Reinforcement and Online Learning

Mahesan Niranjan

Week One: Recursive Least Squares

February 2, 2021

Starting Point: Linear Regression

- Data: $\{\boldsymbol{x}_n, \, y_n\}_{n=1}^{N}, \quad \boldsymbol{x} \in \mathcal{R}^p, \quad y_n \in \mathcal{R}$
- Usually $N > p$, Offset / bias term absorbed into $\boldsymbol{w}$.
- Model: $y \;=\; \boldsymbol{w}^T \, \boldsymbol{x}$
- Linear model on fixed nonlinear basis functions:

$$y \;=\; \boldsymbol{w}^T \, \boldsymbol{\Phi}(\boldsymbol{x})$$

  where $\boldsymbol{\Phi}(.)$ is from a set of fixed transforms *e.g.* Radial Basis Functions (RBF), with some hyperparameters in them.
- We minimize error: $E \;=\; ||\boldsymbol{y} \,-\, X\boldsymbol{w}||^2$
- X is an $N \times p$ matrix; $\boldsymbol{y}$ is $N \times 1$ vector.
- There is a closed form solution: $\boldsymbol{w} \;=\; \left(X^t \, X\right)^{-1} \, X^t \, \boldsymbol{y}$
- We can estimate $\boldsymbol{w}$ by a gradient descent algorithm:

$$\boldsymbol{w}^{(k+1)} \;=\; \boldsymbol{w}^{(k)} \;-\; \eta \, \nabla_{\boldsymbol{w}} E$$

- $\nabla_{\boldsymbol{w}} E$ is a vector, dimension p, derivative of E with respect to each of the weights:

$$\nabla_{\boldsymbol{w}} E \;=\; 2 \, X^t \, (\boldsymbol{y} \;-\; X\boldsymbol{w})$$

- This gradient is sum over all the data; we could also perform sample by sample update: Stochastic Gradient Descent

$$\boldsymbol{w}^{(k+1)} \;=\; \boldsymbol{w}^{(k)} \;-\; \eta \, (y(n) - \boldsymbol{w}^t \, \boldsymbol{x}_n) \, \boldsymbol{x}_n$$

- Structure: Constant $\times$ Error $\times$ Input

Recursive Least Squares

- In stochastic gradient descent (and batch gradient descent), we had a learning rate parameter: η
- In the closed form solution (via pseudo inverse), we had to invert a matrix $X^t X$
- Consider sequential arrival of data:
  - At time $n-1$

$$\mathbf{w}^{(n-1)} = \left(X_{n-1}^t X_{n-1}\right)^{-1} X_{n-1}^t \mathbf{y}_{n-1}$$

  - At time n

$$\mathbf{w}^{(n)} = \left(X_n^t X_n\right)^{-1} X_n^t \mathbf{y}_n$$

  - The system we solved has changed from $(n-1) \times p$ to $n \times p$
  - What is the difference?

$$X_n = \begin{bmatrix} X_{n-1} \\ \hline \mathbf{x}_n \end{bmatrix} \quad \text{and} \quad \mathbf{y}_n = \begin{bmatrix} \mathbf{y}_{n-1} \\ \hline y_n \end{bmatrix} \quad i.e. \quad \begin{bmatrix} \text{Data upto time (n} - 1) \\ \hline \text{New data at time (n)} \end{bmatrix}$$

- If we write $R_{n-1} = X_{n-1}^t X_{n-1}$ and $R_n = X_n^t X_n$, then

$$R_n = R_{n-1} + \mathbf{x}_n \mathbf{x}_n^t$$

Basic Idea: Matrix Inversion Lemma

- General form: (look up in Appendix A, Bishop)
- Rank one update:

$$\left(A + \mathbf{x}\mathbf{x}^t\right)^{-1} = A^{-1} - \frac{A^{-1} \mathbf{x} \mathbf{x}^t A^{-1}}{1 + \mathbf{x}^t A^{-1} \mathbf{x}}$$

- We can find inverse of $A + \mathbf{x}\mathbf{x}^t$ from A^{-1} without doing inverse calculations!
- Homework – verify the above is true!
- Back to the regression problem – we had:

$$R_n = R_{n-1} + \mathbf{x}_n \mathbf{x}_n^t$$

- We can calculate R_n^{-1} from R_{n-1}^{-1} without doing additional matrix inversion!
- And our solution of course is $R_n^{-1} X_n^t \mathbf{y}_n$
- It would help (shortly) to have $\mathbf{z}_n = X_n^t \mathbf{y}_n$ and similarly $\mathbf{z}_{n-1}$

Recursive Least Squares Algorithm

- We need at time n: $\mathbf{w}^{(n)} = R_n^{-1}\,\mathbf{z}_n$
- We reach this from time $n-1$: $\mathbf{w}^{(n-1)} = R_{n-1}^{-1}\,\mathbf{z}_{n-1}$
- Error at each data: $e_{n-1} = \mathbf{w}^T\,\mathbf{x}_{n-1} - y_{n-1}$
- We might be interested in forgetting the past: $\min \sum_{i=0}^{n-1} \lambda^{(n-1)-i}\, e_i^2$
- With this forgetting term, we have:

$$
\begin{aligned}
R_{n-1} &= \sum_{i=0}^{(n-1)} \lambda^{(n-1)-i}\,\mathbf{x}_i\,\mathbf{x}_i^T \\
\mathbf{z}_{n-1} &= \sum_{i=0}^{n-1} \lambda^{(n-1)-i}\,\mathbf{x}_i\,y_i
\end{aligned}
$$

- From which:

$$
\begin{aligned}
R_n &= \sum_{i=0}^{n-1} \lambda^{n-i}\,\mathbf{x}_i\,\mathbf{x}_i^T + \mathbf{x}_n\,\mathbf{x}_n^T \\
&= \lambda\left[\sum_{i=0}^{n-1} \lambda^{(n-1)-i}\,\mathbf{x}_i\,\mathbf{x}_i^T\right] + \mathbf{x}_n\,\mathbf{x}_n^T \\
&= \lambda\,R_{n-1} + \mathbf{x}_n\,\mathbf{x}_n^T
\end{aligned}
$$

- We now bring in matrix inversion lemma (for rank one update)

Recursive Least Squares (cont'd)

- Rank one update of inverse:

$$
R_n^{-1} = \frac{1}{\lambda}\,R_{n-1}^{-1} - \frac{\frac{1}{\lambda^2}\,R_{n-1}^{-1}\,\mathbf{x}_n\,\mathbf{x}_n^T\,R_{n-1}^{-1}}{1 + \mathbf{x}_n^T\,\frac{1}{\lambda}\,R_{n-1}^{-1}\,\mathbf{x}_n}
$$

- Define: $P_n = R_n^{-1}$, $P_{n-1} = R_{n-1}^{-1}$
- Define a gain vector $\mathbf{k}_n$:

$$
P_n = \frac{1}{\lambda}\,P_{n-1} - \frac{\frac{1}{\lambda^2}\,P_{n-1}\,\mathbf{x}_n\,\mathbf{x}_n^T\,P_{n-1}}{1 + \mathbf{x}_n^T\,\frac{1}{\lambda}\,P_{n-1}\,\mathbf{x}_n} = \frac{1}{\lambda}\,P_{n-1} - \frac{1}{\lambda}\,\mathbf{k}_n\,\mathbf{x}_n^T\,P_{n-1},
$$

- Which leads to:

$$
\mathbf{k}_n = \frac{\frac{1}{\lambda}\,P_{n-1}\,\mathbf{x}_n}{1 + \frac{1}{\lambda}\,\mathbf{x}_n^T\,P_{n-1}\,\mathbf{x}_n}
$$

$$
\mathbf{z}_n = \sum_{i=0}^{n} \lambda^{n-i}\,\mathbf{x}_i\,y_i = \lambda\,\mathbf{z}_{n-1} + \mathbf{x}_n\,y_n
$$

- and...

$$
\begin{aligned}
\mathbf{k}_n &= \left[\frac{1}{\lambda}\,P_{n-1} - \frac{1}{\lambda}\,\mathbf{k}_n\,\mathbf{x}_n^T\,P_{n-1}\right]\mathbf{x}_n \\
&= P_n\,\mathbf{x}_n
\end{aligned}
$$

Recursive Least Squares (cont'd)

- Update for $\boldsymbol{w}$ at time n

$$
\begin{aligned}
\boldsymbol{w}^{(n)} &= P_n\, \boldsymbol{z}_n \\
&= P_n\, [\lambda \boldsymbol{z}_{n-1} + \boldsymbol{x}_n\, y_n] \\
&= \lambda\, P_n\, \boldsymbol{z}_{n-1} + P_n\, \boldsymbol{x}_n\, y_n \\
&= \lambda \left[\frac{1}{\lambda} P_{n-1} - \frac{1}{\lambda} \boldsymbol{k}_n\, \boldsymbol{x}_n^T\, P_{n-1} \right] \boldsymbol{z}_{n-1} + P_n\, \boldsymbol{x}_n\, y_n \\
&= P_{n-1}\, \boldsymbol{z}_{n-1} - \boldsymbol{k}_n\, \boldsymbol{x}_n^T\, P_{n-1}\, \boldsymbol{z}_{n-1} + P_n\, \boldsymbol{x}_n\, y_n \\
&= \boldsymbol{w}^{(n-1)} - \boldsymbol{k}_n \left(\boldsymbol{x}_n^T\, \boldsymbol{w}^{(n-1)} - y_n \right)
\end{aligned}
$$

- Structure:

$$\boxed{\text{New = Old - Gain X Prediction Error}}$$