Röntgenspektrien sisältämä rakenneinformaatio

Pro Gradu
Turun yliopisto
Fysiikka
2021
Joonas Niemi
Tarkastajat:
Johannes Niskanen
Edwin Kukk

TURUN YLIOPISTO
Fysiikan laitos

Niemi, Joonas Röntgenspektrien sisältämä rakenneinformaatio

Pro Gradu, lxii s.
Fysiikka
Kesäkuu 2021

Röntgenspektroskopisilla menetelmillä on valtavasti sovelluskohteita monien eri alojen parissa. Röntgenspektrien analysoiminen tuottaa paljon tietoa näytteiden rakenteista, mutta kuinka sensitiivisiä spektrit ovat näytteiden rakennejakaumien muutoksille, on epäselvää. Tähän kysymykseen lähdin etsimään vastausta tutkielmassani. Tutkimuksissani pyrin palauttamaan simuloitujen vesimolekyylien tunnetun rakennejakauman rakennejakaumaparametrit niitä vastaavista keskiarvoröntgenspektreistä, jotka määritin koneoppimista hyödyntäen. Keskiarvospektrien laskemiseen koulutin koneoppimismallin ennustamaan röntgenspektrejä simuloiduille vesimolekyyleille lasketuista spektreistä koostuvasta koulutusjoukosta. Käytössäni oli kolmen eri röntgenspektroskopiamenetelmän spektrejä, röntgenabsorptiospektroskopian, röntgenfotoelektronispektroskopian ja röntgenemissiospektroskopian. Näin tarkoituksenani oli myöskin selvittää, että mikä edellä mainituista menetelmistä on sensitiivisin rakennejakauman muutoksiin.

Kappaleissa yksi ja kaksi käyn läpi röntgenspektroskopian sekä koneoppimisen teoriaa, ja kolmannessa kappaleessa esittelen käyttämäni datan, tutkimusmetodini sekä saamani tulokset. Lopuksi ilmaisen ajatuksiani tuloksiin liittyen johtopäätösten muodossa kappaleessa neljä.

Sisällys

Johdanto

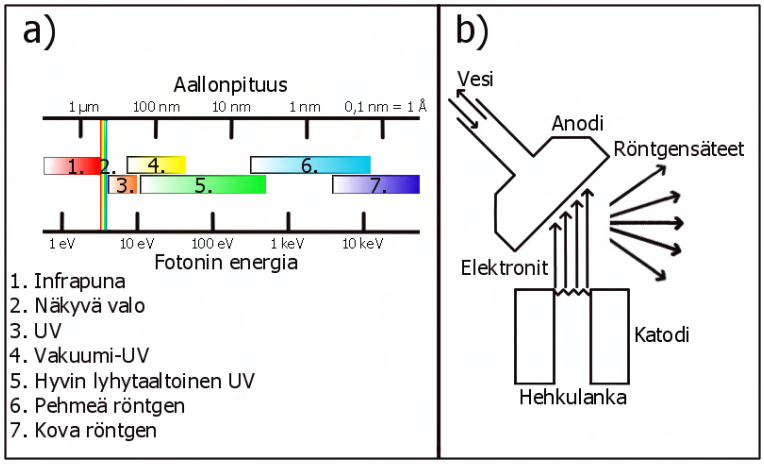
Röntgenspektroskopia on kaikkialla: Sen menetelmiä sovelletaan aina materiaalitieteestä arkkitehtuuriin saakka [1]. Nimensä mukaisesti röntgenspektroskopisissa tutkimuksissa mitataan aineiden röntgenspektrejä, joita analysoimalla saadaan irti kaikenlaista informaatioita näytteiden rakenteista [1]. Analyysin kohteena voi olla vaikkapa molekyylin atomien välinen sidospituus. Sen arvojen havaitaan vaihtelevan näytteessä molekyylikohtaisesti, sillä sidospituudella on jokin tietty rakennejakauma, joka kuvastaa sen mahdollisten arvojen esiintymistodennäköisyyksiä [2]. Kuinka hyvin näytteestä mitatut röntgenspektrit kuvastavat tällaisia rakennejakaumia, on pro gradu -tutkielmani keskeisenä kysymyksenä.

Tutkin simuloitujen vesimolekyylien rakennejakaumien ja kolmen eri röntgenspektroskopiamenetelmän spektrien välisiä yhteyksiä. Nämä kolme menetelmää olivat röntgenabsorptiospektroskopia, röntgenfotoelektronispektroskopia sekä röntgenemissiospektroskopia. Tutkimuksissani koitin löytää jonkin tunnetun rakennejakauman rakennejakaumaparametrit niitä vastaavista keskiarvoröntgenspektreistä. Keskiarvospektrien laskemiseen hyödynsin koneoppimista, sillä näin laskut nopeutuivat tavanomaisempiin keinoihin verrattaessa huomattavasti [3]. Lisäksi halusin selvittää, että mikä röntgenspektroskopiamenetelmistä olisi sensitiivisin rakennejakauman muutoksille. Tätä testasin etsimällä useita tunnettuja rakennejakaumaparametriyhdistelmiä niitä vastaavista keskiarvospektreistä, ja vertailemalla eri menetelmien tuloksia toisiinsa.

1 Röntgenspektroskopia

Röntgenspektroskopialla tutkitaan aineen rakennetta röntgensäteilyn alueella olevien fotonien avulla, ja sillä on useita käyttökohteita esimerkiksi kemian, fysiikan, materiaalitieteen, teollisuuden ja jopa taiteen parissa [1]. Itse röntgensäteily on sähkömagneettisen säteilyn osa-alue, jonka allonpituus ja energia vaihtelevat välillä 10 nm/124 eV - 10 pm/124 keV. Tämä aallonpituusväli voidaan myös jakaa kahteen osioon, pehmeisiin röntgensäteisiin (engl. *soft X-rays*), jotka ulottuvat suunnilleen 10 nanometristä noin 0.1 nanometriin, ja koviin röntgensäteisiin (engl. *hard X-rays*) joiden aallonpituudet ylettyvät 0.1 nanometristä skaalan loppuun saakka [4]. Kuvan 1 a)-osassa on esitettynä sähkömagneettisen säteilyn spektri, josta nähdään näkyvän valon sekä röntgensäteilyn aallonpituudet. Mutta miksi juuri röntgensäteet soveltuvat spektroskopiseen tarkoitukseen? Se johtuu niiden kahdesta tärkeästä ominaisuudesta [1]. Ensinnäkin röntgensäteet ovat alkuaineiden kannalta selektiivisiä, sillä elektronikuoressa tapahtuvien siirtymien energiat riippuvat alkuaineista itsestään [1]. Toisekseen röntgensäteiden kyky tunkeutua materiaalien sisälle tekee niistä hyödyllisiä aineiden rakenteita luodatessa [1]. Röntgenspektroskopia pitää sisällään monia eri menetelmiä, kuten röntgenabsorptiospektroskopian, röntgenfotoelektronispektroskopian ja röntgenemissiospektroskopian, joita käsittelen tarkemmin tässä tutkielmassa [1, 4, 5].

Wilhelm Conrad Röntgen löysi mukaansa nimetyn röntgensäteilyn vuonna 1895 [1]. Hän ilmoitti maailmalle havainnostaan artikkelissaan *Über eine neue Art von Strahlen* [6], ja sai tästä Nobelin fysiikan palkinnon 1901 [1]. Alunperin röntgensäteily oli tieteellinen kuriositeetti, jota käytettiin lääketieteellisiin ja terapeuttisiin tarkoituksiin [1]. 1900-luvun alkupuolella kehitystä alkoi kuitenkin tapahtua. 1913 Lawrence Bragg ja hänen isänsä William H. Bragg esittelivät lain joka tuli myöhemmin heidän mukaansa tunnetuksi nimeltä Braggin laki, josta heille myönnettiin Nobelin fysiikan palkinnon 1915 [1]. Braggin lakia hyödyntäen Louis de Broglie



Kuva 1. a) Sähkömagneettisen säteilyn spektri. Kuva pohjautuu Attwoodin kirjan [4] kuvaan sivulta 2. b) Coolidgen röntgenputken kaaviokuva. Kuva pohjautuu Nielsenin ym. kirjan [7] kuvaan sivulta 31

mittasi yhden ensimmäisistä röntgenabsorptiospektreistä 1913, ja Osvald Lundqvist julkaisi röntgenemissiospektrin ensimmäisten joukossa 1925 [1]. Muita röntgenspektroskopisia laboratoriokokeita 1920- ja 1930-luvuilla teettivät muun muassa Coster, Sommerfeld, Siegbahn ja Kronig [1].

1.1 Röntgensäteilyn tuottaminen

Wilhelm Röntgenin tutkimuksissaan käyttämä röntgenputki ei ollut kovinkaan luotettava säteilyn lähde [7]. Röntgenspektroskopia kokikin varsinaisen harppauksen eteenpäin kun W.D. Coolidge kehitti uudenlaisen röntgenputken vuonna 1912, joka toimitti pääasiallisen röntgensäteilyn lähteen virkaa lähestulkoon muuttumattomana vuosikymmeniä [7]. Coolidgen röntgenputkessa hehkulangalla tuotetut elektronit kiihdytetään katodilta korkeajännitteen avulla vesijäähdytteiselle metallianodille tyhjiöputken sisällä [7]. Elektronien törmätessä anodille syntyy röntgensäteilyä kahdella eri tavalla [7]. Ensimmäinen näistä tavoista on niin kutsuttu jarrutussäteily (joka tunnetaan myös saksankieliseltä termiltään *bremsstrahlung*), jossa elektronit

säteilevät fotoneja anodilla hidastuessaan, mistä johtuu röntgenputkista saatavan säteilyn spektrin jatkuva osa [7]. Tämän jatkuvan osan lisäksi spetristä löytyy selviä, teräviä viivoja [7]. Kyseisiä viivoja syntyy, kun katodilta lähteneen elektronin törmäys irrottaa sisimpien kuorten elektronin atomista anodilla, ja syntynyt sisäkuoren aukkotila purkautuu sille ominaisilla siirtymillä [7]. Näin emittoitunutta säteilyä kutsutaan fluoresentiksi (engl. *fluorescent radiation*), ja sitä hyödynnetään tutkimuksissa usein mikäli tarvitaan monokromaattista sädettä, sillä karakterististen viivojen intensiteetit ovat useita kertoja jarrutussäteilyä korkeampia [7]. Kuvan 1 b)-kohdasta nähdään Coolidgen röntgenputken kaaviokuva. Coolidgen röntgenputkilla sekä vastaavanlaisilla malleilla on kuitenkin omat heikot puolensa. Röntgenputket kykynevät muuttamaan elektronien liike-energiasta vain suunnilleen 1 % säteilyksi, mistä loput 99 % muuttuvat lämmöksi [1]. Vain pientä osaa emittoiduista fotoneista voidaan hyödyntää tarkassa säteessä, sillä ne hajaantuvat anodilta 4π kokoiselle avaruuskulmalle, joista 2π:n verran abosorpoituu takaisin anodiin [7]. Lisäksi spektrin viivaosuus ei ole säädettävissä jatkuvasti tutkimuksen optimaaliselle aallonpituudelle [7]. Röntgenputkien heikkouksia voidaan kiertää synkrotronien avulla.

Synkrotronit ovat rengasmaisia hiukkaskiihdyttimiä, joilla tuotetaan nimeä kantavaa synkrotronisäteilyä [4, 7]. Synkrotronisäteilyä syntyy, kun relativistisilla nopeuksilla liikkuvat elektronit emittoivat röntgensäteilyä kiihdytettyinä ympyräradoille magneettikentässä [4, 7]. Synkrotronisäteilyn löysivät Herb Pollock, Robert Langmuir, Frank Elder ja Anatole Gurewitsch vuonna 1947 työskennellessään synkrotronin parissa yrityksen General Electric tutkimuslaboratoriossa Schenectadyssä, New Yorkissa [8, 9]. Alun perin synkrotronisäteilyä pidettiin riesana, sillä se havaittiin energiahäviönä elektronien varastointirenkaissa (engl. *electron storage ring*), joita varhaiset synkrotronit myös olivat [4, 8]. Vähitellen tiedemaailma kuitenkin alkoi pohtia mahdollisuutta käyttää synkrotronia röntgensäteilyn lähteenä, ja vuonna 1974 Japanissa rakennettiin ensimmäinen varsinainen laite synkrotronisäteilyn tuot-

tamiseksi [1, 8]. Nykyään niin kutsutut kolmannen sukupolven synkrotronit ovat hyvin tärkeässä osassa modernissa tutkimustyössä säteilylähteinä, ja neljännen sukupolven laitoksiakin on olemassa [1].

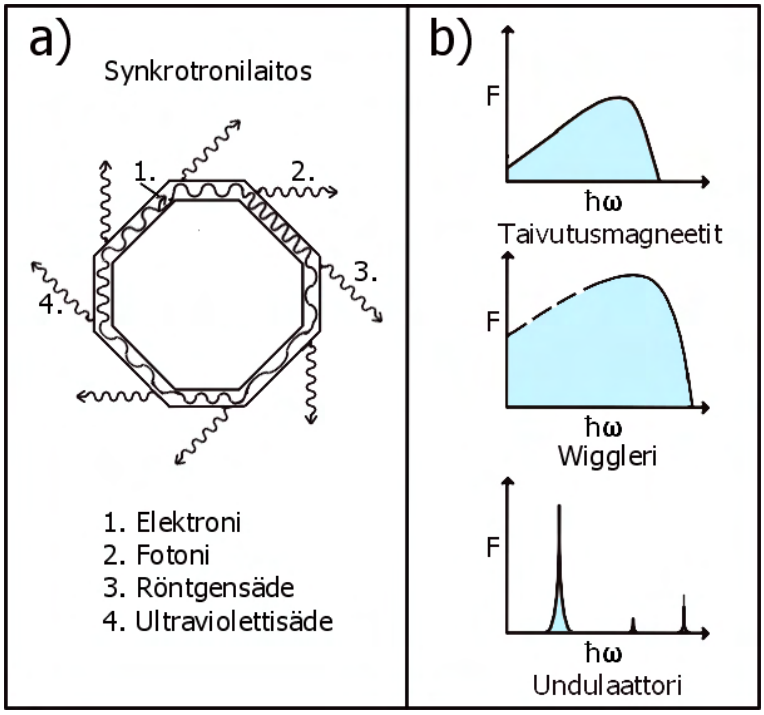
Synkrotronisäteilyä tuotetaan synkrotronilaitoksissa yleensä kolmen eri magneettisen rakennelman avulla: Taivutusmagneettien (engl. *bending magnets*), wigglereiden (engl. *wiggler*) sekä undulaattorien (engl. *undulator*). Taivutusmagneeteilla relativistiset elektronit saatetaan ympyräradalle tasaisella magneettikentällä, jolloin ne säteilevät röntgensäteilyä tangentiaalisesti kapean kartion muotoisesti. Taivutusmagneettien säteilyllä on laaja spektri, ja säteilykartion emissiokulma on yleensä suuruudeltaan noin $1/\gamma$, missä γ on Lorenz-kerroin. Se määritellään seuraavan yhtälön mukaisesti:

$$\gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}}, \tag{1}$$

missä v on inertiaalikoordinaatistojen välinen suhteellinen nopeus, ja c on valonnopeus tyhjiössä. [4]

Wigglereilla sen sijaan säteilyä tuotetaan, kun erittäin relativistiset elektronit liikkuvat jaksollisesti vaihtelevissa vahvoissa magneettikentissä. Näiden magneettikenttien ansiosta elektronit oskilloivat harmonisesti, emittoiden samalla säteilyä. Wigglereiden säteilyn emissiokulma on suurempi kuin $1/\gamma$, eli niiden spektrikin on laaja, vaikkakin sen huippukohta on paljon korkeammalla kuin taivutusmagneettien muuten samankaltaisella spektrillä. Wigglerien tuottama säteily on taivutusmagneettien vastinetta tehokkaampaa, sen vuo on suurempaa sekä röntgensäteiden aallonpituudet ovat pienempiä. [4]

Undulaattorit ovat kovin samankaltaisia kuin wigglerit, niiden magneettikentät vain ovat verrattaen heikkoja vahvojen sijasta. Undulaattoreilla tuotetun säteilyn taajuusjakauma voi olla hyvinkin kapea, sillä oskillaatioamplitudit ovat pieniä heikohkojen magneettikenttien vuoksi, ja näin emittoituneet säteilykartiot ovat erittäin kapeita. Taivutusmagneeteista sekä wigglereistä poiketen undulaattorien säteilykar-



Kuva 2. a) Kaaviokuva synkrotronilaitoksesta. Siinä elektronit kiertävät kehää, ja sen suorilla osuuksilla synkrotronisäteilyä tuotetaan wigglereillä ja undulaattoreilla [4]. Synkrotronisäteilyä syntyy myös taivutusmagneettien ansiosta elektronien liikkuessa suorien osioiden välillä (ei näy tässä kuvassa) [4]. Osa a) pohjautuu Attwoodin kirjan [4] kuvaan sivulta 126. b) Emissionkulmat sekä tyypilliset spektrit taivutusmagneeteille, wigglereille ja undulaattoreille. Osa b) pohjautuu Attwoodin kirjan [4] kuviin sivuilta 124 ja 125

tion emissiokulma on luokkaa $1/\gamma\sqrt{N}$, missä N on magneettisten periodien lukumäärä. [4]

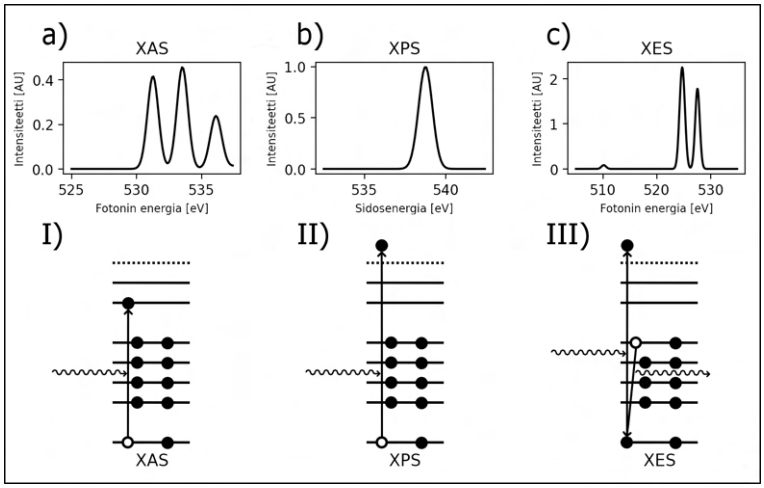
Nykyaikaiset kolmannen sukupolven synkrotronilaitokset koostuvat monista suorista osioista, joilla tuotetaan synkrotronisäteilyä wigglereiden sekä undulaattorien avulla. Wigglereiden ja undulaattorien säteily on osittain koherenttia, ja niiden taajuudet ovat säädettävissä. Näillä laitoksilla syntyy myös säteilyä taivutusmagneettien johdosta, kun elektronit liikkuvat suorasta osiosta toiseen. Kuvan 2 a)-kohdassa nähdään synkrotronin kaaviokuva, ja b)-kohdassa taivutusmagneettien, wigglereiden ja undulaattorien tuottaman säteilyn tyypilliset spektrit. [4]

Vaikkakin wigglereiden ja undulaattorien säteily on koherenttia yksittäistä elekt-

ronia kohti, säteily useista elektroneista kokonaisuudessaan on epäkoherenttia [7]. Tämän ongelman korjaamisen voidaankin käyttää vapaaelektronilasereita (engl. *free-electron laser*) [7]. Vapaaelektronilaserit sisältävät pitkiä undulaattoreita, joiden voimakkaiden säteilykenttien ansiosta niiden lävitse liikkuvat elektronit järjestäytyvät koherentteihin mikrokimppuihin, jotka interferoivat konstruktiivisesti keskenään [7]. Näin syntyvä säteily on erittäin intensiivistä, sekä koherenttia [7, 10]. Vapaalelektronilaserit vaativat kuitenkin tiheän elektronipilven toimiakseen, eikä kolmannen sukupolven laitoksilla kyetä saavuttamaan riittävää elektronitiheyttä [7]. Maailman ensimmäinen kovaa röntgensäteilyä tuottava vapaaelektronilaser nimeltään *The LINAC coherent light source, LCLS* avattiinkin Yhdysvalloissa vuonna 2009 [7, 10]. Sen jälkeen useita vapaaelektronilaserlaitoksia on avattu ympäri maailman, kuten esimerkiksi SACLA Japanissa vuonna 2011 ja Swiss-FEL Sveitsissä vuoden 2017 lopulla [10, 11].

1.2 Röntgenabsorptiospektroskopia

Röntgenabsorptiospektroskopiassa (engl. *X-ray absorption spectroscopy, XAS*) mitataan muutoksia röntgensäteiden absorptoitumisessa materiaaleihin [1, 12]. Täten saadaan tietoa muun muassa näytteiden rakenteesta ja lähiympäristöstä [1, 12]. Röntgenabsorptio soveltuu hyvin tutkimustarkoituksiin, koska sillä sekä kohdealkuaineen järjestysluvulla Z on merkittävä riippuvuus keskenään, vahvuudeltaan likimain Z^4, perustuen kokeellisiin havaintoihin [7]. Näin eri materiaaleja voidaan erottaa hyvin toisistaan luodatessa niitä XAS:n avulla [7]. Lisäksi röntgensäteiden tunkeutumiskyky vaihtelee alkuainekohtaisesti likimääräisesti fotonin energian E kuution käänteisluvun mukaisesti, $1/E^3$ [7]. Tämän seikan ansiosta röntgensäteen energiaa säätämällä kyetään saavuttamaan haluttu tunkeutumissyvyys tutkittavaan materiaaliin [7]. Kuvassa 3 on esitettynä röntgenabsorptiospektroskopian, röntgenfotoelektronispektroskopian ja röntgenemissiospektroskopian tyypilliset spektrit, sekä



Kuva 3. Yllä on kuvattuna osassa a) röntgenabsorptiospektroskopian (XAS), osassa b) röntgenfotoelektronispektroskopian (XPS) ja osassa c) röntgenemissiospektroskopian (XES) tyypilliset spektrit vesimolekyyleille. Piirtämäni spektrit perustuivat tässä tutkielmassani käyttämääni dataan, josta enemmän kappaleessa 3.1. Osassa I) on kuvattuna XAS:lle ominaisen elektronisiirtymän kaaviokuva vesimolekyylissä, osassa II) XPS:lle ja osassa III) XES:lle. Vaakasuorat viivat kuvaavat energiatasoja, mustat pisteet elektroneja, valkoiset pisteet elektroniaukkoja, katkoviivat jatkumon rajapintaa ja koukeroiset viivat absorpoituneita ja emittoituneita fotoneja. Osat I), II) ja III) pohjautuvat Franssonin ym. artikkelin [13] kuvaan yksi

niille ominaisten elektronisiirtymien kaaviokuvat.

Absorptiotapahtumassa itsessään röntgenfotonin energia siirtyy atomin elektronille, joka voi sitten virittyä korkeammalle energiatasolle tai irrota atomista kokonaan ionisaation myötä [7]. Absorptiota johonkin materiaaliin voidaan havainnollistaa Beer–Lambertin lailla

$$I = I_0 e^{-\mu z}, \tag{2}$$

missä I on röntgensäteilyn havaittu intensiteetti, I_0 sen alkuperäinen intensiteetti ennen osumista absorpoivaan aineeseen, μ on materiaalista riippuvainen lineaarinen absorptiokerroin ja z on absorpoivan aineen paksuus [7]. Lineaarinen absorptiokerroin μ on yhteydessä absorption vaikutusalaan σ, jota XAS:ssa mitataan fotonin energian funktiona [7]. μ ja atomikohtainen absorption vaikutusala σ_a riippuvat toi-

sistaan seuraavan yhtälön mukaisesti:

$$\mu = \left(\frac{\rho_m N_A}{M} \right) \sigma_a, \tag{3}$$

missä ρ_m on tiheys, N_A on Avogadron luku ja M on moolimassa [7].

Tietyillä fotonin energioilla XAS-spektrissä absorptio nousee jyrkästi [12]. Tällaista nousua kutsutaan absorptioreunaksi (engl. *absorption edge*), ja sellainen syntyy spektriin, kun fotonien energia riittää siirtämään sisäkuorielektronin (engl. *core electron*) joko virittyneeseen sidottuun tilaan tai jatkumoon (engl. *continuum*) [12]. Absorptioreunaa kutsutaan K-reunaksi, jos elektroni virittyy 1s-orbitaalilta, ja L-reunaksi mikäli elektroni on lähtöisin 2s- tai 2p-orbitaaleilta [12]. Sisäkuorielektronien siirtymistä virittyneisiin sidottuihin tiloihin ollaan kiinnostuneita röntgenabsorptioreunan lähirakenteessa (engl. *X-ray absorption near-edge structure, XANES*), ja sisäkuorielektronien siirtymistä jatkumoon laajennetussa röntgenabsorptiohienorakentessa (engl. *Extended X-ray absorption fine structure, EXAFS*) [12]. Alue, jota XANES koskee ulottuu XAS-spektrissä noin 50 eV:sta absorptioreunan alapuolelta suunnilleen 100 eV:n reunan yläpuolelle, ja XANES:lla saadaan tietoa esimerkiksi absorption kohteen hapetusluvusta, koordinaatioluvusta, symmetriasta ja elektronikuoren rakenteesta [1, 12]. EXAFS:lla tutkittava alue sen sijaan yltää XANES-alueen lopusta aina satojen elektronivolttien päähän absorptioreunasta, ja EXAFS:n avulla voidaan selvittää muun muassa kohdeatomin ja toisten atomien välisiä sidospituuksia sekä ympäröivää epäjärjestystä [1, 12].

Juuri absorptioreunaa edeltävällä alueella XANES-spektrissä röntgenfotonin energia riittää siirtämään sisäkuorielektronin virittyneelle sidotulle tilalle [13]. Virittyneet sidotut tilat ovat avaruudelliselta alaltaan rajoittuneita, minkä ansiosta tutkimalla XANES-spektrin absorptioreunan läheisrakennetta saadaan kohteiden mikroskooppisesta rakenteesta paljon tietoa irti [13]. Esimerkiksi nestemäisen veden XANES-spektrin muoto vaihtelee hieman lämpötilan vaikutuksesta [13]. Lisäksi itse absorptioreunan energia riippuu kohteena olevasta alkuaineesta, sekä sen hapetus-

luvusta, sillä monielektronissa atomissa elektronit hylkivät toisiaan ytimen vetäessä niitä samanaikaisesti puoleensa, ja mitä suurempi hapetusluku atomilla on, sen positiivisempi on sen varaus [12]. Näin vaaditaan enemmän energiaa elektronien viritämiseen, ja täten absorptioreuna löydetään korkeammalta energialta [12]. Kuvan 3 osassa a) nähdään XAS-spektri, ja sen alla osassa I) kaaviokuva sisäkuorielektronin siirtymästä, joka on aiheuttanut absorption rajua nousua spektriin.

EXAFS-modulaatioita syntyy XAS-spektriin fotoelektronin sirotessa naapuroivista atomeista sen liikkuessa poispäin ionisoituneesta atomista. Fotoelektronin energian muuttuessa sen aallonpituuskin muuttuu, ja sen voidaan käsittää etenevän atomista aaltona. Elektroniaallon levitessä ympäristöönsä heijastuu se takaisin naapuriatomeista. Jos energialla E_1 liikkeelle lähtenyt sekä takaisinheijastunut aalto ovat samasssa vaiheessa, interferoivat ne konstruktiivisesti, mikä johtaa kyseisellä energialla E_1 lineaarisen absorptiokertoimen μ:n arvon nousuun. Mikäli taas jollain toisella energialla $E_2 > E_1$ takaisinheijastunut ja atomista lähtöisin oleva aalto ovat eri vaiheissa, interferoivat ne destruktiivisesti, ja μ:n arvo laskee. Otettaessa EXAFS-spektristä Fourier-muunnos saadaan selville etäisyydet absorption kohteena olleen atomin ja naapureiden välillä. [12]

1.3 Röntgenfotoelektronispektroskopia

Röntgenfotoelektronispektroskopia (engl. *X-ray photoelectron spectroscopy, XPS*) on laajalti käytetty röntgenspektroskopian muoto, jota hyödynnetään varsinkin materiaalitieteen, kemian ja kemiantekniikan parissa, ja sen avulla saadaan tietoa muun muassa pintojen kemiasta sekä niiden sidoksien rakenteesta. G. Greczynski ja L. Hultman käyvät läpi XPS:n periaatteita artikkelissaan *X-ray photoelectron spectroscopy: Towards reliable binding energy referencing* [14]. XPS perustuu valosähköiseen ilmiöön, jossa näytteestä emittoituu elektroneja sähkömagneettisen säteilyn

vaikutuksesta. Emittoituneiden elektronien liike-energiat E_{kin} saadaan kaavasta

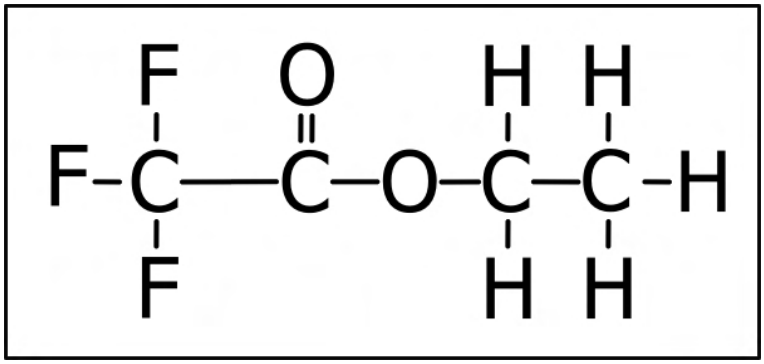
$$E_{kin} = h\nu - E_B, \tag{4}$$

missä h on Planckin vakio, ν on fotonin taajuus ja E_B on elektronin sidosenergia.

XPS-mittauksissa näytteitä säteilytetään röntgenfotoneilla, joiden energia tunnetaan, ja detektorilla havainnoidaan valosähköisen ilmiön ansiosta irronneita elektroneja [14]. Näin XPS-spetrissä esitetään intensiteettiä elektronien sidosenergian funktiona, ja havaitut piikit vastaavat sisäkuorielektronien määrää, jotka eivät törmänneet mihinkään ennen detektorille saapumistaan säilyttäen alkuperäisen energiansa [14]. Matkallaan epäelastisesti törmäilleet elektronit ovat mukana spektrin taustassa sillä puolella piikkiä, jolla sidosenergia on piikin sidosenergiaa korkeampi (spektrin mittaamisen jälkeen tausta vähennetään spektristä itsestään, jos vain voidaan) [14]. XPS:lla on mahdollista mitata myös valenssielektronien sidosenergioita [13].

Kaasufaasissa olevan aineen sisäkuorielektronien sidosenergia E_B saadaan laskettuksi tunnetun fotonin energian $h\nu$ sekä havaittujen elektronien liike-energian E_{kin} avulla yhtälöllä 4. Kiinteille aineille kuitenkaan tämä ei päde, vaan muita seikkoja on myös huomioitava, kuten näytteen sekä spektrometrin työfunktio. Itse asiassa XPS:lla mitatut sisäkuorielektronien sidosenergiat eivät varsinaisesti vastaa mitään tiettyä elektronin energiaa joltakin tietyltä energiatasolta. Elektroneilla ei ole erillisiä energioita perustilalla, vaan ne jakavat systeemin kokonaisenergian keskenään. Täten sisäkuorielektronien sidosenergiat voidaan käsittää ennemminkin erona atomien perustilan energian ja ionisoituneen tilan välillä. Orbitaalikuva on kuitenkin ensimmäinen, laajalti käytetty aproksimaatio jota tässäkin opinnäytetyössä sovelletaan [15]. Kuvan 3 osassa b) nähdään tyypillinen XPS-spektri, ja sen alla osassa II) piikkiä vastaava kaaviokuva sisäkuorielektronin siirtymisestä jatkumoon. [14]

XPS:lla kyetään tunnistamaan mitä alkuaineita näyte sisältää, sillä jokaisella alkuaineella on sille tunnusomaiset sisäkuorielektronien piikit [14]. Homogeeniselle



Kuva 4. Etyylitrifluoriasetaatin rakennekaava. Kuva pohjautuu Greczynskin ja Hultmanin artikkelin [5] kuvaan 12

näytteelle, joka sisältää n alkuainetta, alkuaineen i konsentraatio x_i on

$$x_i = \frac{A_i/s_i}{\sum_{j=1}^{n}(A_j/s_j)}, \tag{5}$$

missä A_i on vastaavan piikin alainen pinta-ala ja s_i on suhteellinen herkkyystekijä (engl. *relative sensitivity factor*, RSF) [14]. RSF on kokeellisesti määritelty suure, joka on ominainen jokaiselle sisäkuorielektronin piikille [14]. Lisäksi XPS:n avulla on mahdollista tutkia alkuaineiden sidostiloja (engl. *bonding state*) niin kutsuttujen kemiallisten siirtymien (engl. *chemical shift*) ansiosta [14]. Kemiallisen siirtymän myötä voidaan havaita, että atomin ympäristöllä on vaikutusta sen sisäkuorielektronien sidosenergioihin [14]. Esimerkiksi etyylitrifluoriasetaatin (jonka rakennekaava on esitettynä kuvassa 4)) hiiliatomien 1s-spektristä nähdään, että niiden sidosenergiat poikkeavat toisistaan selvästi [14]. Tämä johtuu siitä, että tarvitaan enemmän energiaa irrottamaan elektroni hiiliatomista johon fluoriatomit ovat sitoutuneet kuin muista hiiliatomeista [14]. Fluorin kanssa sidoksia muodostavan hiiliatomin valenssin negatiivinen varaustiheys on alentunut fluorin suuren elektronegatiivisuuden ansiosta, joten fotoelektronin lähtiessä hiiliatomista liikkeelle kokee se suuremman Coulombin voiman kuin esimerkiksi kolmen vetyatomin kanssa sidoksia muodostaneen hiilen tapauksessa fotoelektroni kokisi [14]. Näin CF_3:n hiiliatomista lähtöisin oleva elektroni saapuu detektorille alemmalla liike-energialla, minkä ansiosta havaitaan

eroavaisuuksia hiiliatomien 1s-spektrin sidosenergioiden välillä [14]. Kemiallisia siirtymiä nähdään myös vedellä, sillä nestemmäisen veden happiatomin 1s-spektrin piikki havaitaan alemmalla sidosenergialla kuin kaasumaisen veden vastaavan spektrin piikki havaittaisiin [13]. Tässä työssä tutkin vesimolekyylin geometrian vaikutusta XPS-spektriviivan muotoon, ja sen sisältämää rakenneinformaatiota.

1.4 Röntgenemissiospektroskopia

Röntgenemissiospektroskopiassa (engl. *X-ray emission spectroscopy, XES*) mitataan atomeista emittoituneiden röntgenfotonien energioita, kun atomiin syntynyt sisäkuoren elektroniaukko täyttyy ylemmän kuoren elektronilla [1, 16]. XES:ä hyödyntämällä voidaan selvittää emittoivan atomin elektronista rakennetta, ja varsinkin 3d siirtymämetallien ligandiympäristöä [16]. XES:ä kutsutaan usein sekundaariseksi prosessiksi, sillä sisäkuoren elektroniaukko aiheutuu röntgenfotonien absorptiosta atomiin [16]. XES ja XAS ovatkin toisiaan tukevia menetelmiä, sillä XAS:ssa sisäkuorielektroni siirtyy tyhjälle tilalle absorption vaikutuksesta jättäen elektroniaukon taakseen, ja XES:ssa sen sijaan elektroniaukko täyttyy korkeammalta oribitaalilta lähtöisin olevalla elektronilla [16]. XAS-spektri kuvastaa miehittämättömien tilojen tiheyttä, kun taas XES-spektri miehitettyjen tilojen tiheyttä [16].

XES:ssa hyödynnettävä emissioprosessi on seuraavanlainen: Atomiin absorpoituva energia Ω saadaan seuraavasta yhtälöstä:

$$\Omega = h\nu_R, \tag{6}$$

missä h on Planckin vakio ja ν_R on röntgenfotonin taajuus. Nyt atomin energia nousee perustilalta virittyneelle välitilalle, ja tämän jälkeen viritystila purkautuu emission muodossa, ja atomin energia siirtyy lopputilalle. XES:ssa mitataan taajuuden ν_e omaavan emittoituneen fotonin energiaa ω,

$$\omega = h\nu_e. \tag{7}$$

Kun energia Ω tunnetaan, voidaan päätellä atomin energian siirtyneen Ω:n ja ω:n erotuksen verran emissioprosessin ansiosta. Atomissa tapahtuneita tarkkoja siirtymiä ei välttämättä tiedetä, mutta järjestelmän kokonaisenergian muutos on tiedossa. Nyt elektronikonfiguraatio kyetään määrittämään energiatiloista teoreettisten keinojen avulla. Kuvan 3 osassa c) on esitettynä tunnusomainen XES-spektri, ja sen alla osassa III) spektrin aiheuttaneen emission elektronisiirtymän kaaviokuva. [16]

Jos absorption ansiosta atomista poistunut elektroni on lähtöisin K-kuorelta (eli 1s-orbitaalilta), puhutaan silloin XES:n yhteydessä K-fluoresenssista. Mikäli elektroni onkin lähtöisin L-kuorelta (2s- tai 2p-orbitaalilta), on silloin kyseessä L-fluoresenssi, ja niin edelleen. XES-spektrissä esiintyvät piikit luokitellaan niiden intensiteetin perusteella α, β, γ... piikeiksi, missä α on intensiivisin. Näin esimerkiksi intensiivisin K-fluoresenssi Kα syntyy, kun aukko 1s-orbitaalilla täyttyy 2p-orbitaalin elektronilla. Kα piikit löytyvät XES-spektrissä alimpien energioiden kohdilta, ja Kα sekä Kβ piikit eroavat toisistaan satojen eV:n verran. [16]

XES:n pääasiallisena ideana on mitata muutoksia K-emissiospektrissä, kun absorption kohteena olevan atomin kemiallinen ympäristö muuttuu [16]. XES-spektrin piikkejä tulkitsemalla saadaan paljolti tietoa irti näytteestä, esimerkiksi Kβ piikkien analysointi paljastaa informaatiota näytteen kemiallisista sidoksista, ja nestemäisen veden XES-spektrin piikkien muodot vaihtelevat hieman lämpötilan vaikutuksesta [13, 16].

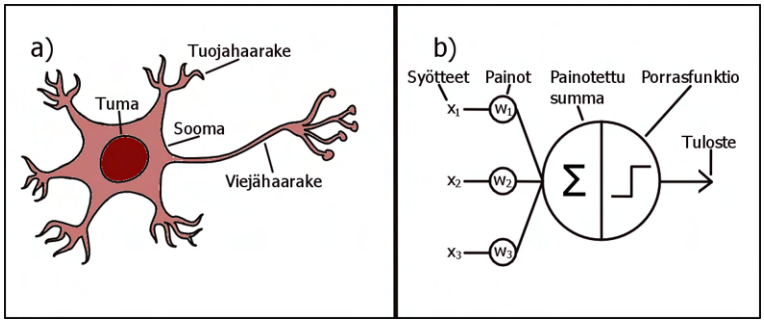
2 Koneoppiminen

A. Géron käsittelee kirjassaan *Hands-On Machine Learning with Scikit-Learn & TensorFlow* [17] koneoppimisen teoriaa ja käytäntöä. Hänen mukaansa koneoppimisessa tietokone ohjelmoidaan omaksumaan jotakin siihen syötetystä datasta, minkä jälkeen se ryhtyy suorittamaan dataan liittyvää tehtävää. Tyypillinen koneoppimisella ratkaistava tehtävä on ennustaminen, ja useimmille ihmisille tuttu esimerkki tästä on sähköpostin roskapostisuodatin.

Muthukrishnan ym. avaavat koneoppimisen historiaa artikkelissaan *Brief History of Artificial Intelligence* [18]. Heidän mukaansa ensimmäisten joukossa tekoälyn (jonka osa-alue koneoppiminen on) alaan liittyvän julkaisun tekivät McCulloch ja Pitts vuonna 1943. Julkaisu koski heidän kehittämäänsä tietokonemallia, joka oppi verrattavissa ihmisaivojen hermosoluihin. He nimesivät mallin MCP-neuroniksi. Se sai syötteenään totuusarvon, jonka se käsitteli esiasetetulla tavalla. Tehokkaamman version MCP-neuronista esitteli Rosenblatt vuonna 1958. Hänen keksintönsä sai nimen perseptroni. Perseptronit olivat joustavampia kuin MCP-neuronit, ja näin niistä kehkeytyi modernien hermoverkkojen keskeisiä rakennusosia. Kuvassa 5 on esitettynä ihmisaivojen hermosolun, sekä siihen pohjautuvan perseptronin kaaviokuvat.

Kaikesta huolimatta perseptroneihin kohdistetut epärealistiset odotukset tulivat pian ilmi. Vuonna 1968 Minsky ja Papert osoittivat, etteivät perseptronit kykyne ratkaisemaan eksklusiivistä disjunktiota. Lisäksi Lighthill tutkiskeli vuonna 1973 alalla esiintyvää ylioptimismia, ja samalla korosti, kuinka vähän todellista kehitystä olikaan tapahtunut. Tekoälyyn annettua rahoitusta leikattiin kautta laidan, mikä johti lopulta niin kutsuttuun ensimmäiseen tekoälytalveen. Näihin aikoihin suurin osa tekoälyn tutkijoista keskittyi tietämyspohjaisiin, sääntöihin perustuviin järjestelmiin, mutta ne eivät kuitenkaan kyenneet suoriutumaan monimutkaisista tehtävistä. [18]

Vuonna 1985 Rumelhart, Hinton ja Williams tarjosivat lisää tuulta hermo-



Kuva 5. a) Ihmisaivojen hermosolu b) Perseptroni, tai LTU (LTU:ta kutsutaan joskus perseptroneiksi [17]). Perseptroni laskee syötteistänsä painotetun summan, jonka se syöttää porrasfunktioon [17]. Jos summa ylittää jonkin kynnysarvon, tulostaa perseptroni positiivisen luokan, ja jos ei, niin negatiivisen luokan [17]. Kuvan a)-kohta pohjautuu Muthukrishnanin ym. artikkelin [18] kuvaan 2, ja b)-kohta Géronin kirjan [17] kuvaan sivulta 259

verkkojen purjeisiin, kun he esittelivät gradienttilaskeutumismetodin (engl. *gradient descent*) hermoverkkojen parametrien optimoimiseksi. Lisäksi Rumelhart ja kollegat julkistivat seuraavana vuonna vastavirta-algoritmin (engl. *backpropagation algorithm*), joka mahdollisti monikerroksisten hermoverkkojen kouluttamisen. Vastavirta-algoritmi mullisti hermoverkkojen oppimiskyvyt, mikä lisäsi kiinnostusta ja rahoitusta koneoppimisen alalle lopettaen samalla ensimmäisen tekoälytalven. Siitä huolimatta 1990-luvun alussa ymmärrettiin, etteivät tämänkaltaiset hermoverkot ole skaalautuvia. Sen aikaisten tietokoneiden laskentateho oli myöskin varsin alhainen hermoverkkojen vaatimuksiin nähden, mikä oli ristiriidassa niiden ympärillä liikkuvan innostuksen kanssa. Alkoi toinen tekoälytalvi. Tämän johdosta alalla alettiin suosimaan yksinkertaisempia algoritmeja kuten tukivektorikoneita, sillä ne pystyivät hyödyntämään 1990-luvun alun tarjoamaa laskentatehoa. Vaikka ne olivat kehitettäneet Vapnik ja Chervonenkis jo vuonna 1963, eivät tukivektorikoneet tulleet suosituiksi kuin vasta vuoden 1992 jälkeen Boserin ym. esittelemän niin kutsutun "kernelitempun"ansiosta. [18]

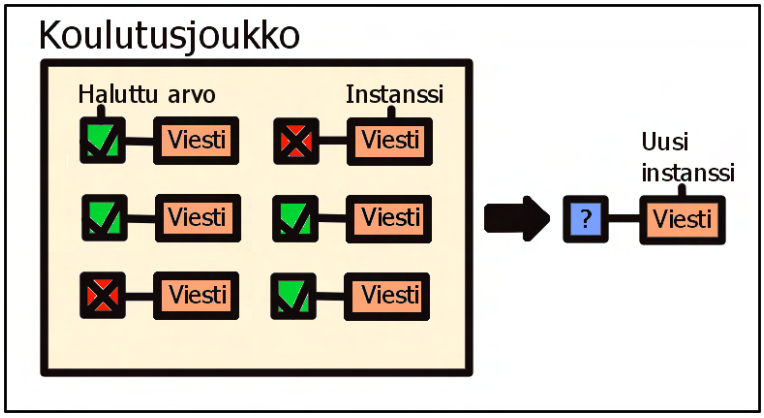
Tietokoneiden laskentateho kasvoi läpi 1990-luvun, mikä tuki hermoverkkojen kehitystä. Tämän ansiosta 90-luvun keskivaiheilla innostus tekoälyyn alkoi taas kas-

vaa. Vuonna 1997 IBM:n supertietokone Deep Blue voitti shakkimestari Garry Kasparovin shakkiottelussa, minkä ansiosta yleisön kiinnostus tekoälyyn nousi jälleen. 2000-luvulle siirryttäessä näytönohjaimet sekä suuret datavarastot olivat viemässä koneoppimista eteenpäin. Koska dataa oli nyt saatavilla valtavasti, koneoppimisalgoritmien suorituskyvyt kehittyivät huomattavasti. Vuonna 2006 Hinton ym. esittelivät syväoppimisen (engl. *deep learning*), jonka avulla he saivat erinomaisia tuloksia puheentunnistuksen parissa, jota aiemmin pidettiin haastavana tehtävänä tekoälylle. Ja vuonna 2012 syväoppimista hyödyntäen Krizhevsky ym. esittelivät AlexNetin, hermoverkon joka käytti hyödykseen ImageNetin 1.2 miljoonan kuvan datajoukkoa, jonka he valjastivat voittaakseen Image Net Large Scale Visual Recognition Challengen. Huomattavasti muita kilpailijoita paremman suorituskyvyn ansiosta aiheutti AlexNet vallankumouksen tekoälyn alalla, ja samalla palautti tähän päivään saakka jatkuvan innostuksen syväoppimiseen. [18]

2.1 Koneoppimisen periaatteet ja luokittelu

Koneoppimisen periaatteet ilmenevät hyvin roskapostisuodattimen toiminnassa. Suodatin kykenee ennustamaan automaattisesti, ovatko viestit roskaposteja ennen kuin lukija ennättää niitä edes näkemään, perustuen käyttäjien merkitsemiin esimerkkiviesteihin. Näitä esimerkkiviestejä kutsutaan koulutusjoukoksi (engl. *training set*), ja jokainen esimerkkiviesti itsessään on koulutusinstanssi (engl. *training instance*). Perustuen vaikkapa viestien väliseen samankaltaisuuteen, suodatin luokittelee saapuvan viestin joko roskapostiksi tai normaaliksi sähköpostiksi. Tätä luokittelutarkkuuta voidaan mitata esimerkiksi oikein roskapostiksi luokiteltujen viestien suhteella kaikkiin luokiteltuihin viesteihin. Kuvasta 6 nähdään roskapostisuodattimen toimintaperiaate visuaalisesti havainnollistettuna. [17]

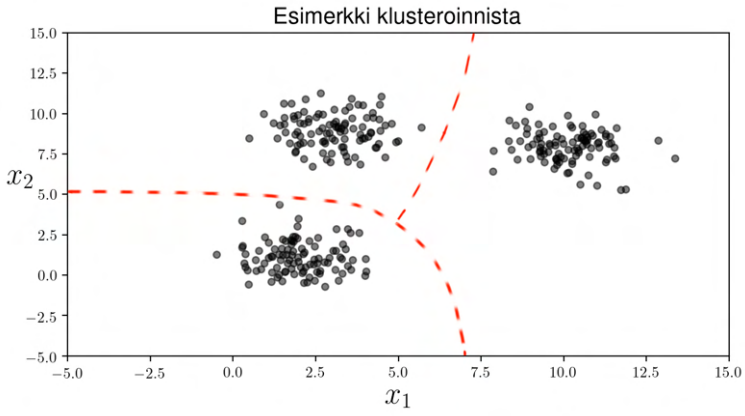
Tämänkaltaisen roskapostisuodattimen laatiminen koneoppimisen avulla on paljolti käytännöllisempää, yksinkertaisempaa ja tehokkaampaa kuin käyttäen perin-



Kuva 6. Koneoppimisella toteutetun roskapostisuodattimen toimintaperiaate. Koulutusinstansseille löytyy koulutusjoukosta niitä vastaavat halutut arvot, roskaposti tai normaali viesti, ja kouluttamisen jälkeen roskapostisuodatin luokittelee ennennäkemättömän viestin jommaksi kummaksi. Kuva pohjautuu Géronin kirjan [17] kuvaan sivulta 8

teisempiä ohjelmointikeinoja. Tavanomaisesta ohjelmoimisesta poiketen koneoppimisen avulla voi tietokone suoriutua vaivatta tehtävistä, jotka tarvitsisivat lukuisia ohjeita sekä hienosäätöä toimiakseen, usein paremmin tuloksin. Perinteisin keinoin roskapostien tunnistamiseen tarvittaisiin monia eri sääntöjä havaitsemaan niissä esiityviä yleisiä ilmaisuja, ja mikäli ne muuttuisivat tarvitsisi suodattimen laatijan kirjoittaa jälleen uusia sääntöjä niiden erottamiseksi. Koneoppimiselle soveltuvat myöskin ongelmat, joille ei löydy tavanomaisilla algoritmeilla hyviä ratkaisuja. Näiden ongelmien sekaan kuuluu esimerkiksi puheentunnistus. Lisäksi kouluttamisen jälkeen voidaan koneoppimisalgoritmeja tutkia ja näin selvittää, mitä ne ovat tarkalleen oikein datasta oppineet. Roskapostisuodattimen tapauksessa voidaan nähdä minkälaisia fraaseja roskapostit paljolti sisältävät. Tähän tapaan kyetään löytämään uutta informaatiota sekä odottamatomia yhteyksiä datan sisältä. Mikäli dataa löytyy valtavat määrät, niin kutsutulla tiedonlouhinnalla (engl. *data mining*) etsitään näitä tuntemattomia ja ennenkuulumattomia riippuvuuksia. [17]

Koneoppimisalgoritmeja voidaan niiden kouluttamisen kannalta jakaa useisiin eri kategorioihin. Pääeroavaisuus näiden kategorioiden välillä liittyy siihen, että sisältä-



Kuva 7. Visuaalinen kuvaus klusteroinnista. Ominaisuuksista x_1 ja x_2 koostuva sattumanvaraisesti generoitu datajoukko on klusteroitu kolmeen klusteriin punaisen katokoviivan osoittamalla tavalla. Kuva pohjautuu osittain Géronin kirjan [17] kuvaan sivulta 11

vätkö koulutusdatan instanssit niitä vastaavat halutut arvot. Roskapostisuodattimen tapauksessa halutut arvot olisivat tunnettu luokittelu roskapostiksi tai tavalliseksi sähköpostiksi; jokainen esimerkkiviesti olisi merkattu jommaksi kummaksi. Neljä kategoriaa näin jaeteltuna ovat ohjaamaton oppiminen (engl. *unsupervised learning*), ohjattu oppiminen (engl. *supervised learning*), puoliohjattu oppiminen (engl. *semi-supervised learning*) ja vahvistusoppiminen (engl. *reinforcement learning*). [17]

Ohjaamattomassa oppimisessa koulutusdata ei sisällä haluttuja arvoja, ja oppimisalgoritmeillä pyritään etsimään datasta korrelaatioita sekä erilaisia ryhmiä. Näitä ryhmiä kutsutaan myös klustereiksi. Esimerkkinä ohjaamattomasta oppimisesta jonkin verkkosivun ylläpitäjä voi kerätä dataa sivunsa käyttäjistä, ja jaetella klusterointialgoritmeilla käyttäjiään alaryhmiin samanlaisten piirteiden kannalta. Visuaalinen kuvaus klusteroinnista löytyy kuvasta 7. [17]

Ohjaamattoman oppimisen tapaan puoliohjatussa oppimisessa suurin osa datasta ei sisällä haluttuja arvoja, mutta joihinkin instansseihin ovat arvot merkattu. Aluksi data klusteroidaan ohjaamattoman oppimisen keinoin, jonka jälkeen klusterin datapisteille annetaan arvot perustuen haluttujen tulosten sisältäviin esimerkkeihin klusterissa. [17]

Sen sijaan näistä kahdesta kategoriasta poiketen vahvistusoppimisessa oppija (jota kutsutaan agentiksi) havainnoi ympäristöänsä, ja tekee siihen liittyviä päätöksiä. Perustuen tehtyihin päätöksiin agentti saa palkintoja (tai rangaistuksia), ja ajan kuluessa määrittää se parhaan strategian palkintojen maksimoimiseksi. Ympäriinsä liikkuvat robotit ovat hyvä esimerkki vahvistusoppimisesta. [17]

Tämän tutkielman kannalta kuitenkin tärkein kategoria on ohjattu oppiminen. Sen hyödyntäminen soveltuu tutkielmaani hyvin, sillä simuloitujen vesimolekyylien spektreille löytyy niitä vastaavia rakenneparametreja, kuten sidospituudet sekä niiden väliset kulmat. Ohjatussa oppimisessa koulutusjoukko sisältää halutut arvot/tulokset sekä niitä vastaavat syötteet, ja nämä halutut arvot voivat olla vaikka kategorisia tai numeerisia. Mikäli arvot ovat kategorisia, kyseessä on luokittelu (engl. *classification*), ja haluttuja tuloksia kutsutaan englanniksi termillä *labels*. Ne kuvastavat datassa esiintyvien luokkien nimiä. Roskapostisuodatin on oiva esimerkki luokittelusta (mitä havainnollistetaan myös kuvassa 6). Siinä luokkia ovat roskaposti ja normaalit viestit, ja algoritmi luokittelee saapuvat viestit jommaksi kummaksi. Luokitteluun käytettyjä algoritmeja ovat esimerkiksi logistinen regressio ja tukivektorikoneet. Numeeristen ennustusten tapauksessa puhutaan regressiosta (engl. *regression*). Regressiossa koulutusdata koostuu yhdestä tai useammasta ominaisuudesta (engl. *feature*) ja niitä vastaavista kohdearvoista (engl. *target*), joita halutaan ennustaa. Kohdearvo voisi vaikkapa olla asunnon hinta, ja ominaisuudet huoneiden määrä, pinta-ala, sijainti ja niin edelleen. Ohjatun oppimisen algoritmeja ovat muun muassa lineaarinen regressio, päätöspuut ja satunnaiset metsät sekä hermoverkot. [17]

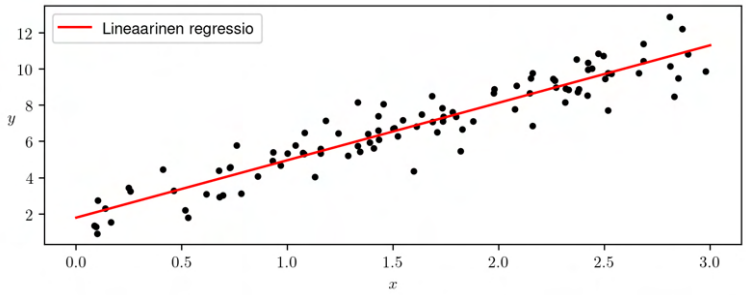
Koneoppimisalgoritmeja voidaan myös luokitella sen perusteella, kuinka ne yleistyvät (engl. *generalize*). Yleistämisellä tarkoitetaan koneoppimisalgoritmien kykyä tehdä hyviä ennustuksia ennennäkemättömistä instansseista. Kaksi tällaista kategoriaa ovat instanssipohjainen oppiminen (engl. *instance based learning*) ja mallipoh-

jainen oppiminen (engl. *model based learning*). Instanssipohjaisessa oppimisessa koulutusjoukko toimii mallina itsenään, ja koulutettaessa algoritmi sisäistää datan itseensä. Tämän jälkeen uusia instansseja yleistetään jonkinlaisen samankaltaisuuden mittapuun mukaisesti. Roskapostisuodattimen esimerkissä saapuvia viestejä voitaisiin luokitella roskapostiksi sen perusteella, kuinka paljon ne sisältävät roskaposteissa usein esiityviä fraaseja suhteessa koulutusjoukon esimerkkeihin. K:n lähimmän naapurin menetelmä kuuluu instanssipohjaisen oppimisen algoritmien helmaan. [17]

Mallipohjaisessa oppimisessa sen sijaan rakennetaan nimensä mukaisesti malli koulutusjoukon esimerkkien pohjalta, jonka jälkeen tätä mallia käytetään ennustusten tuottamiseen. Malli sovitetaan koulutusdataan niin, että sen sisäiset parametrit tuottavat parhaita mahdollisia tuloksia ennustettaessa uusista instansseista. Koulutettaessa algoritmi etsii parametreille arvot jotka sopivat dataan parhaiten, ja tämä tehdään yleensä minimoimalla niin kutsutun sakkofunktion (engl. *cost function*) arvo. Sakkofunktio mittaa, kuinka huonosti malli suoriutui tehtävästään määrittämällä eron mallin ennustusten ja koulutusesimerkkien väliltä. Regressiotapauksissa sakkofunktiona voidaan käyttää vaikkapa keskineliövirheen neliöjuurta (engl. *root mean squared error, RMSE*). Kun sakkofunktion arvo on saatu minimoitua ja täten mallin parametrit ovat selvillä, kyetään uusista instansseista antamaan ennustuksia niiden avulla. Lineaarinen regressio, tukivektorikoneet ja hermoverkot ovat esimerkkejä mallipohjaisen oppimisen koneoppimisalgoritmeista. Kuva 8 on esimerkki lineaarisesta regressiosta. [17]

2.2 Koneoppimismallien koulutus ja valinta

Koneoppimismallien yleistämiskykyä arvioidaan erillisellä testijoukolla. Testijoukosta määritetyn virheen tarkoitus on simuloida ennestään tuntemattomilla instansseilla mitattavaa yleistysvirhettä (engl. *generalization error*). Täten testijoukosta lasketulla virheellä voidaan evaluoida mallien suorituskykyä uusilla tapauksilla. Kou-

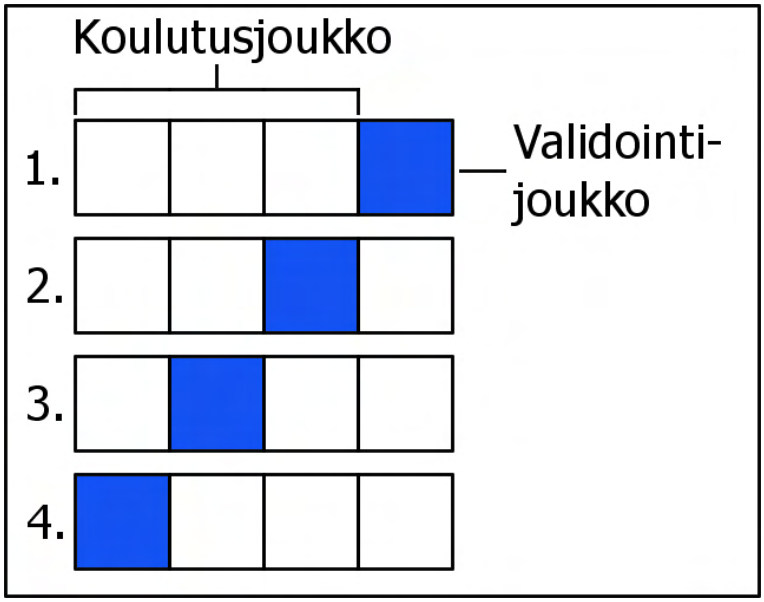


Kuva 8. Esimerkki lineaarisesta regressiosta. x ja y ovat sattumanvaraisesti generoituja, ja kuvaaja pohjautuu Géronin kirjan [17] sivun 111 kuvaan

luttamiseen saatavilla olevasta datasta Géron suosittelee 80% käytettäviseksi koulutusjoukkoon, ja 20% testijoukkoon. [17]

Tärkeä osa koneoppimisalgoritmien kouluttamista ovat niin kutsutut hyperparametrit, jotka ovat oppimisalgoritmien parametreja, eivätkä niinkään mallin itsessään. Hyperparametrien arvot asetetaan etukäteen, ja ne pysyvät muuttumattomina läpi koulutusprosessin. Hyperparametrit vaikuttavat siihen, kuinka malleja koulutetaan. Esimerkiksi neuronien määrää hermoverkossa säätelee hyperparametri, samoin kuin naapurien määrää K:n lähimmän naapurin menetelmässä. Hyperparametrien arvoja säätämällä koneoppimismallien suorituskykyä voidaan parantaa, ja tätä varten koulutusjoukosta eristetään erillinen validointijoukko. Hyperparametrejä säädettäessä malli koulutetaan koulutusjoukolla, jonka jälkeen sen ennustuksista määritetään virheet validointijoukosta hyperparametrien optimaalisten arvojen selvittämiseksi. Tätä kutsutaan validoinniksi. Kun hyperparametrien parhaat arvot ovat selvillä testataan mallin suorituskykyä lopullisesti testijoukolla. [17]

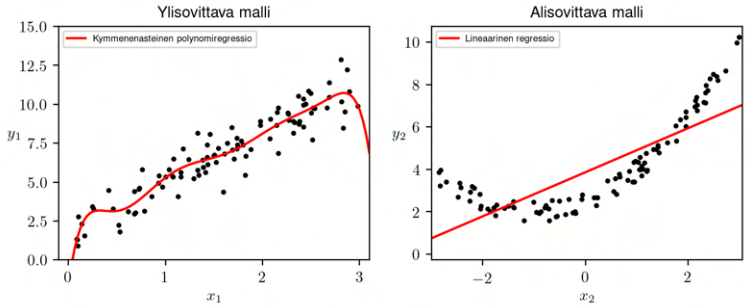
Erillinen validointiprosessi on tarpeellinen, sillä jos yleistysvirhettä mitattaisiin testijoukolla valittaessa hyperparametrejä, olisivat ne sovitettu parhaiten siihen testijoukkoon eivätkä ylestäisi hyvin sen ulkopuolelle. Jotta vältettäisiin koulutusdatan tuhlausta erilliseen validointijoukkoon, voidaan hyödyntää ristiinvalidointia (engl. *cross-validation*). Ristiinvalidoinnissa koulutusjoukko pilkotaan yhtä suuriin lahkoihin, ja malli koulutetaan kaikilla paitsi yhdellä näistä lahkoista. Tämän jälkeen vali-



Kuva 9. Havainnollistus nelilahkoisesta ristiinvalidoinnista. Tarvitaan neljä iteraatiota, että kaikkia lahkoja on käytetty kerran validoimiseen

doidaan se sillä lahkolla jota ei käytetty kouluttamiseen. Kyseinen prosessi toistetaan niin monta kertaa, että jokaista lahkoa on käytetty kerran validoimiseen. Lopulta lahkojen tuloksista lasketaan keskiarvo ristiinvalidoinnin tulokseksi. Ristiinvalidointi suoritetaan jokaista haluttua arvoa kohti, ja kun parhaat arvot ovat saatu selville, koulutetaan malli parhaiden hyperparametrien kera koko koulutusjoukolla. Lopullinen suorituskyvyn arviointi tehdään testijoukolla. Ristiinvalidointia on demonstroitu kuvassa 9. [17]

Koneoppimiseen, kuteen kaikkiin muihinkin asioihin, liittyy omat haasteensa. Jotta algoritmit toimisivat kunnolla, tarvitaan paljon koulutusdataa. Géronin mukaan yksinkertaisetkin ongelmat vaativat tuhansia esimerkkejä, ja monimutkaisiin tehtäviin kuten kuvantunnistukseen tarvitaan jopa miljoonia. Koulutusdatan pitää myös kuvastaa uusia tapauksia joista ennustuksia tullaan tekemään, jotta malli yleistäisi hyvin. Mikäli otos on liian pieni, otantakohinaa voi syntyä sattumanvaraisen vaihtelun vuoksi dataan, ja jos otantametodi on huono, voi suuriinkin koulutusjoukkohin syntyä virhettä otosvirheen takia. Yleensäkin huonolaatuinen data joka



Kuva 10. a) Ylisovittava malli. Lineaarisesti käyttäytyvään dataan on sovitettu kymmenenasteinen polynomi b) Alisovittava malli. Paraabelin mallisesti käyttäytyvään dataan on sovitettu lineaarinen malli. Molempien kuvaajien data generoitiin sattumanvaraisesti, ja kuvaajien piirtämiseen käyttämäni koodi pohjautuu Géronin kirjan [17] sivujen 110-111 ja 123 koodinpätkiin

sisältää virheitä, kohinaa, puuttuvia arvoja sekä poikkeavia havaintoja vaikeuttaa mahdollisten rakenteiden ja yhteyksien löytymistä datasta. [17]

Ylisovittamisessa (engl. *over fitting*) malli tuottaa hyviä ennustuksia koulutusjoukolla, mutta yleistää kouluttamisen jälkeen huonosti. Tällöin malli sovittaa kohinaan datassa eikä täten pysty havaitsemaan datan varsinaisia sisäisiä rakenteita. Mallit omaksuvat koulutusdatan hyvin ja ennustavat sen pisteet mainiosti, mutta koulutusjoukon pisteiden ulkopuolella ennustukset ovat hyvinkin epätarkkoja. Ylisovittamista tapahtuu silloin, kun malli on liian monimutkainen verrattuna kohinaan datassa. Esimerkkinä kun lineaarisesti käyttäytyvään dataan sovitetaan kymmenenasteinen polynomi, on tämänkaltainen malli selvästi ylisovittava. Ylisovittamista voidaan vähentää käyttämällä yksinkertaisempaa mallia, suuremmalla koulutusjoukolla, vähentämällä koulutusjoukon kohinaa ja regularisoimalla mallia. [17]

Regularisaatiossa pakotetaan mallin parametrien arvot pysymään pieninä, vähentäen näin mallin vapausasteita. Sen tarkoitus on yksinkeraistaa mallia, ja miten regularisaatio sen tekee riippuu käytettävissä olevasta mallista itsestään. Esimerkiksi lineaarisissa regressiossa sakkofunktioon lisätään regularisaatiotermi, ja päätöspuiden tapauksessa voidaan vaikkapa rajoittaa puun maksimisyvyyttä. Regularisaation

määrää koulutettaessa kontrolloivat hyperparametrit, ja niiden arvoja muuttamalla kyetään mallin monimutkaisuutta säätelemään. [17]

Päinvastainen ongelma ylisovittamiseen nähden on alisovittaminen (engl. *under fitting*). Mallin alisovittaessa on se liian yksinkertainen ja tehoton oppimaan datan sisäisiä rakenteita. Esimerkkinä alisovittamisesta lineaarinen malli ei tule ehkä kuvastamaan hyvin paraabelin mallisesti käyttäytyvää datajoukkoa. Alisovittamista vältetään tehokkaammalla mallilla jolla on useampia parametrejä, syöttämällä enemmän ja parempia ominaisuuksia algoritmiin ja vähentämällä mahdollista regularisaatiota. Kuvasta 10 nähdään esimerkit yli- ja alisovittamisesta. [17]

2.3 Lineaarinen- ja polynomiregressio

Lineaarinen regressio on mallipohjainen, ohjatun oppimisen joukkoon kuuluva algoritmi, jonka tarkoituksena on tehdä dataan lineaarinen sovitus [17, 19]. Lineaarinen regressio tuottaa ennustuksia laskemalla syötteistään sekä vakiosta nimeltä vakiotermi (engl. *bias term*) painotetun summan seuraavan yhtälön mukaisesti:

$$\hat{y} = h_\theta(\boldsymbol{x}) = \theta^T \cdot \boldsymbol{x}, \tag{8}$$

missä $\hat{y}$ on ennustettu arvo, $\boldsymbol{x}$ on instanssin ominaisuusvektori, θ on mallin parametrivektori ja h_θ on parametreja θ käyttävä hypoteesifunktio [17]. Hypoteesifunktiolla tarkoitetaan käytettävää mallia, ja parametrivektori θ on muotoa

$$\theta = (\theta_0, \theta_1, \theta_2, \ldots, \theta_n)\,, \tag{9}$$

missä θ_0 on vakiotermi, ja $\theta_1, \theta_2, \ldots, \theta_n$ ovat ominaisuuksia vastaavat painot [17]. Kuvassa 8 on havainnollistettu lineaarista regressiota.

Lineaarista regressiomallia koulutettaessa parametrivektorille θ etsitään mahdollisimman hyvin koulutusdataan sopivat arvot [17, 19]. Ne löytyvät minimoimalla jonkin sakkofunktion arvo, ja yleisimmin regressiomalleille käytetty sakkofunktio on RMSE [17]. RMSE:n hyvänä puolena on se, että sillä laskettu virhe ja tutkittava

suure y ovat samassa mittakaavassa, sekä niiden yksiköt ovat samat [19]. Käytännössä malleja koulutettaessa RMSE:n sijasta minimoidaan keskineliövirhettä (engl. *mean squared error, MSE*), sillä se on yksinkertaisempaa kuin RMSE:n minimoiminen [17]. Tulos on myös sama, koska arvo joka minimoi funktion minimoi myös sen neliöjuuren [17]. MSE:n kaava on seuraavanlainen:

$$\mathrm{MSE}(\boldsymbol{X}, h_\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \cdot \boldsymbol{x}^{(i)} - y^{(i)}\right)^2, \tag{10}$$

missä $\boldsymbol{X}$ on koulutusjoukko matriisimuodossa, h_θ on lineaarisen regression hypoteesi, m on koulutusinstanssien määrä, $\boldsymbol{x}^{(i)}$ on i:s koulutusinstanssi ja $y^{(i)}$ on $\boldsymbol{x}^{(i)}$:ä vastaava kohdearvo [17].

Sakkofunktion minimoimiseen voidaan käyttää useita keinoja, kuten normaaliyhtälöä sekä erilaisia gradienttilaskeutumiseen pohjautuvia algoritmeja [17]. Gradienttilaskeutumisessa parametrien arvoja muutellaan iteratiivisesti sakkofunktion minimin löytämiseksi, ja MSE:tä käytettäessä erägradienttilaskeutumisen (engl. *batch gradient descent*) tapauksessa on kaavaltaan seuraavanlainen:

$$\theta^{(\text{seuraava askel})} = \theta - \eta\nabla_\theta \mathrm{MSE}(\theta) = \theta - \eta\left(\frac{2}{m}\boldsymbol{X}^T \cdot (\boldsymbol{X}\cdot\theta - \boldsymbol{y})\right), \tag{11}$$

missä η on oppimistahti joka määrää otettavan askeleen pituuden, ja $\boldsymbol{y}$ on kohdearvovektori [17]. Erägradienttilaskeutumisessa gradientin laskemiseen käytetään koko koulutusjoukkoa $\boldsymbol{X}$ [17].

Lineaarinen malli on myös mahdollista sovittaa epälineaariseen dataa [17, 19]. Tämä johtuu siitä, että lineaarinen regressio onkin oikeastaan lineaarinen parametrien suhteen, joten syötteille voidaan tehdä epälineaarisia muutoksia [19]. Tällaisen muutoksen tulokseksi saadaan sen komponenttien lineaarikombinaatio [19]. Näin on kyse polynomiregressiossa, ja esimerkiksi toisen asteen polynomimuunnoksen jälkeen N instanssia sisältävän ominaisuuden $x_n \in \mathbb{R}, n = 1, \ldots, N$ ominaisuusmatriisi Φ

on muotoa

$$\Phi = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix} \tag{12}$$

[19]. Nyt lineaarinen regressiomalli voidaan sovittaa paraabelin lailla käyttäytyvään dataan [17].

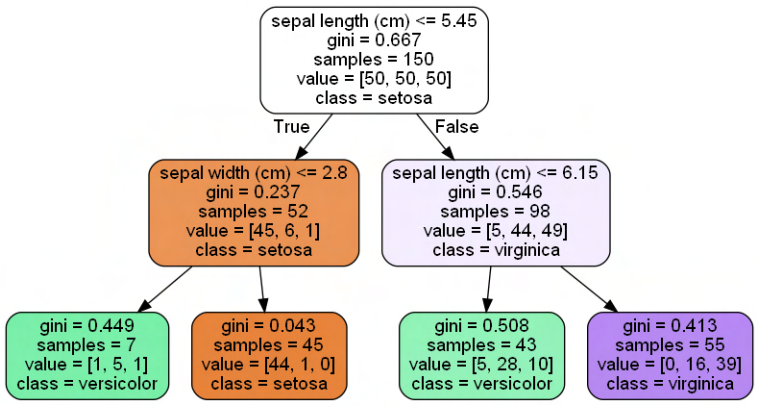
Ylisovittamisen riskin välttämiseksi lineaarisessa ja polynomiregressiossa voidaan hyödyntää regularisaatiota, ja yleensä näiden mallien tapauksessa se toteutetaan rajoittamalla parametrien θ arvoja [17, 19]. Eräs käytetty regularisaation muoto on Ridge regularisaatio, joka tunnetaan myös nimellä Tikhonovin regularisaatio [17]. Siinä sakkofunktioon lisätään regularisaatiotermi

$$\alpha \sum_{i=1}^{n} \theta_i^2, \tag{13}$$

missä α sanelee regularisaation määrän, ja n on ominaisuuksien määrä [17]. Jos α on nolla, on kyse regularisoimattomasta mallista, ja jos taas α:n arvo on suuri, ovat parametrien θ arvot hyvin pieniä, ja täten malli on hyvinkin loiva ja tuottaa siis hillitympiä ennustuksia [17]. Nyt MSE:ä käytettäessä sakkofunktioksi $J(\theta)$ saadaan

$$J(\theta) = \mathrm{MSE}(\theta) \; + \; \alpha \frac{1}{2} \sum_{i=1}^{n} \theta_i^2 \tag{14}$$

[17]. On huomioitavaa, että yllä olevassa yhtälössä vakiotermiä θ_0 ei regularisoida [17]. Regularisaatiotermi lisätään sakkofunktioon vain mallia koulutettaessa, ja kun mallin suorituskykyä evaluoidaan tehdään se ilman kyseistä termiä [17].



Kuva 11. Päätöspuu kurjenmiekan suvun kukkien verholehden pituudelle ja leveydelle. Pohjautuu Géronin kirjan [17] kuvaan sivulta 170

2.4 Päätöspuut ja satunnaiset metsät

Päätöspuut ovat tehokkaita mallipohjaisia ohjatun oppimisen koneoppimisalgoritmeja, joita hyödynnetään sekä luokittelu- että regressio-ongelmien ratkaisemiseen [17]. Niitä käytettäessä koulutusdataa jaetaan rekursiivisesti, ja samalla yksittäisiin jakoihin sovitetaan yksinkertaisia ennustusmalleja [20]. Päätöspuiden toimintaperiaate selviää hyvin tutkimalla graafista esitystä päätöspuusta, ja kuvassa 11 onkin esitettynä Géronin kirjan [17] sivun 170 kuvaan pohjautuva kaksiportainen päätöspuu, jonka tarkoituksena on luokitella kurjenmiekan suvun kukkia perustuen niiden verholehtien pituuksiin ja leveyksiin. Mikäli halutaan luokitella instanssi jonka lehden pituus olisi vaikkapa 5.1 cm ja leveys 2.9 cm, lähdetään puuta hyödynnettäessä liikkeelle sen juuresta (nollatasolta): Juurisolmu kysyy, että onko instanssin verholehden pituus $\leq$ 5.45 cm? Tässä tapauksessa vastaus on kyllä, eli siirrytään alas vasemmanpuoleiseen solmuun. Sitten tämä solmu kysyy, että onko lehden leveys $\leq$ 2.8 cm? Tällä kertaa vastaus on ei, joten päädytään alas oikeanpuoleiseen solmuun. Nyt kyseessä on lehtisolmu, joten puu ennustaa solmun ennustaman luokan instanssille, *Iris setosa*.

Ennustuksen tuottaneessa solmussa 45 koulutusinstanssia täyttää kriteerit verholehden pituus $\leq$ 5.45 cm ja leveys $>$ 2.8 cm, ja 44 instanssia ovat luokkaa *Iris*

setosa, yksi instanssi luokkaa *Iris versicolor* sekä nolla instanssia luokkaa *Iris virginica*. Todennäköisyys, että solmun kriteerit täyttävä instanssi kuuluu luokkaan *Iris setosa* on $44/45 \approx 0.98$, joten se ennustetaankin instanssin luokaksi. [17]

Kyseisen solmun gini-epäpuhtaus G_i on 0.043, ja se lasketaan seraavasti:

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2, \tag{15}$$

missä $p_{i,k}$ on luokan k suhde kaikkiin solmun i koulutusinstanssien luokkiin, ja n luokkien määrä. Gini-epäpuhtaus mittaa solmun epäpuhtautta, ja ennustuksen tuottaneen solmun G_i käyttämällä kaavaa 15 olisi $1-(44/45)^2-(1/45)^2-(0/45)^2 \approx 0{,}043$. Puhtaassa solmussa i kaikki sen kriteerit täyttävät instanssit kuuluvat samaan luokkaan, ja gini-epäpuhtaus G_i on tällöin 0. Regressio-ongelmissa käytetään MSE:ä gini-epäpuhtauden sijasta, ja lehtisolmun ennustus on keskiarvo siihen sopivista koulutusinstansseista. [17]

Päätöspuita koulutetaan CART (*Classification And Regression Tree*) algoritmilla. CART:n ideana on jakaa koulutusjoukko kahteen perustuen yhteen ominaisuuteen k sekä kynnykseen t_k. Kynnys voi olla vaikkapa verholehden pituus ≤ 5.45 cm. Pari (k, t_k) etsitään minimoimalla sakkofunktion arvo. Luokittelussa pyritään minimoimaan gini-epäpuhtaus, ja regressiossa MSE, josta saadaan sakkofunktioksi

$$J(k, t_k) = \frac{m_{\text{vasen}}}{m}\text{MSE}_{\text{vasen}} + \frac{m_{\text{oikea}}}{m}\text{MSE}_{\text{oikea}}, \text{missä} \begin{cases} \text{MSE}_{\text{solmu}} = \sum_{i \in \text{solmu}} \left(\hat{y}_{\text{solmu}} - y^{(i)}\right)^2 \\ \hat{y}_{solmu} = \frac{1}{m_{\text{solmu}}} \sum_{i \in \text{solmu}} y^{(i)} \end{cases}. \tag{16}$$

Yllä olevassa yhtälössä $\text{MSE}_{\text{vasen/oikea}}$:lla mitataan vasemman/oikean koulutusjoukon alajoukon virhettä, ja $m_{\text{vasen/oikea}}$ ilmaisee vasemmalla/oikealla olevien koulutusinstanssien lukumäärän. Ensimmäisen jaon jälkeen alajoukot jaetaan uudestaan, ja tätä jatketaan rekursiivisesti kunnes hyperparametrin määrittämä maksimisyvyys on saavutettu, tai jos algoritmi ei löydä jakoa, joka vähentäsi MSE:n arvoa. Puulle maksimisyvyyden määrittäminen on esimerkki regularisaatiosta, ja muita regularisaatiokeinoja ovat esimerkiksi lehtisolmua koskevien koulutusinstanssien minimi-

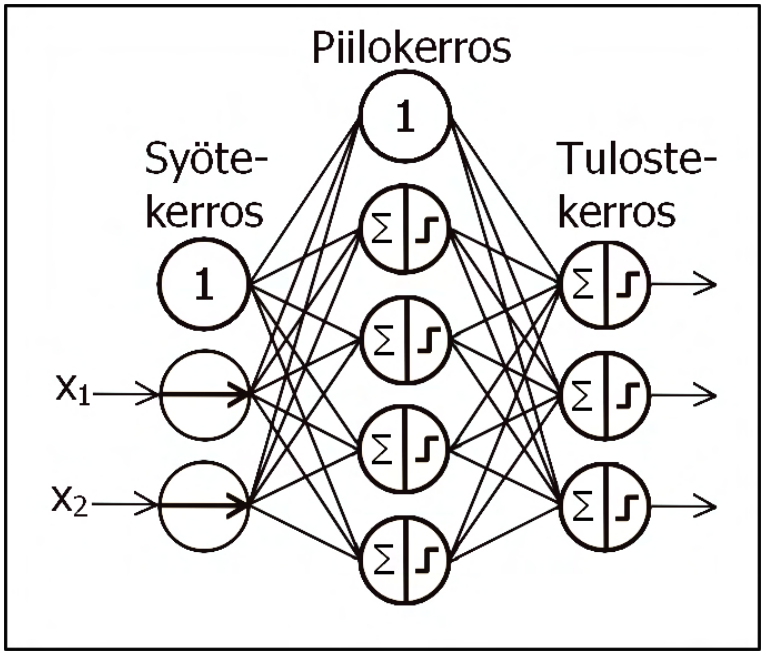
määrän asettaminen, ja lehtisolmujen maksimimäärän säätäminen. [17]

Päätöspuilla on joitakin heikkouksia, ja niistä tärkeimpänä ne ovat herkkiä koulutusjoukon pienille vaihteluille [17]. Tätä ja muita ongelmia voidaankin välttää satunnaisten metsien avulla [17]. Satunnaiset metsät kuuluvat kaikkein tehokkaampien koneoppimisalgoritmien joukkoon, ja ovat esimerkki kokonaisuusoppimisesta (engl. *ensemble learning*) [17]. Kokonaisuusoppimisessa usean ennustajan ennustukset kerätään yhteen, ja tällaisen kokonaisuuden suorituskyky on usein parempi kuin parhaimman yksittäisen ennustajan suorituskyky [17]. Ennustajien kokonaisuudessa tulisi olla täsmällisiä, eli niiden ennustusten pitäisi olla sattumanvaraisia arvauksia parempia, ja monimuotoisia, eli niiden tuottamien virheiden uusilla instansseilla kuuluisi olla erilaisia [21].

Satunnaiset metsät koostuvat useista päätöspuista, jotka on koulutettu sattumanvaraisilla koulutusjoukon alajoukoilla [17, 21]. Satunnaisilla metsillä alajoukkojen otannassa käytetään usein bagging-menetelmää, jonka nimi tulee englannin kielen termistä *bootstrap aggregating* [17, 21]. Bagging-menetelmässä koulutusjoukosta otetaan näytteitä sattumanvaraisesti alajoukkoihin yksittäisiä päätöspuita varten niin, että samoja koulutusinstansseja voidaan valita eri päätöspuille useaan otteeseen, sekä lisäksi niin, että jollekkin tietylle päätöspuulle voidaan instansseja valita moneen kertaan [17, 21]. Bagging-menetelmällä saatavien alajoukkojen koot ovat yleensä yhtä suuria kuin koulutusjoukon koko [17].

Kun yksittäisiä päätöspuita koulutetaan satunnaisessa metsässä alajoukoilla, ei koulutusjoukon jakoa tehdessä etsitä parasta ominaisuutta k kaikkien ominaisuuksien joukosta, vaan sattumanvaraisesta osajoukosta ominaisuuksia [17, 21]. Näin puut poikkeavat enemmän toisistaan, mikä johtaa parempaan malliin [17].

Yksittäisten päätöspuiden kouluttamisen jälkeen malli tuottaa ennustuksia kasaamalla yhteen puiden ennustukset [17, 21]. Luokittelun tapauksessa yleisin ennustettu luokka on mallin ennustus, kun taas regressiossa ennustusten keskiarvo [17].



Kuva 12. MLP:n kaaviokuva. Syötekerroksessa syötetään kaksi ominaisuutta verkkoon vakioneureuronin tulosteen kera, piilokerros koostuu neljästä LTU:sta ja vakioneuronista, ja tulostekerroksessa on tulosteiden veroinen määrä LTU:ta. Kuva pohjautuu osittain Géronin kirjan [17] kuvaan sivulta 263

2.5 Hermoverkot

Keinotekoiset hermoverkot (engl. *artificial neural network, ANN*) ovat monipuolisia ja tehokkaita koneoppimismalleja. Ne soveltuvat hyvin monimutkaisten tehtävien ratkaisemiseen, ja niitä käytetään esimerkiksi miljardien kuvien luokittelemiseen. Hermoverkot kuuluvat mallipohjaisen oppimisen pariin, ja niitä hyödynnetään ohjatussa, ohjaamattomassa, puoliohjatussa sekä vahvistusoppimisessa. Ne koostuvat yksittäisistä keinotekoisista neuroneista, joita kutsutaan lineaarisiksi kynnysarvoyksiköiksi (engl. *linear threshold unit, LTU*). LTU laskee syötteistään $\boldsymbol{x}$ painotetun summan z, joka on muotoa

$$z = \boldsymbol{w}^T \cdot \boldsymbol{x}, \tag{17}$$

missä $\boldsymbol{w}$ on painovektori. Laskettuaan z:n LTU syöttää sen porrasfunktioon step(z), ja tulostaa tuloksen

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \text{step}(\boldsymbol{w}^T \cdot \boldsymbol{x}). \tag{18}$$

Rosenblattin kehittämissä perseptroneissa yleisimmin LTU:lla käytetty porrasfunktio on Heavisiden funktio, joka on muotoa

$$\text{heaviside}(z) = \begin{cases} 0 & \text{if } z < 0 \\ & \\ 1 & \text{if } z \geq 0 \end{cases}. \tag{19}$$

LTU:n kaaviokuva löytyy kuvan 5 b)-kohdasta. [17]

Eräs LTU:ta hyödyntävä hermoverkkoarkkitehtuuri on monikerroksinen perseptroni (engl. *multilayer perceptron, MLP*), joka kykenee approksimoimaan mitä tahansa sileää ja mitattavaa funktiota syötteidensä sekä tulosteidensa välillä [17, 22]. MLP:t koostuvat syötekerroksesta, yhdestä tai useammasta LTU-kerroksesta nimeltään piilokerros sekä tulostekerroksesta [17, 22]. Syötekerroksen neuronit vain syöttävät syötevektorin $\boldsymbol{x}$ hermoverkkoon, sekä yleensä ylimääräisen vakiotermin $x_0 = 1$ [17, 22]. Vakiotermiä varten käytetään vakioneuronia, joka tulostaa aina arvon 1 [17]. Piilokerrokset sisältävät LTU:ta sekä vakioneuronin, ja tulostekerros muodostuu haluttujen tulosteiden veroisesta määrästä LTU:ta [17]. MLP:ssä jokaisen kerroksen neuronit tulostekerrosta lukuunottamatta ovat täysin kytkettyjä seuraavaan kerrokseen, ja jokaisessa kytköksen yhteydessä on oma painonsa [17, 22].

Jotta MLP toimisi kunnolla, on sen neuroneissa porrasfunktio korvattu jollakin muulla aktivaatiofunktiolla (engl. *activation function*), kuten logistisella funktiolla tai ReLU- (engl. *rectified linear unit*) funktiolla [17]. MLP:n kaltaisissa hermoverkoissa ReLU(z)-funktion käyttämistä pidetään oletuksena, ja se määritellään seuraavasti:

$$\text{ReLU}(z) = \max(0, z) \tag{20}$$

[17, 23]. Kuvassa 12 on esitettynä MLP:n kaaviokuva.

MLP:n kouluttamiseen käytetään vastavirta-algoritmia [17, 22]. Vastavirta-algoritmin tarkoituksena on löytää mallin painoille sellaiset arvot, että hermoverkko tuottaisi mahdollisimman pienen virheen [17, 22]. Algoritmi toimii seuraavasti: Jokaista koulutusinstanssia kohti tehdään ennustus, ja ennustuksen virhe määritetään [17, 22].

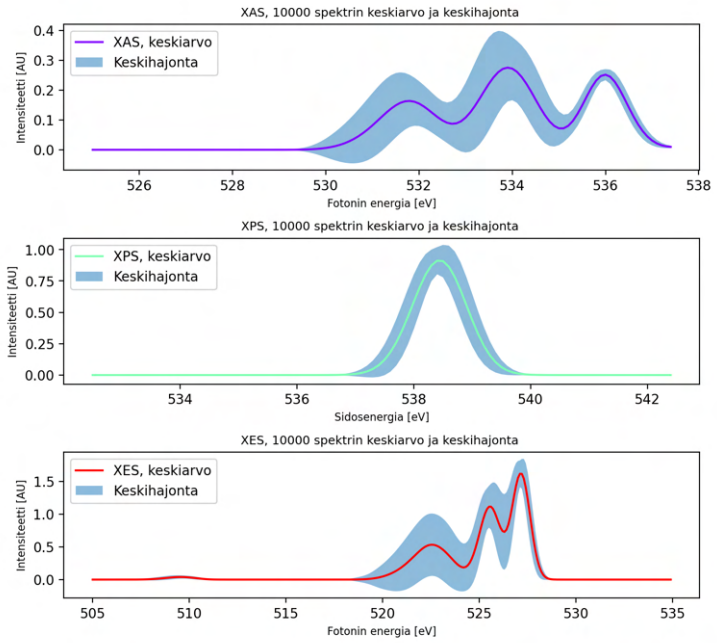
Tämän jälkeen verkko käydään läpi kerros kerrokselta takaperin, ja jokaisen kytköksen aiheuttama kontribuutio virheeseen mitataan[17, 22]. Lopuksi painojen arvoja parannellaan gradienttilaskeutumisen avulla [17, 22]. Näitä vaiheita toistetaan, kunnes kokonaisvirhe on riittävän alhainen [22].

3 Rakennejakauman sovittaminen koneoppimista soveltaen

Pro gradu -tutkielmassani tarkoitukseni oli selvittää, että voidaanko jotkin tietyt vesimolekyylien rakennejakaumaparametrit löytää niitä vastaavista röntgenspektreistä. Eli toisin sanoen, kuinka tarkkaa informaatiota molekyylien rakenteista niiden spektrit oikein sisältävät? Tutkielmassani rakenneparametreilla tarkoitetaan vesimolekyylien sidosten välistä kulmaa sekä niiden sidospituuksia, ja rakennejakaumilla kulman ja sidospituuksien todennäköisyysjakaumia [2]. Käyttämässäni datassa tiettyjen rakenneparametrien omaavalle vesimolekyylille oli määritetty kolme eri röntgenspektriä, XAS-, XPS- ja XES-spektri. Näin selvitin myös, että mikä näistä menetelmistä olisi sensitiivisin rakennejakauman muutoksiin. Molekyylien rakenneparametrit sekä spektrit olivat simuloituja, ja Johannes Niskanen oli laskenut ne käyttäen tiheysfunktionaaliteoriaa. Tässä tutkielmassa tein kaikki laskelmat Python 3 ohjelmointikielen avulla, ja hyödynsin etenkin `scikit-learn`-pakettia [24].

Tunnettujen rakennejakaumaparametrien etsiminen niitä vastaavista spektreistä eteni seuraavasti: Aluksi annoin arvot tunnetuille rakennejakaumaparametreille, ja tallensin ne vektoriin $\boldsymbol{p}_t$. Tämän jälkeen laskin niitä vastaavan keskiarvospektrin integroimalla sen yli asetetun rakenneparametriavaruuden, painottaen sen samalla rakennejakaumalla. Rakennejakauman vuorostaan laskin tunnetusta rakennejakaumaparametrivektorista $\boldsymbol{p}_t$.

Seuraavaksi määritin tunnetuista parametreistä eroavan yriterakennejakaumaparametrivektorin $\boldsymbol{p}_0$. Se sekä tunnettu keskiarvospektri voitiin syöttää kirjoittamaani sakkofunktioon. Sakkofunktio laski $\boldsymbol{p}_0$:lle keskiarvospektrin $\boldsymbol{p}_t$:n tapaan, jonka jälkeen sen sekä tunnetun keskiarvospektrin välinen neliövirhe laskettiin. Nyt syötin tunnetun keskiarvospektrin, yritteen sekä sakkofunktion optimointialgoritmiin, jonka tarkoituksena oli minimoida sakkofunktion arvo muuttelemalla $\boldsymbol{p}_0$:n arvoja. Lo-



Kuva 13. XAS-, XPS- ja XES-spektrien keskiarvot sekä keskihajonnat. Keskiarvot ja keskihajonnat laskettiin jokaista spektrin pistettä kohden

pulta optimoinnin tulosta verrattiin $\boldsymbol{p}_t$:n; kuinka hyvin optimointialgoritmi onnistui löytämään tunnettua spektriä vastaavat rakennejakaumaparametrit?

Keskiarvospektrien laskemiseen hyödynsin koneoppimista. Spektrit voitaisiin laskea tiheysfunktionaaliteorian avulla, mutta optimointivaihe olisi näin erittäin raskas [3]. Ennustamalla spektrit koneoppimismalleilla nopeutuu prosessi huomattavasti. Testailin neljän eri koneoppimismallin suorituskykyä spektrien ennustamisessa, polynomiregression, päätöspuiden, satunnaisten metsien sekä hermoverkkojen, ja parhaiten suoriutuvia näistä käytin XAS-, XPS- ja XES-spektrien ennustamiseen optimointivaiheessa.

3.1 Data

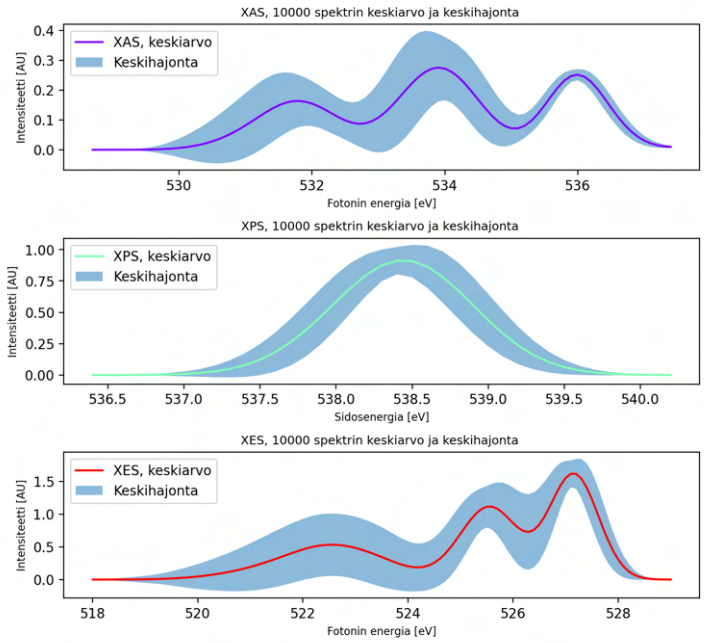
Käyttämäni data koostui 10000 simuloidusta vesimolekyylistä. Jokaista vesimolekyyliä itsessään esitti joukko rakenneparametreja: Molekyylin sidoksien välinen kul-

Taulukko I. Rakenneparametrien keskiarvot ja keskihajonnat 10000 simuloidulle vesimolekyylille

Rakenneparametri	**Keskiarvo**	**Keskihajonta**
Kulma [°]	103.2 °	12.6 °
Pidempi sidos [Å]	1.05 Å	0.07 Å
Lyhyempi sidos [Å]	0.97 Å	0.06 Å

ma yksikössä aste, pidempi sidospituus yksikössä Ångström, lyhyempi sidospituus yksikössä Ångström, molekyylin atomien järjestysluvut yksiulotteisessa taulukossa sekä atomien x-, y- ja z-koordinaatit kolmiulotteisen taulukon muodossa. Koneoppimismallien kouluttamisessa ja optimoinnissa käytin rakenneparametreista vain sidoksien välistä kulmaa sekä pidempää ja lyhyempää sidospituutta, muita rekenneparametreja en tässä tutkielmassa hyödyntänyt.

Jokaiselle simuloidulle vesimolekyylille oli myös määritetty sen XAS-, XPS- ja XES-spektrit. Spektrit olivat yksiulotteisia taulukoita, jotka sisältivät niiden y-koordinaatit mielivaltaisissa yksiköissä. XAS-spektrissä oli 125 pistettä, XPS-spektrissä 100 ja XES-spektrissä 300. Lisäksi kaikkien kolmen spektrityypin x-koordinaatit olivat tallennettu energia-asteikoihin. Nekin olivat yksiulotteisia taulukoita, joissa pisteitä oli yhtä paljon kuin spektreissäkin. 10000 spektrin y-koordinaattitaulukkoja varten oli vain yksi x-koordinaatit sisältävä energia-asteikko, sillä näin spektrejä oli helpompi verrata toisiinsa sekä niiden kuvaajia oli helpompi piirtää. Kuvassa 13 on esitettynä eri spektrityyppien keskiarvospektrit ja niiden keskihajonnat pistekohtaisesti, sekä taulukossa I käytettyjen rakenneparametrien keskiarvot sekä keskihajonnat.



Kuva 14. Leikattujen XAS-, XPS- ja XES-spektrien keskiarvot sekä keskihajonnat

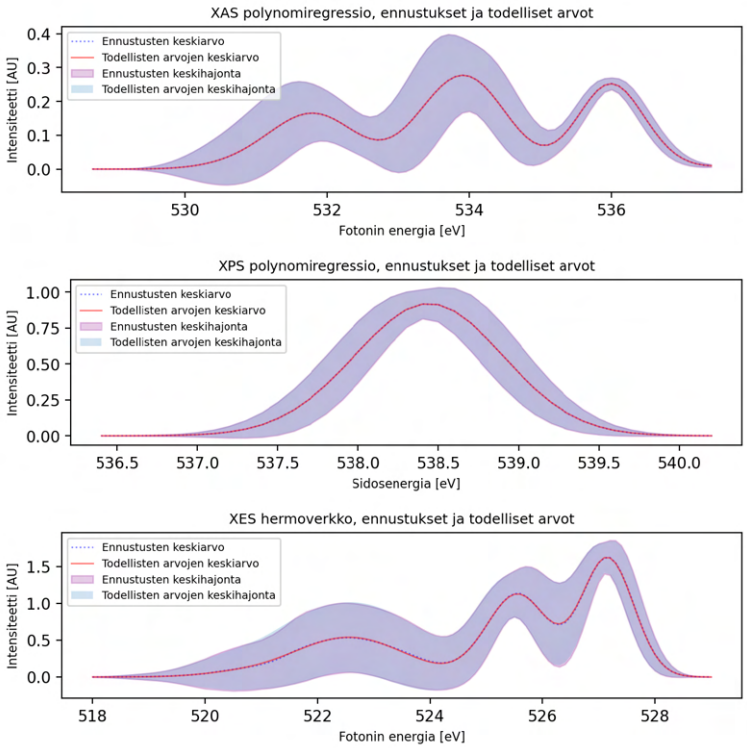
3.2 Metodit

Aluksi leikkasin pois spektrien alku- ja loppupäästä aluetta, jolla spektri pysytteli tasaisesti nollan tienoilla. Näin spektrit sisältivät koneoppimismallien kannalta vähemmän ei-relevanttia informaatiota. Nyt XAS-spektrin (ja sitä vastaavan energia-asteikon) pituus oli 88 pistettä, XPS-spektrin 39 ja XES-spektrin 111. Leikatut spektrit nähdään kuvasta 14. Sitten jaoin datan kolmeen eri koulutus- ja testijoukkopariin niin, että eri spektrityypeille oli omat koulutus- ja testijoukkonsa. Ominaisuuksina toimivat rakenneparametrit kulma, pidempi sidospituus sekä lyhyempi sidospituus, ja kohdearvoina kutakin spektroskopiamenetelmää vastaavat spektrit. Eri koulutus- ja testijoukoissa ominaisuusmatriisit olivat samat, mutta kohdearvovektorit vaihtelivat spektrityypistä riippuen. Géronin suosituksen mukaisesti koulutusjoukkoihin käytin 80 % datasta, ja näin 20 % jäi testijoukoiksi [17]. Täten koulutusjoukoissa oli 8000 instanssia, kolme ominaisuutta, ja kohdearvoina spektrintyypin y-koordinaattipisteitä vastaava määrä arvoja.

Seuraavaksi siirryin koneoppimismallien valitsemiseen ja kouluttamiseen. Testasin neljän eri koneoppimismallin soveltuvuutta spektrien ennustamiseen, polynomiregression, päätöspuiden, satunnaisten metsien ja hermoverkkojen. Parhaimpien hyperparametriyhdistelmien löytämiseksi malleille käytin `scikit-learn`-paketin funktiota `GridSearchCV()` [24]. Se etsii mahdollisten hyperparametrien joukosta parhaimman kombinaation käyttäen ristiinvalidointia, ja palauttaa näiden hyperparametrien kera koulutusjoukolla koulutetun mallin. Käytin kymmenenlahkoista ristiinvalidointia, ja kokeilemani arvot mallien hyperparametreille löytyvät taulukosta III. Polynomiregression ja hermoverkkojen hyperparametrien harkitut arvot pohjautuivat Niskasen ym. artikkelikäsikirjoitukseen (lähetetty vertaisarvioitavaksi) *Neural networks in interpretation of electronic core-level spectra* [25], ja päätöspuun sekä satunnaisen metsän Mantovanin ym. artikkeliin *An empirical study on hyperparameter tuning of decision trees* [26].

Kuitenkin ennen mallien kouluttamista ja validointia standardisoin koulutusjoukon ominaisuudet ja kohdearvot käyttäen `scikit-learn`-paketin `StandardScaler()`-funktiota, sillä koneoppimismallien suorituskyky yleensä heikkenee mikäli ominaisuuksien arvoilla on toisistaan hyvinkin poikkeavat skaalat [17, 24]. Ennen standardisointia sovitin eri `StandardScaler()`:t koulutusjoukkojen ominaisuuksiin ja kohdearvoihin.

Kun koulutusjoukolla koulutetut ja validoidut mallit olivat selvillä, testasin niiden suorituskykyä testijoukolla. Aluksi standardisoin testijoukkojen omainaisuudet ja kohdearvot käyttäen samoja `StandardScaler()`:ta kuin koulutusjoukkojen tapauksessa. Sitten määritin virheen jokaisen koneoppimismallin parhaimman mallin sekä testijoukon välillä RMSE:tä käyttäen. Alhaisimman RMSE:n omaavat mallit valitsin parhamimmiksi malleiksi ennustamaan tietyn spektroskopiamenetelmän spektrejä. Piirsin vielä parhaimpien mallien suoritukset kuvaajiin, joissa testijoukkojen keskiarvospektrit sekä keskihajonnat olivat esitettyinä testijoukkojen ominaisuuksis-



Kuva 15. Parhaimpien mallien ennustukset XAS-, XPS- ja XES-spektreille sekä niitä vastaavat todelliset spektrit

ta ennustettujen keskiarvospektrien ja niiden keskihajontojen kera (kuva 15). Jotta mallin ennustuksista pystyttiin piirtämään kuvaaja, piti ennustettu spektri käänteismuuntaa koulutusjoukon kohdearvoihin sovitetulla `StandardScaler()`:lla.

Kuten kuvasta 15 nähdään, parhaimpien mallien ennustukset sekä testijoukon spektrit vastasivat toisiaan erittäin hyvin: Jokaisen mallin ennustusten ja todellisten arvojen keskiarvot sekä keskihajonnat osuvat kuvaajissa aivan toistensa päälle. Parhaimmat mallit XAS-, XPS- ja XES-spektrien ennustamiseen sekä niiden RMSE:t testijoukoilla on esiteltynä taulukossa IV.

Halusin myös tutkia mallien mahdollista yli- tai alisovittamista. Tätä varten piirsin malleille niin kutsutut oppimiskäyrät. Oppimiskäyräkuvaajassa mallia koulutetaan useaan otteeseen koulutusjoukon alajoukoilla alkaen yhdestä koulutusinstanssista aina koko koulutusjoukon kokoon saakka. Jokaisen koulutuskerran jälkeen mitataan mallin tuottama virhe senhetkisellä koulutusjoukolla, sekä erillisellä validointijoukolla. Oppimiskäyrässä koulutusvirhe ja validontivirhe esitetään kasvavan koulutusjoukon funktiona. Mikäli molempien virheiden käyrät ovat kohtalaisen korkealla, lähellä toisiaan ja tasaiset suuremman kokoisilla koulutusjoukoilla, on malli alisovittava. Jos taas koulutusvirheen käyrä on hyvin alhainen, ja sen sekä validointivirheen käyrien välillä on selkeä väli, malli ylisovittaa. Kuitenkin ylisovittavan mallin tapauksessa käyttämällä paljon suurempaa koulutusjoukkoa koulutus- ja validointivirheen käyrät lähestyvät toisiaan, ja näin ylisovittamisen vaaraa voidaan vähentää. [17]

Piirsin parhaimmille malleille oppimiskäyrät niiden koulutusjoukkoja käyttäen, mistä 10 % käytin validointijoukkoihin. Kuvassa 16 on esitettynä parhaimpien mallien oppimiskäyrät, ja XAS:n ja XPS:n tapauksessa validointivirhe on pienimmän kokoisilla koulutusjoukoilla hyvin korkea. Koulutusjoukon koon kasvaessa molemmilla menetelmillä kuitenkin validointivirhe pienenee ja tasaantuu koko ajan alhaisena pysyneen koulutusvirheen tasolle. Näin voidaan päätellä, että pienemmällä koulu-

tusjoukolla XAS:n ja XPS:n mallit mahdollisesti ylisovittaisivat, mutta käyttämäni suuren koulutusjoukon ansiosta onnistuin sitä välttämään. XES:llä sen sijaan kummatkin virheet poukkoilevat melkoisesti, vaikkakin validointivirhe on alussa korkeampi kuin koulutusvirhe. Molempien virheiden arvot myöskin laskevat koulutusjoukon koon kasvaessa, ja validointivirheen arvo korkeimmillaan on huomattavasti alhaisempi kuin XAS:lla ja XPS:llä. Sikäli kun koulutusvirhe sekä validointivirhe suurimman kokoisilla koulutusjoukoilla ovat varsin alhaisia, sanoisin ettei XES:n paras mallikaan ylisovita. Lisäksi mainittakoon, että hermoverkko jossa on neljä piilokerrosta jotka kukin sisältävät 200 neuronia tuskin alisovittaa.

Kirjasin vielä ylös koneoppimismallien koulutuksessa ja validoinnissa kuluneet ajat. Ne ovat esillä taulukossa II. Hermoverkkoihin kului odotetusti eniten aikaa, sillä niillä on huomattavasti enemmän optimoitavia parametreja kuin muilla malleilla.

Kun parhaimmat mallit XAS-, XPS- ja XES-spektrien ennustamiseen olivat selvillä, siirryin käyttämään niitä simuloitujen vesimolekyylien tunnetun rakennejakaumaparametrivektorin $\boldsymbol{p}_t$ etsimiseen. Rakennejakaumaparametreista laskin kulman ja sidospituuksien rakennejakaumat, ja niinä käytin kulman keskiarvoa μ_k, sidospituuden keskiarvoa μ_s, kulman keskihajontaa σ_k, sidospituuden keskihajontaa σ_s sekä skaalauskerrointa s. Pidemmän ja lyhyemmän sidospituuden painotin samalla jakaumalla, että optimoitavia parametreja olisi vähemmän. Keskiarvospektrejä integroitaessa rakenneparametriavaruuden arvot perustuivat kulman ja sidospituuksien minimi- ja maksimiarvoihin koulutusjoukoissa.

Integrointiprosessi eteni seuraavasti: Aluksi määritin rakenneparametreille kulma ja sidospituus tasaiset jakovälit. Kulma sai arvoja väliltä $60\,^\circ - 150\,^\circ$ jakovälillä $10\,^\circ$, ja sidospituus arvot $0{,}6 - 1{,}5$ Å jakovälillä 0,1 Å. Näin kulmalla ja sidospituuksilla oli kymmenen arvoa kullakin. Sitten muodostin niistä matriisin $\boldsymbol{R} \in \mathbb{R}^{m \times n}$, johon rakenneparametrien mahdolliset yhdistelmät olivat koottuna. Matriisissa oli

tuhat riviä ($m = 1000$) ja kolme saraketta ($n = 3$), esimmäinen sarake sisälsi kulman arvoja, ja toinen ja kolmas sisälsivät sidospituuden arvoja. Jokaisella rivillä toisen sarakkeen sidospituuden arvo oli suurempi kuin kolmannen sarakkeen, sillä näin rivin arvot vastasivat koneoppimismallien syötteitä kulma, pidempi sidospituus ja lyhyempi sidospituus.

Matriisin rivit $\boldsymbol{R}$ toimivat rakenneparametrivektoreina $\boldsymbol{r}_i,\ i = 1, \ldots, m$, jotka syötin tietyn spektroskopiamenetelmän spektrejä ennustavaan koneoppimismalliin. Ennen tätä skaalasin $\boldsymbol{r}_i$:n arvot `StandardScaler()`:lla, jonka olin sovittanut spektrityyppiä vastaavaan koulutusjoukon ominaisuuksiin. Ennustamisen jälkeen käänteismuunsin spektrit koulutusjoukon kohdearvoihin sovitetulla `StandardScaler()`:lla. Sitten kerroin pisteestä $\boldsymbol{r}_i$ lasketun spektrin tilavuusalkiolla, joka oli arvoltaan rakenneparametrien jakovälien tulo, $10 \cdot 0{,}1 \cdot 0{,}1 = 0{,}1$, sekä rakennejakauman arvolla pisteessä $\boldsymbol{r}_i$.

Rakennejakaumafunktio sai syötteenään $\boldsymbol{r}_i$:n, sekä rakennejakaumaparametrivektorin $\boldsymbol{p}$, ja laskin sen arvon kolmen normaalijakauman tiheysfunktion tulona pisteessä $\boldsymbol{r}_i$. Rakennejakaumafunktioon syötettyä $\boldsymbol{r}_i$:tä en skaalannut, ja kulmalle sekä sidospituuksille oli omat tiheysfunktionsa. Normaalijakauman tiheysfunktio on määritelty seuraavasti:

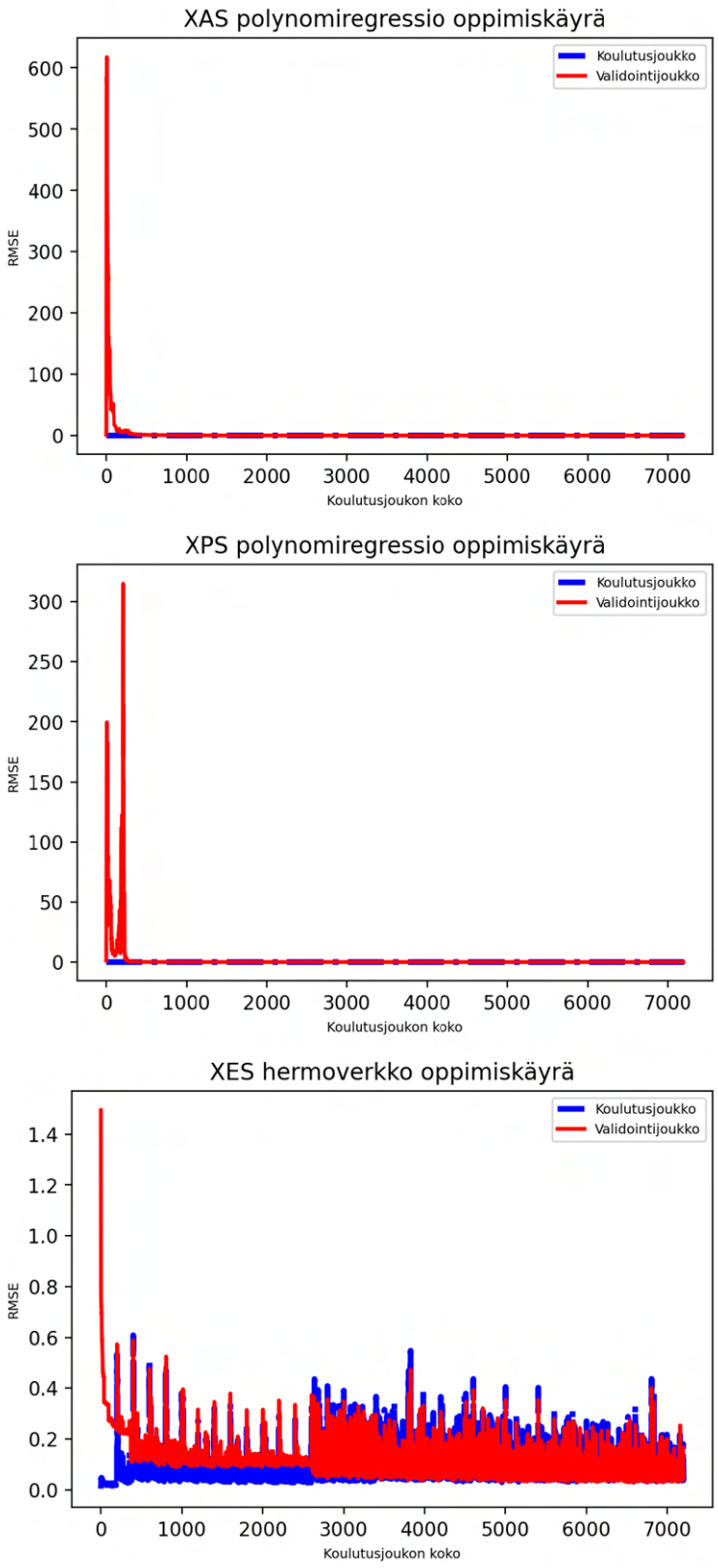
$$p\left(x|\mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{21}$$

missä x on satunnaismuuttuja (tässä tapauksessa kulma tai sidospituus), μ on sen keskiarvo ja σ sen keskihajonta [19]. Rakennejakaumaparametrivektori $\boldsymbol{p}$ sisälsi kulman ja sidospituuden keskiarvot μ sekä keskihajonnat σ, ja kulman ja sidospituuksien normaalijakauman tiheysfunktioiden arvot laskin kaavalla 21 käyttäen pistettä $\boldsymbol{r}_i$ sekä rakenneparametria vastaavia $\boldsymbol{p}$:n arvoja. Sidospituuksille käytin samoja $\boldsymbol{p}$:n arvoja tiheysfunktion laskemiseen. Seuraavaksi laskin erillisten tiheysfunktioiden arvojen tulon, ja kerroin sen vielä skaalaustekijällä s, jotta integraalin arvo ilman ennustettavaa spektriä normittuisi yhteen. Skaalaustekijä oli osa vektoria $\boldsymbol{p}$.

Taulukko II. Koneoppimismallien koulutukseen ja validointiin kuluneet ajat

	XAS	**XPS**	**XES**
Polynomiregressio	27,2 s	24,9 s	27,4 s
Päätöspuu	17,8 min	7,6 min	20,1 min
Satunnainen metsä	71,2 min	37,2 min	85,4 min
Hermoverkko	255,5 min	236,4 min	264,1 min

Kun matriisin kaikille riveille oli laskettu spektrit ja ne olivat kerrottu rakennejakauman arvoilla sekä tilavuusalkiolla, summasin tulokset yhteen keskiarvospektriksi. Näin määritin tunnettua rakennejakaumaparametrivektoria $\boldsymbol{p}_t$ sekä yriterakennejakaumaparametrivektoria $\boldsymbol{p}_0$ vastaavat keskiarvospektrit. Tunnetun spektrin laskin erillisellä funktiolla, ja yritettä vastaavan spektrin laskin osana sakkofunktiota. Itse sakkofunktiota hyödynsin optimoinnissa, ja siihen käytin `scikit-learn`in funktiota `optimize.minimize()` [24]. Se palautti optimoinnin tuloksen $\boldsymbol{p}_r$, jota lopulta vertasin $\boldsymbol{p}_t$:n.



Kuva 16. Parhaimpien mallien oppimiskäyrät. Kuvaajien tekemiseen kirjoitettu koodi pohjautuu Géronin kirjan [17] koodinpätkään sivulta 126

Taulukko III. Hyperparametrien testatut arvot koneoppimismalleille. Hyperparametrin nimenä olen antanut mallin `scikit-learn`-funktiossa esiintyvän nimen, polynomin astetta lukuunottamatta [24]. Optimoitavien hyperparametrien lisäksi polynomiregressiolle käytin hyperparametria `'solver' = 'svd'`, ja hermoverkolle `'activation' = 'relu'`, `'solver' = 'adam'` ja `'max_iter' = 10000000` [25]

Hyperparametri	**Arvot**
Polynomiregressio	
Polynomin aste	$1, 2, 3, 4, 5, 6, 7, 8, 9$
`'alpha'`	$10^{-10}, 10^{-9}, \ldots, 1, 10, 10^2, 10^3, 10^4$
Päätöspuu	
`'splitter'`	`'best'`, `'random'`
`'max_depth'`	$5, 10, 15, 20, 25, 30, 35, 40, 45, 50$
`'min_samples_split'`	$2, 5, 10, 15, 20, 25, 30, 35, 40$
`'min_samples_leaf'`	$1, 2, 5, 10, 15, 20$
`'max_leaf_nodes'`	`None`, $200, 150, 100, 50, 20$
`'max_features'`	`None, 'log2'`
Satunnainen metsä	
`'n_estimators'`	$50, 100, 200, 300$
`'max_depth'`	$10, 15, 20, 25, 30$
`'min_samples_split'`	$2, 5, 10, 15$
`'min_samples_leaf'`	$1, 2, 5, 10$
Hermoverkko	
`'hidden_layer_sizes'`	$(5, 5), (10, 10), (50, 50), (100, 100), (200, 200), (500, 500)$ $(5, 5, 5), (10, 10, 10), (50, 50, 50), (100, 100, 100),$ $(200, 200, 200), (500, 500, 500), (5, 5, 5, 5), (10, 10, 10, 10),$ $(50, 50, 50, 50), (100, 100, 100, 100), (200, 200, 200, 200)$ $(500, 500, 500, 500)$
`'alpha'`	$10^{-10}, 10^{-9}, \ldots, 1, 10, 10^2, 10^3, 10^4$

Taulukko IV. Parhaimpien mallien hyperparametrit XAS-, XPS- ja XES-spektrien ennustamiseen, sekä kyseisen mallin vastaavasta testijoukosta määritetty RMSE. Taulukon III tapaan hyperparametrin nimenä olen antanut mallin `scikit-learn`-funktiossa esiintyvän nimen [24]

XAS: polynomiregressio	
Hyperparametri	**Arvo**
Polynomin aste	9
`'alpha'`	1
`'solver'`	`'svd'`
RMSE $= 0,060246$	
XPS: polynomiregressio	
Polynomin aste	9
`'alpha'`	10^{-10}
`'solver'`	`'svd'`
RMSE $= 0{,}001098$	
XES: hermoverkko	
`'hidden_layer_sizes'`	$(200, 200, 200, 200)$
`'alpha'`	1
`'activation'`	`'relu'`
`'solver'`	`'adam'`
`'max_iter'`	10000000
RMSE $= 0{,}032783$	

3.3 Tulokset

Tein jokaiselle spektroskopiamenetelmälle kaksi testiä, joissa tutkin $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n vastaavuutta. Testeistä ensimmäisessä $\boldsymbol{p}_t$:n arvot pohjautuivat koulutusjoukon ominaisuuksien keskiarvoihin ja keskihajontoihin. Sikäli kun XAS-, XPS- ja XES-spektrien kouluttamiseen käytetyt ominaisuudet olivat samat, käytin ensimmäisessä testissä samaa $\boldsymbol{p}_t$:tä kaikille spektrityypeille. Laskin aluksi $\boldsymbol{p}_t$:stä tunnetun keskiarvospektrin, ja syötin sen `optimize.minimize()`-funktioon alkuarvauksen $\boldsymbol{p}_0$ kera. Optimoinnin tuloksen ollessa selvillä laskin RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n komponenttien väliltä. Taulukossa V on esitettynä käyttämäni $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_0$:n arvot, sekä optimoinnin tulokset.

Ensimmäisten testien tuloksista nähdään, että tunnetuista XAS- ja XES-keskiarvospektreistä onnistui `optimize.minimize()`-funktio löytämään niitä vastaavat rakennejakaumaparametrit varsin hyvin; molemmilla menetelmillä suurimman RMSE:n kertaluokka oli 10^{-6}. XPS:n tapauksessa kuitenkin $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvot poikkesivat enemmän toisistaan, esimerkiksi kulman keskiarvo μ_k jopa 3 ° verran. Näin ollen tunnettujen rakennejakaumaparametrien etsiminen XAS- ja XES-spektreistä vaikutti ainakin olevan mahdollista.

Toisissa testeissä määritin $\boldsymbol{p}_r$:n muutoin ensimmäisten testien tapaan, nyt vain toistin optimointiprosessin sattumanvaraisilla $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_0$:n arvoilla kymmeneen kertaan. Jokaisella iteraatiolla kulman keskiarvo μ_k sai arvon väliltä $90{,}0 - 116{,}0$, ja sidospituuden keskiarvo vuorostaan väliltä $0{,}8 - 1{,}2$. Valitsemani välit pohjautuivat kolutusjoukon keskiarvoon ja keskihajontaan niin, että välien ala- ja ylärajat olivat yhden keskihajonnan päässä keskiarvosta. Kulman keskihajonnan σ_k sekä sidospituuden keskihajonnan σ_s välien ala- ja ylärajat sen sijaan määritin mielivaltaisesti; σ_k sai arvoja väliltä $10{,}0 - 14{,}0$ ja σ_s väliltä $0{,}05 - 0{,}09$.

$\boldsymbol{p}_t$:n skaalauskertoimen s arvon laskin jokaisen iteraation yhteydessä seuraavasti: Määritin aluksi listan mahdollisista skaalauskertoimista s, jotka saivat arvoja välil-

tä $0{,}0 - 2{,}0$ jakovälillä 0,002. Tämän jälkeen suoritin integrointiprosessin jokaiselle skaalauskertoimen arvolle vektorilla $\boldsymbol{p}_t$, joka sisälsi sen hetkiset μ_k:n, μ_s:n, σ_k:n, σ_s:n ja s:n arvot. Integrointiprosessista jätin spektriennustajan pois, sillä skaalauskertoimen tarkoitus oli varmistaa, että ilman spektriennustajaa integroinnin tulokseksi saataisiin 1,0. Valitsin sen arvon $\boldsymbol{p}_t$:n skaalauskertoimeksi, jolla integroinnin tulos oli lähimpänä arvoa 1,0. Sen sijaan s vektorissa $\boldsymbol{p}_0$ oli aina 1,0.

Taulukosta VI löytyvät toisen testin viiden ensimmäisen iteraation tulokset XAS:lle, ja taulukosta VII iteraatioiden $6 - 10$. Samaan tapaan XPS:n toisen testin tulokset nähdään taulukoista VIII ja IX, ja XES:n taulukoista X ja XI. XAS:n tapauksessa seitsemälle iteraatiolle kymmenestä onnistui `optimize.minimize()`-funktio laskemaan $\boldsymbol{p}_r$:n, joka vastasi erittäin hyvin $\boldsymbol{p}_t$:tä. Iteraatioilla 1, 3, 5, 6, 7, 9 ja 10 suurimmat RMSE:t olivat kertaluokkaa 10^{-5}, tai tätäkin pienempiä. Niin ikään XES:lle onnistuivat optimoinnit hyvin, kahdeksassa iteraatiossa kymmenestä olivat suurimmat RMSE:t joko kertaluokkaa 10^{-6}, tai alle. Näin oli asian laita iteraatiolla 1, 2, 4, 5, 6, 7, 8 ja 9. Sen sijaan XPS:llä oli vain iteraatiolla 6 suurin RMSE kertaluokkaa 10^{-5}, muilla iteraatioilla olivat RMSE:t varsin korkeita kautta laidan. Vaikuttaisi siis siltä, että ainakin XAS ja XES olisivat sensitiivisiä rakennejakauman muutoksille.

Taulukko V. Ensimmäisen testin tulokset kutakin spektroskopiamenetelmää kohden. Riveillä on esitettynä rakennejakaumaparametrit kulman keskiarvo μ_k, sidospituuden keskiarvo μ_s, kulman keskihajonta σ_k, sidospituuden keskihajonta σ_s ja skaalauskerroin s, ja sarakkeissa niiden arvot jakaumissa $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä rakennejakaumaparametrin RMSE laskettuna $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä

XAS				
	$\boldsymbol{p}_t$	$\boldsymbol{p}_0$	$\boldsymbol{p}_r$	RMSE
μ_k	103,2	100,0	103,2	$8 \cdot 10^{-7}$
μ_s	1,00	1,00	1,00	$3 \cdot 10^{-9}$
σ_k	12,7	12,0	12,7	$6 \cdot 10^{-6}$
σ_s	0,07	0,05	0,07	$8 \cdot 10^{-9}$
s	1,000	1,000	1,000	$6 \cdot 10^{-11}$
XPS				
	$\boldsymbol{p}_t$	$\boldsymbol{p}_0$	$\boldsymbol{p}_r$	RMSE
μ_k	103,2	100,0	100,0	3,2
μ_s	1,00	1,00	1,06	0,05
σ_k	12,7	12,0	12,2	0,5
σ_s	0,07	0,05	0,08	0,002
s	1,000	1,000	1,000	0,0001
XES				
	$\boldsymbol{p}_t$	$\boldsymbol{p}_0$	$\boldsymbol{p}_r$	RMSE
μ_k	103,2	100,0	103,2	$2 \cdot 10^{-7}$
μ_s	1,00	1,00	1,00	$7 \cdot 10^{-9}$
σ_k	12,7	12,0	12,7	$2 \cdot 10^{-6}$
σ_s	0,07	0,05	0,07	$3 \cdot 10^{-8}$
s	1,000	1,000	1,000	$10 \cdot 10^{-9}$

Taulukko VI. Toisen testin iteraatioiden $1-5$ tulokset XAS:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t1}$	91,9	1,01	11,5	0,06	1,000
$\boldsymbol{p}_{01}$	110,4	1,07	12,5	0,08	1,000
$\boldsymbol{p}_{r1}$	91,9	1,01	11,5	0,06	1,000
RMSE_1	$6 \cdot 10^{-7}$	$4 \cdot 10^{-9}$	$6 \cdot 10^{-6}$	$1 \cdot 10^{-8}$	$1 \cdot 10^{-8}$
$\boldsymbol{p}_{t2}$	111,6	1,09	14,0	0,09	1,000
$\boldsymbol{p}_{02}$	101,7	0,87	12,4	0,08	1,000
$\boldsymbol{p}_{r2}$	98,4	0,95	1,7	0,02	74,852
RMSE_2	13,1	0,14	12,3	0,07	73,852
$\boldsymbol{p}_{t3}$	108,8	1,08	13,8	0,08	1,000
$\boldsymbol{p}_{03}$	111,1	1,04	13,5	0,07	1,000
$\boldsymbol{p}_{r3}$	108,8	1,08	13,8	0,08	1,000
RMSE_3	$3 \cdot 10^{-5}$	$8 \cdot 10^{-8}$	$2 \cdot 10^{-5}$	$5 \cdot 10^{-8}$	$2 \cdot 10^{-7}$
$\boldsymbol{p}_{t4}$	109,8	1,02	14,0	0,08	1,000
$\boldsymbol{p}_{04}$	107,4	0,84	12,5	0,07	1,000
$\boldsymbol{p}_{r4}$	155,8	0,90	25,5	0,001	0,310
RMSE_4	46,0	0,12	11,5	0,08	0,690
$\boldsymbol{p}_{t5}$	97,2	0,97	12,4	0,08	1,000
$\boldsymbol{p}_{05}$	114,1	0,80	11,9	0,07	1,000
$\boldsymbol{p}_{r5}$	97,2	0,97	12,4	0,08	1,000
RMSE_5	$7 \cdot 10^{-6}$	$2 \cdot 10^{-8}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-8}$	$5 \cdot 10^{-8}$

Taulukko VII. Toisen testin iteraatioiden $6-10$ tulokset XAS:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t6}$	110,3	1,06	13,4	0,08	1,000
$\boldsymbol{p}_{06}$	113,4	1,00	13,9	0,08	1,000
$\boldsymbol{p}_{r6}$	110,3	1,06	13,4	0,08	1,000
RMSE_6	$5 \cdot 10^{-5}$	$1 \cdot 10^{-7}$	$3 \cdot 10^{-5}$	$8 \cdot 10^{-8}$	$7 \cdot 10^{-8}$
$\boldsymbol{p}_{t7}$	108,0	1,15	12,2	0,06	1,004
$\boldsymbol{p}_{07}$	97,6	1,15	12,9	0,05	1,000
$\boldsymbol{p}_{r7}$	108,0	1,15	12,2	0,06	1,004
RMSE_7	$10 \cdot 10^{-5}$	$3 \cdot 10^{-8}$	$8 \cdot 10^{-5}$	$9 \cdot 10^{-8}$	$5 \cdot 10^{-6}$
$\boldsymbol{p}_{t8}$	106,4	0,81	12,1	0,05	0,982
$\boldsymbol{p}_{08}$	106,9	0,93	12,6	0,09	1,000
$\boldsymbol{p}_{r8}$	106,9	0,81	12,6	0,05	1,002
RMSE_8	0,5	0,001	0,6	0,001	0,020
$\boldsymbol{p}_{t9}$	105,8	1,00	12,0	0,06	0,998
$\boldsymbol{p}_{09}$	102,9	1,15	11,9	0,07	1,000
$\boldsymbol{p}_{r9}$	105,8	1,00	12,0	0,06	0,998
RMSE_9	$3 \cdot 10^{-8}$	$1 \cdot 10^{-9}$	$2 \cdot 10^{-7}$	$4 \cdot 10^{-9}$	$2 \cdot 10^{-8}$
$\boldsymbol{p}_{t10}$	114,7	0,97	10,6	0,06	1,000
$\boldsymbol{p}_{010}$	113,9	1,20	10,1	0,08	1,000
$\boldsymbol{p}_{r10}$	114,7	0,97	10,6	$-0,06$	1,000
RMSE_{10}	$2 \cdot 10^{-6}$	$7 \cdot 10^{-9}$	$7 \cdot 10^{-6}$	0,1	$1 \cdot 10^{-8}$

Taulukko VIII. Toisen testin iteraatioiden $1-5$ tulokset XPS:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t1}$	102,3	1,20	10,1	0,07	1,000
$\boldsymbol{p}_{01}$	93,8	0,89	13,6	0,09	1,000
$\boldsymbol{p}_{r1}$	114,7	0,93	24,2	0,07	1,065
RMSE_1	12,4	0,26	14,1	0,002	0,065
$\boldsymbol{p}_{t2}$	108,7	1,07	12,2	0,06	0,996
$\boldsymbol{p}_{02}$	96,0	0,93	13,8	0,08	1,000
$\boldsymbol{p}_{r2}$	121,5	0,87	16,1	0,02	1,419
RMSE_2	12,8	0,21	3,9	0,04	0,423
$\boldsymbol{p}_{t3}$	107,0	1,07	11,6	0,08	1,000
$\boldsymbol{p}_{03}$	98,2	0,98	10,6	0,09	1,000
$\boldsymbol{p}_{r3}$	104,0	1,14	11,9	0,03	2,561
RMSE_3	2,9	0,07	0,3	0,05	1,561
$\boldsymbol{p}_{t4}$	90,7	0,92	13,1	0,08	1,002
$\boldsymbol{p}_{04}$	103,1	1,02	10,6	0,09	1,000
$\boldsymbol{p}_{r4}$	96,4	0,90	10,7	0,0002	0,028
RMSE_4	5,7	0,02	2,4	0,08	0,974
$\boldsymbol{p}_{t5}$	107,2	1,07	12,7	0,05	1,006
$\boldsymbol{p}_{05}$	90,6	0,94	13,3	0,06	1,000
$\boldsymbol{p}_{r5}$	107,5	1,07	12,7	0,05	1,006
RMSE_5	0,3	0,004	0,01	0,003	0,0005

Taulukko IX. Toisen testin iteraatioiden $6-10$ tulokset XPS:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t6}$	109,7	1,16	13,8	0,05	1,014
$\boldsymbol{p}_{06}$	108,1	0,92	11,8	0,05	1,000
$\boldsymbol{p}_{r6}$	109,7	1,16	13,8	0,05	1,014
RMSE_6	$6 \cdot 10^{-5}$	$7 \cdot 10^{-7}$	$3 \cdot 10^{-5}$	$1 \cdot 10^{-6}$	$3 \cdot 10^{-6}$
$\boldsymbol{p}_{t7}$	115,7	0,90	11,6	0,05	0,986
$\boldsymbol{p}_{07}$	94,9	1,10	10,9	0,07	1,000
$\boldsymbol{p}_{r7}$	96,6	1,20	10,8	0,002	0,009
RMSE_7	19,1	0,30	0,8	0,05	0,977
$\boldsymbol{p}_{t8}$	101,9	1,17	10,6	0,07	1,000
$\boldsymbol{p}_{08}$	115,3	1,07	10,8	0,07	1,000
$\boldsymbol{p}_{r8}$	113,5	0,98	15,3	0,01	2,561
RMSE_8	11,6	0,20	4,7	0,06	1,561
$\boldsymbol{p}_{t9}$	97,8	0,85	11,2	0,06	1,002
$\boldsymbol{p}_{09}$	111,7	1,18	10,8	0,07	1,000
$\boldsymbol{p}_{r9}$	103,0	1,13	1,2	0,12	6,583
RMSE_9	5,2	0,29	10,0	0,06	5,581
$\boldsymbol{p}_{t10}$	114,4	1,14	11,6	0,07	1,000
$\boldsymbol{p}_{010}$	103,0	1,19	10,5	0,06	1,000
$\boldsymbol{p}_{r10}$	103,2	1,20	10,5	0,0002	0,004
RMSE_{10}	11,2	0,06	1,1	0,07	0,996

Taulukko X. Toisen testin iteraatioiden $1-5$ tulokset XES:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t1}$	92,4	0,82	10,6	0,09	1,002
$\boldsymbol{p}_{01}$	93,4	1,19	13,1	0,08	1,000
$\boldsymbol{p}_{r1}$	92,4	0,82	10,6	0,09	1,002
RMSE_1	$2 \cdot 10^{-7}$	$5 \cdot 10^{-9}$	$4 \cdot 10^{-6}$	$5 \cdot 10^{-8}$	$2 \cdot 10^{-8}$
$\boldsymbol{p}_{t2}$	91,9	1,17	13,4	0,06	1,002
$\boldsymbol{p}_{02}$	95,8	1,20	10,6	0,08	1,000
$\boldsymbol{p}_{r2}$	91,9	1,17	13,4	0,06	1,002
RMSE_2	$4 \cdot 10^{-7}$	$3 \cdot 10^{-8}$	$1 \cdot 10^{-7}$	$1 \cdot 10^{-7}$	$1 \cdot 10^{-8}$
$\boldsymbol{p}_{t3}$	95,9	0,82	10,3	0,05	0,996
$\boldsymbol{p}_{03}$	111,1	1,17	12,5	0,08	1,000
$\boldsymbol{p}_{r3}$	95,9	0,84	10,9	0,02	13,051
RMSE_3	0,05	0,02	0,6	0,03	12,055
$\boldsymbol{p}_{t4}$	113,2	1,16	12,2	0,07	1,000
$\boldsymbol{p}_{04}$	103,2	0,87	10,9	0,06	1,000
$\boldsymbol{p}_{r4}$	113,2	1,16	12,2	0,07	1,000
RMSE_4	$5 \cdot 10^{-7}$	$1 \cdot 10^{-8}$	$1 \cdot 10^{-6}$	$4 \cdot 10^{-8}$	$1 \cdot 10^{-9}$
$\boldsymbol{p}_{t5}$	91,1	1,06	13,1	0,06	1,008
$\boldsymbol{p}_{05}$	99,4	0,86	13,1	0,08	1,000
$\boldsymbol{p}_{r5}$	91,1	1,06	13,1	0,06	1,008
RMSE_5	$4 \cdot 10^{-7}$	$2 \cdot 10^{-8}$	$4 \cdot 10^{-6}$	$1 \cdot 10^{-7}$	$1 \cdot 10^{-7}$

Taulukko XI. Toisen testin iteraatioiden $6-10$ tulokset XES:lle. Jokaisen iteraation vektorit $\boldsymbol{p}_t$, $\boldsymbol{p}_0$ ja $\boldsymbol{p}_r$ sekä RMSE:t $\boldsymbol{p}_t$:n ja $\boldsymbol{p}_r$:n arvojen väliltä ovat esitettyinä riveillä, ja rakennejakaumaparametrit itsessään sarakkeilla

	μ_k	μ_s	σ_k	σ_s	s
$\boldsymbol{p}_{t6}$	112,6	0,93	13,0	0,09	1,000
$\boldsymbol{p}_{06}$	111,5	0,86	10,7	0,08	1,000
$\boldsymbol{p}_{r6}$	112,6	0,93	13,0	0,09	1,000
RMSE_6	$4 \cdot 10^{-7}$	$8 \cdot 10^{-9}$	$3 \cdot 10^{-6}$	$3 \cdot 10^{-8}$	$9 \cdot 10^{-9}$
$\boldsymbol{p}_{t7}$	95,5	0,92	11,3	0,07	1,000
$\boldsymbol{p}_{07}$	102,5	0,91	10,3	0,06	1,000
$\boldsymbol{p}_{r7}$	95,5	0,92	11,3	0,07	1,000
RMSE_7	$2 \cdot 10^{-7}$	$8 \cdot 10^{-9}$	$2 \cdot 10^{-6}$	$4 \cdot 10^{-8}$	$1 \cdot 10^{-8}$
$\boldsymbol{p}_{t8}$	112,5	0,82	13,6	0,08	1,000
$\boldsymbol{p}_{08}$	109,6	0,81	11,2	0,074	1,000
$\boldsymbol{p}_{r8}$	112,5	0,82	13,6	0,08	1,000
RMSE_8	$1 \cdot 10^{-6}$	$1 \cdot 10^{-8}$	$10 \cdot 10^{-7}$	$2 \cdot 10^{-10}$	$1 \cdot 10^{-8}$
$\boldsymbol{p}_{t9}$	114,3	1,01	11,8	0,08	1,000
$\boldsymbol{p}_{09}$	105,4	1,07	12,0	0,08	1,000
$\boldsymbol{p}_{r9}$	114,3	1,01	11,8	0,08	1,000
RMSE_9	$1 \cdot 10^{-7}$	$1 \cdot 10^{-8}$	$3 \cdot 10^{-6}$	$4 \cdot 10^{-8}$	$6 \cdot 10^{-9}$
$\boldsymbol{p}_{t10}$	106,7	0,90	10,6	0,08	1,000
$\boldsymbol{p}_{010}$	111,4	1,00	13,6	0,09	1,000
$\boldsymbol{p}_{r10}$	107,1	0,90	14,0	0,002	0,009
RMSE_{10}	0,4	0,01	3,4	0,08	0,991

4 Johtopäätökset

Saamien tulosten perusteella voin sanoa, että simuloitujen vesimolekyylien rakennejakaumaparametrit oli mahdollista palauttaa molekyyleille määrityistä XAS- ja XES-spektreistä varsin hyvin. Ensimmäisissä testeissä $\boldsymbol{p}_t$:n perustuessa spektriennustajien koulutusjoukkoihin sekä $\boldsymbol{p}_0$:n ollessa melko lähellä rakennejakaumaparametrien tunnettuja arvoja, onnistui optimointialgoritmi laskemaan tuloksen $\boldsymbol{p}_r$, joka oli hyvinkin lähellä $\boldsymbol{p}_t$:tä. Toisetkin testit puolsivat tätä johtopäätöstä, sillä suurimmassa osassa optimoinneista $\boldsymbol{p}_t$ ja $\boldsymbol{p}_r$ vastasivat toisiaan hyvin. Tästä herääkin kysymys: Miksi eivät kaikki testien kaksi optimoinnit tuottaneet yhtä hyviä tuloksia? Voi olla, että alkuarvaus $\boldsymbol{p}_0$ oli liian kaukana $\boldsymbol{p}_t$:stä, tai että sattumanvaraisesti määritetylle rakennejakaumaparametrien yhdistelmälle vektorissa $\boldsymbol{p}_t$ ei ollut olemassa fysikaalisesti järkevää vastinetta.

XPS-spektreille eivät kummatkaan testit onnistuneet yhtä hyvin kuin XAS:n ja XES:n tapauksessa. Tämä viittaisi siihen, ettei XPS ole läheskään niin sensitiivinen rakennejakauman muutoksille kuin XAS ja XES, jotka molemmat vaikuttivat kutakuinkin yhtä sensitiivisiltä. Jo ensimmäisessä testissä XPS:lle tulokset eivät olleet kovinkaan lähellä $\boldsymbol{p}_t$:tä, ja toisessa testissä useimmilla optimoinneilla $\boldsymbol{p}_r$:t poikkesivat $\boldsymbol{p}_t$:stä melkoisesti, esimerkiksi kulman arvot usean asteen verran. Kysymykseen miksi XPS ei ole yhtä sensitiivinen kuin XAS ja XES en osaa suoranaisesti vastata, mutta spekuloida voin kuitenkin. Ehkäpä orbitaalien energioiden suoranainen luotaaminen absorpoituneiden/emittoituneiden fotonien muodossa antaa molekyylistä tarkemman kuvan kuin epäsuorasti valosähköisen ilmiön vaikutuksesta irronneiden elektronien havainnoiminen antaisi.

Kaiken kaikkiaan vaikuttaisi siltä, että ainakin vesimolekyylien tietyt röntgenspektrit todellakin sisältävät rakenneinformaatiota molekyyleistä itsestään. Koneoppiminen myöskin soveltui erinomaisesti keskiarvospektrin laskemiseen integrointivaiheessa tunnetun rakennejakauman etsimisessä. Nämä molemmat lienevät mieleisiä

tuloksia tulevaisuuden kannalta, sillä näin tutkimalla pelkästään molekyylien röntgenspektrejä saataisiin tietoa niiden rakennejakaumasta kohtalaisen vaivattomasti. Lisäksi koneoppimisen hyödyntäminen voi nopeuttaa monia laskelmia huomattavasti, kuten tämän tutkielman kaltaista optimointia, joka olisi ollut paljon hitaampaa jos optimoinnin yhteydessä olisin laskenut spektrit käyttäen tiheysfunktionaaliteoriaa [3].

Viitteet

[1] P. Zimmermann, S. Peredkov, P. M. Abdala, S. DeBeer, M. Tromp, C. Müller ja J. A. van Bokhoven, Coordination Chemistry Reviews **423**, 213466 (2020).

[2] D. V. Schroeder, *An introduction to thermal physics* (Addison-Wesley LongmanSan Francisco (CA), 2000).

[3] M. P. Allen ja D. J. Tildesley, *Computer Simulation of Liquids* (Clarendon PressUSA, 1989).

[4] D. Attwood, *Soft X-Rays and Extreme Ultraviolet Radiation: Principles and Applications* (Cambridge University Press, 1999).

[5] G. Greczynski ja L. Hultman, PROGRESS IN MATERIALS SCIENCE **107**, (2020).

[6] W. C. Röntgen, Annalen der Physik **300**, 12 (1898).

[7] J. Als-Nielsen ja D. McMorrow, *Elements of Modern X-ray Physics*, 2nd ed. (John Wiley & Sons, Ltd, 2011).

[8] J. Blewett, JOURNAL OF SYNCHROTRON RADIATION **5**, 135 (1998).

[9] F. Eder, A. Gurewitsch, R. Langmuir ja H. Pollock, PHYSICAL REVIEW **71**, 829 (1947).

[10] C. Feng ja H.-X. Deng, NUCLEAR SCIENCE AND TECHNIQUES **29**, (2018).

[11] R. Abela, A. Alarcon, J. Alex, C. Arrell, V. Arsov, S. Bettoni, M. Bopp, C. Bostedt, H.-H. Braun, M. Calvi, T. Celcer, P. Craievich, A. Dax, P. Dijkstal, S. Dordevic, E. Ferrari, U. Flechsig, R. Follath, F. Frei, N. Gaiffi, Z. Geng, C. Gough, N. Hiller, S. Hunziker, M. Huppert, R. Ischebeck, H. Johri, P. Juranic, R. Kalt, M. Kaiser, B. Keil, C. Kittel, R. Kunzi, T. Lippuner, F. Lohl, F. Marcellini, G. Marinkovic, C. O. Loch, G. L. Orlandi, B. Patterson, C. Pradervand, M. Paraliev, M. Pedrozzi, E. Prat, P. Ranitovic, S. Reiche, C. Rosenberg, S. Sanfilippo, T. Schietinger, T. Schmidt, K. Schnorr, C. Svetina, A. Trisorio, C. Vicario, D. Voulot, U. Wagner, H. J. Worner, A. Zandonella, L. Patthey ja R. Ganter, JOURNAL OF SYNCHROTRON RADIATION **26**, 1073 (2019).

[12] J. Yano ja V. K. Yachandra, PHOTOSYNTHESIS RESEARCH **102**, 241 (2009).

[13] T. Fransson, Y. Harada, N. Kosugi, N. A. Besley, B. Winter, J. J. Rehr, L. G. M. Pettersson ja A. Nilsson, CHEMICAL REVIEWS **116**, 7551 (2016).

[14] G. Greczynski ja L. Hultman, Progress in materials science **107**, 100591 (2020).

[15] P. Atkins, *Molecular quantum mechanics*, fifth edition. ed. (Oxford University PressOxford, 2011 - 2011).

[16] U. Bergmann ja P. Glatzel, Photosynthesis research **102**, 255 (2009).

[17] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 1st ed. (O'Reilly Media, Inc., 2017).

[18] N. Muthukrishnan, F. Maleki, K. Ovens, C. Reinhold, B. Forghani ja R. Forghani, NEUROIMAGING CLINICS OF NORTH AMERICA **30**, 393+ (2020).

[19] M. P. Deisenroth, *Mathematics for machine learning* (Cambridge university pressCambridge, 2020 - 2020).

[20] W.-Y. Loh, WILEY INTERDISCIPLINARY REVIEWS-DATA MINING AND KNOWLEDGE DISCOVERY **1**, 14 (2011).

[21] K. Fawagreh, M. M. Gaber ja E. Elyan, Systems science & control engineering **2**, 602 (2014).

[22] M. Gardner ja S. Dorling, Atmospheric Environment **32**, 2627 (1998).

[23] I. Goodfellow, Y. Bengio ja A. Courville, *Deep Learning* (MIT Press, 2016).

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot ja E. Duchesnay, Journal of Machine Learning Research **12**, 2825 (2011).

[25] J. Niskanen, A. Vladyka, J. A. Kettunen ja C. J. Sahle, Neural networks in interpretation of electronic core-level spectra, 2021.

[26] R. G. Mantovani, T. Horváth, R. Cerri, S. B. Junior, J. Vanschoren ja A. C. P. d. L. F. de Carvalho, (2018).