# Introduction

Useful Sources: Section 28.3 of textbook, Wikipedia articles on Gradient Descent and Machine Learning

*Machine learning* is concerned with algorithms that can learn from data (or experience) to improve performance.

Assume the **performance measure** is an error function (or loss, risk, or cost function).

Assume algorithm finds values for parameters and/or constructs a specific kind of data structure such as a neural network or Bayesian network.

---

# Introduction Continued

A *hypothesis* is a specific assignment to the parameters/data structure.

The typical basic machine learning algorithm performs *incremental improvement*. That is, it has an initial hypothesis, then makes incremental changes to decrease its performance measure.

*Gradient descent* is one approach for implementing incremental improvement.

There are many more sophisticated variations, but they generally follow the above ideas.
Textbook Example

As an example of producing a least-squares fit, suppose that we have five data points

\[(x_1, y_1) = (-1, 2),\]
\[(x_2, y_2) = (1, 1),\]
\[(x_3, y_3) = (2, 1),\]
\[(x_4, y_4) = (3, 0),\]
\[(x_5, y_5) = (5, 3),\]

shown as black dots in Figure 28.3. We wish to fit these points with a quadratic polynomial

\[F(x) = c_1 + c_2 x + c_3 x^2.\]

Matrix Representation

Minimize \[\| A \mathbf{c} - \mathbf{y} \|^2 \]

\[
A = \begin{pmatrix}
1 & x_1 & x_1^2 \\
1 & x_2 & x_2^2 \\
1 & x_3 & x_3^2 \\
1 & x_4 & x_4^2 \\
1 & x_5 & x_5^2
\end{pmatrix} = \begin{pmatrix}
1 & -1 & 1 \\
1 & 1 & 1 \\
1 & 2 & 4 \\
1 & 3 & 9 \\
1 & 5 & 25
\end{pmatrix} \quad \mathbf{c} = \begin{pmatrix}
c_1 \\
c_2 \\
c_3
\end{pmatrix} = \begin{pmatrix}
1.200 \\
-0.757 \\
0.214
\end{pmatrix}
\]

\[
\mathbf{Y} = \begin{pmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5
\end{pmatrix} = \begin{pmatrix}
2 \\
1 \\
1 \\
0 \\
3
\end{pmatrix}
\]