Comprimeren en Beleggen: korte beschrijvingen zijn goud waard

Peter D. Grünwald

December 16, 2008

Universele datacompressie is een belangrijke tak van de informatietheorie. Veel state-of-the-art datacompressieprogramma's zoals *gzip, bzip* enz. zijn voorbeelden van universele datacompressiealgoritmes. Anders dan soms gedacht wordt ligt er een stuk elegante wiskunde ten grondslag aan zulke algoritmes. Het doel is als volgt: stel, we hebben een aftelbaar oneindige reeks codes $\mathcal{C} = \{C_1, C_2, \ldots\}$ ter beschikking om data (computerbestanden) mee te comprimeren. De ene code is bijv. beter in het comprimeren van Engelse text, de tweede is beter in Franse text, de derde is beter in het comprimeren van computercode, de vierde is beter in plaatjes. We denken dat, voor elk bestand, een van de codes het wel goed zal doen, maar we weten niet welke. Een *universele compressiemethode* $\bar{C}$ voor $\mathcal{C}$ is een code die zo werkt dat, *wat voor data we ook krijgen*, $\bar{C}$ die data ongeveer evengoed comprimeert als de code $C \in \mathcal{C}$ die *achteraf gezien* het beste blijkt. *gzip* is bijvoorbeeld een universele compressiemethode relatief tot de set $\mathcal{C}$ van alle compressiemethodes die met behulp van een willekeurig grote eindige automaat geimplementeerd kunnen worden.

Bij *Universeel beleggen* hebben we de beschikking over een reeks beleggingsstrategieen $\mathcal{S} = \{S_1, \ldots, S_m\}$. S_1 zou bijvoorbeeld de beleggingsstrategie gevolgd door het Aegon Mix Fund kunnen zijn, S_2 de strategie van een of ander fonds van Delta Lloyd, S_3 van ABN AMRO enz. Een beleggingsstrategie is een methode om, iedere dag opnieuw, bepaalde aandelen te kopen en andere verkopen, afhankelijk van wat er in het verleden is gebeurd en verdere relevante informatie zoals werkloosheidscijfers e.d. We willen nu een meta-strategie $\bar{S}$ creeren die beschikking heeft tot de strategieen (fondsen) $S_1, \ldots S_m$ en die het voor elkaar krijgt dat, als we enige dagen achtereen beleggen volgens $\bar{S}$, het volgende geldt: *wat er ook op de beurs is gebeurd*, $\bar{S}$ heeft het ongeveer zo goed gedaan als de strategie $S \in \mathcal{S}$ die *achteraf gezien* het beste presteerde, dus $\bar{S}$ is evengoed als de strategie die we, met kennis achteraf, *het liefst hadden willen volgen.*

Het blijkt dat, wiskundig gezien, "universeel beleggen" vrijwel hetzelfde probleem is als "fundamenteel coderen". Dit heeft geleid tot een geheel nieuwe, *niet op kansrekening gebaseerde* aanpak voor het samenstellen van beleggingsfondsen. Het doel van dit project is (a) het bestuderen van de relatie tussen universeel comprimeren en universeel beleggen; en (b) het onderzoeken of met behulp van het recente, succesvolle "switching model" voor universele datacompressie ook fundamenteel betere beleggingsresultaten

behaald kunnen worden. Hoewel er geen aannames worden gedaan over kansverdelingen (we willen universeel zijn voor *alle* mogelijke bestanden/koersen), speelt het kansgerelateerde begrip "entropie" hierbij een fundamentele rol. Voor de geinteresseerden volgt wat gedetailleerdere informatie.

1 Universeel Comprimeren

Stel, we hebben een aftelbaar oneindige reeks codes $\mathcal{C} = \{C_1, C_2, \ldots\}$ ter beschikking om data (een reeks letters van willekeurige lengte) mee te comprimeren. Een code is een functie die datasequenties van willekeurige lengte afbeeldt op de binaire strings $\bigcup_{k\geq 1}\{0,1\}^k$. We zijn geinteresseerd in lossless compression, en beperken ons daarom tot bijectieve codes C, zodat, gegeven een codering $\mathbf{y} = C(\mathbf{x})$ van data $\mathbf{x} = x_1 \ldots x_n$, de oorspronkelijke data $\mathbf{x}$ altijd weer precies kan worden gereconstrueerd via de inverse functie $\mathbf{x} = C^{-1}(\mathbf{y})$. Ons doel is om data zoveel mogelijk te comprimeren, m.a.w. we zouden graag een code C willen gebruiken zodat, voor de data $\mathbf{x}$ die we uiteindelijk krijgen, de lengte $|C(\mathbf{x})|$ van de codering $C(\mathbf{x})$ klein is. Het idee van universele compressie is als volgt: stel dat we denken dat een van de codes in de set $\mathcal{C}$ de data die we gaan krijgen goed zal gaan comprimeren, we weten alleen niet welke. We zouden daarom graag een code $\bar{C}$ willen gebruiken met de eigenschap dat, *wat voor datasequentie* $x_1 \ldots x_n$ *we ook observeren*, $|\bar{C}(x_1 \ldots x_n)| \approx \min_{C\in\mathcal{C}} |C(x_1 \ldots x_n)|$; dus: wat er ook gebeurt, we comprimeren ongeveer evengoed als de code die, *achteraf gezien*, het beste bleek te zijn. Verbazingwekkend genoeg bestaan zulke codes. Een eenvoudig voorbeeld hiervan is de *two-part code*. Het idee van deze code is om eerst een getal j te coderen met behulp van een "standaard" zgn. prefix-vrije code $C_0 : \mathbb{N} \to \bigcup_{k>0}\{0,1\}^k$ voor natuurlijke getallen, en vervolgens de data $\mathbf{x}$ te coderen met de code C_j. De prefix-vrije eigenschap van C_0 zorgt ervoor dat de resulterende twee-delige code decodeerbaar is: gegeven het codewoord $\mathbf{y} = C_0(j)C_j(\mathbf{x})$ kan de oorspronkelijke data $\mathbf{x}$ gereconstrueerd worden. Er bestaan prefix-vrije codes die een een getal j coderen in ongeveer $2\log j$ bits. De two-part code $\bar{C}$ is nu gedefinieerd als de code die elke $\mathbf{x}$ codeert door eerst j te coderen en vervolgens $\mathbf{x}$ te coderen met de code C_j zodanig dat $|C_o(j)C_j(\mathbf{x})| = 2\log j + |C_j(\mathbf{x})|$ geminimaliseerd wordt. $\bar{C}$ is dus een code zodanig

dat, voor alle $\mathbf{x} = x_1, \ldots, x_n$ en alle $C_j \in \mathcal{C}$, $|\bar{C}(\mathbf{x})| \leq |C_j(\mathbf{x})| + 2 \log j$. De universele code $\bar{C}$ heeft dus een constante overhead, die alleen van de index j afhangt maar niet van n. Omdat in het algemeen, voor alle j, $C_j(x_1 \ldots x_n)$ lineair toeneemt in n, is zo'n constante overhead in veel gevallen verwaarloosbaar.

2 Universeel Beleggen

Stel, we hebben toegang tot een aandelenbeurs met N verschillende aandelen, ieder met zijn eigen, dagelijks fluctuerende koers. Laat $\vec{x}_i$ de N-dimensionale vector zijn die de koersen op dag i aangeeft. Een belegger begint op dag 1 met startkapitaal K_1 en verdeelt dat op een of andere wijze over de N aandelen. Zijn keuze kan worden omschreven via een vector $\vec{a}_1 = (a_{1,1}, \ldots, a_{N,1})$: hij koopt $a_{1,1}$ exemplaren aandeel 1, $a_{1,2}$ exemplaren aandeel 2 enz., zodanig dat $\sum_{j=1}^{N} a_{1,j} x_{1,j} = \langle \vec{a}_1, \vec{x}_1 \rangle = K_1$. $\langle \cdot, \cdot \rangle$ staat voor het standaard inproduct, en voor het gemak gaan we ervan uit dat je fractionele aandelen kunt kopen. Op dag 2 zijn sommige aandelen gestegen en andere gedaald; hierdoor is de waarde van de portfolio veranderd in $K_2 = \langle \vec{a}_1, \vec{x}_2 \rangle$. De belegger heeft nu K_2 Euro in bezit en gebruikt dat om zijn portfolio aan te passen, dwz. hij koopt en verkoopt een aantal aandelen. Voor het gemak gaan we ervan uit dat er hierbij geen transactiekosten zijn. Dit resulteert in een nieuwe vector $\vec{a}_2$ van aantallen aandelen. Vervolgens beweegt de beurs weer één dag verder en wordt het bezit van de belegger gelijk aan $K_3 = \langle \vec{a}_2, \vec{x}_3 \rangle$. En zo gaat het verder...

Stel dat de belegger elke dag nog extra relevante informatie (rentestand, werkloosheidscijfers e.d.) ter beschikking heeft, gecodeerd als een M-dimensionale vector $\vec{r} \in \mathbb{R}^M$. Een beleggingsstrategie is een functie S die, voor elke tijdsduur n, en elke investerings-, koers- en verdere informatie-geschiedenis $\vec{a}_1, \vec{x}_1, \vec{r}_1, \ldots, \vec{a}_n, \vec{x}_n, \vec{r}_n$ een herinvesteringskeuze $\vec{a}_{n+1} = S(\vec{a}_1, \vec{x}_1, \vec{r}_1, \ldots, \vec{a}_n, \vec{x}_n, \vec{r}_n)$ maakt. In feite is S dus een functie $S : \bigcup_{n>0}((\mathbb{R}^+)^N \times (\mathbb{R}^+)^N \times \mathbb{R}^M)^n \to (\mathbb{R}+)^N$.

Zo'n beleggingsstrategie kan bijv. een steeds grotere fractie van het kapitaal inzetten op een aandeel dat het in het verleden al goed deed; of juist niet natuurlijk. Elk commercieel verkrijgbaar beleggingsfonds is eigenlijk een bepaalde beleggingsstrategie. Stel dat we kunnen kiezen uit een verzameling mogelijke beleggingsstrategieen $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$. Bijv. S_1 is het fonds van Aegon, S_2 het fonds van Delta Lloyd etc. Een *universele beleggingsstrategie* $\bar{S}$ is een strategie die de strategieen $S_1, S_2, \ldots$ op de een of andere wijze combineert zodanig dat *wat er ook met de aandelenkoersen gebeurt*, $\bar{S}$ bijna evenveel koerswinst oplevert als de strategie in $\mathcal{S}$ die het, *achteraf gezien het beste bleek te doen*. Dus als Aegon het beste was, willen we het minstens zo goed doen als Aegon; als Delta LLoyd het beste was, moeten we het minstens zo goed doen als Delta Lloyd, enz.

Switching De laatste 10 jaar is er een belangrijke nieuwe ontwikkeling geweest in universele data compressie, het zgn. "switching model". Dit levert vaak betere data-compressie op dan andere bestaande methoden. De centrale vraag is: kunnen we dit

switching model uitbreiden naar sequentieel beleggen? En zo ja, wat levert dit (letterlijk) op? Er is een enkel paper dat probeert simpele "switching" datacompressiemethoden over te brengen naar beleggingstrategieen ("switching portfolios", Yoram Singer, 1999). Daarin wordt geclaimd dat, zelfs als transacties niet gratis zijn, "switching" methodes het beter doen op Wall Street dan welk ander algoritme dan ook. Toch is er later niets meer van vernomen. Hoe komt dat? Een mogelijkheid is dat de algoritmes voor "switching" strategieen niet efficient geimplementeerd kunnen worden. Misschien kan de recente doorbraak in het ontwikkelen van snelle "switching" algoritmes hier verandering in brengen?

Dit project behelst het leren van de basistheorie van universele datacompressie en universeel beleggen, en het onderzoeken of switch-strategien tot betere universele strategieen leiden. Hierbij kan niet alleen gedacht worden aan het bewijzen van stellingen, maar ook (of in plaats daarvan) aan analyse van de resultaten van Singer, het uitwerken van voorbeelden, of evt. een toy-versie op de computer implementeren en daarmee enkele experimenten uitvoeren.