# Gradient boosting

LuK-tutkielma
Oona Polviander
2591588
Matemaattisten tieteiden yksikkö
Oulun yliopisto
Syksy 2021

# Sisältö

# Johdanto

Tutkielmassa on käytetty pääasiassa teosta [1]. Tässä tutkielmassa käsittelemme gradient boosting algoritmia, jonka on kehittänyt Friedman vuonna 2001. Algoritmi on yksi lähestymistapa koneoppimiseen eritoten ohjattuun oppimiseen. Nykyään algoritmia käyttää muun muassa hakukoneet mutta sitä voi myös soveltaa esimerkiksi kuvantunnistukseen, DNA tutkimukseen, automaattiseen tekstin kääntämiseen, saneluun ja markkinahintojen ennustamiseen.

Koneoppiminen on yksi tietotekniikan osa-alueista. Se käsittelee ohjattua ja ohjaamatonta oppimista. Käymme luvussa 1 läpi koneoppimista, mallinnusta ja päätöspuiden teoriaa.

Luvussa 2 käymme teoreettisemmin läpi funktion gradienttia sekä ääriarvon laskemista.

Luku 3 tarkastelemme algoritmin syntyä ja sen matemaattista taustaa. Algoritmi on myös kokonaisuudessaan esillä kappaleessa 3.2

Luku 4 käy suppean datan avulla läpi algoritmin toimintaa ja näyttää, miten ennusteet syntyvät datan ulkopuolella oleville kohteille.

Luku 5 tuo esiin koneoppimisen käytön eri puolia lyhyesti.

# 1 Koneoppiminen

## 1.1 Johdanto

Tuotamme jatkuvasti valtavan määrän dataa tekemisillämme. Arkiset valintamme, kuten, mitä ostamme kaupasta, sekä viimeisimpien tutkimusten tulokset tuottavat dataa ja tietoa paljon enemmän kuin osaamme sitä hyödyntää. Tietomäärän paljous aiheuttaa kuitenkin ongelman. Oikean tiedon löytäminen käy vaikeammaksi datan määrän kasvaessa.

Noin 3000 eaa. syntyi ensimmäiset kirjoutusjärjestelmät ja ihmisen matka rajattoman datan määrään alkoi. Jo varhain kirjoitimme ylös arjen asioitamme, kuten kaupankäyntiä ja sopimuksia. Näin varmistettiin, että tieto olisi mahdollista myöhemmin katsoa ja kertoa jollekin ulkopuoliselle.

Myöhemmin kehitettiin kirjastojärjestelmiä. Niissä tietoa säilytettiin ja järjestettiin aiheen tai muun kriteerin mukaan, mikä johti luokitusjärjestelmien syntyyn. Globalisaation myötä pienempiä kirjastoja ja tietoarkistoja yhdistyi ja datan määrä kasvoi entisestään.

Lopulta mikään määrä paperia ja kirjoja ei kykene kattamaan sitä tiedon määrää, jota maailmassa liikkuu. Tämä johti tietokoneiden kehitykseen. Ongelmana oli kuitenkin se, miten saada kone ajattelemaan samalla tavalla kuin ihminen ja yhdistelemään asioita oikean tiedon löytämiseksi.

Nykypäivänä, kun haluat etsiä jotain tietoa, kaivat puhelimen taskustasi ja laitat mieleiseen hakukoneeseen asian, mistä haluat tietoa. Jo ennen kuin olet kirjoittanut hakusanasi loppuun kone ehdottaa sinulle mahdollisa vaihtoehtoja hakusanoiksi. Se myös osaa ennakoida mahdolliset kirjoitusvirheet, tulkiten näppäilytottumuksiasi.

Kun painat hae-nappia, kone tarjoaa sinulle ensimmäisenä juuri sen tiedon, mitä etsit. Kaiken datan keskeltä kone osaa tulkita hakusanasi. Se yhdistää käyttämäsi hakusanan dataan ja valitsee niistä sopivimman käyttämällä hakuhistoriaa, sijaintia ja kaikkea tietoa, mitä se vain saa sinusta ja muista samankaltaisista ihmisistä irti. Taustatietojen avulla kone pystyy antamaan sopivimman hakutuloksen.

Gradient boosting on yksi tällaisista algoritmeistä.

## 1.2 Mallinnus

Koneoppiminen pyrkii löytämään kaavoja ja ymmärtämään ympärillämme olevien systeemien toimintaa. Tähän tarvitaan algoritmeja, jotka kykenevät parhaalla mahdollisella tavalla vastaamaan todellisuutta ja erottamaan relevantin tiedon valtavista määristä dataa.

Yksi koneoppimisen haaroista on ohjattu oppiminen. Data syötetään funktioon, jonka avulla sitä käsitellään. Käsittelyn jälkeen tulkitaan tuloksia ja vertaillaan niitä arvioihin. Syötteiden ja tulosten perusteella on tarkoitus luoda mallinnus, joka kykenee tuottamaan luotettavia tuloksia syötteelle alkuperäisen datan ulkopuolelta.

Ohjattua oppimista on hyödynnetty esimerkiksi biologian alalla genetiikassa ja perinnöllisyystutkimuksessa. Tutkimalla DNA:ta voimme etsiä yhteyksiä genettisiin sairauksiin ja kehittää mallin, jonka avulla voimme ennustaa geneettisten tautien ilmentymistä. Syötteinä käytetään DNA:ta ihmisiltä, joilla on joku geneettinen sairaus. Tulosten puolella ovat taas itse geneettiset sairaudet. Lopulta voidaan käyttää ennustavaa mallia DNA:n ja geneettisen sairauden välillä.

Ohjatun oppimisen sovelluksia löytyy tieteen muiltakin aloilta niin teknologiasta kuin yhteiskunnasta. Koska ohjattu oppiminen vie koneelta paljon tehoa ja muistia, on algoritmien kehitys ollut välttämätöntä. Tällöin tulosten tarkkuus ei kärsi.

Kuten jo edellä mainittiin, ohjattu oppiminen perustuu syötettyjen arvojen ja tulosten väliseen funktion oppimiseen. Voimme ajatella syötteet vektoreina $x = (x_1, ..., x_n)$ ja tuloksia vektorina $y = (y_1, .., y_n)$

$$Imput : (x_1, ..., x_n) \Rightarrow Funktio \Rightarrow Output : (y_1, ..., y_n)$$

Luokittelun puolella tulokset jaotellaan kahteen eri ryhmään. Tällöin ne voivat esimerkiksi olla $y \in \{Kyllä, Ei\}$. Regression puolella tulos on reaaliluku $y \in \mathbb{R}$. Tarkastelemme tässä tutkielmassa vain yksinkertaisia tapauksia regressiosta avaruudessa $D = \{x_i, y_i\} \in X \times Y\}_{i=1}^{n}$.
Regression tapauksessa funktioina käytetään yleensä tappiofunktiota. Yleisesti käytössä on tappion neliön funktio

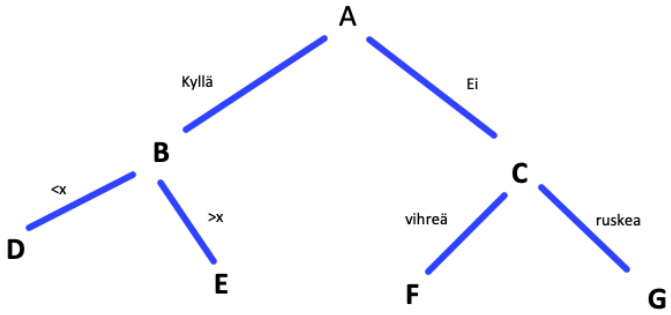
$$L(y, y') = \frac{1}{2}(y - y')^2,$$

missä $y$ on havainto ja $y'$ ennuste. Mikäli halutaan vahvaa näyttöä poikkeavista tuloksista käytetään tappiofunktiona absoluuttista tappiota
$L(y, y') = |y - y'|$. Tutkielman esimerkeissä käytämme neliöllistä tappiota.

Luokittelussa käytetää yleensä joko exponentiaalista tai logistista funktiota. Näistä logistinen funktio on enemmän käytössä gradient boosting algoritmissa. Exponentiaalinen funktio on taas hyvin yleinen adaptive boosting algoritmissa.

## 1.3 Päätöspuut

Päätöspuut ovat yksi ohjatun oppimisen menetelmä. Niillä pyritään vastaamaan yhden muuttujan kysymyksiin ja tekemään sen pohjalta ennusteita.




Kuva 1: Päätöspuu

Yllä olevassa kuvassa puu muodostuu yhden muuttujan jakamisesta osiin. Kohta A jaetaan ensin kahteen vastaamalla kyllä tai ei. Kyllä vastaukset jaetaan kahtia vertaamalla niiden suuruutta valittuun arvoon x kohdassa B. C-kohta jakautuu samalla tavalla kuin A vastaamalla yhden muuttujan kysymykseen: onko näyte vihreä vai ruskea.

Yksistään päätöspuut eivät ole kovin tarkkoja mutta, kun niitä kootaan monia yhteen kuten satunnaismetsä-mallissa, saadaan hyvinkin tarkkoja ennusteita.

Satunnaismetsä-mallissa puiden kasautuminen ennusteen saamiseksi on paljon satunnaisempaa kuin boosting algoritmeissa. Malli rakentaa suuria rönsyileviä puita ja kasaa niitä päällekäin löytääkseen yhtäläisyyksiä ja eroja. Se käyttää paljon päätöspuiden sääntöjä ja algoritmiä pohjanaan mutta hyödyntää satunnaisuutta puiden muodostuksessa enemmän kuin päätöspuiden algoritmissa.

Päätöspuumenetelmän sovelluksia kuten boostrap-menetelmää hyödynnetään esimerkiksi biologian alalla sukupuiden muodostamiseen.

Puuta, jossa on vain muutama lehti, kutsutaan kantapuuksi. Näitä käytetään paljon etenkin gradient boosting sekä adaboosting algoritmeissa. Yleensä myös pyritään että puun tarkkuus on vain hieman parempi kuin täysin satunnainen arvaus. Tällaisia puita kutsutaan heikoiksi puiksi.

Satunnaismetsämalli pyrkii löytämään vahvimman ja parhaimman puun kun taas boostingmallit pyrkivät muokkaamaan heikon mallin tarkaksi.

# 2 Teoriaa

## 2.1 Gradientti

Kun gradient boosting algoritmia tutkitaan teoreettiselta kannalta, pitää huomioida gradientti ja suunnattu derivaatta.

**Määritelmä 2.1.** Olkoon $f : D \subset \mathbb{R}^n \to \mathbb{R}$ differentoituva funktio  Vektoria

$$\text{grad } f(x) = (\partial_1 f(x), \partial_2 f(x), ..., \partial_n f(x))$$

kutsutaan funktion $f$ gradientiksi pisteessä $x \subset D$. Usein merkitään

$$\text{grad } f(x) = \nabla f(x)$$

**Määritelmä 2.2.** Olkoon $f : D \subset \mathbb{R}^n \to \mathbb{R}$ differentoituva funktio sekä $\bar{a} \subset D$ ja $\bar{v} \in \mathbb{R}^n$ jokin yksikkövektori. Jos raja-arvo

$$\partial_{\bar{v}} f(\bar{a}) = \lim_{t \to 0} \frac{f(\bar{a} + t\bar{v}) - f(\bar{a})}{t}$$

on äärellisenä olemassa, niin sitä kutsutaan funktion $f$ suunnatuksi derivaataksi suuntaan $\bar{v}$ pisteessä $\bar{a}$. Suunnatun derivaatan voi myös merkitä

$$\frac{\partial f}{\partial \bar{v}}(\bar{x}).$$

**Määritelmä 2.3.** Funktiolla $f : D \subset \mathbb{R}^n \to \mathbb{R}$ on *paikallinen minimiarvo* eli *lokaali minimiarvo* pisteessä $\bar{a} \in D$, jos on olemassa sellainen pisteen $\bar{a}$ ympäristö $\mathsf{N}$, että $f(x) \geq f(\bar{a})$ kaikilla $\bar{x} \in \mathsf{N}$. Piste $\bar{a}$ on tällöin funktion $f$ paikallinen eli *lokaali minimikohta*. Paikallista minimikohtaa voidaan kutsua myös ääriarvoksi.

**Lause 2.4.** *Jos funktiolla $f : D \subset \mathbb{R}^n \to \mathbb{R}$ on paikallinen ääriarvokohta pisteessä $\bar{a}$, jossa $f$ on differentioituva, niin $\partial_{\bar{v}} f(\bar{a}) = 0$ kaikilla yksikkövektoreilla $\bar{v} \in \mathbb{R}^n$.*

*Todistus.* Olkoon $\bar{a} \in D$ esimerkiksi paikallinen maksimikohta ja $\bar{v} \in \mathbb{R}^n$ mielivaltainen yksikkövektori. Tällöin

$$f(\bar{a} + t\bar{v}) - f(\bar{a}) \leq 0,$$

kun $t$ on riittävän pieni. Nyt on voimassa epäyhtälöt

$$\lim_{t \to 0^+} \frac{f(\bar{a} + t\bar{v}) - f(\bar{a})}{t} \leq 0$$

ja

$$\lim_{t \to 0^-} \frac{f(\bar{a} + t\bar{v}) - f(\bar{a})}{t} \geq 0$$

Koska f on differentoituva, kyseiset raja-arvot ovat samat ja siten

$$\lim_{t \to 0} \frac{f(\bar{a} + t\bar{v}) - f(\bar{a})}{t} = \partial_{\bar{v}} f(\bar{a}) = 0.$$

□

# 3 Algoritmi

## 3.1 Algoritmin pohja

Gradient boosting algoritmin perusperiaate on yksi kerrallaan lisätä uusia päätöspuita malliin ja sillä korjata edellisen mallin virheet. Lähdetään liikkeelle kantapuusta, joka on hieman tarkempi kuin täysin satunnainen arvaus ja lisätään uusia puita kunnes päästään haluttuun tulostarkkuuteen.

Boosting algoritmeja on myös muitakin kuin gradient boosting. Esimerkiksi adaptive boosting on toimii hyvin samalla periaatteella kuin gradient boosting. Algoritmien pohja on sama mutta käyttö kuitenkin erilainen. Molemmilla algoritmeilla on omat tapansa ratkaista pohja-algoritmin yhtälö (2).

Olkoon käytössämme M kappaletta heikkoja päätöspuita. Summa näistä heikoista malleista $\{f_m\}_{m=1}^{M}$ voidaan kirjoittaa seuraavaan muotoon

$$f(x) = \sum_{m=1}^{M} \alpha_m f_m(x) \tag{1}$$

jossa $\{\alpha_m \in \mathbb{R}\}_{m=1}^{M}$ on painottava tekijä ja $f_m(x)$ on kunkin mallin laskettu ennuste.

Gradient boosting käyttää erilaisia yhtälöitä riippuen datasta. Luokitteluun sopii parhaiten logistinen tai exponentiaalinen tappiofunktio kun taas regressioon sopii parhaiten lineaarisesta regressiosta tuttu pienimmän neliösumman menetelmä.

Olkoon funktio $l : y \times y \longrightarrow \mathbb{R}^+$ valittu tappiofunktio käytössä olevalle datalle $D = \{x_i, y_i\} \in X \times Y\}_{i=1}^{n}$.

Koska algoritmi haluaa minimoida valitsemansa tappiofunktion saadaan yhtälö (1) muotoon

$$f(x) = \arg \min_{\alpha_m, f_m(x)} \sum_{i=1}^{n} l\big(y_i, \alpha_m f_m(x_i)\big) \tag{2}$$

jossa $y_i$ on datasta D havaittu arvo ja $\alpha_m f_m(x_i)$ ennustettu arvo, jossa on huomioitu painottava tekijä.

Boostingmetodien toistuvuus ja jäljiteltävyys tuovat vaihtoehdon yhtälön (2) ratkaisemiselle. Lisäämällä uusia malleja yhtälöön, malli $f_m(x)$ rakentaa aina edellisen mallin $m - 1$ päälle. Uusi malli pyrkii korjaamaan edellisen mallin suurimmat virheet.

$$\arg \min_{\alpha_m, f_m(x)} \sum_{i=1}^{m} l\big(y_i, \sum_{l=1}^{m-1} \alpha_l f_l(x_i) + \alpha_m f_m(x_i)\big) \tag{3}$$

Uusi malli $m$ parantaa vanhan mallin $m-1$ tulosta painottavan tekijän $\alpha_m$ avulla. Tavoitteena on pienentää tulosten harhoja ja pitää varianssi lähes muuttumattomana. Boosting aloitetaan yleensä kantapuusta, jossa saattaa olla vain yksi lehti. Tällöin puun ennusteiden harha on hyvin suuri ja varianssi puolestaan erittäin matala. Mitä enemmän puita lisätään sitä pienemmäksi saadaan harha.

Tässä kohdassa adaptive boosting ja gradient boosting erovat toisistaan. Adaptive boosting pyrkii muokkaamaan ja skaalaaman edellisen mallin ennen uuden mallin lisäämistä. Se sopii parhaiten binäärisiin luokittelun ongelmiin $y = \{-1, 1\}$ ja keskittyy suoraan yhtälön (3) ratkaisemiseen.

Koska adaptive boosting ei välttämättä anna hyviä arvioita regressiotehtäville, tarvitaan erilainen lähestymistapa yhtälön (3) ratkaisemiselle.

Gradient boosting pyrkii ratkaisemaan tehtävän hyödyntämällä tappiofunktion derivaattaa. Kun lisäämme uuden mallin $f_m$ ennusteeseen, käyttämällä pienimmän neliön summan tappiofunktiota $l$ näytteelle $(x, y)$ saadaan algoritmi kirjoitettua seuraavasti

$$\begin{aligned} L(y, f(x)) &= \frac{1}{2}(y - f(x))^2 \\ &= \frac{1}{2}(y - \sum_{l=1}^{m-1} \alpha_l f_l(x) - \alpha_m f_m(x))^2 \\ &= \frac{1}{2}(r_m(x) - \alpha_m f_m(x))^2 \end{aligned}$$

jossa $r_m(x) = y - \sum_{l=1}^{m-1} \alpha_l f_l(x)$ on jäännösresiduaali mallin $m-1$ ennusteesta näytteelle $x$. Tämä lähestymistapa on pienimmän neliön regressiivinen boosting. Yhtälön (3) ratkaiseminen tällä menetelmällä vaatisi paljon kärsivällisyyttä ja tarkkaa menettelyä päätöspuiden lehtien nimeämisessä. Voimme kuitenkin vielä lähteä eri tavalla ratkaisemaan yhtälöä (3).

Kirjoitetaan tappiofunktion minimointi seuraavasti

$$\hat{f} = \arg\min_f l(f) = \min \sum_{(x,y)\in D} l(y, f(x)) \tag{4}$$

Lyhyemmin kirjoitettuna

$$\hat{f} = \sum_{m=0}^{M} h_m \qquad , h_m \in \mathbb{R}^n \tag{5}$$

$h_0$ on ensimmäinen ennuste ja $h_1, ...h_m$ ovat uusia ennnusteita riippuen valitusta algoritmista.

Gradient boosting käyttää hyödyksi gradientin laskua. Siinä $h_m$ on laskettu seuraavalla tavalla

$$h_m = -p_m g_m \tag{6}$$

jossa $p_m$ on skalaari ja $g_m \in \mathbb{R}^n$ on funktion $L(f)$ gradientti $f$:n suhteen pisteessä $\hat{f} = \sum_{l=1}^{m-1} p_l h_l$

$$g_m^i = \left[ \frac{\partial}{\partial y'} l(y^i, y') \right], \tag{7}$$

jossa $y' = \hat{f}$. Määritelmän 2.1 mukaan $g_m^i$ on funktion $l$ osittaisderivaatta. Skalaari $p_m$ määrää kuinka iso askel otetaan pitkin negatiivista gradienttia $-g_m$,. Skalaari valitaan siten, että käytetty tappiofunktio $l$ saataisiin minimoitua, toisin sanoen

$$p_m = \arg\min_{p_m} l(\hat{f} - p_m g_m) \tag{8}$$

Edellä laskettiin gradientti opetettavan datan näytteille. Jotta voimme tehdä ennustuksia uusista näytteistä, pitää arvioida negatiivisen gradientin suuntaa regressiomallin $g_m$ avulla. Funktio mallintaa heikoista päätöspuista muodostunutta dataa.

$$g_m = \arg\min_{g} \sum_{i=1}^{n} \big( - g_m^i - g(x^i) \big)^2 \tag{9}$$

Uuden näytteen ennustaminen toimii siis seuraavasti:
Aloitetaan alustavalla ennustuksella $p_0 \in \mathbb{R}$. Negatiivisen gradientin funktiolle $l$ määrittää regressiomalli $g_m$, joka on sovitettu datan mukaan. Skalaari $p_m$ määrää optimaalisen askeleen pituuden minimoimalla funktion $l$ suuntaan $g_m$.

$$f(x) = p_0 + \sum_{m=1}^{M} p_m g_m(x) \tag{10}$$

Algoritmi on esitetty kappaleessa 3.2. Se on määritelty silloin, kun (a) aloitusmalli eli vakio, joka minimoi annetun tappiofunktion, (b) tappiofunktion gradientti ovat molemmat olemassa ja määritelty.

Optimaalinen askeleen pituus saadaan yleensä laskettua pienimmän neliösumman menetelmällä mutta myös muita menetelmiä juurien löytämiseen käytetään.

Opetusnopeus $\mu \in [0, 1]$ lisätään, jotta askeleen pituus $p_m$ ei olisi liian suuri. Tällä vältetään näytteiden ylisovittamista.

## 3.2 Algoritmi kokonaisuudessaan

1. $L = \{(x_i, y_i) \in X \times Y\}_{i=1}^{n}; l; H; M$

2. $f_0 = p_0 = \arg\min_{p \in \mathbb{R}} \sum_{i=1}^{n} l(y^i, p)$

3. *from m=1 to M*

4. $$g_m^i = \left[ \frac{\partial}{\partial y'} l(y^i, y') \right]_{y'=\hat{f}} \qquad \forall i \in \{1, ..., n\}$$

5. $$g_m = \arg\min_{g \in H} \sum_{i=1}^{n} \big( - g_m^i - g(x^i) \big)^2$$

6. $$p_m = \arg\min_{p \in \mathbb{R}} \sum_{i=1}^{m} l\Big( y^i, f_{m-1}(x^i) + p_m g_m(x^i) \Big)$$

7. $$f_m(x) = f_{m-1}(x) + \mu p_m g_m(x)$$

   .

   .

   .

   $f_M(x)$ output

# 4 Esimerkki

Mallinnetaan regressiivistä dataa gradient boosting algoritmin avulla. Käytämme tulevissa esimerkeissä yksinkertaista dataa. Data merkitään yleensä $\{(x_i, y_i)\}_{i=1}^n$. Tässä $x_i$ on meidän tiedossa oleva data ja $y_i$ on datan havainnot. Tarvitsemme datan lisäksi tappiofunktion $L(y_i, F(x))$

Seuraavassa taulukossa on kolmen kissan sukupuoli, turkin väri ja paino. Ennustetaan gradient boosting algoritmin avulla painoa kissoille. Tässä esimerkissä $x_i$ on sukupuoli sekä väri ja $y_i$ on kissan paino.

| | Sukupuoli | Väri | Paino kg |
|---|---|---|---|
| Kissa 1 | Naaras | Ruskea | 4,5 |
| Kissa 2 | Uros | Valkoinen | 3,9 |
| Kissa 3 | Naaras | Valkoinen | 3,6 |

Määritetään ensimmäisenä funktion $L$ gradientti

$$\begin{aligned}\frac{\partial L(y, y')}{\partial y'} =& \frac{\partial}{\partial y'}\frac{1}{2}(y - y')^2 \\ =& \frac{2}{2} \cdot (y - y') \cdot (-1) \\ =& -(y - y')\end{aligned}$$

$y$ on siis meidän datan näyttämä paino ja $y'$ ennustettu paino.

Tarvitsemme aloitusarvon $F_0 = y'$, joka muodostaa kantapuun. Lasketaan nyt $y'$ hyödyntäen edellä laskettua derivaattaa yhtälön (4) avulla

$$\begin{aligned}F_0 =& \arg\min \sum_{i=1}^{n} l(y_i, y') \\ =& \arg\min \left\{ \frac{1}{2}(4,5\text{kg} - y')^2 + \frac{1}{2}(3,9\text{kg} - y')^2 + \frac{1}{2}(3,6\text{kg} - y')^2 \right\}\end{aligned}$$

Hyödyntämällä lausetta 2.4 sekä pienimmän neliösumman menetelmää saamme aloitusarvon $F_0$

$$\begin{aligned}0 =& -(4,5\text{kg} - y') - (3,9\text{kg} - y') - (3,6\text{kg} - y') \\ 3y' =& (4,5 + 3,9 + 3,6)\text{kg} \\ y' =& \frac{4,5 + 3,9 + 3,6}{3}\text{kg} \\ y' =& 4,0\text{kg}\end{aligned}$$

Alustava ennuste $F_0$ on siis painojen keskiarvo. Ennuste kertoo, että kaikki kissat painavat $4,0$kg.

Jokaiselle kissalle on laskettava residuaali $r_m^i$ yhtälön (7) avulla seuraavasti

$$r_m^i = -\left[\frac{\partial}{\partial y'} l(y^i, y')\right]$$

$i$ kertoo minkä kissan painon residuaalia ollaan laskemassa ja $m$ kertoo rakentuvan puun numeron. Käytetään funktiosta $L$ laskettua derivaattaa

$$-(-(y - y') = y - y'$$

Kissalle 1 residuaali $r_1^1$ voidaan laskea seuraavasti

$$y - y' = (4,5 - 4,0)\text{kg} = 0,5\text{kg}$$

Samalla tavalla saadaan myös kissalle 2 ja 3 laskettua $r_1^2$ ja $r_1^3$

Lisätään lasketut residuaalit taulukkoon

| | Sukupuoli | Väri | Paino kg | $r_1^i$ |
|---|---|---|---|---|
| Kissa 1 | Naaras | Ruskea | 4,5 | 0,5 |
| Kissa 2 | Uros | Valkoinen | 3,9 | -0,1 |
| Kissa 3 | Naaras | Valkoinen | 3,6 | -0,4 |

Gradient boosting käyttää puun muodostamiseen yleensä enemmän lehtiä. Koska datassa on vain kolme näytettä, piirrettävällä puulla voi olla vain kaksi lehteä. Jos jokainen kissa laitettaisiin omaan lehteensä, ei algoritmia tarvittaisi, koska dataa käytetään sellaisenaan. Lehtien määrä tulee siis aina olla pienempi kuin itse näytteiden määrä.

Muodostuneita lehtiä kutsutaan terminaalialueiksi $R_{jm}$. Ne määräävät, miten data jaetaan pienempiin osiin. Puu voidaan muodostaa seuraavasti turkin värin perusteella. Valkoiset kissat menevät samaan lehteen ja ruskea kissa toiseen lehteen.
Lehden 1 $R_{1.1}$ residuaali on $0,5$. Lehden 2 $R_{2.1}$ residuaalit ovat $-0,1$ ja $-0,4$. Lehtien nimeäminen on tärkeää jatkaessa algoritmia. Siten pystymme seuraamaan näytteiden sijoittumista seuraavissa puissa.

Seuraavaksi määritetään terminaalialueelle tulosarvo, joka saadaan hyödyntämällä yhtälöitä (4) ja (8).

$$y'_{jm} = \arg\min_{y' \in \mathbb{R}} \sum_{x_i \in R_{jm}} l\Big(y_i, f_{m-1}(x_i) - y'\Big)$$

Tässä $y'_{jm}$ kertoo lehden loppullisen tulosarvon. Summa tarvitaan niihin lehtiin, johon on päätynyt useampia residuaaleja. Lehden $R_{1.1}$ kohdalla lasku menee seuraavasti

$$\begin{aligned} y'_{1.1} &= \arg\min \frac{1}{2}(y_1 - f_0(x_1) - y')^2 \\ &= \arg\min \frac{1}{2}(4,5 - 4,0 - y')^2 \\ &= \arg\min \frac{1}{2}(0,5 - y')^2 \end{aligned}$$

Käyttämällä taas funktion $L$ derivaattaa ja minimiä saadaan

$$0 = -(0,5 - y') \Rightarrow y'_{1.1} = 0,5$$

Koska lehhdessä $R_{1.1}$ on vain yksi arvo, pysyy lehden lopullinen arvo samana kuin sen residuaali.

Lehdelle $R_{2.1}$ lasku eroaa hieman. Koska kaksi näytettä päätyi samaan lehteen, käytetään summausta.

$$\begin{aligned} y'_{2.1} &= \arg\min_{y'} \sum_{x_i \in R_{ij}} l\Big(y_i, f_{m-1}(x_i) - y'(x_i)\Big) \\ &= \arg\min_{y'} \sum_{x_i \in R_{ij}} \frac{1}{2}(y_i - f_0(x_i) - y')^2 \\ &= \arg\min \left(\frac{1}{2}(y_2 - f_0(x_2) - y')^2 + (y_3 - f_0(x_3) - y')^2\right) \\ &= \arg\min \left(\frac{1}{2}(3,9 - 4,0 - y')^2 + (3,6 - 4,0 - y')^2\right) \\ &= \arg\min \left(\frac{1}{2}(-0,1 - y')^2 + (-0,4 - y')^2\right) \end{aligned}$$

Samalla tavalla kuin edellisessä kohdassa käytetään derivaatan ketjusääntöä

$$\begin{aligned} 0 &= -\left(-0,1 - y'\right) - \left(-0,4 - y'\right) \\ y'_{2.1} &= \frac{-0,1 - 0,4}{2} \\ &= -0,25 \end{aligned}$$

Huomataan, että lehtiin päätyneiden residuaalien keskiarvo on sama kuin saatu tulosarvo.

Jotta gradient boosting algoritmi pystyisi kehittämään itseään, tarvitsemme opetusnopeuden $\mu \in [0, 1]$. Tällöin jokainen rakentama puu kehittyy tarkemmaksi arvioksi. Opetusnopeus tulee valita siten, että terminaalialueilla olisi mahdollisimman vähän ylioppimista. Valinnassa tulee myös huolehtien samalla sopivasta etenemisnopeudesta.

Aloitetaan uuden ennusteen $F_1$ laskeminen käyttämällä yhtälöä (10) ja lisäämällä siihen oppimisarvon $\mu$. Valitaan tähän esimerkkiin oppimisarvoksi $\mu = 0, 2$.

$$f_m(x) = f_{m-1}(x) + \mu y'_{jm}(x)$$

$f_{m-1}(x) = f_0(x) = 4, 0$ ja $y'_{jm}$ ovat edellisessä kohdassa lasketut tulosarvot. Summamerkintää tulee käyttää kaavassa, mikäli sama näyte on päätynyt useampaan lehteen. Muuten jokaiselle näytteelle lasketaan uusi ennuste kaavan mukaan.

$$\begin{aligned} f_1(x_1) &= 4, 0\text{kg} + 0, 2 \cdot 0, 5\text{kg} = 4, 1\text{kg} \\ f_1(x_2) &= 4, 0\text{kg} + 0, 2 \cdot (-0, 25)\text{kg} = 3, 95\text{kg} \\ f_1(x_3) &= 4, 0\text{kg} + 0, 2 \cdot (-0, 25)\text{kg} = 3, 95\text{kg} \end{aligned}$$

| | Sukupuoli | Väri | Paino kg | $r_1^i$ | $F_1$ |
|---|---|---|---|---|---|
| Kissa 1 | Naaras | Ruskea | 4,5 | 0,5 | 4,1 |
| Kissa 2 | Uros | Valkoinen | 3,9 | -0,1 | 3,95 |
| Kissa 3 | Naaras | Valkoinen | 3,6 | -0,4 | 3,95 |

Gradient boosting toistaa tätä sarjaa $M$ kertaa. Yleensä $M$ on 100 tai suurempi. Valitaan tähän esimerkkiin $M = 2$. Lasketaan toisen kerran algoritmi läpi saadaksemme ennusteen $F_2(x)$ kissojen painoille. Lasketaan ensin residuaalit.
Kissalle 1 residuaali $r_2^1$ voidaan laskea seuraavasti

$$y - y' = (4, 5 - 4, 1)\text{kg} = 0, 4\text{kg}$$

Samalla tavalla saadaan myös kissalle 2 ja 3 laskettua $r_2^2$ ja $r_2^3$

| | Sukupuoli | Väri | Paino kg | $r_1^i$ | $F_1$ | $r_2^i$ |
|---|---|---|---|---|---|---|
| Kissa 1 | Naaras | Ruskea | 4,5 | 0,5 | 4,1 | 0,4 |
| Kissa 2 | Uros | Valkoinen | 3,9 | -0,1 | 3,95 | -0,05 |
| Kissa 3 | Naaras | Valkoinen | 3,6 | -0,4 | 3,95 | -0,35 |

Määritetaan uudet terminaalialueet uuden puun rakentamista varten

$$R_{1.2} = 0,4$$
$$R_{2.2} = 0,05; -0,35$$

Määritetään terminaalialueen $R_{1.2}$ tulosarvo

$$\begin{aligned} y'_{1.2} &= \arg\min \frac{1}{2}(y_1 - f_1(x_1) - y')^2 \\ &= \arg\min \frac{1}{2}(4,5 - 4,1 - y')^2 \\ &= \arg\min \frac{1}{2}(0,4 - y')^2 \end{aligned}$$

Laskemalla derivaatta ja minimi saadaan

$$0 = 0,4 + y'_{1.2} \Rightarrow y'_{1.2} = 0,4$$

Määritetään terminaalialueen $R_{2.2}$ tulosarvo

$$\begin{aligned} y'_{jm} &= \arg\min_{y'} \sum_{x_i \in R_{jm}} l\Big(y_i, f_{m-1}(x_i) - y'(x_i)\Big) \\ &= \arg\min_{y'} \sum_{x_i \in R_{jm}} \frac{1}{2}(y_i - f_1(x_i) - y')^2 \\ &= \arg\min \left(\frac{1}{2}(y_2 - f_1(x_2) - y')^2 + (y_3 - f_1(x_3) - y')^2\right) \\ &= \arg\min \left(\frac{1}{2}(3,9 - 3,95 - y')^2 + (3,6 - 3,95 - y')^2\right) \\ &= \arg\min \left(\frac{1}{2}(-0,05 - y')^2 + (-0,35 - y')^2\right) \end{aligned}$$

Laskemalla derivaatta ja minimi saadaan

$$\begin{aligned} 0 =& -(-0,05 - y'_{2.2}) - (-0,35 - y'_{2.2}) \\ y'_{2.2} =& \frac{-0,05 - 0,35}{2} \\ =& -0,20 \end{aligned}$$

Tulosarvojen laskun jälkeen voidaan määrittää jokaiselle kissalle uusi ennuste painosta $F_2(x)$

$$\begin{aligned} f_2(x_1) =& 4,0\text{kg} + 0,2 \cdot 0,5\text{kg} + 0,2 \cdot 0,4\text{kg} = 4,18\text{kg} \\ f_2(x_2) =& 4,0\text{kg} + 0,2 \cdot (-0,25)\text{kg} + 0,2 \cdot (-0,20)\text{kg} = 3,91\text{kg} \\ f_2(x_3) =& 4,0\text{kg} + 0,2 \cdot (-0,25)\text{kg} + 0,2 \cdot (-0,20)\text{kg} = 3,91\text{kg} \end{aligned}$$

Uudet ennusteet voidaan laittaa taulukkoon

| | Sukupuoli | Väri | Paino kg | $r_1^i$ | $F_1$ | $r_2^i$ | $F_2$ |
|---|---|---|---|---|---|---|---|
| Kissa 1 | Naaras | Ruskea | 4,5 | 0,5 | 4,1 | 0,4 | 4,18 |
| Kissa 2 | Uros | Valkoinen | 3,9 | -0,1 | 3,95 | -0,05 | 3,91 |
| Kissa 3 | Naaras | Valkoinen | 3,6 | -0,4 | 3,95 | -0,35 | 3,91 |

Jos meillä olisi tiedot neljännestä kissasta, joka olisi väriltään ruskea, voisimme ennustaa sille painoa. Koska kissa 4 on ruskea, se päätyisi samoihin terminaalialueisiin kuin kissa 1. Tällöin algoritmi ennustaisi sille painoksi $4,18$kg. Ennuste olisi paljon tarkempi, jos puita olisi rakennettu enemmän kuin kaksi. Gradient boosting yleensä muodostaa puita huomioiden myös muut muuttuja datassa, kuten esimerkiksi sukupuolen.

# 5 Pohdintaa

Koneoppiminen on mahdollistanut paljon arkea hyödyntäviä asioita. Tiedonhaussa siitä on suuri apu mutta on myös paljon kohteita, joissa käytämme erilaisia oppivia algoritmeja vain vaikuttaaksemme ihmisten mielipiteisiin tai ostopäätöksiin. Tämä näkyy eniten kohdennetussa mainonnassa.

Algoritmien pyöritys vaatii suuret määrät tehoa ja tilaa tietokoneilta. Tietokoneservereiden ylläpitäminen taas kuluttaa paljon energiaa vain saavuttaakseen kenties paremmat mainoksen esille esimerkiksi sosiaalisessa mediassa. Olisiko tarpeellisempaa suunnata oppivia algoritmeja johonkin muuhun?

Kun puhutaan koneoppimisesta, pyritään opettamaan koneelle syy-seuraus suhteita ja optimoimaan näitä parhaimman tuloksen saamiseksi. Mutta jos katsomme, miten elämä maapallolla pyörii, huomaamme paljon täysin satunnaisia asioita. Niillä ei ole mitään järkevää ja optimaalista pohjaa, mikä tekee niiden opettamisesta koneelle hyvin hankalaa.

Jos haluaisimme, että gradient boosting algoritmi toimisi mahdollisimman tarkasti ja luonnollisesti, joutuisimme yksinkertaisesti vaan hyväksymään sattuman olemassaolon ja varianssi tulisi tällöin nousemaan eikä malli olisi optimaalinen. Jotta kaikki vaihtoehdot olisi huomioitu pitäisi puita piirtää huomattavan suuri määrä, jolloin algoritmin käyttö olisi hidasta eikä se välttämättä tuottaisi optimaalista tulosta.

Nyt jo näemme erilaisissa ennusteissa kuten poliittisissa gallupeissa tai muissa tilastollisissa ennusteissa suuriakin eroja, vaikka ennuste oli valmisteltu suuresta pohjadatasta useammalta vuodelta. Näiden ennusteiden epäonnistumiset ovat sattuman ansiota. Saada tietokone ymmärtämään ja ennustamaan jotain, mikä on ristiriidassa kaiken sen suhteen, mitä se on ohjelmoitu tekemään, voi olla aina täysi mahdottomuus.

# Lähdeluettelo

[1] Arnaud Joly: Exploiting random projections and sparsity with random forests ant gradient boosting methods, PhD thesis 2016