KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

Rokas Staniulis

Paukščių migravimo analizė

Baigiamasis magistro projektas

Vadovas
lekt. Mindaugas Kavaliauskas

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

Paukščių migravimo analizė

Baigiamasis magistro projektas

Taikomoji matematika (621G10003)

Vadovas
lekt. Mindaugas Kavaliauskas

Recenzentas
dr. Evaldas Vaičiukynas

Projektą atliko
Rokas Staniulis MGTMM - 5

KAUNAS, 2017

TURINYS

Paveikslėlių sąrašas

Lentelių sąrašas

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Matematikos ir gamtos mokslų

(Fakultetas)

Rokas Staniulis

(Studento vardas, pavardė)

Taikomoji matematika

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Paukščių migravimojos analizė“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

0 ___ . _____ ____ .

Kaunas

Patvirtinu, kad mano, **Roko Staniulio**, baigiamasis projektas tema „Paukščių migravimo analizė“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

_____ _____

(vardą ir pavardę įrašyti ranka) *(parašas)*

PRATARMĖ

Vidutinei žemės temperatūrai per pastaruosius keliolika metų stipriai pakilus, žmonių ir gyvūnų elgesys gan ženkliai pasikeitė. Šiame darbe bandysime naudodami matematinį modeliavimą nustatyti, kokią įtaką paukščių migracijai daro oro sąlygos. Šiais laikais vis daugiau matematikų dirba kartu su biologais ir zoologais, kad galėtų sujungti savo žinias ir geriau suprasti gyvūnus. Yra atlikta daug tyrimų, kurie padeda nuspėti gyvūnų judėjimą ir elgesį. Pasitelkus matematinius skaičiavimus galima labai ženkliai palengvinti gyvūnų elgesio analizę ir padaryti tikslesnes išvadas apie jų elgseną bei jos priežastis.

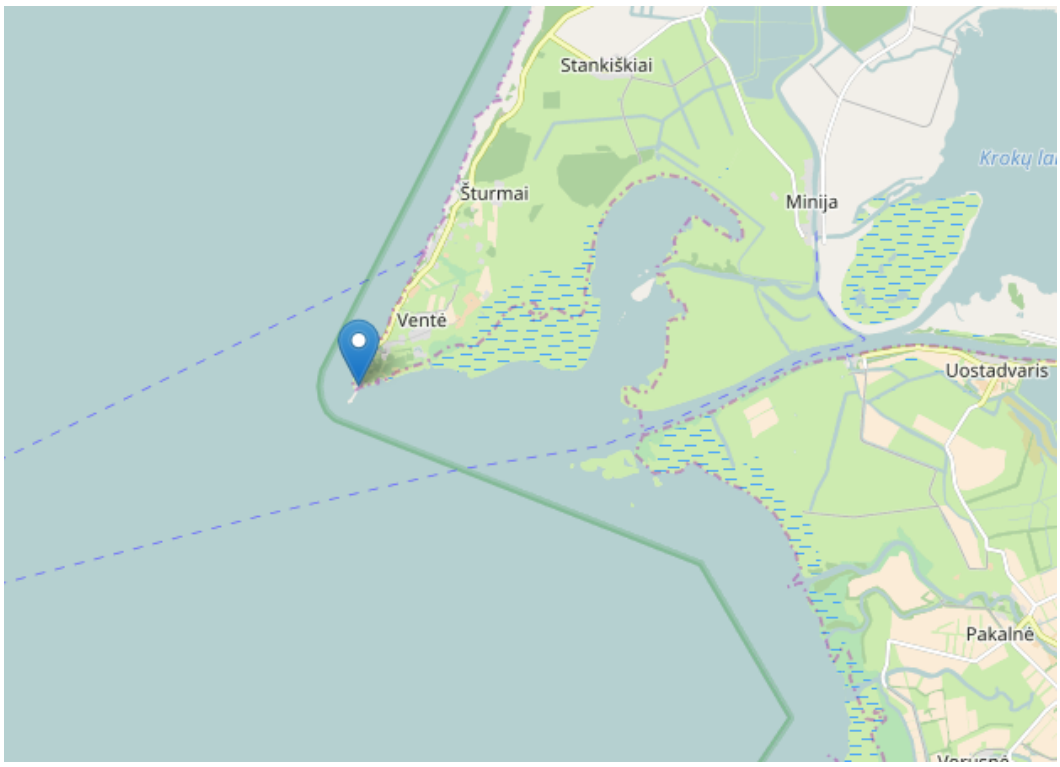
ĮVADAS

Pirmas žmogus, kuris pradėjo moksliniais tikslais žieduoti paukščius buvo Viborgo (Danija) parapijinės mokyklos mokytojas Hansas Christianas Kornelius Mortensenas (Hans Christian Cornelius Mortensen, 1856–1921). 1899 m. birželio 5 d. jis sužiedavo pirmąjį savo varnėną. Tai jis padarė naudodamas iš aliuminio padarytą žiedą, tame žiede buvo toks įrašas: „VIBORG 1“. Tai reiškia, Kad H. Mortensenas buvo pirmasis, kuris paukščiams ženklinti panaudojo numeruotus žiedus su įrašytu adresu. Iš šio proceso atsirado paukščių gyvenimo tyrimo metodas - paukščių žiedavimas. Mokslininkams yra labai svarbu visus metus registruoti paukščių elgsenos kitimus. Tačiau ypač svarbių duomenų galima gauti žieduojant neįprastomis oro sąlygomis, užsitęsusiomis arba šiltomis dienomis, šie duomenys gali padėti ateityje išsaugoti paukščių rūšis, pastebėjus neįprastų pasikeitimų jų fenologijoje. Dabar mokslininkai, pasiremdami šiais duomenimis, gali planuoti tiek paukščių apsaugos, tiek jų gyvenamųjų vietų priežiūros priemonės.

Ventės rago ornitologinė stotis buvo įkurta 1929 m. profesoriaus T. Ivanausko iniciatyva. Per vienerius metus pro Ventės ragą kartais praskrenda net iki 300 tūkst paukščių. Stotyje per metus sužieduojama apie 60 - 80 tūkst. paukščių. Rekordas buvo pasiektas 2002 m. rugsėjo 23 d., kai buvo sužieduoti net 6519 sparnuočiai per vieną parą. Stotyje galima rasti muziejų, kuris supažindina lankytojus su Kuršių marių gamtovaizdžiu, būdingiausiais biotopais, dažniausiai aptinkamais gyvūnais. Šiuo metu Ventės rago ornitologinei stočiai vadovauja Vytautas Jusys.

Pirmasis žmogus, kuris Ventės rage ėmė žieduoti paukščius, buvo Mikas Posingis, 1924 – 1944 m. dirbęs švyturio prižiūrėtoju. Jis pasiūlė prof. T. Ivanauskui čia įkurti žiedavimo stotį, kuri ir buvo 1929 m. padaryta profesoriaus pastangų dėka.

Iš pradžių paukščiai buvo gaudomi užmetant tinklus ant krūmų ir į juos įvarant paukščius. 1959 m. pastatytos specialios gaudyklės. Nuo 1962 m. stoties darbuotojai Kuršių Nerijoje pastatė gaudyklę ir ėmė ten žieduoti migruojančius paukščius (nuo 1986 m. žiedavimo darbus Neringoje perėmė naujai įsikūrusi Juodkrantės paukščių žiedavimo stotis). 1978 m. Ventės rage buvo pastatyta didžioji paukščių gaudyklė, kuri buvo iškeliama net į 25 metrų aukštį. 1982 m. stoties vedėju dirbęs Leonas Jezerskas sukūrė naujo tipo – zigzaginę gaudyklę į kurią paukščiai patekdavo tiek iš vienos, tiek ir iš kitos pusės.



1.1 pav. Ventės žiedavimo stoties vieta

Šiame darbe bus analizuojami valandiniai paukščių žiedavimo duomenys. Duomenys buvo surinkti Ventės rage. Darbe analizuosime duomenis, surinktus tarp 2009 ir 2013 metų. Dažniausiai paukščių žiedavimo duomenys buvo renkami dienos metu, įrašai nebuvo fiksuojami kiekvieną valandą, yra ir praleistų valandų. Duomenys sudaryti iš dviejų stulpelių. Pirmajame yra data, kurią sudaro metai, mėnuo, diena ir valanda, kada įrašas buvo fiksuotas. Antrajame stulpelyje yra skaičius, kiek paukščių buvo sužieduota per tą valandą. Šį kintamąjį galima priskirti santykių skalės kintamiesiems.

Paukščių žiedavimo duomenis papildysime oro sąlygomis, kurios buvo fiksuojamos žiedavimo metu Ventės rage[1]. Šį duomenų rinkinį sudarys penki stulpeliai: data, kartu su laiku kada reikšmės buvo fiksuojamos, oro temperatūra, oro drėgnumas, vėjo kryptis ir vėjo greitis. Darysime prielaidą, kad šie keturi parametrai daro didžiausią įtaką sužieduotų paukščių skaičiui. Oro temperatūra bus pateikta Celsijaus laipsniais, oro drėgnumas – milimetrais. Oro drėgnumas, rodo kiek milimetrų lietaus iškrito per vieną valandą. Pavyzdžiui, 1 milimetras lietaus reiškia, kad kiekviena vieno kvadratinio metro dydžio sritis yra pripildyta vandens, kurio aukštis yra vienas milimetras. Vėjo greitis bus matuojamas metrais per sekunde. Vėjo kryptis kis intervalu [0;360], kur 0 reikš šiaurinį vėją, 360 pietinį vėją ir t. t.

Remdamiesi šiais parametrais bandysime nustatyti, kaip sužieduotų paukščių skaičių veikia šie keturi parametrai. Analizei atlikti daugiausiai naudosime „Microsoft Excel“ ir atviro kodo „R“ programą. „Microsoft Excel“ naudojau įvairioms manipuliacijomis su duomenimis ir dviejų duomenų rinkinių sujungimui, o R bus naudojama statistinės analizės procedūroms atlikti.

R programa yra atvirojo kodo duomenų analizės įrankis, skirtas statistikos, skaičiavimo, grafinėms ir daugeliui kitų funkcijų atlikti. Išmokus naudotis R programa, galima užmiršti daugelį atskirų statistinių ar grafinių programų, kurių prireikdavo atlikti įvairias statistines ar grafines užduotis. Nors R yra viena iš populiariausių programavimo kalbų, tačiau joje labai mažai išvystyta meniu sistema (pvz., palyginus su SPSS, STATISTICA), o komandos rašomos į komandinę eilutę. Dėl šios priežasties R aplinka nėra patogi pradedantiesiems, bet egzistuoja daug literatūros (vadovai, knygos, straipsniai) ir internetinės medžiagos (skriptai, filmuotos medžiagos, dienoraščių, diskusijų bei forumų), iš kurių galima nesunkiai pačiam išmokti naudotis R programa.

„Excel“ – tai „Microsoft Office“ skaičiuoklės programa. Naudodami „Excel“ galite kurti ir formatuoti darbaknyges (skaičiuoklių rinkinius), kad galėtumėte analizuoti duomenis ir priimti įžvalgesnius verslo sprendimus. Konkrečiai kalbant, „Excel“ galima naudoti duomenų analizei skirtiems modeliams kurti, tų duomenų skaičiavimams skirtoms formulėms rašyti, duomenims daugybe būdų transformuoti, ir duomenims pateikti įvairiose profesionaliai atrodančiose diagramose.

Darbo tikslai bus susipažinti su programos R statistinės analizės galimybėmis, susipažinti su paketais naudojamais prognozavimui bei modeliavimui atlikti, panaudojant kelis statistinės analizės metodus, sukurti modelius sužieduotų paukščių skaičiui prognozuoti, juos palyginti tarpusavyje ir įvertinti jų tikslumą. Darbo metu susipažinsime su vienmatėms laiko eilutėms, analizuoti skirtais metodais bei palyginsime juos su regresinės analizės metodais.

Staniulis, Rokas. Paukščių migravimo analizė. Magistro baigiamasis projektas, vadovas lekt. Mindaugas Kavaliauskas; Kauno technologijos universitetas, Matematikos ir gamtos mokslų fakultetas.

Mokslo kryptis ir sritis: Laiko eilutės, regresinė analizė.

Reikšminiai žodžiai: regresinė, laiko eilutės.

Kaunas, 2017. 50 p.

SANTRAUKA

Šiame darbe aš tirsiu kaip oro sąlygos veikia paukščių migracijos intensyvumą. Darbe tirsime valandinius duomenis, kurie buvo renkami nuo 2009 iki 2013 metų. Duomenys buvo renkami Ventės rage, paukščių žiedavimo stotyje. Ši stotis yra Lietuvos pajūryje. Tirsiu du duomenų rinkinius. Viename iš jų yra duomenys, kuriuose yra nurodyta kiek paukščių buvo sužieduota tam tikrą valandą. Kitame duomenų rinkinyje yra oro sąlygų rodikliai, tam tikru metu toje pačioje vietoje. Šiuos du rinkinius sujungsime į vieną ir panaudodami įvairius analizės metodus tirsime priklausomybę tarp sužieduotų paukščių ir oro sąlygų. Tyrimui naudosiu laiko eilučių ir regresinės analizės metodus. Naudodamasis sukurtais modeliai atliksiu paukščių migracijos intensyvumo analizę ir į vertinsiu atliktos analizės tikslumą.

Staniulis, Rokas. BIRD MIGRATION ANALYSIS: Master‘s thesis in Data analysis supervisor assoc. prof. Mindaugas Kavaliauskas. The Faculty of Math and nature sciences, Kaunas University of

Technology.

Research area and field: Time series, regression analysis.

Key words:Time series, regression.

Kaunas, 2017. 50 p.

SUMMARY

In this work we will analyse how air conditions affects birds migration intensity. We will analyse hourly data which was collected from year 2009 to 2013. Data was colected in Ventes ragas birds ringing station. This station is in the coast of Lithuania. In one data set we have numbers which show how many birds was captured in particular hour. I other data set we can see what air conditions was in that time in that particular place. In this work we will connect those two data sets into one and then analise it by various methods. Using that data set we will try to forecast mird migration intensity. Firstly we will check if we should create different forecasting models for different months in the year.

Then we will create forecasting model using regression analysis methods. After tahat we will try to forecast bird migration analysis with time series analysis methods. After creation of these two models we will calculate error of both models. When we compare those errors we will know which model is better.

1. LITERATŪROS APŽVALGA

1.1. LAIKO EILUTĖS SAMPRATA

Tarkime turime tam tikro atsitiktinio dydžio arba kintamojo Z reikšmes. Jeigu mes stebime jas laikui einant pavyzdžiui: kas minutę, dieną, mėnesį, metus ir t.t. Realybėje tai dažniausiai būna miesto gyventojų skaičiaus didėjimas, įvairių visuomenės sluoksnių socialinio aktyvumo kitimas, pramonės ir žemės ūkio rodiklių kitimai ir t.t. Surašę visus gautus stebėjimus į vieną seką gauname atsitiktinį dydį, kurio reikšmių seka $Z_1,...,Z_T$ yra vadinama laiko eilute.

Atliekant tyrimus su laiko eilutėmis yra laikoma, kad mes žinome $Z(t_i)$ reikšmes laiko momentais $t_1<t_2<...<t_n$, o visi stebėjimai yra atliekami laikantis vienodų laiko intervalų, t.y.

$t_{i+1}-t_i = \Delta t$.

Jeigu mes stebime vieno rodiklio reikšmių kitimą, tai gauname vienmatę laiko eilutę, o stebint vieno objekto *m* rodiklių reikšmių kitimą, yra gaunama daugialypė laiko eilutė.

Pagrindiniai skirtumai tarp laiko eilutės ir atsitiktinio duomenų rinkinio, kurį galima naudoti regresinėje analizėje yra: 1) stebiniai sudarantys laiko eilutę nėra tarpusavyje nepriklausomi: 2) dydžiai sudarantys laiko eilutę nėra tolydžiai pasiskirstę laiko ašyje, t.y.

$$P\{Z(t_1) < Z\} \neq P\{Z(t_2) < Z\}, \text{ kai } t_1 \neq t_2. \quad \textbf{(1.1)}$$

Z(t) – bus tolydi jeigu laikas bus nenutrūkstamas. Jeigu turime diskretų laiką tai tokią laiko eilutę vadinsime diskrečia ir žymėsime Z_t.

Tiriant įvairius procesus, dažniausiai analizuojame diskrečias laiko eilutes. Apibrėžiant tokias eilutes yra svarbu fiksuoti ne tik laiko intervalą Δt bei stebėjimų skaičių *n*, bet ir pradinį laiko momentą t_0. Laiko eilutės gali būti dviejų rūšių: 1) momentinės; 2) intervalinės.

Analizuojant laiko eilutes dažniausiai sprendžiame tris pagrindinius uždavinius:

1. Identifikacijos, t.y. modelio parametrų statistinis įvertinimas;
2. Verifikacijos, t.y. sudaryto modelio adekvatumo patikrinimas;
3. Prognozavimo, t.y. laiko eilutės reikšmių laiko momentais Z_i nustatymas, kai $l > n$.

Modelio identifikacijos procedūra:

- Laiko eilutės duomenų vizualizacija ir tinkamos transformacijos parinkimas;
- ACF ir PACF analizė;
- Modelio parametrų įvertinimas – įvertinami parametrai, naudojant maksimalaus tikėtinumo metodą.

Laiko eilutės stacionarumo įvertinimas - vienas pirmųjų eilučių analizės uždavinių. Nuo laiko eilutės stacionarumo priklauso vidurkio funkcijos pavidalas. Stacionariame procese laiko eilutės reikšmės kinta atsitiktinai kiekvienu momentu, tačiau vidurkis gana ilgą laiką nekinta. Nestacionarių laiko eilučių vidurkis nėra pastovus, bet ilgainiui kinta. Dažnai realiose situacijose

laiko eilutės yra nestacionarios. Visada yra svarbu, kad laiko eilutė būtų stacionari siaurąja prasme, t.y. būtų išpildytos stacionarumo sąlygos vidurkio ir kovariacinės funkcijos atžvilgiu. Natūralus būdas apriboti stacionarumo efektus, yra tiesinio trendo taikymas laiko eilutės Z_t $t = 0, \pm 1, \pm 2...$ duomenims, naudojant paprasčiausią tiesinę regresiją. Stebimo kintamojo vidurkių kitimo tendencija yra vidurkių trendas. Trendo pavadinimas nurodo, kokią kreivę jo grafikas primena: tiesinio – tiesę, kvadratinio – parabolę ir pan.

Norint iš duomenų pašalinti trendą, yra naudojami skirtumai. Vienas skirtumas pašalina tiesinį trendą, kitas gali pašalinti kvadratinį trendą ir t.t.

Eilutės turinčios ilgą laikinę priklausomybę dažnai pasižymi autokoreliacijomis, kurių reikšmės yra nebūtinai didelės, tačiau išlieka ilgą laiką.

1.2.PROGNOZĖS TIKSLUMO VERTINIMAS

Tarkime, kad y_i žymi i – tąjį stebėjimą, o $\hat{y}_i$ žymi y_i prognozę. Prognozės paklaida yra tiesiog:

$$e_i = y_i - \hat{y}_i, \tag{1.2}$$

Šios paklaidos skalė sutampa su duomenų skale. Todėl tikslumo matavimai, kurie remiasi, e_i yra priklausomi nuo skalių ir negali būti naudojami lyginant dviejų atskirų laiko eilučių su skirtingomis skalėmis. Du populiariausi nuo matavimo skalės priklausomi matavimai remiasi absoliučia arba kvadratinė paklaida:

Vidutinė absoliutinė paklaida:

$$\text{MAE} = \text{mean}(|e_i|), \tag{1.3}$$

Vidutinė kvadratinė paklaida:

$$\text{RMSE} = \sqrt{\text{mean}(e_i^2)}. \tag{1.4}$$

Kai lyginame prognozavimo metodus tame pačiame duomenų rinkinyje, MAE yra populiariausia paklaida, nes ją lengva apskaičiuoti ir suprasti.

Procentinės paklaidos yra apskaičiuojamos taip:

$$p_i = 100e_i/y_i. \tag{1.5}$$

Didžiausias procentinių paklaidų privalumas yra tas, kad jos nėra priklausomos nuo dydžių skalės, todėl yra labai dažnai naudojamos palyginti rezultatus tarp dviejų skirtingų duomenų rinkinių.

Vidutinė absoliutinė procentinė paklaida:

$$\text{MAPE} = \text{mean}(|p_i|). \tag{1.6}$$

Pagrindinis priemonių, kurios remiasi procentinėmis paklaidomis trūkumas yra tas, kad ji gali būti begalinė arba neapibrėžta kai $y_i = 0$ bent vienam i tiriamame laiko tarpe ir gali turėti ekstremalias reikšmes kai y_i yra arti 0. Kita problema su procentinėmis paklaidomis yra ta, kad jos nekreipia dėmesio į reikšmingus 0. Pavyzdžiui jeigu turime prognozę temperatūrai Celsijaus arba Farenheito skalėje, Procentinės paklaidos neturi jokios prasmės. Taip pat šios rūšies paklaidos suteikia didesnį svorį neigiamiems skirtumams. Dėl šio trūkumo atsirado taip vadinama simetrinė MAPE (sMape). Ji apibrėžiama taip:

$$\text{sMAPE} = \text{mean}\left(200|y_i - \hat{y}_i|/(y_i + \hat{y}_i)\right). \tag{1.7}$$

Tačiau, jei y_i yra arti 0, greičiausiai ir $\hat{y}_i$ bus arti 0. Visgi į skaičiavimus įeina dalyba iš skaičiaus artimo nuliui, tai padaro skaičiavimus nestabilius. Taip pat sMAPE gali būti neigiama.

Skalei atsparios paklaidos pirmą kartą buvo pasiūlytos 2006 – aisiais metais. Jas išrado mokslininkai Hyndmanas ir Koeleris. Ši paklaidos skaičiavimo metodika buvo naudojama kaip

alternatyva procentinėms paklaidoms. Tada kai norima palyginti prognozės tikslumus laiko eilutėse su skirtingomis skalėmis. Mokslininkai pasiūlė paklaidų mastelį, paprastiems prognozavimo metodams, pritaikyti į apmokomajai MAE. Ne sezoninėms laiko eilutėms, paprastas būdas apibrėžti masteliui pritaikytą paklaidą yra naudojant naivų prognozavimą:

$$q_j = \frac{e_j}{\frac{1}{T-1}\sum_{t=2}^{T}|y_t - y_{t-1}|}. \quad \textbf{(1.8)}$$

Kadangi trupmenos skaitiklis ir vardiklis turi reikšmių iš pradinių duomenų mastelio, q_j yra nepriklausoma nuo duomenų mastelio. Masteliui pritaikytos paklaidos reikšmė yra mažiau už vienetą jei prognozė yra geresnė nei vidutinė naivi prognozė, kuri buvo apskaičiuota apmokomajai imčiai. Jeigu paklaidos reikšmė didesnė už vienetą tai prognozė yra blogesnė už vidutinę naivią prognozę. Sezoninėms laiko eilutėms, masteliui pritaikyta paklaida yra apskaičiuojama naudojant sezoninį naivų prognozavimą:

$$q_j = \frac{e_j}{\frac{1}{T-m}\sum_{t=m+1}^{T}|y_t - y_{t-m}|}. \quad \textbf{(1.9)}$$

Vidutinė absoliutinė masteliui pritaikyta paklaida (MASE) skaičiuojama taip:

$$\mathrm{MASE} = \mathrm{mean}(|q_j|). \quad \textbf{(1.10)}$$

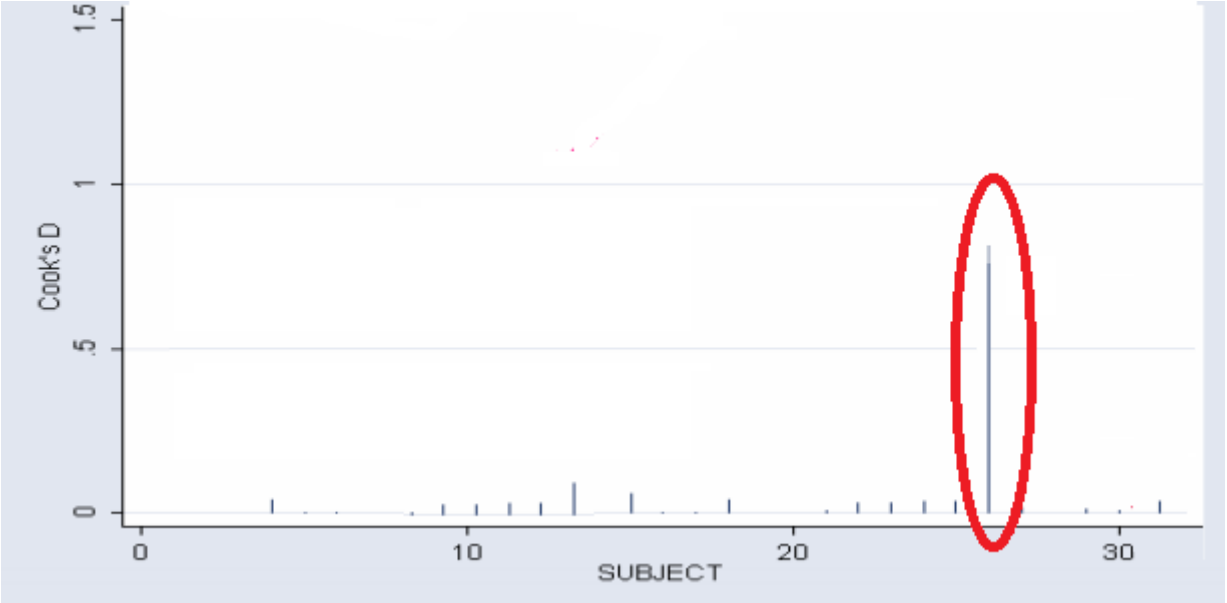
Akaikės informacinis kriterijus yra matmuo, parodantis sudaryto statistinio modelio tinkamumą. Su šiuo kriterijumi galime įvertinti informacijos praradimą modelį pritaikius realiems duomenims, taip pat galime apibūdinti taip sukonstruoto modelio poslinkį ir dispersiją kitaip tariant tai yra ryšį tarp modelio tikslumo ir sudėtingumo. Šis kriterijus negali patikrinti hipotezių apie modelio korektiškumą, todėl naudodami jį nenustatysime ar modelis yra tinkamas. AIC kriterijaus reikšmė turi būti kuo mažesnė. Matematinė kriterijaus išraiška:

$$AIC = 2k - 2\ln(\mathrm{L}) \quad \textbf{(1.11)}$$

k yra parametrų skaičius, n – laiko eilutės ilgis (duomenų skaičius), L – maksimali modelio, tikėtinumo funkcijos reikšmė.

1.3. KUKO ATSTUMAS

Kuko atstumas D_i yra naudojamas regresinėje analizėje, reikšmingoms išskirtims nepriklausomų kintamųjų rinkiniuose surasti. Kitaip tariant tai yra būdas identifikuoti taškus, kurie neigiamai veikia jūsų regresijos modelį. Šis rodiklis yra kombinacija susidedanti iš kiekvienos reikšmės įtakos ir liekanos reikšmių. Kuo didesnė įtaka ir liekana tuo didesnis Kuko atstumas.



1.2 pav. Kuko atstumas

- Pagrindinė taisyklė yra tokia, kad jeigu stebėjimo Kuko atstumas daugiau nei tris kartus viršija vidurkį, tas stebėjimas yra išskirtis.
- Kitas požiūris į Kuko atstumą yra toks, kad kiekvienas taškas, kurio Kuko atstumas viršija 4/n, kur n yra stebėjimų skaičius, yra išskirtis.
- Kiti autoriai siūlo, kad kiekviena didelė D_i turi būti ištirta. Bet kiek didelė yra per didelė? Egzistuoja sutarimas, kad D_i didesnis negu 1 rodo įtakingą reikšmę, bet galima atidžiau pažiūrėti ir į taškus su didesne negu 0.5 reikšme. Taip pat reikėtų atidžiau pažiūrėti ir į kiekvieną reikšmę, kuri stipriai išsiskiria iš aplinkinių.

Jeigu gaunate labai daug taškų su didele D_i tai gali reikšti, kad turite problemų su visu regresiniu modeliu. Techniškai Kuko atstumas yra skaičiuojamas išimant i – tąjį stebėjimą iš regresinio modelio ir perskaičiuojant visą modelį. Taip surandama kiek kiekviena reikšmė regresijos modelyje, pasikeičia kai i – toji reikšmė yra pašalinta. Kuko atstumo apskaičiavimo formulė atrodo taip:

$$D_i = \frac{\sum_{j=1}^{n} (\hat{Y}_j - \hat{Y}_{j(i)})^2}{p\ \mathrm{MSE}}. \quad \textbf{(1.12)}$$

čia:

· $\hat{y}_j$ yra j – toji pritaikyta reikšmė.

· $\hat{y}_{j(i)}$ yra j – toji pritaikyto modelio reikšmė, kai modelis neįtraukia i – tojo stebėjimo.

· MSE yra vidutinė kvadratinė paklaida.

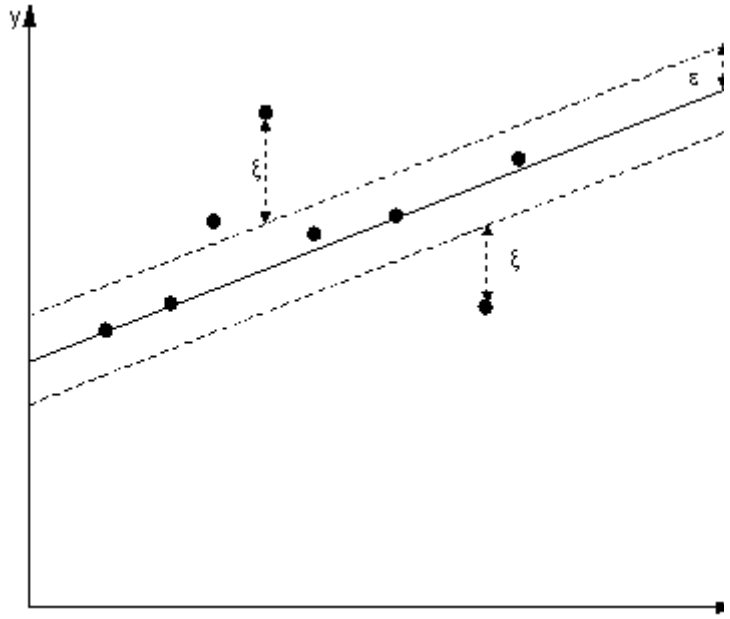
· p parodo kiek iš viso yra koeficientų regresiniame modelyje.

1.4. ATRAMINIŲ VEKTORIŲ REGRESIJA

Atraminių vektorių regresija yra labai specifiška algoritmų klasė, ją aprašo branduolių naudojimas, lokalių minimumų nebuvimas, atraminių vektorių skaičius ir t.t.

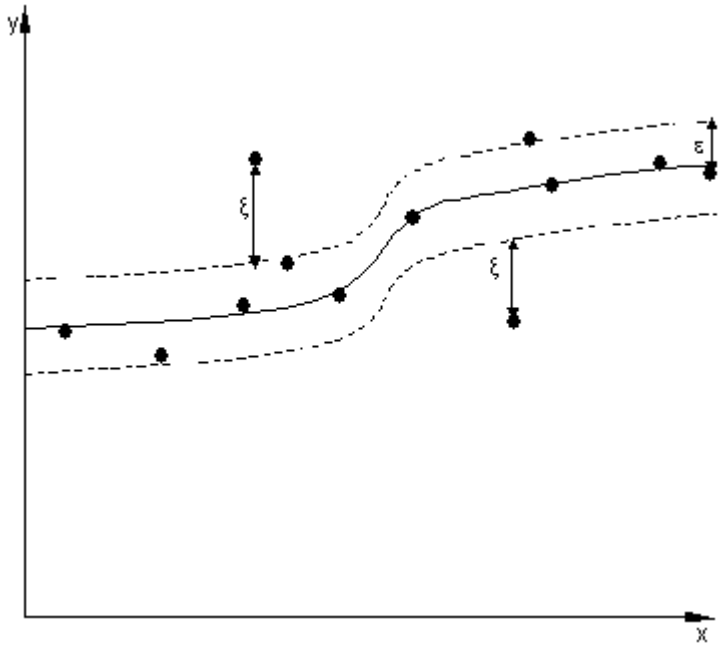
Atraminius vektorius atrado Vladimiras Vapnikas kartu su savo bendradarbiais 1992 metais. Atraminių vektorių analizė išpopuliarėjo tik po trijų metų 1995. Iš pradžių daugiausiai ji buvo naudojama kintamųjų klasifikacijai.

Atraminių vektorių mašina gali būti taikom ne tik su klasifikacija susijusioms problemoms spręsti, bet ir regresiniams modeliams kurti. Tai darom tokiu būdu: tiesinio mokymosi mašina išlanksto netiesinę funkciją, projektuodama ją daugiadimensinio branduolio paveiktų savybių erdvėje. Sistemos talpa yra kontroliuojama naudojant parametrus, kurie nepriklauso nuo savybių erdvės dimensijos. Kaip ir klasifikavime yra siekiama optimizuoti taisyklės, kuria parašome regresiją, ribas. Tai daroma remiantis baudos funkcija, kuri ignoruoja klaidas, esančias tam tikrose ribose nuo tikrosios reikšmės. Tokio tipo funkcijos dažniausiai yra vadinamos epsilon intensyvumo baudos funkcijomis. Žemiau esantis paveikslėlis vaizduoja vienos dimensijos tiesinę regresiją su epsilon intensyvumo juosta. Kintamieji parodo apmokomųjų taškų klaidų kainas. Kainos yra lygios 0 kai taškai patenka į epsilon zoną.



1.3 pav. Tiesinė regresija su epsilon pločio zona

Žemiau esantis paveikslėlis parodo panašią situaciją tik su netiesine regresija.



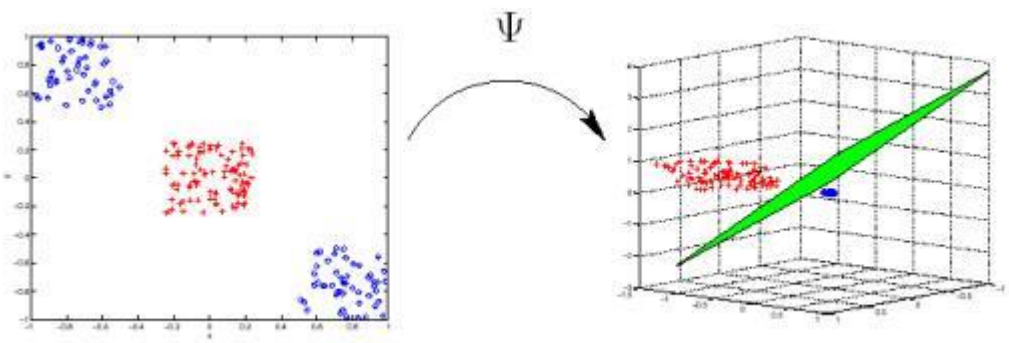
1.4 pav. Netiesinė regresija su epsilon pločio zona

Viena svarbiausių idėjų atraminių vektorių klasifikavime ir regresijoje yra tai, kad ieškodami atsakymo galime ignoruoti tam tikro dydžio klaidas ir tai mums duoda labai didelius pranašumus skaičiavimų prasme. Naudojant epsilon intensyvumo baudos funkciją mes užtikriname globalaus minimumo egzistavimą ir tuo pačiu metu optimizuojame patikimą duomenų aprašymo taisyklę.

SVM regresijoje pradinis duomenų rinkinys x yra suprojektuojamas į m – dimensinę savybių erdvę. Tai daroma naudojant fiksuotą projektavimą, ir tada tiesinis modelis yra sukuriamas šioje savybių erdvėje. Naudojant matematinį žymėjimą, tiesinis modelis f(x, ω) užrašomas taip:

$$f(\mathbf{x}, \omega) = \sum_{j=1}^{m} \omega_j g_j(\mathbf{x}) + b \qquad \textbf{(1.13)}$$

Kur $g_j(x)$, j = 1,...,m žymi netiesinių transformacijų rinkinį, o b yra poslinkio dydis.



1.5 pav. Pradinių duomenų netiesinė projekcija į daugiamatę savybių erdvę

Duomenų vertinimo kokybė yra nustatoma naudojant baudos nuostolių funkciją L(y,f(x, ω)). Atraminių vektorių regresija naudoja naują nuostolių funkciją, kuri vadinasi epsilon intensyvumo nuostolių funkcija:

$$L_\varepsilon(y, f(\mathbf{x}, \omega)) = \begin{cases} 0 & if\ |y - f(\mathbf{x}, \omega)| \le \varepsilon \\ |y - f(\mathbf{x}, \omega)| - \varepsilon & otherwise \end{cases} \quad \textbf{(1.14)}$$

Empirinė rizika yra:

$$R_{emp}(\omega) = \frac{1}{n}\sum_{i=1}^{n} L_\varepsilon(y_i, f(\mathbf{x}_i, \omega)) \quad \textbf{(1.15)}$$

SVM regresija atlieka tiesinę regresiją daugiadimensinėje savybių erdvėje naudodama epsilon intensyvius nuostolius ir tuo pačiu metu bando minimizuoti $\|\omega\|^2$. Tai galime aprašyti įvesdami laisvuosius kintamuosius $\xi_i, \xi_i^* \quad i = 1,\ldots n$, jų pagalba išmatuosime apmokymo imties taškų, kurie nepatenka į epsilon zoną, dispersiją. Tokiu būdu SVM regresija galėsime suformuluoti kaip žemiau esančios funkcijos minimizavimą:

$$\min \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \quad \textbf{(1.16)}$$

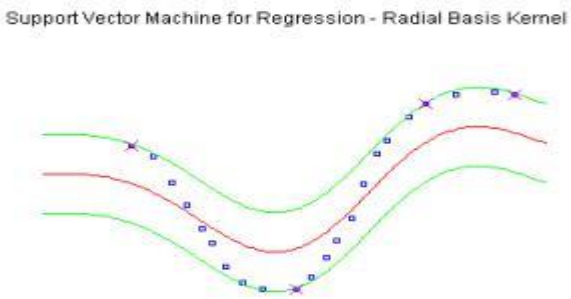
$$\text{s.t.} \begin{cases} y_i - f(\mathbf{x}_i, \omega) \le \varepsilon + \xi_i^* \\ f(\mathbf{x}_i, \omega) - y_i \le \varepsilon + \xi_i \\ \xi_i, \xi_i^* \ge 0, i = 1,\ldots,n \end{cases} \quad \textbf{(1.17)}$$

Ši optimizacijos problema gali būti paversta į dvilypį uždavinį ir jo sprendinys randamas taip:

$$f(\mathbf{x}) = \sum_{i=1}^{n_{SV}}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) \quad \text{s.t.}\ 0 \le \alpha_i^* \le C,\ 0 \le \alpha_i \le C, \quad \textbf{(1.18)}$$

Kur n_{sv} yra atraminių vektorių skaičius o branduolio funkcija atrodo taip:

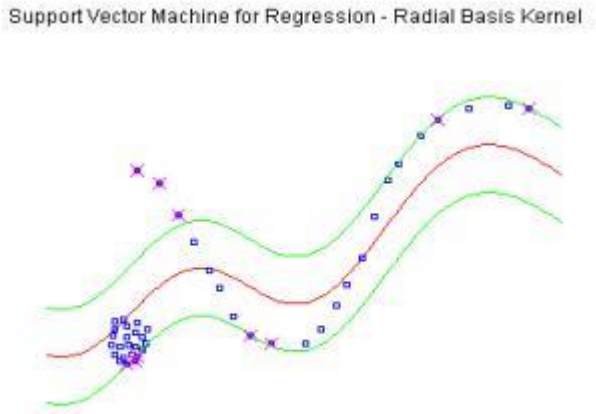
$$K(\mathbf{x}, \mathbf{x}_i) = \sum_{j=1}^{m} g_j(\mathbf{x}) g_j(\mathbf{x}_i) \quad \textbf{(1.19)}$$



1.6 pav. SVM regresijos atvejis. Epsilon ribos yra pažymėtos žaliomis linijomis, o mėlyni taškai žymi pradinius duomenis

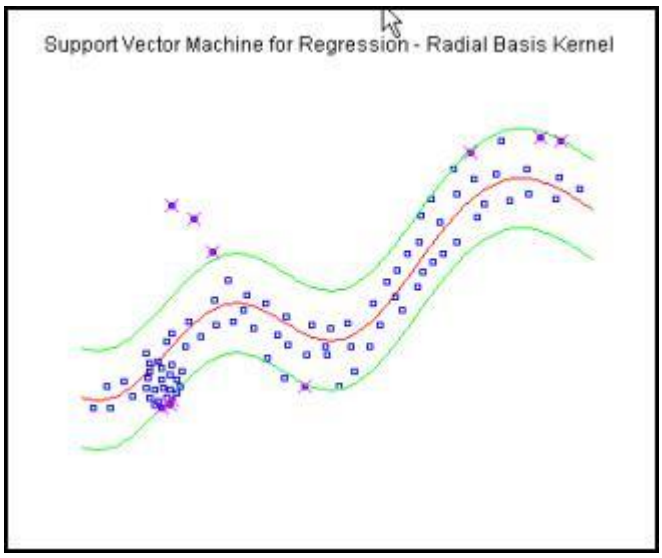
Yra žinoma, kad SVM tikslumas priklauso nuo gero meta parametrų C ir epsilon parinkimo. SVM modelio sudėtingumas priklauso nuo abiejų anksčiau minėtų parametrų. Parametras C nustato santykį tarp metodo sudėtingumo ir laipsnį iki kurio dispersijos didesnės už epsilon dar yra toleruojamos optimizacijos formulavime. Jei C reikšmė yra per didelė , tada bandome minimizuoti empirinę riziką , nekreipiant dėmesio į modelio sudėtingumo dalį optimizacijos formulavime.

Parametras ε valdo ε - intensyvumo zonų plotį, kurios yra naudojamos apmokyti modelį apmokomajai imčiai. ε reikšmė gali paveikti atraminių vektorių skaičių, kuris bus naudojamas konstruojant regresijos funkciją. Kuo ε didesnis tuo mažiau atraminių vektorių modelis naudos. Tačiau didelės ε reikšmės gali lemti labai jau apytikslius paskaičiavimus, kitaip tariant modelis mažiau atspindės imties savybes.



1.7 pav. SVM regresijos atvejis. Epsilon ribos yra pažymėtos žaliomis linijomis, o mėlyni taškai žymi pradinius duomenis

Aukščiau matomame paveikslėlyje į modelį yra įtraukiama naujų reikšmių. Kadangi kai kurios reikšmės iš naujai įvestų atsidūrė už epsilon ribų, regresijos funkcija šiek tiek pasikeičia. Dėl šios priežasties kai kurio senosios reikšmės nebepatenka į epsilon ribas.



1.8 pav. SVM regresijos atvejis. Epsilon ribos yra pažymėtos žaliomis linijomis, o mėlyni taškai žymi imties taškus

Aukščiau matomame paveikslėlyje į modelį yra įtraukiama naujų reikšmių. Kadangi visos reikšmės yra epsilon ribose, dėl šios priežasties modelis visiškai nepakinta.

1.5. KELI PAPRASTI PROGNOZAVIMO METODAI

Kai kurie prognozavimo metodai yra labai paprasti ir stebėtinai efektyvūs. Žemiau apžvelgsiu tris, kuriuos naudosime kaip etaloną kitiems prognozavimo metodams.

1.5.1.VIDURKIO METODAS

Šiame metode visos reikšmes, kurias išprognozuojame turi vienodas reikšmes ir jos yra lygios mūsų turimos imties vidurkiui. Taigi savo imtį pažymėsime $y_i,\ldots,yT$, tada mūsų prognozė atrodys taip:

$$\hat{y}_{T+h|T} = \bar{y} = (y_1 + \cdots + y_T)/T. \tag{1.20}$$

Čia $\hat{y}_{T+h|T}$ reiškia y_{T+h} įvertį remiantis $y_i,\ldots,yT$. Nors mes ir naudojome laiko eilučių žymėjimą, šis metodas gali būti naudojamas prognozuoti ir reikšmėms, kurių nėra mūsų duomenyse. Kiti du metodai bus tinkami tik laiko eilutėms prognozuoti.

1.5.2.NAIVUS METODAS

Šis metodas yra tinkamas tik laiko eilučių prognozėms atlikti. Visi spėjimai paprasčiausiai yra lygūs paskutiniai turimų duomenų reikšmei. Taigi visos ateities prognozės yra lygios y_T, jeigu y_T yra paskutinio stebėjimo reikšmė. Šis metodas veikia labai gerai su daugeliu ekonominių ir finansinių laiko eilučių.

1.5.3.PASLANKUS METODAS

Šis metodas leidžia prognozėms didėti arba mažėti kintant laikui, to kitimo dydis yra lygus vidutiniam kitimui pradiniuose duomenyse. Taigi prognozė periodui T + h yra užrašoma tokiu būdu:

$$y_T + \frac{h}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) = y_T + h\left(\frac{y_T - y_1}{T-1}\right). \tag{1.21}$$

Paprastai kalbant šis metodas tiesiog sujungia pirmą ir paskutinį imties taškus, toliau ekstrapoliuodamas tą tiesę į ateitį.

1.6. AUTOREGRESINIAI MODELIAI

AR metodologija bando aprašyti stacionarios laiko eilutės judėjimą naudojant funkciją su taip vadinamais autoregresyviais ir slenkančio vidurkio parametrais. Šie parametrai žymimi AR(autoregresiniai) ir MA(slenkantys vidurkiai). AR modelis su vienu parametru gali būti užrašomas taip:

$$X(t) = A(1) * X(t-1) + E(t) \quad \textbf{(1.22)}$$

kur X(t) = tiriama laiko eilutė

A(1) = pirmos eilės autoregresyvus parametras

X(t-1) = laiko eilutės reikšmė ankstesniame periode

E(t) = modelio paklaida

Šis žymėjimas reiškia, kad kiekviena X(t) reikšmė gali būti išreikšta tam tikromis jos ankstesnės reikšmės X(t-1) reikšmėmis ir tam tikromis nepaaiškinamomis atsitiktinėmis paklaidomis E(t). Jei apskaičiuota A(1) reikšmė buvo 0.30 tai dabartinė laiko eilutės reikšmė bus 30% susijusi su savo reikšme prieš vieną periodą. Žinoma eilutė gali būti susijusi ir su daugiau nei viena praeities reikšme Pavyzdžiui:

$$X(t) = A(1) * X(t-1) + A(2) * X(t-2) + E(t) \quad \textbf{(1.23)}$$

Tai rodo, kad dabartinė laiko eilutės reikšmė yra dviejų paskutinių reikšmių kombinacija, X(t-1) ir X(t-2), plius kelios atsitiktinės paklaidos E(t). Mūsų modelis tada būtų antros eilės autoregresyvus modelis. Kita Box-Jenkins modelio rūšis yra vadinama slenkančio vidurkio modeliu. Nors šis modelis ir atrodo labai panašus į AR modelį, jų principai gan stipriai skiriasi. Slenkančio vidurkio parametrai yra priklausomi tik nuo to kas atsitiko periode t su atsitiktinėmis paklaidomis praėjusiuose E(t-1), E(t-2) ir t.t. Priešingai nei autoregresyviuose modeliuose kur viskas priklauso nuo X(t-1), X(t-2), X(t-3). Slenkančio vidurkio modelis su vienu MA gali būti užrašytas taip:

$$X(t) = -B(1) * E(t-1) + E(t) \quad \textbf{(1.24)}$$

B(1) yra vadinamas pirmos eilės slenkančiu vidurkiu. Minuso ženklas priekyje yra naudojamas tik dėl patogumo ir yra automatiškai vaizduojamas daugumoje kompiuterinių programų. Aukščiau esantis modelis tiesiog teigia, kad kiekviena X(t) reikšmė yra tiesiogiai susijusi tik su periodo E(t-1) atsitiktinėmis paklaidomis ir su dabartinio periodo paklaida E(t). Kaip ir autoregresyviame modelyje slenkančio vidurkio modelis gali būti papildytas iki aukštesnės eilės struktūrų, kurios atspindėtų skirtingas kombinacijas ir slenkančio vidurkio ilgius.

1.6. ARIMA MODELIS

Stacionarios laiko eilutės prognozavimo lygties lagai yra vadinami autoregresyvumu. Prognozavimo paklaidų lagai yra vadinami slenkančiu vidurkiu, o laiko eilutės, kurias reikia diferencijuoti, kad jos taptų stacionariomis, yra vadinamos integruotomis stacionarios laiko eilutės versijomis. Kintančio žingsnio, atsitiktinio trendo, autoregresiniai modeliai ir eksponentinio glodinimo modeliai yra specialūs arima modelio atvejai. Tarkime turime tokį "ARIMA(p,d,q)" ne sezoninį ARIMA modelį. Parametrai p, d ir q turi tokias reikšmes:

- p yra autoregresyvių išraiškų skaičius,
- d yra ne sezoninių diferencijacijų skaičius, kurio reikia stacionarumui pasiekti,
- q parodo kiek prognozės lygtyje bus atsiliekančių prognozės paklaidų.

Taigi sukonstruokime prognozavimo lygtį. Pirmiausia raide y pažymėkime Y skirtumą, kuris reiškia:

Jei d=0: $y_t = Y_t$

Jei d=1: $y_t = Y_t - Y_{t-1}$

Jei d=2: $y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2}) = Y_t - 2Y_{t-1} + Y_{t-2}$

Verta pastebėti, kad antras Y nėra skirtumas pieš 2 periodus buvusia reikšme. Teisingiau tai yra pirmo skirtumo skirtumas. Tai atitinka antros eilės išvestinę.

Bendroji y prognozavimo lygtis yra užrašoma taip:

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} - \theta_1 e_{t-1} - \ldots - \theta_q e_{t-q} \qquad \textbf{(1.25)}$$

Čia slenkančio vidurkio parametras θ yra apibrėžiamas taip, kad jo ženklai yra neigiami. Taip yra dėl taisklės, kurią išrado Box ir Jenkins. Kai kurie autoriai ir programos (taip pat ir „R“) šiuos koeficientus laiko teigiamais. Kai naudojame realius skaičius ši dviprasmybė išnyksta, tačiau yra svarbu prisiminti, kad reikia pasitikrinti, kurią formą jūsų programa naudoja kai jūs skaitote išvestis.

1.7.“BOX-COX“ TRANSFORMACIJA

Dauguma statistinių testų ir intervalų remiasi normalumo prielaida. Normalumo prielaida dažnai lemia daug paprastesnius testus, aiškesnį matematinį interpretavimą ir stiprumą, priešingai nei testai neturintys normalumo prielaidos. Deja daugelis realių duomenų nėra bent apytiksliai pasiskirstę pagal normalųjį dėsnį. Tačiau panaudojus tinkamą transformaciją galime gauti duomenų rinkinį, kuris yra labai artimas normaliai pasiskirsčiusiam duomenų rinkiniui. Tai padidina normalumo prielaida paremtų testų tinkamumą ir naudingumą.

Box – Cox transformacija yra labai naudinga transformacijų šeima. Minėtoji transformacija yra apibrėžiama taip:

$$T(Y)=(Y\lambda-1)/\lambda \qquad \textbf{(1.26)}$$

čia Y yra atsako kintamasis, o λ yra transformacijos parametras. Kai $\lambda = 0$, yra tiesiog randamas duomenų logaritmas ir nenaudojama aukščiau esanti formulė. Turint konkrečią Box – Cox transformaciją, kokia yra apibrėžta aukščiau yra pravartu apsibrėžti rodiklį, kuriuo bus galima išmatuoti po transformacijos pasikeitusių duomenų normalumą. Vienas iš būdų yra apskaičiuoti normalaus tikimybinio grafiko, koreliacijos koeficientus. Koreliacija gali būti skaičiuojama tarp kintamojo esančio vertikalioje grafiko ašyje ir tarp kintamojo esančio horizontalioje grafiko ašyje. Toks koreliacijos rodiklis yra labai patogus tiesiškumo įvertis.

1.8. NEURONINIŲ TINKLŲ AUTOREGRESIJA

Kai turime laiko eilutės duomenis tai galime praeities periodų reikšmes panaudoti kaip neuroninio tinklo įvestis. Taip kaip praeities reikšmės yra naudojamos tiesinės autoregresijos modelyje, taip pat jas galime panaudoti ir neuroninių tinkle autoregresijoje.

Apžvelgsiu tik neuroninius tinklus su vienu paslėptu neuronų sluoksniu ir naudosiu žymėjimą NNAR(p,k), kuris reiškia, kad modelį sudaro p praeities reikšmių ir k mazgų paslėptame sluoksnyje. Pavyzdžiui, modelis NNAR(9,5) yra neuroninis tinklas su paskutinėmis devyniomis reikšmėmis (y_{t-1},y_{t-2},…,y_{t-9}(y_{t-1},y_{t-2},…,y_{t-9}), kurios yra naudojamos kaip įvesties reikšmės prognozuoti y_t ir su 5 neuronais paslėptame sluoksnyje. Modelis NNAR (p,0) yra ekvivalentus ARIMA(p,0,0) modeliui tik be stacionarumą užtikrinančių parametrų apribojimų. Sezoniniams duomenims yra naudinga pridėti vienu sezonu atsiliekantį stebėjimą. Pavyzdžiui modelis NNAR$(3,1,2)_{12}$ turi įvestis y_{t-1}, y_{t-2}, y_{t-3} ir y_{t-12} bei 2 neuronus paslėptame sluoksnyje. Kitaip tariant modelis NNAR(p,P,k)m turi įvestis (y_{t-1},y_{t-2},…,yt−p,yt−m,y_{t-2m},y_{t-Pm}) ir k neuronų paslėptame sluoksnyje. Modelis NNAR(p,P,o)m yra ekvivalentus ARIMA(p,0,0)(P,o,o)m modeliui tik be stacionarumą užtikrinančių parametrų apribojimų.

Funkcija nnetar() pritaiko NNAR(p,P,k)m modelį. Jei p ir P reikšmių nenurodome jos automatiškai yra parenkamos. Ne sezoninėms laiko eilutėms numatytosios reikšmės yra lygios optimaliam praeities reikšmių skaičiui (remiantis AIC kriterijumi) tiesiniam AR(p) modeliui. Sezoninėms laiko eilutėms numatytosios reikšmės yra: P = 1, o p parenkama iš optimalaus tiesinio modelio pritaikyto sezoniškai paveiktiems duomenims. Jei k nėra apibrėžta ji apskaičiuojama pagal tokią formulę:

$$k = (p + P + 1)/2 \tag{1.27}$$

Čia k yra suapvalintai iki artimiausio sveiko skaičiaus.

1.9. KITŲ PASAULYJE TAIKOMŲ PROGNOZAVIMO METODŲ APŽVALGA

Vektoriniai laiko eilučių metodai yra naudojami tiriant daugiamates laiko eilutes. Šis metodas nuo vienmačių laiko eilučių skiriasi tuo, kad kintamieji yra įvertinami naudojant tiek savo, tiek kitų kintamųjų praeities rezultatus. Galime sudaryti modelį tiek tarp priklausomų, tiek tarp nepriklausomų kintamųjų. Taip daug efektyviau panaudojame sąryšius tarp skirtingų kintamųjų, sumažinamas prielaidų skaičius bei panaudojama platesnė informacija. Vektorinės autoregresijos modelyje esantys kintamieji yra aprašomi naudojant jų pačių ir kitų sistemos kintamųjų vėlavimų tiesinės regresinės funkcijos. p – eilės vektorinė autoregresija žymime VAR(p) ir matematiškai išreiškiame tokiu būdu:

$$y_t = c + A_i y_{t-i} + \epsilon_t \quad \textbf{(1.28)}$$

Čia y_t - n dimensijos kintamųjų vektorius, c – n dimensijos konstantų vektorius, A_i - kvadratinės $n \times n$ paklaidų vienalaikių kovariacijų ir parametrų matricos, kai $i = 1, \dots, p$, p – autoregresijos eilė, ϵ_t – baltojo triukšmo paklaidų vektorius.

Kiekvienas VAR(p) turi kelis savo pavidalus, pavyzdžiui: VAR(1) pavidalu naudojant kintamųjų peržymėjimą. Taip pat be esminio informacijos praradimo galima išanalizuoti paprastesnį modelį. Jeigu turime bendrą VAR(p) modelio išraišką, ją galima užrašyti žemiau pateikta forma:

$$\tilde{y}_t = \tilde{c} + \tilde{A}_1 \tilde{y}_{t-1} + \tilde{\epsilon}_t \quad \textbf{(1.29)}$$

Čia $\tilde{y}_t = \begin{pmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-p+1} \end{pmatrix}$, $\tilde{c} = \begin{pmatrix} c_t \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, $\tilde{\epsilon}_t = \begin{pmatrix} \epsilon_t \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, $\tilde{A} = \begin{pmatrix} A_1 & A_2 & \cdots & \\ I & 0 & \cdots & \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & & 0 \end{pmatrix}$.

Šioje išraiškoje galime pastebėti, kad pirmoji vektorių ir matricų eilutė yra ekvivalenti VAR(p), o likusieji nariai užtikrina kairės ir dešinės formos lygybės adekvatumą.

2. TYRIMŲ REZULTATAI IR JŲ APTARIMAS

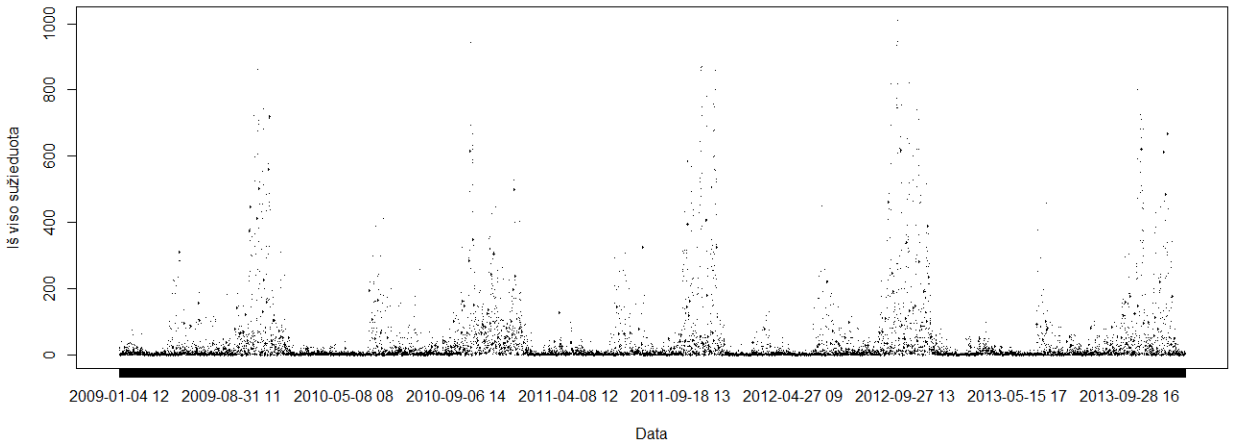
2.1. DUOMENŲ APDOROJIMAS IR VIZUALIZACIJA

Taigi turime du failus, kuriuose yra mums reikalingi duomenys. Darbo pradžioje turime sujungti šiuos duomenų rinkinius į vieną. Tą padarysime naudodami programą „Microsoft Excel“ ir jos funkciją *Vlookup.* Sujungus šiuos du duomenų masyvus, gautas rezultatas atrodys taip:

eilnr	data	Temp	Precipitation	Wind_spd	Wind_di	viso
1	2009-01-04 12	-8.5	0	2	51	4
2	2009-01-05 10	-10.4	0	3.3	145	1
3	2009-01-05 12	-8.6	0	3.9	152	4
4	2009-01-05 13	-7.9	0	4.2	151	14
5	2009-01-05 14	-7.6	0	4.2	159	24
6	2009-01-05 15	-7.4	0	4.5	167	20
7	2009-01-05 16	-7.3	0	4.8	171	3
8	2009-01-06 09	-0.3	0	8.5	240	1

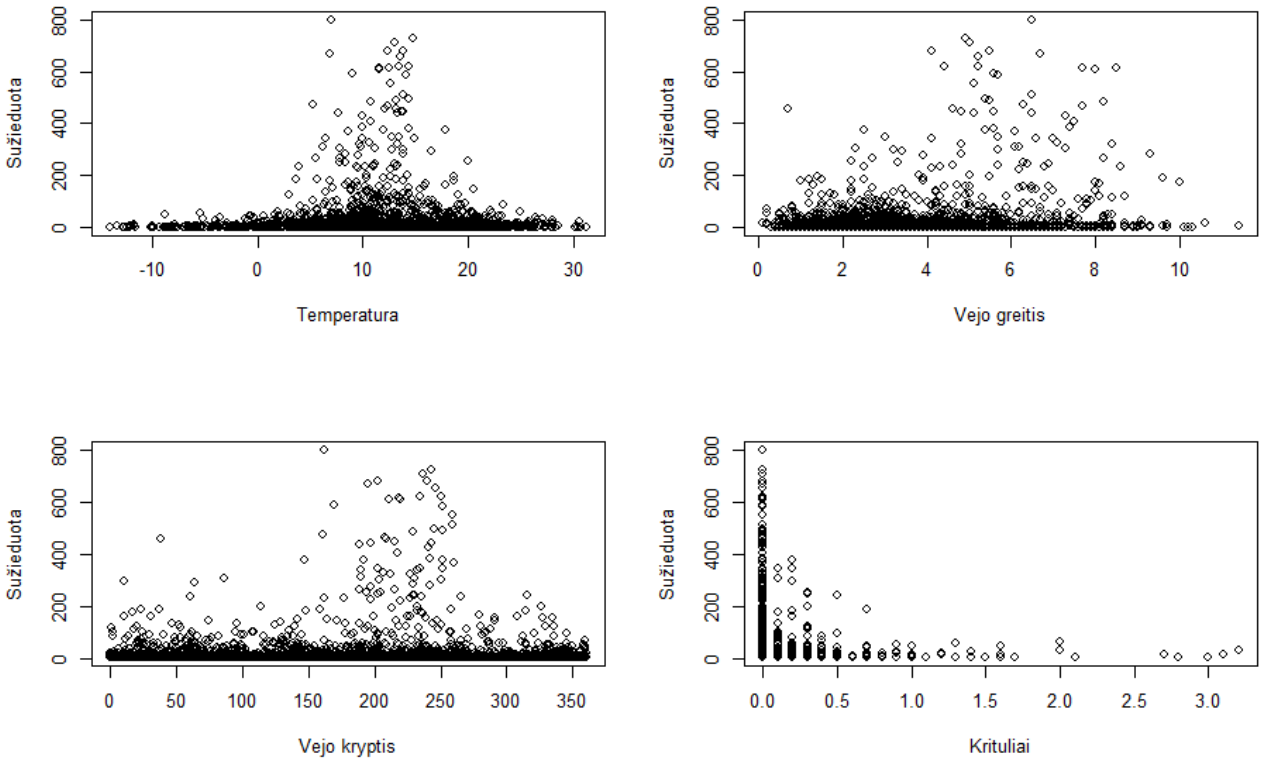
2.1 pav. Duomenų fragmentas

Toliau galime pavaizduoti turimus duomenis grafiškai, kad galėtume geriau susipažinti su jų pasiskirstymu laike.



2.2 pav. Duomenys atvaizduoti grafiškai

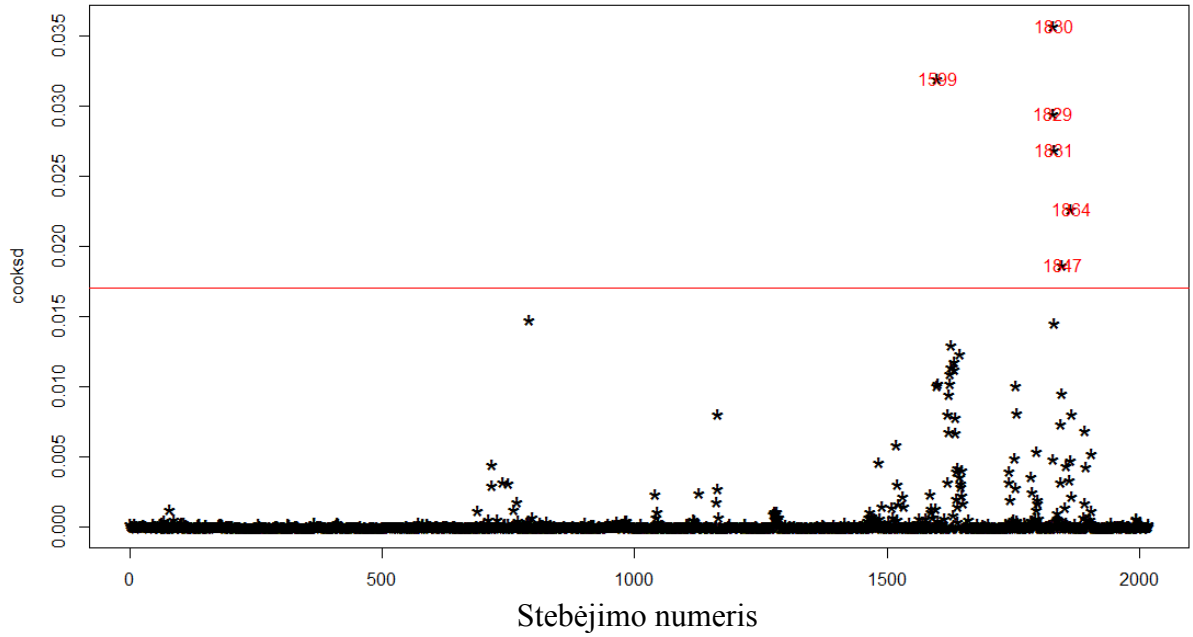
Iš šio grafiko galime matyti, kad duomenys yra stipriai paveikti sezoniškumo faktoriaus. Tai galime pagrįsti tuo, kad paukščių migracija suaktyvėja gegužės, birželio ir rugsėjo, spalio mėnesiais. Žiemos mėnesiais migracija pasidaro žymiai pasyvesnė. Toliau, bandysime susidaryti aiškesnį vaizdą apie nepriklausomus kintamuosius, kuriais remiantis bandysime prognozuoti priklausomą kintamąjį t.y. sužieduotų paukščių skaičių.



2.3 pav. Priklausomo kintamojo priklausomybė nuo nepriklausomų

2.2. REGRESINIO MODELIO KŪRIMAS

Prieš pradėdami regresinio modelio kūrimą iš imties bandysime pašalinti išskirtis. Pašalinus išskirtis turėtume sukurti tiksliau prognozuojantį modelį. Išskirtys yra viena iš daugelio problemų į kurią turime atsižvelgti kurdami regresinius modelius. Taip pat svarbu įsitikinti, kad turimoje imtyje nėra trūkstamų ar klaidingų reikšmių. Išskirtims nustatyti naudosime Kuko atstumą (Cooks distance). Kadangi sužieduotų paukščių skaičius koncentruojasi aplink žemas vertes ir negalima būti neigiamas, galima iškart nuspėti kad išskirtys bus tik per didelės reikšmės. Kad modelis kuo geriau atitiktų realią situaciją pašalinsime tik labai stipriai išsiskiriančias reikšmes t.y reikšmes, kurių Kuko distancija viršys 0,016. Suskaičiavus Kuko distancijos reikšmes kiekvienam stebėjimui, jas grafiškai pavaizduojame naudodami *plot* funkciją. Kaip jau anksčiau minėjau pašalinsime tik stebėjimus, kurių Kuko distancijos reikšmė viršys 0,016. Kuko distancijų grafikas atrodo taip:



2.4 pav. Kuko distancija kiekvienam stebėjimui

Paveikslėlyje numeris 2.4, raudona linija žymi anksčiau pasirinktą tolerancijos ribą, kuri yra lygi 0,016. Tai gi matome, kad šią ribą viršija 6 stebėjimai. Skaičiai esantys ant taškų atitinkančių tuos stebėjimus reiškia stebėjimo numerį. Taigi stebėjimus, kurių eilės numeriai yra tokie:

- 1599
- 1829
- 1830
- 1831
- 1832
- 1847

- 1864.

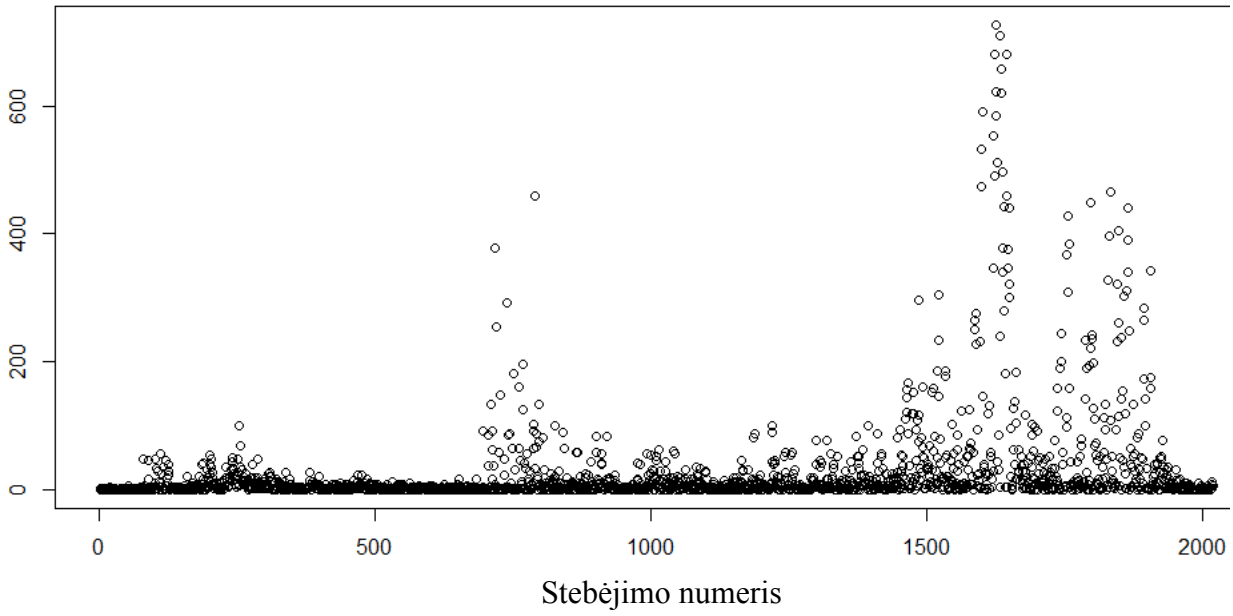
Surandame duomenų rinkinyje šiuos įrašus ir jų reikšmes pakeičiame šalia reikšmių vidurkiu. Taigi 1599 – tame įraše sužieduotų paukščių skaičius yra 800, o kaimyniniai įrašai turi reikšmes 473 ir 591, todėl 1599 – am įrašui priskiriame reikšmę:

(473 + 591)/2=532.

Toliau turime keturias iš eilės einančias išskirtis t.y 1829, 1830, 1831 ir 1832. Surandame šiems keturiems įrašams kaimyninius įrašus, kurių reikšmės yra lygios: 327 ir 108. Tada visiems keturiems stebėjimams priskiriame naujas reikšmes, kurias apskaičiuojame taip:

(327+108)/2=217.5

Atitinkamai šiuos veiksmus pakartojame su 1847 – tu ir 1867 – tu stebėjimu. Pataisyti paukščių žiedavimo duomenys atrodo taip:



2.5 pav. Imtis su pašalintomis išskirtimis

Iš paveikslėlio nr 2.5 galima nesunkiai pastebėti, kad tarp mūsų kintamųjų neegzistuoja stiprus tiesinis ryšys, todėl negalėsime taikyti tiesinės regresijos. Šiam duomenų rinkiniui pamėginsime sukurti atraminių vektorių regresijos modelį (*support vector regression).* Kad sukurtume atraminių vektorių modelį su "R" mums prireiks paketo *e1071*. Prieš pradedant darbą reikia nepamiršti parsisiųsti ar užsikrauti(jei jis jau buvo parsiųstas) šį paketą. Programoje "R" tai padaroma su komanda *library*. Pačiam modeliui sukurti naudosime funkciją *svm*. Į modelį įtrauksime 5 rodiklius:

1. Vėjo greitis
2. Oro temperatūra
3. Vėjo kryptis
4. Drėgnumas
5. Metų diena kai stebėjimas buvo fiksuotas

Modelio sukūrimo kodas atrodo taip:

```
model <- svm(viso ~ Temp+Wind_dir+Wind_spd+Precipitation+Met_Diena kernel= "radial", data)
```

Modelio tikslumui patikrinti naudosime vidutinę kvadratinę paklaidą (root mean square error(rmse)). Sukursime savo funkciją vidutinei kvadratinei paklaidai apskaičiuoti, ji atrodys taip:

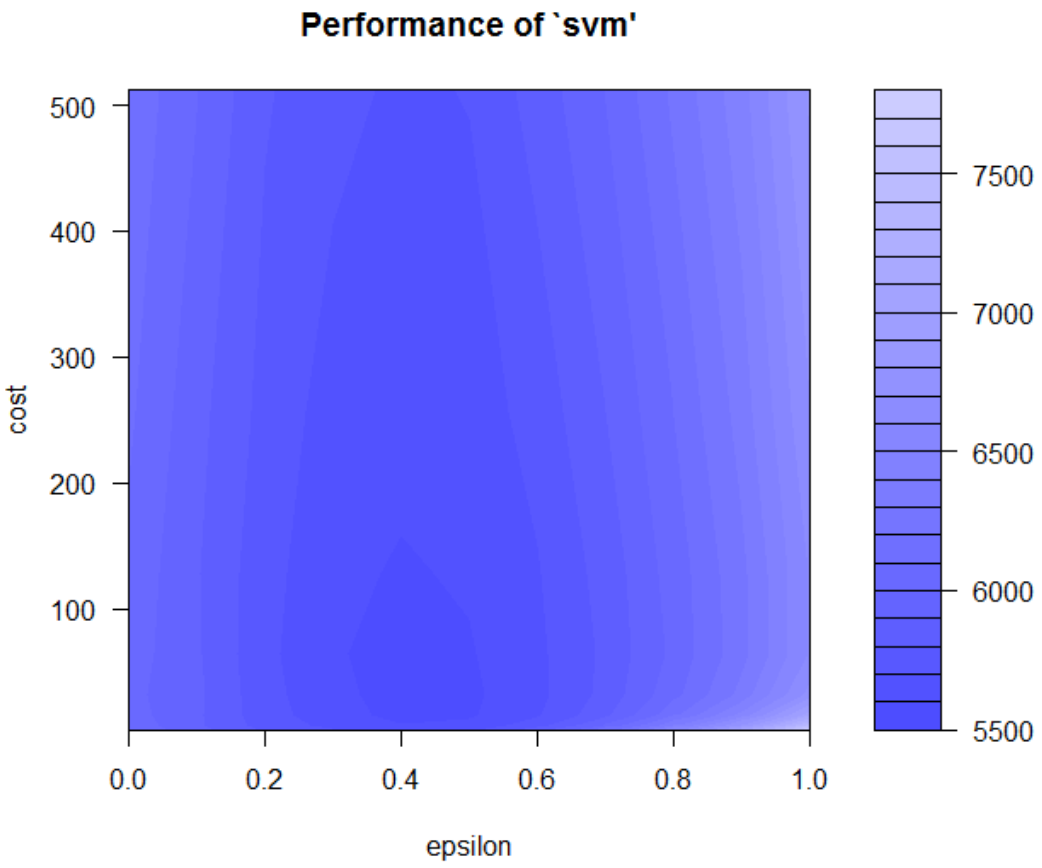
```
rmse <- function(error)

{  sqrt(mean(error^2))}
```

Pasinaudodami šia funkcija suskaičiuojame mūsų sukurt modelio paklaidą. Ji yra lygi: 68.48. Tokia paklaida gaunasi naudojant numatytuosius metodo parametrus. Ar galima paklaidą sumažinti? Tai padaryti bandysime parinkdami geresnius parametrus modeliui. Dabartiniame mūsų pavyzdyje mes atlikome epsilon regresija, nes nenustatėme jokios epsilon(ϵ) reikšmės, todėl regresija buvo sukurta naudojant reikšmę 0.1. Taip pat modelyje dar yra kainos parametras, kurį galima keisti norint išvengti persotinimo. Procesas, kurio metu mes parinksime šiuos parametrus yra vadinamas hyperparametrine optimizacija arba modelio parinkimu. Standartiškai tai atliekama darant tinklelio paiešką. Tai reiškia, kad mes apmokysime daug modelių skirtingoms epsilon ir kainos reikšmių kombinacijoms ir iš gautų rezultatų pasirinksime geriausią. Epsilon bandysime reikšmes iš intervale [0;1]. Reikšmės didės po 0.1. Kainai parinksime dvejeto laispnių reikšmes, kai laipsnio rodiklis kinta intervale: [1;9]. Tinklelio paieškos programinis kodas atrodo taip:

```
tuneResult<-tune(svm,viso~Temp+Wind_dir+Wind_spd+Precipitation+Metu_diena, data = data, ranges = list(epsilon = seq(0,1,0.1), cost = 2^(2:9))
```

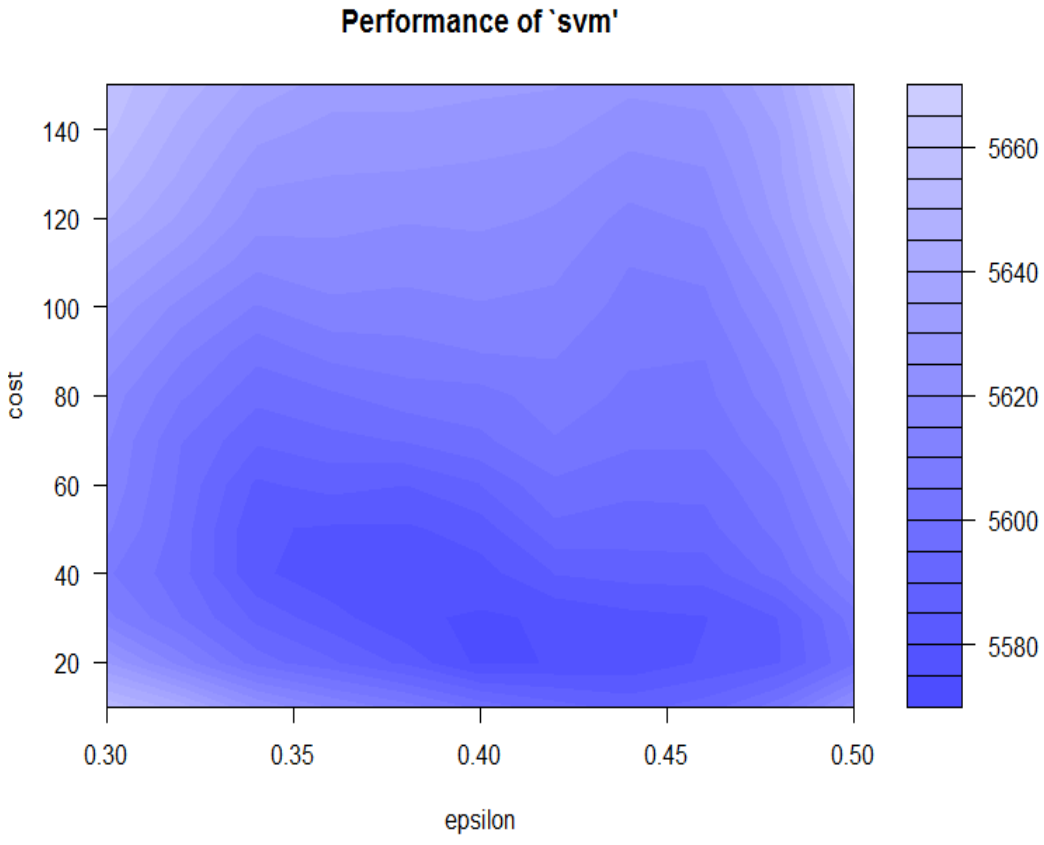
Gauti rezultatai parodė, kad mažiausia paklaida gaunama kai epsilon yra lygus 0.4, o metodo kaina lygi 80. Su tokiomis parametrų reikšmėmis vidutinė kvadratinė paklaida yra lygi 49.93. Primenu, kad su numatytaisiais parametrais vidutinė kvadratinė paklaida buvo 68.48. Taigi mes šiek tiek pagerinome savo modelį.

Žemiau esančiame paveikslėlyje 2.6 galime pamatyti grafiškai atvaizduotą tinklelio paieškos rezultatą. Šiame paveikslėlyje galima matyti, kad tuo regionas yra tamsesnis, tuo geresnis mūsų modelis (nes rmse yra arčiau nulio tamsesniuose regionuose). Tai reiškia, kad galime pabandyti dar vieną tinklelio paiešką, tik šį kartą ją atliksime siauresniame intervale. Paveikslėlyje tamsiausią spalvą matome kai epsilon yra tarp 0.3 ir 0.5, o metodo kaina tarp 20 ir 120.



2.6 pav. Tinklelio paieškos grafinis vaizdas

Taigi sekančiame etape susiaurinsime paiešką tamsesnėje. Gautas rezultatas atrodo taip:



2.7 pav. Siauresnės tinklelio paieškos grafinis vaizdas

Susiaurinta paieška parodė, kad mažiausia paklaidą gauname kai epsilon yra 0.43, o metodo kaina 35. Įstačius šias reikšmes vidutinę kvadratinę paklaidą gauname 46.96. Tai gan nemažas pokytis nuo pradinės vertės, kuri buvo: 68.48.

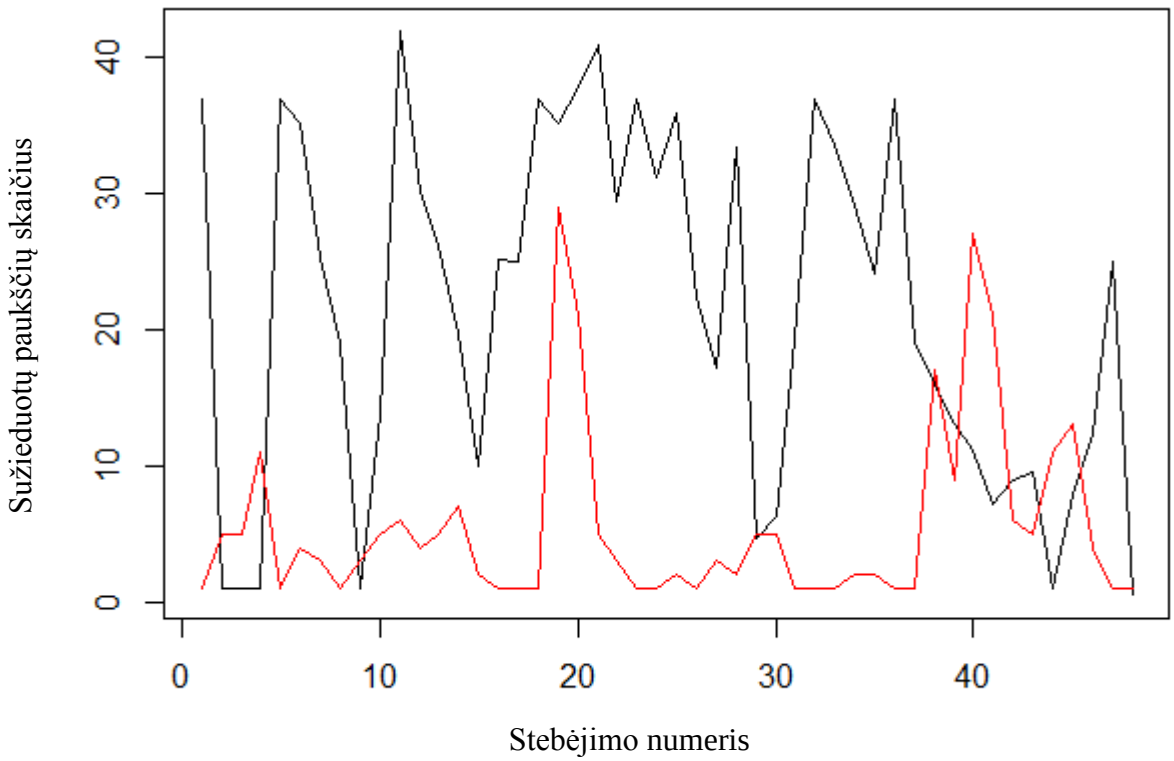
Norint dar atidžiau ištirti šį metodą reikia jam atlikti kryžminį patikrinimą. Tai yra apmokyti modelius 2009 – 2010 metams patikrinti su 2011 metais, apmokyti 2009 – 2011 metams patikrinti su 2012 metais ir apmokyti 2009 – 2012 metams, tada patikrinti su 2013 metais. Šį tikrinamą su pilnu metų periodu, bet pridėsime ir dar kelis trumpesnius periodus, kad galėtume geriau įvertinti kaip kinta paklaidos. Visą tai atlikus apskaičiuoti vidutines paklaidas. Žemiau galite matyti gautus rezultatus:

2.1 lentelė

Modelio paklaida po kryžminio patikrinimo

	RMSE
2 paros	28.48
1000 valandų	38.68
Metai	83.58

Taigi atlikę kryžminį patikrinimą suradome metodo paklaidas prognozuojant įvairius periodus. Neturint su kuo palyginti yra sunku vertinti šio modelio adekvatumą, todėl toliau darbe sukursime dar kelis prognozavimo modelius, su kuriais galėsime palyginti atraminių vektorių metodo rezultatą. Grafiškai gauta prognozė atrodo šitaip:



2.8 pav. Prognozė naudojant atraminių vektorių metodą

Čia raudona linija rodo tikrąją reikšmę, o juoda linija atliktą prognozę. Naudojant Shapiro - Wilk statistiką patikriname ar paklaidos pasiskirsčiusios pagal normalųjį skirstinį. Gauname, kad $p = 0.06$. Taigi 95% tikimybe galime teigti, kad paklaidų skirstinys yra normalus arba gana arti jo.

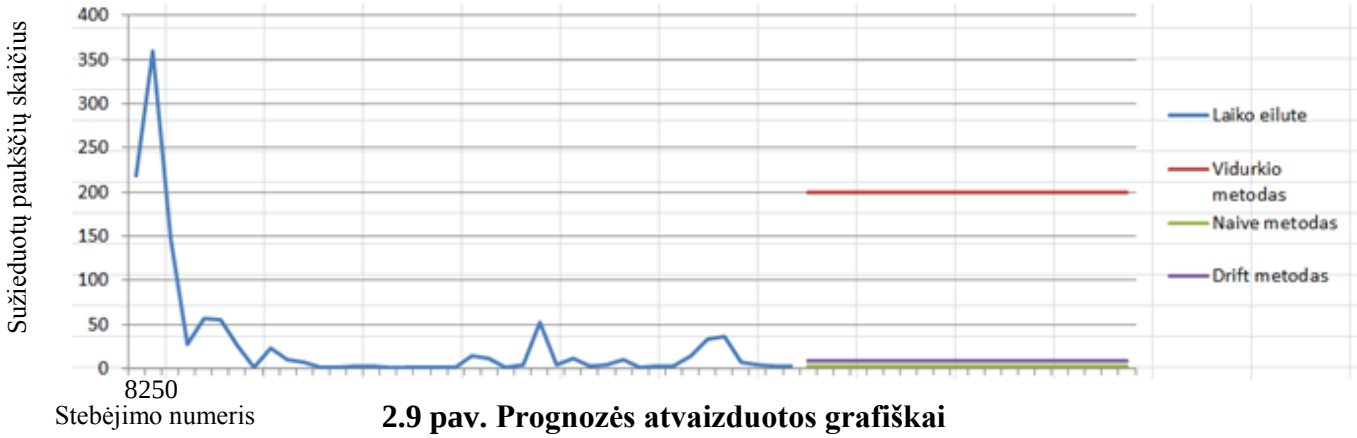
2.3. LAIKO EILUČIŲ MODELIO KŪRIMAS

Toliau pabandysime sukurti paukščių migracijos intensyvumo prognozavimo modelį naudojant laiko eilučių analizės metodus. Tirdami laiko eilučių analizės metodus naudosime tuos pačius paukščiu žiedavimo duomenis.

Iš pradžių pradėsime nuo pačių paprasčiausių metodų. Tai darysime todėl, vėliau turėtume su kuo palyginti kitus metodus. Tai gi trys patys paprasčiausi metodai yra:

- *Naive* (naivus) metodas yra pats paprasčiausias jis, prognozuojamam skaičiui naujų reikšmių priskiria vienodas reikšmes, kurio yra lygios paskutinio stebėjimo reikšmei
- *Drift* (poslinkio) metodas sujungia pirmą ir paskutinę stebėjimo reikšmes ir naujas reikšmes gauna pratęsdamas tą liniją.
- *Mean* (vidurkio) metodas suskaičiuoja visos laiko eilutės vidurkį ir prognozuojamam skaičiui naujų reikšmių priskiria vienodas reikšmes, kurios yra lygios tam vidurkiui.

Taigi panaudodami paketą *forecast* ir jo funkcijas sudarome prognozavimo modelius turimai laiko eilutei su visais trimis aukščiau išvardintais metodais. Rezultatus galite matyti žemiau:



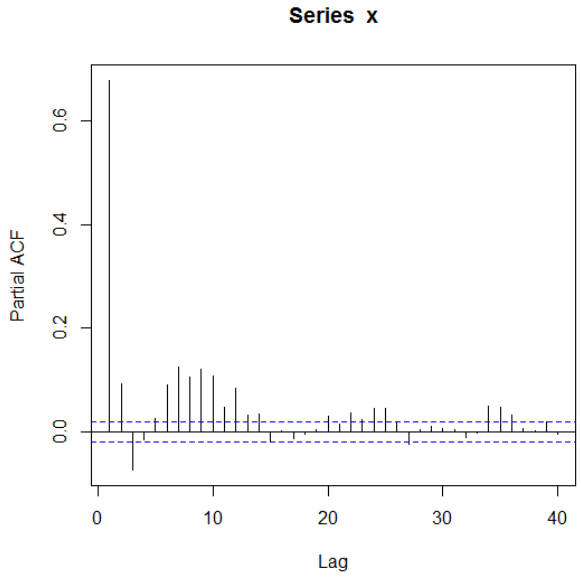
2.9 pav. Prognozės atvaizduotos grafiškai

Iš grafiko galima pastebėti, kad sudarytos prognozės nelabai tiksliai atitinka mūsų turimą laiko eilutę, tačiau kartais net šie paprasčiausi metodai duoda gan neblogus rezultatus. Toliau modelius, kuriuos suskursime lyginsime su šiais, jei jų rezultatai bus prastesni už šių trijų, nebus racionalu juos laikyti tinkamais. Tikslesniam šių modelių vertinimui surasime vidutinę kvadratinę paklaidą.

1. Naivaus metodo vidutinė kvadratinė paklaida: 70.54
2. Paslankaus metodo vidutinė kvadratinė paklaida: 70.56
3. Vidurkio metodo vidutinė kvadratinė paklaida: 87.96

Toliau išbandysime sudėtingesnius prognozavimo metodus. Vieni populiariausių metodų laiko eilučių prognozavimui yra: arima ir neuroninių tinklų. Šiuos metodus pritaikysime turimai laiko eilutei. Optimalius parametrus šiems metodams parinksime naudojant automatines funkcijas. Modelius vertinsime pagal Akaikės koeficientą(aicc). Kadangi arima modeliai yra naudojami tik stacionarioms

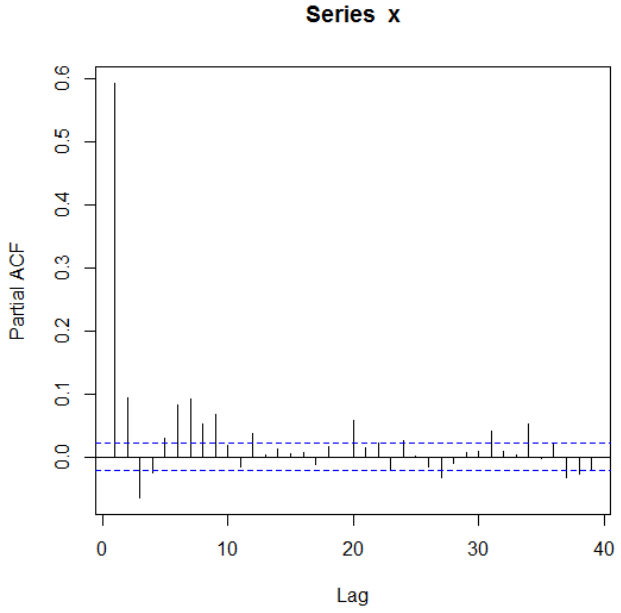
laiko eilutėms prognozuoti, iš pradžių ištirsiu turimos laiko eilutės stacionarumą. Stacionarumą vertinsiu naudodamas dalinę auto koreliaciją. Paukščių stebėjimų laiko eilutės dalinės auto koreliacijos grafikas atrodo taip:



2.10 pav. Dalinių autokoreliacijų grafikas

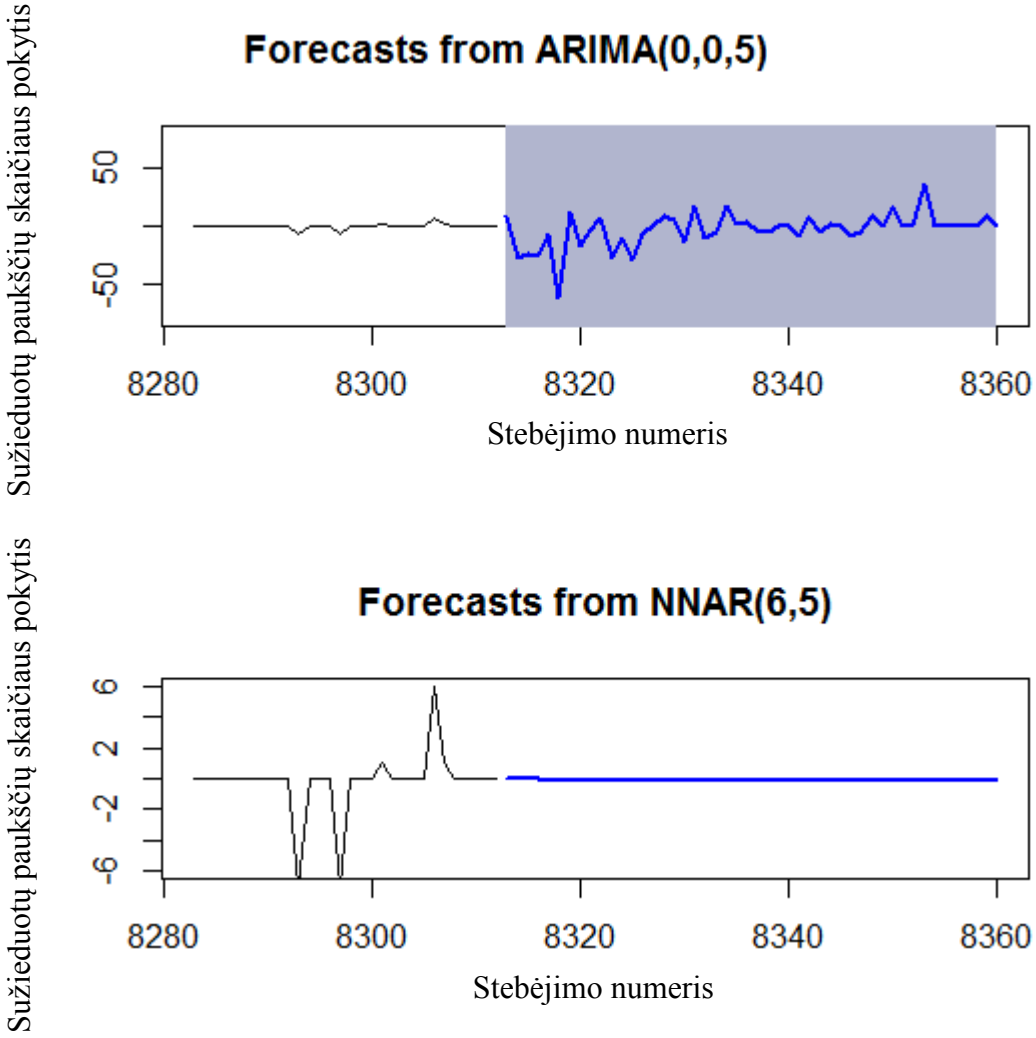
Paveikslėlyje numeris 2.10 matosi, kad gan daug PACF reikšmių išeina už leidžiamos zonos ribų. Kadangi paveikslėlyje numeris 2.2 matėme, kad turima laiko eilutė yra veikiama sezoniškumo pabadysime jai atlikti sezoninį diferencijavimą. Vienu sezonu laikysiu vienus metus. Taigi diferencijuojant eilutę iš pirmos vienų metų reikšmės atimsiu pirmą sekančių metų reikšmę ir t.t.

Po diferencijavimo gautos laiko eilutės dalinių auto koreliacijų grafikas atrodo taip:



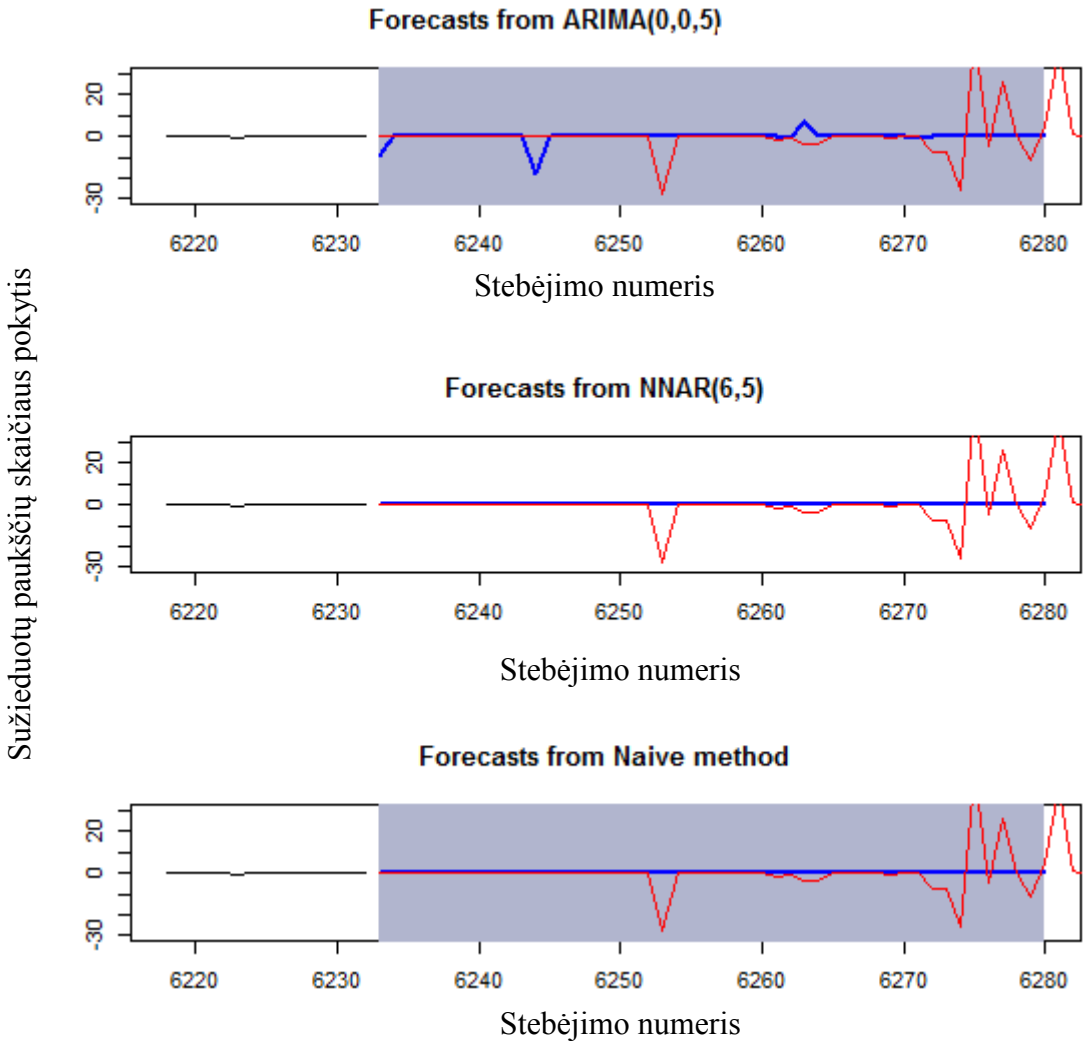
2.11 pav. Diferencijuotų reikšmių dalinių autokoreliacijų grafikas

Toliau prognozuosiu 48 valandų periodą, grafiškai atvaizduosiu jį ir dar paskutines 30 imties reikšmių.



2.12 pav. Prognozės pavaizduotos grafiškai

Iš gautų grafikų gan sunku nustatyti, kuris metodas geriausiai prognozuoja turimą laiko eilutę, todėl pabandysiu jų tikslumą įvertinti naudojant vidutinę kvadratinę paklaidą. Iš pradžių imtį padaliname į apmokymo imtį ir testinę imtį. Apmokymo imčiai priskiriame 2009, 2010, 2011 ir 2012 metus. Na, o testinė imtis susidarys iš įrašų užfiksuotų 2013 – aisiais metais. Tada pritaikome anksčiau tris pavaizduotus metodus apmokomajai imčiai. Gautą rezultatą pavaizduojame grafiškai pridėdami vidurkio metodo grafiką, į kiekvieną grafiką įdėjau tikrąsias reikšmes, kurias žymi raudona linija.



2.13 pav. Prognozės atvaizduotos grafiškai

Žiūrint į grafikus atrodo, kad modeliai netiksliai atkartoja kreivės kryptį. Bet žinoma iš grafikų sunku nustatyti, kuris metodas teisingiausiai atspindi imtį, todėl apskaičiuosime, visų metodų paklaidas testinėje imtyje. Gauti rezultatai atrodo taip:

2.2 lentelė

Modelių paklaidos testinėje imtyje

	RMSE
Arima	10.495
Neuroninių tinklų	9.969
Naivus	9.976

Pagal RMSE paklaidą geriausias yra naivus metodas. Kiti metodai nepasiekia net paprasčiausio vidurkio metodo rezultato. Tačiau norint įsitikinti, kad nėra taip jog kuriam nors metodui tiesiog pasisekė, patikrinkime modelius ir su apmokymo imtimi.

2.3 lentelė

Modelių paklaidos apmokymo imtyje

	RMSE
Arima	105.77
Neuroninių tinklų	71.62
Naivus	85.81

Galima matyti, kad neuroninių tinklų metodas parodo geriausią RMSE rodiklį apmokymo imtyje, tačiau testinėje imtyje šio modelio rezultatas buvo labai blogas. Tai reiškia, kad modelis per daug prisitaikė prie apmokymo imties. Apmokymo imties paklaidomis reikėtų mažiau pasikliauti, būtent dėl anksčiau paminėtos priežasties. Kai kurie metodai puikiai atkartoja apmokomąją imtį, tačiau rodo prastus rezultatus, kai juos tikriname testinėje imtyje. Lentelėje 2.3 taip pat matome, kad naivus metodas apmokomojo imtyje irgi atrodo gan prastai, taip greičiausiai yra dėl to nes testinė imtis buvo daug mažesnė už apmokomąją. Paprasčiausi metodai dažniausiai atrodo tuo geriau kuo trumpesnį periodą prognozuojame. Norint dar atidžiau ištirti šiuos metodus reikėtų atlikti jiems kryžminį patikrinimą. Tai yra apmokyti modelius 2009 – 2010 metams patikrinti su 2011 metais, apmokyti 2009 – 2011 metams patikrinti su 2012 metais. Papildomai patikrinsiu kokias paklaidas modelis demonstruoja įvairaus ilgio laikotarpiuose. Pasirinkau anksčiau minėta metų periodą bei dar pridėsiu 1000 valandų ir dviejų parų periodus. Tada paskaičiuosiu vidutines paklaidas, kiekviename periode.

2.4 lentelė

Modelių paklaidos testinėje imtyje atlikus kryžminį patikrinimą

	RMSE		
	2 paros	1000 valandų	Metai
Arima	10.506	35.442	111.706
Neuroninių tinklų	15.743	35.485	111.481
Naivus	10.883	35.232	111.461

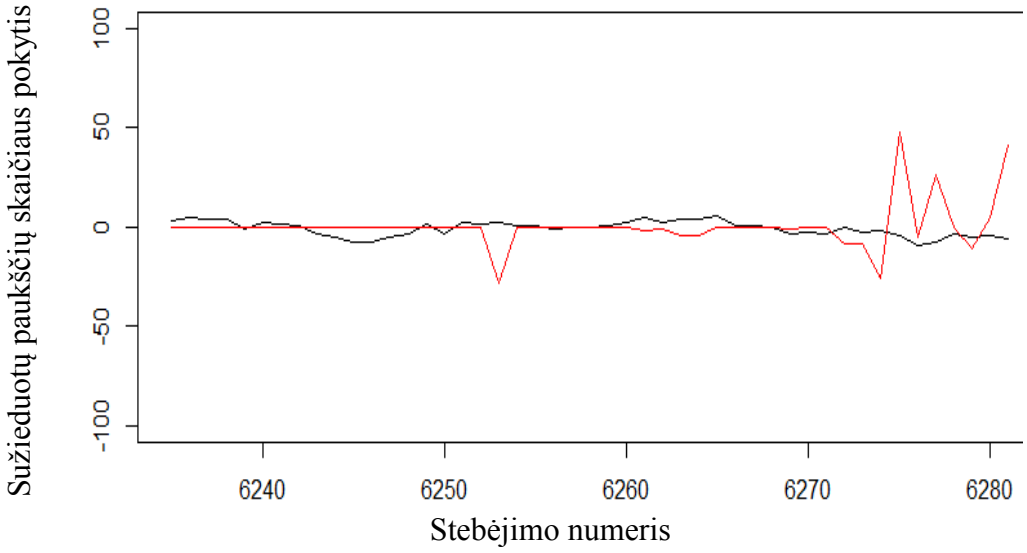
Taigi matome, kad prognozuojant metų periodą arima modelis net nelenkia paprasto laiko eilutės vidurkio metodo. Todėl galime daryti išvadą, kad labai ilgam laikotarpiui prognozuoti mūsų sukurtas modelis nėra labai tinkamas. Dėl šios priežasties pabandysime jį patobulinti pridedant orų stebėjimo duomenis. Arima modelis su papildomais kintamaisiais, kurie daro įtaką laiko eilutės reikšmei yra vadinamas ARIMAX modeliu. Taigi sukuriame ARIMAX modelį į jį įtraukdami dar ir oro sąlygų duomenis. Šio modelio taip pat kaip ir kitų paklaidas apskaičiuosime prognozuodami tokį patį skaičių periodų ir atlikdami kryžminį patikrinimą. Gautus rezultatus matote žemiau pateiktoje lentelėje:

2.5 lentelė

Visų modelių paklaidos testinėje imtyje atlikus kryžminį patikrinimą

	RMSE		
	2 paros	1000 valandų	Metai
Arima	10.506	35.442	111.706
Neuroninių tinklų	15.743	35.485	111.481
Naivus	10.883	35.232	111.461
Arimax	11.66	35.38	111.45
SVM	28.48	38.68	83.58

Taigi lentelėje galime matyti, kad prognozuojant 2 parų periodą geriausią rezultatą pademonstruoja arima prognozavimo metodas tai reiškia, kad galime teigti jog jeigu norime sužinoti kiek paukščių bus sužieduota po dviejų dienų tą pačią valandą tai geriausiai gali nuspėti Arima modelis. Vidutiniame laikotarpyje geriausią rezultatą pademonstravo paprasčiausias naivus prognozavimo modelis. Na o prognozuoti visų metų duomenis geriausiai tinka atraminių vektorių modelis.



2.14 pav. Prognozė Arimax metodu

Naudojant Shapiro - Wilk statistiką patikriname ar paklaidos pasiskirsčiusios pagal normalųjį skirstinį. Gauname, kad $p = 4.907*10^{-11}$. Taigi 95% tikimybe galime teigti, kad paklaidų skirstinys nėra normalus.

IŠVADOS

Šio darbo metu susipažinau su daug įvairių programos „Excel“ funkcijų, kurios yra duomenų transformacijoms atlikti. Išbandžiau ir pritaikiau daug programos „R“ paketų bei jiems priklausančių funkcijų. Darbo metu susipažinau su paukščių migracijos valandinių reikšmių dinamika, oro sąlygų stebėjimų duomenimis. Darbo metu buvo atlikta imties išskirčių šalinimas, naudojant Kuko distancijos charakteristiką. Pašalinus išskirtis sukūriau penkis metodus sužieduotų paukščių skaičiui prognozuoti. Sukūriau atraminių vektorių modelį, kuris yra paremtas kintamųjų regresija. Tada sukūriau arima, neuroninių tinklų ir paprastą naivųjį modelį. Šie trys metodai priklauso laiko eilučių analizės metodų klasei. Po to prie arima modelio pridėjus oro stebėjimų valandinius duomenis buvo gautas arimax modelis. Palyginus visų šių penkių modelių vidutines kvadratines paklaidas, trijuose perioduose, buvo nustatyta, kad jeigu norime sužinoti kiek paukščių bus sužieduota po dviejų dienų tą pačią valandą tai geriausiai gali nuspėti Arima modelis su pridėtomis oro sąlygomis. Vidutiniame laikotarpyje geriausią rezultatą pademonstravo paprastas, naivus modelis. Na o prognozuoti visų metų duomenis geriausiai tinka atraminių vektorių modelis.

LITERATŪRA

1. https://rokka.shinyapps.io/shinyweatherdata/
2. http://kernelsvm.tripod.com/
3. http://forecastingsolutions.com/arima.html
4. https://www.otexts.org/fpp/7/7
5. https://stats.stackexchange.com/questions/54818/how-to-extract-compute-leverage-and-cooks-distances-for-linear-mixed-effects-mo
6. https://www.r-bloggers.com/basic-assumptions-to-be-taken-care-of-when-building-a-predictive-model/
7. http://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html
8. https://www.svm-tutorial.com/2014/10/support-vector-regression-r/
9. Kenneth P. Burnham, David R. Anderson. Understanding AIC and BIC in Model Selection / Sociological Methods & Research, Vol. 33, No. 2, 261 – 304 p.
10. Introduction to Time Series Forecasting Using SAS / ETS Software: Course Notes. SAS Institute Inc., Cary, nC 27513, USA. – 430 p.

1 PRIEDAS. PROGRAMOS KODAS

Arimax procedūra:

```
library(forecast)
data<- read.table("Galutinis.csv", sep=";", header=T)
data<-data[-c(8313:10390),]
data<-data[-c(4157:10390),]
test<-data[c(4157:6234),]
test<-test[-c(1001:2078),]
test<-test[-c(49:1000),]
xreg <- cbind(Temp=diff(data$Temp,2078),
              Precipitation=diff(data$Precipitation,2078),
          Wind_spd=diff(data$Wind_spd,2078),
Wind_dir=diff(data$Wind_dir,2078))
xreg_test <- cbind(Temp=test$Temp,
              Precipitation=test$Precipitation,
          Wind_spd=test$Wind_spd,
Wind_dir=test$Wind_dir)
lam <- BoxCox.lambda(data$viso)
diffed<-diff(data$viso,2078)
viso <- ts(diffed)
modArima <- auto.arima(viso, xreg=xreg,lambda=lam)
forcasted_arima <- forecast.Arima(modArima , h=14)
predictedY <- predict(modArima , newxreg=xreg_test)
rmse <- function(error)
{
  sqrt(mean(error^2))
}
error <- test$viso - predictedY$pred
svrPredictionRMSE <- rmse(error)
svrPredictionRMSE
plot(predictedY$pred)
```

SVM PROCEDŪRA

```
data<- read.table("one_year.csv", sep=";", header=T)
library(e1071)
```

```
model <- svm(viso ~ Temp+Wind_dir+Wind_spd+Precipitation+Menuo+Diena+Valanda,epsilon = 0.4, cost = 152, data=data)
model <- svm(viso ~ Temp+Wind_dir+Wind_spd+Precipitation+Menuo+Diena+Valanda,data=data,epsilon =0.39, cost = 152)
tuneResult <- tune(svm,viso ~ Temp+Wind_dir+Wind_spd+Precipitation+Menuo+Diena+Valanda,data=data, epsilon =0.32, cost = 142)
tuneResult <- tune(svm,viso ~ Temp+Wind_dir+Wind_spd+Precipitation+Menuo+Diena+Valanda,data=data, ranges = list(epsilon = seq(0.37,0.41,0.01), cost = seq(149,153,1)))
print(tuneResult)
plot(tuneResult)
ranges = list(epsilon = seq(0,0.1,0.1), cost = 2^(2:9))
 summary(model)
predictedY <- predict(model, data)
par(mfrow=c(2,2))
 plot(predictedY)
plot(data$Temp,data$viso, xlab="Temperatura", ylab="Su¸ieduota")
plot(data$Wind_spd,data$viso, xlab="Vejo greitis", ylab="Su¸ieduota")
plot(data$Wind_dir,data$viso, xlab="Vejo kryptis", ylab="Su¸ieduota")
plot(data$Precipitation,data$viso, xlab="Krituliai", ylab="Su¸ieduota")
points(data$viso, predictedY, col = "red", pch=4)
rmse <- function(error)
{
  sqrt(mean(error^2))
}
error <- data$viso - predictedY
svrPredictionRMSE <- rmse(error)
svrPredictionRMSE
data$Precipitation<-log(data$Precipitation)
data1<- read.table("Data1.csv", sep=",", header=T)
data2<- read.table("Data2.csv", sep=",", header=T)
data$Wind_dir<-as.factor(data$Wind_dir)
```

```
data.lm <- lm( viso ~ Wind_dir + Wind_spd + Temp + Precipitation+Menuo+Diena+Valanda, data=data)
summary(data.lm)
data.lm2 <- lm( viso ~ Wind_spd + Precipitation, data=data)
summary(data.lm2)
plot ( resid ( data.lm )~ fitted ( data.lm ) , xlab ="
Fitted values ", ylab =" Residuals ", main = "
Original Data ")
par(mfrow=c(2,2))
cooksd <- cooks.distance(mod)
plot(cooksd, pch="*", cex=2, main="Influential Obs by Cooks distance")  # plot cook's distance
abline(h = 0.004, col="red")
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>0.004,names(cooksd),""), col="red")
influential<-[4*mean(cooksd, na.rm=T))])0
plot(density(data$viso));plot(density(data$Wind_spd));plot(density(data$Precipitation));
plot(density(data$Temp))
shapiro.test(data$Wind_dir); shapiro.test(data$Wind_spd); shapiro.test(data$Precipitation); shapiro.test(data$Temp)
```

Laiko eilučių procedūros:

```
data<- read.table("dieniniai.csv", sep=",", header=T)
#library("strucchange")
#library("tsoutliers")
library("forecast")
plot(data$viso,range = 100)
drift<-rwf(data$viso, 20, drift=TRUE,lambda=0.1)
plot(tail(drift$x,30), xlim=c(0, 1650), ylim=c(0,300))
drift<-rwf(data$viso, 20, drift=TRUE,lambda=0.1)
mean.fit<-meanf(data$viso,h=30)
naive.fit<-rwf(data$viso,h=30)
drift.fit<-drift$fitted

fit<-ets(data$viso)#Auto
seas_exp<-ses(data$viso, h=30, alpha=0.1, initial="simple")#seasonal exponential
lam <- BoxCox.lambda(dat$viso) # = 0.131
```

```
fit <- ets(data$viso, additive=TRUE, lambda=lam)
x<-dat$viso
mod_arima <- auto.arima(x, ic='aicc', stepwise=FALSE, lambda=lam)
mod_exponential <- ets(x, ic='aicc', restrict=FALSE, lambda=lam)
mod_neural <- nnetar(x, p=12, size=25)
par(mfrow=c(3, 1))
plot(forecast(mod_arima, 14), include=20)
plot(forecast(mod_exponential, 14), include=20)
plot(forecast(mod_neural, 14), include=20)

dat <- read.csv("dieniniai.csv", sep=",", header=T)
x <- ts(dat$viso, frequency=326)
test_x[1305:1630] <- x[1305:1630]
test_x[1:1305] <- NA
#test_x2<-dat$viso[979:1078]
#test_x2<-dat$viso[653:752]
test_x2<-dat$viso[1305:1404]
#x<-x[1:978]
#x<-x[1:652]
x<-x[1:1304]

lam <- BoxCox.lambda(dat$viso)
models <- list(
  mod_arima = auto.arima(x, ic='aicc', stepwise=FALSE,lambda=lam),
  mod_exp = ets(x, ic='aicc', restrict=FALSE, lambda=lam),
  mod_neural = nnetar(x, p=6, size=5),
  mod_sts = StructTS(x)
  )
forecasts <- lapply(models, forecast, 100)
forecasts$naive <- naive(x, 100,lambda=lam)
par(mfrow=c(3, 2))
for(f in forecasts){
  plot(f,include=15)
  lines(test_x, col='red')
}
```

```
acc <- lapply(forecasts, function(f){
  accuracy(f, test_x2)[2,,drop=FALSE]
})

acc <- Reduce(rbind, acc)
row.names(acc) <- names(forecasts)
acc <- acc[order(acc[,'MASE']),]
round(acc, 3)

acc <- lapply(forecasts, function(f){
  accuracy(f, test_x2)[1,,drop=FALSE]
})
acc <- Reduce(rbind, acc)
row.names(acc) <- names(forecasts)
acc <- acc[order(acc[,'MASE']),]
round(acc, 2)
```