CS 412 — Introduction to Machine Learning (UIC)	January 30, 2025

Lecture 3

Instructor: Aadirupa Saha *Scribe(s): Charis Hulu / Lokesh Boggavarapu*

Overview

In the last lecture, we covered the following main topics:

1. Empirical Risk Minimization(ERM)
2. Hypothesis/Function Class
3. Linear Regression
4. Linear Regression with Regularization

This lecture focuses on:

1. Regression Models: Linear and Polynomial
2. Overfitting

Based on the building blocks of the previous lecture, we will explore how model complexity impacts generalization, with a focus on linear and polynomial regression. Linear regression assumes a simple linear relationship, while polynomial regression introduces higher-order terms for greater flexibility. Overfitting occurs when a model is too complex, fitting noise in the training data and performing poorly on unseen data. Using examples of low- and high-degree polynomial models, we illustrate the trade-off between minimizing training error and ensuring good generalization.

1 Regression Models: Linear and Polynomial

1.1 Linear Regression

Definition 1 (Linear Regression Model)**.** *: Given a dataset with n data points and d features, linear regression models the relationship between the input features $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ and the target variable y. The linear regression model assumes the following form:*

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b$$

where:

- *y is the target variable,*
- *$w_1, w_2, \ldots, w_d$ are the weights (coefficients) corresponding to each feature,*
- *b is the bias term,*

- $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ *is the input feature vector.*

In matrix form, the model can be written as:

$$\mathbf{y} = \mathbf{w}\mathbf{X} + \mathbf{b}$$

where:

- $\mathbf{y}$ *is the vector of target values for all data points,*
- $\mathbf{X}$ *is the matrix of input features, with each row representing one data point,*
- $\mathbf{w}$ *is the vector of model weights.*

Linear regression aims to find the values of w and b that minimize the error in predictions across the dataset.

Theorem 1.1: Mathematical Representation of Linear regression model

Let $\mathcal{F}_{\Theta}^{linear}$ be a class of functions parameterized by $\theta \in \Theta$, where $\Theta = \mathbb{R}^d \times \mathbb{R}^1, d \in \mathbb{Z}^+$. Then, a linear regression model of the input $X \subseteq \mathbb{R}^d$ is $f \in \mathcal{F}_{\Theta}^{linear}$, where

$$\mathcal{F}_{\Theta}^{linear} = \left\{ f_\theta : \mathbb{R}^d \to \mathbb{R} \mid f_\theta(x) = w^T x + b, (w, b) \in \theta, w \in \mathbb{R}^d, b \in \mathbb{R}, \forall x \in X \right\}.$$

1.2 Example: Predicting House Prices

Let's consider a simple example where we predict the price of a house (y) based on its size (x) in square feet.

$$y = wx + b$$

where x is the house size and y is the price of the house. We have the following data for 5 houses:

House Size (sq ft), x	Price (in \$1000), y
1500	400
1800	500
2400	600
3000	700
3500	750

We want to fit a line to this data to predict the price based on the size of a house.

1.3 Objective: Minimizing the Error

The objective in linear regression is to minimize the difference between the predicted values $\hat{y}$ and the actual values y. This is typically done using the *Mean Squared Error* (MSE) loss function:

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

where:

- y_i is the actual value for the i-th data point,

- $\hat{y}_i = wx_i + b$ is the predicted value for the i-th data point.

To find the optimal parameters w and b, we differentiate the loss function with respect to both w and b, and then set the derivatives equal to zero. This results in a system of equations, which are solved to obtain the values of w and b.

Exercise 1.1: Linear Regression

Solve the optimization problem for simple linear regression (with $d = 1$).

The optimized values for w and b should be the following

$$w = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

$$b = \frac{\sum y_i - w\sum x_i}{n}$$

where:

- w is the weight (slope),
- b is the bias (intercept),
- n is the number of data points.

1.4 Polynomial Regression

Polynomial regression is a type of regression where the relationship between the input feature(s) and the target variable is modeled as an k-degree polynomial.

Definition 2 (Polynomial Regression Model)**.** *Given a dataset with n data points and d features, polynomial regression models the relationship between the input features $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ and the target variable y. Unlike linear regression, polynomial regression assumes that the relationship between the features and target is captured by a polynomial equation, allowing for curves instead of straight lines. The polynomial regression model of degree k assumes the following form:*

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + w_{d+1} x_1^2 + w_{d+2} x_2^2 + \cdots + w_p x_d^k + b$$

where:

- *y is the target variable,*
- *$w_1, w_2, \ldots, w_p$ are the weights (coefficients) corresponding to the features,*
- *b is the bias term (sometimes merged in the weight vector as w_0),*
- *$\mathbf{x} = [x_1, x_2, \ldots, x_d]$ is the input feature vector,*
- *p is the number of polynomial features, which depends on the degree of the polynomial k.*

The model allows the input features to be raised to powers, enabling the capture of non-linear relationships between the features and the target variable.

Theorem 1.2: Mathematical Representation of Polynomial regression model

Let $\mathcal{F}_{\Theta}^{poly(k)}$ be a class of functions parameterized by $\theta \in \Theta$, where $\Theta = \mathbb{R}^p$ and $\varphi_k(x) : \mathbb{R}^d \to \mathbb{R}^p$ be a polynomial feature map, $p = \binom{d+k}{k}, d, k \in \mathbb{Z}^+$. Then, a k-degree polynomial regression model of the input $X \subseteq \mathbb{R}^d$ is a function f in

$$\mathcal{F}_{\Theta}^{poly(k)} = \left\{ f_\theta : \mathbb{R}^d \to \mathbb{R} \mid f_\theta(x) = \varphi_k(x)^T w, w \in \theta, \forall x \in X \right\}.$$

1.5 Example

For example, assume we have an input feature x with $d = 3$ features, and we want to apply a polynomial regression model with degree $k = 2$. This means we will add squared terms, interaction terms, and linear terms to our features.
The number of polynomial features created by $\varphi_2(x)$ is:

$$p = \binom{3+2}{2} = 10$$

This results in 10 features for the polynomial regression model. The polynomial feature map $\varphi_2(x)$ for $d = 3$ features is shown below. The corresponding weight vector w also has 10 parameters

$$\varphi_2(x) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_1^2 \\ x_2^2 \\ x_3^2 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \end{bmatrix} \qquad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \end{bmatrix}$$

Theorem 1.3: Linear and Polynomial Models Relationship

Linear regression is a special case of polynomial regression, that is, $\mathcal{F}_{\Theta}^{linear} = \mathcal{F}_{\Theta}^{poly(1)}$

2 Overfitting

Our goal in training a supervised machine learning model is not only to minimize the loss function on the training data but also to create a model that generalizes well to unseen data. If the difference between the training loss and test loss is too large, it suggests that the model fits the training data too closely and fails to recognize patterns in the test data. In this case, the model is **overfitting**.

To formulate this, we partition the dataset D into a training set D_{train} and a test set D_{test}. Instead of finding the optimal parameter $\hat{\theta}$ that minimizes the loss function $\mathcal{L}$ on the entire dataset D, we aim to fit the model to the training set, which can be expressed as:

$$\hat{\theta} = \arg\min_\theta \mathcal{L}(D_{train}; \theta) = \arg\min_\theta \frac{1}{|D_{train}|} \sum_{(x,y)\in D_{train}} \ell(y, f(x, \theta))$$

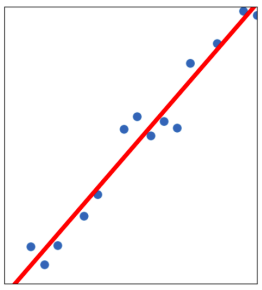
Once we have the optimal parameter $\hat{\theta}$, we can evaluate the trained model by calculating the difference $\mathcal{L}(D_{test}; \hat{\theta}) - \mathcal{L}(D_{train}; \hat{\theta})$. The larger this value, the more overfitting our model is.

Overfitting can be avoided using two techniques:

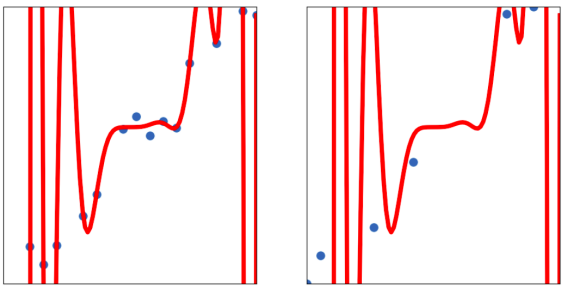
- **Using simpler functions**: Choosing simpler models helps prevent fitting noise in the data.
- **Regularization**: Adding a penalty to the model's complexity encourages it to avoid overfitting by limiting how much it can adapt to the training data.

2.1 Using simpler functions

A key cause of overfitting is using a model that's too complex. In polynomial regression, a higher-degree polynomial can fit the training data very closely, capturing even minor variations or noise. While this lowers the training error, it often results in poor performance on new data. In Figure 1, we compare polynomial regression models with degrees 1 (linear regression) and 15. The lower-degree model has higher training error but captures the main patterns, fitting the test data well. On the other hand, a higher-degree polynomial can overfit, creating a very flexible curve that fits the training data well but doesn't perform as well on the test data. This shows the trade-off between model complexity and generalization.



(a) 1**-degree polynomial regression.** The model (red line) fits the training set (left) while still capture the pattern of the test set (right)



(b) 15**-degree polynomial regression.** The model (red curve) fits the training set well (left) but performs poorly on the test set (right).

Figure 1: Comparison of simple and complex polynomial regression models.

2.2 Regularization

Regularization is another technique used to prevent overfitting by adding a penalty to the complexity of the model. This penalty discourages the model from fitting the noise in the data and helps it generalize better to unseen data.
There are two common types of regularization techniques:

- **L2 Regularization (Ridge Regression)**: In L2 regularization, the penalty term that is added is proportional to the square of the weights. The objective function becomes:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{d}\theta_j^2$$

  where λ is the regularization parameter, and θ_j are the model weights. The larger the value of λ, the stronger the penalty on the weights, forcing them to be smaller.

- **L1 Regularization (Lasso Regression)**: In L1 regularization, the penalty term that is added is proportional to the absolute value of the weights. The objective function becomes:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{d}|\theta_j|$$

  This regularization technique helps simplify the model by making some of the weights zero, which effectively selects the most important features

Regularization helps balance between fitting the training data well and creating a model that works well on new, unseen data.

Next Lecture

The next lecture will cover the following topics:
(i) Logistic Regression
(ii) Multiclass Logistic Regression (MLR)

References:

1. Saha, Aadirupa. Lecture Notes for CS 412 - Introduction to Machine Learning. University of Illinois at Chicago, Spring 2025.

2. Murphy, Kevin P. "Introduction to Probabilistic Machine Learning." Probabilistic Machine Learning: An Introduction, MIT Press, 2022, pp. 11–13.