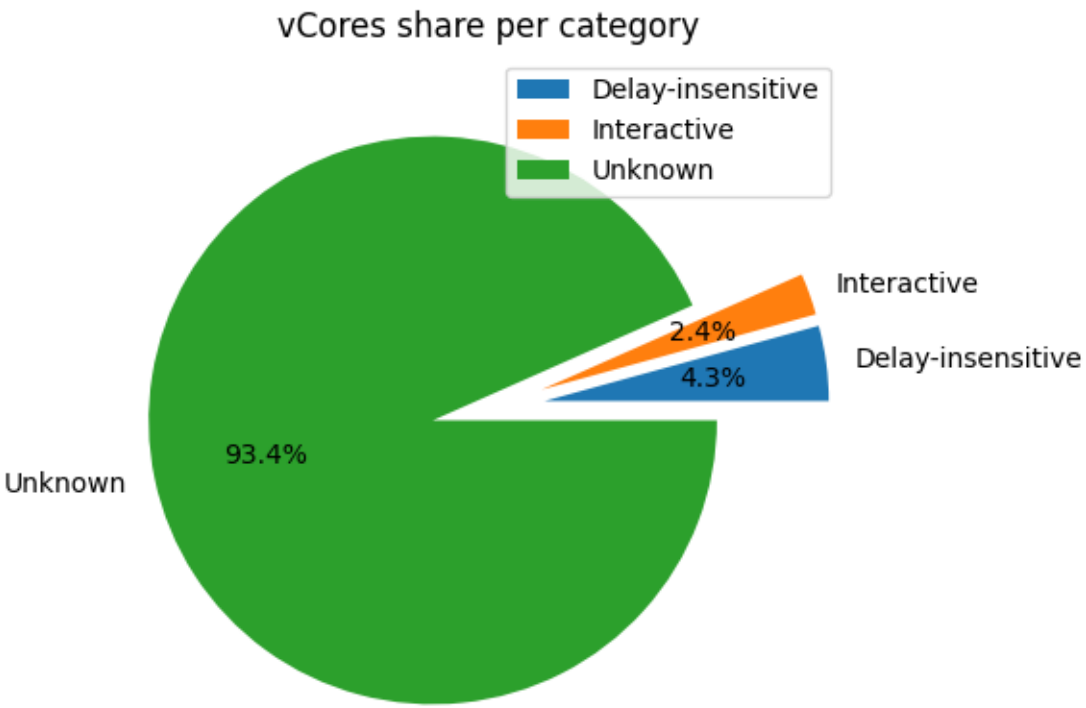
scatter_vCore_vMem_vCPU_utilization.py



This plot shows the average CPU utilizations over all the combinations of VMs specificatons (vCores and vMemory). It is a simple scatter plot that shows that CPU utilization is at its best at very big VMs or very tiny ones and the more we approach the center the less efficient the configuration will be.

Size of each Category in terms of vCores (not vCPUs).

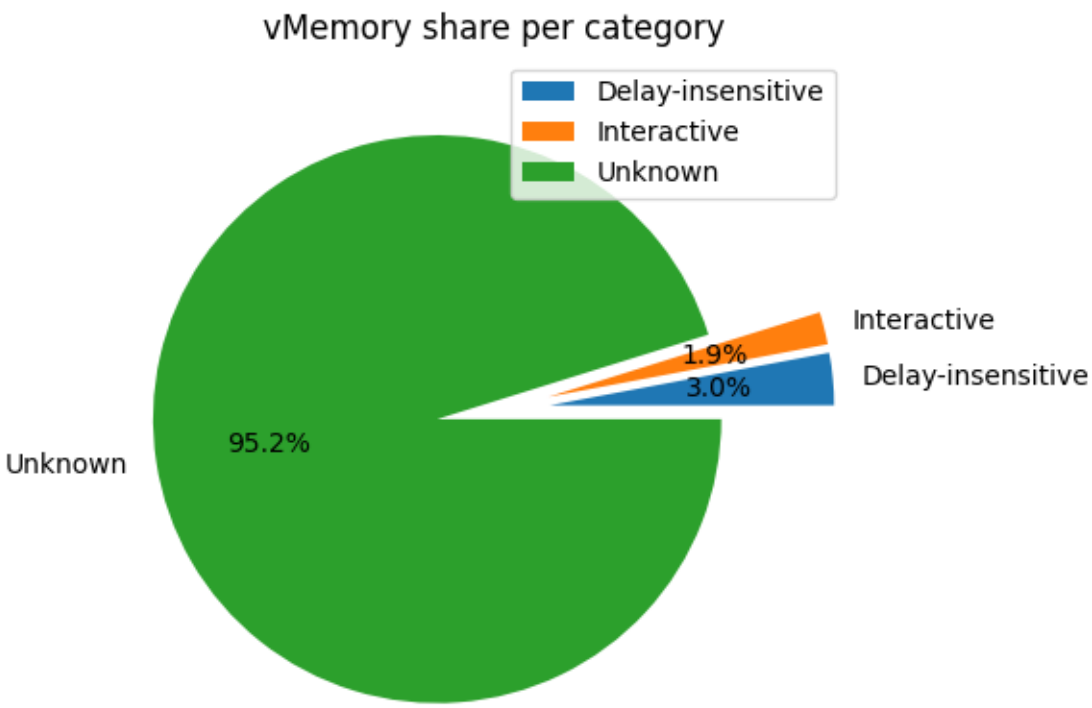
category_vCPUs.py



This plot shows the share of assigned vCores for each VM class. As we can see, the unknown class (VMs that do not 6.last more than 3 days) are most dominant in the cloud environment.

Size of each Category in terms of vMemory.

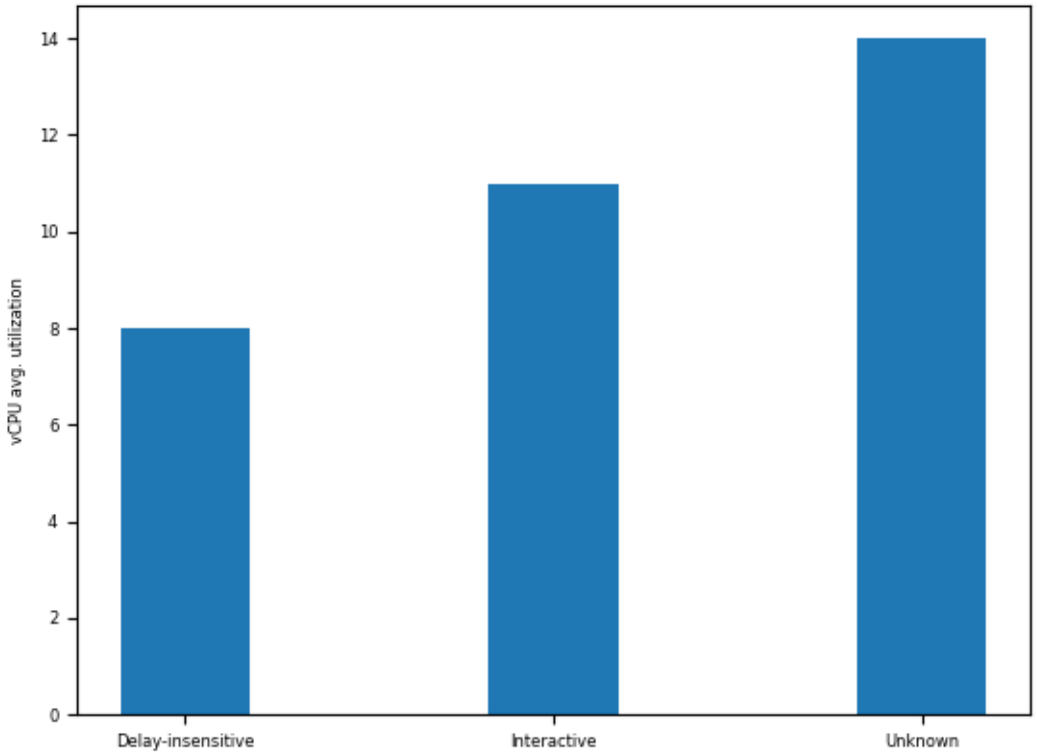
category_vMem.py



This plot shows the share of assigned vMemory for each VM class. As we can see, the unknown class (VMs that do not 7.last more than 3 days) are, again, the most dominant in the cloud environment.

Average-aggregated vCPU utilization for each category.

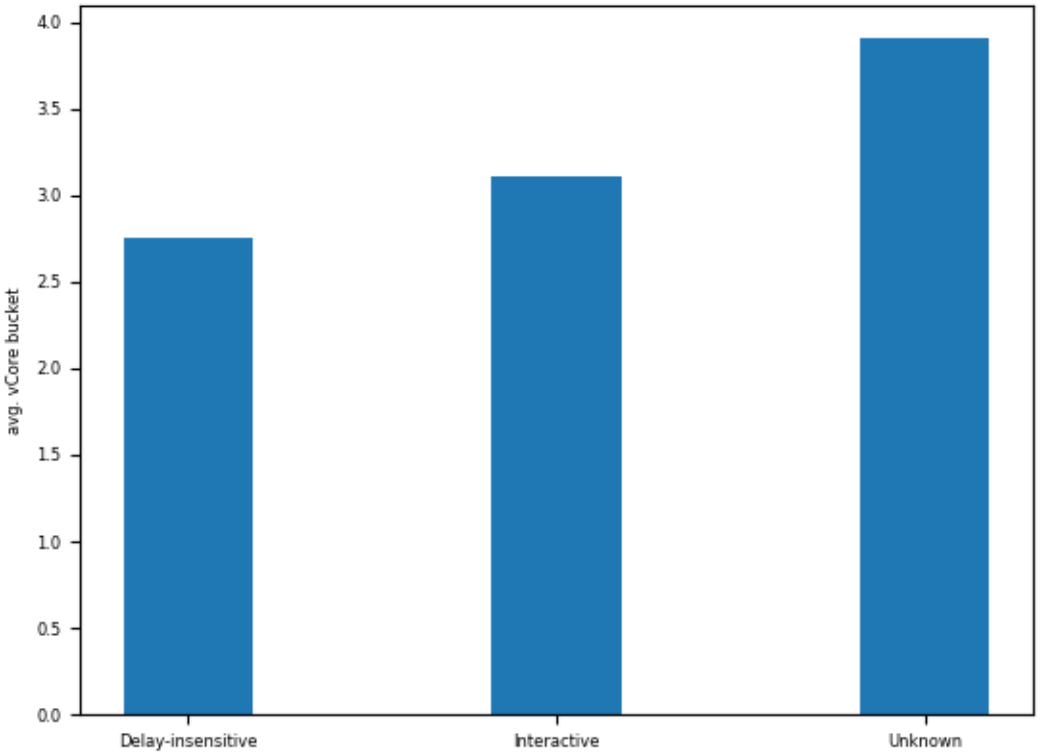
vCPU_utilization_category.py



This plot shows the CPU utilization of the three main classes of VMs. Since most VMs are from third party customers we cannot determine their exact class, so they are marked as unknown. In addition, third party VMs are the most dominant in the cloud environment and thus it is logical to see that the third party VMs have the highest CPU utilization, they service the internet!!!

Average vCore bucket for each category.

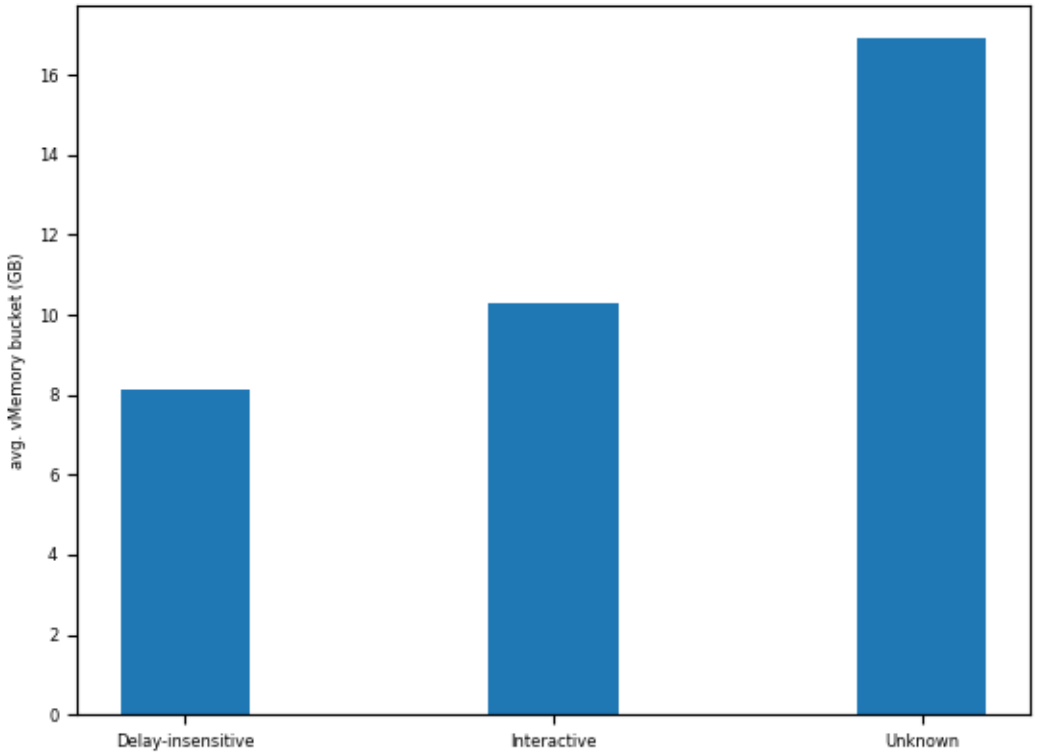
vCore_bucket_category.py



This plot shows the average vCore bucket for each VM class. Delay-insensetive and interactive are first-paty VMs tha mainly do maintenance, logging, monitoring … etc. They don't need large VMs with high buckets of vCores and vMemory.

Average vMemory bucket for each category.

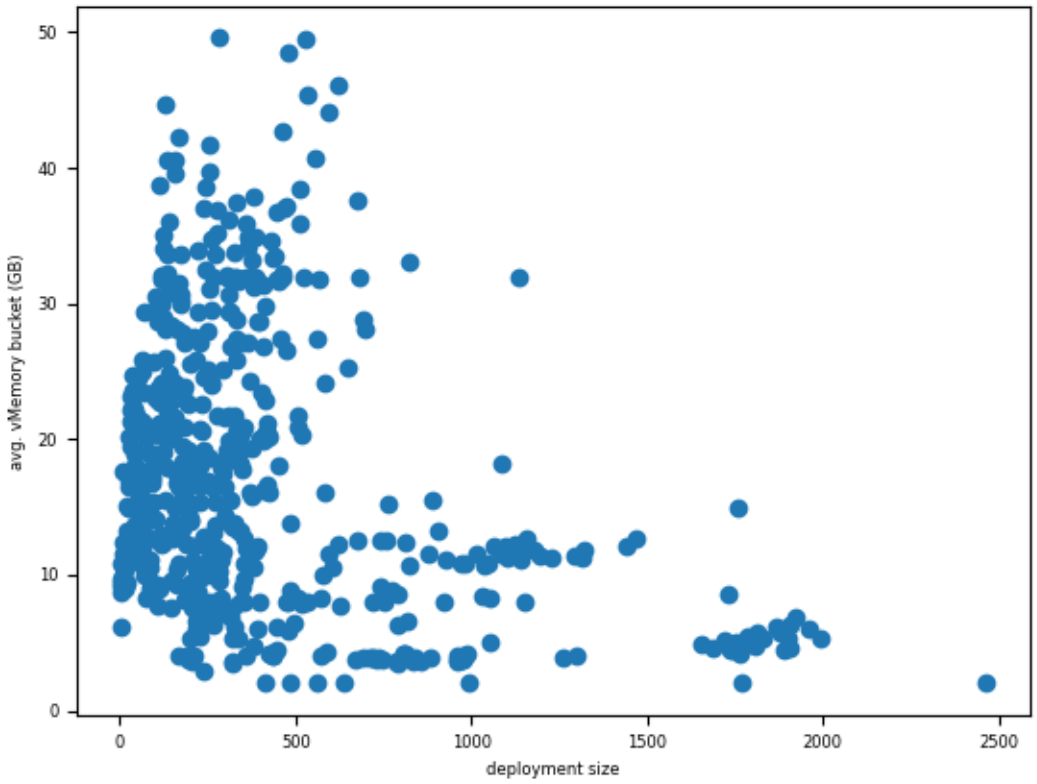
category_vMem.py



This plot shows the average vCore bucket for each VM class. Delay-insensetive and interactive are first-paty VMs tha mainly do maintenance, logging, monitoring … etc. They do not need large VMs with high buckets of vCores and vMemory.

Size of deployment versus the mean of vMemory buckets.

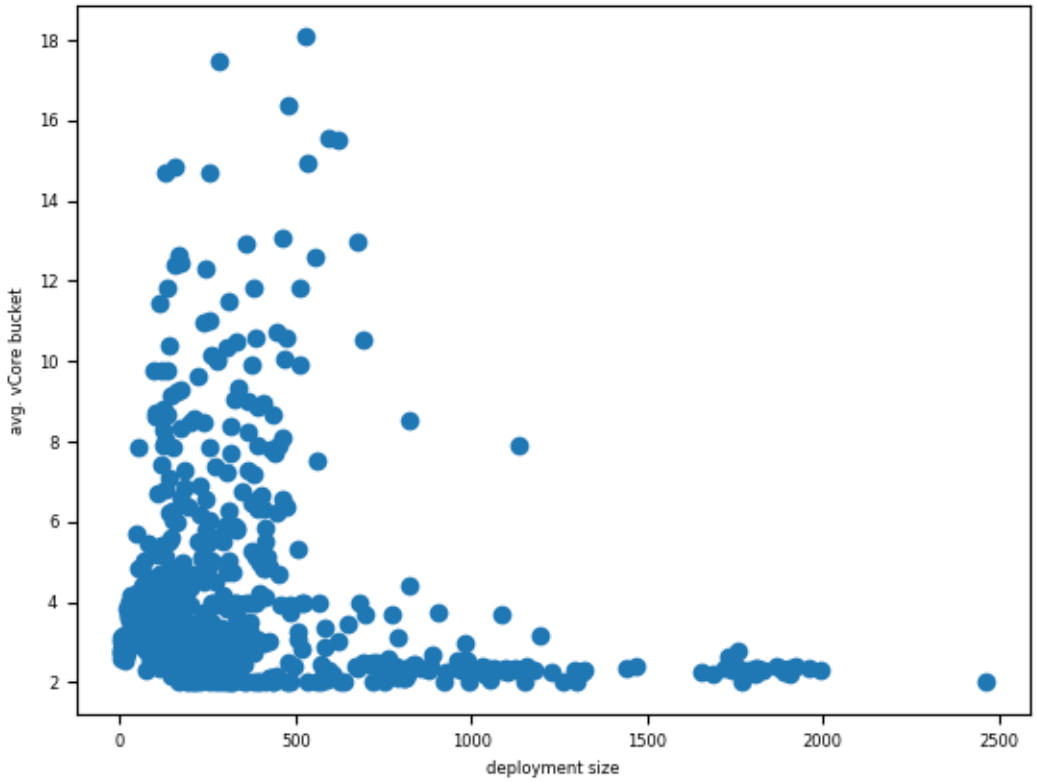
size_of_deployment_mean_of_vMem.py



This plot depicts the deployment size and its' corresponding average vMemory bucket. The average bucket value is calculated as the weighted mean of the bucket values of the deployments that fall into a certain deployment size. As we can see, the higher the deployment size the lower the average vMemory allocated for each VM. This conveys the two approaches for clustering VMs for workloads, the old approach which is the large VM with high vCores count and high vMemory and the new approach of low specs VM but in many numbers. We can also notice that most deployment sizes are no more than 1000 VMs, this is because not everyone is an international corporation!

Size of deployment versus the mean of vCore buckets (LinePlot).

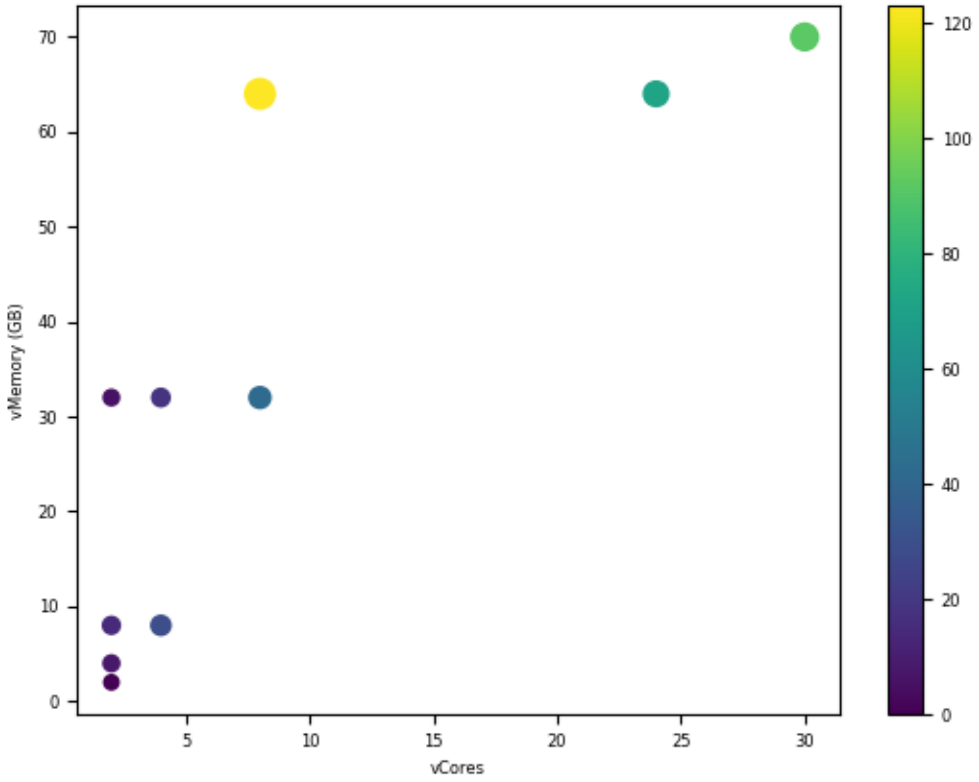
size_of_deployment_mean_of_vCore_bucket.py



This plot depicts the deployment size and its' corresponding average vCore bucket. The average bucket value is calculated as the weighted mean of the bucket values of the deployments that fall into a certain deployment size. As we can see, the higher the deployment size the lower the average vCore allocated for each VM. This conveys the two approaches for clustering VMs for workloads, the old approach which is the large VM with high vCores count and high vMemory and the new approach of low specs VM but in many numbers. We can also notice that most deployment sizes are no more than 1000 VMs, this is because not everyone is an international corporation!

Average Size of deployment versus vMemory buckets versus vCore buckets.

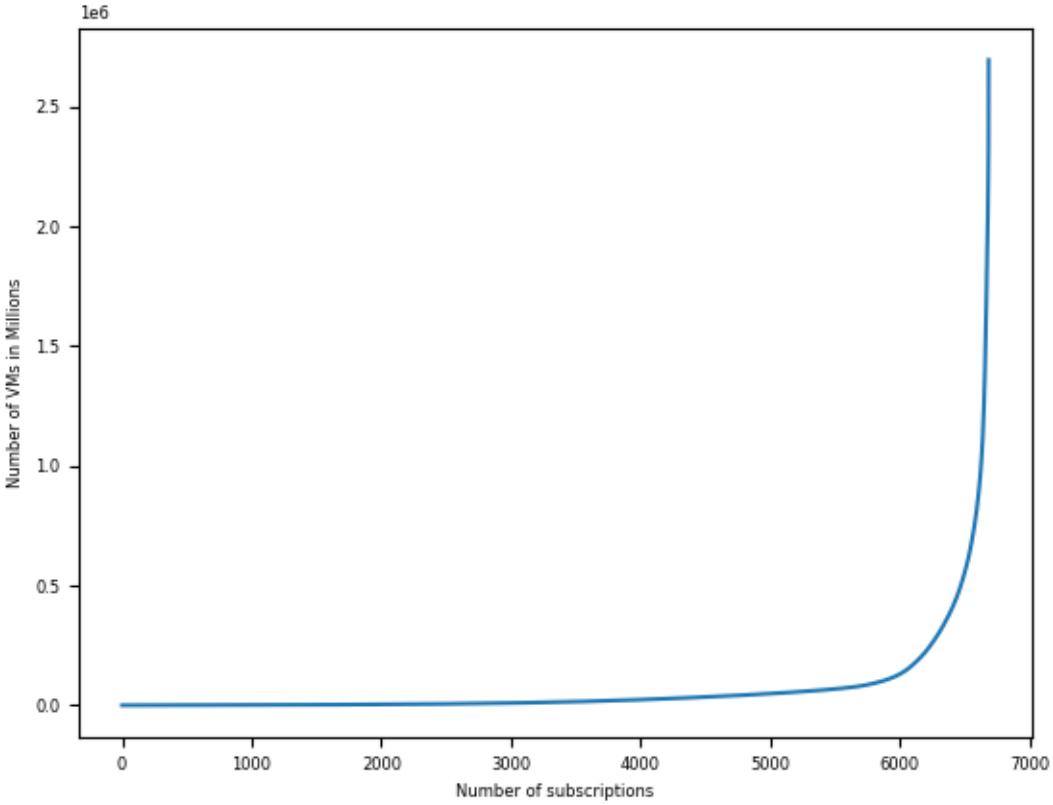
avg_size_deployment_vMem_vCore.py



This plot depicts the average deployment size based on the VM class (i.e., vCPU and vMemory). This plot may lead us at first time to the conclusion that vMemory plays more important role than vCores in deciding the deployment sizes. However, this is not true, applications are diverse and many of them require high deployemnt sizes but not of the same VM specs (i.e, vMemory and vCores). For example, deep learning needs a lot of vMemory but not much of vCores, the deplyments of the yellow circle can be for some deep learning workloads. In contrast, batch processing of raw data for classifications and clustering require high number of vCores but not as much for vMemory.

Percentages of VMs versus percentages of subscribers (Line plot).

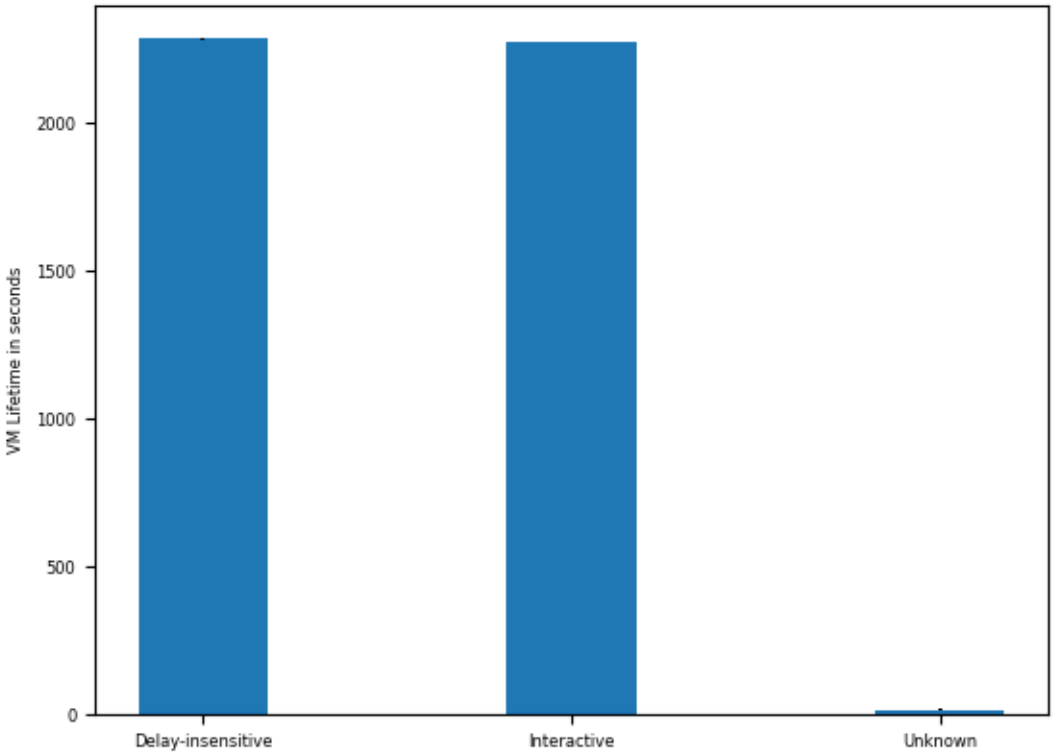
vms_subscribers.py



This plot depicts the number of subscribers and the corresponding created VMs at the datacenter. From the plot we can conclude that the majority of VMs created belongs to minority of subscribtions (customers). This, ofcourse, correlates with reality of capitalisim that there are multi billion dollars companies that have most of the market and the smaller the company is the smaller its market size and the smaller its' running VMs to serve their customers or their needs.

Lifetimes of VMs (for each category).

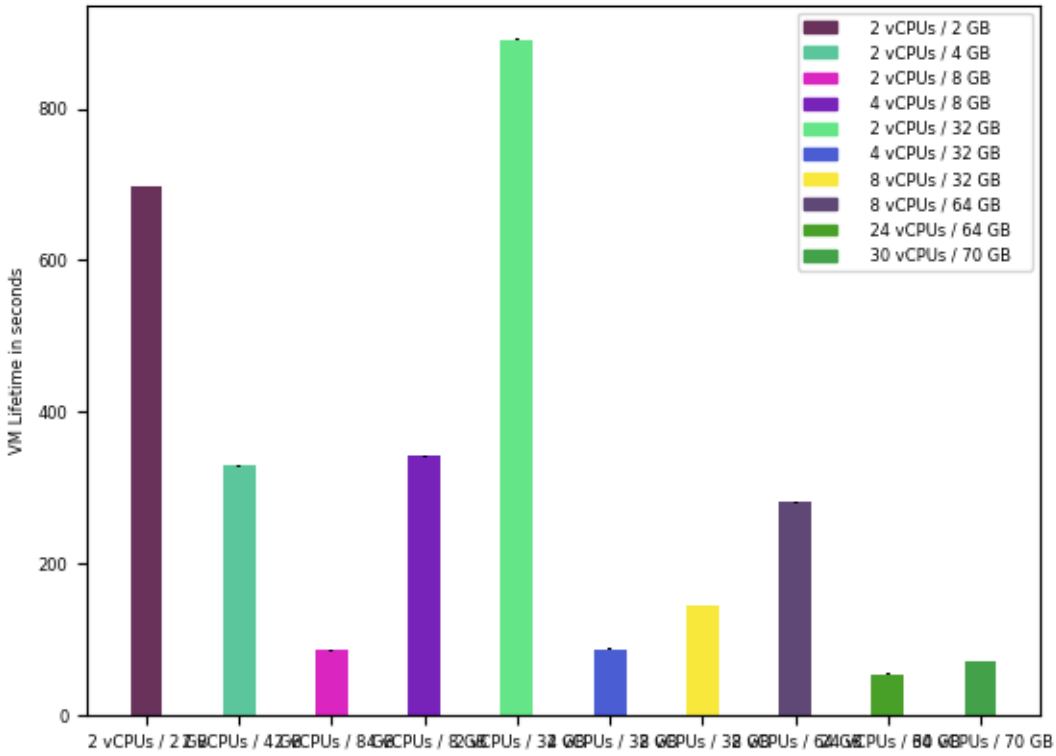
category_lifetime.py



This plot depicts the average lifetime of each VM category. As we can see, the Unknown category (third party VMs) do not live very long at all. This is because most custemers rely on PaaS VMs which are dynamically managed and scheduled by the cloud provider based on demand, so the scheduler deletes them as soon as they are not needed, to increase oversubscription ratio.

Lifetimes of VMs (for each VM spec).

spec_lifetime.py



This plot depicts the average lifetime of each deployed VM spec. As we can see, VMs with highest lifetimes are the following:

- 2 vCores/2GB, these are most probably mainly clusters of mini webservers that are always busy serving requests.
- 2vCores/32GB, these are most probably VMs deployed to do some heavy background processing or machine learning training that spend too much time before finishing their job.

Also, we can see that the VMs with the lowest lifetime are the following:

- 24 vCore/64 GB
- 30 vCore/70 GB

These are most probably background tasks that finish fast due to the high concurrency of the nature of the task and providing the suitable hardware for it.