

MATH (160)

Mathematics for Data Analytics and Decision Making

Jesus De Loera

UC Davis, Mathematics

Monday, March 28, 2011

Syllabus, Class Schedule, Office Hours, Textbook, etc.

See course website.

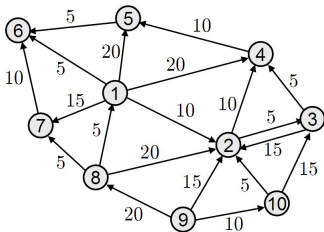
<https://www.math.ucdavis.edu/~deloera/TEACHING/MATH160/>

Also I will use your UC Davis email for announcements and quizzes!

This course is about...Thinking before deciding!!

MATHEMATICAL MODELS!!!
That use data to make intelligent
decisions!!

- How does Google's search engine and GPS routing work?




- The engine behind the engines is all MATHEMATICS!!!

Today the world has abundant data of all sorts, we need to go FROM DATA TO DECISIONS!

Big Data Shows Flu Spread Factors, Including Weather and Geography

Tue, 02/27/2018 - 10:03am By [Seth Augenstein](#) - Senior Science Writer - [@SethAug](#)






National Science Foundation
WHERE DISCOVERIES BEGIN

NSF Web Site

Home Funding Awards Discoveries **News** Publications Statistics About FastLane

News



[News](#)
[News From the Field](#)
[For the News Media](#)
[Special Reports](#)
[Research Overviews](#)
[NSF-Wide Investments](#)
[Speeches & Lectures](#)
[NSF Current Newsletter](#)
[Multimedia Gallery](#)
[News Archive](#)


News by Research Area
[Arctic & Antarctic](#)
[Astronomy & Space](#)
[Biology](#)
[Chemistry & Materials](#)
[Computing](#)
[Earth & Environment](#)

All Images

Press Release 10-029
Fighting Crime With Math

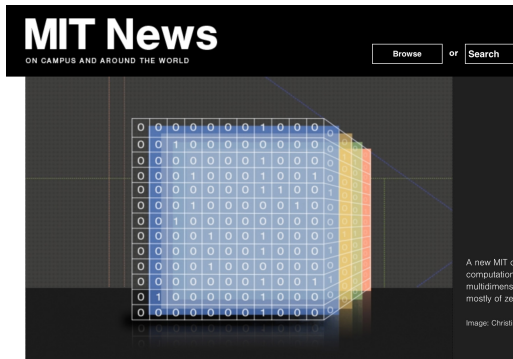
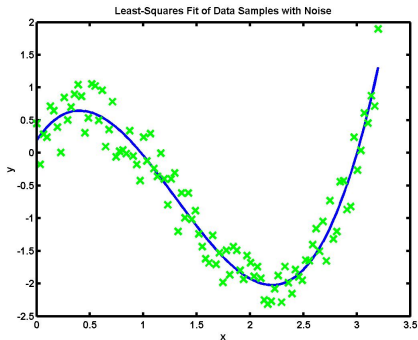
Sophisticated math models give insights on crime hotspots
[Back to article](#) | [Note about images](#)

Video



Why does police presence extinguish some crime surges, but just move others? Researchers teamed up with the LAPD to model the math behind spikes in crime-- and what kind of policing works.

and MATH is the only way we can analyze it to extract relevant information.



Faster big-data analysis

System for performing “tensor algebra” offers 100-fold speedups over software packages.

By the end of this course, you should make BETTER
(quantitative) DECISIONS!!

By the end of this course, you should make BETTER (quantitative) DECISIONS!!

MULTITRANS is a fictional bus company that is run by a student association in an unnamed college town with partial funding from the city.

(It started out in 1972 with a fleet of historic triple-deck buses from Moscow, but has added more modern buses since.)

During most of the day, though, all scheduled buses are completely empty.

A scientific study was conducted in 2010 to find out why this is the case. The result of the study was:

[...] 85% of the representative sample of 1256 potential users of MULTITRANS replied that “I would use it regularly instead of using my bike or car, but the MULTITRANS schedule [is suboptimal].” [...]

MULTITRANS hires YOU as a consultant to help them improve the schedule.

What are the next steps you need to take to solve the challenge?

Part I of the course:

LINEAR ALGEBRA
MODELS

Least-squares applications to modeling.

What is the best team? What is the best candidate? etc: The science of ranking

- Say we are given data about teams with game scores, how could we predict the best team of the season?
- Say we are given voters information about candidates, how can we rank them? (arrange them in order of importance).
- Say we are given a list of movies with user preferences (prefer to star wars v.s. star trek?)

Goal

We will see five ways to make predictions of WHO IS NUMBER ONE using data of comparisons and solving linear equations.

The models depend on how much information we have on the situation. We hope that using these five methods allow users to determine who is the most viable candidate and show the user that different methods may provide different results.

Two old “COUNTING” methods

Plurality

Candidate rated number one is the one with the most wins or the most votes.

The plurality method was originally used in determining a winner in voting systems. It soon became one of the most commonly used methods for selecting a winner. But what if we knew an ordered list of candidates, one for each voter? We could use more sophisticated methods!! Borda Count is a method that dates back to 1770

Borda Count

Count the number of times a candidate appears in i th position and apply the following set of weights as follows (here n is the total number of candidates)
 $(n - 1)$ points are given to first place, $(n - k)$ to k -th place, and 0 is given to last place. The tallies are summed and the winner is the candidate with the highest sum.

Voter	HC	BS	JK	TC	DT
Voter 1	5	4	3	2	1
Voter 2	5	4	3	2	1
Voter 3	4	5	3	1	2
Voter 4	2	5	1	3	4
Voter 5	3	5	2	1	4
Voter 6	4	5	2	1	3
Voter 7	1	5	3	4	2
Voter 8	5	4	3	1	2
Voter 9	4	5	2	3	1
Voter 10	4	5	2	1	3

Table: Data set (only 10 voters).

1st Place votes	HC	BS	JK	TC	DT
Votes	3	7	0	0	0

Table: Plurality Scores.

	HC	BS	JK	TC	DT
Average Rating	3.7	4.7	2.4	1.9	2.3

Table: Average Scores.

Ranking	HC	BS	JK	TC	DT
1st Place	3	7	0	0	0
2nd Place	4	3	0	1	1
3rd Place	1	0	5	2	2
4th Place	1	0	4	2	4
5th Place	1	0	1	5	3

Table: Borda Rankings.

Candidate	Score
Hillary Clinton	27
Bernie Sanders	37
John Kasich	14
Ted Cruz	9
Donald Trump	11

Table: Borda Scores.

Massey Ranking

- The fundamental idea: Find the rankings or score values of team r_i by noting $r_i - r_j = y_{ij}$ where y_{ij} is the margin of victory of team i over j .
- For each game played there is one such equation, and then there is a system of linear equations. The matrix X is $m \times n$ with m = number of games played and n = number of teams.

$$Xr = y.$$

Massey Ranking

- The fundamental idea: Find the rankings or score values of team r_i by noting $r_i - r_j = y_{ij}$ where y_{ij} is the margin of victory of team i over j .
- For each game played there is one such equation, and then there is a system of linear equations. The matrix X is $m \times n$ with m = number of games played and n = number of teams.

$$Xr = y.$$

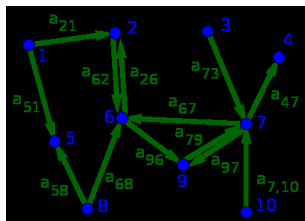
- Each row of the matrix X is nearly all zeros, except a 1 and a -1 .

Massey Ranking

- The fundamental idea: Find the rankings or score values of team r_i by noting $r_i - r_j = y_{ij}$ where y_{ij} is the margin of victory of team i over j .
- For each game played there is one such equation, and then there is a system of linear equations. The matrix X is $m \times n$ with m = number of games played and n = number of teams.

$$Xr = y.$$

- Each row of the matrix X is nearly all zeros, except a 1 and a -1 .
- It is the transpose of a **node-arc incidence matrix of a network** Typically the number of games is very large, larger than the number of teams.



- Note: y_{ij} can be interpreted in many ways: can mean traffic levels, to rank websites, y_{ij} can be number of in-links or outlinks.

- We may not have an exact solution of $Xr = y$, but we can solve the associated **least-squares problem**.

- We may not have an exact solution of $Xr = y$, but we can solve the associated **least-squares problem**.
- Recall, solving the least squares problem $Xr = y$ same as solving the system of equations: $X^T Xr = Mr = X^T y = d$.

$$\begin{pmatrix} t_1 & -m_{12} & \cdots & -m_{1n} \\ -m_{21} & t_2 & \cdots & -m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & -m_{n2} & \cdots & t_n \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}$$

- We may not have an exact solution of $Xr = y$, but we can solve the associated **least-squares problem**.
- Recall, solving the least squares problem $Xr = y$ same as solving the system of equations: $X^T Xr = M r = X^T y = d$.

$$\begin{pmatrix} t_1 & -m_{12} & \cdots & -m_{1n} \\ -m_{21} & t_2 & \cdots & -m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & -m_{n2} & \cdots & t_n \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}$$

- Here t_i is the total number of games, m_{ij} is the number of games team i played against j and d_i is the score differential of the team.

- We may not have an exact solution of $Xr = y$, but we can solve the associated **least-squares problem**.
- Recall, solving the least squares problem $Xr = y$ same as solving the system of equations: $X^T Xr = Mx = X^T y = d$.

$$\begin{pmatrix} t_1 & -m_{12} & \cdots & -m_{1n} \\ -m_{21} & t_2 & \cdots & -m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & -m_{n2} & \cdots & t_n \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}$$

- Here t_i is the total number of games, m_{ij} is the number of games team i played against j and d_i is the score differential of the team.
- The elements of \vec{d} are calculated as follows:

$$d_i = \sum_{j=1}^x (n_i - n_j) = (n_i - n_1) + (n_i - n_2) + \cdots + (n_i - n_x)$$

- We may not have an exact solution of $Xr = y$, but we can solve the associated **least-squares problem**.
- Recall, solving the least squares problem $Xr = y$ same as solving the system of equations: $X^T Xr = M r = X^T y = d$.

$$\begin{pmatrix} t_1 & -m_{12} & \cdots & -m_{1n} \\ -m_{21} & t_2 & \cdots & -m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & -m_{n2} & \cdots & t_n \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}$$

- Here t_i is the total number of games, m_{ij} is the number of games team i played against j and d_i is the score differential of the team.
- The elements of \vec{d} are calculated as follows:

$$d_i = \sum_{j=1}^x (n_i - n_j) = (n_i - n_1) + (n_i - n_2) + \cdots + (n_i - n_x)$$

- Outputs the ratings of candidates by approximately solving the system of linear equalities (or least-squares!). We are trying to find the ratings r_i .

	HC	BS	JK	TC	DT
HC	40	-10	-10	-10	-10
BS	-10	40	-10	-10	-10
Jk	-10	-10	40	-10	-10
TC	-10	-10	-10	40	-10
DT	-10	-10	-10	-10	40

Table: Massey Matrix.

	HC	BS	JK	TC	DT
HC	40	-10	-10	-10	-10
BS	-10	40	-10	-10	-10
JK	-10	-10	40	-10	-10
TC	-10	-10	-10	40	-10
DT	-10	-10	-10	-10	40

Table: Massey Matrix.

Games	HC	BS	JK	TC	DT
1 - 4 (V_1)	10	5	0	-5	-10
5 - 8 (V_2)	10	5	0	-5	-10
9 - 12 (V_3)	5	10	0	-10	-5
13 - 16 (V_4)	-5	10	-10	0	5
17 - 20 (V_5)	0	10	-5	-10	5
21 - 24 (V_6)	5	10	-5	-10	0
25 - 28 (V_7)	-10	10	0	5	-5
29 - 32 (V_8)	10	5	0	-10	-5
33 - 36 (V_9)	5	10	-5	0	-10
37 - 40 (V_{10})	5	10	-5	-10	0
Total	35	85	-30	-55	-35

Table: Candidates' Point Differentials: how many times candidate won or lost to another

	HC	BS	JK	TC	DT
HC	40	-10	-10	-10	-10
BS	-10	40	-10	-10	-10
JK	-10	-10	40	-10	-10
TC	-10	-10	-10	40	-10
DT	-10	-10	-10	-10	40

Table: Massey Matrix.

Games	HC	BS	JK	TC	DT
1 - 4 (V_1)	10	5	0	-5	-10
5 - 8 (V_2)	10	5	0	-5	-10
9 - 12 (V_3)	5	10	0	-10	-5
13 - 16 (V_4)	-5	10	-10	0	5
17 - 20 (V_5)	0	10	-5	-10	5
21 - 24 (V_6)	5	10	-5	-10	0
25 - 28 (V_7)	-10	10	0	5	-5
29 - 32 (V_8)	10	5	0	-10	-5
33 - 36 (V_9)	5	10	-5	0	-10
37 - 40 (V_{10})	5	10	-5	-10	0
Total	35	85	-30	-55	-35

Table: Candidates' Point Differentials: how many times candidate won or lost to another

There is a problem with M : it is not full rank. Without a full rank we will not be able to find a unique solution.

Therefore we replace the entire last row of Matrix M with 1's and replace the last element of \vec{p} with a 0. This yields the system

There is a problem with M : it is not full rank. Without a full rank we will not be able to find a unique solution.

Therefore we replace the entire last row of Matrix M with 1's and replace the last element of \vec{p} with a 0. This yields the system

$$\begin{bmatrix} 40 & -10 & -10 & -10 & -10 \\ -10 & 40 & -10 & -10 & -10 \\ -10 & -10 & 40 & -10 & -10 \\ -10 & -10 & -10 & 40 & -10 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{bmatrix} = \begin{bmatrix} 35 \\ 85 \\ -30 \\ -55 \\ 0 \end{bmatrix}$$

The unique solution gives now a ranking

The unique solution gives now a ranking

$\vec{r} =$

$$\begin{bmatrix} 0.700 \\ 1.700 \\ -0.600 \\ -1.100 \\ -0.700 \end{bmatrix}$$

The largest rating of 1.700 corresponds to Bernie Sanders. Therefore, Bernie Sanders is the winner for Massey's Method.

The unique solution gives now a ranking

$\vec{r} =$

$$\begin{bmatrix} 0.700 \\ 1.700 \\ -0.600 \\ -1.100 \\ -0.700 \end{bmatrix}$$

The largest rating of 1.700 corresponds to Bernie Sanders. Therefore, Bernie Sanders is the winner for Massey's Method.

Rank	Candidate
1	Bernie Sanders
2	Hillary Clinton
3	John Kasich
4	Donald Trump
5	Ted Cruz

Table: Ranked Candidates.

Colley's method

- Colley's method (invented in 2001 by Astrophysicists Wesley Colley), looks very similar but comes from a different intuition!

Colley's method

- Colley's method (invented in 2001 by Astrophysicists Wesley Colley), looks very similar but comes from a different intuition!
- INITIAL IDEA: Winning percentage score of team $i = r_i = \frac{w_i}{t_i}$. Where w_i is number of wins, t_i number of games played.

Colley's method

- Colley's method (invented in 2001 by Astrophysicists Wesley Colley), looks very similar but comes from a different intuition!
- INITIAL IDEA: Winning percentage score of team $i = r_i = \frac{w_i}{t_i}$. Where w_i is number of wins, t_i number of games played.
- Problems: There are ties, the strength of an opponent is not represented (defeating a weak team not the same as defeating a champion).

Colley's method

- Colley's method (invented in 2001 by Astrophysicists Wesley Colley), looks very similar but comes from a different intuition!
- INITIAL IDEA: Winning percentage score of team $i = r_i = \frac{w_i}{t_i}$. Where w_i is number of wins, t_i number of games played.
- Problems: There are ties, the strength of an opponent is not represented (defeating a weak team not the same as defeating a champion).
- BETTER IDEA: Fix the problems with a small modification:

$$r_i = \frac{1 + w_i}{2 + t_i}$$

Colley's method

- Colley's method (invented in 2001 by Astrophysicists Wesley Colley), looks very similar but comes from a different intuition!
- INITIAL IDEA: Winning percentage score of team $i = r_i = \frac{w_i}{t_i}$. Where w_i is number of wins, t_i number of games played.
- Problems: There are ties, the strength of an opponent is not represented (defeating a weak team not the same as defeating a champion).
- BETTER IDEA: Fix the problems with a small modification:

$$r_i = \frac{1 + w_i}{2 + t_i}$$

- Intuitively all teams start with score $1/2$ at the beginning and things change. Denote l_i number of games lost by i .

$$w_i = \frac{w_i - l_i}{2} + \frac{w_i + l_i}{2} = \frac{w_i - l_i}{2} + \frac{t_i}{2} = \frac{w_i - l_i}{2} + \sum_{j=1}^{t_i} 1/2$$

Thus

$$\sum_{j=1}^{t_i} 1/2 \sim \sum_{j \text{ opponent of } i} r_j \quad \text{and} \quad w_i \sim \frac{w_i - l_i}{2}$$

Colley's method continued

- We get

$$r_i = \frac{1 + \frac{w_i - l_i}{2} + \sum r_j}{2 + t_i}$$

Colley's method continued

- We get

$$r_i = \frac{1 + \frac{w_i - l_i}{2} + \sum r_j}{2 + t_i}$$

- Rewriting in terms of linear equations and solving for the ratings r_k 's, we get a matrix C

$$C = \begin{pmatrix} t_1 + 2 & -n_{12} & \cdots & -n_{1n} \\ -n_{21} & t_2 + 2 & \cdots & -n_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -n_{n1} & -n_{n2} & \cdots & t_n + 2 \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Colley's method continued

- We get

$$r_i = \frac{1 + \frac{w_i - l_i}{2} + \sum r_j}{2 + t_i}$$

- Rewriting in terms of linear equations and solving for the ratings r_k 's, we get a matrix C

$$C = \begin{pmatrix} t_1 + 2 & -n_{12} & \cdots & -n_{1n} \\ -n_{21} & t_2 + 2 & \cdots & -n_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -n_{n1} & -n_{n2} & \cdots & t_n + 2 \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

- Where we are trying to find the rating r_i and t_i is the total number of games, $-n_{ij}$ is the number of games team i played against j and $b_i = 1 + \frac{1}{2}(w_i - l_i)$.

Colley's method continued

- We get

$$r_i = \frac{1 + \frac{w_i - l_i}{2} + \sum r_j}{2 + t_i}$$

- Rewriting in terms of linear equations and solving for the ratings r_k 's, we get a matrix C

$$C = \begin{pmatrix} t_1 + 2 & -n_{12} & \cdots & -n_{1n} \\ -n_{21} & t_2 + 2 & \cdots & -n_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -n_{n1} & -n_{n2} & \cdots & t_n + 2 \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

- Where we are trying to find the rating r_i and t_i is the total number of games, $-n_{ij}$ is the number of games team i played against j and $b_i = 1 + \frac{1}{2}(w_i - l_i)$.
- The matrix C is a real symmetric matrix and $Cr = b$ is the system we need to solve.
- In the example of candidates: Hillary Clinton played forty games in total: ten games against Bernie Sanders, ten games against John Kasich, ten games against Ted Cruz, and ten games against Donald Trump. Each voter rated five candidates which is equivalent to four games since each candidate was indirectly being compared to the other four candidates.

	HC	BS	JK	TC	DT
HC	42	-10	-10	-10	-10
BS	-10	42	-10	-10	-10
Jk	-10	-10	42	-10	-10
TC	-10	-10	-10	42	-10
DT	-10	-10	-10	-10	42

Table: C Matrix.

Games	HC	BS	JK	TC	DT
Won	27	37	14	9	13
Lost	13	3	26	31	27
Total	40	40	40	40	40

Table: Candidate's Scoreboard.

$$\begin{bmatrix} 42 & -10 & -10 & -10 & -10 \\ -10 & 42 & -10 & -10 & -10 \\ -10 & -10 & 42 & -10 & -10 \\ -10 & -10 & -10 & 42 & -10 \\ -10 & -10 & -10 & -10 & 42 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{bmatrix} = \begin{bmatrix} 8 \\ 18 \\ -5 \\ -10 \\ -6 \end{bmatrix}$$

After inputting all the matrices into MATLAB we use $\vec{r} = \text{linsolve}(C, \vec{b})$ to get:

$\vec{r} =$

$$\begin{bmatrix} 0.6346 \\ 0.8269 \\ 0.3846 \\ 0.2885 \\ 0.3654 \end{bmatrix}$$

Is the solution unique? Why?

The largest rating of 0.8269 corresponds to Bernie Sanders. Therefore, Bernie Sanders is the winner for Colley's method.

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

- We have y_i to denote the response, training or target variable that we are *trying to predict*.

Example: We wish to predict the price of the house.

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

- We have y_i to denote the response, training or target variable that we are *trying to predict*.

Example: We wish to predict the price of the house.

- A pair (x_i, y_i) is called a *training example*. The dataset that we use to learn a list of m training examples. $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ is called a *training set*. We will also use X denote the space of input values, and Y the space of output values.

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

- We have y_i to denote the response, training or target variable that we are *trying to predict*.

Example: We wish to predict the price of the house.

- A pair (x_i, y_i) is called a *training example*. The dataset that we use to learn a list of m training examples. $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ is called a *training set* We will also use X denote the space of input values, and Y the space of output values.
- **GOAL** Given a training set, to *learn* a function $h : X \rightarrow Y$ so that $h(x)$ is a “good” predictor for the corresponding value of y . This is the *supervised learning problem*

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

- We have y_i to denote the response, training or target variable that we are *trying to predict*.

Example: We wish to predict the price of the house.

- A pair (x_i, y_i) is called a *training example*. The dataset that we use to learn a list of m training examples. $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ is called a *training set* We will also use X denote the space of input values, and Y the space of output values.
- **GOAL** Given a training set, to *learn* a function $h : X \rightarrow Y$ so that $h(x)$ is a “good” predictor for the corresponding value of y . This is the *supervised learning problem*
- When target variables are continuous, we call the supervised learning a **regression problem**.
- When y can take on discrete values (e.g., want to predict if a dwelling is a house or an apartment), we call it a **classification problem**.

A classic application: Regression Analysis

- We have data points x_1, x_2, \dots, x_m each has n input variables x_{ij} , also called **features**.

Example: measurements such as living-space area, number of rooms, etc.

- We have y_i to denote the response, training or target variable that we are *trying to predict*.

Example: We wish to predict the price of the house.

- A pair (x_i, y_i) is called a *training example*. The dataset that we use to learn a list of m training examples. $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ is called a *training set* We will also use X denote the space of input values, and Y the space of output values.
- **GOAL** Given a training set, to *learn* a function $h : X \rightarrow Y$ so that $h(x)$ is a “good” predictor for the corresponding value of y . This is the *supervised learning problem*
- When target variables are continuous, we call the supervised learning a **regression problem**.
- When y can take on discrete values (e.g., want to predict if a dwelling is a house or an apartment), we call it a **classification problem**.
To perform supervised learning, we must decide how to represent functions/hypotheses h in a computer!

Basic Regression analysis

- LINEAR CHOICE: Approximate y as a linear function of x :

$$h(x) = \sum_k^n w_k x_{ik} = w^T x_i$$

Basic Regression analysis

- LINEAR CHOICE: Approximate y as a linear function of x :

$$h(x) = \sum_k^n w_k x_{ik} = w^T x_i$$

- Formally we wish to find a function that measures, for each value of the w , how close the $h(x_i)$'s are to the corresponding y_i 's.
We define the cost function that minimizes the error.

$$\min \sum_i^m (h(x_i) - y_i)^2 = \min_{w \in \mathbb{R}^m} \|X^T w - \hat{y}\|_2$$

- We use the notation X to denote the $m \times n$ matrix whose rows are the data points x_i . Let \hat{y} the row vector with y_i 's. This yields the traditional LEAST SQUARES PROBLEM:
- **Proposition** The optimal solution is the solution of a linear system of equations

$$X^T X w = X^T \hat{y}$$

Polynomial Regression

- We may add nonlinear relations on the features by considering a polynomial of degree M over n variables as the learning function:

$$h(x) = \sum_{a_1 + a_2 + \dots + a_n \leq M} c_{i_1, i_2, \dots, i_n} x_{i_1}^{a_1} x_{i_2}^{a_2} \dots x_{i_n}^{a_n}$$

Polynomial Regression

- We may add nonlinear relations on the features by considering a polynomial of degree M over n variables as the learning function:

$$h(x) = \sum_{a_1 + a_2 + \dots + a_n \leq M} c_{i_1, i_2, \dots, i_n} x_{i_1}^{a_1} x_{i_2}^{a_2} \dots x_{i_n}^{a_n}$$

- The monomials can be used to represent all such polynomials as a vector space basis (of which dimension??)

Polynomial Regression

- We may add nonlinear relations on the features by considering a polynomial of degree M over n variables as the learning function:

$$h(x) = \sum_{a_1+a_2+\dots+a_n \leq M} c_{i_1, i_2, \dots, i_n} x_{i_1}^{a_1} x_{i_2}^{a_2} \dots x_{i_n}^{a_n}$$

- The monomials can be used to represent all such polynomials as a vector space basis (of which dimension??)
- ANSWER:

$$m = \binom{M+n}{M}$$

- Thus we have a least squares problem over a matrix $\Phi(X)$ with m columns and r rows (number of data points). This is a **polynomial learning model of degree M**

$$\min \sum_i^m (h(x_i) - y_i)^2 = \min_{w \in \mathbf{R}^m} \|\Phi(X)^T w - \hat{y}\|_2.$$

Polynomial Regression

- We may add nonlinear relations on the features by considering a polynomial of degree M over n variables as the learning function:

$$h(x) = \sum_{a_1+a_2+\dots+a_n \leq M} c_{i_1, i_2, \dots, i_n} x_{i_1}^{a_1} x_{i_2}^{a_2} \dots x_{i_n}^{a_n}$$

- The monomials can be used to represent all such polynomials as a vector space basis (of which dimension??)
- ANSWER:

$$m = \binom{M+n}{M}$$

- Thus we have a least squares problem over a matrix $\Phi(X)$ with m columns and r rows (number of data points). This is a **polynomial learning model of degree M**

$$\min_i \sum_i (h(x_i) - y_i)^2 = \min_{w \in \mathbf{R}^m} \|\Phi(X)^T w - \hat{y}\|_2.$$

- It might seem that the more features we add, the better. However, there is a danger in adding too many features! We have to be careful with **underfitting** or **overfitting**.

Evaluating the regression result

- How to compare different regression models? How good is the prediction?

Evaluating the regression result

- How to compare different regression models? How good is the prediction?
- **Leave-one-out approach** Instead of using the entire data set to fit our model we use all but one point x_j . Missing point will be our test

Evaluating the regression result

- How to compare different regression models? How good is the prediction?
- **Leave-one-out approach** Instead of using the entire data set to fit our model we use all but one point x_j . Missing point will be our test
- Solve the smaller Least squares problem, we get a prediction for x_j , $\hat{y}(x_j)$. We can repeat the above process for all points (leave a different one out). Compute the *test set error*:

$$1/m \sum_{j=1}^m (\hat{y}(x_j) - y_j)$$

Evaluating the regression result

- How to compare different regression models? How good is the prediction?
- **Leave-one-out approach** Instead of using the entire data set to fit our model we use all but one point x_j . Missing point will be our test
- Solve the smaller Least squares problem, we get a prediction for x_j , $\hat{y}(x_j)$. We can repeat the above process for all points (leave a different one out). Compute the *test set error*:

$$1/m \sum_{j=1}^m (\hat{y}(x_j) - y_j)$$

- As the degree M of the polynomial grows the training error decreases, while the test set error typically decreases and then then increases. This is *overfitting*

Evaluating the regression result

- How to compare different regression models? How good is the prediction?
- **Leave-one-out approach** Instead of using the entire data set to fit our model we use all but one point x_j . Missing point will be our test
- Solve the smaller Least squares problem, we get a prediction for x_j , $\hat{y}(x_j)$. We can repeat the above process for all points (leave a different one out). Compute the *test set error*:

$$1/m \sum_{j=1}^m (\hat{y}(x_j) - y_j)$$

- As the degree M of the polynomial grows the training error decreases, while the test set error typically decreases and then then increases. This is *overfitting*
- Models of higher M complexity fit the training data better, but a model that is too complex (high M) does not behave well on unseen data. We are looking for a compromise.

Evaluating the regression result

- How to compare different regression models? How good is the prediction?
- **Leave-one-out approach** Instead of using the entire data set to fit our model we use all but one point x_j . Missing point will be our test
- Solve the smaller Least squares problem, we get a prediction for x_j , $\hat{y}(x_j)$. We can repeat the above process for all points (leave a different one out). Compute the *test set error*:

$$1/m \sum_{j=1}^m (\hat{y}(x_j) - y_j)$$

- As the degree M of the polynomial grows the training error decreases, while the test set error typically decreases and then then increases. This is *overfitting*
- Models of higher M complexity fit the training data better, but a model that is too complex (high M) does not behave well on unseen data. We are looking for a compromise.
- **leave-more out**