Անհավասար տվյալներ

Հայկ Կարապետյան

1 Անհրաժեշտություն

Մեքենայական ուսուցման մեջ տվյալները շատ մեծ դեր ունեն։ Վատ ուսուցանման տվյալներով և գերազանց ցանցի կառուցվածքով ցանցը լավ ճշգրտություն չի ունենա։ Անհավասար տվյալները նույնպես կարող են խնդիր առաջացնել նեյրոնային ցանցերը ուսուցանելիս։ Դիտարկենք հետևյալ օրինակը. ունենք շան և կատվի նկարներ։ Մուտքում նկարը ստանալիս ցանցը պետք է դասակարգի շուն է պատկերված նկարում, թե կատու։ Շան նկարների քանակը 7000 հատ է, իսկ կատվի նկարների քանակը 3000։ Այսինքն տվյալների 70%-ը շան նկարներ են, 30%-ը՝ կատվի։ Այս դեպքում ցանցը ուսուցանելիս, ավելի շատ կհարմարվենք շան նկարներին, քան կատվի նկարներին։ Այսինքն նոր մուտքային նկար գալու դեպքում, ցանցը ավելի հակված կլինի ասելու շուն, քան կատու։ Ուսուցման ընթացքում 70% ճշգրտությունը կարող է նշանակել, որ ինչ նկար էլ մուտքում եկել է, մենք պիտակավորել ենք, որպես շուն։ Այդ պատճառով անհավասար տվյալները ինչ որ ձև անհրաժեշտ է կարգավորել։

2 Դասակարգման եղանակներ

2.1 Undersampling

Առաջին եղանակը undersampling-ն է։ Այս դեպքում մենք դեն ենք նետում այն class-ի տվյալները, որոնք շատ են և հավասարեցնում ենք քիչ քանակի class-ի տվյալներին։ 7000 նկաից դեն ենք նետում 4000-ը և տվյալների քանակները հավասարվում են։ Այս տարբերակը լավ չէ, քանի որ լավ ցանց ուսուցանելու համար հարկավոր են բավարար քանակով տվյալներ, իսկ մենք ուղղակի դեն ենք նետում դրանք։ Եթե մեր տվյալների քանակը բավականաչափ շատ լինի (700000 շան նկար, 300000 կատվի նկար), այդ դեպքում կարող ենք դեն նետել

2.2 Oversampling

Երկրորդ եղանակը oversampling-ն է։ Այս դեպքում մենք քիչ քանակությամբ նկարները կրկնում ենք այնքան ժամանակ, մինչև հավասարվի շատ քանակի նկարներին։ 3000 նկարները կմիացնենք նույն 3000 նկարներին և 1000 հատ էլ պատահական կերպով կընտրենք կմիացնենք ու կստանանք 7000 նկար։ Բայց նույն նկարները ամեն անգամ կրկնելու փոխարեն, նկարի վրա կարող ենք կատարել որոշ փոփոխություններ (augmentation), հետո ավելացնել սկզբնական նկարներ։ Բայց միայն մի class-ի համար augmentation կատարելը ճիշտ չէ։ Օրինակ կատարել ենք աղմուկի ավելացման (noise adding) augmentation միայն կատուների նկարների համար։ Երբ թեստավորման ժամանակ մուտքում ցանցը ստացավ աղմուկով նկար, այն ավելի հակված է լինելու ասել կատու, քան շուն։ Այդ պատճառով երկուսի վրա էլ կատարում ենք augmenatation, բայց մեկի վրա քիչ չափով մյուսի վրա շատ չափով (կարող ենք շան նկարների 25%-ի վրա կատարել աղմուկի ավելացում, իսկ կատուների՝ բոլոր նկարների)։

2.3 Փոփոխություն կորստի ֆունկցիայում

Երրորդ եղանակը կորստի ֆունկցիայի մեջ փոփոխություն անելն է։ Վերհիշենք binary cross entropy կորստի ֆունկցիան։

$$L = -ylog(f(x)) - (1-y)log(1-f(x))$$

Այս դեպքում երկու class-ն էլ հաշվի ենք առնում նույն չափով

$$L = -\alpha ylog(f(x)) - (1-\alpha)(1-y)log(1-f(x)),\ 0 < \alpha < 1$$

Այս դեպքում, որ class-ի տվյալները քիչ են, դրա գործակիցը կվերցնենք մեծ

Եթե շան պիտակը լինի 1, իսկ կատվինը՝ 0, ապա α-ն կվերցնենք 0.5-ից փոքր, որպեսզի կատվի կորստին ավելի շատ ուշադրություն դարձնենք։ Երկու class-ի դեպքում կունենանք balanced binary cross entropy: Վերհիշենք cross entropy կորուստը մի քանի class-ի դեպքում։

$$L = \frac{1}{n}\sum_{i=1}^{n} -y_i log(f(x_i)),\ y_i, f(x_i) \in R^k,\ k = class\_count$$

Այս դեպքում բոլոր class-երը հաշվի ենք առնում նույն չափով

$$L = \frac{1}{n}\sum_{i=1}^{n} -(\alpha y_i) log(f(x_i))$$

$$\alpha = [\alpha_1, \ldots, \alpha_k],\ \alpha \in (0,1),\ \sum_{i=1}^{k} \alpha_i = 1$$

α-ն հիպերպարամետր է և մենք ենք որոշելու թե ինչ արժեքներից է բաղկացած լինելու այդ վեկտորը։ Առաջին արժեքները, որ կարող ենք փորձարկել հենց տվյալների հարաբերակցությունն է։ Ունենք 3 class (1՝ շուն, 2՝ կատու, 3՝ փիղ)։ Շան նկարները կազմում են ամբողջ տվյալների 50%-ը, կատվի նկարները՝ 25%, փղի նկարները՝ 25%։ Այս դեպքում $\alpha = [0.5, 0.25, 0.25]$: Այս կորստի ֆունկցիան կոչվում է balanced cross entropy։