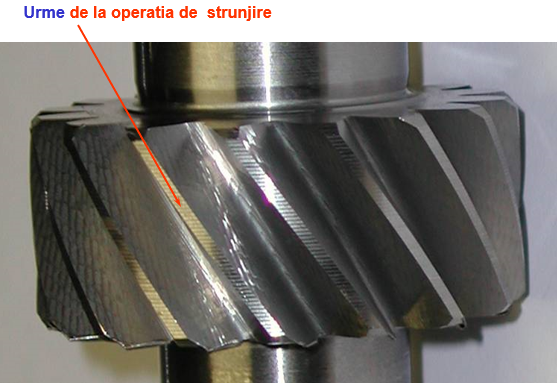
**Nivelul de zgomotul provocat de transmisiile masinii**

Transmisia reprezinta procesul de propagare a puterii mecanice de la cutia de viteza prin planetare catre puntea fata. Una din sursele de zgomot in acest proces de propagare a puterii mecanice este cutia de viteza. Cutia de viteza reprezinta un ansamblu de lanturi cinematice (multitudine de pinioane) format din: arbore primar, arbore secundar, baladoare (responsabile cu schimbarea pinioanelor in timpul procesului de schimbare de viteza) si sincroane.

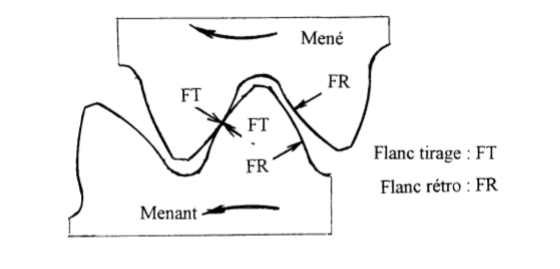
Principala sursa de zgomot in cutia de viteza este reprezentata de dantura neconforma a pinioanelor. Neconformitatea danturii este data de:

* Profilul excentric (decalarea axei de origine) al pinionului sau al danturii.
* Dantura pinionului deformata.
* Diferenta cumulata a totalului de pasi neconformi.
* Rugozitate mare.



*Pinion cu dantura neconforma: rugozitate*

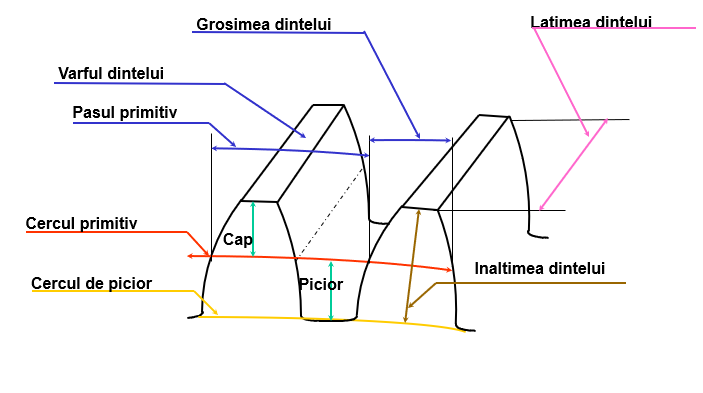
Nivelul de zgomot provocat de cutia de viteza, in speta de pinioane este masurat printr-un banc de ascultare care este destinat identificarii anumitor defecte acustice, generate de o neconformitate legata de dantura, care reprezinta riscul aparitiei zgomotului in cutia de viteza.



*Producere de zgomot datorata neconformitatii danturii pinioanelor*

Bancul de test simuleaza conditiile de angrenare ale cuplului de pinioane in cutia de viteza. Bancul de test determina printr-o interpretare subiectiva daca piesa ascultata genereaza zgomot cand este angrenata pe banc. Prin acest control se pot identifica mai multe surse de zgomot: socuri, diametrul de fund al dintelui neconform, erori de elice sau profil, erori de divizare, erori ale distantei dintre axe.

Parametrii geormetrici ai danturii pinionului ce influenteaza nivelul de zgomot din cutia de viteza si pentru care s-au semnalat neconformitati ale danturii ca: rugozitate mare, profil excentric , dantura deformata sunt :



*Parametrii geometrici ai danturii pinionului. Profil dantura*

ffα = abaterea de la forma conforma a profilului.

fHα = abaterea de la directia nominala a profilului sau abaterea de la unghiul de inclinare

al profilului.

Cα = bombaj profil dantura.

ffβ = abaterea de la forma conforma a elicei.

fHβ = abaterea de la directia nominala a elicei sau abatarea de la unghiul de inclinare al

elicei.

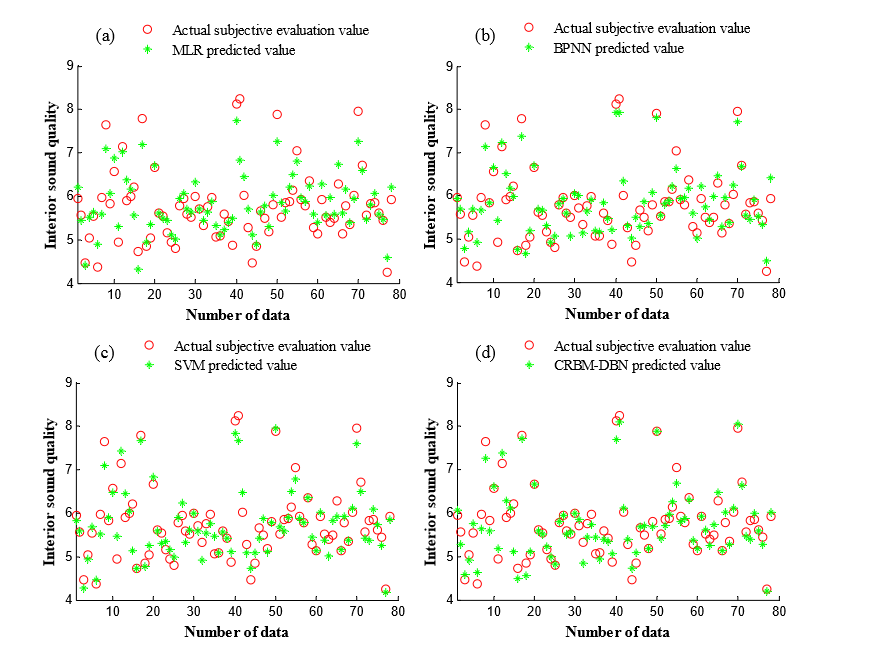
Cβ = bombaj elice.

**Retelele neuronale in evaluarea nivelului de zgomot din interiorul masinii**

De-a lungul timpului marii furnizori de autovehicule si companiile de automotive , si-au indreptat atentia catre digitalizarea vehiculelor , ajungand astazi in punctul in care vehiculul poate lua singur decizii in trafic , poate face o analiza a preferintelor muzicale ale soferului , poate fi controlat de la distanta si multe alte specificatii ce au propulsat rapid industria de automotive in top utilizatori ai inteligentei artificiale. Marile fabrici de autovehicule precum : Dacia-Renault , Ford , Volkswagen, implementeaza metode de digitalizare a liniilor de productie prin robotizare, software inteligent ( retele neuronale , invatare automata ) , renuntand astfel, treptat , acolo unde este cazul , la contributia umana, in productie si nu numai. Unul din aspectele importante asupra caruia s-au realizat de-a lungul timpului nenumarate studii pentru predictie , aproximare , diminuare sau evaluare , este calitatea sunetului NVH (zgomot , vibratie , asprime). Zgomotul produs de diferitele activitati mecanice , neconformitatea pieselor ce realizeaza diferite procese de transmitere a puterii mecanice, calitatea materialelor din care este alcatuit habitaclul masinii , sunt doar cateva din motivele ce pot fi luate in considerare cand se vorbeste despre sesizarea unui anumit nivel de zgomot in interiorul masinii.

Predictia calitatii sunetului in interiorul unui vehicul a suportat de-a lungul timpului diferite interpretari si i-au fost aduse diferite solutii ce se pot incadra in doua categorii: motode de predictie bazate pe psihoacustica si metode bazate pe invatare automata.Metodele bazate pe invatare automata se impart la randul lor in metode bazate pe modele de clasificare si metode bazate pe modele de regresie. Metodele regresive de predictie a nivelului de zgomot au fost realizate prin utilizarea la randul lor a diferitelor modele de invatare automata si invatare profunda precum: modele de tip ANN(*Artificial Neural Network*) , SVM(*Support Vector Machine)* si MLR(*Multiple regression models*). S-au folosit modele de regresie MLR cat si retele neuronale bazate pe algoritmul de propagare inversa (BPNN) , ce ofera rezultate mai bune, fata de MLR care rezolva eficient probleme de regresie in care relatia dintre variabilele de intrare este una corelata si liniara, in schimb ofera rezultate mai putin satisfacatoare cand variabilele de intrare prezinta relatii neliniare . Modelele de regresie ANN si SVM ofera rezultate bune in cazul datelor de intrare neliniare , fiind un inlocuitor bun pentru modelul MLR , acestea doua prezinta trei sau mai multe straturi de neuroni, ce pot astfel sa determine o ierarhie a nivelelor de abstractizare a datelor, de la variabilele de intrare pana la cele de iesire.

Pentru imbunatatirea rezultatelor unui model de predictie a calitatii zgomotului , s-a introdus o tehnica de invatare profunda ce are la baza retelele neuronale *Deep Belief .* Retelele neuronale *Deep Belief*  au ca si caracteristici: densitatea conexiunilor intre straturile retelei si neliniaritatea straturilor ascunse (modelul RBM,*Restricted Boltzmann Machine*).



*Comparare valori prezise cu valori reale pe cele 4 modele de baza ale predictiei calitatii sunetului in interiorul unei masini*

Testele au aratat ca in momentul de fata modelul CRBM-DBN este mai precis fata de cele trei modele MLR , BPNN si SVM pe problemele ce impun variabile cu o structura complicata (neliniare , greu de corelat) , in functie de cerinta problemei si caracteristicile datelor aceastea din urma pot totusi oferii rezultate bune.

1. **Inteligenta Artificiala vs. Invatare automata vs. Invatare profunda**

Inteligenta artificiala, invatarea automata si invatarea profunda sunt trei termeni, ce formeaza o ramificatie pentru care fiecare din noduri caracterizeaza termul de soft inteligent. Invatarea profunda este unul din nodurile ce are drept parinte nodul numit invatare automata, care la randul sau are parinte nodul principal numit inteligenta artificiala. Nodurile invatare automata si invatare profunda reprezinta un subset din setul parinte, ambele fiind subseturile setul inteligenta artificiala.

Inteligenta artificiala inglobeaza toate caracteristicile ce descriu comportamentul inteligent in calcule, capacitatea unei masinarii de a imita comportamentul uman, sisteme informatice ce pot indeplinii sarcini umane precum, recunoasterea vorbirii, luarea deciziilor, previziune.

Un subset al inteligentei artificiale, invatarea automata, este una din cele mai de succes ramuri din sfera dezvoltarii software din ultimii ani. Invatarea automata este descrisa de catre Arthur Samuel, unul din pionierii invatarii automate ca fiind un “*camp de studiu care ofera calculatoarelor capacitatea de a invata fara a fi programat explicit* “. Algoritmii de invatare automata prin procesul de invatare incearca o optimizare a previziunii prin minimizarea erorii (diferenta dintre valoarea prevazuta de program si valorea reala). Aceasta minimizare a erorii este posibila prin integrarea a trei caracteristici: functia de eroare, functia de pierdere si algoritmul specific fiecarui tip de problema in parte (regresie, clasificare).

Invatarea profunda este o subcategorie a invatarii automate, invatarea profunda este caracterizata de modele ce prin structura lor incearca o asemanare cu retelele neuronale biologice. Cresterea rapida a cantitatii datelor si a informatiilor publicate, precum si noile specificatii ale calculatoarelor actuale, a facut posibila dezvoltatea noilor tipuri de retelele neuronale pentru invatarea profunda, mult mai performante si mai precise. Performantele cele mai insemnate pe care le-au obtinut noile tipuri de retele neuronale provin din invatarea supravegheata sau invatarea prin etichetare. In concluzie invatarea profunda este caracterizata de retelele neuronale, ce presupun

prelucrarea unui numar foarte mare de date de intrare si foarte multe straturi de neuroni, care la randul lor ridica cerinte de sistem foarte complexe. Invatarea profunda actuala este concentrata pe instruirea unor modele de retele neuronale profunde, cu multe straturi de neuroni ce sunt parcurse prin algoritmul de propagare inversa. Cele mai populare tehnici de manipulare a retelelor neuronale profunde si in speta a conceptului de invatare profunda (*deep learning*) sunt:

* Retelele neuronale multistrat (*MLP*);
* Retelele neuronale convolutionale (*CNN*);
* Retelele neuronale recurente (*RNN*), in special modelul *LSTM (Long Short-Term Memory).*

1. **Invatarea supervizata**

Invatarea supervizata cuprinde o serie de probleme ce implica utilizarea unui model pentru a invata o potrivire intre datele de intrare si datele tinta.

Modelele retelelor neuronale se antreneaza pe setul de datele de antrenare formate din valori de intrare si valori tinta, urmand ca mai apoi sa se faca predictii pe setul datelor de test. Setul datelor de test cuprinde valorile de intrare si valorile de iesire care ulterior sunt comparate cu valorile tinta retinute la pasul de antrenare al retelei, pentru a estima mai apoi eroarea dintre valorile prezise si cele reale (puterea de predictie a modelului).

Exista doua tipuri principale de problemele specifice invatarii supervizate: probleme de clasificare si probleme de regresie.

**2.1 Probleme de clasificare**

Problemele de clasificare, sunt probleme ce implica prezicerea unei valori discrete (a unei clase sau a unei categorii). Functia de mapare prezice clasa sau categoria pentru un set de date de intrare. Setul de date de intrare prezinta anumite caracteristici particulare uneia sau mai multor clase sau categorii. Algoritmul capabil sa invete un model de predictie al clasificarilor se numeste algoritm de clasificare, preformanta acestuit algoritm poate fi stabilita urmarind procentul de exemple clasificate corect din numarul total de clasificari facute.

Tipuri de modele de clasificare:

* Regresie logistica (*Logistic Regression*);
* Masina de suport vectorial (*Support Vector Machine*) ;
* Bayes naiv (*Naive Bayes*) ;
* Arbori de decizie (*Decision Tree*);
* Retea neuronala (*Neural Network*);
* *Random forest;*

**2.2 Probleme de regresie**

Problemele de regresie sunt probleme ce implica prezicerea unei valori continue, valoare reala, cum ar fi o valoare intreaga sau o valoare flotanta. Aceste valori continue sunt adesea cantitati de tipul valoare si marime. O problema de regresie poate primi date de intrare atat de tipul valori discrete cand si de tipul valori continue. Problemele cu multiple valori de intrare sunt numite probleme de previziune regresiv multivariate, problemele cu valori de intrare ordonate in timp se numesc probleme de previziune a seriei de timp. Performata unui algoritm de regresie este stabilita, cel mai des, prin calcularea erorii medii a radacinii.

Tipuri de modele de regresie:

- Regresie liniara (*Linear Regression*);

- Arbori de decizie (*Decision Tree*);

- *Random Forest;*

- Retea neuronala (*Neural Network*);

**2.2.1 Regresia Liniara (Linear Regression)**

Regresia Liniara este un model specific problemelor de regresie utilizat in problemele de predictie a unor cantitati numerice. Modelul regresiei liniare impune o valoare de iesire ce este calculata ca o combinatie liniara a datelor de intrare, ceea ce sugereaza faptul ca reprezentarea modelului este o ecuatie liniara:

*y = B0 + B1\* x* **,** unde y este predictia modelului, x este o valoare

deintrare**,** coeficientiiB0 siB1 (coeficienti de

libertate).

Coeficientii B0 siB1 vor fi alesi in asa fel incat sa se minimizeze suma erorilor patrate intre iesirile prezise de retea si valorile reale. Procesul de estimare a celor doi parametrii B0 siB1 se numeste proces de invatareal modelului , utilizand datele de intrare disponibile.

Metode de estimare a coeficientilor B0 siB1 pentru minimizarea sumei erorilor patrate:

- Metoda *Ordinary Least Squares;*

- Metoda gradientului descendent (*Gradient Descent) ;*

*-* Metode de regularizare.

Structurarea datelor astfel incat modelele de regresie liniara sa ofere cele mai bune rezultate:

- Relatia dintre datele de intrare si raspunsul asteptat de retea sa fie

una liniara;

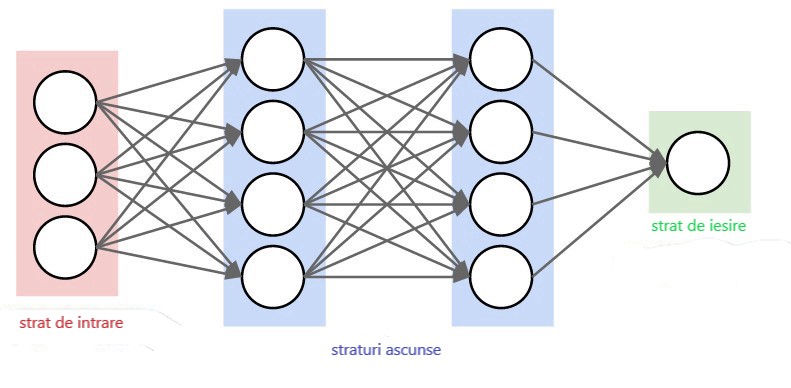
- Datele de intrare este indicat as fie puternic corelate ;

- Modelul regresiei liniare va face predictii mai bune daca datele de

intrare sunt standardizate sau normalizate.

**3. Retele neuronale (*Neural Networks*)**

Elementele unei retele neuronale:

****

Stratul de intrare (*input layer*) introduce datele ce urmeaza sa fie procesate de catre reteaua neuronala.

Stratul ascuns (*hidden layer*) impreuna cu nodurile ce intra in componenta sa este caracterizat ca fiind partea ascunsa si abstracta oferita de orice retea neuronala. In straturile ascunse ale retelei neuronale se efectueaza calcule pe datele intruduse in stratul de intrare, urmand ca mai apoi acestea sa fie furnizate catre stratul de iesire. Nodurile retelei neuronale imita neuronii creierului uman, legaturile dintre neuronii artificiali, numite ponderi sau greutati(*weight*) imita sinapsele. Nodurile retei neuronale primesc informatiile prin nodurile straturilor anterioare, le inmultesc cu greutatile aferente urmand ca la final sa adauge o prejudecata(*bias*). Prejudecata (*bias*) este cumva similara constantei b dintr-o functie liniara:

*Y = aX +b*

Dreapta care se formeaza, fara prezenta constantei b ar trece prin origine (0, 0), ceea ce ar duce la o eroare provocata de diferenta dintre datele de intrare si datele prezise de retea foarte mare.

Stratul de iesire (*output layer*) pune laolalta informatiile din ultimul strat ascuns, si afiseaza raspunsul final al retelei.

**3.1 Tipuri de retele neuronale artificiale**

Trei dintre cele mai importante clase de retele neuronale artificiale, care de-a lungul timpului s-au dovedit eficiente si flexibile pe majoritatea problemelor.

- Retele neuronale de tip *Multilayer Perceptrons* (MLPs) ;

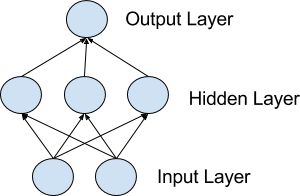
- Retele neuronale de tip *convolutional, Convolutional Neural Networks* (CNN) ;

- Retele neuronale de tip recurent, *Recurrent Neural Network* (RNN) ;

La randul lor cele trei mari clase de retele neuronale au de asemnea subtipuri, particularizate de diferite tipuri de probleme, seturi de date, sau cerinte de predictie.

**3.1**.**1** **Retele neuronale de tip *Multilayer Perceptrons (MLPs)***

Reteaua neuronala de tip multilayer perceptron este tipul generic de retea neuronala, cu un set de date de intrare reprezentand stratul de intrare, un strat sau mai multe straturi ascunse, in functie de nivelul de abstractizare dorit, si predictiile, ce iau forma stratului de iesire.



Retelele neuronale de tip multilayer perceptron sunt potrivite pentru problemele clasice de clasificare, in care intrarilor li se atribuie o clasa sau o eticheta. Acest tip de retea neuronala este potrivita si pentru problemele de predictie regresive, probleme ce prezic un set de date set de date , valori reale sau continue precum salariul unui angajat sau greutatea unei persoane , avand ca date de intrare un alt set de valori . De cele mai multe ori setul de date primit la intrare este sub forma tabulara, foi de calcul sau fisiere CSV.

Folosim retelele neuronal multilayer perceptron (MLPs) pentru :

- Date de intrare sub forma tabulara ;

- Probleme de predictie pe baza de regresie ;

- Probleme de predictie pe baza de clasificare ;

Tipul de retea neuronala multilayer perceptron este foarte flexibil si poate fi folosit pentru invatarea unei mapari de la datele de intrare catre cele de iesire. Aceasta flexibilitate poate fi aplicata pe diferite tipuri de date de intrare, spre exemplu pixelii unei imagini pot fi redusi la un rand lung de date si introdusi intr-o retea de tip MLPs.Putem incerca sa folosim un model de retea MLPs pe diferite tipuri de date de intrare precum :

-Date de intrare de tip imagine ;

- Date de intrare de tip text ;

- Date de intrare bazate pe serii temporale ;

- Alte tipuri de date de intrare ;

**3.1**.**2 Retele neuronale de tip convolutional (*Convolutional Neural Networks)***

Retelele neuronale de tip convolutional au fost create pentru a mapa datele de intrare de tip imagine intr-o variabila de iesire de tip parametru. Acestea s-au dovedit atat de eficiente in probleme de predictie ce implica date de intrare de tip imagine incat au devenit o metoda principala pentru acest tip de problema de predictie. Avantajul folosirii unei retele de tip convolutional pentru acest gen de problema de predictie, este capacitatea de a crea o reprezentare interna a unei imagini bidimensionale.

Utilizam retelele neuronale de tip convolutional pentru:

- Date de intrare de tip imagine ;

- Probleme de predictie pe baza de clasificare ;

- Probleme de predictie pe baza de regresie ;

In general retelele de tip convulutional se muleaza pe probleme de predictie ce implica date cu relatii spatiale. Datele de intrare ale unei retele de tip convulutional sunt in general construite pe doua dimensiuni(bidimensionale), insa acest aspect poate fi modificat pentru a fi unidimensional, permitandu-i retelei neuronale as dezvolte o reprezentare interna pe o singura dimensiune. Acest aspect face ca retelele neuronale sa fie folosite si cu alte tipuri de date de intrare ce prezinta o relatie spatiala. Putem incerca as folosim un model de retea convulutionala pentru urmatorele date de intrare :

- Date de intrare de tip text ;

- Date de intrare bazate pe serii temporale ;

- Date de intrare secventiale ;

**3.1**.**3 Retele neuronale de tip recurent (*Recurrent Neural Networks*)**

Retelele neuronale de tip recurent au fost dezvoltate pentru rezolvarea problemelor de predictie a secventelor de date.

Problemele de predictie a secventelor vin sub diferite forme fiind descrise in functie de tipurile de date de intrare si de iesire suportate. Majoritatea problemelor de predictie a secventelor de date includ urmatoarele relatii intre date :

*- one to many –* o observatie primita ca *input* este mapata ca o secventa

cu mai multi pasi ca *output*.

- *many to one* – o secventa cu mai multi pasi ca *input* este mapata ca o

clasa sau o valoare ca *output.*

- *many to many* – o secventa cu mai multi pasi ca *input* este mapata ca

secventa cu mai multi pasi ca *output.*

Retelele neuronale de tip recurent sunt in general greu de antrenat, *The Long Short-Term Memory*, sau LSTM, este cel mai de succes model de retea neuronala, acesta reusind sa depaseasca problemele si greutatea aparuta in urma antrenarii retelei neuronale de tip recurent, acesta reusind sa se plieze si sa ofere rezultate bune pe mai multe tipuri de probleme de predictie.

Retelele neuronale de tip recurent si in special modelul LSTM au dat rezultate foarte bune pe secvente de date pe baza de cuvinte si paragrafe, acest proces primeste numele de prelucrare a limbajului natural, *natural language processing.* Folosim retelele neuronale de tip recurent pentru:

- Date de intrare de tip text;

- Date de intrare de tip inregistrare audio;

- Probleme de predictie pe baza de clasificare;

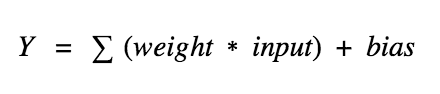
- Probleme de predictie pe baza de regresie;

- Modele generative;

Retelele neuronale de tip recurent nu se pliaza foarte bine pe probleme de predictie ce implica date de intrare sub forma tubulara, precum foile de calcul, fisierele CSV, de altfel nu sunt potrivite nici pentru datele de intrare sub forma de imagini. Putem incerca as folosim un model de retea recurenta, sau LSTM pentru date de intrare sub forma de serii temporale.

**3.2 Functiile de activare in Retelele Neuronale**

Neuronul artificial este cel care transmite mai departe informatia in retea, acesta insumeaza produsul dintre fiecare intrare a sa si pondere, adaugand la final asa numita prejudecata(*bias*).



Dupa calculul sumei ponderate si adaugarea prejudecatii, rezultatul trece printr-o functie de activare, care la randul ei decide daca neuronul trebuie activat sau nu. Scopul functiei de activare este acela de a transforma iesirea unui neuron intr-un raspuns neliniar. Neliniaritatea intrarii unui neuron face reteaua neuronala capabila sa invete si sa rezolve probleme mult mai complexe, lisa functiei de activare transforma reteaua neuronala intr-un simplu model de regresie liniara.

* + 1. **Functia Liniara**

*f(x)=x.*

Functia liniara este utilizata in straturile finale de iesire, ecuatia functiei este similara cu ecuatia unei drepte.

* + 1. **Functia Sigmoid**

*f (x)=*

Functia sigmoid furnizeaza un interval valoric intre 0 si 1. Este utilizata in special in straturile finale ale problemelor pentru clasificare binara, unde rezultatele sunt intre 0 si 1.

* + 1. **Functia tangenta hiperbolica (tanh)**

*f(x) =*

Functia tanh furnizeaza un interval valoric intre -1 si +1, este o functie neliniara si este folosita de obicei in straturile ascunse. Prin furnizarea intervalului valoric [-1,1] media stratului ascuns este de 0, ceea ce duce la centrarea datelor si prin urmare faciliteaza procesul de invatare.

* + 1. **ReLU (*Rectified Linear Unit*)**

*f(x) = max (0, x)*

Functia ReLU furnizeaza doar raspuns pozitiv. Functia furnizeaza un raspuns pozitiv diferit de 0, ceea ce inseamna ca neuronul este activ, daca x este pozitiv si 0 daca neuronul nu este activ. Functia ReLU este cea mai folosita functie de activare, datorita mecanismului simpu de calcul al starii neuronului, care face ca in final doar cativa dintre neuroni sa ramana activi. Acest lucru reduce dimensiunea retelei neuronale, facand-o mai usor din punct de vedere al sistemului de calcul si mai eficienta.

* + 1. **Functia Softmax**

f(xi) =

Functia Softmax este o functie neliniara, utilizata in special la problemele de clasificare, in straturile de iesire, unde se incearca definirea probabilitatilor de a definii o clasa pentru fiecare intrare in parte.

**3.3 Functii de cost (*loss functions*)**

Este important de spus faptul ca functia de cost (loss *function*) este puternic legata de functia de activare folosita in stratul de iesire al retelei neuronale. Momentul in care se alege functia de activare pentru stratul de iesire se realizeaza si incadrarea problemei intr-o anumita tipologie, astfel functia de cost nu vine decat ca o completare a functiei de activare. Functia de cost este o modalitate de calculare a erorii de predictie a retelei neuronale.

* + 1. **Eroarea medie patrata (*Mean Squared Error Loss)***

Eroarea medie patrata este calculata ca media diferentelor patrate intre valorile prezise de retea si valorile reale sau cele pe care se doreste sa le prezica reteaua. In urma aplicarii acestui tip de functie de cost, valorile furnizate sunt doar valori pozitive, o valoare perfecta este considerata valoare 0,0. Este folosita cu preponderenta in problemele de regresie.

* + 1. **Entropia incrucisata (*Cross-Entropy Loss*)**

Este folosita cu preponderenta in problemele de clasificare cu mai multe clase. Functia de pierdere in entropia incrucisata masoara performata unui model de clasificare a carui iesire este o valoare intre 0 si 1. Pe masura ce valoarea prezisa se indeparteaza de eticheta reala, functia de pierdere creste. Un raspuns bun al functiei de pierdere in acest caz este 0.

**3.4 Metoda de optimizare Gradient Descendent. Algoritmul de descendenta cu gradient stocastic (Stochastic Gradient Descent). Algoritmul de propagare inversa (Backpropagation)**

Problema ce sta la baza conceptului de *machine learning* se rezuma la o problema de optimizare, anume, modul in care reteaua neuronala isi perfectioneaza procesul de invatare. Procesul de invatare este perfectionat prin reducerea diferentei dintre valoarea prevazuta de retea si valoarea reala, valoarea ce se doreste a fi prevazuta. Procesul de perfectionare al retelei neuronale foloseste algoritmul “*Gradient Descent “,* algoritm de descendenta in gradient si actualizeaza ponderile folosind algoritmul “Backpropagation error”, algoritmul de propagare inversa a erorii.

*Gradient descent* este o metoda de optimizare sau perfectionare a retelei neuronale cea mai utilizata care poate fi folosita pe majoritatea algortimilor de invatare automata. Gradientul in natura sa este o panta a unei functii. Algoritmul “*Gradient descet”* incearca sa schimbe ponderile astfel incat sa se reduca eroarea predictiilor. Pornind de o la valoare initiala algoritmul de “*Gradient Descent*” este rulat de mai multe ori, pana cand se ajunge la valorile optime ale parametrilor, ulterior la o functie de cost minima.

*Stochastic Gradient Descent* este o metoda de optimizare asemanatoare “*Gradient descent*” cu deosebirea ca, in acest caz nu se parcurg toate datele din setul de antrenare, algoritmul selecteaza aleator un subset din setul de antrenare si doar pe acest subset actualizeaza ponderil, ajungand astfel la valorile optime ale ponderilor.

Adesea se foloseste tehnica “*Stochastic Gradient Descent” in defavoarea “Gradient descent”* in special in cazul unei retele neuronale cu foarte multe date in setul de antrenare. Tehnica “*Stochastic Gradient Descent”* este mult mai rapida, ajungandu-se la aflarea gradientului functiei de cost intr-un timp semnificativ mai scurt.

Algoritmul de propagare inversa (*Backpropagation*) este o metoda de perfectionare a procesului de invatare al retelei neuronale cu mai multe straturi. Algoritmul “*Backpropagation****”*** actualizeaza ponderile retelei neuronale prin manipularea unei functii de cost, pentru a minimiza pe cat posibil eroarea dintre valoarea prezisa de retea si valoarea reala, cea la cara se doreste sa ajunga rezultatul prezis de retea.

Cu ajutorul metodei “Gradient Descent” algoritmul de propagare inversa practic propaga inapoi prin retea, de la stratul de iesire la stratul ascuns noile actualizari asupra ponderilor care arata fie o crestere fie o descrestere a ponderilor, pana in momentul in care eroarea este minimizata cat mai aproape de 0. Este de mentionat cazul in care dupa o scadere eroarea incepe sa creasca din nou, in acest moment este indicata oprirea propagarii si actualizarii ponderilor, deoarece s-a ajuns la o valoarea optima a ponderilor

* + 1. **Algoritmi de optimizare**

**3.4.1.1 Algoritmul de optimizare *Momentum***

Algoritmul de optimizare *Momentum* este similar algoritmului SGD (*Stochastic Gradient Descent*) cu mentiunea ca de aceasta data se adauga doi noi paramentrii in ecuatia algoritmului pentru actualizarea ponderilor si anume un element v ce reprezinta viteza (suma curenta a gradientilor) si p ce reprezinta impulsul(poate fi comparat cu o forta de frecare ce incetineste viteza).Algoritmul *Momentum* a fost creat ca o varianta ajutatoare pentru algoritmul de optimizare *SGD* , prin accelerarea coborarii gradientului , luand in considerare gradientul pasului curent si gradientul pasului anterior . Prin accelerarea coborarii in gradient, *Momentum* face ca algoritmul de SGD sa restrictioneze oscilatia pe o singura directie si anume catre optime locale relevante eliminand oscilatiile pe optime locale irelevante.

**3.4.1.2 Algoritmul de optimizare *Nesterov***

Datorita faptului ca algoritmul de optimizare *Momentum* intampina probleme in momentul in care impulsul de coborare in gradient este prea mare, s-a dezvoltat algoritmul de optimizare *Nesterov* ce in loc de gradientul curent se foloseste de gradientul viitor. Ecuatiei de baza a SGD se adauga un element temporal alcatuit din gradientul pasului anterior si gradientul pasului viitor, astfel se poate face o evaluare asupra tendintei de crestere sau coborare a gradientului, urmand ca mai apoi sa se efectueze actualizarea ponderilor.

**3.4.1.3 Algoritmul de optimizare *Adagrad***

Algoritmul de optimizare Adagrad , actualizeaza ponderile diferit una fata de alta , actualizarea se face pentru fiecare pondere in parte si cu rata de invatare diferita , spre deosebire de algoritmii de optimizare precum *Nesterov* sau *Momentum* care aduc toate ponderile la aceasi valoare si aplica aceasi rata de invatare pentru fiecare pondere in parte.Gradientul ponderii curente este calculat din suma gradientilor patrati, astfel datorita pozitivitatii fiecarui termen , rata de invatare devine mult mai mica , ducand astfel la o optimizare a procesului de invatare pana in punctul in care reteaua nu mai poate invata .

**3.4.1.4 Algoritmul de optimizare RMSProp (*Root Mean Square Propagation*)**

Algoritmul de optimizare *RMSProp* este similar algoritmului *Momentum,* *RMSProp* restrictioneaza oscilatiile pe verticalala, concentrand atentia pe oscilatiile orizontale, astfel algoritmul grabeste procesul de convergenta catre un optim local satisfacator. *RMSProp* de foloseste de marimea gradientilor anteriori pentru normalizarea gradientilor curenti. Algoritmul *RMSProp* calculeaza in permanenta o suma curenta a gradientilor patrati anteriori pe care ulterior o inmulteste cu o rata de descompunere. Adaugarea elementului de descompunere a sumei curente a gradientilor patrati anteriori, algoritmul *RMSProp* rezolva problema algoritmului *Adagrad*, prin faptul ca nu lasa suma gradientilor sa creasca atat de mult.

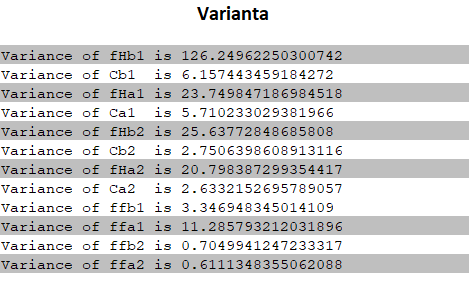
**3.4.1.5 Algoritmul de optimizare Adam**

Algoritmul de optimizare *Adam* este o varianta inlocuitoare a algoritmului “*Stochastic Gradient Descent*” pentru actualizarea ponderilor dintr-o retea neuronala. Una din diferentele cele mai insemnate dintre cei doi algoritmi este modificarea ratei de invatare, pentru *Stochastic Gradient Descent* rata de invatare se pastreaza pentru fiecare pondere in parte, urmand sa fie modificata ulterior pe parcursul procesului de invatare. Algoritmul de optimizare Adam aduce la un loc algoritmii *Stochastic Gradient Descent with Momentum* si *RMSProp.* Algoritmul *Adam* aduce la un loc cele doua ecuatii ale celor doi algoritmi, introducand ulterior doi hiperparametrii, anume momentul de grad 1 si momentul de grad 2.Cei doi hiperparametrii este indicat a fi setati la valorile 0.9(β1) si 0.999(β2), rata de invatare = 0.001–0.0001.

**Prezentarea aplicatiei de invatare automata**

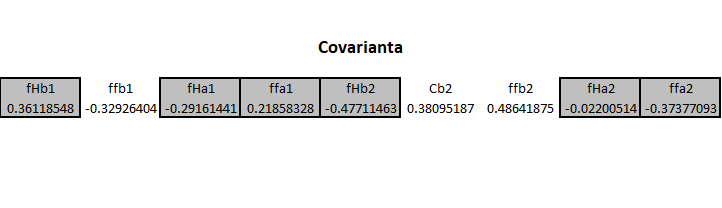
**Pasul 1 . Incarcarea datelor**

Primul pas in crearea modelului de invatare automata a fost incarcarea setul de date.Setul de date consta in sase dintre cei mai importanti parametrii in influentarea nivelului de zgomot al transmisiilor . Cei sase parametrii au fost alesi pe baza variantei , covariantei si a influentelor in procesele mecanice si de transfer a puterii, intre toti parametrii geometrici ai danturii pinioanelor.



*Varianta fiecarui parametru geometric al danturii pinionului in influentarea*

*zgomotului transmisiilor*

**

*Covarianta fiecarui parametru geometric al danturii pinionului in influentarea*

*zgomotului transmisiilor*

Aplicatia pe care am dezvoltat-o rezolva in sine o problema de regresie liniara ( nivelul de zgomot al transmisiilor pe baza parametrilor geometrici ai danturii pinioanelor ). Toate variabilele de intrare ce descriu cotele danturii pinionului sunt numerice:

fHβ1 = abaterea de la directia nominala a elicei sau abatarea de la unghiul de inclinare al

elicei, pentru pinionul 1

fHβ2 = abaterea de la directia nominala a elicei sau abatarea de la unghiul de inclinare al

elicei, pentru pinionul 2

fHα1 = abaterea de la directia nominala a profilului sau abaterea de la unghiul de inclinare

al profilului, pentru pinionul 1

fHα2 = abaterea de la directia nominala a profilului sau abaterea de la unghiul de inclinare

al profilului, pentru pinionul 2

ffα1 = abaterea de la forma conforma a profilului pinionului 1

ffα2 = abaterea de la forma conforma a profilului pinionului 2

Variabila de iesire , masoara

**Theano**

Theano este o biblioteca Python, gratuita si un compilator optimizat. Theano este folosit atat pentru evaluarea si manipularea expresiilor matematice cat si pentru aplicatii de invatare automata precum retelele neuronale.

**Tensorflow**

Tensorflow este o biblioteca software gratuita, utilizata pentru aplicatii de invatare automata asistata precum retelele neuronale, permite dezvoltatorilor sa creeze retele neuronale cu mai multe straturi de neuroni.

**Keras**

Keras este o arhitectura de invatare profunda pentru Python, dezvoltata de echipa *Google Brain* pentru utilizare interna *Google*. Keras permite utilizatorilor sa dezvolte si sa antreneze modele de retele neuronale profunde intr-un mod rapid si usor. Suporta retele neuronale recurente, convolutionale, cat si combinatia dintre cele doua (retele neuronale hibrid). Keras impacheteaza bibliotecile *Tensorflow* si *Theano* specializate pe manipularea calculelor numerice avansate.