# WEEK1 任务

## 1. Kubernetes 基础概念：

kubernetes，简称 K8s，是用 8 代替 8 个字符“ubernete”而成的缩写。是一个开源的，用于管理云平台中多个主机上的容器化的应用，Kubernetes 的目标是让部署容器化的应用简单并且高效（powerful）,Kubernetes 提供了应用部署，规划，更新，维护的一种机制。

传统的应用部署方式是通过插件或脚本来安装应用。这样做的缺点是应用的运行、配置、管理、所有生存周期将与当前操作系统绑定，这样做并不利于应用的升级更新/回滚等操作，当然也可以通过创建虚拟机的方式来实现某些功能，但是虚拟机非常重，并不利于可移植性。

新的方式是通过部署容器方式实现，每个容器之间互相隔离，每个容器有自己的文件系统 ，容器之间进程不会相互影响，能区分计算资源。相对于虚拟机，容器能快速部署，由于容器与底层设施、机器文件系统解耦的，所以它能在不同云、不同版本操作系统间进行迁移。

容器占用资源少、部署快，每个应用可以被打包成一个容器镜像，每个应用与容器间成一对一关系也使容器有更大优势，使用容器可以在 build 或 release 的阶段，为应用创建容器镜像，因为每个应用不需要与其余的应用堆栈组合，也不依赖于生产环境基础结构，这使得从研发到测试、生产能提供一致环境。类似地，容器比虚拟机轻量、更“透明”，这更便于监控和管理。

容器优势总结：

- **快速创建/部署应用：**与 VM 虚拟机相比，容器镜像的创建更加容易。
- **持续开发、集成和部署：**提供可靠且频繁的容器镜像构建/部署，并使用快速和简单的回滚(由于镜像不可变性)。
- **开发和运行相分离：**在 build 或者 release 阶段创建容器镜像，使得应用和基础设施解耦。
- **开发，测试和生产环境一致性：**在本地或外网（生产环境）运行的一致性。
- **云平台或其他操作系统：**可以在 Ubuntu、RHEL、 CoreOS、on-prem、Google Container Engine 或其它任何环境中运行。
- **Loosely coupled，分布式，弹性，微服务化：**应用程序分为更小的、独立的部件，可以动态部署和管理。
- **资源隔离**
- **资源利用：**更高效

使用 Kubernetes，可以在物理或虚拟机的 Kubernetes 集群上运行容器化应用，Kubernetes 能提供一个以“容器为中心的基础架构”，满足在生产环境中运行应用的一些常见需求，如：

- 多个进程（作为容器运行）协同工作。（Pod）
- 存储系统挂载
- Distributing secrets
- 应用健康检测

- 应用实例的复制
- Pod 自动伸缩/扩展
- Naming and discovering
- 负载均衡
- 滚动更新
- 资源监控
- 日志访问
- 调试应用程序
- 提供认证和授权

## 2.云、微服务相关概念：

微服务是一种开发软件的架构和组织方法，其中软件由通过明确定义的 API 进行通信的小型独立服务组成。这些服务由各个小型独立团队负责。微服务架构使应用程序更易于扩展和更快地开发，从而加速创新并缩短新功能的上市时间。

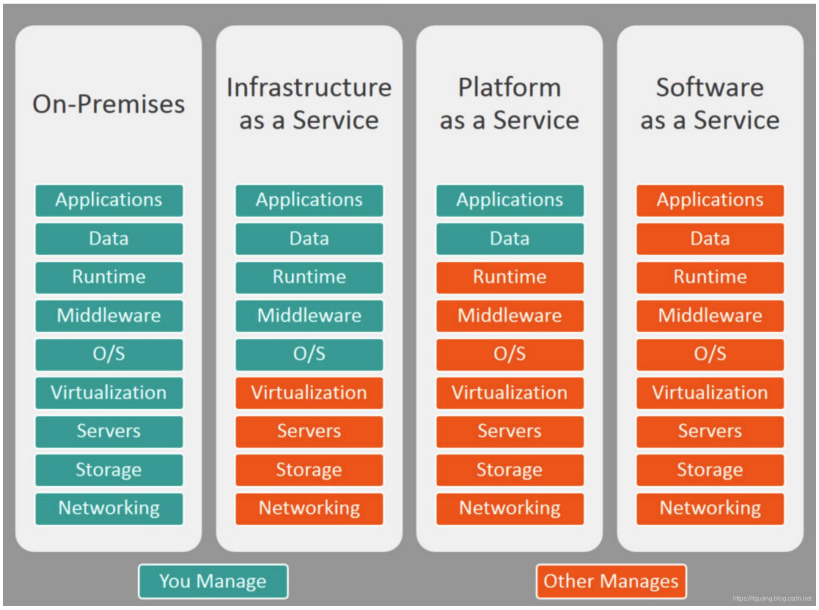
**整体式架构与微服务架构：**

通过整体式架构，所有进程紧密耦合，并可作为单项服务运行。这意味着，如果应用程序的一个进程遇到需求峰值，则必须扩展整个架构。随着代码库的增长，添加或改进整体式应用程序的功能变得更加复杂。这种复杂性限制了试验的可行性，并使实施新概念变得困难。整体式架构增加了应用程序可用性的风险，因为许多依赖且紧密耦合的进程会扩大单个进程故障的影响。、

使用微服务架构，将应用程序构建为独立的组件，并将每个应用程序进程作为一项服务运行。这些服务使用轻量级 API 通过明确定义的接口进行通信。这些服务是围绕业务功能构建的，每项服务执行一项功能。由于它们是独立运行的，因此可以针对各项服务进行更新、部署和扩展，以满足对应用程序特定功能的需求。

最流行的云计算有三种模式，它们分别是：

- **基础设施即服务（Infrastructure as a Service，IaaS）；**
- **平台即服务（Platform as a Service，PaaS）；**
- **软件即服务（Software as a Service，SaaS）。**




每个模型中的关键项就是控制：由谁来负责维护基础设施，以及构建应用程序的技术选择是什么？

在 IaaS 模型中，云供应商提供基础设施（通常是服务器，存储，宽带），但你需要选择技术并构建最终解决方案；而在 SaaS 模型中，你就是供应商所提供的服务（比如微信，淘宝等）的被动消费者，无法对技术进行选择，同时也没有任何责任来维护应用程序的基础设施。

## 3.因果推断笔记：

**相关性与因果性：**

因果性建立在相关性之上，相关性不能代表因果性。

因果推断的核心思想在于反事实推理 counterfactual reasoning，即在我们观测到 X 和 Y 的情况下，推理如果当时没有做 X，Y'是什么。

因果推断的目的是要判断因果性，即计算因果效应（有无 X 的情况下 Y 值的变化量）。在进行反事实推理后，可得出因果效应 e = |Y - Y'|，进而判断因果性。

实际上，对于一个对象，我们永远只能观察到 Y 和 Y'的其中一个，因果推断所做的就是从已有数据中估计因果效应，所以我认为因果推断的本质是，对因果效应的估计。

**随机实验 Randomization：**

**A/B Test：**

以推荐算法为例，判断推荐算法是否有效，ABTest 通过将用户随机分为两组，分别应用不同的算法，通过判断两组用户点击率的差异来估计因果效应。通过随机分组，排除了混淆变量的影响。

A/B Test 实际上是判断因果性的很有效的方法，但有时候成本过高无法采用，如这里的推荐算法——

- 可能新的推荐算法太差导致用户流失
- 如果有很多新的算法要测试，A/B Test 效率较低

**Multi-armed bandits：**

针对上述问题，另一种随机实验方法是强化学习中的多臂老虎机，实际上是对 explore 和 exploit 的平衡。

• explore，随机选择一个动作，在上面的问题中是随机选择一个算法
• exploit，选择收益最高的动作，在上面的问题中是选择当前效果最好的算法
通过某种规则（e-greedy 等）重复上述过程，优点是可以同时测试多种算法，并且每个用户都能使用到最好的算法，减少流失可能性。缺点是效果难以评估，也很难让用户按照我们的想法行动。

**自然实验 Natural Experiments：**

理想的实验需要：随机分配(分组)、人为干预(施加不同的 treatment)、结果比较。

自然实验实际上是一种观察性研究，满足上述三个条件中的两个，是指不加干预地、实验对象“自然”地分为若干组，对实验对象的结果进行观察比较。

显然自然实验法的关键在于，实验对象是否能“自然”/随机地分组。比如，将是否民主将国家分为两组，探究制度与国家对外战争的关系。但是在这里，是否民主不是随机的分给各个国家，所以无法满足自然实验所需的随机分配原则。

**断点回归 Regression discontinuity：**

断点回归是自然实验中的一种观察方法，简单理解就是在回归过程中，观察在临界点处是否出现断层/断点。

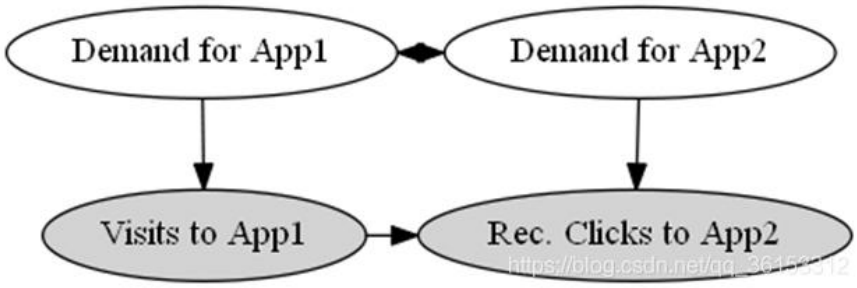
举一个简单的例子，假设现在有一个产品，收集 500 个金币后就可以得到一个勋章，现在要判断有无勋章对用户在线时长的影响。

断点回归法观察金币在 500 附近的用户，如 497 到 502，观察【接近 500 但小于 500(无勋章)】与【接近 500 但大于 500(有勋章)】的用户在线时长是否有显著区别，若有，说明有勋章很可能会增加用户的在线时长。

**工具变量 Instrumental Variables：**

对于要判断因果关系的两个变量间，如果存在其他混淆变量，在计量经济学中采用工具变量的方法解决。

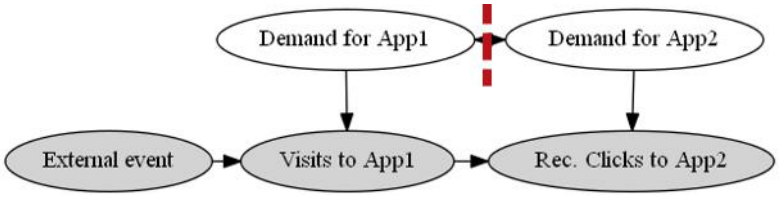
以下述关系为例，要判断对 APP1 的访问，是否会导致对 APP2 的点击。实际上由于 APP1 和 APP2 之间的需求关系，误差项与解释变量相关，即计量经济学中的内生性。




引入工具变量的目的是为了让误差项与解释变量不相关。具体地，通过找到一个变量，满足与解释变量相关且与误差项无关，那在引入这一变量之后，解释变量变化的部分就与误差项无关。

同样是上面的例子，假设某一天有个活动，下载 APP1 的人有奖励，这个活动与解释变量相关，但不会影响到 APP2 的需求，那根据多出来的 APP1 访问量与多出来的 APP2 点击率就不再受到需求关系的影响，就可以判断对 APP1 的访问，是否

会导致对 APP2 的点击。




Conditioning：

**分层 Stratification：**

分层的核心思想是控制条件变量，一般步骤如下：

- 尽可能完整的绘制出变量之间的因果图
- 选择影响要判断因果性的变量的条件变量
- 对用户进行分层/分组，满足组内的用户条件变量取值一致（上层的变量将全部不需要再考虑，类似贝叶斯网络中的 d 分隔）
- 比较两组用户的输出，计算因果效应

这种方式有点类似要找到相似的用户，当条件变量很多的时候，这种方法很难实现，很难找到很多条件变量都相同的用户，即使找到也会使得分组偏小。

**倾向得分匹配 Propensity score matching：**

当条件变量很多的时候，可以考虑使用倾向得分匹配。

以推荐算法为例，当条件变量很多的时候，通过逻辑回归等方法对这些变量进行训练，并计算出一个倾向得分，在这里是用户被施加新算法的概率。因此倾向得分匹配的一般步骤如下：

- 尽可能完整的绘制出变量之间的因果图
- 选择影响要判断因果性的变量的条件变量
- 对用户进行分层/分组，满足组内的用户计算得出的倾向得分接近（上层的变量将全部不需要再考虑，类似贝叶斯网络中的 d 分隔）
- 比较两组用户的输出，计算因果效应

进行因果推断，上面共总结了三类方法。如果可以的话，尽可能使用随机实验（ABtest……）；如果无法进行随机实验，则探索自然实验（断点回归……）；如果自然实验也无法找到，考虑使用基于条件的方法（倾向得分匹配……）