Neural Networks

Lecturer: Masha Itkina

Announcements

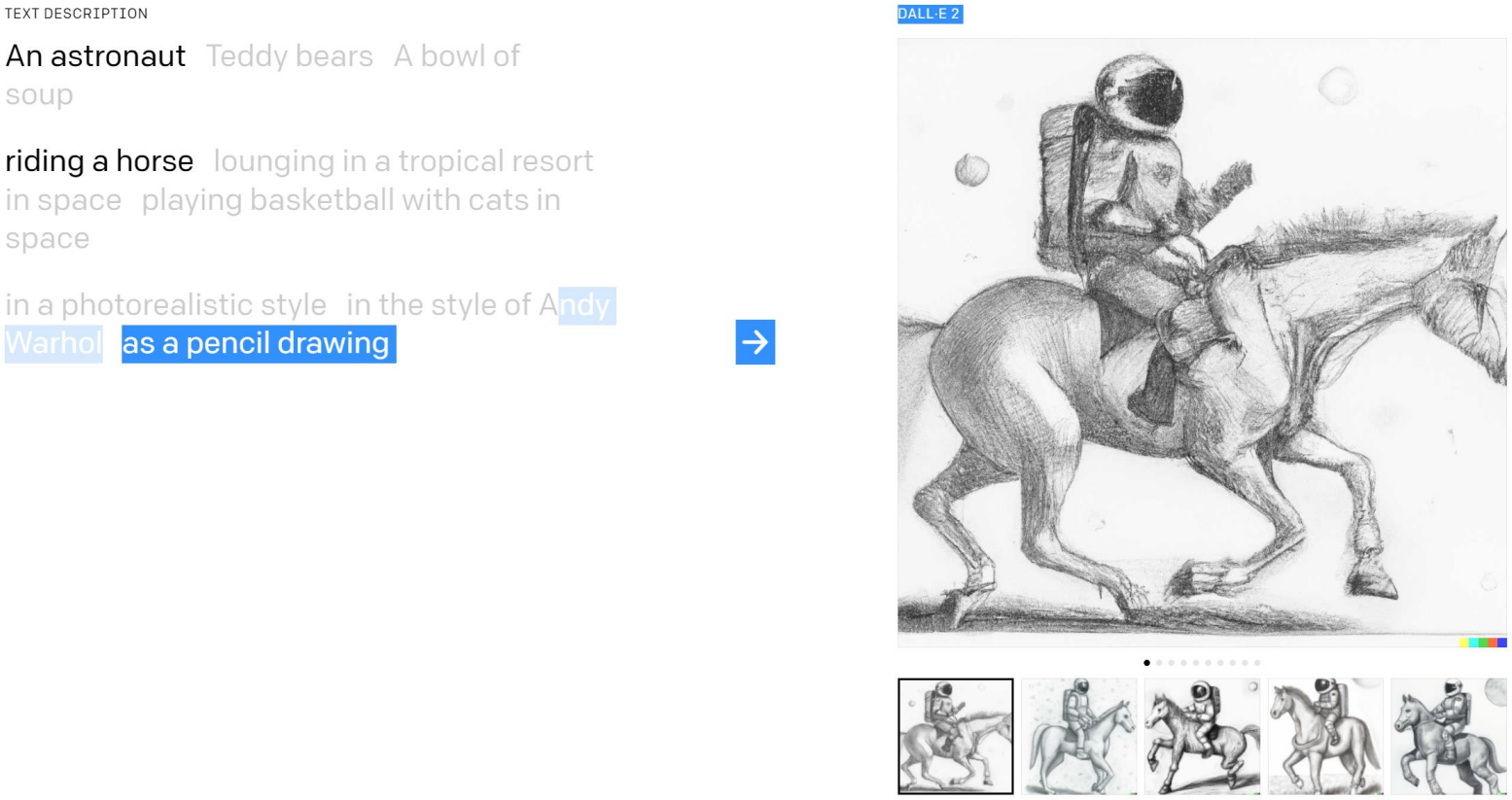
- Problem Set 1 is due today
- Problem Set 2 will be out later tonight; due May 4th
- Feedback on Project Proposals will be released within a week

Neural Networks in the Wild

```
-------- Generated Poem 1 --------

I must have shadows on the way
If I am to walk I must have
Each step taken slowly and alone
To have it ready made

And I must think in lines of grey
To have dim thoughts to be my guide
Must look on blue and green
And never let my eye forget
That color is my friend
And purple must surround me too

The yellow of the sun is no more
Intrusive than the bluish snow
That falls on all of us.  I must have
Grey thoughts and blue thoughts walk with me
If I am to go away at all.
```

GPT-3: Brown et. Al, “Language Models are Few-Shot Learners”, NeurIPS 2020.

Neural Networks in the Wild



DALLE-2: Ramesh et. Al, “Hierarchical Text-Conditional Image Generation with CLIP Latents”, ArXiv 2022.

Agenda for Today

- Supervised learning with non-linear models
- Neural networks

Linear Regression Review

Non-Linear Models: Kernels

Non-Linear Models

Non-Linear Models

We want to optimize: $\min_{\theta} J(\theta)$

Gradient Descent (GD): $\theta := \theta - \alpha \nabla_{\theta} J(\theta)$

Non-Linear Models

We want to optimize: $\min_{\theta} J(\theta)$

Stochastic Gradient Descent (SGD):

Algorithm 1 Stochastic Gradient Descent

1: Hyperparameter: learning rate α, number of total iteration n_{iter}.
2: Initialize θ randomly.
3: **for** $i = 1$ to n_{iter} **do**
4: Sample j uniformly from $\{1, \ldots, n\}$, and update θ by

$$\theta := \theta - \alpha \nabla_{\theta} J^{(j)}(\theta)$$

Non-Linear Models

We want to optimize: $\min_{\theta} J(\theta)$

Mini-batch SGD:

Algorithm 2 Mini-batch Stochastic Gradient Descent

1: Hyperparameters: learning rate α, batch size B, # iterations n_{iter}.
2: Initialize θ randomly
3: **for** $i = 1$ to n_{iter} **do**
4: Sample B examples $j_1, \ldots, j_B$ (without replacement) uniformly from $\{1, \ldots, n\}$, and update θ by

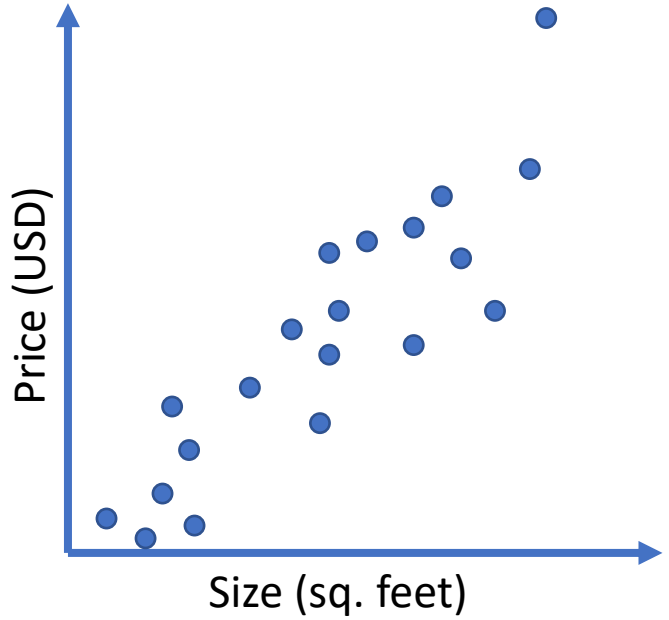
$$\theta := \theta - \frac{\alpha}{B} \sum_{k=1}^{B} \nabla_\theta J^{(j_k)}(\theta)$$

Neural Networks

- How to define $h_\theta(x)$?
  - Neural network!

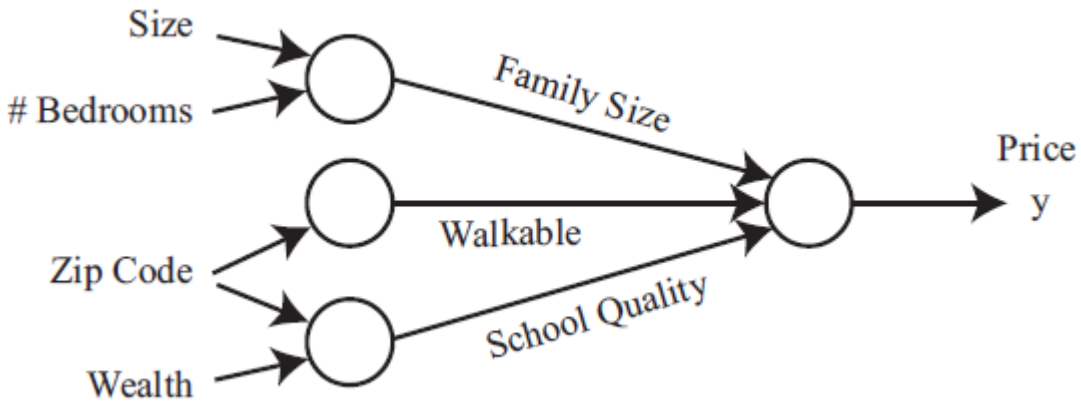
- How to compute $\nabla J^{(j)}(\theta)$?
  - Backpropagation (next lecture)

Housing Price Prediction



Housing Price Prediction

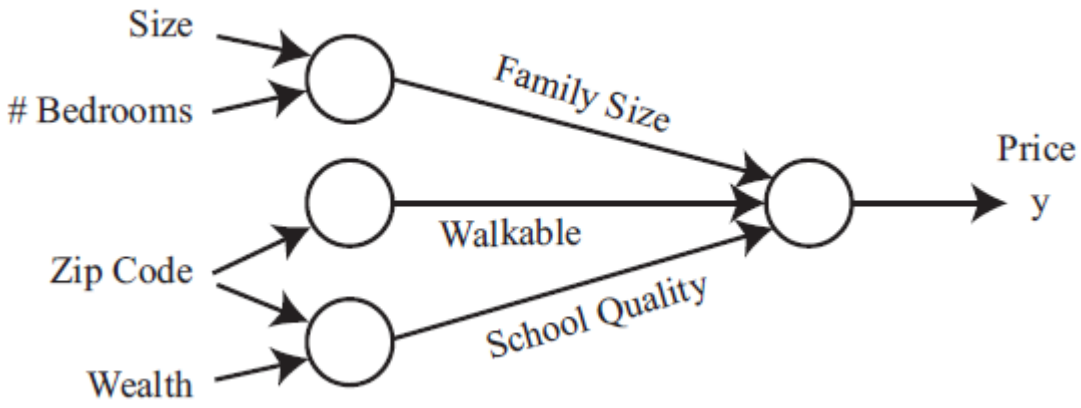
Housing Price Prediction



Two-Layer Neural Network

What if we do not have prior knowledge?

- Fully connected neural network
- Intermediate variables -> hidden units



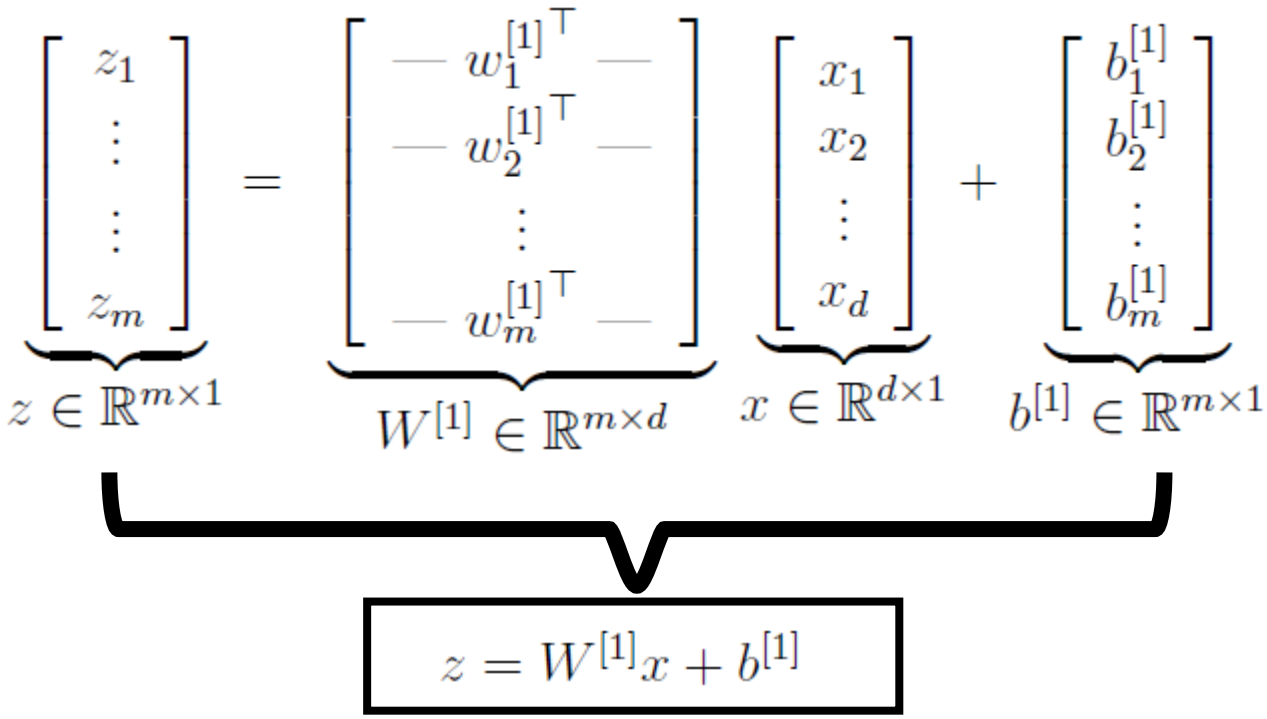
Two-Layer Neural Network

$$
\begin{aligned}
\forall j \in [1, ..., m], \quad z_j &= {w_j^{[1]}}^\top x + b_j^{[1]} \text{ where } w_j^{[1]} \in \mathbb{R}^d, b_j^{[1]} \in \mathbb{R} \\
a_j &= \mathrm{ReLU}(z_j), \\
a &= [a_1, \ldots, a_m]^\top \in \mathbb{R}^m \\
h_\theta(x) &= {w^{[2]}}^\top a + b^{[2]} \text{ where } w^{[2]} \in \mathbb{R}^m, b^{[2]} \in \mathbb{R},
\end{aligned}
$$

Vectorization

$$W^{[1]} = \begin{bmatrix} — & {w_1^{[1]}}^\top & — \\ — & {w_2^{[1]}}^\top & — \\ & \vdots & \\ — & {w_m^{[1]}}^\top & — \end{bmatrix} \in \mathbb{R}^{m \times d}$$

Vectorization



Vectorization

Multi-Layer Fully-Connected Neural Networks

$$
\begin{aligned}
a^{[1]} &= \mathrm{ReLU}(W^{[1]}x + b^{[1]}) \\
a^{[2]} &= \mathrm{ReLU}(W^{[2]}a^{[1]} + b^{[2]}) \\
&\cdots \\
a^{[r-1]} &= \mathrm{ReLU}(W^{[r-1]}a^{[r-2]} + b^{[r-1]}) \\
h_\theta(x) &= W^{[r]}a^{[r-1]} + b^{[r]}
\end{aligned}
$$

Why do we need an activation function (e.g., ReLU)?

Connection to Kernel Methods

Summary

- Supervised learning with non-linear models
- Neural networks
- Next time: backpropagation