deep_ensembles

deep ensemble 网络结构：

- 输入层节点：7
- 隐藏层 1 节点：50
- 隐藏层 2 节点：50
- 输出层节点：2（mean、var）
- cost function：NLL

deep ensemble 参数设置：

```
    parser = argparse.ArgumentParser()
    # Ensemble size
    parser.add_argument('--ensemble_size', type=int, default=5,
                        help='Size of the ensemble')
    # Maximum number of iterations
    parser.add_argument('--max_iter', type=int, default=5000,
                        help='Maximum number of iterations')
    # Batch size
    parser.add_argument('--batch_size', type=int, default=10,
                        help='Size of batch')
    # Epsilon for adversarial input perturbation
    parser.add_argument('--epsilon', type=float, default=1e-2,
                        help='Epsilon for adversarial input perturbation')
    # Alpha for trade-off between likelihood score and adversarial score
    parser.add_argument('--alpha', type=float, default=0.5,
                        help='Trade off parameter for likelihood score and
adversarial score')
    # Learning rate
    parser.add_argument('--learning_rate', type=float, default=0.005,
                        help='Learning rate for the optimization')
    # Gradient clipping value
    parser.add_argument('--grad_clip', type=float, default=100.,
                        help='clip gradients at this value')
    # Learning rate decay
    parser.add_argument('--decay_rate', type=float, default=0.99,
                        help='Decay rate for learning rate')
    # Dropout rate (keep prob)
    parser.add_argument('--keep_prob', type=float, default=0.8,
                        help='Keep probability for dropout')
```

esn 参数设置：

```
esn_param = {'n_readout': 1000,
             'n_components': 20,
             'damping': 0.5,
             'weight_scaling': 0.9,
             'discard_steps': 0,
             'random_state': None}
```

box-cox 变换：

设 $wp \sim N(\mu, \sigma^2)$

则 $wp_{ln} = ln(wp + 0.01)$

$wp_{pred} = exp(f(X, wp_{ln})) - 0.01$

预测方差：$Var(wp_{ln}) = e^{2\mu+\sigma^2}(e^{\sigma^2} - 1)$

1. 西班牙数据集

train index: [6426, 10427] train_len: 4000

test index: [14389, 15390] test_len: 1000

- **输入特征：**

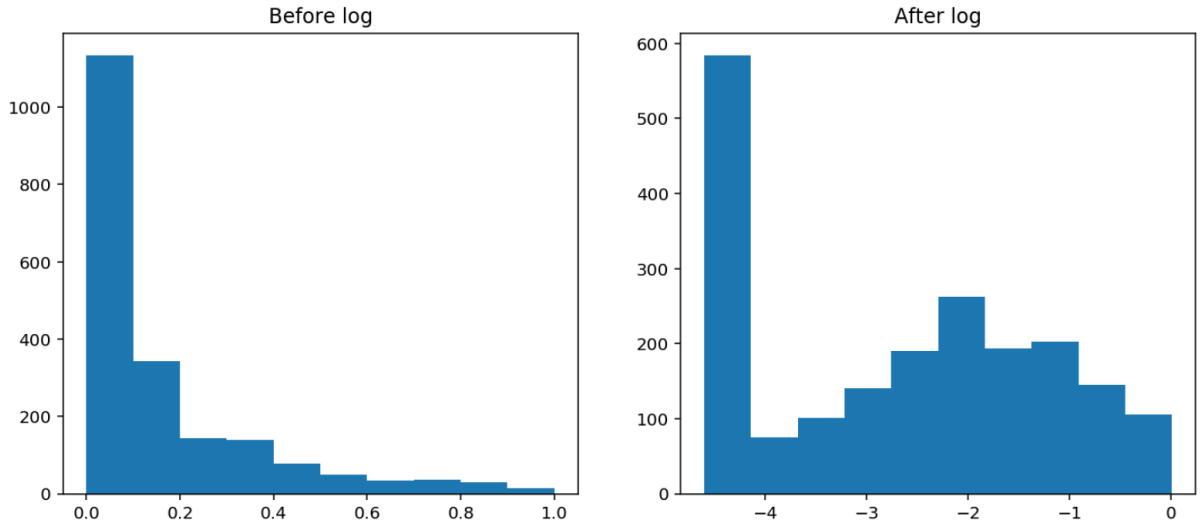
```
'wind_speed', 'sin(wd)', 'cos(wd)', 【t期】
'wind_speed-1', 'sin(wd)-1','cos(wd)-1', 'wind_power-1'【t-1期】
```

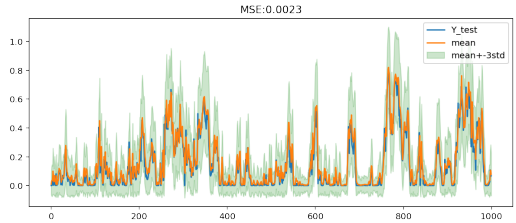
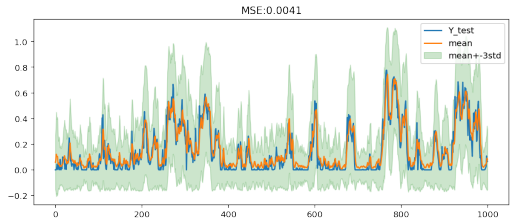
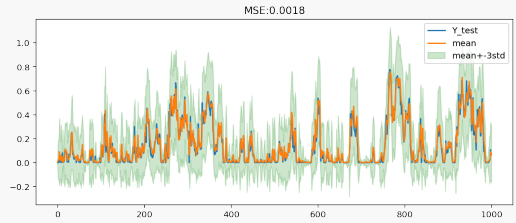
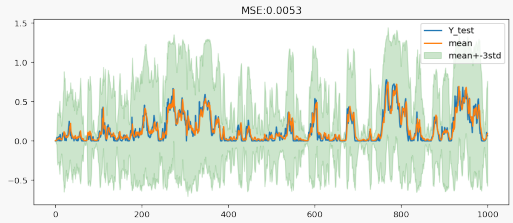
- **输出：**wind_power

box-cox 转换前后数据分布：

```
---- Spain dataset ----
get_data(hour_num=1, transform='sin+cos', drop_time=True, scale=True)

Input space: Index(['wind_speed', 'sin(wd)', 'cos(wd)', 'wind_speed-1', 'sin(wd)-
1',
       'cos(wd)-1', 'wind_power-1'],
      dtype='object')
train index: [6426, 8427] train_len: 2000
test index: [14389, 15390] test_len: 1000
```

: Text(0.5, 1.0, 'After log')



	无 esn	有 esn
无 box-cox		
有 box-cox		

2. 美国数据集

train index: [3001, 7002] train_len: 4000

test index: [2000, 3001] test_len: 1000

- **输入特征：**

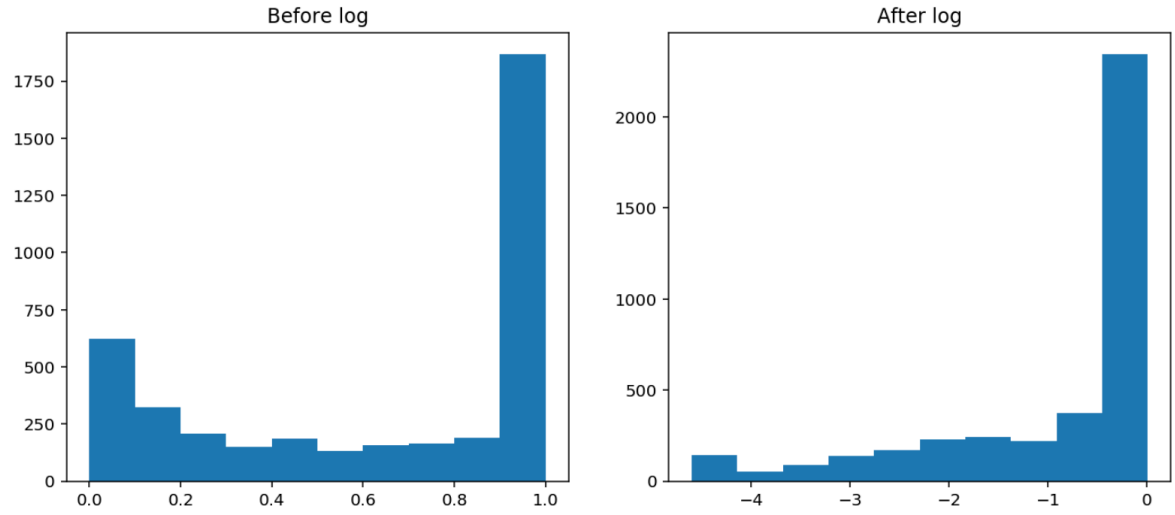
```
'wind_speed', 'sin(wd)', 'cos(wd)', 【t期】
'wind_speed-1', 'sin(wd)-1','cos(wd)-1', 'wind_power-1'【t-1期】
```

- **输出**：wind_power

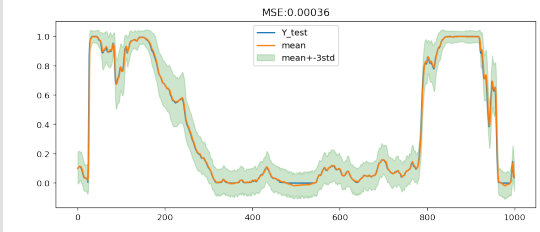
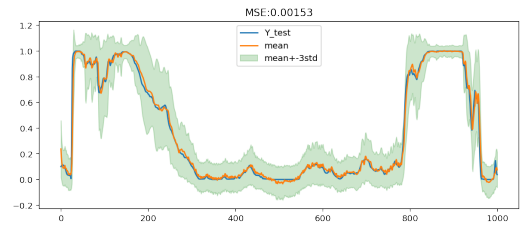
box-cox 转换前后数据分布：

```
---- US dataset ----
get_data2(hour_num=1, transform='sin+cos', drop_time=True, drop_esle=True, scale=
True)

Data: ['相近8个地点2012年数据', '20738-2012.csv']

Input space: Index(['wind_speed', 'sin(wd)', 'cos(wd)', 'wind_speed-1', 'sin(wd)-
1',
       'cos(wd)-1', 'wind_power-1'],
      dtype='object')
train index: [3001, 7002] train_len: 4000
test index: [2000, 3001] test_len: 1000
```

: Text(0.5, 1.0, 'After log')



结果：

	无 esn	有 esn
无 box-cox		
有 box-cox