Part I
Math

1. Linear algebra

1.1. Identity and Inverse Matrices

We denote the identity matrix that preserves n-dimensional vectors as I_n:

$$\mathbf{I}_n\mathbf{x} = \mathbf{x}$$

The structure of the identity matrix is simple: all of the entries along the main diagonal are 1, while all of the other entries are zero. Единичная матрица квадратная.

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

Таким образом мы можем определить обратную матрицу как: $\mathbf{A}^{-1}$, для нее справедливо:

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_n$$

Part II
Machine learning

2. Loss Functions

2.1. MSE, OLS

Все это одно и то же по сути, **RSS** - residual sum of squares, **OLS** - ordinary least squares, **LS** - least squares, **MSE** - mean squared error, **SE** - squared error. В разных источниках можно встретить разные названия. Суть у этого всего одна: квадратичное отклонение. Можно запутаться конечно, но к этому быстро привыкаешь.

Стоит отметить, что MSE это средне-квадратичное отклонение, некое среднее значение ошибки для всего тренировочного набора данных. На практике обычно MSE и используется. Формула особо ничем не отличается:

$$MSE(\beta) = \frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

N - размер датасета, $\hat{y}_i$ - предсказание модели для y_i.

3. Supervised Learning

3.1. Linear regression

Обычная линейная модель:$\hat{y}(x) = f_\theta = \sum_{i=1}^{n} x_i\theta_i + \theta_0$. В векторном (матричном если тренируем сразу на батче) виде: $\hat{Y}(X) = \mathbf{X}^T\theta$.
При этом смещение (bias) θ_0 поместим в общий вектор, $x_0 = 1$.Используем функцию ошибки **RSS** (residaul sum of squares). Почти тоже самое, что и **MSE** (Смотреть в разделе subsection 2.1).

$$L = RSS = \sum_{i}^{N} (y_i - \hat{y}_i)^2$$

Или в векторном виде (далее в этом разделе все будет в вектрном виде):

$$L(\mathbf{X}) = (\mathbf{Y} - \hat{\mathbf{Y}})^2 = (\mathbf{Y} - \mathbf{X}^T\theta)^2$$

Наша цель найти параметры θ, для этого возьмем частную производную от функции ошибки L по θ и приравняем ее к нулю.

$$\frac{\delta L}{\delta \theta} = \frac{1}{2}(\mathbf{Y} - \mathbf{X}^T\theta)\mathbf{X} = 0$$

Отсуда можно найти θ:

$$(\mathbf{Y} - \mathbf{X}^T\theta)\mathbf{X} = \mathbf{Y}^T\mathbf{X} - \mathbf{X}^T\mathbf{X}\theta = 0$$

$$\theta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{Y}^T\mathbf{X}$$

Это прямой способ нахождения параметров, минусы заключаются в том, что для этого надо находить обратную матрицу $(\mathbf{X}^T\mathbf{X})^{-1}$, а она не всегда существует (плюс она должна быть квадратной, смотреть subsection 1.1). Такой способ называется решение на прямую или через **The Normal Equation** Так же: размерность обратной матрицы $n \times n$ The computational complexity of inverting such a matrix is typically about $O(n^2.4)$ to $O(n^3)$ (depending on the implementation). On the positive side, this equation is linear with regards to the number of instances in the training set (it is O(m)), so it handles large training sets efficiently, provided they can fit in memory. Использование итерационного метода (**Gradient Descent** или **Batch Gradient Descent**): Далее будет говорится о Batch Gradient Descent. Поэтому будем использовать **MSE** в качетсве ошибки:

$$L = MSE = \frac{1}{m}\sum_{i}^{N} (y_i - \hat{y}_i)^2$$

где m-размер батча.

$$\frac{\delta MSE}{\delta \theta_j} = \frac{1}{m}\sum_{j}^{m}(\mathbf{y}_j - \mathbf{x}_j^T\theta)\mathbf{x}_j$$

В данном случае к нулю приравнивать не надо, наща цель найти градиент функции стоимости (или функции ошибки, в данном случае MSE).

$$\nabla\theta = \frac{1}{m}(\mathbf{Y} - \mathbf{X}^T\theta)\mathbf{X}$$

Отсюда получаем значения параметров на следующем шаге:

$$\theta_{(nextstep)} = \theta - \eta\nabla\theta$$

Index