ORIE 4741: Learning with Big Messy Data

Feature Engineering

Professor Udell

Operations Research and Information Engineering
Cornell

October 16, 2021

Announcements 9/16/21

- section posted
- bonus section from last year: linear algebra review
- hw1 due today at 9:15am
- form project groups by this Sunday. see `https://people.orie.cornell.edu/mru8/orie4741/projects.html`
- looking for a project group? post your idea on zulip in the #project channel

Announcements 9/21/21

- hw2 posted, due next Thursday at 9:15am
- section this week: Linear algebra and gradient descent
- submit project groups immediately if you haven't yet!
- project proposals due Sunday night 10/3

Announcements 9/23/21

- hw2 posted, due next Thursday at 9:15am
  - select which pages correspond to which question (we may deduct points...)
- project:
  - submit your group by midnight tonight or we will assign you
  - you can edit the form if you add (or drop) a member
  - groups of 2: if you want a 3rd team member, message mad333 on zulip
  - project proposals due Sunday night 10/3
- zulip: topics with check marks are done; if you want an answer, open a new topic or remove the check mark from the topic

What makes a good project?

- Clear outcome to predict
- Linear regression should do something interesting
- A data science project; not an NLP or Vision project
- New, interesting model; not a Kaggle competition

Outline

Feature engineering

Polynomial transformations

Boolean, nominal, ordinal

Missing values

Nonlinear transformations, location

Text, images, ...

Time series

Linear models

To fit a linear model (= linear in parameters w)

- pick a transformation $\phi : \mathcal{X} \to \mathbf{R}^d$
- predict y using a linear function of $\phi(x)$

$$h(x) = w^T \phi(x) = \sum_{i=1}^{d} w_i (\phi(x))_i$$

- we want $h(x_i) \approx y_i$ for every $i = 1, \ldots, n$

Linear models

To fit a linear model (= linear in parameters w)

- pick a transformation $\phi : \mathcal{X} \to \mathbf{R}^d$
- predict y using a linear function of $\phi(x)$

$$h(x) = w^T \phi(x) = \sum_{i=1}^{d} w_i (\phi(x))_i$$

- we want $h(x_i) \approx y_i$ for every $i = 1, \ldots, n$

Q: why do we want a model linear in the parameters w?

Linear models

To fit a linear model (= linear in parameters w)

- pick a transformation $\phi : \mathcal{X} \to \mathbf{R}^d$
- predict y using a linear function of $\phi(x)$

$$h(x) = w^T \phi(x) = \sum_{i=1}^{d} w_i (\phi(x))_i$$

- we want $h(x_i) \approx y_i$ for every $i = 1, \ldots, n$

Q: why do we want a model linear in the parameters w?
A: because the optimization problems are easy to solve!
e.g., just use least squares.

Feature engineering

How to pick $\phi : \mathcal{X} \to \mathbf{R}^d$?

- so response y will depend linearly on $\phi(x)$
- so d is not too big

Feature engineering

How to pick $\phi : \mathcal{X} \to \mathbf{R}^d$?

- so response y will depend linearly on $\phi(x)$
- so d is not too big

if you think this looks like a hack: you're right

Feature engineering

examples:

- adding offset
- standardizing features
- polynomial fits
- products of features
- autoregressive models
- local linear regression
- transforming Booleans
- transforming ordinals
- transforming nominals
- transforming images
- transforming text
- concatenating data
- all of the above

`https://xkcd.com/2048/`

Outline

Feature engineering

Polynomial transformations

Boolean, nominal, ordinal

Missing values

Nonlinear transformations, location

Text, images, ...

Time series

Adding offset

- $\mathcal{X} = \mathbf{R}^{d-1}$
- let $\phi(x) = (x, 1)$
- now $h(x) = w^T \phi(x) = w_{1:d-1}^T x + w_d$

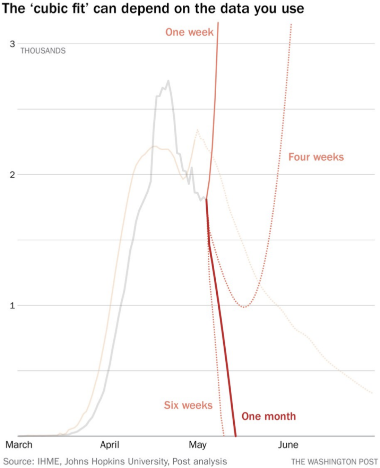
Fitting a polynomial

- $\mathcal{X} = \mathbf{R}$
- let
$$\phi(x) = (1, x, x^2, x^3, \dots, x^{d-1})$$
be the vector of all monomials in x of degree $< d$
- now $h(x) = w^T \phi(x) = w_1 + w_2 x + w_3 x^2 + \cdots + w_d x^{d-1}$

Demo: Linear models

`https://github.com/ORIE4741/demos`

IMHE and the cubic fit



https://www.washingtonpost.com/politics/2020/05/05/white-houses-self-serving-approach-estimating-deadliness-

Fitting a multivariate polynomial

- $\mathcal{X} = \mathbf{R}^2$
- pick a maximum degree k
- let

  $$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3 \dots, x_2^k)$$

  be the vector of all monomials in x_1 and x_2 of degree $\leq k$
- now $h(x) = w^T \phi(x)$ can fit any polynomial of degree $\leq k$ in $\mathcal{X}$

Fitting a multivariate polynomial

- $\mathcal{X} = \mathbf{R}^2$
- pick a maximum degree k
- let

  $$\phi(x) = (1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3 \ldots, x_2^k)$$

  be the vector of all monomials in x_1 and x_2 of degree $\leq k$
- now $h(x) = w^T\phi(x)$ can fit any polynomial of degree $\leq k$ in $\mathcal{X}$

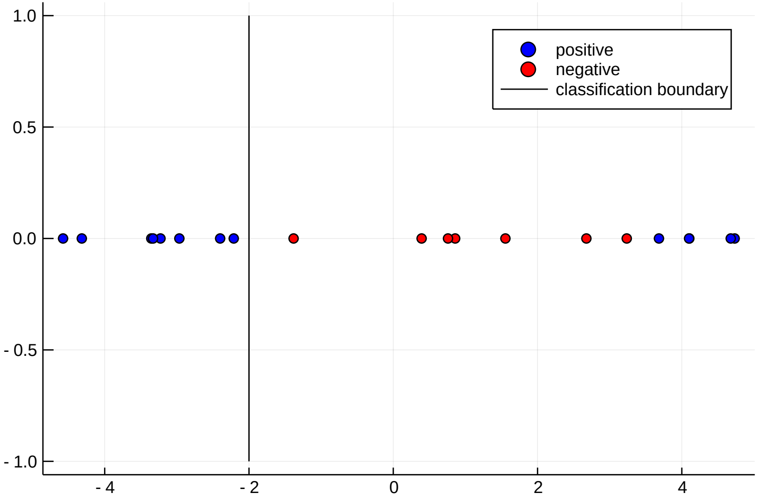
and similarly for $\mathcal{X} = \mathbf{R}^d \ldots$

Demo: Linear models

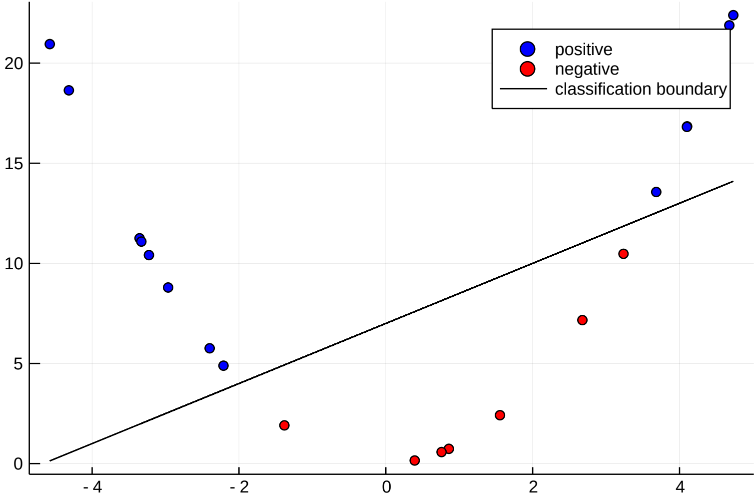
polynomial classification

`https://github.com/ORIE4741/demos`

Linear classification



Polynomial classification



Example 1: multivariate polynomial classification

- $\mathcal{X} = \mathbf{R}^2$, $\mathcal{Y} = \{-1, 1\}$
- let
$$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$
be the vector of all monomials of degree ≤ 2
- now let $h(x) = \mathbf{sign}(w^T \phi(x))$

Q: if $h(x) = \mathbf{sign}(-30 - 9x_1 + 2x_2 + x_1^2 + x_2^2)$, what is
$$\{x : h(x) = 1\}?$$

A. a circle
B. an ellipse
C. a line
D. a hyperbola
E. a half-plane

Example 2: multivariate polynomial classification

- $\mathcal{X} = \mathbf{R}^2$, $\mathcal{Y} = \{-1, 1\}$
- let
$$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$
be the vector of all monomials of degree ≤ 2
- now let $h(x) = \mathbf{sign}(w^T \phi(x))$

Q: if $h(x) = \mathbf{sign}(-5 - 3x_1 + 2x_2 + x_1^2 - x_1 x_2 + 5x_2^2)$, what is
$$\{x : h(x) = 1\}?$$

A. a circle
B. an ellipse
C. a line
D. a hyperbola
E. a half-plane

Example 3: multivariate polynomial classification

- $\mathcal{X} = \mathbf{R}^2$, $\mathcal{Y} = \{-1, 1\}$
- let
$$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$
be the vector of all monomials of degree ≤ 2
- now let $h(x) = \mathbf{sign}(w^T \phi(x))$

Q: if $h(x) = \mathbf{sign}(-5 - 3x_1 + 2x_2 + x_1^2 - x_1 x_2 + 5x_2^2)$, what is
$$\{x : h(x) = 1\}?$$

A. a circle
B. an ellipse
C. a line
D. a hyperbola
E. a half-plane

Example 3: multivariate polynomial classification

- $\mathcal{X} = \mathbf{R}^2$, $\mathcal{Y} = \{-1, 1\}$
- let
  $$\phi(x) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$
  be the vector of all monomials of degree ≤ 2
- now let $h(x) = \mathbf{sign}(w^T \phi(x))$

Q: if $h(x) = \mathbf{sign}(-5 - 3x_1 + 2x_2 + x_1^2 - x_1 x_2 - 2x_2^2)$, what is

$$\{x : h(x) = 1\}?$$

A. a circle
B. an ellipse
C. a line
D. a hyperbola
E. a half-plane

Outline

Feature engineering

Polynomial transformations

Boolean, nominal, ordinal

Missing values

Nonlinear transformations, location

Text, images, ...

Time series

Notation: boolean indicator function

define

$$\mathbb{1}(\text{statement}) = \begin{cases} 1 & \text{statement is true} \\ 0 & \text{statement is false} \end{cases}$$

examples:

- $\mathbb{1}(1 < 0) = 0$
- $\mathbb{1}(17 = 17) = 1$

Boolean variables

- $\mathcal{X} = \{\text{true}, \text{false}\}$
- let $\phi(x) = \mathbb{1}(x)$

Boolean expressions

- $\mathcal{X} = \{\text{true}, \text{false}\}^2 = \{(\text{true}, \text{true}), (\text{true}, \text{false}), (\text{false}, \text{true}), (\text{false}, \text{false})\}$.
- let $\phi(x) = [\mathbb{1}(x_1), \mathbb{1}(x_2), \mathbb{1}(x_1 \text{ and } x_2), \mathbb{1}(x_1 \text{ or } x_2)]$
- equivalent: polynomials in $[\mathbb{1}(x_1), \mathbb{1}(x_2)]$ span the same space
- encodes logical expressions!

Nominal values: one-hot encoding

- nominal data: *e.g.*, $\mathcal{X} = \{\text{apple}, \text{orange}, \text{banana}\}$
- let

$$\phi(x) = [\mathbb{1}(x = \text{apple}), \mathbb{1}(x = \text{orange}), \mathbb{1}(x = \text{banana})]$$

- called **one-hot encoding**: only one element is non-zero

Nominal values: one-hot encoding

- nominal data: *e.g.*, $\mathcal{X} = \{\text{apple}, \text{orange}, \text{banana}\}$
- let

$$\phi(x) = [\mathbb{1}(x = \text{apple}), \mathbb{1}(x = \text{orange}), \mathbb{1}(x = \text{banana})]$$

- called **one-hot encoding**: only one element is non-zero

extension: sets

Nominal values: the long tail

- **problem:** too many nominal categories
- **solution:**

Nominal values: the long tail

- **problem:** too many nominal categories
- **solution:**
  - cluster the categories by some known ontology (eg, "squamous cell carcinoma" $\rightarrow$ "cancer")

Nominal values: the long tail

- **problem:** too many nominal categories
- **solution:**
  - cluster the categories by some known ontology (eg, "squamous cell carcinoma" $\rightarrow$ "cancer")
  - lump the least common categories into a single category: "Other"

Nominal values: the long tail

- **problem:** too many nominal categories
- **solution:**
  - cluster the categories by some known ontology (eg, "squamous cell carcinoma" $\rightarrow$ "cancer")
  - lump the least common categories into a single category: "Other"
  - feature hashing

Nominal values: the long tail

- **problem:** too many nominal categories
- **solution:**
  - cluster the categories by some known ontology (eg, "squamous cell carcinoma" $\rightarrow$ "cancer")
  - lump the least common categories into a single category: "Other"
  - feature hashing
  - ... be creative!

Nominal values: look up features!

why not use other information known about each item?

- $\mathcal{X} = \{\text{apple}, \text{orange}, \text{banana}\}$
  - price, calories, weight, ...
- $\mathcal{X} =$ zip code
  - average income, temperature in July, walk score, % residential, ...
- ...

database lingo: **join** tables on nominal value

Ordinal values: real encoding

- ordinal data: *e.g.*,
  $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- let

$$\phi(x) = \begin{cases} 1, & x = \text{Stage I} \\ 2, & x = \text{Stage II} \\ 3, & x = \text{Stage III} \\ 4, & x = \text{Stage IV} \end{cases}$$

- default encoding

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

A. $b = 6$, $w = -2$

B. $b = 2$, $w = 0$

C. $b = 6$, $w = 2$

D. $b = 0$, $w = -2$

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years: $b = 6$, $w = -2$.

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years: $b = 6$, $w = -2$.

Q: How long does the model predict a persion with Stage IV cancer will survive?

Ordinal values: real encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- use real encoding ϕ to transform ordinal data
- fit linear model with offset to predict y as $w\phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years: $b = 6$, $w = -2$.

Q: How long does the model predict a persion with Stage IV cancer will survive?

A. 6 years
B. 2 years
C. 0 years
D. -2 years

Ordinal values: boolean encoding

- ordinal data: *e.g.*,
  $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- let

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

Ordinal values: boolean encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- define transformation $\phi : \mathcal{X} \to \mathbf{R}$ as

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

- fit linear model with offset to predict y as $w^\top \phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Ordinal values: boolean encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- define transformation $\phi : \mathcal{X} \to \mathbf{R}$ as

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

- fit linear model with offset to predict y as $w^\top \phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

Ordinal values: boolean encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- define transformation $\phi : \mathcal{X} \to \mathbf{R}$ as

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

- fit linear model with offset to predict y as $w^\top \phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

A: $b = 4$, $w_1 = -2$, w_2 and w_3 not determined

Ordinal values: boolean encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- define transformation $\phi : \mathcal{X} \to \mathbf{R}$ as

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

- fit linear model with offset to predict y as $w^\top \phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

A: $b = 4$, $w_1 = -2$, w_2 and w_3 not determined

Q: How long does the model predict a persion with Stage IV cancer will survive?

Ordinal values: boolean encoding

- $\mathcal{X} = \{\text{Stage I}, \text{Stage II}, \text{Stage III}, \text{Stage IV}\}$
- $\mathcal{Y} = \mathbf{R}$, number of years lived after diagnosis
- define transformation $\phi : \mathcal{X} \to \mathbf{R}$ as

$$\phi(x) = [\mathbb{1}(x \geq \text{Stage II}), \mathbb{1}(x \geq \text{Stage III}), \mathbb{1}(x \geq \text{Stage IV})]$$

- fit linear model with offset to predict y as $w^\top \phi(x) + b$

Suppose model predicts a person diagnosed with Stage II cancer will survive 2 more years, and a person diagnosed with Stage I cancer will survive 4 more years.

Q: What is w? b?

A: $b = 4$, $w_1 = -2$, w_2 and w_3 not determined

Q: How long does the model predict a persion with Stage IV cancer will survive?

A: can't say without more information

Outline

Missing values

handling missing values:

- remove rows/columns with missing entries

Missing values

handling missing values:

- remove rows/columns with missing entries
- (for time series) back-fill with most recent observed value

Missing values

handling missing values:

- remove rows/columns with missing entries
- (for time series) back-fill with most recent observed value
- impute with mean, median, or mode

Missing values

handling missing values:

- remove rows/columns with missing entries
- (for time series) back-fill with most recent observed value
- impute with mean, median, or mode
- fancier imputation methods (covered later in this class): matrix completion, copula models, deep learning, ...

Missing values

handling missing values:

- remove rows/columns with missing entries
- (for time series) back-fill with most recent observed value
- impute with mean, median, or mode
- fancier imputation methods (covered later in this class): matrix completion, copula models, deep learning, ...
- add new feature: Boolean indicator $\mathbb{1}$(data is missing)
  - can detect if missingness is informative
  - can complement imputation method
  - can use different indicators for different kinds of missingness (refused, missing, illegible response, ...)

Poll

In an ambulance dataset (data taken by instruments on board an ambulance), we want to predict if the patient died. The variable "heart rate" is sometimes missing. Is missingness

A. informative?

B. uninformative?

Poll

In a weather dataset, the batteries in the instruments occasionally run out before the experimenter can replace them, leaving missing data for eg temperature, humidity, or barometric pressure. Is missingness

A. informative?

B. uninformative?

Talk to your neighbor

Can you think of a dataset in which missing values would be

- informative?
- uninformative?

Outline

Nonlinear transformations

sometimes data is easy to predict with a simple but **non**linear relation, e.g.

$$\log(y) = w^T x$$

can transform x or (even more important) y

Nonlinear transformations

sometimes data is easy to predict with a simple but **non**linear relation, e.g.

$$\log(y) = w^T x$$

can transform x or (even more important) y

hints that your data might benefit from a nonlinear transform:

- y is positive and heavy-tailed? try $y \leftarrow \log(y)$
- residuals $r = y - w^T x_i$ are skewed (not normal)
  - check with quantile-quantile plot (see ORIE 3120 slides)

Nonlinear transformations

sometimes data is easy to predict with a simple but **non**linear relation, e.g.

$$\log(y) = w^T x$$

can transform x or (even more important) y

hints that your data might benefit from a nonlinear transform:

- y is positive and heavy-tailed? try $y \leftarrow \log(y)$
- residuals $r = y - w^T x_i$ are skewed (not normal)
  - check with quantile-quantile plot (see ORIE 3120 slides)

useful nonlinear transforms:

- log, exp, quantile, ...

more systematic ways to handle nonlinearities:
copula models, deep learning

Location

can be given as

- latitude, longitude
- zip code
- neighborhood, county, state, country

can be transformed between these!

Location

can be given as

- latitude, longitude
- zip code
- neighborhood, county, state, country

can be transformed between these!

which makes sense for your problem?

- does nearness matter?
- are there sharp boundaries?
- are other properties of the location (eg, mean house price or crime rate) more important?

Demo

stop, question, frisk:
`https://github.com/ORIE4741/demos/blob/master/feature_engineering.ipynb`

Outline

Text

$\mathcal{X}$ = sentences, documents, tweets, ...

- **bag of words** model (one-hot encoding):
  - pick set of words $\{w_1, \ldots, w_d\}$
  - $\phi(x) = [\mathbb{1}(x \text{ contains } w_1), \ldots, \mathbb{1}(x \text{ contains } w_d)]$
  - ignores order of words in sentence

Text

$\mathcal{X}$ = sentences, documents, tweets, ...

- **bag of words** model (one-hot encoding):
  - pick set of words $\{w_1, \ldots, w_d\}$
  - $\phi(x) = [\mathbb{1}(x \text{ contains } w_1), \ldots, \mathbb{1}(x \text{ contains } w_d)]$
  - ignores order of words in sentence
- **pre-trained neural networks**:
  - sentiment analysis: `https://medium.com/@b.terryjack/nlp-pre-trained-sentiment-analysis-1eb52a9d742c`
  - Universal Sentence Encoder (USE) embedding: `https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/semantic_similarity_with_tf_hub_universal_encoder.ipynb`
  - lots of others: `https://modelzoo.co/`

Neural networks: whirlwind primer

$$\mathrm{NN}(x) = \sigma(W_1\sigma(W_2 \ldots \sigma(W_\ell x))))$$

- σ is a nonlinearity applied elementwise to a vector, *e.g.*
  - ReLU: $\sigma(x) = \max(x, 0)$
  - sigmoid: $\sigma(x) = \log(1 + \exp(x))$
- each W is a matrix
- trained on very large datasets, *e.g.*, Wikipedia, YouTube

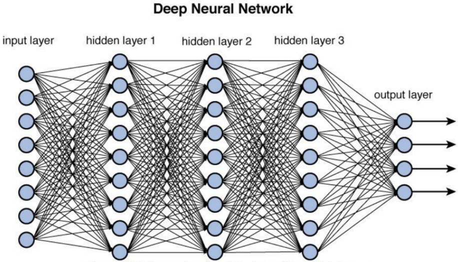
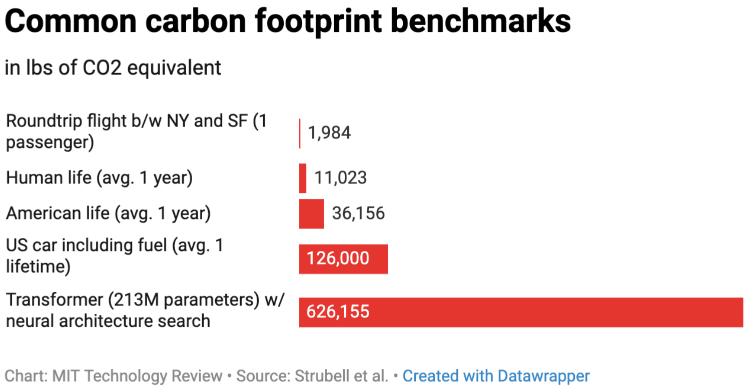


Figure 12.2 Deep network architecture with multiple layers.

Why not use deep learning?



towards a solution: `https://arxiv.org/abs/1907.10597`

Outline

Time series

- given a time series $a_t \in \mathbf{R}$, $t = 1, \ldots, T$
- for each time t, we want to predict the value at the next time $t+1$

Time series

- given a time series $a_t \in \mathbf{R}$, $t = 1, \ldots, T$
- for each time t, we want to predict the value at the next time $t + 1$

Time series

- given a time series $a_t \in \mathbf{R}$, $t = 1, \ldots, T$
- for each time t, we want to predict the value at the next time $t + 1$

Q: what is input space $\mathcal{X}$? output space $\mathcal{Y}$?

Time series

- given a time series $a_t \in \mathbf{R}$, $t = 1, \ldots, T$
- for each time t, we want to predict the value at the next time $t + 1$

Q: what is input space $\mathcal{X}$? output space $\mathcal{Y}$?

- input is time series $a_{1:t}$ up to time t.
  input space is $\mathcal{X} = \mathbf{R}^t$.
- output is the prediction $\hat{a}_{t+1}$ at the next time.
  output space is $\mathcal{Y} = \mathbf{R}$.

Auto-regressive (AR) model

- for each time t, let

$$\phi(t, x) = (x_{t-1}, x_{t-2}, \ldots, x_{t-d})$$

(called the "lagged outcomes")

- now $h(x) = w^T \phi(x) = w_1 x_{t-1} + w_2 x_{t-2} + \cdots + w_d x^{t-d}$

AR moving average (ARMA) model

idea: view current value a_t as linear combination of

- p most recent observations $a_{t-1}, a_{t-2}, \ldots, a_{t-p}$
- q past residuals,
  $r_{t-1}, r_{t-2}, \ldots, r_{t-p} = \hat{a}_{t-1} - a_{t-1}, \ldots, \hat{a}_{t-p} - a_{t-p}$
- current residual (noise) $r_t = \hat{a}_t - a_t$

AR moving average (ARMA) model

idea: view current value a_t as linear combination of

- p most recent observations $a_{t-1}, a_{t-2}, \ldots, a_{t-p}$
- q past residuals,
  $r_{t-1}, r_{t-2}, \ldots, r_{t-p} = \hat{a}_{t-1} - a_{t-1}, \ldots, \hat{a}_{t-p} - a_{t-p}$
- current residual (noise) $r_t = \hat{a}_t - a_t$

Definition

A time series $\{a_t\}$ is ARMA(p, q) if it is stationary and the prediction is of the form

$$\hat{a}_t = \sum_{i=1}^{p} w_i a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j}.$$

parameters: $w \in \mathbf{R}^p$ and $\theta \in \mathbf{R}^q$

recall time series is **stationary** if a_t is independent of t

ARMA model as linear regression

- ARMA prediction is a linear function of $a_{t-1}, a_{t-2}, \ldots, a_{t-p}, r_{t-1}, r_{t-2}, \ldots, r_{t-p}$.
- ARMA model predicts the current value a_t as

$$h(a_{1:t-1}; t) = \sum_{i=1}^{p} w_i a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j}$$

- we want $\hat{a}_t = h(a_{1:t-1}; t) \approx a_t$.

ARMA model as linear regression

- ARMA prediction is a linear function of $a_{t-1}, a_{t-2}, \dots, a_{t-p}, r_{t-1}, r_{t-2}, \dots, r_{t-p}$.
- ARMA model predicts the current value a_t as

$$h(a_{1:t-1}; t) = \sum_{i=1}^{p} w_i a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j}$$

- we want $\hat{a}_t = h(a_{1:t-1}; t) \approx a_t$.

poll: can we fit ARMA model with one linear regression?

A. yes

B. no

Least squares fitting

for $t = 1, \ldots, T$

- define residual at time t

$$r_t = a_t - \hat{a}_t = a_t - h(a_{1:t-1}; t)$$

- choose w, θ to minimize squared residuals

$$\sum_{t=1}^{T} r_t^2 = \sum_{t=1}^{T} (\sum_{i=1}^{p} w_i a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j} - a_t)^2$$

Least squares fitting

for $t = 1, \ldots, T$

- define residual at time t

$$r_t = a_t - \hat{a}_t = a_t - h(a_{1:t-1}; t)$$

- choose w, θ to minimize squared residuals

$$\sum_{t=1}^{T} r_t^2 = \sum_{t=1}^{T} (\sum_{i=1}^{p} w_i a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j} - a_t)^2$$

sequential solves are a lot of work! with QR, complexity is $O(T^2 d^2)$ where $d = p + q$

Speeding it up: online GD

let's use online gradient descent (similar to SGD):

- let w^t and θ^t denote the parameter estimated at time t
- w^t : each parameter w_i estimated at time t is an entry of w^t.
- θ^t : each parameter θ_j estimated at time t is an entry of θ^t.

Online gradient descent:

- initialize w^0, θ^0
- for $t = 1 : T$,

$$w_i^t = w_i^{t-1} - 2\alpha \sum_{t'=1}^{t} r_{t'} a_{t'-i} \quad \text{for } i = 1, \ldots, p$$

$$\theta_j^t = \theta_j^{t-1} - 2\alpha \sum_{t'=1}^{t} r_{t'} \theta_i^{t'-1} \quad \text{for } j = 1, \ldots, q$$

From ARMA to ARIMA

idea:

- sometimes $\{a_t\}$ is non-stationary, but the difference is stationary.
  - **example:** random walk $a_t = a_{t-1} + r_t$ where $r_t \sim \mathcal{N}(0, 1)$. $\{a_t\}$ is not stationary, but $\{r_t : r_t = a_t - a_{t-1}\}$ is stationary and ARMA.

From ARMA to ARIMA

idea:

- sometimes $\{a_t\}$ is non-stationary, but the difference is stationary.
  - **example:** random walk $a_t = a_{t-1} + r_t$ where $r_t \sim \mathcal{N}(0,1)$. $\{a_t\}$ is not stationary, but $\{r_t : r_t = a_t - a_{t-1}\}$ is stationary and ARMA.
- define backshift operator B such that $Ba_t = a_{t-1}$. Then

$$
\begin{aligned}
a_t - a_{t-1} &= (1-B)a_t \\
a_t - a_{t-2} &= (1-B)^2 a_t \\
&\vdots \\
a_t - a_{t-d} &= (1-B)^d a_t
\end{aligned}
$$

ARIMA

Definition

A process a_t is ARIMA(p, d, q) if

$$(1 - B)^d a_t$$

is ARMA(p, q).

ARIMA(p, d, q) model can be written as

$$(1 - B)^d \hat{a}_t = r_t + \sum_{i=1}^{p} w_i (1 - B)^d a_{t-i} + \sum_{j=1}^{q} \theta_j r_{t-j}.$$

Exponential smoothing

- Forecasts are calculated using weighted averages;
- The weights decrease exponentially as observations come from further in the past;
- The smallest weights are associated with the oldest observations;

$$\hat{a}_t = \alpha a_{t-1} + \alpha(1-\alpha) a_{t-2} + \cdots \alpha(1-\alpha)^{t-1} a_1$$

Holt winter: forecasting with trend

For now, assume data has trend but no seasonality.

Holt's forecasting method uses a linear trend

$$\text{estimate at time } t \text{ of } a_{t+l} := \hat{a}_t(l) = \hat{c}_t + \hat{b}_t \ell$$

- t is "origin" — the point in time when forecasts are being made
- ℓ is the "lead" — how far ahead one is forecasting
- $\hat{c}_t$ is called the level
- $\hat{b}_t$ is called the slope.

Both $\hat{c}_t$ and $\hat{b}_t$ are updated as we make more observations t.

Holt winter: updating the level

In the Holt model, the level $\hat{c}_t$ is updated by the equation:

$$\hat{c}_{t+1} = (1 - \alpha)(\hat{c}_t + \hat{b}_t) + \alpha a_{t+1}$$

or equivalently,

$$\hat{c}_{t+1} = \hat{c}_t + (1 - \alpha)\hat{b}_t + \alpha(a_{t+1} - \hat{c}_t)$$

- β is for updating the level.

Holt winter: updating the slope

In the Holt model, the slope $\hat{b}_t$ is updated by the equation:

$$\hat{b}_{t+1} = (1-\beta)\hat{b}_t + \beta(\hat{c}_{t+1} - \hat{c}_t)$$

or equivalently,

$$\hat{c}_{n+1} = \hat{c}_t + \beta\left\{(\hat{c}_{t+1} - \hat{c}_t) - \hat{b}_t\right\}$$

- $\hat{c}_t + \hat{b}_t$ is predicted value at time $t+1 =$ lead time 1
- α is for updating the level.

Holt-Winters additive and multiplicative seasonal method

The forecasts are periodic.

- Additive methods:

$$\begin{aligned}\hat{a}_t(l) &= \hat{c}_t + \hat{b}_t\ell + \hat{S}_{n+\ell-s} \quad \text{for } \ell = 1, 2, \ldots, s \\ &= \hat{c}_t + \hat{b}_t\ell + \hat{S}_{n+\ell-2s} \quad \text{for } \ell = s+1, s+2, \ldots, 2s\end{aligned}$$

- Multiplicative methods:

$$\begin{aligned}\hat{a}_t(l) &= (\hat{c}_t + \hat{b}_t\ell)\hat{S}_{n+\ell-s} \quad \text{for } \ell = 1, 2, \ldots, s \\ &= (\hat{c}_t + \hat{b}_t\ell) + \hat{S}_{n+\ell-2s} \quad \text{for } \ell = s+1, s+2, \ldots, 2s\end{aligned}$$