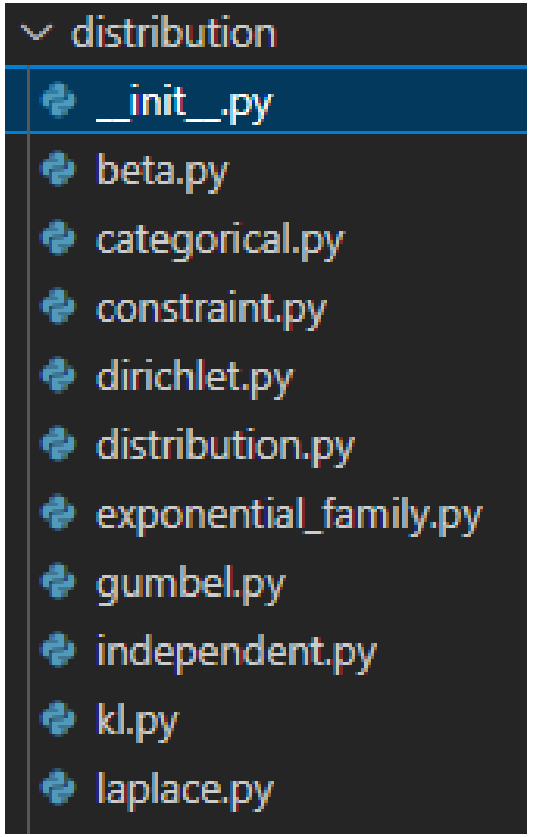
python新增API
——以概率分布为例

概率分布

- paddle.distribution中实现了丰富的概率分布方案

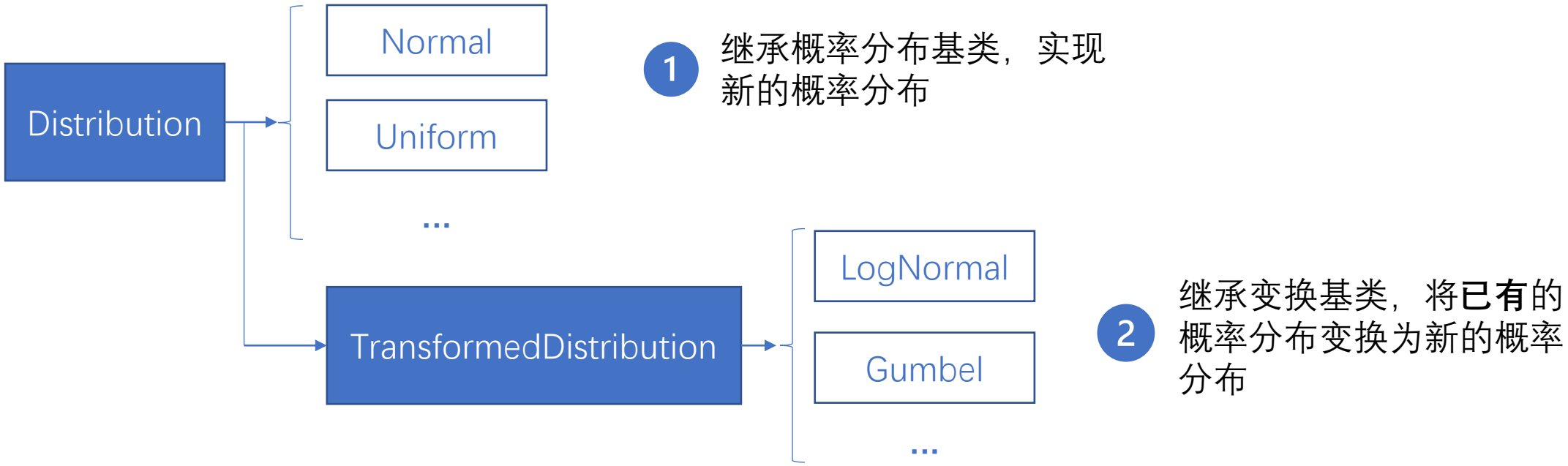
  Bernoulli, Multinomial, Normal, …
- 飞桨黑客松有概率分布相关的题目

  第三期-7/8/9/10 第四期-11/12/13
- 框架中尚有未实现的概率分布



概率分布的实现

- 主要有两种实现方式



概率分布基类

```python
class Distribution:
    # 均值
    def mean(self):
        raise NotImplementedError

    # 方差
    def variance(self):
        raise NotImplementedError

    # 采样
    def sample(self, shape=()):
        raise NotImplementedError

    # 重参数化采样
    def rsample(self, shape=()):
        raise NotImplementedError

    # 熵
    def entropy(self):
        raise NotImplementedError

    # 相对熵
    def kl_divergence(self, other):
        raise NotImplementedError

    # 对数概率密度
    def log_prob(self, value):
        raise NotImplementedError

    # 概率密度
    def prob(self, value):
        return self.log_prob(value).exp()
```

Normal分布

- 若随机变量 X 服从均值为 μ，方差为 σ^2 的 Normal 分布，则 X 的概率密度函数为

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- 熵

$$entropy(\sigma) = 0.5 \log(2\pi e \sigma^2)$$

- 相对熵

$$KL\_divergence(\mu_0, \sigma_0; \mu_1, \sigma_1) = 0.5(ratio^2 + (\frac{diff}{\sigma_1})^2 - 1 - 2\ln ratio)$$

$$ratio = \frac{\sigma_0}{\sigma_1}$$

$$diff = \mu_1 - \mu_0$$

Normal分布

- 数值计算

  均值 mean

  方差 variance

  概率密度 prob

  对数概率密度 log_prob

  熵 entropy

  相对熵 kl_divergence

```
class Normal(Distribution):
    def __init__(self, loc, scale):
        …

    def log_prob(self, value):
        var = self.scale * self.scale
        log_scale = paddle.log(self.scale)
        return paddle.subtract(
            -1.0 * ((value - self.loc) *
        (value - self.loc)) / (2.0 * var),
            log_scale +
        math.log(math.sqrt(2.0 * math.pi))
        )
```

Normal分布

- 采样
  sample(采样)和rsample (重参数化采样)的实现是难点
  生成一些满足特定分布的样本
  rsample生成的样本需支持反向传播
  可参考pytorch的实现

如果是通过变换已有的概率分布，来实现新的概率分布，一般不用自己实现sample和rsample

TransformedDistribution

- 如果概率分布p(x)可以表示成另一个概率分布p(y)，以及一个可微可逆的变换y=g(x)，那么可通过继承变换基类，将概率分布 p(x) 变换为新的概率分布 p(y)。

```
class TransformedDistribution(Distribution):
    def __init__(self, base, transforms):
        …

    def sample(self, shape=()):
        x = self._base.sample(shape)
                return xfor t in self._transforms:
            x = t.forward(x)
```

Transform

- 框架中已实现常用的变换

  PowerTransform：$y=x^p$

  ExpTransform：$y=e^x$

  …

- 可组成变换链

  例如：[ExpTransform(), Transform()]

```
__all__ = [  # noqa
    'Transform',
    'AbsTransform',
    'AffineTransform',
    'ChainTransform',
    'ExpTransform',
    'IndependentTransform',
    'PowerTransform',
    'ReshapeTransform',
    'SigmoidTransform',
    'SoftmaxTransform',
    'StackTransform',
    'StickBreakingTransform',
    'TanhTransform',
]
```

LogNormal分布

- 若lnX~Normal(μ, σ)，则称X服从LogNormal分布
- LogNormal分布与Normal分布的关系

$$X \sim Normal(\mu, \sigma)$$

$$Y = exp(X) \sim LogNormal(\mu, \sigma)$$

LogNormal的实现

- 基础分布Normal + 变换方法 ExpTransform

```
class LogNormal(TransformedDistribution):
    def __init__(self, loc, scale):
        self._base = Normal(loc=loc, scale=scale)
        self.loc = self._base.loc
        self.scale = self._base.scale
        super().__init__(self._base, [ExpTransform()])
```

LogNormal的实现

- 实现mean、variance、entropy(熵)方法
- prob(概率密度)、log_prob (对数概率密度) 、sample (采样) 、rsample (重参数采样)方法继承父类

单测

- 测试mean、variance、entropy、prob、log_prob、kl_divergence的准确性
- 可使用Numpy作为测试基准

单测

- 对sample和rsample的样本检验

  ①样本总体的均值和方差

  paddle.mean paddle.var

  ②KS检验：检验样本同分布

```
def _kstest(self, loc, scale, samples):
    # Uses the Kolmogorov-Smirnov test for goodness of fit.
    ks, _ = scipy.stats.kstest(
        samples, scipy.stats.lognorm(s=scale, scale=np.exp(loc)).cdf
    )
    return ks < 0.02
```