521160P Johdatus Tekoälyyn

Harjoitus #2

Regressio

Kevät 2018

Regressio on tilastollinen menetelmä, joka arvioi riippuvuutta tulomuuttujan X ja lähtömuuttujan Y välillä. Yleisesti ottaen yksinkertaisin regressio-ongelma käsittelee yhtä riippuvaa muuttujaa Y, joka riippuu ainoastaan riippumattomasta muuttujasta X. Tehtävänä on löytää tälle tilastolliselle ongelmalla käyrä/malli, joka sopii parhaiten annettuun dataan arvioiden X:n ja Y:n välistä riippuvuutta. Regressio-analyysissa peruskysymys on: Mitä matemaattista mallia tulisi käyttää ongelman ratkaisussa (suora, paraabeli, logaritmi-funktio jne.)? Toinen haaste on, että kuinka sovitamme parhaiten sopivan mallin (best-fitted model) kuvaajaan? Näitä asioita pohdimme seuraavaksi lineaariselle regressiolle ja polynomiselle regressiolle.

Lineaarinen regressio

Yksinkertaisessa lineaarisessa regressiossa ennustetut Y:n arvot piirretään X:n funktiona, josta muodostuu malliksi suora. Matemaattisesti suoran yhtälö kuvataan seuraavalla tavalla:

$$y = kx + b$$

,missä k on kulmakerroin ja b on y-akselin leikkauskohta

Jos data sallii meidän piirtää suoran viivan kaikkien datan pisteiden läpi, meillä ei tule olemaan ongelmaa löytää parhaiten sopivaa mallia. Valitettavasti oikeassa elämässä esimerkiksi saman ikäisillä ihmisillä ei ole tapana olla sama pituus tai paino. Tästä syystä jokaista yksittäistä datapistettä Y ei voida ennustaa täydellisesti X:n arvoista.

Ylivoimaisesti yksinkertaisin ja nopein menetelmä sovittaa suora viiva kuvaajaan on valita viiva silmämääräisesti. Vaikka kyseinen menetelmä antaa meille ymmärrettävän kuvan ongelmasta, saatu ratkaisu on tilastollisesti merkityksetön. Suoran sovittaminen datapisteisiin on pitkään pohdittu ongelma regressio-analyysissa ja lineaarisessa regressiossa ongelman ratkaisemiseksi sopivat parhaiten pienimmän neliösumman menetelmä (least squares method) ja varianssin minimointi menetelmä (minimum-variance method).

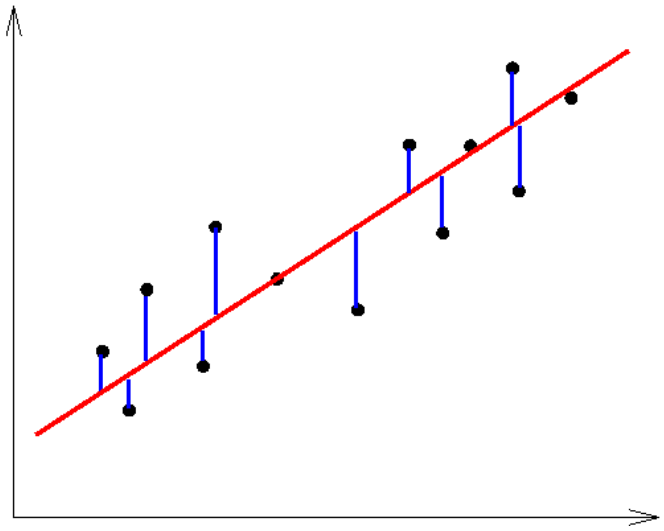
Pienimmän neliösumman menetelmä käyttää L2 normalisointia, joka yrittää löytää parhaiten sopivan suoran minimoimalla pystysuuntaisten viiva segmenttien neliöiden summan. Toinen yleisesti käytetty normalisointi-menetelmä on L1 normalisointi, joka puolestaan yrittää minimoida pystysuuntaisten viiva segmenttien itseisarvojen summan. Kaksiulotteisessa tapauksessa L1 normalisoinnille ja L2 normalisoinnille voidaan esittää yhtälöt:

$$L1_{norm} = \sum_{i=1}^{n} |Y_i - f(X_i)|$$

$$L2_{norm} = \sum_{i=1}^{n} (Y_i - f(X_i))^2$$

,missä Y_i viittaa kohdearvoihin ja $f(X_i)$ viittaa sovitetun suoran yhtälön avulla ennustettuihin y:n arvoihin

L2 normalisointi laskee neliöidyn virheen, kun taas L1 normalisointi laskee itseisarvoistetun virheen. Mitä pienempi havaittujen näytteiden etäisyyksien neliöiden summa (tai itseisarvojen summa) on, sitä lähempänä parhaiten sovitettu suora on datapisteitä. Kuvassa 1 on esitetty pienimmän neliösumman menetelmän toimintaperiaate. Mustat pisteet esitettävät datapisteitä, siniset viivat esittävät pystysuuntaisia segmenttejä eli virhevektoreita ja punainen käyrä kuvaa parhaiten sovitettua suoraa. Pienimmän neliösumman menetelmä minimoi näytteiden pystysuuntaiset etäisyydet sovitetusta suorasta.



Kuva 1. Pienimmän neliösumman menetelmän toimintaperiaate

Pienimmän neliösumman menetelmä ja varianssin minimointi menetelmä johtavat täsmälleen samaan lopputulokseen, kun sovitettavana on lineaarinen suora. Varianssin minimointi menetelmä muistuttaa enemmän klassista tilastollista menetelmää kuin pienimmän neliösumman menetelmä, ja sen avulla pystymme määrittämään parhaiten sovitetun suoran otoskeskiarvojen avulla. Kulmakerroin k ja y-akselin leikkauskohta b voidaan laskea seuraavien yhtälöiden avulla

$$\hat{k} = \frac{SS(xy)}{SS(xx)} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$$\hat{b} = \bar{Y} - \hat{k}\bar{X}$$

,missä SS on "sum of squares" (neliöiden summa), $\bar{X}$ viittaa otoskeskiarvoon X:n arvoille ja $\bar{Y}$ viittaa otoskeskiarvoon Y:n arvoille.

Suoran sovittamisen suorituskyvyn mittaamiseen käytetään usein korrelaatiokerrointa r (tai korrelaatiokertoimen neliötä r^2) tilastollisena työkaluna, joka kertoo kuinka hyvin datapisteet ja malli riippuvat toisistaan. Kun korrelaatiokerroin on 1, niin muuttujien välillä on täydellinen lineaarinen riippuvuus. Kun korrelaatiokerroin on -1, niin muuttujien välillä on täydellinen negatiivinen lineaarinen riippuvuus. Kun korrelaatiokerroin on 0, niin muuttujat eivät ole riippuvaisia toisistaan. Korrelaatiokerroin voidaan laskea seuraavien yhtälöiden avulla:

$$r = \frac{SS(xy)}{\sqrt{SS(xx)SS(yy)}} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2 \sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

tai

$$r = \frac{SS(xx)}{SS(yy)}\hat{k} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}\hat{k}$$

Tehdessäsi harjoitukset omalla tietokoneella, voit asentaa numpy, scikit-learn ja matplotlib kirjastot tietokoneellesi seuraavien linkkien avulla.

Numpy: https://docs.scipy.org/doc/numpy-1.14.0/user/install.html

scikit-learn: http://scikit-learn.org/stable/install.html

matplotlib: https://matplotlib.org/users/installing.html

Tehtävä 1

Sinun tehtäväsi on luoda lineaarinen regressio-malli yhdelle seuraavista dataseteistä:

- **data_ects_accumulation.txt**: Kuvitteellinen datasetti 50 opiskelijan opintopistekertymistä. Data kuvaa kuinka paljon opintopisteitä on kertynyt joukolle opiskelijoita eri ajan hetkillä heidän opintojensa aikana. Vaaka-akselilla on vuodet ja pystyakselilla opintopistekertymä.
- **data_life_expectancy_finland.txt**: Eliniänodote Suomessa vuosina 1960-2015. Vaaka-akselilla on vuosiluku ja pystyakselilla eliniänodote vuosina.
- **data_population_growth_finland.txt**: Väkiluku Suomessa vuosina 1960-2016. Vaaka-akselilla on vuodet ja pystyakselilla väkiluku.
- **data_sea_level.txt**: Vedenpinnantason kasvaminen millimetreissä Venetsiassa vuosina 1909-2000. Vaaka-akselilla on vuodet ja pystyakselilla vedenpinnantaso millimetreissä.
- Voit myös vaihtoehtoisesti käyttää jotain omaa lineaarista riippuvuutta noudattavaa dataa kahdelle eri muuttujalle.

Datasetit on annettu tekstitiedostoina, missä x-akselin ja y-akselin pisteet ovat omilla riveillään eroteltuna toisistaan pilkulla. Muokkaa tiedostoa nimeltä **linearregression.py**.

Koneoppimisessa testisettiä käytetään opetetun mallin suorituskyvyn arvioimiseen ja opetus-settiä käytetään mallin opettamiseen. Tässä tapauksessa opettaminen tarkoittaa sopivien suoran yhtälön parametrien löytämistä opetussetin näytteiden perusteella. Aluksi data jaetaan sattumanvaraisesti opetus-setiksi ja testisetiksi. Kyseinen jakaminen on helppo suorittaa sklearn-kirjaston **train_test_split()** funktiolla. Funktio ottaa parametreina x-akselin pisteet, y-akselin pisteet sekä testisetin absoluuttisen koon (10 prosenttia testisetin kooksi on riittävä).

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1)
```

Laskeaksesi lineaarisen regression mallin kertoimet ($\hat{k}$ ja $\hat{b}$), käytä numpy-kirjaston funktiota **np.polyfit()**. Funktio ottaa parametreina opetus-setin X:n arvoille, opetus-setin Y:n arvoille ja monennenko asteen polynominen funktio on kyseessä (lineaarinen malli käyttää ensimmäisen asteen funktiota). Kun oikeat kertoimet on saatu mallille, voidaan muodostaa funktio numpy-kirjaston funktiolla **np.poly1d()**. Alla on esitetty esimerkki koodirivit, miten lineaarisen regression funktio luodaan. Siinä X_train viittaa opetus-setin

X:n arvoihin, Y_train viittaa opetus-setin Y:n arvoihin ja numero 30 viittaa 30:n asteen polynomiseen funktioon.

```
coeffs = np.polyfit(X_train, Y_train, 30)
```

```
function = np.poly1d(coeffs)
```

Seuraavaksi luodaan x:n ja y:n arvot suoran piirtämistä varten. Datapisteet vastaavat opetus-setin datapisteitä, joten x saa arvoja väliltä [min(x_train), max(x_train)]. Muuttuja y:n arvot saadaan laskettua aiemmin luodusta funktiosta suoralle.

```
x_line = np.linspace(min(X_train), max(X_train))
```

```
y_line = function(x_line)
```

Tämän jälkeen piirretään opetus-setin pisteet ja sovitettu suora samaan kuvaajaan funktioiden **plt.scatter()** ja **plt.plot()** avulla. Kuvaajaan voi myös lisätä otsikon sekä nimetä akselit komennoilla **plt.title()**, **plt.xlabel()** ja **plt.ylabel()**.

Kun saat piirrettyä kuvaajaan datapisteet ja suoran, ota kommentti merkki pois funktion performance() päältä ja saat komentoriville suorituskyvyn sovitetulle suoralle. Kyseinen osio laskee absoluuttisen virheen, keskimääräisen neliövirheen (MSE), explained variance score:n, R2 score:n (korrelaatiokertoimen neliö) sekä suoran yhtälön kertoimet.

Arvioi lisäksi toteuttamasi mallin perusteella printtaamalla vastaus kysymykseen komentoriville datasetistä riippuen:

- ECTS: Kuinka monta opintopistettä oli kerätty keskimäärin 2 vuoden jälkeen?
- Life expectancy: Mikä on suomalaisten eliniänodote vuonna 2030?
- Population growth: Mikä on Suomen väkiluku vuonna 2035?
- Sea level: Mikä on vedenpinnankorkeus vuonna 2025?

Seuraavan linkin esimerkeistä voi olla hyötyä tehtävien tekemiseen: https://docs.scipy.org/doc/numpy-1.14.0/reference/generated/numpy.polyfit.html

Polynominen regressio

Korkeamman asteen käyrä saattaa olla parempi malli tuntemattomalle datalle kuin pelkkä lineaarinen suora. Seuraavaksi selvitään, että pääsemmekö merkittävästi parempaan lopputulokseen kasvattamalla sovitetun suoran kompleksisuutta. Yksinkertaisin laajennus suoran yhtälöstä on toisen asteen polynominen käyrä, joka tunnetaan nimellä paraabeli. Lisäämällä malliin korkeamman asteen termejä kuten X^2 tai X^3, voidaan tätä rinnastaa uusien riippumattomien muuttujien lisäämisenä perusmalliin. Matemaattisesti k:nnen asteen polynomista suoraa voidaan mallintaa yhtälöllä:

$$y = a_0 + a_1 x + a_2 x^2 + \cdots + a_k x^k$$

Matriisialgebran merkinnöillä kertoimet polynomisen suoran sovitukseen pienimmän neliösumman menetelmällä voidaan laskea yhtälöllä:

$$Y = XA \Leftrightarrow A = (X^T X)^{-1} X^T Y$$

,missä X viittaa polynomisen yhtälön x:n saamiin arvoihin matriisissa, Y viittaa y:n saamiin arvoihin matriisissa ja A viittaa laskettuihin kertoimiin $[a_0, a_1, a_2, \ldots, a_k]$*.*

Esimerkiksi käytetään mallina toisen asteen yhtälöä, joka on muotoa $y = a_0 + a_1x + a_2x^2$ ja yritetään sovittaa mallia pisteille (-1,2), (0,1), (1,2) ja (2,5). Tällöin matriisit X,Y ja A saa muodon:

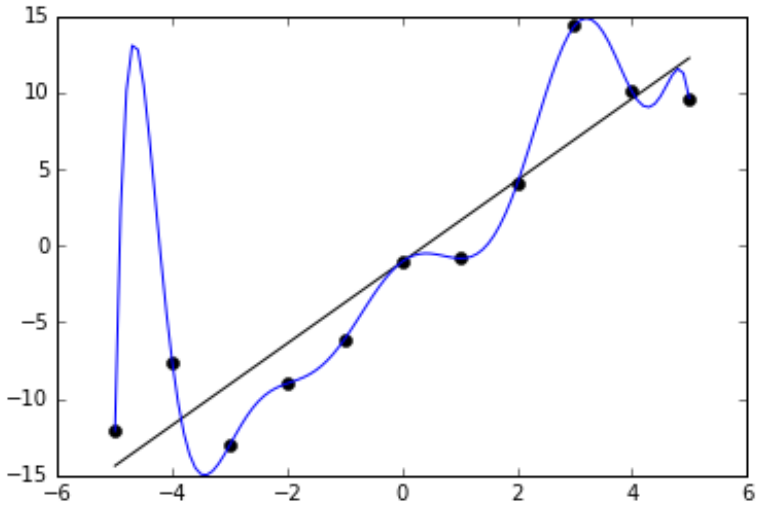
$$X = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix}, \quad A = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}, \quad Y = \begin{bmatrix} 2 \\ 1 \\ 2 \\ 5 \end{bmatrix}$$

Käyttämällä aiemmin johdettua yhtälöä saadaan

$$A = (X^TX)^{-1}X^TY = \begin{bmatrix} 0.55 & 0.15 & -0.25 \\ 0.15 & 0.45 & -0.25 \\ -0.25 & -0.25 & 0.25 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 0 & 1 & 2 \\ 1 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Eli kerroinmatriisiksi A saadaan ratkaisuksi $A = [1\ 0\ 1]^T$ ja toisen asteen yhtälöksi $y = 1 + x^2$.

Ylisovittaminen on yleinen ongelma ohjatussa oppimisessa sekä käyrän sovitus ongelmissa. Vaikka sovitetulla polynomisella mallilla olisi erittäin pieni virheiden määrä, se ei tarkoita sitä, että malli olisi parhaiten sovitettu malli. Jos malliksi valitaan liian korkea asteinen polynominen yhtälö, malli alkaa mallintamaan datassa esiintyvää kohinaa ja tapahtuu ylisovittaminen. Tämän kaltainen tilanne on esitetty kuvassa 2. Kuvasta huomataan, että parhaiten sovitettu malli kyseisille pisteille olisi yksinkertainen lineaarinen suora korkeamman asteen funktion sijaan.



Kuva 2. Ylisovittaminen

Tehtävä 2

Tehtäväsi on muodostaa yhdelle seuraavista dataseteistä polynominen regressiomalli:

- **data_exchange_rate.txt**: Euron (EUR) ja Ruotsin kruunun (SEK) välinen valuuttakurssi 90 päivän aikajaksolta. Vaaka-akselilla on aika päivinä aloitusajanhetkestä ja pystyakselilla on valuuttakurssi.
- **data_weather_oulu.txt**: Sunnuntaina 18.3.2018 Oulun vihreäsaaren sääaseman mittaamat lämpötilat. Vaaka-akselilla on kellonaika (00:00-24:00) ja pystyakselilla kyseisellä ajanhetkellä mitattu lämpötila.
- Voit myös vaihtoehtoisesti käyttää jotain omaa polynomista riippuvuutta noudattavaa dataa kahdelle eri muuttujalle.

Muokkaa tiedostoa nimeltä **polynomialregression.py**. Käytä numpy-, sklearn- ja matplotlib-kirjaston funktioita tehtävän 1 mukaisesti.

Piirrä kuvaajaan kolme eriasteista mallia eri väreillä ja viivoilla: Selvästi ylisovitettu malli, lineaarisesti sovitettu malli sekä silmämääräisesti arvioituna parhaiten sovitettu malli. Voit piirtää samaan kuvaajaan eri asteen polynomisia malleja esimerkiksi for-silmukan avulla ja arvioida performance() funktiolla mallien suorituskykyä.

Palauta

Palauta muokkaamasi python tiedostot (.zip tai .rar tiedostoon pakattuna) Optiman palautuslaatikkoon Harjoitus 2 **6.4.2018 klo 23:59** mennessä. Tästä harjoituksesta on mahdollisuus tienata maksimissaan 5 pistettä (2.5p + 2.5p).