LEC003 Demand Forecasting

VG441 SS2021

Cong Shi
Industrial & Operations Engineering
University of Michigan

ARMA(p,q)

Auto regression moving average orders

- AR(p)

$$X_t = \mu + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t$$

error

base (intercept) | weights | past p realizations

- MA(q)

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

weights | past p residuals

- ARMA(p,q)

classical time series model

predict stationary time series

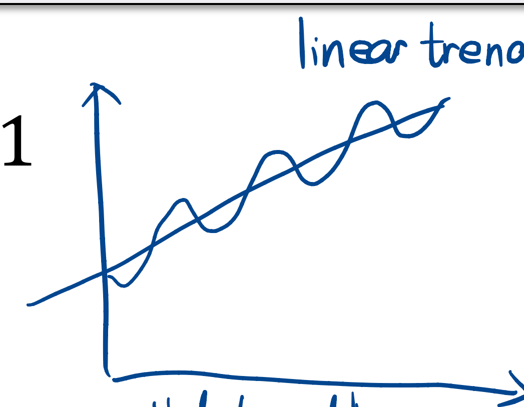
moving average order

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

ARIMA(p,d,q)

ARIMA(p,1,q) = ARMA(p,q)

- d is the degree of difference, e.g., d =1



$$Y_t = X_t - X_{t-1}$$

$$Y_t = \mu + \varepsilon_t + \sum_{i=1}^{p} \varphi_i Y_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

"detrend"

make $\boxed{\text{time series stationary}}$

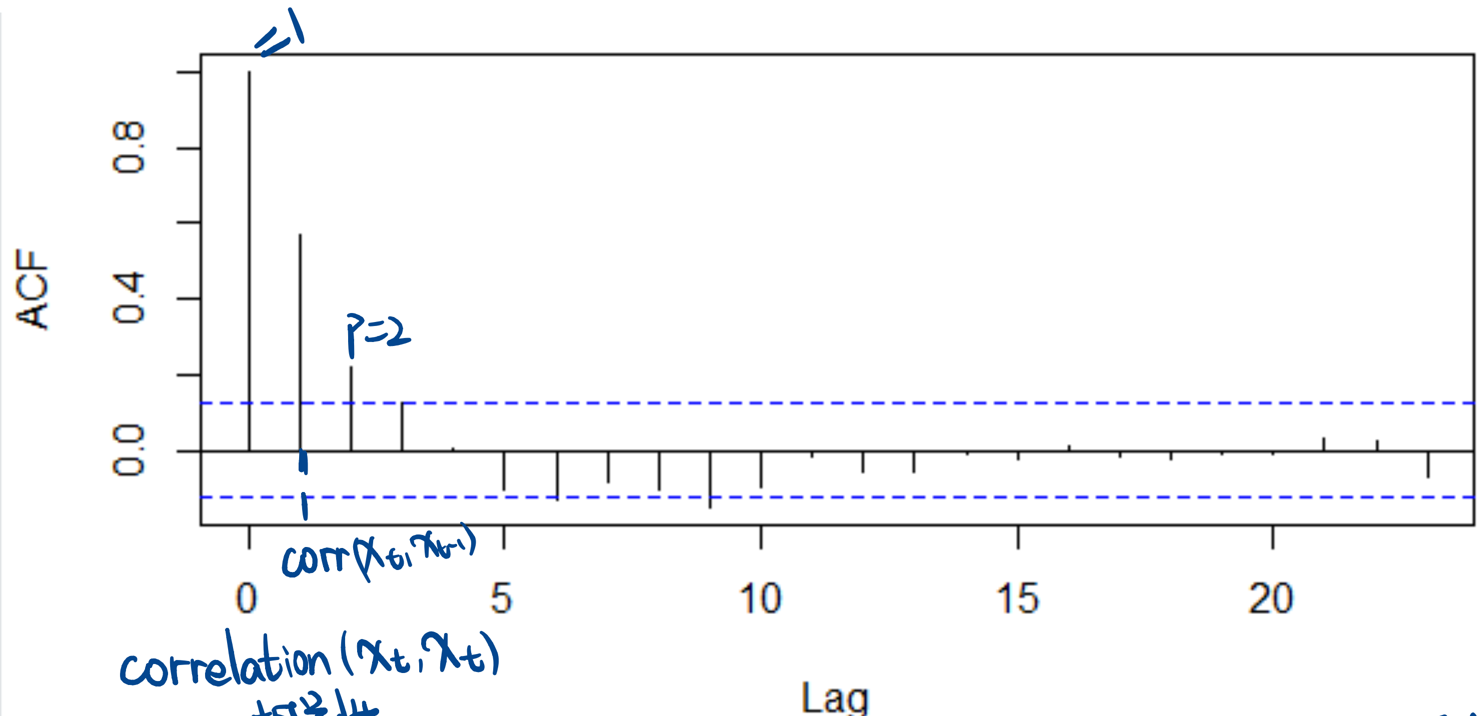
- How about d = 2? $Z_t = Y_t - Y_{t-1}$

ACF/PACF

how to determine d

Auto-correlation function

↳ python



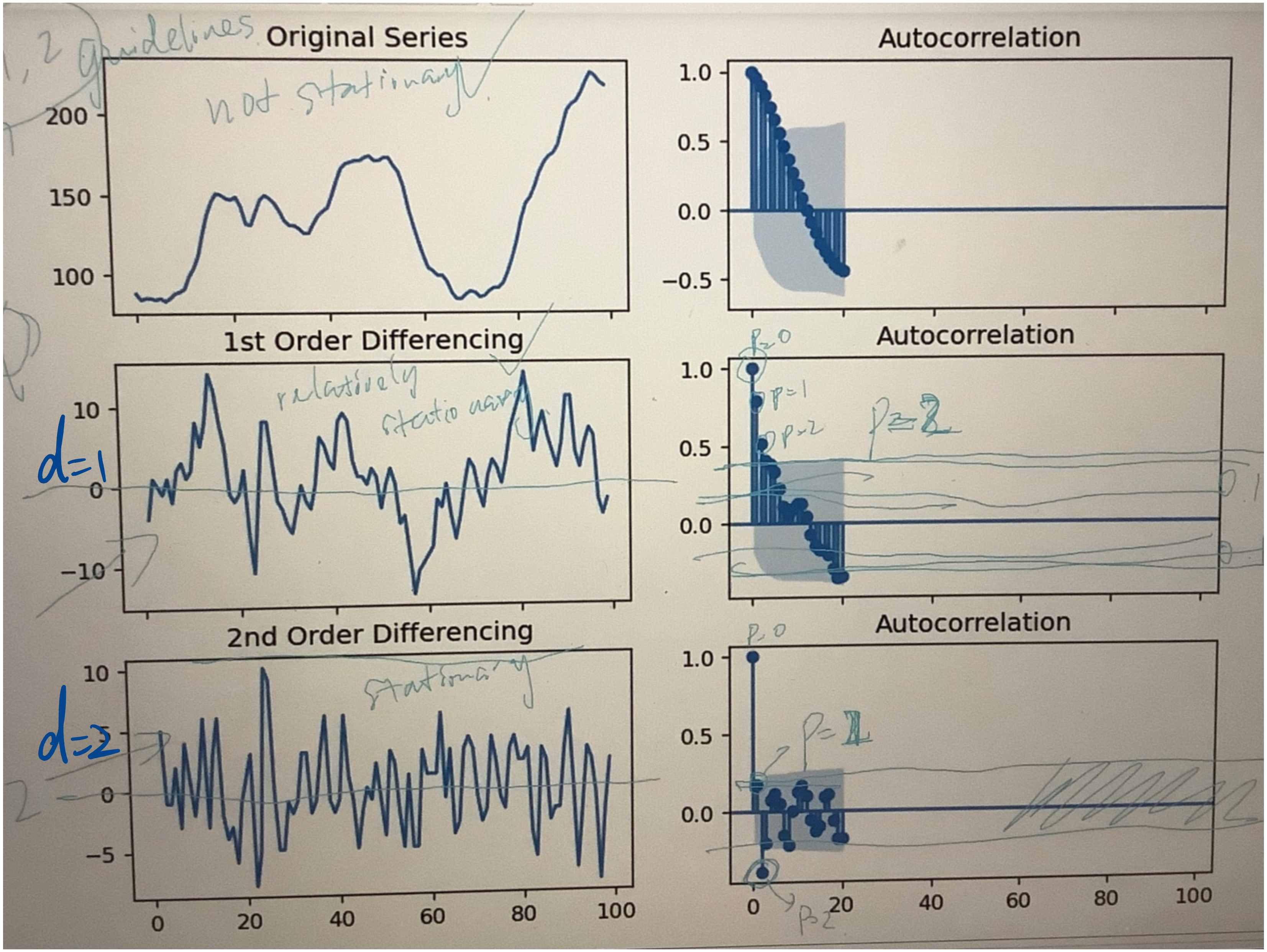
correlation(x_t, x_t)

相关性

$$\text{corr}(x_t, x_{t-1}) = \frac{E[(x_t - E(x_t))(x_{t-1} - E(x_{t-1}))]}{\sigma_{x_t}\sigma_{x_{t-1}}}$$

Python Time!

- statsmodels.tsa.arima_model



随着 lag↑, correlation 会↓

Adfuller test

H_0 it has the trend

H_a it does not have the trend

Adfuller test

if P-value small

$\Rightarrow$ reject H_0 $\Rightarrow$ time-series are stationary

Logistic Regression (0-1)

- Given features *x*, predict either 1 or 0 (on or off)

p

sigmoid/logistic function

$$p(1 \mid \boldsymbol{x}, \boldsymbol{w}) := \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) := \frac{1}{1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})}$$

1-p

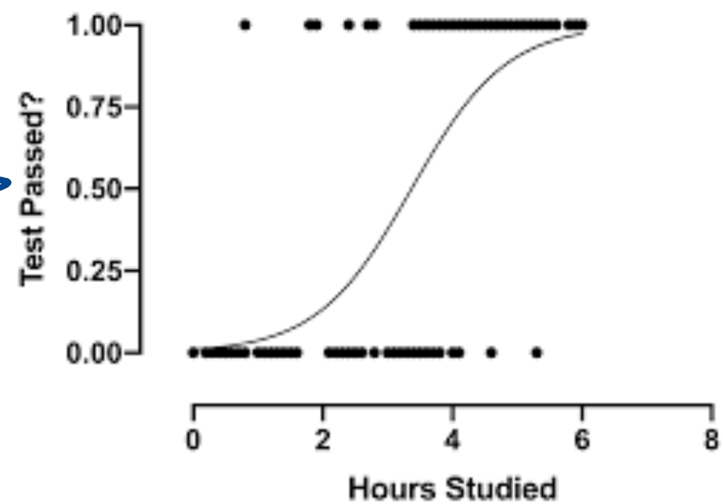
$$p(0 \mid \boldsymbol{x}, \boldsymbol{w}) = 1 - \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) = \frac{\exp(-\boldsymbol{w})}{1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})}$$

Connection to linear regression…

$$\log\left(\frac{p}{1-p}\right) = \boldsymbol{w} \cdot \boldsymbol{x}$$

log(odds) = $\frac{\#\text{wins}}{\#\text{losses}}$

logit function/log odds



Logistic Regression (1-layer NN)

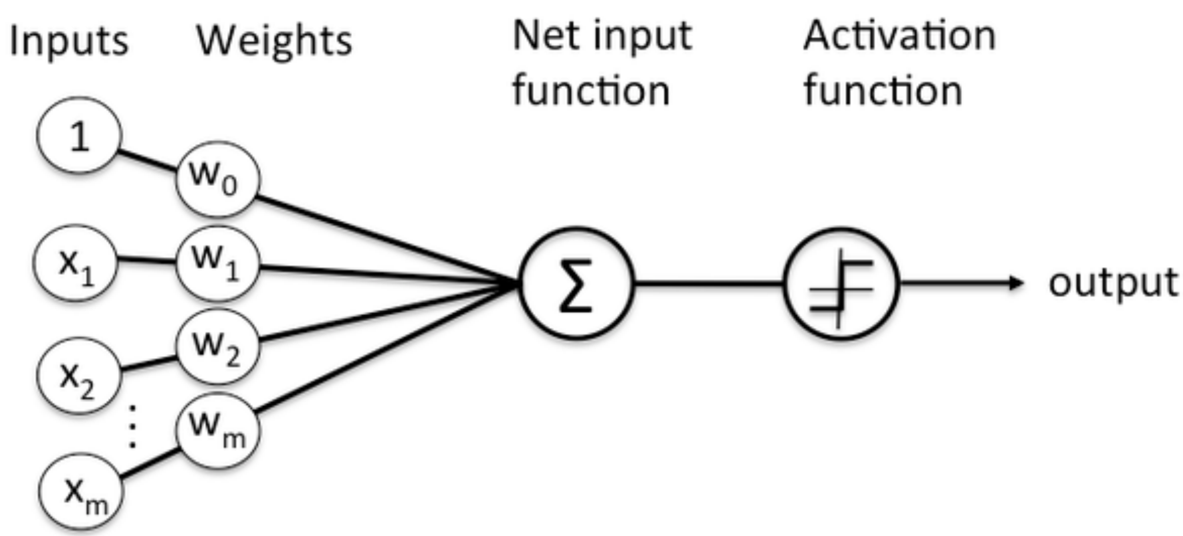
how do you train

- Given features *x*, predict either 1 or 0 (on or off)

p

sigmoid/logistic function

$$p(1 \mid \boldsymbol{x}, \boldsymbol{w}) := \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) := \frac{1}{1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})}$$



core question in ML

minimize weight of loss function

for linear regression. Loss function: least-square functions $\sum_i (y^{(i)} - w x^{(i)})^2$

for logistic regression.

Logistic Regression (0-1)

- Given features *x*, predict either 1 or 0 (on or off)

$$p(1 \mid \boldsymbol{x}, \boldsymbol{w}) := \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) := \frac{1}{1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})}$$

- Minimizing cross-entropy loss function:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{m} \Big(\underbrace{-y^{(i)} \log \sigma(\boldsymbol{w} \cdot \boldsymbol{x}^{(i)})}_{\text{如果 } y^{(i)}=1,} - \underbrace{(1 - y^{(i)}) \log \sigma(-\boldsymbol{w} \cdot \boldsymbol{x}^{(i)})}_{\text{如果 } y^{(i)}=0} \Big)$$

Probabilistic Interpretation

bad: no closed form

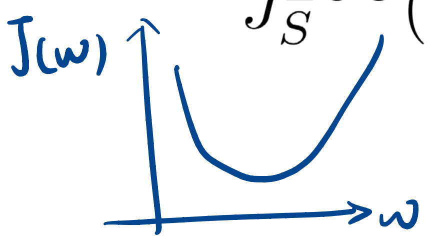
good: convex + closed form gradient

- Given features *x*, predict either 1 or 0 (on or off)

$$p(1 \mid \boldsymbol{x}, \boldsymbol{w}) := \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) := \frac{1}{1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})}$$

- Minimizing the "negative" MLE:

$$J_S^{\mathrm{LOG}}(\boldsymbol{w}) := -\frac{1}{m} \sum_{i=1}^{m} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{w}\right)$$



$$\boxed{\begin{aligned} -\log p(y \mid \boldsymbol{x}, \boldsymbol{w}) &= -y \log \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) - (1-y) \log \sigma(-\boldsymbol{w} \cdot \boldsymbol{x}) \\ &= \begin{cases} \log(1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x})) & \text{if } y = 1 \\ \log(1 + \exp(\boldsymbol{w} \cdot \boldsymbol{x})) & \text{if } y = 0 \end{cases} \end{aligned}}$$

convex in w

$$\nabla_{\boldsymbol{w}}(-\log p(y \mid \boldsymbol{x}, \boldsymbol{w})) = -(y - \sigma(\boldsymbol{w} \cdot \boldsymbol{x}))\boldsymbol{x}$$

Gradient Descent

iterative

- Minimizing the "negative" MLE:

$$J_S^{\mathrm{LOG}}(\boldsymbol{w}) := -\frac{1}{m}\sum_{i=1}^{m} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{w}\right)$$

- Gradient?

$$\nabla J_S^{\mathrm{LOG}}(\boldsymbol{w}) = \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - \sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^{(i)}\right)\right)\boldsymbol{x}^{(i)}$$

$$\boxed{\begin{aligned} -\log p(y \mid \boldsymbol{x}, \boldsymbol{w}) &= -y \log\sigma(\boldsymbol{w}\cdot\boldsymbol{x}) - (1-y)\log\sigma(-\boldsymbol{w}\cdot\boldsymbol{x}) \\ &= \begin{cases} \log(1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x})) & \text{if } y = 1 \\ \log(1+\exp(\boldsymbol{w}\cdot\boldsymbol{x})) & \text{if } y = 0 \end{cases} \end{aligned}}$$

$$\nabla_{\boldsymbol{w}}(-\log p(y \mid \boldsymbol{x}, \boldsymbol{w})) = -(y - \sigma(\boldsymbol{w}\cdot\boldsymbol{x}))\boldsymbol{x}$$

Gradient Descent

Input: training objective

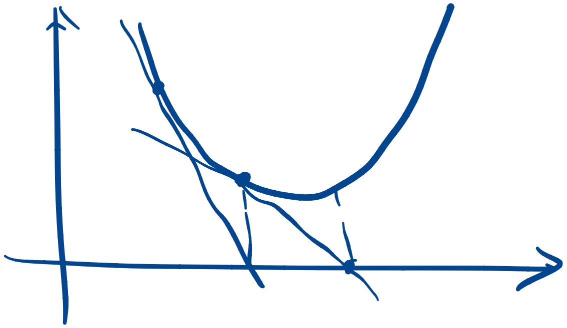
$$J_S^{\mathrm{LOG}}(\boldsymbol{w}) := -\frac{1}{m}\sum_{i=1}^{m} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{w}\right)$$

Output: parameter $\hat{\boldsymbol{w}} \in \mathbb{R}^n$ such that $J_S^{\mathrm{LOG}}(\hat{\boldsymbol{w}}) \approx J_S^{\mathrm{LOG}}\left(\boldsymbol{w}_S^{\mathrm{LOG}}\right)$

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \ldots T-1$,

$$\theta^{t+1} = \theta^t + \frac{\eta^t}{m}\sum_{i=1}^{m}\left(y^{(i)} - \sigma\left(\boldsymbol{w} \cdot \boldsymbol{x}^{(i)}\right)\right)\boldsymbol{x}^{(i)}$$

3. Return θ^T.



From Binary to Multinomial

- Given features x, predict $y \in \{1, \ldots, k\}$

$P(y)$

weights for each class

$$p(y \mid \boldsymbol{x}, \theta) = \frac{\exp\left(\boldsymbol{w}^{y} \cdot \boldsymbol{x}\right)}{\sum_{y'=1}^{k} \exp\left(\boldsymbol{w}^{y'} \cdot \boldsymbol{x}\right)}$$

softmax function

From Binary to Multinomial

- Minimize cross-entropy loss function

$$J_S^{\mathrm{LLM}}(\boldsymbol{w}) := -\frac{1}{m}\sum_{i=1}^{m} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}, \theta\right)$$

$$\boxed{-\log p(y \mid \boldsymbol{x}, \theta) = \underbrace{\log\left(\sum_{y'=1}^{k} \exp\left(\boldsymbol{w}^{y'} \cdot \boldsymbol{x}\right)\right)}_{\text{constant wrt. } y} - \underbrace{\boldsymbol{w}^{y} \cdot \boldsymbol{x}}_{\text{linear}}}$$

$$\nabla_{\boldsymbol{w}^l}(-\log p(y \mid \boldsymbol{x}, \theta)) = \begin{cases} -(1 - p(l \mid \boldsymbol{x}, \theta))\boldsymbol{x} & \text{if } l = y \\ p(l \mid \boldsymbol{x}, \theta)\boldsymbol{x} & \text{if } l \neq y \end{cases}$$

grad in closed form

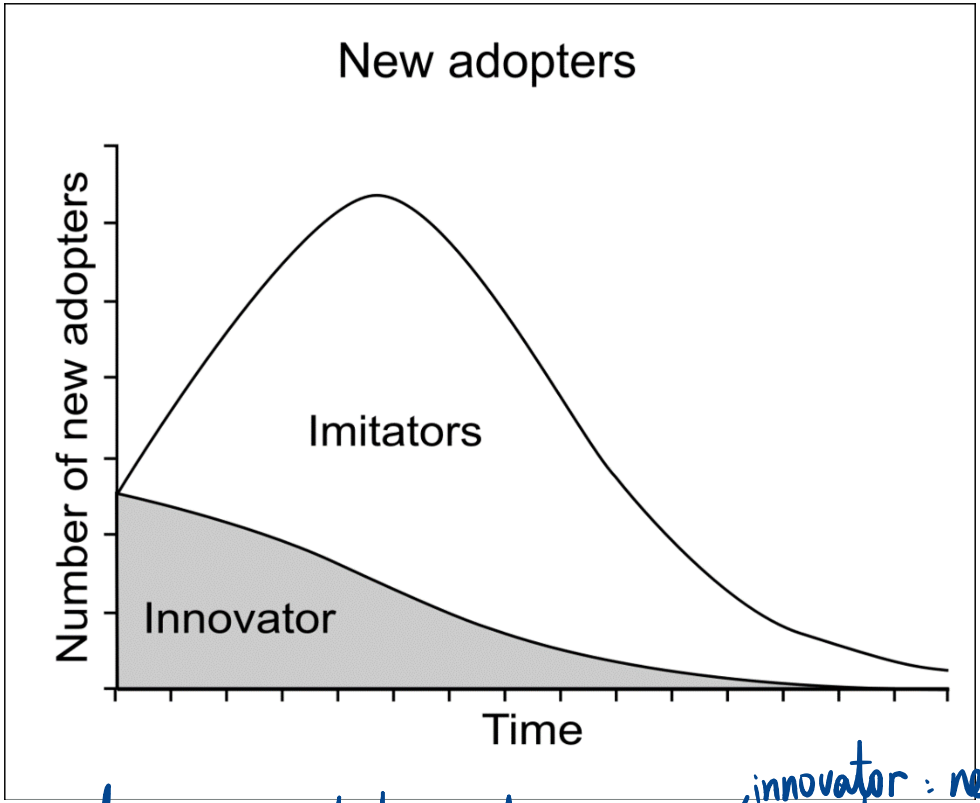
Python Time!

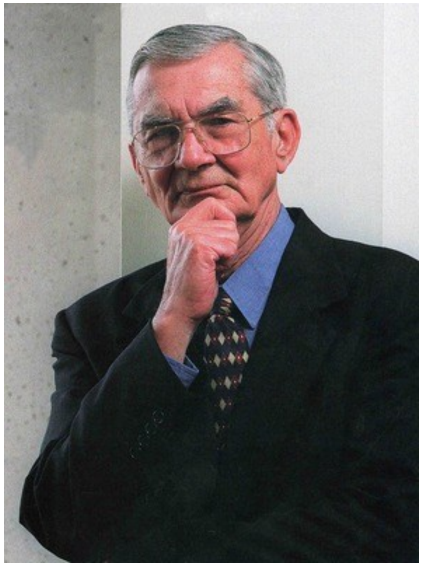
- from sklearn.linear_model import LogisticRegression

Bass Diffusion Model

predict product cycle

- How about projecting sales of new products?





Frank Bass

derive population into 2 groups < innovator : new products
imitators : more patients

Bass Diffusion Model

Cumulative purchase probability of a random customer $F(t)$

Purchase probability at time t $f(t) = F'(t)$

prob of purchasing

The rate of purchase at time t (given no purchase so far)

governing equation

pool of already purchased ↑

$$\frac{f(t)}{1 - F(t)} = p + q\underline{F(t)}$$

ODE

Coefficient of innovation

Coefficient of imitation

contagion effective

Bass Solution

$$\frac{dF/dt}{1-F} = p + qF$$

$$\frac{dF}{dt} = p + (q-p)F - qF^2$$

$$\int \frac{1}{p + (q-p)F - qF^2} dF = \int dt$$

$$\begin{aligned}\frac{1}{(p+qF)(1-F)} &= \frac{A}{p+qF} + \frac{B}{1-F} \\ &= \frac{A - AF + pB + qFB}{(p+qF)(1-F)} \\ &= \frac{A + pB + F(qB - A)}{(p+qF)(1-F)}\end{aligned}$$

$$A = q/(p+q)$$
$$B = 1/(p+q)$$

Bass Solution

$$\int \frac{1}{(p+qF)(1-F)} dF = \int dt$$

$$\int \left(\frac{A}{p+qF} + \frac{B}{1-F} \right) dF = t + c_1$$

$$\int \left(\frac{q/(p+q)}{p+qF} + \frac{1/(p+q)}{1-F} \right) dF = t + c_1$$

$$\frac{1}{p+q} \ln(p+qF) - \frac{1}{p+q} \ln(1-F) = t + c_1$$

$$\frac{\ln(p+qF) - \ln(1-F)}{p+q} = t + c_1$$

Boundary Condition

$$t = 0 \quad \Rightarrow \quad F(0) = 0$$

$$t = 0 \quad \Rightarrow \quad c_1 = \frac{\ln p}{p+q}$$

Bass Solution

purchasing pool at time t

$$F(t) = \frac{p\left(e^{(p+q)t} - 1\right)}{pe^{(p+q)t} + q}$$

how do you determine p, q

Calibration

m: total market size

- Sales in any period are $s(t) = mf(t)$
- Cumulative sales up to time t are $S(t) = mF(t)$

$$\frac{s(t)/m}{1 - S(t)/m} = p + qS(t)/m$$

$$s(t) = [p + qS(t)/m][m - S(t)]$$

$$s(t) = \beta_0 + \beta_1 S(t) + \beta_2 S(t)^2 \quad (BASS)$$
$$\beta_0 = pm$$
$$\beta_1 = q - p$$
$$\beta_2 = -q/m$$

$$m = \frac{-\beta_1 \pm \sqrt{\beta_1^2 - 4\beta_0\beta_2}}{2\beta_1}$$

$$p = \frac{\beta_0}{m}; \quad q = -m\beta_2$$

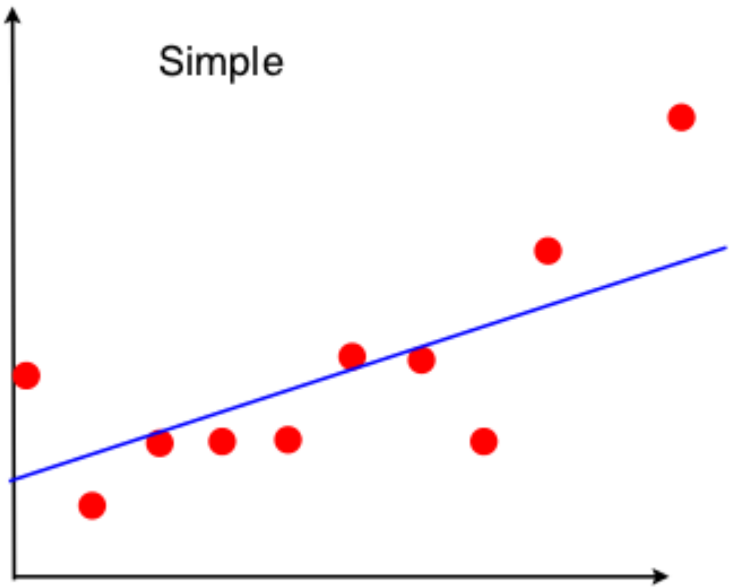
Conduct a linear regression!

Python Time!

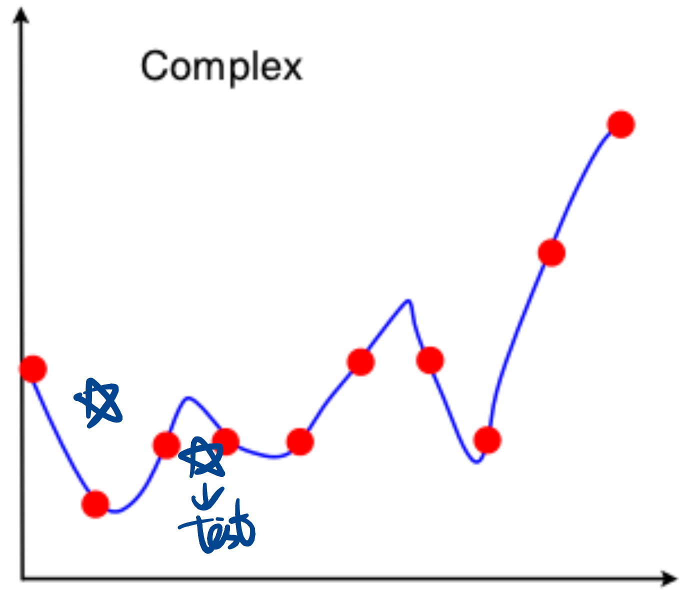
- from sklearn import linear_model
- from statsmodels.api import OLS

Bias-Variance Tradeoff

high bias
low variance



low bias
high variance



这种test可能不行

Kaggle Competition

All Competitions

Active | Completed | InClass

All Categories | Default Sort

Jigsaw Multilingual Toxic Comment Classification — $50,000
Use TPUs to identify toxicity comments across multiple languages
Featured • a month to go • Code Competition • 862 Teams

M5 Forecasting - Accuracy — $50,000
Estimate the unit sales of Walmart retail goods
Featured • 2 months to go • 3589 Teams

M5 Forecasting - Uncertainty — $50,000
Estimate the uncertainty distribution of Walmart unit sales.
Featured • 2 months to go • 389 Teams

University of Liverpool - Ion Switching — $25,000
Identify the number of channels open at each time point
Research • 16 days to go • 2333 Teams

TReNDS Neuroimaging — $25,000
Multiscanner normative age and assessments prediction with brain function, structure, and connectivity
Research • 2 months to go • 275 Teams

ALASKA2 Image Steganalysis — $25,000
Detect secret data hidden within digital images
Research • 2 months to go • 237 Teams

Prostate cANcer graDe Assessment (PANDA) Challenge — $25,000
Prostate cancer diagnosis using the Gleason grading system
Featured • 2 months to go • Code Competition • 309 Teams