Työkyvyttömyyseläkeratkaisun ennustaminen koneoppimismenetelmin

Suppea SHV-työ

Sami Hårdh

25.4.2022

Abstract

A Finnish pension insurance company provides statutory earnings-related pension insurance. In addition to old-age pensions, insurance provides protection against disability. Insuring against disability causes pension insurance company to accumulate data about disability pension applications. The main purpose of this data is to decide an applicant's entitlement to a disability pension. However, a remarkable volume of data can make it possible to apply machine learning techniques to solve complex problems around the phenomenon of disability.

In this paper, machine learning is applied to solve a binary classification problem to predict the outcome of a disability pension application. This is done with the Finnish pension insurance company Varma's data about disability pension applications from the last ten years. The classification problem will be solved with four machine learning models, including logistic regression, random forest and two versions of gradient boosting. The paper also includes basics about interpreting the behavior of a machine learning model.

As a result, the best machine learning model achieves an accuracy of 84 %. In addition, some basic behaviors of the model can be explained. The Applicant's age seems to have the highest effect on the model output. But also, some combinations of the applicant's diagnosis and other explanatory variables can be observed in terms of model sensitivity.

At the end of the paper, it will be briefly considered how the prediction could be useful from the point of view of a pension insurance company and an actuary. Three points of view will be introduced. The first one focuses on how a disability pension handling process could be optimized in the case of desirable handling time. The second one gives views about the quality control of disability pension decisions. The last one focuses on the short-term prediction of disability pension expenses.

Sisällys

1. Johdanto

Työeläkevakuutusyhtiön keskeinen tehtävä on hoitaa lakisääteisen eläketurvan toimeenpanoa. Vaikka kaikkein suurin osa vakuutusliikkeestä koostuu vanhuuseläkkeistä, merkittävä osuus työeläkevakuutusyhtiön harjoittamasta vakuutusliikkeestä koostuu myös työkyvyttömyyseläkkeistä sekä työeläkekuntoutuksesta.

Työkyvyttömyysliikkeen harjoittamisesta kertyy paljon dataa työeläkevakuutusyhtiölle. Tämä data sisältää runsaasti tietoa työkyvyttömyyseläkkeenhakijoista. Vaikka kertyvien tietojen pääasiallinen käyttötarkoitus on yksittäisten eläkkeenhakijoiden eläkeoikeudesta päättäminen, näiden tietojen hyödyntäminen on keskeisessä roolissa esimerkiksi työkyvyttömyyteen johtavien taustatekijöiden ymmärtämisessä, työkyvyttömyyden ennaltaehkäisyssä sekä viime kädessä työkyvyttömyysetuuksien rahoituksen järjestämisessä. Edellä mainittuja on perinteisesti lähestytty tilastoanalyysien keinoin, mutta datan suuri määrä voi mahdollistaa myös koneoppimisen hyödyntämisen.

Tässä työssä keskitytään koneoppimismallien tarjoamiin mahdollisuuksiin työkyvyttömyysliikkeestä syntyvän datan hyödyntämiseksi. Työssä pyritään vastaamaan kysymykseen siitä, voitaisiinko työeläkevakuutusyhtiöön saapuvaa työkyvyttömyyseläkehakemusta seuraavaa myönteistä tai kielteistä ratkaisua eläkeoikeudesta ennustaa datasta, sekä minkälaisia sovellusalueita tämä voisi mahdollistaa työeläkevakuutusyhtiön sekä aktuaarin näkökulmasta. Työn tarkoituksena ei ole ehdottaa eläkeoikeuden ratkaisemisen automatisointia koneoppimista hyödyntäen.

Työssä kuvataan aluksi lyhyesti mitä koneoppiminen on. Tämän jälkeen esitellään joitakin kahden vaihtoehdon luokitteluongelmien ratkaisemiseen sopivia koneoppimismenetelmiä. Samassa yhteydessä esitellään myös tapoja tulkita koneoppimismenetelmällä tuotetun luokittelumallin hyvyyttä sekä sen käyttäytymistä. Seuraavaksi hyödynnetään esiteltyjä tekniikoita ennustamaan työkyvyttömyyseläkehakemuksen ratkaisua sekä selittämään luokittelumallin tuottamien ennusteiden käyttäytymistä. Lopuksi esitellään käytännön sovellusalueita, joita työkyvyttömyyseläkehakemusten ratkaisujen ennustaminen voisi mahdollistaa.

2. Koneoppiminen lyhyesti

Tässä kappaleessa esitellään lyhyesti mitä koneoppiminen on. Asiaa lähestytään vertaamalla koneoppimista perinteisempään tilastolliseen mallinnukseen. Lisäksi pohditaan mitä oppiminen voisi tarkoittaa koneälystä puhuttaessa.

2.1. Tilastollinen mallinnus ja koneoppiminen

Tilastollinen mallinnus lähtee yleensä siitä, että valitaan joukko tilastollisia malleja, joiden katsotaan kuvaavan mahdollisimman hyvin mallinnuksen kohteena olevien selittävien muuttujien ja vastemuuttujan välistä yhteyttä. Käytettävät mallit sovitetaan dataan ja näistä valitaan paras. Tilastollisessa mallinnuksessa kuitenkin sopivien mallien valinta voi olla haasteellista. Etenkin jos selittäviä muuttujia on paljon, puhumattakaan erilaisten monimutkaisten ristikorrelaatioiden huomioimisesta, voi mallin valinta olla erittäin haastavaa.

Koneoppimisen tavoite on lähtökohtaisesti sama kuin tilastollisessa mallinnuksessa, eli pyritään löytämään paras mahdollinen malli kuvaamaan selittävien muuttujien ja vastemuuttujan välistä yhteyttä. Koneoppimisessa kuitenkin oikeanlaisen mallin valintaan liittyvä vaihe pyritään automatisoimaan siten, että useiden mallien valinnan ja vertailun sijaan pyritään oppimaan suoraan datasta paras mahdollinen malli. Käytännössä kuitenkin on tehtävä valinta sen osalta, minkälaisella koneoppimismenetelmällä varsinainen malli pyritään löytämään.

Pienimmän neliösumman menetelmää käyttäen, matemaattisesti muotoiltuna tilastollisessa mallinnuksessa pyrkimyksenä on valita paras mahdollinen malli ennalta valittujen mallien joukosta, ratkaisemalla yksittäisten mallien minimointi ongelma

$$\min_{p} \sum_{i} (y_i - f(x_i, p))^2$$

missä y_i on datapistettä i vastaava vastemuuttuja, x_i on datapistettä i vastaava selittäviä muuttujien joukko, f potentiaalinen tilastollinen malli ja p mallin estimoitavat parametrit. Koneoppimisessa taas pyritään päinvastoin löytämään suoraan paras mahdollinen malli ratkaisemalla minimointiongelma

$$\min_{f} \sum_{i} (y_i - f(x_i))^2$$

oppimalla datan lainalaisuudet koneoppimismenetelmää hyödyntäen.

Tilastollisten mallien käyttäytymisen selittäminen on melko suoraviivaista, sillä selittävien muuttujien vaikutukset ovat analysoitavissa estimoitujen parametrien kautta. Koneoppimisessa selitettävyys on usein haasteellisempaa, sillä lopputuotteena syntyneen mallin havaitseminen tai esittäminen

funktionaalisessa muodossa voi olla hankalaa, jopa lähes mahdotonta. Etenkin monimutkaisemmilla koneoppimismenetelmillä, puhumattakaan syväoppimisessa käytetyistä neuroverkoista, opittu malli saatetaan nähdä mustana laatikkona, jonka selittäminen on oma taiteenlajinsa. Toisaalta monimutkaisemmat koneoppimismenetelmät pystyvät parhaimmillaan oppimaan datasta todella monimutkaisia ilmiöitä, jotka voivat lisätä mallin ennustetarkkuutta. Yleensä selitettävyyteen pystytään vaikuttamaan valitsemalla yksinkertaisempi koneoppimismenetelmä. Tällöin tosin saatetaan luopua osasta potentiaalista ennustetarkkuutta.

Kiteytettynä voitaneen sanoa, että tilastollisessa mallinnuksessa valittavat mallit ovat keskiössä, kun taas koneoppimisessa keskiössä on itse data. Käyttötapauksien osalta tarkkojen tilastollisten syy-yhteyksien tunnistaminen ja osoittaminen voisi puoltaa tilastollisen mallinnuksen käyttämistä, kun taas äärimmäisen ennustetarkkuuden tavoittelu voisi puoltaa koneoppimisen hyödyntämistä.

2.2. Oppiminen ja jyrkimmän laskun menetelmä

Kuten edellä kuvattiin, koneoppimisen perusperiaatteena on löytää malli, joka kuvaa selittävien muuttujien ja vastemuuttujan välisen yhteyden mahdollisimman hyvin. Kysymys kuuluukin seuraavaksi, mikä kuvaa edellä mainittua hyvyyttä. Mallin hyvyyttä voidaan kuvata käyttäen niin sanottua virhefunktiota, joka kuvastaa mallin ennustamien arvojen ja vastemuuttujan todellisten arvojen välistä poikkeamaa. Tarkoituksena onkin löytää sellainen malli, jolla virhefunktio saavuttaa pienimmän mahdollisen arvonsa. Käytettäessä virhefunktiona residuaalien neliösummaa, esimerkiksi lineaariselle regressiolle löytyy tunnettu analyyttinen ratkaisu. Koneoppimisessa tilanne ei kuitenkaan ole näin yksinkertainen, sillä mallille ei yleensä ole analyyttistä ratkaisutapaa, tai se on liian monimutkainen.

Koneoppimisessa edellä mainitun virhefunktion minimointi voidaan joissain tapauksissa toteuttaa hyödyntäen niin sanottua jyrkimmän laskun menetelmää (gradient descent). Sen perusajatuksena on liikkua pitkin mallin taustalla olevaa parametriavaruutta etsien virhefunktion minimoivaa pistettä. Jyrkimmän laskun menetelmässä minimointiongelman ratkaisevan parametriavaruuden pisteen etsiminen ei kuitenkaan tapahdu sattumanvaraisesti. Merkitään askeleen t sijaintia parametriavaruudessa merkinnällä a_t. Tällöin askel pisteestä a_t pisteeseen a_{t+1} toteuttaa säännön

$$a_{t+1} = a_t - \nabla L(a_t) \cdot lr,$$

missä $\nabla L(a_t)$ on virhefunktion gradientti ja lr on niin sanottu oppimisaste (learning rate). Kaava tarkoittaa, että parametriavaruuden askeleen suunnan ja suuruuden määrää negatiivinen gradientti, joka osoittaa virhefunktion nopeimman pienenemisen suuntaan. Lisäksi gradienttia skaalataan oppimisasteella, jonka sopiva valinta on tärkeää. Liian pieni oppimisaste tarkoittaa pieniä liikkeitä, jolloin virhefunktion minimoivan pisteen saavuttaminen voi kestää todella kauan, jopa äärettömän

kauan. Jos taas oppimisaste on liian iso, voi liikehdintä olla liian suurta. Tämä johtaa pahimmillaan tilanteeseen, että askeleet hyppivät virhefunktion minimoivan pisteen ympärillä saavuttamatta sitä koskaan. Käytännön tilanteissa jyrkimmän laskun menetelmässä voidaan joutua hyödyntämään satunnaisuutta siten, että menetelmää sovelletaan useita kertoja valitsemalla parametriavaruuden lähtöpiste sattumanvaraisesti. Tällä pyritään varmistamaan parametriavaruuden todellisen minimin löytäminen sen sijaan, että päädyttäisiin johonkin paikalliseen minimiin.

Jyrkimmän laskun menetelmä sopii tilanteisiin, joissa virhefunktion gradientti on laskettavissa suhteessa parametreihin tai painokertoimiin. Koneoppimismenetelmiin lukeutuu myös ei-parametrisoitavia malleja, joihin kyseinen tekniikka ei ole sovellettavissa. Hyvä esimerkki näistä ovat puumallit, joihin palataan tuonnempana.

Jyrkimmän laskun menetelmä kuvastaa kuitenkin ehkäpä parhaiten koneoppimisessa käytetyn termin "oppiminen" ideologiaa. Se että mallia, ja tarkemmin sen parametreja tai painokertoimia, muokataan pikkuhiljaa perustuen gradienttiin ja oppimisasteeseen, voidaan nähdä koneen tapana oppia.

3. Koneoppimismenetelmistä

Tässä kappaleessa esitellään kahden vaihtoehdon luokitteluongelman ratkaisemiseen tähtääviä koneoppimismenetelmiä sekä niiden lopputuotteena syntyvien luokittelumallien toimivuuden arviointiin ja vertailuun liittyviä työkaluja. Lisäksi esitellään, miten koneoppimisella synnytetyn luokittelumallin käyttäytymistä voitaisiin tulkita. Tässä kappaleessa esitettävät tekniikat pohjautuvat pääosin teokseen The Elements of Statistical Learning [1].

3.1. Logistinen regressio

Logistinen regressio on ehkäpä yksi helpoimmin lähestyttävimmistä luokitteluongelmien ratkaisemiseen käytettävistä malleista. Logistisen regression perusperiaatteena on niin sanottu logit-muunnos, joka olettaa että

$$\ln\left(\frac{P(x_i)}{1-P(x_i)}\right) = b_0 + \sum_{j=1}^{n} b_j \cdot x_{i,j}\,,$$

missä $P(x_i)$ kuvaa datapisteen $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,n-1}, x_{i,n})$ todennäköisyyttä kuulua mallinnuksen kohteena olevaan luokkaan. Ratkaisemalla tästä $P(x_i)$, saadaan kaava muotoon

$$P(x_i) = \frac{e^{b_0+\sum_{j=1}^{n} b_j \cdot x_{i,j}}}{1+e^{b_0+\sum_{j=1}^{n} b_j \cdot x_{i,j}}}.$$

Logistisessa regressiossa tarkoituksena on sovittaa datapisteet tähän funktioon ratkaisemalla sopivat parametrien arvot $b_0, \dots, b_n$. Tämä voidaan tehdä esimerkiksi jyrkimmän laskun menetelmällä.

Kuten kaavoistakin huomaa, on selittävien muuttujien suhde vastemuuttujaan varsin yksinkertainen. Logistisen regression oletukset, sekä hyvässä että pahassa, kulminoituvatkin siihen, että yksittäisen selittävän muuttujan oletetaan vaikuttavan vastemuuttujaan ainoastaan yhden parametrin kautta. Tämä tarkoittaa lineaarista riippuvuutta. Erityisesti jatkuvien muuttujien osalta, edellä mainittu ominaisuus ei sovi kovin hyvin tilanteeseen, jossa selittävän muuttujan suhde vastemuuttujaan onkin epälineaarinen. Toisaalta huolellisella jatkuvien muuttujien mallinnuksella, kuten käyttämällä linkkifunktio- ja splini-muunnoksia, taikka diskretisoimalla jatkuvia muuttujia luokittelumuuttujiksi, voidaan päästä eroon epälineaarisuuden tuomista ongelmista. Toinen haaste liittyy siihen, että logistinen regressio olettaa selittävät muuttujat riippumattomiksi. Tämä voi osoittautua ongelmalliseksi.

Logistista regressiota saadaan jonkin verran monipuolistettua hyödyntämällä regularisointia, kuten L1- (Lasso) tai L2-regularisointia (Ridge). Niiden käyttö tapahtuu lisäämällä virhefunktioon sakkotermi

$$\lambda \cdot \sum_{j=1}^{n} |b_j|^r ,$$

missä λ on regularisoinnin määrää ohjaava kerroin ja r on 1 tai 2 riippuen käytetäänkö L1 vai L2-regularisointia. L1-regularisoinnin tarkoituksena on saattaa niiden selittävien muuttujien regressiokertoimet nollaksi (tai hyvin pieneksi), jotka selittävät vähiten vastemuuttujaa. Se siis vähentää datan dimensioita karsimalla selittäviä muuttujia. L2-regularisointi taas on apuväline esimerkiksi multikollineaarisuutta vastaan lisäämällä harhaa (bias), tavoitteenaan saada sovite toimimaan paremmin myös uuteen dataan, jota ei ole käytetty mallin sovittamiseen. On yleistä käyttää myös kahden edellä mainitun regularisoinnin yhdistelmää, joka kantaa kirjallisuudessa nimeä elastinen verkko (Elastic net). Tällöin sakkotermi on muotoa

$$\lambda \cdot \sum_{j=1}^{n} \left[(1-\alpha)|b_j| + \alpha {b_j}^2 \right],$$

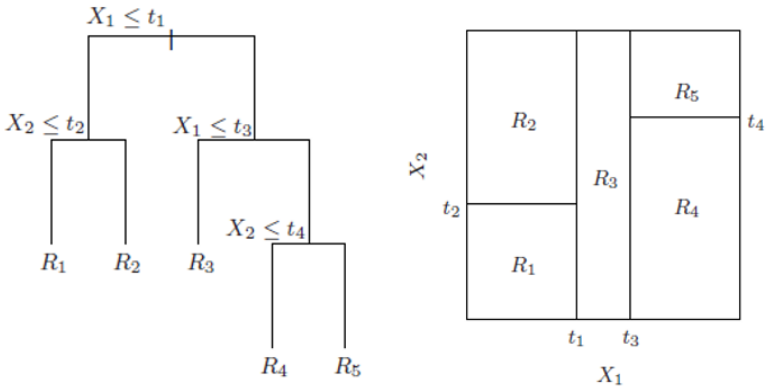
missä α määrittää painotuksen L1- ja L2-regularisoinnin välillä.

Vaikka edellä todettiin logistisen regression olevan melko yksinkertainen malli, ei se tarkoita, etteikö se olisi varsin käyttökelpoinen. Logistinen regressio toimii usein hyvänä alkumallina, jonka luokittelutarkkuutta kehittyneemmillä tekniikoilla pyritään parantamaan.

3.2. Puumallit ja random forest

Puumallit (decision trees) ovat yksi koneoppimismenetelmien alalaji. Puumallien keskeisenä periaatteena on rakentaa erilaisia sääntöjä selittäville muuttujille siten, että aineisto saadaan pilkottua palasiin, joissa vastemuuttuja toteutuu mahdollisimman tarkasti. Erityisesti kahden vaihtoehdon luokitteluongelmassa tavoitteena on saada aineisto jaettua palasiin siten, että yksittäisessä palassa esiintyy mahdollisimman vähän molempia vastemuuttujan arvoja.

Päättelypuun muodostaminen voidaan kuvata seuraavasti: Otetaan aluksi koko aineisto tarkasteluun. Muodostetaan sääntö, joka jakaa aineiston vastemuuttujan parhaalla mahdollisella tavalla. Aineisto on tässä kohtaa kahdessa palassa. Tämän jälkeen muodostetuille palasille muodostetaan seuraavat säännöt jakamaan vastemuuttujaa taas parhaalla mahdollisella tavalla. Prosessia jatketaan niin kauan, kunnes aineisto on pilkkoutunut osiin, jotka jaottelevat aineiston vastemuuttujan täydellisesti, taikka sääntöketju on saavuttanut ennalta määritetyn maksimi pituuden.



[1, s.306]. Vasemmalla on esimerkki siitä miten kahden selittävän muuttujan data voisi tulla jaotelluksi puumallilla 5 eri ryhmään. Oikealla on puumallin muodostama kaksiulotteinen vastine siitä miten selittävien muuttujien eri arvot jaottelevat datan.

Käytännön tilanteissa usein vastemuuttuja ei ole jonkin puun solmukohdan jälkeen enää jaoteltavissa käytettävissä olevilla selittävillä muuttujilla, taikka jakamisen seurauksena muodostuu osajoukkoja, jotka eivät paranna luokittelutarkkuutta juurikaan. Päätöspuun muodostamisen olennaisiin vaiheisiin kuuluukin myös jakotekijöiden karsinta. Tarkoitus tällä on karsia ne päätöspuun solmukohdat, jotka eivät kasvata luokittelun hyvyyttä riittävästi päätöspuun kompleksisuuden kasvattamisen kustannuksella.

Päättelypuut toimivat varsin hyvin aineiston jaottelussa, mutta niiden ehkäpä suurimpana ongelmana on ylisovittuminen. Toisin sanoen päättelypuu toimii hyvin käytettyyn aineistoon, mutta ei välttämättä lainkaan uuteen tuntemattomaan aineistoon. Tämän ongelman ratkaisemista voidaan lähestyä hyödyntämällä random forest -tekniikkaa.

Random forest -tekniikan lähtökohtana on rakentaa yhden päättelypuun sijaan useita päätöspuita hyödyntämällä satunnaisuutta. Prosessi etenee seuraavasti:

1. Muodostetaan satunnaisotos aineistosta (takaisinpanolla).
2. Muodostetaan päätöspuu 1. kohdan otoksella siten, että jokainen päätöspuun jakosääntö muodostetaan satunnaisesti valittujen selittävien muuttujien joukosta.
3. Toistetaan kohtia 1 ja 2 niin kauan kunnes sopiva määrä puita on saavutettu.

Edellä kuvatulla tavalla muodostetut päätöspuut antavat luonnollisesti erilaisia ennusteita. Datalle muodostetaan lopulta yksi ennuste yli kaikkien päätöspuiden. Luokitteluongelmissa tyypillisesti ennusteeksi valitaan eniten esiintyvä lopputulos yli päätöspuiden (majority vote).

Random forest -tekniikan hyödyllisyys johtuu pitkälti siitä, että se keskiarvoistaa yksittäisiä satunnaisotannalla muodostettuja ylisovittuneita puita. Tällä tavoin saadaan muodostettua pienemmän varianssin omaava edustava puu, jonka pitäisi saavuttaa parempi yleistettävyys ja sitä kautta luokittelumallin parempi toiminen myös tuntemattomaan dataan.

3.3. Gradient boosting

Gradient boosting -menetelmän ajatuksena on muodostaa useita heikon ennustekyvyn omaavia malleja, jotka yhdessä muodostavat vahvan ennustekyvyn omaavan mallin. Tämä tapahtuu hyödyntämällä pohjamallia (base learner), joka voi periaatteessa olla mielivaltaisella koneoppimismenetelmällä muodostettu, sekä aiemmin tässä työssä esiteltyä jyrkimmän laskun menetelmää.

Gradient boosting -menetelmä koostuu useista perättäin muodostetuista pohjamalleista, joiden summa muodostaa lopullisen mallin. Yksittäiset pohjamallit toimivat siis lisäyksinä aiemmin muodostettujen pohjamallien päälle. Ideana on, että jokainen lisäys pienentää mallista lasketun virhefunktion arvoa pikkuhiljaa. Jotta lisäykset olisivat optimaalisesti valittuja, lisäykset pyritään muodostamaan siten, että ne osoittavat virhefunktion negatiivisen gradientin suuntaan. Toisin sanoen lisäykset perustuvat jyrkimmän laskun menetelmään.

Olkoon käytettävä virhefunktio L ja pohjamalli T. Gradient boosting -menetelmä voidaan muotoilla matemaattisesti algoritmiksi pääpiirteittäin seuraavalla tavalla:

1. Asetetaan $f_0(x_i) = \gamma$ kaikilla datapisteillä i, missä γ minimoi virhefunktion $\sum_i L(y_i, \gamma)$.
2. Iteroidaan järjestyksessä kaikille $m = 1, 2, \dots, M-1, M$:
    a. Lasketaan virhefunktion negatiiviset gradientit $g_{mi} = -\frac{\partial L(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)}$.
    b. Sovitetaan pohjamalli T_m datapisteisiin (g_{mi}, x_i).
    c. Valitaan δ_m, joka minimoi kaavan $\sum_i L(y_i,\ f_{m-1}(x_i) + \delta_m \cdot T_m(x_i))$.
    d. Asetetaan $f_m(x_i) = f_{m-1}(x_i) + \delta_m \cdot T_m(x_i)$ kaikilla i.

Menetelmän lopputulos f_M koostuu siis pienemmistä lisäyksistä $\delta_m \cdot T_m$, jotka pohjautuvat jyrkimmän laskun menetelmän soveltamiseen. On hyvä huomata, että jyrkimmän laskun menetelmää hyödynnetään tässä yhteydessä parametriavaruuden sijasta funktioavaruudessa.

Gradient boosting -menetelmän ideologiaa voidaan vielä hieman yksinkertaistaa huomioimalla se, että algoritmin kohdassa 2a laskettavat virhefunktion gradientit ovat jollain tavalla sidoksissa sen hetkiseen poikkeamaan vastemuuttujan todellisista arvoista. Voidaan siis ajatella, että malliin tehtävät lisäykset perustuvat jäljellä oleviin residuaaleihin. Mitä enemmän askeleita otetaan, sitä pienemmäksi residuaalit lopulta menevät.

Myöhemmin tässä työssä gradient boosting -menetelmää sovellettaessa pohjamalleina käytetään puumalleja. Menetelmää tullaan soveltamaan kahdella tavalla, joista ensimmäinen perinteisempi versio kantaa tässä työssä nimeä Gradient Boosting Tree, kun taas toinen hieman kehittyneempi versio kantaa nimeä XGBoost [2]. Jälkimmäisen taustalla on pääosin Gradient Boosting Tree -menetelmän rakenne, mutta se hyödyntää lisäksi regularisointia enenevissä määrin. Yleisesti ottaen gradient boosting -menetelmä pyrkii voimakkaasti pienentämään harhaa. Regularisoinnin lisäämisen voisi täten olettaa saavuttavan paremman yleistettävyyden ja sitä kautta paremman ennustetarkkuuden myös dataan, jota ei ole käytetty mallin opettamiseen.

3.4. Luokittelumallin arviointi

Kun valitulla koneoppimismenetelmällä on muodostettu luokittelumalli, on vuorossa mallin hyvyyden tarkastelu. Tämä toteutetaan tutkimalla mallin antamia ennusteita. Keskeistä kuitenkin on, että malli opetetaan ja testataan eri datalla, jotta voidaan varmistaa mallin toimivuus myös sellaiseen dataan, jota se ei ole vielä nähnyt. Mallin opettamiseen käytettävää aineistoa kutsutaan opetusaineistoksi, ja vastaavaa testaamiseen käytettävää testiaineistoksi.

Yksinkertaisimmillaan mallin toimivuutta voidaan tarkastella tutkimalla tarkkuutta (accuracy), joka vastaa kysymykseen, kuinka monta prosenttia ennusteista meni oikein. Tarkkuuden tulkitsemisen kanssa pitää kuitenkin olla tarkkana, sillä tarkkuus tulee aina suhteuttaa testiaineiston vastemuuttujan jakaumaan. Jos esimerkiksi 2 luokkainen vastemuuttuja on tasajakautunut, on puhtaalla arvauksella 50 % todennäköisyys mennä oikein. Jos taas vastemuuttujan jakauma olisikin 90 % / 10 %, on vastaava todennäköisyys 90 %. Jos luokittelumallin antama tarkkuus on 90 %, ensimmäisessä tilanteessa voidaan puhua oikein hyvästä tarkkuudesta, mutta jälkimmäisessä tarkkuus on yhtä hyvä kuin puhdas arvaus.

Mallin ennusteiden hyvyyttä tarkastellaan tyypillisesti myös luokittelumatriisin (confusion matrix) kautta. Luokittelumatriisi kertoo ennusteiden jakautumisen oikein ja väärin ennustettuihin molemmissa vastemuuttuja vaihtoehdoissa:

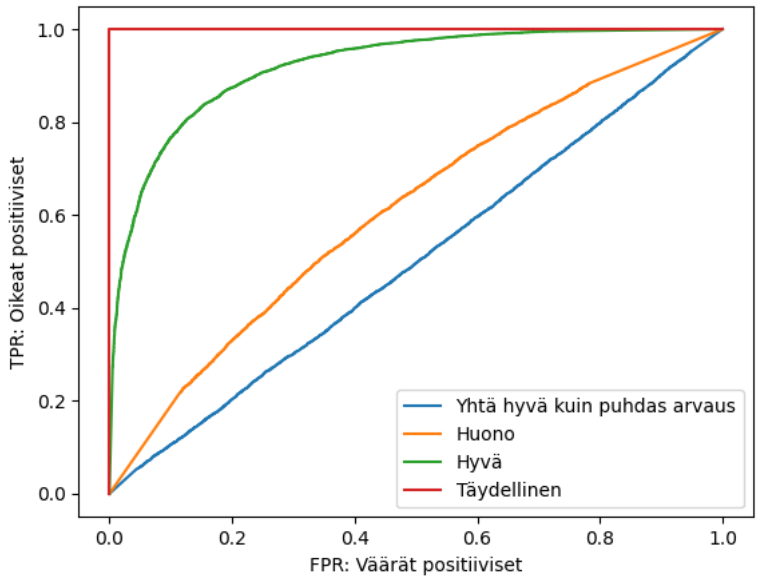
	POSITIIVINEN	NEGATIIVINEN
ENNUSTETTU POSITIIVINEN	Oikea positiivinen (TP)	Väärä positiivinen (FP)
ENNUSTETTU NEGATIIVINEN	Väärä negatiivinen (FN)	Oikea negatiivinen (TN)

Luokittelumatriisista johdetaan myös erilaisia tunnuslukuja, jotka voivat olla helpommin tulkittavia:

- TPR: Löydetyt positiiviset kaikista positiivisista
- FPR: Virheellisesti positiiviseksi luokitellut kaikista negatiivisista
- FNR: 1 - TPR
- TNR: 1 - FPR

Asetelma riippuu aina siitä, kumpi luokista ajatellaan positiiviseksi ja päinvastoin.

Kun edellä mainitut TPR ja FPR yhdistetään, voidaan muodostaa luokittelun hyvyyden tarkastelua varten sopiva ROC-käyrä (Receiver operating characteristic). ROC-käyrä kuvaa sitä kuinka iso osa vääriä arvauksia kaikista negatiivisista (FPR) joudutaan tekemään, jotta löydetään tietty osuus kaikista positiivisista (TPR). Alla olevassa kuvassa ROC-käyrät "Hyvä" ja "Huono" ovat vain havainnollistuksia. Hyvyys tulee todellisuudessa suhteuttaa aina ratkottavaan ongelmaan.



Vaikka ROC-käyrän muodosta voidaan tehdä erilaisia johtopäätöksiä, on tavallista yleistää ROC-käyrä yhdeksi luvuksi. Tämä tapahtuu laskemalla ROC-käyrän alle jäävän alueen pinta-ala, jota kutsutaan nimellä AUC (Area under curve). Tulkintana voidaan pitää, että mitä lähempänä AUC on ykköstä, sen paremmin luokittelumalli toimii. Usein vastaavana mittarina käytetään myös niin sanottua gini-indeksiä, joka kytkeytyy AUC-arvoon kaavan

$$\text{gini-indeksi} = 2 \cdot \text{AUC} - 1$$

kautta. Konkreettisena erona on se, että puhtaan arvauksen tavoin käyttäytyvä malli saa AUC-arvon 0,5 kun taas gini-indeksi saa arvon 0. Molemmilla mittareilla ykkönen kuvastaa täydellistä mallia.

ROC-käyrät ja AUC ovat siinä mielessä tarkkuutta parempia mittareita, että ne eivät riipu yhtä vahvasti aineiston vastemuuttujan jakaumasta. Toisin sanoen vastemuuttujan jakauman mahdollista vinoumaa ei tarvitse huomioida samoin kuin tarkkuutta tutkiessa.

3.5. Mallien selittäminen

Edellä tutkittiin vain sitä, miten koneoppimisella muodostetun mallin toimivuutta voidaan arvioida luokittelutarkkuuden näkökulmasta. Yleensä on myös ainakin jossain määrin kiinnostavaa mitä malli "ajattelee". Logistisen regression tapauksessa selittävien muuttujien vaikutus lopputulokseen saadaan yksinkertaisesti regressiokertoimista. Monimutkaisempien koneoppimismenetelmien tapauksessa mallien selittäminen voi olla hankalaa, jopa mahdotonta, tutkimalla vain ja ainoastaan mallin parametrejä. Mallien selittämistä varten on kuitenkin olemassa erilaisia tekniikoita. Monet näistä mittaavat selittävien muuttujien tärkeyttä vain yleisesti, ottamatta kantaa esimerkiksi vaikutuksen suuntaan. Yksi lähestymistapa selittämiseen voisi olla takaisinmallinnus (reverse engineering), jossa monimutkaisen mallin antamiin ennusteisiin sovitetaan esimerkiksi yksinkertaisempi tilastollinen malli. Tällöin monimutkaisen mallin käyttäytymistä voidaan analysoida helpommin tulkittavan tilastollisen mallin avulla. Tässä työssä esitellään niin sanottu SHAP-tekniikka, joka omaa joitakin takaisinmallinnuksen piirteitä. Esiteltävä tekniikka pohjautuu teokseen A Unified Approach to Interpreting Model Predictions [3].

SHAP (Shapley Additive Explanations) on tekniikka, jonka ajatuksena on tutkia selittävien muuttujien vaikutusta vastemuuttujaan datapisteittäin. SHAP hyödyntää niin kutsuttuja shapley-arvoja, jotka ovat tutumpia esimerkiksi taloustieteessä käytetystä peliteoriasta [4]. Ideana on muodostaa datapisteittäin jokaiselle yksittäiselle selittävälle muuttujalle j arvo φ_j, joka kertoo kuinka paljon selittävä muuttuja j kontribuoi koneoppimismallilla laskettuun ennusteeseen. Näitä kutsutaan SHAP-arvoiksi. Teoreettisesti katsoen φ_j lasketaan kaavalla

$$\sum_{S \subseteq F \setminus \{j\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \left(f_{S \cup \{j\}}(x_{S \cup \{j\}}) - f_S(x_S)\right)$$

missä F on kaikkien selittävien muuttujien joukko, j tutkittava selittävä muuttuja ja f_S koneoppimismalli kun mallin käyttämät selittävät muuttujat on rajoitettu joukkoon S. Kaavan jälkimmäinen termi $f_{S \cup \{j\}}(x_{S \cup \{j\}}) - f_S(x_S)$ tarkoittaa sitä, kuinka paljon mallilla tehtävä ennuste muuttuu, jos mallin selittävien muuttujien joukkoon S lisätään selittävä muuttuja j. Tämä lasketaan kaikille mahdollisille selittävien muuttujien osajoukoille, joiden painotettu summa muodostaa selittävän muuttujan j kontribuution määrän.

SHAP-arvojen yhteys mallilla laskettuun ennusteeseen muodostuu laskemalla SHAP-arvot yhteen. Yksittäisen datapisteen i osalta tämä tarkoittaa, että

$$f(x_i) = \varphi_0 + \sum_j \varphi_{i,j},$$

missä φ_0 on datan vastemuuttujan jakaumasta tuleva pohjaluku ja $\varphi_{i,j}$ datapisteen i selittävän muuttujan j SHAP-arvo.

Jos selittäviä muuttujia on kovin paljon, SHAP-arvojen laskeminen voi olla käytännössä lähes mahdotonta. Tästä syystä käytännön laskennassa hyödynnetään otantamenetelmään perustuvaa approksimaatiota.

4. Työkyvyttömyyseläkeratkaisun ennustaminen

Tässä kappaleessa hyödynnetään koneoppimista käytännön ongelmaan. Aluksi ongelman asettelu kuvataan tarkemmalla tasolla, sekä esitellään käytettävä aineisto. Lopuksi hyödynnetään koneoppimista ennustamaan työkyvyttömyyseläkehakemuksen ratkaisua, sekä pyritään selittämään yhden käytettävän mallin käyttäytymistä.

4.1. Ongelman asettelu

Yksittäinen henkilö sairastuu ja päätyy hakemaan pysyvää työkyvyttömyyseläkettä työeläkevakuutusyhtiöstä. Hakemusta seuraa eläkeratkaisu. Eläkeratkaisu on päätös siitä, myönnetäänkö vai hylätäänkö haettu työkyvyttömyyseläke. Työeläkevakuutusyhtiö tekee eläkeratkaisun pohjautuen hakemuksessa saataviin tietoihin, sekä arvioiden työkyvyttömyyttä suhteessa siihen mitä TyEL-laissa säädetään oikeudesta työkyvyttömyyseläkkeeseen. Ratkaistavaksi ongelmaksi muodostuu se, että pystyykö koneoppimista hyödyntäen ennustamaan edellä kuvatun pysyvän työkyvyttömyyseläkehakemuksen ratkaisua.

Koneoppimisen näkökulmasta kyse on kahden vaihtoehdon luokitteluongelmasta. Tarkoituksena onkin opettaa koneoppimismalleja löytämään mahdollisimman tarkasti ne syyt, jotka ennustavat myönteistä tai kielteistä eläkeratkaisua. Ongelmassa rajaudutaan tutkimaan vain täysiä pysyviä työkyvyttömyyseläkkeitä. Toisin sanoen osatyökyvyttömyyseläkkeet sekä määräaikaiset työkyvyttömyys- ja kuntoutusetuudet, eli kuntoutustuet ja kuntoutusrahat, rajataan ongelman ulkopuolelle.

Vertailun vuoksi, Eläketurvakeskus oli tehnyt vuonna 2017 työkyvyttömyyseläkkeiden teemaan liittyvän koneoppimiskokeilun, jossa pyrittiin ennustamaan työkyvyttömyyseläkkeen alkamista kahden vuoden sisään tarkasteluhetkestä [5]. Tämän työn ongelmanasettelu on tietyssä mielessä edellä mainitun kokeilun osaongelma, sillä tutkittavat henkilöt ovat jo päätyneet hakemaan eläkettä. Asetelma tässä työssä on kuitenkin työeläkevakuutusyhtiö lähtöisempi, kuten myöhemmin esiteltävistä sovellusalueista tullaa huomaamaan.

4.2. Aineisto

Käytettävä aineisto koostuu työeläkevakuutusyhtiö Varman datasta koskien työkyvyttömyyseläkkeenhakijoihin sekä hakemuksiin liittyviä anonymisoituja tietoja. Aineisto sisältää työkyvyttömyyseläkehakemuksia vuosilta 2010–2021. Vastemuuttujana on tieto siitä, onko eläkeratkaisu myönteinen vai kielteinen. Selittävät muuttujat sisältävät pääosin seuraavia tietoja:

- Hakijan ikä, sukupuoli ja asuinpaikka (AVI ja maakunta tasolla).
- Hakemuksen päädiagnoosi, sekä mahdollinen toinen päädiagnoosi.
- Hakijan aiemmin voimassaolleet etuudet sekä niihin liittyvät päädiagnoosi(t) viimeisen 3 vuoden ajalta
- Hakijan aiemmin hylätyt eläkehakemukset viimeisen 3 vuoden ajalta.

Muita mahdollisesti ennustetarkkuuteen vaikuttavia tietoja voisivat olla esimerkiksi henkilön työhistoriaan sekä mahdollisiin muihin sosiaalietuuksiin liittyvät tiedot. Lisäksi mainittakoon, että hakemuksen liitteenä annettava hoitavan lääkärin kirjoittama, niin sanottu B-lausunto, sisältää viime kädessä hakijan yksilöllisiä eläkeratkaisuun vaikuttavia tietoja. Näiden tietojen puuttuessa, aineisto sisältää jossain määrin selittämätöntä kohinaa.

Olipa kyse sitten koneoppimisesta tai tilastollisesta mallinnuksesta, lähtödataan on suhtauduttava kriittisesti. Lähtödatassamme aikaulottuvuus on vahvasti läsnä, sillä tutkittavia työkyvyttömyyseläkehakemuksia on usealta vuodelta. Mallin toimivuuden kannalta kompastuskiveksi voisi muodostua se, että työkyvyttömyyseläkkeen myöntökäytännöt olisivat jollain lailla aikaan sidottuja. Tällainen tilanne voisi syntyä esimerkiksi lainsäädäntöön tehdyistä muutoksista. Tässä työssä tehtävä ennustaminen koskee tilanteita, joissa henkilö on jo hakenut työkyvyttömyyseläkettä. Toisin sanoen syyt, jotka vaikuttavat eläkkeen hakemiseen, eivät vaikuta selittävien muuttujien ja eläkeratkaisun välillä. Toisaalta aineisto eri vuosina voi painottua tietynlaisiin eläkkeenhakijoihin, mutta tämä ei aiheuta ongelmaa, mikäli eläkepäätöksen myöntämiseen liittyvä lainsäädäntö ja käytännöt ovat pysyneet samoina.

Työntekijän eläkelain 35§:ssa [6] säädetään oikeudesta työkyvyttömyyseläkkeeseen sanatarkasti seuraavalla tavalla:

Työntekijällä on oikeus työkyvyttömyyseläkkeeseen, jos hänen työkykynsä arvioidaan olevan heikentynyt sairauden, vian tai vamman vuoksi vähintään kahdella viidesosalla yhtäjaksoisesti ainakin vuoden ajan. Työkyvyttömyyseläke myönnetään täytenä eläkkeenä, jos työntekijän työkyky on heikentynyt vähintään kolmella viidesosalla. Muussa tapauksessa työkyvyttömyyseläke myönnetään osatyökyvyttömyyseläkkeenä.

Työkyvyn heikentymistä arvioitaessa otetaan huomioon työntekijän jäljellä oleva kyky hankkia itselleen ansiotuloja sellaisella saatavissa olevalla työllä, jota työntekijän voidaan kohtuudella edellyttää tekevän. Tällöin otetaan huomioon myös työntekijän koulutus, aikaisempi toiminta, ikä, asuinpaikka ja muut näihin rinnastettavat seikat. Jos työkyky vaihtelee, otetaan huomioon työntekijän vuotuinen ansio.

Sen lisäksi, mitä 2 momentissa säädetään, arvioitaessa 60 vuotta täyttäneen työntekijän oikeutta työkyvyttömyyseläkkeeseen painotetaan työkyvyttömyyden ammatillista luonnetta.

Laki on tältä osin pysynyt samana sen voimaantulosta 1.1.2007 lähtien. Täyden työkyvyttömyyseläkkeen eläkeoikeuden tulkinnassa käytettävä ehto "kolme viidesosaa" on siis läsnä koko aineistoissa. Mitä toisessa momentissa säädetään, voi jollain tavalla aiheuttaa harhoja aineistoissa. Tämä johtuu siitä, että esimerkiksi ajallinen tai alueellinen työn kysynnän heilunta voi vaikuttaa työkyvyttömyyseläkeoikeuden tulkinnassa. Lisäksi esimerkiksi hakijan koulutus tai vuotuinen ansio ovat mainittuna laissa. Näiden saaminen mukaan selittävien muuttujien joukkoon voisi parantaa ennustetarkkuutta. Kolmannessa momentissa mainitaan vielä lisäksi 60 vuotta täyttäneiden hieman erilaisesta työkyvyttömyyden tulkinnasta. Lähtökohtaisesti mallimme pitäisi havaita tämä ilmiö datan kautta kohonneena myönteisen eläkeratkaisun todennäköisyytenä, vaikkakaan hakijan ammattista johtuvaa syyseurausta ei voida mallille opettaa tällaisen muuttujan puuttuessa.

4.3. Mallien soveltaminen aineistoon

Seuraavaksi hyödynnämme kappaleessa 3 esiteltyjä menetelmiä ennustamaan työkyvyttömyyseläkehakemuksen ratkaisua. Menetelmien käyttö toteutetaan python-ohjelmointikielellä hyödyntäen valmiita koneoppimistyökaluja sisältävää scikit-learn-kirjastoa [7].

Ennen mallien soveltamista aineisto jaetaan kahteen osaan siten, että mallien opettamiseen käytetään 70 % ja testaamiseen 30 % aineistosta. Tällä varmistetaan, että opetettu malli toimii myös aineistoon, jota se ei ole aiemmin nähnyt. Usein myös yhden testaamisen tarkoitetun aineiston sijasta valitaan erikseen validointiaineisto mallien väliseen vertailuun ja parametrien säätämistä varten, sekä erikseen varsinainen testiaineisto lopullisen mallin testaamista varten. Tässä työssä tyydytään tutkimaan sekä testaamaan malleja yhdellä aineistolla, jota nimitetään testiaineistoksi.

Koneoppimismallit sisältävät jonkin verran ennalta päätettäviä tekijöitä, joita kutsutaan hyperparametreiksi. Esimerkiksi aiemmin työssä mainitut oppimisaste ja puumallin maksimisyvyys ovat esimerkkejä hyperparametreista. Hyperparametrien optimointi voidaan toteuttaa esimerkiksi

raakalaskennalla taikka satunnaisuutta käyttäen. Varsinainen mallien opettaminen toteutetaan hyperparametrien optimointi huomioiden seuraavasti:

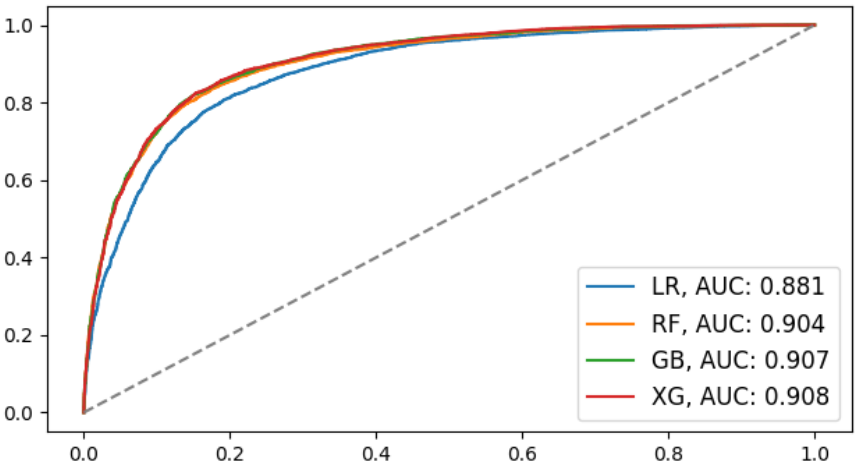
1. Muodostetaan joukko mahdollisia hyperparametreja.
2. Opetetaan koneoppimismalli kaikilla mahdollisilla hyperparametrien kombinaatioilla seuraavasti:
    a. Opetetaan malli 5 kertaa käyttäen opetusaineiston ristiinvalidointia.
    b. Lasketaan tarkkuuksien keskiarvo.
3. Valitaan hyperparametrit, jotka tuottivat suurimman ristiinvalidoidun tarkkuuden.
4. Opetetaan malli käyttäen valittuja hyperparametrejä sekä koko opetusaineistoa.

Opetus- ja testiaineistossa molemmissa hylkäyksien osuus on noin 54 % ja vastaavasti myöntöjen osuus noin 46 %. Eri malleille lasketut tarkkuudet antavat seuraavia tuloksia:

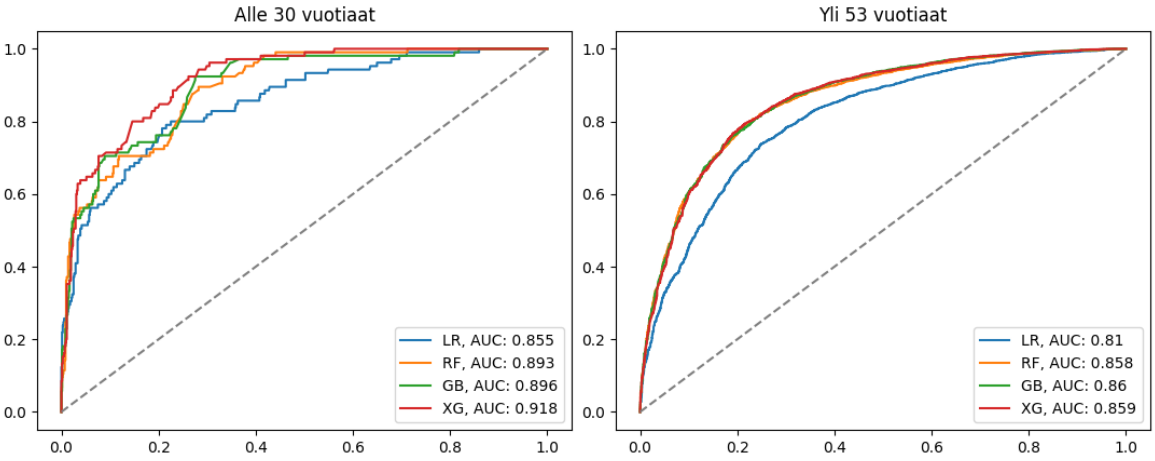
	Tarkkuus: opetusaineisto	**Tarkkuus: testiaineisto**
Logistinen regressio (+ elastinen verkko)	80,9 %	80,6 %
Random Forest	86,3 %	82,9 %
Gradient Boosting Tree	85,3 %	83,5 %
XGBoost	84,6 %	83,6 %

Voidaan havaita, että kaikki mallit pärjäsivät selkeästi puhdasta arvausta paremmin. Kiinnostuksen kohteena ovat testiaineistosta lasketut tarkkuudet, sillä ne kuvaavat mallin toimivuutta datalla, jota ei ole käytetty mallin opettamiseen. Testiaineistosta lasketut tarkkuudet eivät näyttäisi eroavan kovinkaan paljoa eri mallien välillä, sillä parhaimman ja huonoimman ennustetarkkuuden ero on vain 3 prosenttiyksikköä. Kuitenkin gradient boosting -menetelmää hyödyntävät mallit pärjäsivät tarkkuus mielessä parhaiten. Toisaalta XGBoost -malli ei tässä tapauksessa tuonut juurikaan lisää tarkkuutta suhteessa Gradient Boosting Tree -malliin. Kaikilla malleilla opetusaineistosta laskettu tarkkuus on suurempi kuin testiaineistosta laskettu. Voidaan siis puhua hienoisesta ylisovittumisesta. Tässä mielessä Random Forest -malli ylisovittui eniten.

Arvioitaessa malleja ROC-käyrien ja AUC-arvojen avulla, havaitaan tarkkuuden tavoin logistisen regression toimivan muita malleja hieman heikommin. Muiden mallien toimivuudesta ei voida silmämääräisesti havaita eroja. Random Forest -mallin AUC-arvo jää vain hieman alle sen mihin Gradient Boosting Tree ja XGBoost yltävät.



Aineistossa ikäriippuvuus on kuitenkin vahvasti läsnä, joten ROC-käyriä tutkitaan vielä tarkemmin ikäluokkien ääripäille. Kuvassa alle 30-vuotiaiden ROC-käyrän sahalaitaisuus johtuu siitä, että alle 30-vuotiaiden joukossa myönteisten eläkeratkaisujen määrä on huomattavasti hylkäyksiä pienempi.



Näyttäisi siltä, että muut mallit toimivat huomattavasti logistista regressiota paremmin nuoremmilla sekä vanhemmilla henkilöillä. Taustalla voi olla se, että syyt hylkäyksien ja myöntöjen välillä ovat jokseenkin erilaiset eri ikäisillä, jolloin logistinen regressio ei lineaarisuutensa ja riippumattomuusoletuksensa takia kykene havaitsemaan niitä. Lisäksi havaitaan, että nuoremmilla henkilöillä XGBoost -malli näyttäytyy edukseen suhteessa kaikkiin muihin malleihin. Tämä selittänee sen parhainta ennustekykyä myös tarkkuustarkastelussa.

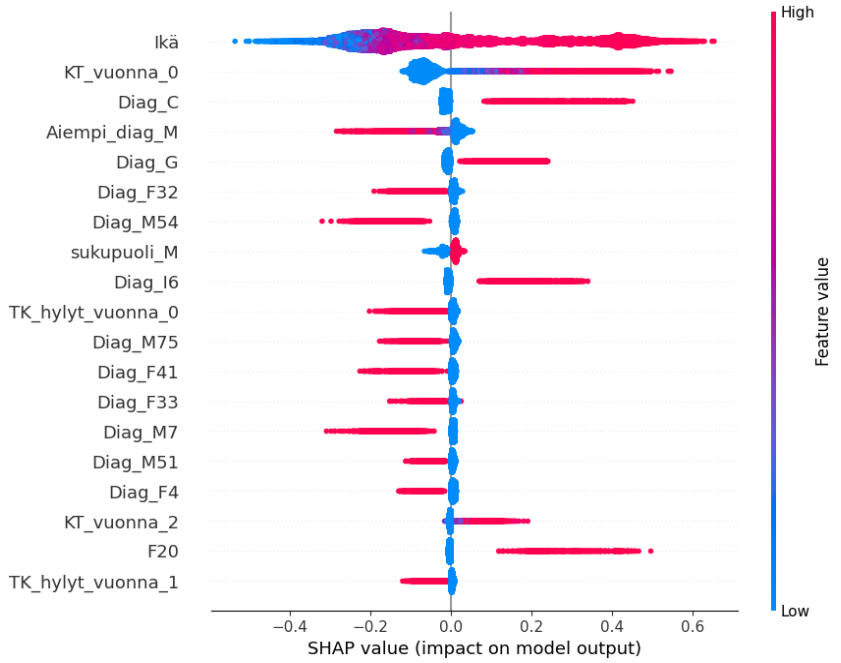
Mallien toimivuudessa näkyy se ero, että mitä monimutkaisemmasta mallista on kysymys, sen paremmin se onnistuu luokittelussa. Erot eivät kuitenkaan ole kovin isoja. On myös mahdollista, että huolellisella selittävien muuttujien mallinnuksella myös logistinen regressio olisi saavuttanut monimutkaisempia malleja vastaavan tarkkuuden. Kuten kuitenkin aiemminkin todettiin, monimutkaisempien mallien käyttäytymistä on hankalampi selittää. Täten onkin toivottua, että monimutkaisuus kompensoituu parempana ennustekykynä. Valinta monimutkaisuuden ja ennustekyvyn välillä on kuitenkin tapauskohtaista. Ei ole olemassa selkeää vastausta kuinka suuri

ennustekyvyn lisäys on riittävä kompensoimaan monimutkaisuuden lisäyksen. Tässä työssä ei kuitenkaan paneuduta tähän sen syvällisemmin, vaan todetaan XGBoost-mallin tarjoavan parhaan ennustekyvyn.

4.4. Havaintoja mallin käyttäytymisestä

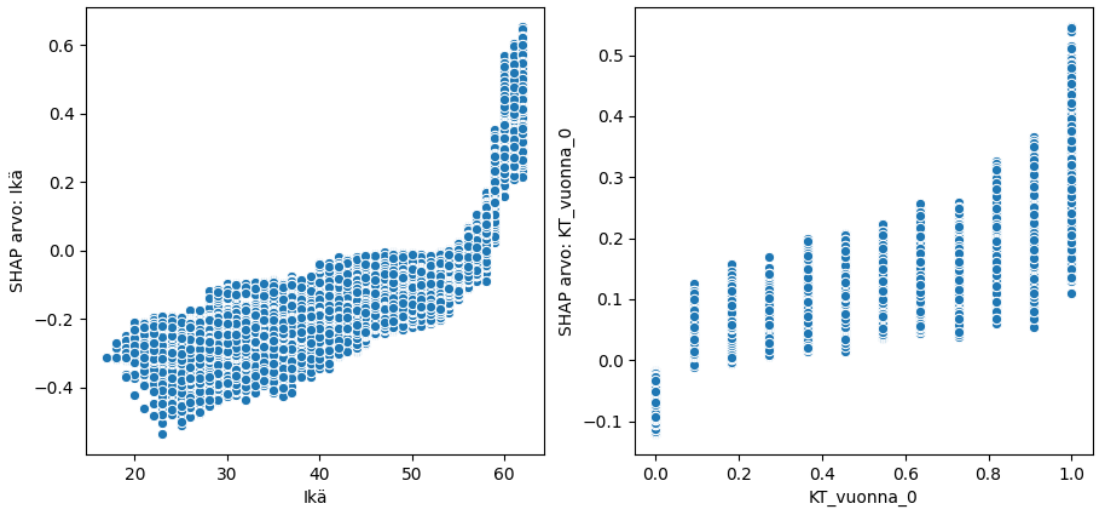
Tässä osiossa yritetään tulkita joitakin keskeisimpiä tekijöitä, jotka vaikuttavat mallin antamiin ennusteisiin. Tämä toteutetaan tutkimalla XGBoost-mallin ennusteita hyödyntäen kappaleessa 3 esiteltyä SHAP-tekniikkaa. Käytännön toteutuksessa hyödynnetään vastaavaa SHAP-nimistä python-kirjastoa. On syytä huomata, että seuraavat havainnot ovat vain tulkintoja siitä, miten käytettävä malli muodostaa ennusteen eläkeratkaisusta. Toisin sanoen ne eivät välttämättä päde yleisesti. Toisaalta mitä parempi mallin ennustetarkkuus on, sitä luultavammin mallista tehtävät havainnot ovat myös oikeita reaalimaailman ilmiöitä.

Seuraavassa kuvassa on keskeisimpiä mallin ennusteita ohjaavia muuttujia SHAP-arvojen perusteella. Erityisesti ikä näyttää ohjaavan mallin ennusteita erittäin paljon. Pääsääntöisesti näyttää siltä, että matalampi ikä pienentää myönteisen eläkeratkaisun todennäköisyyttä, kun taas korkea ikä nostaa sitä. Lisäksi kuntoutustuen voimassaoloaika viimeisen vuoden aikana ennen hakemusta (KT_vuonna_0) näyttäisi vaikuttavan voimakkaasti. Jos kuntoutustukea ei ole ollut, pienentää se myönteisen eläkepäätöksen todennäköisyyttä pääsääntöisesti hieman. Jos kuntoutustuki on ollut voimassa, sen vaikutus on myönteisen eläkeratkaisun todennäköisyyttä tapauskohtaisesti joko maltillisesti tai erittäin voimakkaasti korottava. Lisäksi kuvassa on joukko erilaisia hakijan työkyvyttömyyteen liittyviä diagnooseja, joista myönteistä eläkeratkaisua näyttäisivät puoltavan C (pahanlaatuiset kasvaimet), G (hermoston sairaudet), F20 (skitsofrenia) ja I6 (aivoverisuonien sairaudet) ryhmien diagnoosit, missä suluissa kuvatut selitteet ovat ICD-10 tautiluokituksen [8] mukaisia.



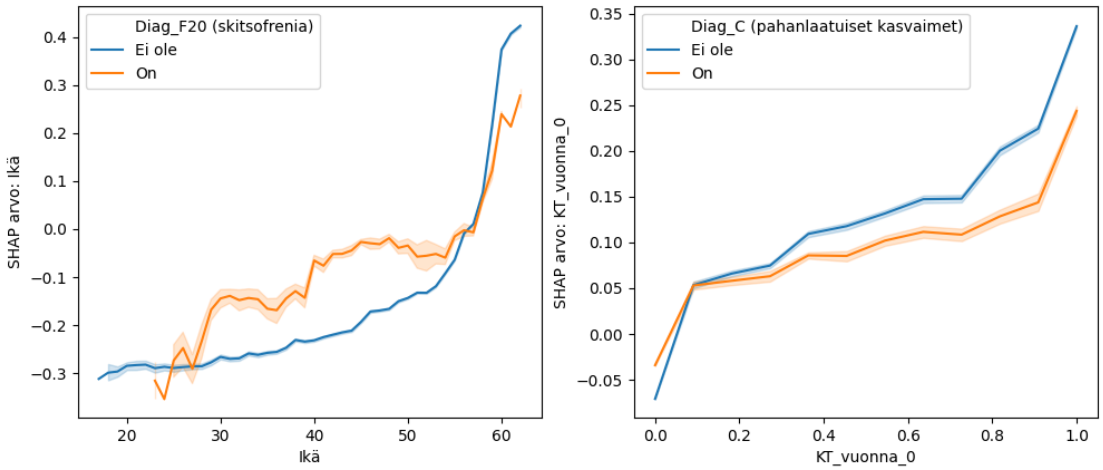
SHAP arvo kuvaa kuinka paljon yksittäiset selittävät muuttujat selittävät luokittelumallilla laskettua myönteisen työkyvyttömyyseläkepäätöksen todennäköisyyttä, kun yksittäiselle datapisteelle ennustettu todennäköisyys muodostuu pohjatodennäköisyydestä 0,45 sekä datapiste- ja selittävämuuttujakohtaisesti lasketusta SHAP arvosta.

Silmämääräisesti tärkeimmät muuttujat mallin kannalta ovat siis henkilön ikä ja kuntoutustuen voimassaoloaika vuoden sisään ennen työkyvyttömyyseläkehakemusta. Seuraavassa tutkitaan tarkemmin, kuinka nämä vaikuttavat mallin tuottamiin ennusteisiin.



Kuvasta voidaan nähdä, että vaikka korkeampi ikä lisää myönnön todennäköisyyttä (korkeampi SHAP-arvo), vaikutus ei ole lineaarinen. Lisäksi yksittäisissä ikäluokissa hajonnatkin ovat verrattain isoja, jonka voi tulkita siten, että malli havaitsee ikäluokkien sisältävän hyvinkin erilaisia henkilöitä. Kaikkein iäkkäimpien henkilöiden kohdalla (60–62-vuotiaat) iän vaikutus kasvaa merkittävästi. Malli onkin mahdollisesti datan kautta oppinut aiemmin mainitun Työntekijän eläkelaista tulevan määritelmän 60-täyttäneiden työkyvyttömyyseläkeoikeuden tulkinnalle. Hakemusta edeltävän kuntoutustuen osalta

vaikutus näyttää oleva lähes poikkeuksetta myönteisen eläkeratkaisun todennäköisyyttä nostava. On kuitenkin hyvä huomata hajonnan olevan voimakasta.



Vasemmalla ikäluokittain lasketut keskimääräiset SHAP arvot, kun hakijalla ei ole tai on skitsofreniaan viittaava diagnoosi. Oikealla keskimääräiset SHAP arvot laskettuna hakemusta vuoden sisään edeltäneen kuntoutustuen kestoittain (kesto skaalattuna välille 0–1) kun hakijalla ei ole tai on pahanlaatuiseen kasvaimeen viittaava diagnoosi. Kuvat sisältävät keskiarvon 95 % luottamusvälin.

Kun sekä iän että kuntoutustuen vaikutusta mallin tuottamaan ennusteeseen verrataan kuvassa olevien hakijan diagnoosien tapauksessa, saadaan mielenkiintoisia lisähavaintoja aikaiseksi. Iän osalta näyttää siltä, että vaikka alhainen ikä keskimäärin laskee myönnön todennäköisyyttä, diagnoosiryhmä F20 muuttaa iän vaikutusta olennaisesti. Erityisesti 30–50-vuotiailla henkilöillä iän vaikutus ennusteeseen lähenee kohti nollaa, kun hakijalla on todettu F20 diagnoosiryhmään lukeutuva diagnoosi. Aiemman kuntoutustuen vaikutus mallin ennusteisiin oli pääosin myönnön todennäköisyyttä kasvattava. Kuitenkin tilanteissa, joissa on kysymyksessä C diagnoosiryhmä, kuntoutustuen merkitys on olennaisesti pienempi. Toisaalta mitä lyhyemmästä kuntoutustuesta on kysymys, sen pienempi on vaikutuksen muutos. Molemmissa edellä mainituissa tilanteissa syy on mahdollisesti se, että malli tulkitsee sekä F20 että C diagnoosiryhmät erittäin vahvasti myönteistä eläkeratkaisua ennustavaksi, jolloin muiden muuttujien merkitys ennusteen muodostamiseen pienenevät.

5. Yhteenveto ja pohdintoja ennusteen sovellusalueista

Tässä työssä esiteltiin erilaisia luokitteluongelmien ratkaisemiseen tähtääviä koneoppimismenetelmiä. Esiteltyjä menetelmiä lopulta sovellettiin ennustamaan TyEL:in mukaisen pysyvän työkyvyttömyyseläkehakemuksen ratkaisua. Valitulla selittävien muuttujien joukolla, parhaaksi malliksi osoittautui gradient boosting -menetelmää hyödyntävä XGBoost-malli. Mallilla saatu ennustetarkkuus oli lopulta noin 84 %, testiaineiston vastemuuttujan jakauman ollessa lähes

tasajakautunut. Tarkkuuksia tarkasteltaessa huomattiin toisaalta myös, ettei huomattavasti yksinkertaisempi logistinen regressio jäänyt kovinkaan paljon monimukaisemmista malleista jälkeen sen tarkkuuden saavuttaessa noin 81 %. ROC-käyriä tutkittaessa, voitiin havaita XGBoost-mallin parempaa tarkkuutta selittävän ikäluokkien ääripäät, joissa kyseinen malli toimi verrokkimallejaan paremmin.

Työssä esiteltiin myös koneoppimismallien käyttäytymisen tutkintaan tähtäävä SHAP-tekniikka, jota käytettiin selittämään XGBoost-mallin käyttäytymistä. Osoittautui, että eläkkeenhakijan ikä ohjaa vahvasti mallin tekemiä ennusteita. Tämän lisäksi havaittiin, että malli oli oppinut tulkitsemaan eri diagnooseja eri tavoin. Toisin sanoen mallin mielestä jotkin diagnoosit nostavat myönteisen eläkepäätöksen todennäköisyyttä voimakkaasti, kun päinvastoin toiset laskevat sitä. Todennäköisesti syvällisempi tutkiminen olisi voinut paljastaa monimutkaisiakin syy-yhteyksiä selittämään mallin käyttäytymistä.

Kaiken kaikkiaan voidaan todeta koneoppimiskokeilun olleen varsin tuloksellinen. Kysymys kuuluukin mitä hyötyä tällaisesta eläkeratkaisua suhteellisen tarkasti ennustavasta mallista voisi olla. Seuraavaksi nostetaan esiin joitakin ajatuksia ja ideoita siitä miten tällaista mallia voisi hyödyntää aktuaarillisesta näkökulmasta sekä työeläkevakuutusyhtiön toimintaan. Vaikka koneoppimiskokeilussa tyydyttiin paneutumaan pysyviin ja täysiin työkyvyttömyyseläkkeisiin, oletetaan seuraavissa pohdinnoissa mallin toimivan menestyksekkäästi mihin tahansa työkyvyttömyyseläkkeisiin liittyviin hakemuksiin.

5.1. Eläkekäsittelytoiminnan resursointi

Työeläkevakuutusyhtiön täytyy järjestää riittävät resurssit eläkkeiden ratkaisemista, sekä myönteisen eläkeratkaisun tapauksessa eläkkeen laskentaa ja maksuunpanoa varten. Työkyvyttömyyseläkkeiden tapauksessa myönteisen ja kielteisen eläkeratkaisun tekeminen voi henkilötyömäärän näkökulmasta olla kovin erilainen. Lisäksi myönteisestä eläkepäätöksestä seuraa eläkelajeittain erilaisia jatkotoimenpiteitä.

Oletetaan, että työeläkevakuutusyhtiö tuntee erilaisten hakemuksien vaatiman henkilötyömäärän eläkelajeittain sekä -ratkaisuittain. Ennustamalla jonkin hetken hakemuskannan myönteiset ja kielteiset eläkeratkaisut, eläkeyhtiöllä on mahdollisuus arvioida sen hetkisen hakemuskannan sisältävää henkilötyömäärän tarvetta. Tämän avulla pystytään hallitsemaan esimerkiksi työeläkevakuutusyhtiön käsittelyaikoja kohdistamalla käytettävissä olevat resurssit oikea-aikaisesti eläkekäsittelytoiminnan tehostamiseen.

Luokitteluongelmien taustalla on koneoppimismallin arvioima todennäköisyys kuulua mahdollisiin luokkiin. Parhaimmillaan mallin antama todennäköisyys voi kertoa jotain päätöksen teon haastavuudesta myös tilanteessa, jossa eläkeratkaisun tekee ihminen. Esimerkiksi jos malli ennustaa

vain hieman yli tai alle 50 % todennäköisyyttä myönteiselle eläkeratkaisulle, voi tämä kertoa, että eläkeratkaisun tekeminen ei ole täysin suoraviivainen prosessi ihmisellekään. Voisi siis olla mahdollista, että todennäköisyyden sisältäessä epävarmuutta, myös ratkaisun tekemiseen vaadittavat henkilötyömäärä on suurempi. Tämän osoittaminen vaatisi kuitenkin jatkotutkimusta.

5.2. Työkyvyn riippumaton arviointi

Koneoppimismallin opettamiseen käytetään suuri määrä dataa, joka pohjautuu historiaan. Eläkeratkaisua ennustava koneoppimismalli opetetaan siis aineistolla, joka koostuu toteutuneista käytännöistä päättää työkyvyttömyyseläkeoikeudesta. Mallin opettamiseen käytetty aineisto eläkehakemuksista ja niiden ratkaisuista voisi ajatella olevan ennakkotapauksia siitä, minkälaiseen eläkeratkaisuun tietynlaiset tiedot ovat lopulta johtaneet. Kun opetetulla koneoppimismallilla ennustetaan eläkehakemuksen ratkaisua, on viime kädessä kyse aiemmin toteutuneiden ratkaisukäytäntöjen toistamisesta. Koneoppimismallilla voisikin siis tuoda näkyväksi minkälainen eläkeratkaisu tietynlaisista lähtötiedoista tyypillisesti seuraa.

Oikeus työkyvyttömyyseläkkeeseen perustuu viime kädessä työeläkevakuutusyhtiön lääkärin, siis ihmisen, ratkaisuehdotukseen. Koneoppimismalli voi kuitenkin parhaimmillaan auttaa lääkäriä laadukkaampaan päätöksen tekoon, sillä se tarjoaa vaivatta näkökulman siitä, millaiseen eläkeratkaisuun tietyillä lähtötiedoilla on tyypillisesti päädytty. Mainittakoon kuitenkin, että koneälyn ei tule missään tilanteessa tehdä päätöstä eläkeoikeudesta, vaan sen tehtävä on toimia apuvälineenä.

5.3. Hakemuskannan riskisyyden mittaaminen

Työntekijän eläkelain mukaista työkyvyttömyysliikettä harjoittava työeläkevakuutusyhtiö rahoittaa tulevaisuudessa alkavista työkyvyttömyysetuuksista syntyvät menot vakuutusmaksuun sisältyvällä työkyvyttömyyseläkeosalla. Työkyvyttömyyseläkeosan taustalla on ikäluokkakohtainen työkyvyttömyystariffi, joka mitoitetaan ikäluokkakohtaiset työkyvyttömyysalkavuudet huomioiden.

Työkyvyttömyystariffin mitoittamista varten joudutaan vastaamaan kysymykseen siitä, kuinka paljon uusista työkyvyttömyyseläkkeistä syntyy menoa tulevina vuosina. Tähän voisi ajatella vaikuttavan seuraavat asiat:

- Kuinka paljon uusia työkyvyttömyyseläkkeitä haetaan tulevaisuudessa?
- Kuinka suuri osa haetuista eläkkeistä myönnetään ja hylätään?
- Minkä suuruisia myönnettävät eläkkeet ovat, sekä mikä on niiden pääoma-arvo?

Edellä mainituista tehtävällä estimaatilla pystytään todennäköisesti arvioimaan tulevaisuudessa alkavista työkyvyttömyysetuuksista syntyviä menoja varsin hyvin, kun oletetaan ettei

eläkehakemusten luonne eikä hakemusmäärien trendissä tapahdu äkillisiä muutoksia. Ennustettavuus kuitenkin kärsii, jos jokin ulkoinen tekijä alkaa vaikuttamaan yllättäen johonkin edellä mainituista tekijöistä. Tässä työssä esitelty työkyvyttömyyseläkehakemuksen ratkaisun ennustaminen voisi tarjota potentiaalisen lähestymistavan edellä mainituista tekijöistä kahteen jälkimmäiseen.

Olkoon työeläkevakuutusyhtiössä vireillä työkyvyttömyyseläkehakemus i. Eläkehakemuksesta i syntyvä odotusarvoinen meno on siten

$$E_i * P(\text{"hakemus i on myönteinen"}),$$

missä E_i on eläkkeen pääoma-arvo hakemuksen i ratkaisun ollessa myönteinen. Tällöin koko vireillä olevan hakemuskannan meno suhteessa mahdolliseen menoon on

$$\frac{\sum E_i * P(\text{"hakemus i on myönteinen"})}{\sum E_i}.$$

Tällä tavoin laskettua suuretta voidaan pitää mittarina sille, kuinka riskinen hakemuskanta on suhteessa mahdolliseen eläkemenoon. Mittarista saadaan hieman toisenlainen, kun E_i korvataan euromääräisen pääoma-arvon sijasta pääoma-arvokertoimella. Tällöin mittari ei ole niin herkkä esimerkiksi joidenkin poikkeuksellisten suurien eläkkeiden ilmestyessä vireillä olevaan hakemuskantaan. Kaikkein yksinkertaisin tapa saadaan, kun luovutaan pääoma-arvosta kokonaisuudessaan, jolloin mittari ottaa kantaa hakemuskannan hylkäys/myöntö prosenttiin.

Edellä mainitut mittarit voivatkin toimia hakemuskannan rakenteen ja sitä kautta syntyvien eläkemenojen lyhyen aikavälin ennakoivina mittareina. Ennakoinnin näkökulmasta ei kuitenkaan ole mielekästä tutkia hakemuskantaa vireillä olohetken mukaan, vaan hakemusten saapumishetken perusteella, jolloin riskitaso reagoi nopeammin muuttuneeseen hakemuskantaan. Tällöin se on myös riippumaton käsittelyajoista. Vuonna 2021 yksityisen sektorin työkyvyttömyyseläkkeen kokonaiskäsittelyaika oli keskimäärin 40 päivää [9]. Toisin sanoen ennustamalla eläkeratkaisuja, saapuvien hakemusten riskisyys paljastuu keskimäärin reilun kuukauden etuajassa.

Yksittäisen eläkevakuutusyhtiön aktuaarin arvioidessa esimerkiksi kuluvan vuoden eläkemenoja, voi tällainen mittari kyseenalaistaa trendistä poikkeavuuden ennakoivasti, ja siten antaa aktuaarille lisätietoa arvioinnin tueksi. Tämä johtaa lopulta myös työkyvyttömyystariffin mitoittamisen näkökulmaan, sillä lähivuosina alkavista työkyvyttömyysetuuksista johtuvat menot voivat korreloida jossain määrin trendissä tapahtuneiden muutosten kanssa. Todettakoon kuitenkin, että tällä hetkellä tapahtuvat muutokset hakemuskannanriskisyyden mittareissa eivät välttämättä päde pitkälle tulevaisuuteen. Ennustettavuus pidemmälle aikavälille tällaista nopeasti reagoivaa ennakoivaa mittaria hyödyntäen vaatisi lisätutkimusta aiheen ympärillä.

6. Lähteet

[1] Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.

[2] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

[3] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in neural information processing systems, 30.

[4] Shapley Ll, S. (1953). A value for n-person games. Contributions to the Theory of Games II, Annals of Mathematical Studies, 28.

[5] Jarno Varis, Eläketurvakeskus. (2018). Eläketurvakeskuksen koneoppimiskokeilu. Viitattu 14.2.2022. https://www.etk.fi/blogit/elaketurvakeskuksen-koneoppimiskokeilu-nain-se-tehtiin/.

[6] L 19.5.2006/395. Työntekijän eläkelaki, Valtion säädöstietopankki Finlex, Ajantasainen lainsäädäntö. Viitattu 14.2.2022.

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.

[8] Terveyden ja hyvinvoinnin laitos. (2011). Tautiluokitus ICD-10. https://urn.fi/URN:NBN:fi-fe201205085423.

[9] Eläketurvakeskus. Työeläkehakemukset. Viitattu 14.2.2022. https://www.etk.fi/tutkimus-tilastot-ja-ennusteet/tilastot/tyoelakehakemukset/.