

سوال (۱)

• \hat{y} Decision tree \hat{y} Random forests \hat{y} (overfit) \hat{y} (1.1) \hat{y} \hat{y}

4= میز Random forests, Decision tree, انجمن و سید و آت (صق) ر

مبادی نسبت به این که D.T. داشته باشد

$\Delta =$ الفحص R.F. نسبة P.T. وبقائها داده من مقام تركب

$\sigma =$ R.F. کی نوع Ensemble کی نسبت D.T. واریانس کمتر دارد.

$\Delta = RF$ بے $D.T$ روایت کے طور پر تفسیر

(1.4 OB

- Rainforest forests \rightarrow

• \sqrt{d} overfit \Rightarrow XGBoost \Rightarrow R.F \Rightarrow \sqrt{d}

• ~~inverse~~ similarity score, also cosine similarity $\cos(\theta)$

hyperparameter γ, b : RF parameters γ, b are hyperparameters

۱- تعریف بیست و نهم در باره Functional نیز

4. $\vec{r} = r \hat{r}$ (طول و جهت) و $\hat{r} = \frac{\vec{r}}{r}$ (جهت واحد)

✓ به صورت $1/N$ نیت به دور در 60° و 16 R.F. در 16 کلاسیک است

دارد شود، و همدر $XG\text{Boost}$ نتایج به صورت 99% و 99% قابل مشاهده است.

: AdaBoost \checkmark

$\Delta =$ سرعت بیشتر دارد.

$\sigma =$ رقت بیشتر دارد .

$\sigma =$ دقت بیشتر دارد .
 $\Delta =$ از AdaBoost بیشتر binary classification کاربرد دارد و آماروس

- Regression or Classification like XGBoost

$\tau =$ قابلية قصير، يعطى بـ هـ.ج .

۷- نوع کار مختلف از تابع

روش Adenpost فقطای نوع مایع ۱۵۵۵ ستون استوارکس .

1- این مدل دفاع در برابر وجود بایاس آماری Gradient Boosting است.

2- Regularization به آن اضافه شده.

3- overfitting کمتر دارد.

4- سرعت عملکرد بهتر دارد. (مراحل training و paralyze شالوده ریاضی)

5- با توجه به اینکه دیتا سترت داریم و دیتا یان به صورت ترکیبی از دیتا حالت بندی شده است به سبب این است که G.B.

(۲۰۱۳)

روش Random Forests برای پیش بینی کردن دیتا :

- 1- ابتدا K نمونه به صورت تصادفی انتخاب می کنیم.
- 2- استفاده از داده ها مطابق این نمونه ها یک مدل تعریف می کنیم به ازای هر Sample
- 3- Voting را به صورت سانسور کردن از دیتا ها تعریف می کنیم و انجام می دهیم.
- 4- نتیجه = نتیجه که بیشترین Vote را بین دیتا ها داشته باشد : در صورت Classification
نتیجه = سانسور می کنیم : در صورت Regression

در واقع اجازه می دهیم یک subset از feature ها را در نظر بگیریم.

به این صورت که یک subset از feature ها را در نظر بگیریم و به ازای آن دیتا تعریف می کنیم. به سبب این که اکثریت Classification و Majority voting انجام داده از هر دیتا هر دیتا test به صورتی که بیشتر نتیجه را می دهد.

- Subset 1 = {Blocked Arteries, Exc, Gen} \Rightarrow
- Subset 2 = {Block Art, Exc, weight} \Rightarrow
- Subset 3 = {Block Art, weight, Gen} \Rightarrow
- Subset 4 = {Exc, weight, Gen} \Rightarrow
- نتیجه بر اساس majority voting مشخص می شود.

انتخاب K مقدار : K iteration , training dataset , $w_1(i) = \frac{1}{n}$, $(i=1, \dots, n)$

for $r=1$ to K :

$w_r(t)$

for $j=1$ to n :

$$w_r(j) = \frac{w_r(j)}{\sum w_r(j)} \quad \leftarrow \text{normalize } w_r(i)$$

دیتا را در هر iteration

$$h_r := \text{FitWeakLearner}(D, w_r)$$

$\sigma = \text{weak learner}$ فیت کردن

$$\epsilon_r := \sum_{i=1}^n w_r(i) \mathbb{I}(h_r(i) \neq y_i)$$

\leftarrow کلاس را با این وزن ها

if $\epsilon_r > 1 - \frac{1}{C}$ then stop

\leftarrow آنگاه ریزش را به دست آوریم

$$\alpha_r := \frac{1}{2} \log \left[\frac{(1 - \epsilon_r)}{\epsilon_r} \right] + \log(C - 1)$$

\leftarrow α_r وزن است این classifier , نشان دهد

$$w_{r+1}(i) = w_r(i) \times \begin{cases} e^{\alpha_r} & : h_r(x_i) \neq y_i \\ e^{-\alpha_r} & : h_r(x_i) = y_i \end{cases}$$

\leftarrow وزن ها را آپدیت کنیم بر اساس

نیز اضا به وزن ها می دهیم

$$\text{Predict } h_{\text{ens}}(x) = \arg \max_j \sum_r \alpha_r \mathbb{I}(h_r(x) = j)$$

بزرگ ترین وزن را

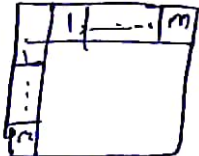
majority voting

۱) حذف داده‌ها که مقدار UT feature $miss$ است.

۲) برگزیدن آن $value$ یا $regression$ یا $classification$:
 median آن برگزیدن
 mode آن برگزیدن

ابتدا چون شیوه مدین و مود را استفاده کنیم، برای $value$ و $miss$ روشی مقدار لایه $leaf$ پس از آن می‌توانیم مقدار $value$ را به صورت زیر آزمون کنیم تا ببینیم به مقدار $miss$ چه می‌شود. تغییر آن ضامن نباشد.

برای صورتی که $Random Forest$ استفاده کنیم و داده‌ها در دانه‌ها هستند، اجزاء کنیم به این 2 دانه از آن دانه $label$ می‌توانیم (که $leaf$ قرار دارند)

در این روش شایع است. به صورت  (مقدار $sample$ ها) دانه

(بزرگ) در (کوچک) را به هم می‌زنیم و به هم می‌زنیم. دانه‌ها را به هم می‌زنیم.

به مقدار دانه‌ها تقسیم کنیم و یک توزیع log به هم می‌زنیم.

حال برای $sample$ که دانه $miss$ شده دارد. مقدار $value$ اش را برای

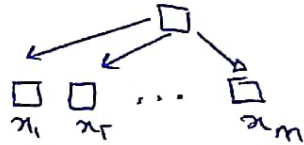
توزیع log های که از ماتریس شایع به دست می‌آوریم. این به هم می‌زنیم و مقدار $value$ را به دست می‌آوریم.

و همانطور که ابتدا گفتیم این روش را تا به قدری ادامه می‌دهیم.

سوال ۳) Gradient Boost درخت گزینش:

- ۱) یک base tree ابتدا داریم.
- ۲) بر اساس درختها درخت مهمی پیدا میکنیم و اضافه میکنیم.
- ۳) درخت ۲ را با درخت ۱ ترکیب میکنیم و سپس به درخت ۱ بازگردیم.

که اجزای آن درختها را همیشه به صورت زیر باشد:



ابتدا درخت اول را به صورت فوق الذکر خود به صورت مقابل داریم:

حال اگر این درخت را حساب میکنیم:

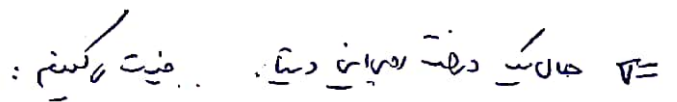
$$\arg \min_{\hat{y}} \sum (y_i - \hat{y})^2 \rightarrow \hat{y} = \frac{1}{m} \sum_{i=1}^m y_i$$

$$\Rightarrow \hat{y}_{[1]} = \frac{1}{n} \sum_{i=1}^n y^{(i)} = \text{weight} = 11,17$$

حال یک سونجیده به عنوان residual اضافه میکنیم:

$$r_1 = y_1 - \hat{y}_1$$

Residual
4,13
17,13
-10,17
-12,17
1,13
0,13

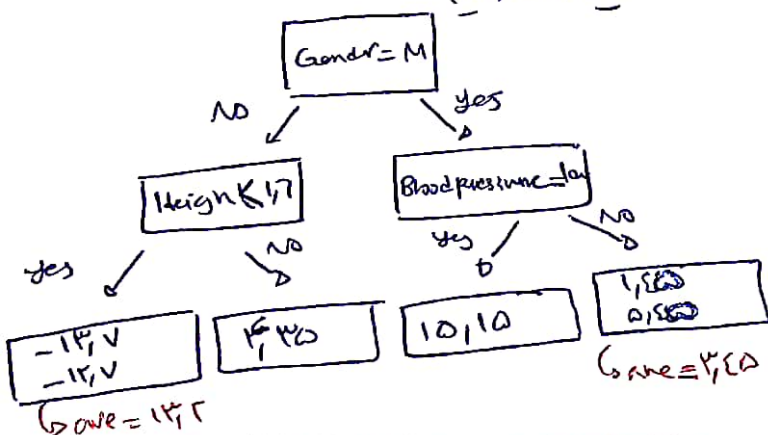

$$h_t(x) = h_{t-1}(x) + \alpha \hat{y}_t$$
$$\begin{array}{c} + \quad \alpha \\ \uparrow \\ (0,1) \end{array}$$
11

دو = residual value
یک = prediction

Residual
F, VQ
10/10
-14 V0
-11 V
1/50
0/50

residual = $y_i - \hat{y}_i$ = difference between observed and predicted values

حال بالاسنادہ از



۴.۱) هدف از مقدار کردن وزن‌ها (به صورت مقدار) این است که منجر به این شود که در هنگام back propagation که این مقادیر تکراراً به صورت زنجیره‌ای در یکدیگر ضرب می‌شوند، منجر به تولید مقادیر بسیار بزرگ نشود (معمولاً در شبکه‌های عمیق با مقدار لای زیاد). که مقادیر بزرگ تکراراً منجر به آلودگی وزن‌ها و مقادیر بزرگ می‌شود. که ممکن است منجر به واگرایی الگوریتم منجر شود. در شبکه stable نباشد. معمولاً در شبکه‌ها با مقدار لای زیاد اجتناب شود. که مقدار زیادی عدد بزرگ در ضرب می‌شوند.

راهکار دیگر:

۱= استفاده از روش back normalization [توزیع ورودی هر لای را می‌توان ثابت نگه داشت]
 ۲= استفاده از توابع مانند sigmoid, tanh, ReLU که خروجی را محدود می‌کنند به بازه‌ای مشخص یا بین مقادیر ثابتی نگه می‌دارند.

۳= تغییر حجمی که با کاهش تعداد لای‌ها، یادگیری را با batch

۴= استفاده از جمله‌های long short term.

۵= کاهش نرخ Learning rate

۶= استفاده از weight regularization با تغییر loss function در وزن‌های بزرگ.

۷= gradient clipping : برای اینکه threshold میزان تکرار لای‌ها کمتر شود و بزرگتر نشود.
 ۸= برای آن threshold بگذاریم مثلاً.

۴.۲) مثال:

۱= افزایش دقت مدل

۲= افزایش قدرت مدل در شناسایی الگوهای پیچیده‌تر و چون یادگیری بیشتر در نظر می‌گیرد. در نهایت کاهش خطا می‌شود.

مقایسه: $\nabla =$ بیشترین

$\nabla =$ مقدار سوزش دینایی برای train افزایش مقدار یادگیریها زیاده.

$\nabla =$ gradient exploding

$\nabla =$ یاد آموختن مدل با تغییر افزایش مقدار یادگیریها زیاده.

$\nabla =$ Vanishing error

$\nabla =$ با افزایش زیاد لایه ها ممکن است منجر به overfitting شود.

نشان دهنده (4.3) shuffle نشده باشد، ممکن است این استوریج در پیدا کردن global opt شده برنجود

با سرعت رسیدن آن به نقطه global opt نسبت به shuffle شده بیشتر شود. و از

نیازها دوری در پیدا کردن global opt در هر gradient descent جبرئتی کند.

مثلاً فرض کنید یک دیتا داریم از یک رتبه بندی که در داده ordered و نشان شده

دهد احتمال سفید شدن داشته باشد، در این صورت اقتراح بیشتر فراتر از هم برود

فرض بیشتر کند از ششصد. رسیدن در راهی طریق تیر کند که با احتمال بیشتر که به خطا دهد.

این رفتار منطقی را ایجاد کند در اینجا که خطا که خوب است shuffle با ششصد

از این روش شود.

(4.4) حافظه که از فرم تابع sigmoid ششصد باشد $\sigma(x) = \frac{1}{1+e^{-x}}$

مقدار لایه آن در یکای بسیار بزرگ به صفر نزدیک و نزدیک به صفر شود و یک مقدار فنیکو که نزدیک به صفر دارد.

شده که ممکن است اینطور Vanishing gradient باشد، به این صورت که چون مقدار لایه

بزرگترین مغز است در فرانت propagation back این مقدار در لایه ها صفت اشتراک کننده لایه ها

به صفر رساند (مکلف است) این صفت Vanishing باشد.

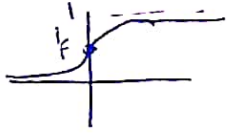
حل شده: به عنوان استفاده از تابع ReLU در لایه های میانی که از طرف خود ~~ReLU~~

البته در وقت (-) که وجود دارد اما در صفت اعلام مثبت شده خواهد بود و لایه 1 داریم.

با نرالی کردن داده ها که کند باشد.

$$t = \text{ReLU}(z) = \begin{cases} 0 & z \leq 0 \\ z & z > 0 \end{cases}$$

$$\sigma(\text{ReLU}(z)) = \sigma(t) = \begin{cases} \sigma(0) = \frac{1}{1+e^0} = \frac{1}{2} & z \leq 0 \\ \sigma(z) > \frac{1}{2} & z > 0 \end{cases}$$



تابع σ
طبق σ مقدار

$$\text{داریم} \Rightarrow \sigma(\text{ReLU}(z)) = \begin{cases} \frac{1}{2} & z \leq 0 \Rightarrow \text{class 1} \\ > \frac{1}{2} & z > 0 \Rightarrow \text{class 2} \end{cases}$$

در نتیجه با مقیاس برداشتن دو درصت به 1 «class» assign می شود.

پس نتیجه دیگر این است که به کلاس 1 assign می شود.

(۴.۶)

نکته دیگر activation function نیست باشد. یعنی هیچ قسیمی برای محاسبه سازی روابط ندارد. و نکته دیگر به صورت روابط خطی باشد، یعنی توان ضرب را به اساس تابع خطی از ورودی هادرون لای اول نوشت. که در نتیجه نودها و hidden layer ها بلا استفاده هستند و صرفاً بار محاسباتی را افزایش می دهند.

۵.۱) ابتدا شبکه را به صورت یک بردار 16×1 در نظر بگیرید. به صورت زیر می توانیم درجدها را بنویسیم:

0	1	2	3
7	6	5	4
8	9	10	11
15	14	13	12

$$\Rightarrow \text{image}^T = [0, 1, 2, \dots, 15]$$

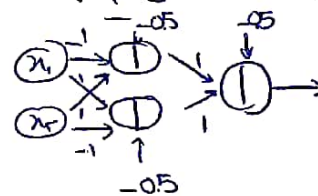
حالا شبکه را خود را به صورت زیر در نظر بگیرید:

یک hidden layer در نظر بگیرید

هر نود ضربی XOR دوتا از نودهای ورودی باشد.

حالا به این صورت که یک perceptron به ازای

بهم XOR کنیزا قبلاً در درس دیدیم به صورت:



باشد.

حالا در انتها کانهاست

نود داشته باشیم. با توجه به -74.5 که از

همان نودها بوجود می آید.

۵.۲) XOR به صورت sequential غیریک رند چون ~~در این~~ خانه ها می توانیم اینها را بنویسیم.

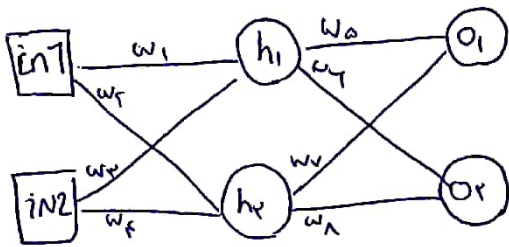
در انتها که می شود که تا آنجا که می توانیم غیریک رند نباشد که بهترین شرایط را داشته باشد.

۵.۲) در MLP برای هر ~~یک~~ از نودهای ^{بیگانه} یک نود در نظر می گیریم. که در نهایت با هم به شبکه

بر می گردیم به دلیل تعداد زیاد پارامترها و سرد شدن training بیشتر و ممکن است

overfitting نیز رخ دهد. شبکه ای که این است که اطلاعات کمی می تواند از آن بیاید

معمولاً به صورت بهر دوری آن. بیشتر از اطلاعات loss می تواند.



$$w_1 = 2 \times 0.1$$

$$in_1 = 0.1$$

$$in_2 = 0.2$$

$$bh_1 = 0.1$$

$$bh_2 = 0.2$$

$$bo_1 = 0.1$$

$$bo_2 = 0.2$$

$$dr = 1$$

$$t_1 = 0.2$$

$$t_2 = 0.9$$

(405)

$$E = \frac{1}{2} \sum (t_i - o_i)^2$$

Forward :

$$h_1 = w_1 \cdot in_1 + w_2 \cdot in_2 + bh_1 = 0.01 + 0.10 + 0.10 = 0.21$$

$$h_2 = w_3 \cdot in_1 + w_4 \cdot in_2 + bh_2 = 0.02 + 0.12 + 0.20 = 0.34$$

σ گزینش : $\sigma(h_1) = \frac{1}{1 + e^{-h_1}} = 0.401 = h'_1$

$\sigma(h_2) = \frac{1}{1 + e^{-h_2}} = 0.710 = h'_2$

$$o_1 = w_5 \cdot h'_1 + w_6 \cdot h'_2 + bo_1 = 1.1011$$

$$o_2 = w_7 \cdot h'_1 + w_8 \cdot h'_2 + bo_2 = 1.203$$

Sigmoid گزینش $\Rightarrow \sigma(o_1) = \frac{1}{1 + e^{-o_1}} = 0.444 = o'_1$

$\sigma(o_2) = \frac{1}{1 + e^{-o_2}} = 0.779 = o'_2$

$$E = \frac{1}{2} \sum (t_i - o'_i)^2 = 0.401$$

Back propagation :

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial o_1'} \times \frac{\partial o_1'}{\partial o_1} \times \frac{\partial o_1}{\partial w_5} = (o_1' - t_1) \times (o_1'(1 - o_1')) \times h_1' = 0.049$$

$$\frac{\partial E}{\partial w_6} = \frac{\partial E}{\partial o_1'} \times \frac{\partial o_1'}{\partial o_r} \times \frac{\partial o_r}{\partial w_6} = (o_1' - t_1) \times (o_1'(1 - o_1')) \times h_1' = -0.019$$

$$\frac{\partial E}{\partial w_7} = \frac{\partial E}{\partial o_1'} \times \frac{\partial o_1'}{\partial o_f} \times \frac{\partial o_f}{\partial w_7} = (o_1' - t_1) \times (o_1'(1 - o_1')) \times h_1' = 0.019$$

$$\frac{\partial E}{\partial w_8} = \frac{\partial E}{\partial o_1'} \times \frac{\partial o_1'}{\partial o_r} \times \frac{\partial o_r}{\partial w_8} = (o_1' - t_1) \times (o_1'(1 - o_1')) \times h_1' = -0.019$$

for w_9 and w_{10}

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_1} \cdot \frac{\partial o_1}{\partial h_1'} \cdot \frac{\partial h_1'}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_1} + \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_r} \cdot \frac{\partial o_r}{\partial h_1'} \cdot \frac{\partial h_1'}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_1}$$

$$\Rightarrow \frac{\partial E}{\partial w_1} = 1.111 \times 10^{-3}$$

$$\frac{\partial E}{\partial w_r} = \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_1} \cdot \frac{\partial o_1}{\partial h_r'} \cdot \frac{\partial h_r'}{\partial h_r} \cdot \frac{\partial h_r}{\partial w_r} + \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_r} \cdot \frac{\partial o_r}{\partial h_r'} \cdot \frac{\partial h_r'}{\partial h_r} \cdot \frac{\partial h_r}{\partial w_r}$$

$$\Rightarrow \frac{\partial E}{\partial w_r} = 1.277 \times 10^{-3}$$

$$\frac{\partial E}{\partial w_f} = \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_1} \cdot \frac{\partial o_1}{\partial h_f'} \cdot \frac{\partial h_f'}{\partial h_f} \cdot \frac{\partial h_f}{\partial w_f} + \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_r} \cdot \frac{\partial o_r}{\partial h_f'} \cdot \frac{\partial h_f'}{\partial h_f} \cdot \frac{\partial h_f}{\partial w_f}$$

$$\Rightarrow \frac{\partial E}{\partial w_f} = 0.093 \times 10^{-3}$$

$$\frac{\partial E}{\partial w_z} = \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_1} \cdot \frac{\partial o_1}{\partial h_z'} \cdot \frac{\partial h_z'}{\partial h_z} \cdot \frac{\partial h_z}{\partial w_z} + \frac{\partial E}{\partial o_1'} \cdot \frac{\partial o_1'}{\partial o_r} \cdot \frac{\partial o_r}{\partial h_z'} \cdot \frac{\partial h_z'}{\partial h_z} \cdot \frac{\partial h_z}{\partial w_z}$$

$$\Rightarrow \frac{\partial E}{\partial w_z} = 0.117 \times 10^{-3}$$

$$w_1' = w_1 - lr \frac{\partial E}{\partial w_1} = 0.99$$

$$w_r' = w_r - lr \frac{\partial E}{\partial w_r} = 0.199$$

$$w_f' = w_f - lr \frac{\partial E}{\partial w_f} = 0.995$$

$$w_z' = w_z - lr \frac{\partial E}{\partial w_z} = 0.395$$

$$w_o' = w_o - lr \frac{\partial E}{\partial w_o} = 0.51$$

$$w_1' = w_1 - lr \frac{\partial E}{\partial w_1} = 0.419$$

$$w_r' = w_r - lr \frac{\partial E}{\partial w_r} = 0.419$$

$$w_z' = w_z - lr \frac{\partial E}{\partial w_z} = 0.119$$

IT