1.1

از آن جا که پیش از گرفتن اطلاعات تابع را ارائه کرده ایم پس دیتایی نداشتیم تا بتوانیم در فضای فرضیه جست‌وجو کنیم و بنابراین فضای فرضیه مان از ابتدا شامل همان یک فرضیه بوده است ⟵ $M=1$

1.2

$$\text{Hoeffding}: \quad P\left[|E_{out} - E_{in}| > \varepsilon\right] \leqslant 2M e^{-2\varepsilon^2 N}$$

$$\rightarrow P\left[|E_{out} - E_{in}| > \frac{2}{100}\right] \leqslant 2 \times 1 \times e^{-2 \times 4 \times 10^{-4} \times 10^{4}} = 2e^{-8}$$

بنابراین طبق هافدینگ به احتمال حداقل $1-2e^{-8}$ خطا کمتر از دو درصد است

1.3

دلیل این اتفاق آن است که دیتایی که بانک در اختیار ما گذاشته نتوانسته به خوبی تابع ما را ارزیابی کند. به این معنا که توزیع داده های زمان آموزش و تست متفاوت بوده و distribution shift داشتیم. به عنوان مثال یکی از عواملی که باعث می‌شود انتخاب حدسی ما ساده‌تر شود آن است که دیتایی که در اختیار ما قرار گرفته، تماماً متعلق به افرادی است که بانک بر اساس روش های پیشین خود آنها را تایید کرده است و بنابراین دیتا به سمتی بایاس است که افراد تأیید شوند.

حال در زمان تست اما داده ها از توزیع دیگری می‌آیند و با مسئله‌ی out of distribution generalization رو به رو می‌شویم و مدل عملکرد ضعیفی را بروز می‌دهد

⊛ در حقیقت باند هافدینگ بر فرض iid بودن توزیع های train و test استوار است

⊛ به طور کلی وقتی به این صورت با مسئله generalization روبه رو می‌شویم یا تعداد سمپل ها نسبت به مدل استفاده شده کم بوده یا توزیع داده های آموزش و تست متفاوت است

1.4

یک راه آن است که سعی کنیم توزیع داده های آموزش و تست را یکی کنیم. هنگامی که متوجه شدیم دیتای آموزش به چه سمتی بایاس شده است می‌توان سمپل هایی که توزیع را به آن سمت برده اند را تا حدی کاهش داد تا این شیفت از بین برود. به عنوان مثال اگر متوجه شدیم که دیتا به سمت افراد تأیید شده بایاس شده است، می‌توان تعدادی از آنها را از دیتا حذف کرد.

البته اضافه کردن دیتایی که در زمان آموزش دیده نشده ولی در زمان تست وجود داشته کار سخت تری است که تنها در شرایط خاص قابل انجام است

پس از اینکه توزیع ها مشابه شوند، باند هافدینگ در خصوص generalization همان مقدار قبلی را می دهد اما این بار عملکرد مدل دیگر perfect نخواهد بود و در همان زمان evaluation متوجه دقت کم آن می‌شویم

2.1

در حوزه یادگیری ماشین منظور از درخت تصمیم متفاوت، متفاوت از نظر لیبل زدن سمپل ها است

بنابراین اگر m ویژگی داشته باشیم، 2^m حالت برای سمپل ها متصور است و چون دسته بندی binary است

پس هر کدام از این سمپل ها نیز می توانند 2 لیبل بگیرند ← 2^{2^m} درخت تصمیم متفاوت داریم

2.2

خیر درخت بهینه نیست

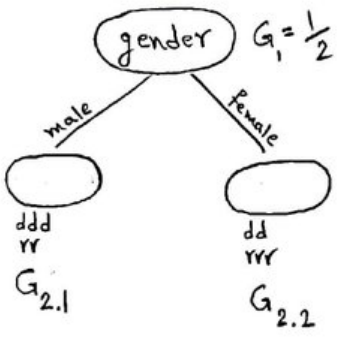
انتخاب حریصانه تنها می تواند به ما بگوید که در هر مرحله (به تنهایی)، کدام split بهتر است اما ممکن است یک split در حال حاضر خوب نباشد اما اگر آن را انجام دهیم در ادامه به جواب بهینه ای برسیم. به همین صورت ممکن است یک split که در حال حاضر بهترین gain را دارد در ادامه بسیار بد باشد.

با اینکه این الگوریتم ساختار درخت بهینه را به دست نمی دهد اما به دلیل ساده بودن و سریع بودن اجرای آن، همچنان به جواب های مناسبی می رسد

2.3.a

در مرحله اول 3 گزینه برای split داریم که gini gain را برای هر کدام به دست می‌آوریم:

① سمپل با لیبل جمهوری‌خواه را ۲ و سمپل با لیبل دموکرات را d می‌نامیم



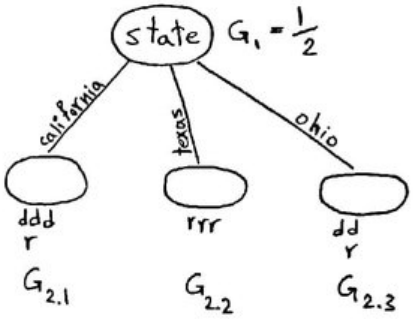
$$G_{2.1} = \frac{3}{5} \times \frac{2}{5} \times 2 = \frac{12}{25}$$

$$G_{2.2} = \frac{2}{5} \times \frac{3}{5} \times 2 = \frac{12}{25}$$

$$\text{gini gain} = \frac{1}{2} - \frac{12}{25} = \frac{1}{50}$$

avg $\rightarrow \frac{12}{25}$

gini gain for "race" is also $\frac{1}{50}$



$$G_{2.1} = \frac{3}{4} \times \frac{1}{4} \times 2 = \frac{3}{8}$$

$$G_{2.2} = \frac{3}{3} \times 0 \times 2 = 0$$

$$G_{2.3} = \frac{2}{3} \times \frac{1}{3} \times 2 = \frac{4}{9}$$

$$\text{gini gain} = \frac{1}{2} - \frac{17}{60} = \frac{13}{60}$$ ✓

avg

$$\frac{4}{10}\left(\frac{3}{8}\right) + \frac{3}{10}\left(\frac{4}{9}\right) = \frac{17}{60}$$

پس ابتدا بر اساس state تقسیم می‌کنیم.

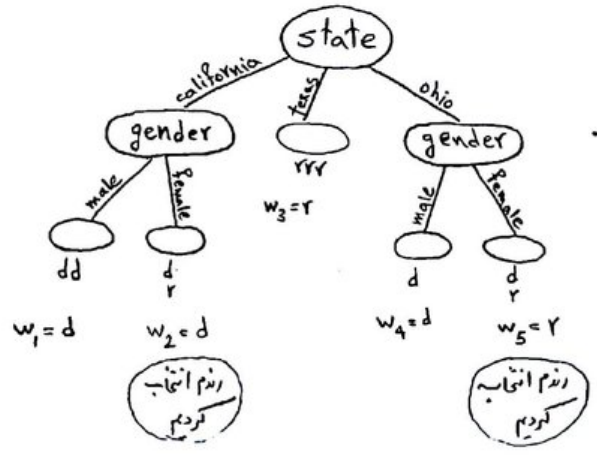
اگر سمپل های کالیفرنیا را بر اساس جنسیت تقسیم کنیم (male: dd، female: d ۲) خواهیم داشت که $\text{gain} = \frac{3}{8} - \frac{1}{4} = \frac{1}{8}$ ✓

اگر سمپل های کالیفرنیا را بر اساس نژاد تقسیم کنیم (white: dd ۲، black: d) خواهیم داشت که $\text{gain} = \frac{3}{8} - \frac{1}{3} = \frac{1}{24}$

اگر سمپل های اوهایو را بر اساس جنسیت تقسیم کنیم (male: d، female: d ۲) خواهیم داشت که $\text{gain} = \frac{4}{9} - \frac{1}{3} = \frac{1}{9}$ ✓

اگر سمپل های اوهایو را بر اساس نژاد تقسیم کنیم (white: dd ۲، black: d) خواهیم داشت که $\text{gain} = \frac{4}{9} - \frac{4}{9} = 0$

درخت نهایی با ارتفاع 2



2.3.b

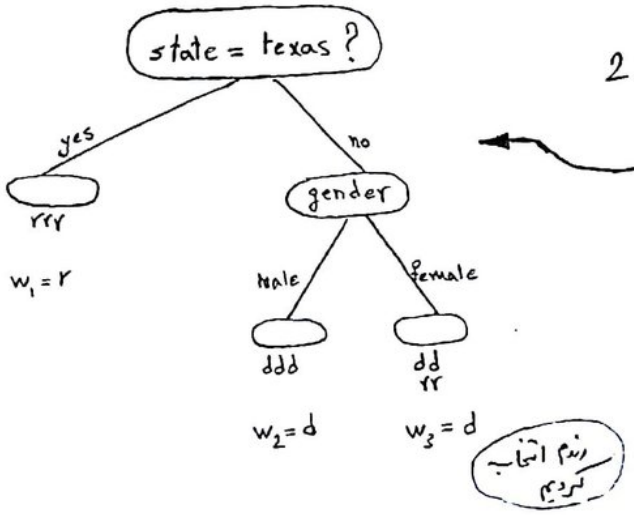
طبق بخش قبل برای اولین انتخاب داریم:

- جنسیت ← $gain = \frac{1}{50}$
- نژاد ← $gain = \frac{1}{50}$
- مکان کالیفرنیا ← $gain = \frac{1}{2} - \left(\frac{4}{10}\left(\frac{3}{8}\right) + \frac{6}{10}\left(\frac{4}{9}\right)\right) = \frac{1}{12}$ ($\frac{5}{12}$) ← california (ddd، ۲) others (dd، ۲۲۲۲)
- مکان تگزاس ← ✓ $gain = \frac{1}{2} - \left(\frac{7}{10}\left(\frac{20}{49}\right)\right) = \frac{3}{14}$ ($\frac{2}{7}$) ← texas (۲۲۲) others (ddddd، ۲۲)
- مکان اوهایو ← $gain = \frac{1}{2} - \left(\frac{3}{10}\left(\frac{4}{9}\right) + \frac{7}{10}\left(\frac{24}{49}\right)\right) = \frac{1}{42}$ ($\frac{10}{21}$) ← ohio (dd، ۲) others (ddd، ۲۲۲۲)

پس ابتدا بر اساس تگزاس تقسیم می‌کنیم. سپس برای تقسیم texas=no داریم:

- جنسیت ← ✓ $gain = \frac{20}{49} - \left(\frac{4}{7}\left(\frac{1}{2}\right)\right) = \frac{6}{49}$ ($\frac{2}{7}$) ← Male (ddd) Female (dd، ۲۲)
- نژاد ← $gain = \frac{20}{49} - \left(\frac{3}{7}\left(\frac{4}{9}\right) + \frac{4}{7}\left(\frac{3}{8}\right)\right) = \frac{1}{294}$ ($\frac{17}{42}$) ← white (dd، ۲) black (ddd، ۲)
- مکان ← $gain = \frac{20}{49} - \left(\frac{4}{7}\left(\frac{3}{8}\right) + \frac{3}{7}\left(\frac{4}{9}\right)\right) = \frac{1}{294}$ ($\frac{17}{42}$) ← california (ddd، ۲) ohio (dd، ۲)

درخت نهایی با ارتفاع 2



2.3.c

به طور کلی می توان به تفاوت های زیر اشاره کرد

- در روش multiway split درخت کوچکتری به دست می آید زیرا درجه هر نود بیشتر است
  اما در روش binary split عمق درخت بیشتر می شود
  اگر عمق درخت را نامحدود فرض کنیم هردو روش دیتای آموزش را کامل مطابق لیبل ها دسته‌بندی می کنند اما واضح است که با محدودیت حداکثر عمق برابر
  روش multiway split به طور کلی بهتر عمل می کند

- روش multiway split درخت‌های قابل تفسیر تری به دست می دهد زیرا dummy node های کمتری ایجاد می کند

- روش binary split به دلیل ارتفاع بیشتر درخت و dummy node های بیشتر ، مقایسه های بیشتری برای دسته‌بندی نیاز دارد
  و از این نظر multiway split می تواند efficient تر باشد

- در روش multiway split ریسک overfitting بیشتر است. مخصوصاً هنگامی که دامنه مقادیر Feature ها بیشتر باشد

- محاسبه information gain به ازای هر split در روش binary split ساده‌تر است و بنابراین فرایند آموزش سریع تری دارد

به طور کلی انتخاب روش مناسب کاملاً وابسته به نوع دیتا و شرایط مسئله است و نمی توان گفت که یکی بهتر از دیگری است

2.4

اگر به اندازه کافی عمق درخت تقسیم را زیاد کنیم می تواند به ازای هر کدام از سمپل های دیتای آموزش برگ ایجاد کند (در بدترین حالت)
و بنابراین ساختار آن به صورتی است که به راحتی می تواند overfit شود و واریانس بالایی دارد

به طور کلی این مشکل در درخت ها را توسط ensemble learning بهبود می دهند. ایده ی این روش آن است که مدل های مستقل (نسبتاً مستقل) ای
آموزش دهیم و سپس برای prediction جواب مدل ها را aggregate کنیم (مثلاً میانگین بگیریم یا majority vote)
طبق قضیه حد مرکزی می دانیم که اگر N متغیر تصادفی iid را میانگین بگیریم واریانس حاصل $\frac{1}{N}$ می شود و از آنجا که $E_{out} = \text{bias} + \text{variance}$
پس عملکرد مدل به طور کلی نیز بهتر می شود

جنگل تصادفی یک روش ensemble کردن درخت هاست. به این صورت که تعدادی درخت را روی بخش های مختلف داده و با پارامتر های مختلف آموزش می دهد
و سعی می‌کند مدل های نسبتاً مستقلی ایجاد کند. در نهایت نیز بسته به مسئله در زمان prediction جواب ها aggregate می شوند

⊛ واضح است که بایاس افزایش پیدا نمی کند زیرا $\text{bias} = \mathbb{E}_D\left[E_{in}\left(h^{(D)}(x)\right)\right]$ و هیچ کدام از مدل ها E_{in} شان بدتر از حالت تکی نمی شود
بنابراین با حفظ دقت درخت ، واریانس کاهش می یابد

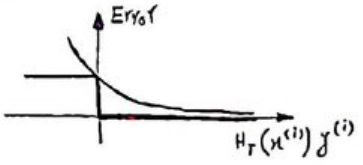
3.1

$$E_T = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left(\underbrace{\text{sign}\left(H_T(x^{(i)})\right)}_{\text{زیرا لیبل‌ها ±1 هستند}} \neq y^{(i)}\right) \qquad H_T(x^{(i)})\,y^{(i)} < 0$$

برای $\text{sign}(H_T(x))$ به صورت نیز 0/1 loss تعریف می‌شود

با توجه به اینکه به ازای هر سمپل داریم $\mathbb{I}\left(H_T(x^{(i)})\,y^{(i)} < 0\right) \leq e^{-H_T(x^{(i)})\,y^{(i)}}$ پس:

$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left(H_T(x^{(i)})\,y^{(i)} < 0\right) \leq \underbrace{\frac{1}{N}\sum_{i=1}^{N} e^{-H_T(x^{(i)})\,y^{(i)}}}_{\text{این مقدار را } E \text{ می‌نامیم}}$$



و بنابراین تابع هزینه نمایی یک upper bound است که اگر آن را min کنیم 0/1 loss نیز کمینه می‌شود. پس در ادامه کار از آن استفاده می‌کنیم

3.2

$$D_{t+1}(i) = D_t(i)\,\frac{e^{-\alpha_t y^{(i)} h_t(x^{(i)})}}{Z_t}$$

رابطه بازگشتی روبه رو را برای $D_{t+1}(i)$ داشتیم:

$$\xrightarrow{\text{جاگذاری}} D_{t+1}(i) = D_1(i) \times \frac{e^{-\alpha_1 y^{(i)} h_1(x^{(i)})}}{Z_1} \times \frac{e^{-\alpha_2 y^{(i)} h_2(x^{(i)})}}{Z_2} \times \cdots \times \frac{e^{-\alpha_t y^{(i)} h_t(x^{(i)})}}{Z_t}$$

$$= \frac{1}{N} \times \frac{e^{-y^{(i)}\left(\sum_{t'=1}^{t} \alpha_{t'} h_{t'}(x^{(i)})\right)}}{\prod_{t'=1}^{t} Z_{t'}}$$

3.3

طبق تعریف داریم که:

$$H_T(x^{(i)}) = \sum_{t=1}^{T} \alpha_t h_t(x^{(i)})$$

$$\xrightarrow[\text{از بخش } 3.1]{\text{جاگذاری در } E \text{ به دست آمده}} E = \frac{1}{N}\sum_{i=1}^{N} e^{-y^{(i)}\sum_{t=1}^{T} \alpha_t h_t(x^{(i)})} = \sum_{i=1}^{N} \underbrace{\frac{1}{N} e^{-y^{(i)}\sum_{t=1}^{T} \alpha_t h_t(x^{(i)})}}_{\circledast}$$

3.4

طبق بخش 3.2 $\circledast = D_{T+1}(i)\prod_{t=1}^{T} Z_t$ پس با جایگذاری آن داریم:

$$E = \sum_{i=1}^{N} D_{T+1}(i) \prod_{t=1}^{T} Z_t$$

$$\xrightarrow[\text{وابسته نیست}]{\text{عبارت دوم به } i} \prod_{t=1}^{T} Z_t \times \underbrace{\sum_{i=1}^{N} D_{T+1}(i)}_{1} = \prod_{t=1}^{T} Z_t$$

تمام $D_t(i)$ ها توزیع هایی هستند که به سمپل ها وزن می‌دهند
پس طبق تعریف $\sum_{i=1}^{N} D_t(i) = 1$ $\forall t$

3.5

$$Z_t = \sum_{i=1}^{N} D_t(i)\, e^{-\alpha_t y^{(i)} h_t(x^{(i)})}$$

$$= \sum_{i=1}^{N} D_t(i)\, e^{-\alpha_t}\, \mathbb{I}\left(h_t(x^{(i)}) = y^{(i)}\right) + \sum_{i=1}^{N} D_t(i)\, e^{\alpha_t}\, \mathbb{I}\left(h_t(x^{(i)}) \neq y^{(i)}\right)$$

$$= e^{-\alpha_t} \underbrace{\sum_{i=1}^{N} D_t(i)\, \mathbb{I}\left(h_t(x^{(i)}) = y^{(i)}\right)}_{1-\varepsilon_t} + e^{\alpha_t} \underbrace{\sum_{i=1}^{N} D_t(i)\, \mathbb{I}\left(h_t(x^{(i)}) \neq y^{(i)}\right)}_{\varepsilon_t}$$

$$= (1-\varepsilon_t)\, e^{-\alpha_t} + \varepsilon_t\, e^{\alpha_t}$$

3.6

مقدار E تا گام t را E_t می نامیم و داریم

طبق بخش 3.4 $E_t = \prod_{t'=1}^{t} Z_{t'}$

طبق بخش 3.5 $Z_t = (1-\varepsilon_t)\, e^{-\alpha_t} + \varepsilon_t e^{\alpha_t}$

$$\longrightarrow E_t = \prod_{t'=1}^{t} \left((1-\varepsilon_{t'})\, e^{-\alpha_{t'}} + \varepsilon_{t'} e^{\alpha_{t'}}\right)$$

حال باید مشتق E_t را بجسب α_t حساب کنیم تا مقدار بهینه را برای آن بدست آوریم

با توجه به اینکه در هر مرحله به صورت greedy عمل می کنیم، وقتی h_t، α_t را به دست می آوریم h_j و α_j $j<t$ ثابت خواهند بود پس

$$E_t = \underbrace{\left(\prod_{t'=1}^{t-1} \left((1-\varepsilon_{t'})\, e^{-\alpha_{t'}} + \varepsilon_{t'} e^{\alpha_{t'}}\right)\right)}_{C} \times \left((1-\varepsilon_t)\, e^{-\alpha_t} + \varepsilon_t e^{\alpha_t}\right)$$

$$\longrightarrow \frac{\partial E_t}{\partial \alpha_t} = C(\varepsilon_t - 1)\, e^{-\alpha_t} + C\varepsilon_t e^{\alpha_t} = 0 \xrightarrow{\times e^{\alpha_t}} \varepsilon_t - 1 + \varepsilon_t e^{2\alpha_t} = 0$$

$$\longrightarrow \alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

بنابراین به ازای هر t نشان دادیم که به خاطر انتخاب greedy و ثابت بودن گام های قبلی مقدار α_t به این صورت به دست می آید