Բովանդակություն

Ներածություն

Թաքնագրությունը[Error! Reference source not found.] գաղտնի տեղեկատվությունը ոչ գաղտնի տեղեկատվության (կոնտեյներ, կամ կրիչ) մեջ թաքցման մեթոդների հավաքածու է: Իսկ թաքնավերլուծությունը[Error! Reference source not found.] (Steganalysis), մի գործընթաց է, որն ուղղված է պարզելուն, թե արդյո՞ք հաղորդագրությունը պարունակում է թաքնված ինֆորմացիա, և հնարավորության դեպքում վերականգնել այն: Թաքնված ինֆորմացիայի ներկայությունը հայտնաբերելու համար սովորաբար օգտագործվում է երկուական դասակարգիչ (Binary classifier): Սույն ուսումնասիրության մեջ ներկայացվելու է մի մոդել, որը ստեղծում է նկար-կրիչներ, հիմնված՝ խորը պարուրման ստեղծարար մրցակցող ցանցերի (Deep Convolutional Generative Adversarial Networks, կրճատ՝ DCGAN)[Error! Reference source not found.,Error! Reference source not found.] վրա: Այս մոտեցումը թույլ է տալիս ստեղծել ավելի թաքնակայուն կրիչ, ներդրված հաղորդագրությամբ, օգտագործելով ստանդարտ թաքնագրային ալգորիթմներ:

Այս թեմայի շուրջ 2016թ.-ին կատարվել է հետազոտություն[Error! Reference source not found.], որի ընթացքում փորձել են գեներացնել մարդկանց դեմքեր: Մոդելը հաջողությամբ մոլորեցրել է թաքնագրային վերլուծիչին, սակայն որոշ դեպքերում մարդու աչքը գեներացված նկարները հեշտությամբ կարող էր տարբերել իրականից, քանզի մոդելին՝ ուսուցման ժամանակ, տրամադրվել էին տարբեր սեռի մարդկանց դեմքեր, սակայն չէին հաշվի առել այդ հանգամանքը:

Ուսուցմանը մասնակցելու են միանգամից 3 մոդել: Դրանք են՝

1. Գեներացնող մոդել (Գեներատոր - Generator) - G
2. Տարբերակող մոդել (Տարբերակիչ - Discriminator) - D
3. Թաքնավերլուծող մոդել (Թաքնավերլուծիչ - Steganalyser) - S

Առաջին մոդելը՝ գեներատորը, պատասխանատու է նկարներ գեներացնելու համար, այն պետք է այնպիսի նկարներ գեներացնի, որ հնարավոր չլինի տարբերել իրական նկարներից: Այս խնդրի լուծման համար օգտագործվելու է երկրորդ մոդելը՝ տարբերակիչը, որի խնդիրն է լինելու տարբերել իրական նկարը կեղծից (կեղծ են բոլոր այն նկարները որոնք ստեղծել է G գեներատորը): Այս ամենից հետո գործի է անցնում 3-րդ մոդելը՝ վերլուծիչը, որի խնդիրն է պարզել արդյո՞ք տրված նկարում առկա է թաքնագրված ինֆորմացիա, թե՞ ոչ: D վերլուծիչին ուսուցման ընթացքում տրամադրվելու են գեներատորի նկարները, որոնք արդեն

պարունակում են թաքնագրված ինֆորմացիա, ինչպես նաև սովորական նկարներ, որոնք չեն պարունակում ոչ մի թաքնագրված ինֆորմացիա:

Այսպիսով D տարբերակիչն ու S վերլուծիչը բարելավելու են իրենց արդյունքը՝ հիմնվելով G գեներատորի տրամադրած և սովորական նկարների վրա, իսկ G-ն բարելավելու է իր արդյունքը՝ հիմնվելով D-ի և S-ի արդյունքի վրա: Հենց այստեղ էլ առաջ է գալիս մրցակցող ցանցերի գաղափարը, քանզի ստացվում է, որ ցանցերը մրցում են միմյանց հետ, թե ում արդյունքն ավելի լավը կլինի:

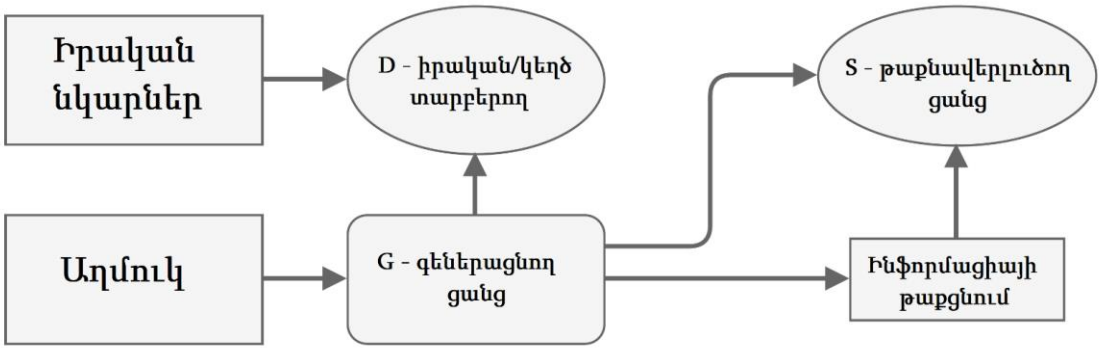
Վերջերս մշակված մրցակցող ցանցերը[Error! Reference source not found.] հզոր գեներացնող մոդելներ են, որոնց հիմնական գաղափարը գեներատորի և տարբերակիչի ուսուցումն է մինիմաքս խաղի[Error! Reference source not found.] միջոցով: G մոդելը մուտքին ստանում է պատահական՝ այսպես ասած անիմաստ նկար, որի հիման վրա փորձում է ստեղծել հնարավորինս իրականին մոտ պատկեր, իսկ D-ն ձգտում է տարբերակել իրական պատկերները կեղծերից:

Գոյություն ունեն նմանատիպ ցանցերի տարբեր ձևափոխություններ՝

- Խորը պարուրման ստեղծարար մրցակցող ցանցեր[Error! Reference source not found.]
  - այս մոդելը ստեղծարար մրցակցող ցանցի (GAN) փոփոխություն է, որը մասնագիտացված է պատկերների առաջացման ուղղությամբ
- Պայմանական մրցակցող ցանցեր[Error! Reference source not found.]
  - թույլ է տալիս ստեղծել որևէ դասի օբյեկտներ
- Պատկերների առաջացում՝ հիմնված տեքստային նկարագրության վրա[Error! Reference source not found.]:

Թաքնագրվող գաղտնի ինֆորմացիան, ինչպես նաև կրիչը, կարող է ներկայացված լինել տարբեր տեսքով՝ նկարի, տեքստի, տեսահոլովակի, ձայնագրության և այլն: Այս ուսումնասիրության մեջ կատարվելու է տեքստի թաքնագրում նկարում և օգտագործվելու է DCGAN տեսակը:

Մոդելները և նրանց միջև կապերը ներկայացված են Նկ. 1-ում



Նկ. 1 Գեներատիվ մրցակցող ցանցի մոդելներն ու նրանց կապերը

Գլուխ 1. Գրականության վերլուծական ակնարկ

1.1 Մեքենայական ուսուցում

Նախքան անցնելը բուն թեմային, ծանոթանանք մեքենայական ուսուցման (Machine Learning[9], կրճատ՝ ML) հետ: Արթուր Սամուելն այն նկարագրում է այսպես «մեքենայական ուսուցումը մի տեխնոլոգիա է, որը համակարգիչներին հնարավորություն է տալիս սովորելու, առանց բացահայտ ծրագրավորված լինելու»: Սա, իհարկե, ոչ պաշտոնական ձևակերպում է, սակայն լավ պատկերացում է տալիս:

Մեքենայական ուսուցման խնդիրներից են.

- Վերահսկվող ուսուցում (Supervised learning)
- Չվերահսկվող ուսուցում (Unsupervised learning)
  - Սրա մասնավոր դեպք է խորհրդատու համակարգը (Recommender system)
- Ուսուցում ամրապնդմամբ (Reinforcement learning)

Վերահսկվող ուսուցման դեպքում մեքենային տրվում է մուտքային տվյալների հավաքածու և այդ տվյալներին համապատասխան ելքային արժեքները: Այսպիսով այս ուսուցման դեպքում մեքենային հայտնի են ամեն մի մուտքային ինֆորմացիային համապատասխանող ելքային արժեքը կամ արժեքները:

1.1.1 Վերահսկվող ուսուցում

Վերահսկվող ուսուցման (Supervised Learning) խնդիրները դասակարգվում են հետևյալ 2 տիպերի.

- Ռեգրսիայի խնդիրներ (Regression problems)
- Դասակարգման խնդիրներ (Classification problems)

Ռեգրեսիայի խնդրներում փորձում ենք կանխատեսել անընդհատ ֆունկցիայի արժեքներ, ինչը նշանակում է, որ մենք փորձում ենք մուտքային փոփոխականները համապատասխանեցնել ինչ-որ անընդհատ ֆունկցիայի ելքային արժեքներին: Դասակարգման հարցում մենք փոխարենը փորձում ենք կանխատեսել ընդհատ ելքային արժեքներ: Այլ կերպ ասած, մենք փորձում ենք մուտքային փոփոխականները համապատասխանեցնել դիսկրետ կատեգորիաների:

Ռեգրեսիայի խնդրի օրինակ՝ «Տրված մարդու նկարից որոշել նրա տարիքը»:

Դասակարգման խնդրի օրինակ՝ «Տրված է որևէ հիվանդի ուռուցքի մասին ինֆորմացիա, որոշել արդյո՞ք ուռուցքը չարորակ է, թե՞ բարորակ»:

1.1.2 Չվերահսկվող ուսուցում

Չվերահսկվող ուսուցումը (Unsupervised Learning) հնարավորություն է տալիս լուծել այնպիսի խնդիրներ, որոնց ելքային արժեքների մասին կա՛մ քիչ ինֆորմացիա ունենք, կա՛մ ընդհանրապես չգիտենք, թե ինչ տեսքի պետք է լինեն: Մենք կարող ենք ստանալ մի այնպիսի ելքային տվյալի կառուցվածք, որի վրա մուտքային տվյալի ազդեցությունն անգամ չգիտենք: Այդ կառուցվածքը հնարավոր է ստանալ տվյալները համախմբելու արդյունքում՝ հիմնված մուտքային տվյալի փոփոխականների միջև կապերի վրա:

Չվերահսկվող ուսուցման ժամանակ կանխատեսման արդյունքների վրա հիմնված հետադարձ կապ չկա: Այսինքն մոդելը չի փոփոխում իր պարամետրերը՝ հիմնվելով կանխատեսման արդյունքների վրա:

Օրինակներ՝

Կլաստերիզացիա. վերցնել 1,000,000 տարբեր գեների հավաքածու և ավտոմատացնել այդ գեների խմբավորումն այնպիսի խմբերում, որոնք ինչ-որ կերպ նման են կամ կապված են տարբեր փոփոխականների հետ՝ ինչպիսիք են կյանքի տևողությունը, գտնվելու վայրը, դերը և այլն:

Ոչ կլաստերիզացիա. «Կոկտեյլային երեկույթի ալգորիթմը», թույլ է տալիս գտնել կառուցվածք քաոսային միջավայրում (այսինքն, առանձնացնել մարդու խոսակցության ձայնը երեկույթում հնչող երաժշտությունից):

1.1.3 Որոշ նշանակումներ

Կատարենք մի քանի նշանակումներ, որոնք կոգտագործվեն հետագայում:

Դիցուք ունենք հետևյալ տվյալները՝

X_1	…	X_n	Y
$Input^{(1)}{}_1$	…	$Input^{(1)}{}_n$	$Output^{(1)}$
…	…	…	…
$Input^{(m)}{}_1$	…	$Input^{(m)}{}_n$	$Output^{(m)}$

$X_1, X_2, \dots X_n$-ը մուտքային պարամետրերի նշանակումներն են, Y-ը՝ ելքային պարամետրի նշանակումը: $Input^{(i)}{}_1, Input^{(i)}{}_2, \dots Input^{(i)}{}_n$-ը մուտքային պարամետրերի արժեքներն են (տվյալի հատկություններ), իսկ $Output^{(i)}$-ն՝ ելքային պարամետրի արժեքն է, որտեղ՝ $i=1,2,\dots,m$: Հարմարավետության համար $Input^{(i)}{}_1, Input^{(i)}{}_2, \dots Input^{(i)}{}_n$-ը նշանակենք $x^{(i)}$-ով, իսկ $Output^{(i)}$-ն՝ $y^{(i)}$-ով: Պարզ է, որ՝ n-ը մուտքային պարամետրերի քանակն է:

$(x^{(i)}, y^{(i)})$ զույգն անվանում ենք ուսուցման օրինակ (training example), իսկ դրանց ցուցակը՝ ուսուցման տվյալներ (training set): Այսինքն m-ը՝ ուսուցման տվյալների քանակն է:

Այժմ կարող ենք տալ վերահսկվող ուսուցման ավելի ֆորմալ ձևակերպում՝ «Վերահսկվող ուսուցման նպատակն է՝ տրված ուսուցման տվյալների հիման վրա ձևավորել մի այնպիսի $h : X \rightarrow Y$ ֆունկցիա, որ $h(x)$-ի ելքային արժեքը բավարար մոտ լինի համապատասխան y-ի արժեքին»: h ֆունկցիան անվանում են «հիպոթեզ»:

Ինչքան $h(x)$-ի արժեքը մոտ լինի համապատասխան y-ի արժեքին, այնքան ավելի ճիշտ արդյունքներ կտա մեր մեքենայական ուսուցման մոդելը:

Բնականաբար *h(x)-ի* ունի գործակիցներ, նշանակենք այդ գործակիցները $\theta_0, \theta_1, \dots \theta_n$ *-ով,* այս պատճառով *h(x) -ը* որոշ դեպքերում կնշանակենք $h_\theta(x)$ *:*

Հասկանալի է, որ մեր խնդիրը հենց այդ θ *-ների* արժեքները գտնելու մեջ է կայանում, քանզի հետագայում՝ երբ արդեն մեր մոդելը բավարար չափով ուսուցանված կլինի, և ունակ կլինի գուշակել ճիշտ արժեքներ, նրան տրվելու են $X_1, X_2, \dots X_n$ արժեքները և քանզի այն ունի արդեն հաշվարկած $\theta_0, \theta_1, \dots \theta_n$ արժեքները, ընդամենը պետք է հաշվի $h_\theta(x)$ *-ի* արժեքը:

1.2 Ուսուցման տարրեր

1.2.1 Արժեքի ֆունկցիա

h(x)-ի արժեքների ճշտությունը կարելի է գնահատել **արժեքի ֆունկցիայի (Cost Function)** միջոցով: Այն իրենից ներկայացնում է *h(x)-ի* բոլոր ելքային արժեքների և իրական *y-ների* արժեքների միջինացված տարբերություն:

Բանաձևը ներկայացված է ստորև.

$$J(\theta_0, \theta_1, \dots \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2$$

Ավելի պարզ այն կարող ենք գրել հետևյալ կերպ՝ $\frac{1}{2}\bar{x}$, որտեղ $\bar{x}$*-ը* $(h_\theta(x_i) - y_i)$*-ի* քառակուսային միջինն է, այսինքն՝ գուշակված և իրական արժեքի տարբերությունը:

Այս ֆունկցիան նաև կոչվում է քառակուսային սխալի ֆունկցիա (Squared error function): Քառակուսային միջինը բաժանվել է 2-ի՝ հետագա հաշվարկների հարմարավետության համար, քանի որ դրա միջոցով $(h_\theta(x_i) - y_i)^2$*-ի* ածանցումից ստացված *2* բազմապատիկը կվերանա:

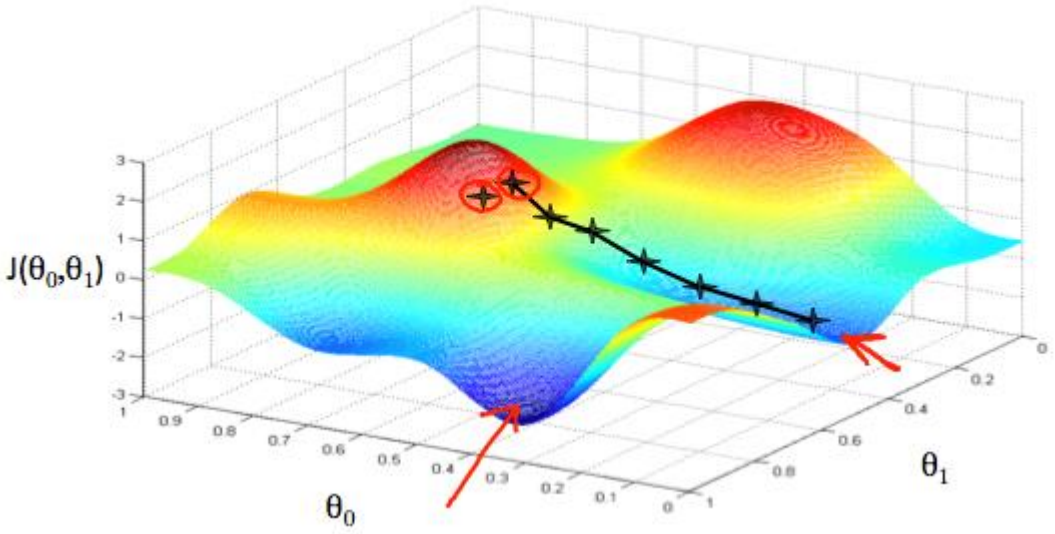
Ստացվեց, որ մեր խնդիրը կայանում է $J(\theta_0, \theta_1, \dots \theta_n)$ *-ը* մինիմիզացնելու մեջ, որն ավելի ֆորմալ կարող ենք ներկայացնել հետևյալ կերպ՝

$$\underset{\theta_0, \theta_1, \dots \theta_n}{minimize} J(\theta_0, \theta_1, \dots \theta_n)$$

1.2.2 Նվազող գրադիենտ

Այսիպսով արդեն պարզաբանվեց, թե ինչ է հիպոթեզ ֆունկցիան և թե ինչպես կարելի է չափել նրա ճշտությունը: Այժմ անհրաժեշտ է որոշել հիպոթեզի պարամետրերը:

Դիտարկենք հիպոթեզ ֆունկցիայի պարզեցված օրինակ, որն ունի ընդհամենը 2 պարամետր՝ θ_0 և θ_1: Պատկերենք այդպիսի հիպոթեզի արժեքի ֆունկցիայի օրինակ (Նկ. 2):



Նկ. 2 2 պարամետրով հիպոթեզի ֆունկցիայի օրինակ

Այստեղ պետք է հստակ պատկերացնել, որ մենք չենք գծում հիպոթեզի գրաֆիկը, այլ փոխարենը գծում ենք նրա **արժեքի ֆունկցիայի** գրաֆիկը, որը ցույց է տալիս, թե θ_0-ի և θ_1-ի արժեքների համար ինչքանով է հիպոթեզը շեղված սպասվելիք արժեքներից: Հասկանալի է, որ պետք է գտնել տվյալ գրաֆիկի վրայի ամենացածր կետը, որի θ_0 և θ_1 արժեքներն էլ հենց կլինեն մեր հիպոթեզի որոնելի պարամետրերի արժեքները (վերևի նկարում կարմիր սլաքներով նշված են տվյալ գրաֆիկի մինիմումները): Քանզի արժեքի ֆունկցիան հիմնականում իրենից ներկայացնում է բարդ մաթեմատիկական բանաձև, այն դժվար է գծել, կամ գտնել, թե θ-ի որ արժեքների դեպքում է այն ընդունում մինիմալ արժեք: Հենց այս խնդիրը լուծելու համար օգտագործվում է նվազող գրադիենտը (Gradient Descent):

Նշվածն իրականացնելու համար կօգտագործենք արժեքի ֆունկցիայի ածանցյալը: Ածանցյալը ցույց է տալիս տվյալ կետում շոշափողի ուղղությունը, ինչն էլ ինֆորմացիա է

տալիս այն մասին, թե որ ուղղությամբ պետք է շարժվել: Ամեն քայլին շարժվում ենք այն ուղղությամբ, որն ամենաշատն է նվազեցնում արժեքի ֆունկցիան:

Յուրաքանչյուր քայլի չափը որոշվում է α պարամետրի միջոցով, որը կոչվում է ուսուցման գործակից (learning rate): Օրինակ, վերը նշված գրաֆիկում յուրաքանչյուր «աստղի» հեռավորությունը ներկայացնում է քայլի հեռավորությունը՝ պայմանավորված α պարամետրով: Փոքր α-ն համապատասխանում է փոքր քայլի, իսկ մեծը՝ մեծ քայլի: Քայլի ուղղությունը, որոշվում է $J(\theta_0, \theta_1)$-ի մասնակի ածանցյալով: Կախված այն բանից, թե որտեղից ենք սկսում դիտարկել գրաֆիկը, հնարավոր է տարբեր մինիմումների հասնել: Վերևում պատկերված են երկու տարբեր սկզբնակետեր (վերցված են կարմիր շրջանագծերի մեջ), որոնք հասնում են երկու տարբեր մինիմումների:

Ընդհանուր դեպքի համար նվազող գրադիենտի ալգորիթմը կլինի. կրկնել հենյալը մինչև զուգամիտում՝ $\theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1, \ldots \theta_n)$ որտեղ՝ $j = 0, 1, \ldots n$ ներկայացնում է հատկության հերթական համարը: Այն անվանում են նաև թարմացման կանոն (update rule): Մեր օրինակի համար՝ $n = 1$:

Յուրաքանչյուր իտերացիային պետք է միաժամանակ թարմացնել բոլոր $\theta_0, \theta_1, \ldots \theta_n$ պարամետրերը: Այսինքն նախ տվյալ իտերացիայի համար հաշվարկել բոլոր θ-ների արժեքները՝ θ', որից հետո θ-ին վերագրել θ': Եթե կամայական θ_j-ի արժեքը թարմացնենք նախքան բոլոր θ-ների արժեքները հաշվարկելը, ապա կստանանք սխալ պատասխան:

Պետք է հաշվի առնել, որ կարևոր է α-ի ճիշտ ընտրությունը, քանզի դրանով է պայմանավորված ալգորիթմի զուգամիտման ժամանակը: Եթե ալգորիթմը չի զուգամիտում կամ շատ ժամանակ է պահանջում մինիմումին հասնելու համար ապա α քայլաչափը սխալ է ընտրված:

Այստեղ կարող է հարց առաջանալ, թե արդյո՞ք հնարավոր է հասնել մինիմումի՝ α-ի անփոփոխ արժեքի դեպքում: Պատասխանը պարզ է դառնում, երբ հաշվի ենք առնում այն հանգամանքը, որ, քանզի ամեն քայլ անելուց մենք ավելի ենք իջնում արժեքի ֆունկցիայի մակերևույթով ներքև, հետևաբար ամեն քայլի հետ մեկտեղ ածանցյալի արժեքը նվազում է: Իսկ դա նշանակում է, որ անգամ, եթե α-ն հաստատուն պահենք, այնուամենայնիվ

$\alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1, \dots \theta_n)$ արտադրյալը ամեն քայլին կնվազի և հասնելով որևէ մինիմումի այն կհավասարվի *0-ի* (իրականում 0-ի չի հավասարվում, այլ մոտենում է ինչ-որ շատ փոքր թվի, որը մեր խնդրի համար համարվում է բավարար) և հետագա քայլերը ոչ մի կերպով չեն ազդի *θ- ների* արժեքների վրա:

Հեշտությամբ կարելի է համոզվել, որ, եթե մեր հիպոթեզն ունի գծային տեսք՝

$$h_\theta(x) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n$$

ապա թարմացման կանոնի մեջ *J(θ)-ի* արժեքը տեղադրելուց հետո թարմացման կանոնի տեսքը կլինի՝

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)} \right) \cdot x_j^{(i)}$$

որտեղ՝ j:=0…n:

Այստեղ և հետագայում կընդունենք, որ $x_0^{(i)} = 1$, բոլոր *i-երի* համար: Սա արվում է բանաձևերը հարմար ներկայացնելու համար: Ստացվեց, որ գծային հիպոթեզն ունի հետևյալ տեսքը՝

$$h_\theta(x) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n$$

1.2.3 Ուսուցման գործակից

Նվազող գրադիենտն իրականացնելուց հետո անհրաժեշտ է հետևել ալգորիթմի աշխատանքին (մոդելի ուսուցման պրոցեսին) և հասկանալ արդյո՞ք այն ճիշտ է աշխատում:

Պատկերացում կազմելու համար, թե ինչքան լավ է սովորում մոդելը, անհրաժեշտ է գծել արժեքի ֆունկցիայի՝ *J(θ)-ի,* կախումը *իտերացիաների քանակից:* Պարզ է, որ, եթե ամեն ինչ ճիշտ է աշխատում, ապա ամեն իտերացիայից հետո *J(θ)-ի* արժեքը պետք է նվազի՝ ձգտելով *0-ի:* Հետևաբար, եթե գրաֆիկը աճում է, ապա ինչ որ բան այն չէ: Հիմնականում դրա պատճառը *α-ի* մեծ արժեքն է լինում. անհաժեշտ է նվազեցնել *α-ի* արժեքը:

Հարկ է նշել՝ ապացուցված է, որ, եթե ուսուցման գործակից (Learning Rate) α-ն բավարար չափով փոքր է ընտրված, ապա $J(\theta)$-ն նվազում է ամեն իտերացիայի: Սակայն, եթե այն շատ փոքր է ընտրված, ապա $J(\theta)$-ն կարող է շատ դանդաղ նվազել:

Կարելի է համարել որ մոդելը բավարար չափով ուսուցանվել է այն պահին, երբ $J(\theta)$-ի փոփոխությունն ինչ-որ իտերացիայից հետո ավելի փոքր է որևէ E արժեքից: E-ն կամայապես ընտրված փոքր թիվ է, օրինակ՝ 10^{-3}: Գործնականում դժվար է ընտրել E-ի օպտիմալ արժեք:

1.2.4 Հատկության մասշտաբավորում

Մենք կարող ենք արագացնել նվազող գրադիենտի աշխատանքը՝ բերելով բոլոր մուտքային պարամետրերը մոտավորապես նույն տիրույթի թվերի: Դա կապված է այն բանի հետ, որ որ θ-ն ավելի արագ է հասնում մինիմումին փոքր միջակայքերում և ավելի դանդաղ՝ մեծ միջակայքերում, հետևաբար այն տատանվելով է այն տատանվելով է ձգտում մինիմումին, երբ փոփոխականները շատ անհավասար են:

Դա կանխելու համար կարող ենք այնպես փոփոխել հատկությունները (մուտքային պարամետրերը), որ նրանք ընկնեն մոտավորապես միևնույն թվային տիրույթ: Իդեալական դեպքում՝

$$-1 < x_i < 1$$

կամ՝

$$-0.5 < x_i < 0.5$$

Սրանք պարտադիր պահանջներ չեն, մենք ընդամենը փորձում ենք կրճատել հաշվարկների ժամանակը: Նպատակն է՝ բերել բոլոր մուտքային փոփոխականները միևնույն տիրույթի:

Հարկ է նշել նաև, որ, եթե չկատարվի հատկությունների մասշտաբավորում, ապա որոշ դեպքերում հնարավոր է, որ ալգորիթմը երբեք չզուգամիտի:

Հատկության մասշտաբավորումն[10] (Feature Scaling) ու միջինով նորմալացումը (mean normalization) այն երկու մեթոդներն են, որոնք կօգնեն լուծել այդ խնդիրը: Առաջինը

ենթադրում է մուտքային տվյալների բաժանում նրանց մեծագույն և փոքրագույն արժեքների տարբերության վրա: Միջինով նորմալացման դեպքում պետք է մուտքային փոփոխականից հանել մուտքային տվյալների միջին արժեքը, ապա նոր բաժանել մեծագույն և փոքրագույն արժեքների տարբերության վրա: Ստացվեց, որ այս երկու մեթոդների իրականացման համար անհրաժեշտ է փոփոխել մուտքային պարամետրերը՝ համապատասխան ներքևի բանաձևի.

$$x_i := \frac{x_i - \mu_i}{s_i}$$

որտեղ s_i-ն i-րդ հատկության մեծագույն և փոքրագույն արժեքների տարբերությունն է, իսկ μ_i-ն՝ այդ հատկության բոլոր արժեքների միջինը: Նշենք, որ s_i-ին կարող ենք ընդունել հավասար մրջին քառակուսային շեղմանը, և այդ դեպքում ստացված արժեքները կտարբերվեն նախորդ տարբերակով ստացված արժեքներից:

Նշվածի օրինակ կարող է ծառայել հետևյալը՝ եթե x_i-ն ներկայացնում է բնակելի տան բարձրություն, և գտնվում է *4-ից 34* միջակայքում, իսկ այդ հատկության բոլոր արժեքների միջինը հավասար է *18-ի,* ապա $x_i := \frac{\text{արժեք}-18}{30}$:

1.3 Պոլինոմալ ռեգրեսիա

Բնականաբար հիպոթեզ ֆունկցիան կարող է լինել կամայական տեսակի: Նրա տեսքը պարզելու համար անհրաժեշտ է կատարել տվյալների ուսումնասիրություն: Եթե ուսումնասիրությունից հետո պարզվում է, որ հիպոթեզը չպետք է լինի գծային, ապա կարևոր է իմանալ, որ հնարավոր է ձևափոխել այն քառակուսայինի, խորանարդայինի կամ այլ տեսքի կորի:

Օրինակ, եթե մեր հիպոթեզ ֆունկցիան ունի հետևյալ տեսքի է՝

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

ապա կարելի է ստեղծել նոր հատկություններ՝ հիմնված x_1-ի վրա այնպես, որ ստանանք քառակուսային ֆունկցիա՝

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2$$

կամ՝ խորանարդային ֆունկցիա՝

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^3$$

Այս օրինակներում ստեղծեցինք նոր՝ x_2 և x_3, հատկություններ, որտեղ $x_2 = x_1^2$, իսկ $x_3 = x_1^3$:

Այն քառակուսի արմատի տեքի դարձնելու համար, կարելի է կատարել հետևյալ ձևափոխությունը՝

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 \sqrt{x_1}$$

Նշված ոչ գծային հիպոթեզները պոլինոմալ ռեգրեսիայի (Polynomial Regression) օրինակներ են:

Շատ կարևոր է հիշել, որ նշված կերպով հատկություններ ավելացնելիս շատ կարևոր է կատարել հատկությունների մաշտաբավորում, քանի որ հատկությունների տիրույթներն իրարից խիստ տարբերվելու են:

Օրինակ, եթե x_1-ը 1-1,000 տիրույթում է, ապա x_1^2-ն կլինի 1-1,000,000, իսկ x_1^2-ը՝ 1-1,000,000,000:

1.4 Դասակարգում

Որպես դասակարգման խնդիր լուծելու մեթոդ կարելի է օգտագործել գծային ռեգրեսիան և 0.5-ից մեծ գուշակված արժեքներն ընդունել որպես 1, իսկ դրանից փոքրերը՝ 0: Սակայն այս մեթոդը լավ չի աշխատում, քանի որ դասակարգումն հիպոթեզն իրականում գծային ֆունկցիա չէ: Այն ռեգրեսիայի խնդիր է, սկսյան այն տարբերությամբ, որ նրա արժեքները վերջավոր քանակի դիսկրետ արժեքներ են:

Մինչ ավելի բարդ դեպքերի անցնելը, կենտրոնանանք երկուական դասակարգման խնդրի (binary classification problem) վրա, որտեղ *y*-ը կարող է ընդունել միայն 2 արժեք՝ *0* և *1:* Օրինակ, եթե պետք է ստեղծել սպամ-նամակների գտնող մոդել, ապա նրա մուտքին տրված ամեն մի նամակի $x^{(i)}$ հատկությունների համար ելքը կարող է լինել 1, եթե այն սպամ է, և 0՝ հակառակ դեպքում:

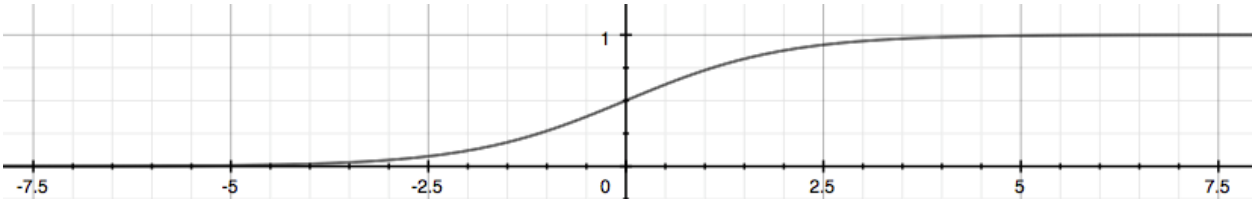
Դասակարգման խնդրի լուծելու համար կարող ենք անտեսել այն հանգամանքը, սպասվող ելքը վերջավոր, դիսկրետ արժեքներ են և օգտագործենք գծային ռեգրեսիան այս խնդրի լուծման համար: Սակայն այս դեպքում անգամ անիմաստ են $h_\theta(x)$-ի 1-ից մեծ և 0-ից փոքր արժեքները, քանի որ մենք գիտենք, որ $y \in \{0, 1\}$: Սրան լուծում տալու համար կձևափոխենք $h_\theta(x)$-ն այնպես, որ նա բավարարի $0 \leq h_\theta(x) \leq 1$ պայմանը: Դա անելու համար կարելի է լոգիստիկ ֆունկցիային (Logistic Function) փոխանցել $\theta^T x$-ը՝

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

Այստեղ g-ն հենց այն լոգիստիկ ֆունկցիան է, որը կամայական իրական թիվ համապատասխանեցնում է (0, 1) տիրույթի որևէ թվի, ինչը թույլ է տալիս կամայական տիրույթի ելքային արժեքներ ունեցող ֆունկցիան փոխակերպել դասակարգման խնդրին ավելի հարմար ֆունկցիայի:

Լոգիստիկ ֆունկցիան նաև անվանում են Սիգմոիդ ֆունկցիա (Sigmoid Function):

Նկ. 3-ում պատկերված է այդպիսի ֆունկցիայի մի օրինակ:



Նկ. 3 Սիգմոիդ ֆունկցիայի օրինակ

Այսպիսով $h_\theta(x)$-ը հավանականությունն է այն բանի, որ ելքային արժեքը հավասար է *1*-ի: Օրինակ, եթե $h_\theta(x) = 0.7$, ապա նշանակում է, որ ելքային արժեքի 1 լինելու հավանականությունը 70% է: Բնականաբար ելքային արժեքի 0 լինելու հավանականությունը հավասար է $(1 - h_\theta(x))$-ի, այսինքն տվյալ օրինակի դեպքում՝ *30%:*

Այս ամենն ավելի ֆորմալ տեսքով կարող ենք գրել այսպես.

$$h_\theta(x) = P(y = 1|x; \theta) = 1 - P(y = 0|x; \theta)$$

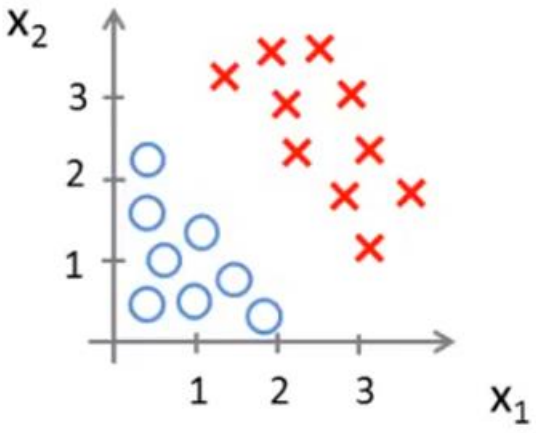
1.4.1 Որոշման սահման

Քանի որ $h_\theta(x)$-ի արդյունքը $y=1$ պայմանի հավանականություն է, անհրաժեշտ է ընտրել մի սահման, և ընդունել, որ այդ սահմանից բարձր $h_\theta(x)$-երի համար $y=1$, հակառակ դեպքում՝ $y=0$: Օրինակ, եթե համարենք, որ $y=1$, երբ $h_\theta(x)>0.5 \Rightarrow g(\theta^T x)>0$, ապա նայելով սիգմոիդ ֆունկցիայի գրաֆիկին, կարող ենք ասել, որ այդ դեպքում $\theta^T x$-ը պետք է մեծ լինի 0-ից:

Ասվածն ավելի պարզ հասկանալու համար դիտարկենք հետևյալ օրինակը. դիցուք՝

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

իսկ ուսուցման տվյալները Նկ. 4-ում պատկերված տեսքն ունեն:

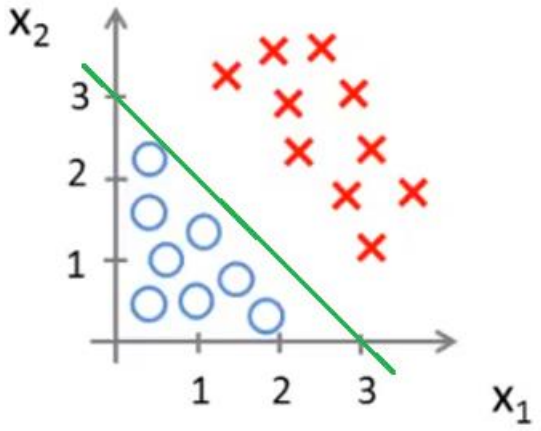


Նկ. 4 Ուսուցման տվյալների (training set) օրինակ №1

ինչպես նաև ենթադրենք, թե ուսուցման վերջում ստացել ենք, որ $\theta_0=-3$, $\theta_1=1$, $\theta_2=1$, կամ մատրիցի տեսքով՝

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

ապա ստացվում է, որ $y=1$, երբ $-3 + x_1 + x_2 \geq 0$, պարզագույն ձևափոխություններից հետո ստանում ենք $x_2 \geq - x_1 + 3$, ինչն իրենից ուղիղ գծի հավասարում է ներկայացնում: Վերջինիս գրաֆիկը գծված է Նկ. 5-ում՝ կանաչ գույնով:

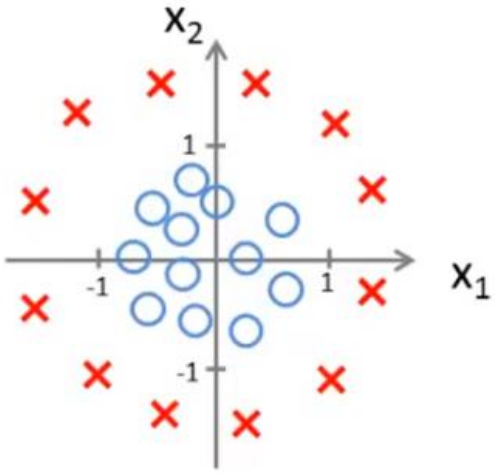


Նկ. 5 Գծային տեսքի որոշման սահմանի օրինակ

Ստացվեց, որ այս կանաչ գծից վերև բոլոր (x_1, x_2) զույգի համար՝ $y=1$: Նույն կերպ նրանից ներքև բոլոր (x_1, x_2) զույգի համար՝ $y=0$:

Հենց այս գիծն էլ կոչվում է **որոշման սահման (Decision Boundary),** քանի որ այն ներկայացնում է մի սահման, որը բաժանում է *y=1-երի* խումբը *y=0-երի* խմբից:

Դիտարկենք մեկ այլ դեպք:



Նկ. 6 Ուսուցման տվյալների (training set) օրինակ № 2

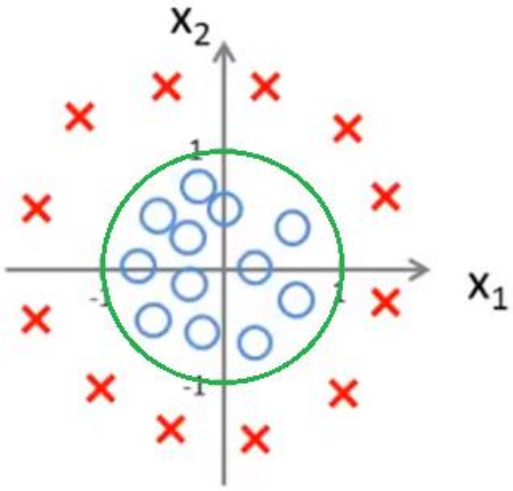
Նկ. 6-ում պատկերված են ուսուցման տվյալները: Պարզ է, որ այստեղ որոշման սահմանը չունի գծային տեսք: Դիցուք այս կոնկրետ օրինակի համար հիպոթեզն ունի հետևյալ պոլինոմալ ֆունկցիայի տեսքը՝

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

իսկ ուսուցման վերջում ստացվել է՝

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Հետևաբար ստացվում է, որ *y=1,* երբ՝ *-1 + x_1^2 + x_2^2 ≥ 0,* որտեղից ստանում ենք, որ *x_1^2 + x_2^2 ≥ 1,* ինչն էլ 1 շառավղով, (0,0) կենտրոնով շրջանագծի հավասարումն է: Որոշման սահմանը կունենա Նկ. 7-ում պատկերված տեսքը:



Նկ. 7 Ոչ գծային որոշման սահմանի օրինակ

Գծված շրջանագծից, դուրս բոլոր *(x_1,x_2)* զույգերի համար *y=1,* իսկ նրա ներսում՝ *y=0:*

Կախված հիպոթեզ ֆունկցիայի բարդությունից, որոշման սահմանը կարող է լինել տարբեր տեսքի:

1.4.2 Լոգիստիկ հիպոթեզի արժեքի ֆունկցիան

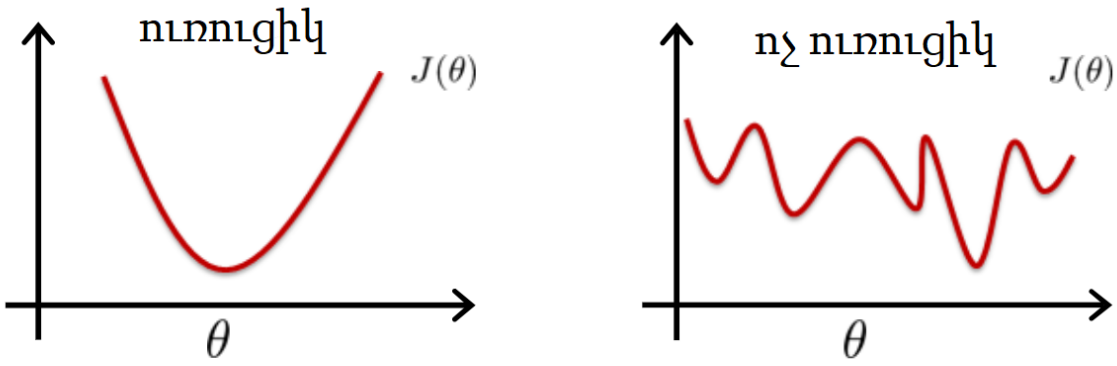
Ընդհանուր դեպքում արժեքի ֆունկցիան ունի հետևյալ տեսքը՝

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x_i), y_i)$$

որտեղ *Cost*-ը այն ֆունկցիան է, որը հաշվում է արժեքը *i*-րդ ուսուցման օրինակի համար:

Արդեն նշվել է, որ գծային հիպոթեզի դեպքում՝ $Cost(h(x_i), y_i) = (h(x_i), y_i)^2$, սակայն լոգիստիկ ֆունկցիայի համար չի կարելի օգտագործել նույն բանաձը, քանի որ այդ կերպ ստացված J ֆունկցիան ալիքային տեսքի է և հետևաբար ունի բազմաթիվ լոկալ մինիմումներ, որոնք բարդացնում են գլոբալ մինիմումը գտնելը: Այլ կերպ ասած *J*-ի գրաֆիկը ուռուցիկ չի լինի:

Ասվածն ավելի լավ պատկերացնելու համար Նկ. 8-ում բերված են ուռուցիկ և ոչ ուռուցիկ ֆունկցիաների գրաֆիկների օրինակներ:

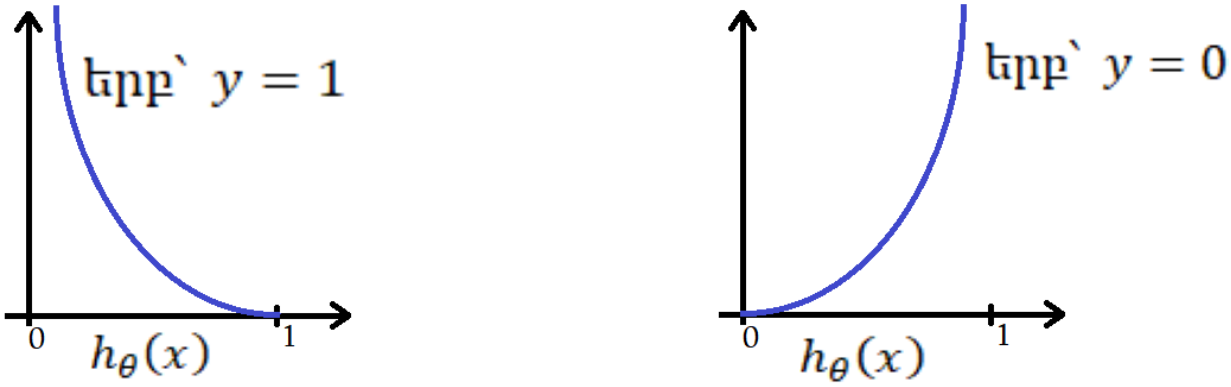


Նկ. 8 Ուռուցիկ և ոչ ուռուցիկ ֆունկցիաների գրաֆիկների որինակ

Փոխարենը կարելի է օգտագործել հետևյալ ֆունկցիան՝

$$\begin{cases} Cost(h_\theta(x), y) = -\log\big(h_\theta(x)\big) & \text{երբ՝ } y = 1 \\ Cost(h_\theta(x), y) = -\log\big(1 - h_\theta(x)\big) & \text{երբ՝ } y = 0 \end{cases}$$

Այս դեպքում ստանում ենք Նկ. 9-ում պատկերված գրաֆիկները:



Նկ. 9 Լոգարիթմական ֆունկցիաների գրաֆիկներ

Այստեղից երևում է, որ, եթե արժեքի ֆունկցիան գրենք այս ձևով, ապա համոզված կարող ենք ասել, որ *J-ն* ունի ուռուցիկ տեսք լոգիստիկ ռեգրեսիայի համար: Ինչը շատ կարևոր է ավելի արագ ուսուցանվող և ճիշտ արդյունքներ գուշակող մոդել ստեղծելու համար:

Ելնելով գրաֆիկից կարող ենք ասել.

- Երբ y=0, ապա արժեքի ֆունկցիան կլինի 0, միայն, եթե հիպոթեզի ֆունկցիայի ելքում նույնպես ստացվի 0: Եթե հիպոթեզը ձգտում է 1-ի, ապա արժեքի ֆունկցիան կձգտի անվերջության:
- Երբ y=1, ապա արժեքի ֆունկցիան կլինի 0, միայն, եթե հիպոթեզի ֆունկցիայի ելքում նույնպես ստացվի 1: Եթե հիպոթեզը ձգտում է 0-ի, ապա արժեքի ֆունկցիան կձգտի անվերջության:

Ասվածը մաթեմատիկորեն կարող ենք ներկայացնել հետևյալ կերպ՝

- $h_\theta(x_i) = y \Rightarrow Cost(h_\theta(x_i)\, y_i) = 0$
- $\begin{cases} y = 0 \\ h_\theta(x_i) \to 1 \end{cases} \Rightarrow Cost(h_\theta(x_i)\, y_i) \to \infty$
- $\begin{cases} y = 1 \\ h_\theta(x_i) \to 0 \end{cases} \Rightarrow Cost(h_\theta(x_i)\, y_i) \to \infty$

Լոգիստիկ հիպոթեզի արժեքի ֆունկցիայի համակարգը կարելի է փոխարինել մեկ արտահայտությամբ հետևյալ կերպ՝

$$Cost(h_\theta(x), y) = -y \log\big(h_\theta(x)\big) - (1 - y) \log\big(1 - h_\theta(x)\big)$$

Հետևաբար *J-ն* կունենա հետևյալ տեսքը՝

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) - \left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right]$$

Կատարելով մաթեմատիկական ձևափոխություններ կարելի է համոզվել, որ այս դեպքում թարմացման կանոնը կլինի նույնն ինչ գծային ռեգրեսիայի համար՝

$$\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)\cdot x_j^{(i)}$$

որտեղ j=0,1,…n:

1.4.3 Բազմադաս դասակարգում

Դասակարգման խնդիրներ քննարկելիս մինչ այս պահը դիտարկել ենք միայն երկուական դասակարգիչ, այսինքն հնարավոր էր միայն 2 ելք՝ *y={0,1}:* Հիմա կդիտարկենք տվյալների դասակարգումը, երբ առկա են երկուսից ավելի կատեգորիաներ: Վերջինս անվանում են բազմադաս դասակարգում (Multiclass Classification): Այսինքն՝ *y={0,1}-ի* փոխարեն ունենք *y={0,1, ... k}:*

Քանի որ *y={0,1, ... k},* ապա կբաժանենք խնդիրը *(k+1)* երկուական դասակարգման խնդիրների և ամեն մեկում կգուշակենք, թե ինչքան է հավանականությունն այն բանի, որ *y-ը* հերթական խմբի անդամն է՝

$$y \in \{0,1,\dots k\}$$

$$h_\theta^{(i)}(x) = P(y = i|x;\theta)$$

այստեղ *i=0, 1, … k* հերթական կատեգորիայի համարն է, իսկ $h_\theta^{(i)}(x)$-ը *y-ի i-րդ* կատեգորիայում գտնվելու հավանականությունն է: Հետևաբար գուշակելու համար, թե տրված մուտքային *x* օրինակին, ո՞ր կատեգորիան է համապատասխանում, անհրաժեշտ է $h_\theta^{(0)}(x)$, $h_\theta^{(1)}(x)$, … $h_\theta^{(k)}(x)$-ից ընտրել մեծագույնը: Սա մաթեմատիկորեն կգրենք այսպես՝

$$\text{գուշակված դասը} = \max\left(h_\theta^{(i)}(x)\right), \qquad i = 0,1,\dots k$$

1.5 Նորմալ հավասարում

Նվազող գրադիենտը *J(θ)-ն* մինիմիզացնելու տարբերակներից մեկն է: Հիմա կդիտարկենք մեկ այլ տարբերակ, որը հնարավորություն կտա մինիմիզացնել *J(θ)-ն,* առանց որևէ իտերացվող ալգորիթմի: Խոսքը նորմալ հավասարման մասին է, որը հնարավորություն է տալիս գտնել որոնելի *θ-ների* արժեքներն առանց իտերացիայի: Բանաձևը հետևյալն է՝

$$\theta = (X^T X)^{-1} X^T Y$$

որտեղ *X-ը (m x (n+1)չափի)* ուսուցման տվյալների մատրիցն է, *Y-ը՝ (m x 1չափի)* ամեն մի ուսուցման օրինակի համապատասխան ելքային արժեքը, իսկ X^T-ն *X-ի* տրանսպոզիցիան է: Այստեղ *X-ը m x (n+1)* չափի է, քանի որ սկզբնական *X* մատրիցին պետք է ավելացնել ամբողջությամբ 1-երով լցված սյունը, որն էլ հենց ամեն ուսուցման տվյալի x_0 արժեքն է:

Հարկ է նշել, որ այս դեպքում պետք չէ կատարել հատկությունների մասշտաբավորում:

Ներքևում բերված է նվազող գրադիենտի և նորմալ հավասարման համեմատության աղյուսակ.

Նվազող գրադիենտ	Նորմալ հավասարում
Պետք **է** ընտրել α	Պետք **չէ** ընտրել α
Անհրաժեշտ է մի քանի իտերացիա	Առանց իտերացիայի
Բարդությունը՝ $O(kn^2)$	Բարդությունը՝ $O(n^3)$
Լավ է աշխատում, երբ *n-ը* շատ մեծ է	Դանդաղ է, երբ *n-ը* շատ մեծ է

Նորմալ հավասարումը[10] (Normal Equation) մատրիցի հակադարձ, տրանսպոզիցիա և բազմապատկում կատարելու հետ է կապված, այդ պատճառով նրա բարդությունը $O(n^3)$ է: Այդ պատճառով *n-ի* մեծ արժեքների դեպքում այն դանդաղ է աշխատում: Գործնականում, երբ *n-ը* գերազանցում է 10,000-ը ավելի լավ է նորմալ հավասարումից անցնել իտերացվող ալգորիթմի:

Հնարավոր է նաև ունենալ այնպիսի մուտքային տվյալների մատրից, որը չունի հակադարձ (անհակադարձելի է): Նշվածի հիմնական պատճառներ կարող են լինել՝

- Ավելորդ հատկությունների առկայությունը, երբ 2 հատկություններ շատ սերտ կապի մեջ են, այսինքն գտնվում են գծային կախվածության մեջ
- Չափից դուրս շատ հատկությունների առկայությունը՝ $m \leq n$: Այս դեպքում կարելի է հեռացնել որոշ հատկություններ, կամ օգտագործել ,կանոնավորումըե, որը կմանրամասնենք հետագայում

Նշված խնդիրների լուծումն կարող է լինել որոշ հատկությունների հեռացումը, որոնք գծային կախման մեջ են գտնվում մեկ այլ հատկությունից կամ պարզապես որոշ՝ քիչ կարևոր, հատկությունների հեռացումը, երբ առկա են մեծ քանակի հատկություններ:

1.6 Նեյրոնային ցանցեր

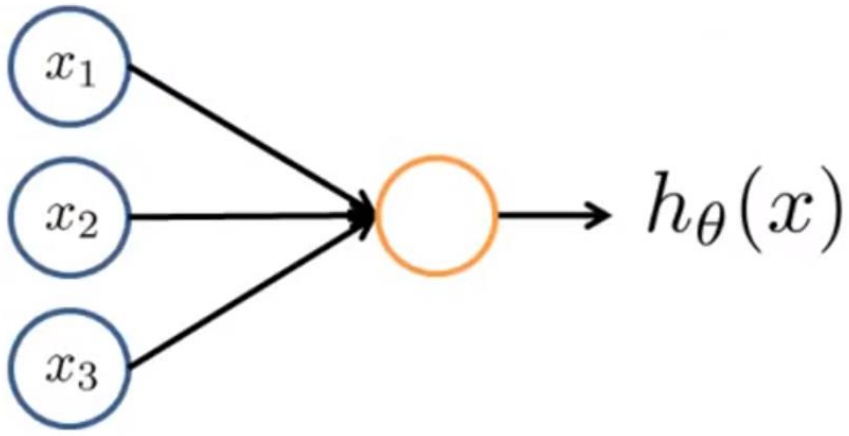
Հասկանալու համար, թե ինչ անհրաժեշտություն կա ուսումնասիրել այլ ուսուցման ալգորիթմ՝ նեյրոնային ցանցեր[11] (Neural Networks), պատկերացնենք մի դեպք, երբ դասակարգման խնդիր լուծելիս հատկությունները բավարար չեն ճշգրիտ մոդել ուսուցանելու համար, և անհրաժեշտություն է առաջացել ավելացնել նոր՝ քառակուսային, խորանարդային կամ այլ,հատկություններ: Այս դեպքում եթե ավելացնենք բոլոր քառակուսային հատկությունները՝

$$x_1^2, x_1x_2, x_1x_3, \dots x_1x_n$$
$$x_2^2, x_2x_3, \dots x_2x_n$$
$$\dots$$
$$x_{n-1}^2, x_{n-1}x_n$$
$$x_n^2$$

ապա կստանանք $\frac{n\cdot(n+1)}{2} + n$ քանակի հատկություն: Այսինքն ստացվում է, որ նոր հատկությունների քանակը նախկինից մոտավորապես քառակուսային $\left(\approx \frac{n^2}{2}\right)$ կախում ունի: Նույն ձևով խորանարդային հատկություններ ավելացնելիս կարելի է համոզվել որ կախումը խորանարդային է: Սա կարող է բերել գերհամապատասխանեցման (overfitting) խնդրին ինչպես նաև բավականաչափ մեծ հաշվողական ռեսուրսներ կպահանջվեն նման մեծ թվով հատկությունների հետ աշխատելու համար:

Մեքենայական ուսուցման մեջ օգտագործվող նեյրոնի մոդելը հնարավորինս մոտ է արված մարդու ուղեղի նեյրոնային կառուցվածքին: Յուրաքանչյուր նեյրոն ստանում է

մուտքին որևէ պարամետրեր, կատարում է որոշակի հաշվարկներ, և արդյուների հիման վրա որոշում է թե ինչ ազդանշան ուղարկի հաջորդ նեյրոնին:



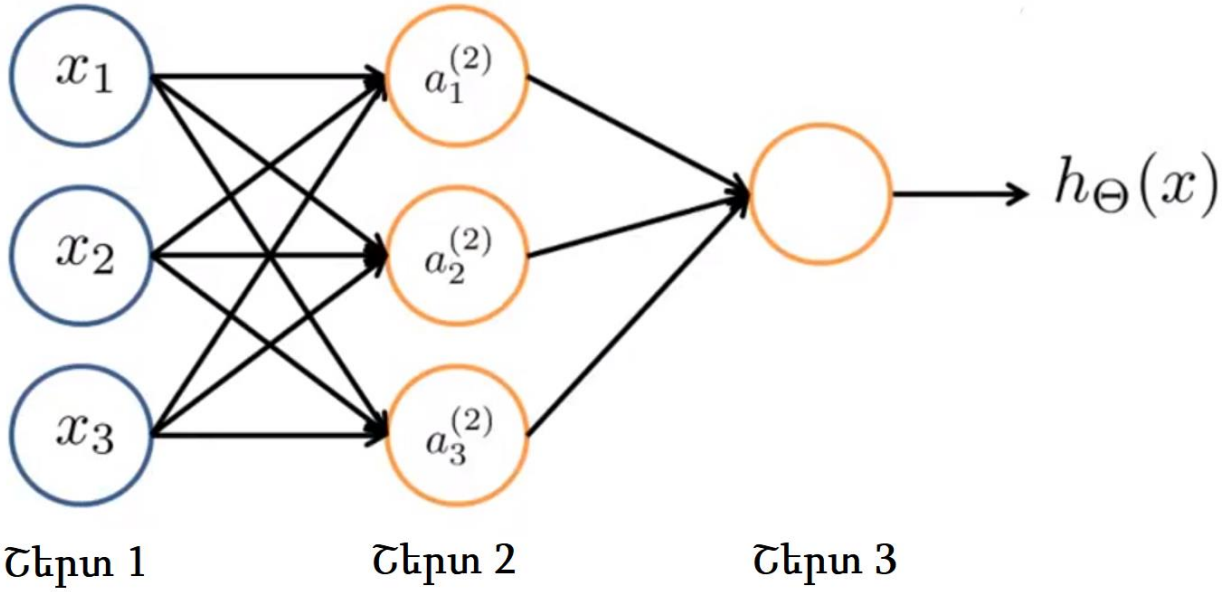
Նկ. 10 Նեյրոնի մոդելի օրինակ

Նկ. 10-ում պատկերված է նեյրոնի պարզեցված մոդելը, որն օգտագործվում է մեքենայական ուսուցման մեջ: x_1, x_2, x_3-ը մուտքային պարամետրերն են, իսկ դեղինով եզրագծվածը նեյրոնի «մարմինն» է: Այստեղ նույնպես կարող ենք ավելացնել $x_0 = 1$ պարամետրը, որը կոչվում է շեղում (bias): Նեյրոնի կատարած հաշվարկների արդյունքը հիպոթեզ ֆունկցիայի արժեքն է,որի բանաձևը լոգիստիկ ռեգրեսիայի բանաձևն է: Նեյրոնի հիպոթեզի ֆունկցիան այլ կերպ անվանում են նաև սիգմոիդ ակտիվացման ֆունկցիա: Նեյրոնի ակտիվացիան դա լոկ այն արժեքն է,որը հաշվարկում է այդ նեյրոնը,այլ կերպ ասած նրա ելքում ստացված արժեքն է: Բնականաբար այդ ֆունկցիան,ինչպես և նախորդ մեր դիտարկած սիգմոիդ ֆունկցիան ունի իր պարամետրերը՝ $\theta_0, \theta_1, \ldots \theta_n$ (մեր օրինակի դեպքում n = 3), որոնց անվանում են կշիռներ (weights):

Նեյրոնային ցանցը, ինչպես բխում է անունից, բազմաթիվ նեյրոններից բաղկացած ցանց է, որոնց ելքերն ու մուտքերը կապված են միմյանց հետ: Նկ. 11-ում պատկերված է նեյրոնային ցանցի մի պարզ օրինակ:

Ցանցը բաժանվում է շերտերի (layers): Առաջին շերտը անվանում են մուտքային շերտ, քանի որ սա այն շերտն է որտեղ գտնվում են հատկությունները: Վերջին շերտը անվանում են ելքային շերտ, այն հաշվարկում է հիպոթեզ ֆունկցիայի վերջնական արժեքը: Առաջին և վերջին շերտերի միջև ընկած բոլոր մնացած շերտերն անվանում են թաքնված շերտեր: Վերջիններս թաքնված են, քանի որ ուսուցման ընթացքում նրանց արժեքներին չենք

հետևում: $a_i^{(j)}$-ով կնշանակված է j-րդ շերտի i-րդ նեյրոնի ,ակտիվացիանե, իսկ $\theta^{(j)}$-ով կնշանակենք կշիռների այն մատրիցը, որը պարունակում է j-ից (j+1) շերտ անցնելու բոլոր ակտիվացիաների ֆունկցիաների պարամետրերը:



Նկ. 11 Նեյրոնային ցանցի պարզ օրինակ

Ընդհանուր դեպքում (j+1)-րդ շերտի նեյրոնների ակտիվացման ֆունկցիաների տեսքը բերված է ներքևում.

$$a_1^{(j+1)} = g(\theta_{10}^{(j)} x_0 + \theta_{11}^{(j)} x_1 + \ldots + \theta_{1n}^{(j)} x_n)$$
$$\cdots$$
$$a_{s_{j+1}}^{(j+1)} = g(\theta_{s_{j+1}0}^{(j)} x_0 + \theta_{s_{j+1}1}^{(j)} x_1 + \ldots + \theta_{s_{j+1}n}^{(j)} x_n)$$

որտեղ n-ը մուտքային պարամետրերի քանակն է, իսկ s_{j+1}-ը՝ (j+1)-րդ շերտում նեյրոնների քանակը: g-ֆունկցիան արդեն պարզաբանվել է 1.4 բաժնում: Նշված ֆունկցիաների օգնությամբ վերջին շերտի արժեքը հաշվելով կարող ենք ստանալ $h_\theta(x)$-ի արժեքը:

Գլուխ 2. Խնդրի դրվածքը

Ուսումնասիրելով գրականությանը կարելի է եզրահանգել նրան, որ ավարտական աշխատանքի շրջանակներում դրվում է խնդիր գեներատիվ մրցակցային ցանցերի միջոցով նկարի տեսքով թաքնագրության կրիչ (կոնտեյներ) ստեղծող համակարգ մշակել:

Այդ նպատակով անհրաժեշտ է ուսուցանել միաժամանակ 3 մոդել՝ գեներատոր, տարբերակիչ, գաղտնավերլուծիչ: Այս մոդելները մրցակցելով միմյանց հետ փորձելու են լավացնել իրենց արդյունքը: Վերջում, երբ կհամարենք, որ մոդելները բավարար չափով ուսուցանված են, գեներացնող մոդելը կկարողանա ստեղծել իրական մարդկանց դեմքերին մոտ այնպիսի նկար-կրիչներ, որոնք կապահովեն բարձր թաքնակայունություն:

2016 թվականին կատարված հետազոտությունները լավ արդյունքներ էին տվել, սակայն գեներացված նկարները մոտ չեին իրական նկարներին: Տվյալ աշխատության մեջ գեներատորին ուսուցանման ժամանակ տրվելու են ինչ-որ հատկանիշներով (սեռ, տարիք, ռասսա) նման մարդկանց նկարներ, ինչը, ենթադրվում է, որ կհանգեցնի իրականին ավելի մոտ նկարների ստեղծմանը:

Գրականություն

1. Steganography An Art of Hiding Data, Shashikala Channalli et al /International Journal on Computer Science and Engineering Vol.1(3), 2009
2. https://en.wikipedia.org/wiki/Steganalysis
3. Generative adversarial nets. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. pp. 2672–2680, 2014.
4. Unsupervised representation learning with deep convolutional generative adversarial networks. Alec Radford, Luke Metz, and Soumith Chintala. arXiv preprint arXiv:1511.06434, 2015.
5. Generative adversarial networks for image steganography. Denis Volkhonskiy, Boris Borisenko and Evgeny Burnaev
6. https://en.wikipedia.org/wiki/Minimax
7. Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
8. Generative adversarial text to image synthesis. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. arXiv preprint arXiv:1605.05396, 2016.
9. https://www.coursera.org/learn/machine-learning/home/week/1
10. https://www.coursera.org/learn/machine-learning/home/week/2
11. https://www.coursera.org/learn/machine-learning/home/week/4