EEE5015: Machine Learning & Artificial Intelligence

Zhiyun Lin

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY
电子与电气工程系
DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

- Regression
  - Features
  - Training examples
  - Model
  - Loss function
  - Optimization

Problems for Today

- What should I watch this Friday?

Problems for Today

- What should I watch this Friday?

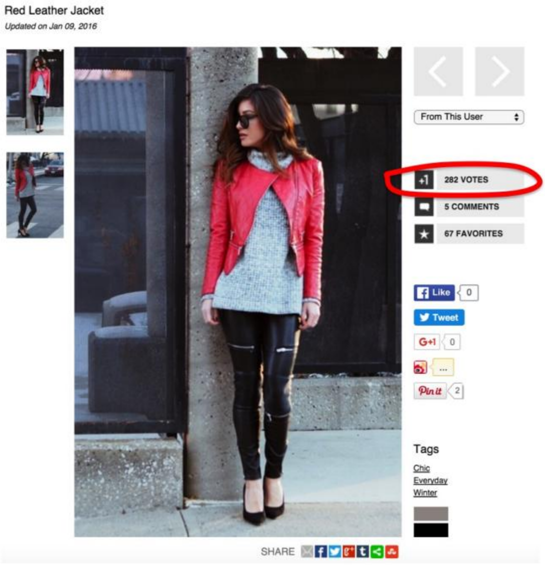
Problems for Today

- **Goal**: Predict movie rating automatically!

Factors:
-Year of release of the film in cinemas
-Length of the film in minutes
-Budget for the films production
-Number of positive votes received by viewers
-Genre of the film including Action, Animation, Comedy, Documentary, Drama, Romance and Short

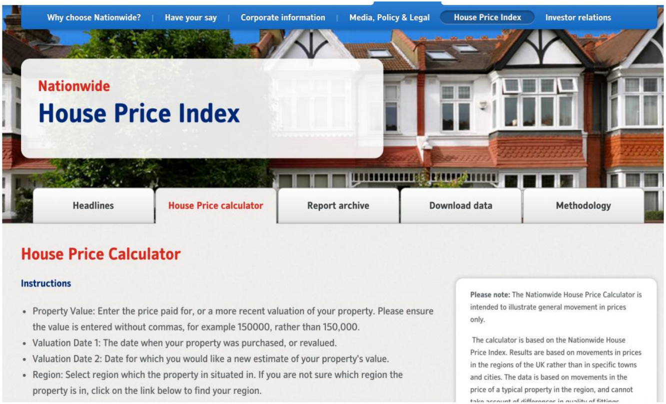
Problems for Today

- **Goal:** How many followers will I get?



Problems for Today

- **Goal:** Predict the price of the house



Regression

- What do all these problems have in common?

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these *t*

  (e.g., a rating: a real number between 0-10, # of followers, house price)

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these t

    (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these t

    (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?

Regression

- What do all these problems have in common?
    - Continuous outputs, we'll call these t

  (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?
    - Features (inputs), we'll call these x (or $\mathbf{x}$ if vectors)

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these t

    (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?
  - Features (inputs), we'll call these x (or $\mathbf{x}$ if vectors)
  - Training examples, many $x(i)$ for which $t(i)$ is known (e.g., many movies for which we know the rating)

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these t

    (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?
  - Features (inputs), we'll call these x (or $\mathbf{x}$ if vectors)
  - Training examples, many $x(i)$ for which $t(i)$ is known (e.g., many movies for which we know the rating)
  - A model, a function that represents the relationship between x and t

Regression

- What do all these problems have in common?
    - Continuous outputs, we'll call these t

  (e.g., a rating: a real number between 0-10, # of followers, house price)

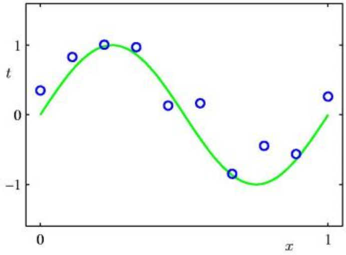
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?
    - Features (inputs), we'll call these x (or $\mathbf{x}$ if vectors)
    - Training examples, many $x(i)$ for which $t(i)$ is known (e.g., many movies for which we know the rating)
    - A model, a function that represents the relationship between x and t
    - A loss or a cost or an objective function, which tells us how well our model approximates the training examples

Regression

- What do all these problems have in common?
  - Continuous outputs, we'll call these t

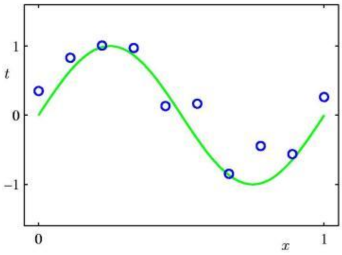
  (e.g., a rating: a real number between 0-10, # of followers, house price)
- Predicting continuous outputs is called regression
- What do I need in order to predict these outputs?
  - Features (inputs), we'll call these x (or $\mathbf{x}$ if vectors)
  - Training examples, many $x(i)$ for which $t(i)$ is known (e.g., many movies for which we know the rating)
  - A model, a function that represents the relationship between x and t
  - A loss or a cost or an objective function, which tells us how well our model approximates the training examples
  - Optimization, a way of finding the parameters of our model that minimizes the loss function

Simple 1-D regression



- Circles are data points (i.e., training examples) that are given to us

Simple 1-D regression

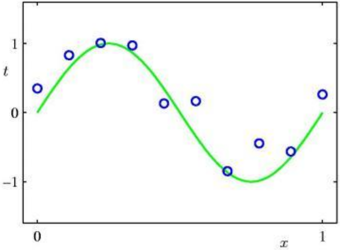


- Circles are data points (i.e., training examples) that are given to us
- The data points are uniform in x, but may be displaced in

$$t(x) = f(x) + \epsilon$$

with ϵ some noise

Simple 1-D regression

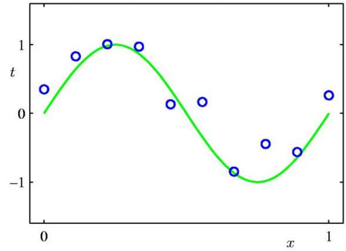


- Circles are data points (i.e., training examples) that are given to us
- The data points are uniform in x, but may be displaced in

$$t(x) = f(x) + \epsilon$$

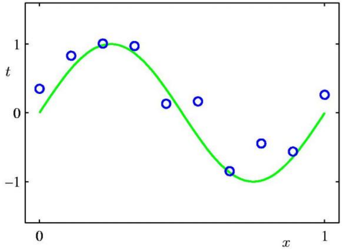
  with ϵ some noise
- In green is the "true" curve that we don't know
- Goal: We want to fit a curve to these points

Simple 1-D regression



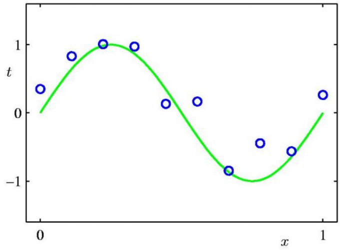
- Key Questions:
  - How do we parametrize the model?

Simple 1-D regression



- Key Questions:
  - How do we parametrize the model?
  - What loss (objective) function should we use to judge the fit?

Simple 1-D regression



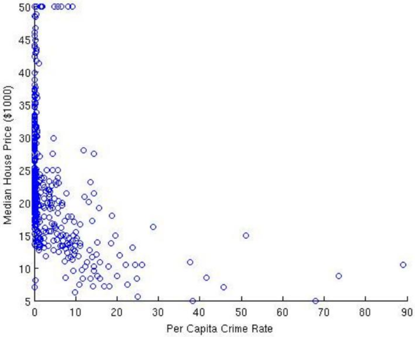
- Key Questions:
  - How do we parametrize the model?
  - What loss (objective) function should we use to judge the fit?
  - How do we optimize fit to unseen test data (generalization)?

Example: Boston Housing data

- Estimate median house price in a neighborhood based on neighborhood statistics

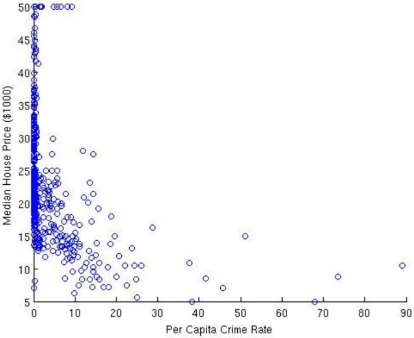
Example: Boston Housing data

- Estimate median house price in a neighborhood based on neighborhood statistics
- Look at first possible attribute (feature): per capita crime rate



Example: Boston Housing data

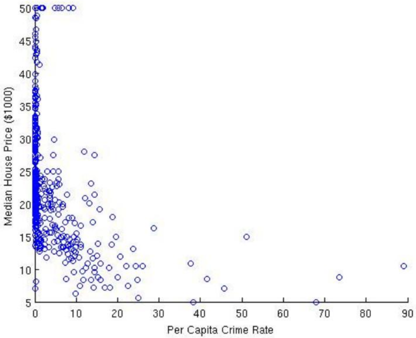
- Estimate median house price in a neighborhood based on neighborhood statistics
- Look at first possible attribute (feature): per capita crime rate



- Use this to predict house prices in other neighborhoods

Example: Boston Housing data

- Estimate median house price in a neighborhood based on neighborhood statistics
- Look at first possible attribute (feature): per capita crime rate



- Use this to predict house prices in other neighborhoods
- Is this a good input (attribute) to predict house prices?

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem
- Model outputs y, an estimate of t

$$y(x) = w_0 + w_1 x$$

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem
- Model outputs y, an estimate of t

$$y(x) = w_0 + w_1 x$$

- What type of model did we choose?

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem
- Model outputs y, an estimate of t

$$y(x) = w_0 + w_1 x$$

- What type of model did we choose?
- Divide the dataset into training and testing examples

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem
- Model outputs y, an estimate of t

$$y(x) = w_0 + w_1 x$$

- What type of model did we choose?
- Divide the dataset into training and testing examples
  - Use the training examples to construct hypothesis, or function approximator, that maps x to predicted y

Represent the Data

- Data is described as pairs $\mathcal{D} = \{(x^{(1)}, t^{(1)}), \cdots, (x^{(N)}, t^{(N)})\}$
  - $x \in \mathbb{R}$ is the input feature (per capita crime rate)
  - $t \in \mathbb{R}$ is the target output (median house price)
  - $^{(i)}$ simply indicates the training examples (we have N in this case)
- Here t is continuous, so this is a regression problem
- Model outputs y, an estimate of t

$$y(x) = w_0 + w_1 x$$

- What type of model did we choose?
- Divide the dataset into training and testing examples
  - Use the training examples to construct hypothesis, or function approximator, that maps x to predicted y
  - Evaluate hypothesis on test set

Noise

- A simple model typically does not exactly fit the data
  - lack of fit can be considered noise

Noise

- A simple model typically does not exactly fit the data
  - lack of fit can be considered noise
- Sources of noise:
  - Imprecision in data attributes (input noise, e.g., noise in per-capita crime)

Noise

- A simple model typically does not exactly fit the data
  - lack of fit can be considered noise
- Sources of noise:
  - Imprecision in data attributes (input noise, e.g., noise in per-capita crime)
  - Errors in data targets (mis-labeling, e.g., noise in house prices)

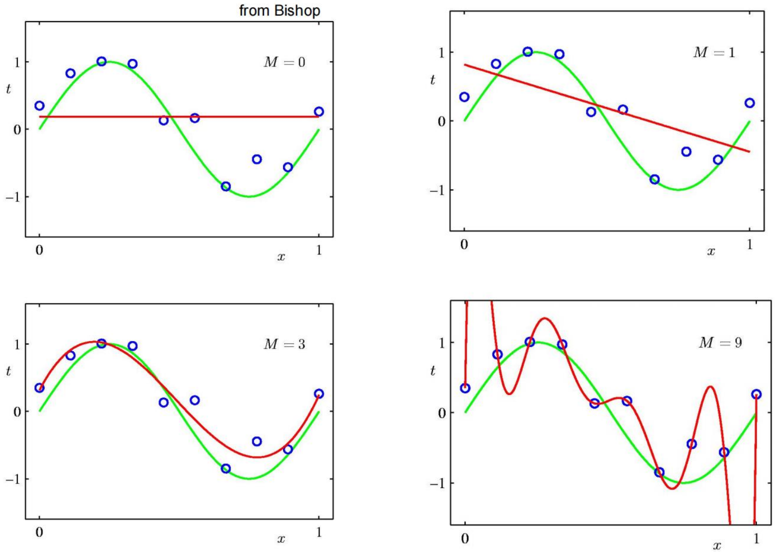
Noise

- A simple model typically does not exactly fit the data
  - lack of fit can be considered noise
- Sources of noise:
  - Imprecision in data attributes (input noise, e.g., noise in per-capita crime)
  - Errors in data targets (mis-labeling, e.g., noise in house prices)
  - Additional attributes not taken into account by data attributes, affect target values (latent variables). In the example, what else could affect house prices?

Noise

- A simple model typically does not exactly fit the data
  - lack of fit can be considered noise
- Sources of noise:
  - Imprecision in data attributes (input noise, e.g., noise in per-capita crime)
  - Errors in data targets (mis-labeling, e.g., noise in house prices)
  - Additional attributes not taken into account by data attributes, affect target values (latent variables). In the example, what else could affect house prices?
  - Model may be too simple to account for data targets

Which fit is best?



A toy example of linear regression

```python
import torch
import torch.nn as nn
import numpy as np
import matplotlib.pyplot as plt


# Hyper-parameters
input_size = 1
output_size = 1
num_epochs = 60
learning_rate = 0.001

i_iter = np.array([])
loss_iter = np.array([])

# Toy dataset
x_train = np.array([[3.3], [4.4], [5.5], [6.71], [6.93], [4.168],
                    [9.779], [6.182], [7.59], [2.167], [7.042],
                    [10.791], [5.313], [7.997], [3.1]], dtype=np.float32)

y_train = np.array([[1.7], [2.76], [2.09], [3.19], [1.694], [1.573],
                    [3.366], [2.596], [2.53], [1.221], [2.827],
                    [3.465], [1.65], [2.904], [1.3]], dtype=np.float32)

# Linear regression model
model = nn.Linear(input_size, output_size)
```

A toy example of linear regression

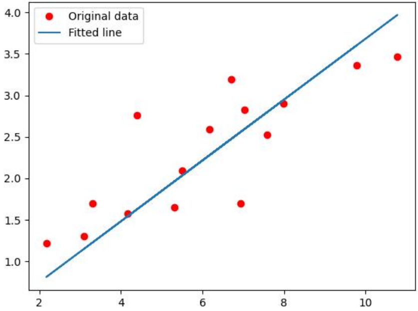
```python
# Loss and optimizer
criterion = nn.MSELoss()
optimizer = torch.optim.SGD(model.parameters(), lr=learning_rate)

# Train the model
for epoch in range(num_epochs):
    # Convert numpy arrays to torch tensors
    inputs = torch.from_numpy(x_train)
    targets = torch.from_numpy(y_train)

    # Forward pass
    outputs = model(inputs)
    loss = criterion(outputs, targets)

    # Backward and optimize
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    i_iter = np.append(i_iter, epoch)
    loss_iter = np.append(loss_iter, loss.item())

    if (epoch+1) % 5 == 0:
        print ('Epoch [{}/{}], Loss: {:.4f}'.format(epoch+1, num_epochs, loss.item()))

```

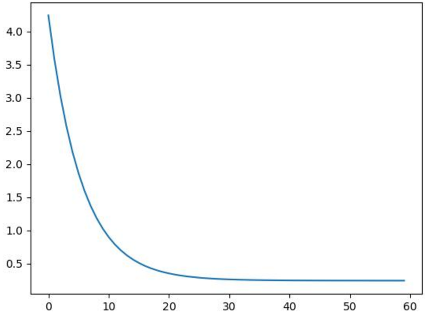
A toy example of linear regression

```python
# Plot the graph
predicted = model(torch.from_numpy(x_train)).detach().numpy()

plt.figure()
plt.plot(x_train, y_train, 'ro', label='Original data')
plt.plot(x_train, predicted, label='Fitted line')
plt.legend()

plt.figure()
plt.plot(i_iter,loss_iter)
plt.show()

# Save the model checkpoint
torch.save(model.state_dict(), 'model.ckpt')
```

A toy example of linear regression

```python
# Plot the graph
predicted = model(torch.from_numpy(x_train)).detach().numpy()

plt.figure()
plt.plot(x_train, y_train, 'ro', label='Original data')
plt.plot(x_train, predicted, label='Fitted line')
plt.legend()

plt.figure()
plt.plot(i_iter,loss_iter)
plt.show()
```





Summary of Regression

- **Regression:** to predict continuous outputs t
  - Consider proper features (inputs): x (or $\mathbf{x}$ if vectors)
  - Training examples, many $x(\mathrm{i})$ for which $t(i)$ is known (labeled)
  - A model, a function that represents the relationship between x and t

$$y = f(x, w)$$

  - A loss or a cost or an objective function, which tells us how well our model approximates the training examples
  - Optimization, a way of finding the parameters w of our model that minimizes the loss function