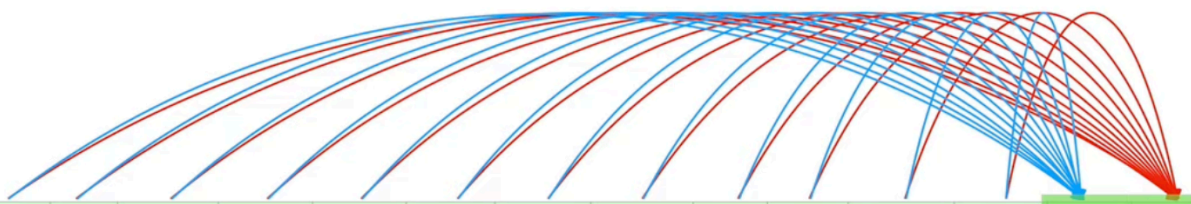


# 10.4강 여러개의 변수 : 여러개의 종속변수



1	2	3	4	5	6	7	8	9	10	11	12	13	14
CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
범죄율			강변		평균 방 수	노후주택 비율			재산세 세율	학생/교사 비율		하위계층 비율	집값
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1
1.23247	0	8.14	0	0.538	6.142	91.7	3.9769	4	307	21	396.9	18.72	15.2
0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9
0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15
8.98296	0	18.1	1	0.77	6.212	97.4	2.1222	24	666	20.2	377.73	17.6	17.8

위의 그림과 같이 여러개의 y(종속변수)가 나올때는 어떻게 처리하면 될까?

## 10.4.html

```
<!DOCTYPE html>
<html>

<head>
  <title>TensorFlow.js Tutorial - lemon</title>
```

```

    <!-- Import TensorFlow.js -->
    <script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@1.0.0/dist/tf.min.js"
></script>
    <script src="10.4.js"></script>

</head>

<body>
    <script>
        // 1. 과거의 데이터를 준비합니다.
        var 원인 = tf.tensor(보스톤_원인);
        var 결과 = tf.tensor(보스톤_결과);

        // 2. 모델의 모양을 만듭니다.
        var X = tf.input({ shape: [12] }); // 입력 데이터 12개로 변경
        var Y = tf.layers.dense({ units: 2 }).apply(X); // 예측하고 싶은 데이
터는 2개 변경
        var model = tf.model({ inputs: X, outputs: Y });
        var compileParam = { optimizer: tf.train.adam(), loss:
tf.losses.meanSquaredError }
        model.compile(compileParam);

        // 3. 데이터로 모델을 학습시킵니다.
        //         var fitParam = {epochs: 100}
        var fitParam = {
            epochs: 3000,
            callbacks: {
                onEpochEnd: // 콜백 함수 얼마나 실행되었는가를 볼 수 있다
                    function (epoch, logs) { // epoch, logs
                        console.log('epoch', epoch, logs, 'RMSE=>',
Math.sqrt(logs.loss)); // loss : 얼마나 잘 학습되었는냐 0에 가까울 수록 학습이 잘 되
었는지 알 수 있다.
                    }
            }
        } // loss 추가 예제
        model.fit(원인, 결과, fitParam).then(function (result) {

            // 4. 모델을 이용합니다.
            // 4.1 기존의 데이터를 이용
            var 예측한결과 = model.predict(원인);
            예측한결과.print();

        });

```

```

// 4.2 새로운 데이터를 이용
// var 다음주온도 = [15,16,17,18,19]
// var 다음주원인 = tf.tensor(다음주온도);
// var 다음주결과 = model.predict(다음주원인);
// 다음주결과.print();

</script>
</body>

</html>

```

```

> var ws = model.getWeights();
Array(2)
> ws[0]
< ▶ e {isDisposedInternal: false, shape: Array(2), dtype: 'float32', size: 24, strides: Array(1), ...}
> ws[0].arraySync();
< ▼ (12) [Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2)]
  ▶ 0: (2) [0.12457739561796188, -0.15169298648834229]
  ▶ 1: (2) [0.017887912690639496, 0.04206229746341705]
  ▶ 2: (2) [0.15971790254116058, -0.11401519179344177]
  ▶ 3: (2) [-0.8495632410049438, 3.251396894454956]
  ▶ 4: (2) [5.44150972366333, -0.8302165269851685]
  ▶ 5: (2) [-2.5784597396850586, 6.471935272216797]
  ▶ 6: (2) [0.10917053371667862, -0.06289663910865784]
  ▶ 7: (2) [0.515999972820282, -1.237860918045044]
  ▶ 8: (2) [-0.06506836414337158, 0.21665240824222565]
  ▶ 9: (2) [0.0039730495773255825, -0.01313089206814766]
  ▶ 10: (2) [0.6066945195198059, -0.7103012800216675]
  ▶ 11: (2) [-0.002926257438957691, 0.014377005398273468]
    length: 12
    ▶ [[Prototype]]: Array(0)
> ws[1].arraySync();
< ▶ (2) [3.320537567138672, 2.992661476135254]

```

위와 같이 가중치와 bais를 구할 수 있다