

[**www.feynlabs.ai**](http://www.feynlabs.ai)

Contents

[Background and References 20](#_gjdgxs)

[Foundations of Machine Learning 21](#_1fob9te)

[Accuracy 21](#_3znysh7)

[artificial general intelligence 21](#_2et92p0)

[artificial intelligence 21](#_tyjcwt)

[Attribute 21](#_3dy6vkm)

[Backpropagation 21](#_1t3h5sf)

[Baseline 22](#_4d34og8)

[Batch 22](#_2s8eyo1)

[batch size 22](#_17dp8vu)

[bias (math) 22](#_3rdcrjn)

[Binning 22](#_26in1rg)

[Boosting 23](#_lnxbz9)

[Broadcasting 23](#_35nkun2)

[Bucketing 23](#_1ksv4uv)

[categorical data 24](#_44sinio)

[Clipping 24](#_2jxsxqh)

[confirmation bias 24](#_z337ya)

[confusion matrix 25](#_3j2qqm3)

[continuous feature 25](#_1y810tw)

[convenience sampling 25](#_4i7ojhp)

[Convergence 25](#_1ci93xb)

[convex function 26](#_3whwml4)

[convex optimization 27](#_2bn6wsx)

[data augmentation 27](#_qsh70q)

[Data set or dataset 28](#_3as4poj)

[decision boundary 28](#_1pxezwc)

[deep model 28](#_49x2ik5)

[deep neural network 28](#_2p2csry)

[Deep Q-Network (DQN) 28](#_147n2zr)

[dense feature 29](#_3o7alnk)

[dense layer 29](#_23ckvvd)

[Depth 29](#_ihv636)

[dimension reduction 29](#_32hioqz)

[Dimensions 29](#_1hmsyys)

[discrete feature 29](#_2grqrue)

[discriminative model 30](#_vx1227)

[discriminator 30](#_3fwokq0)

[Downsampling 30](#_1v1yuxt)

[dropout regularization 30](#_4f1mdlm)

[dynamic model 30](#_2u6wntf)

[eager execution 31](#_19c6y18)

[early stopping 31](#_3tbugp1)

[Embeddings 31](#_28h4qwu)

[embedding space 31](#_nmf14n)

[Ensemble 32](#_37m2jsg)

[Environment 32](#_1mrcu09)

[Episode 32](#_46r0co2)

[Epoch 32](#_2lwamvv)

[equalized odds 32](#_111kx3o)

[Example 34](#_3l18frh)

[experience replay 34](#_206ipza)

[experimenter's bias 34](#_4k668n3)

[fairness constraint 34](#_2zbgiuw)

[false negative (FN) 34](#_1egqt2p)

[false positive (FP) 34](#_3ygebqi)

[false positive rate (FPR) 34](#_2dlolyb)

[Feature 35](#_sqyw64)

[feature engineering 35](#_3cqmetx)

[feature extraction 35](#_1rvwp1q)

[feature set 35](#_4bvk7pj)

[feature spec 35](#_2r0uhxc)

[feature vector 36](#_1664s55)

[federated learning 36](#_3q5sasy)

[feedback loop 36](#_25b2l0r)

[few-shot learning 36](#_kgcv8k)

[fine tuning 36](#_34g0dwd)

[forget gate 36](#_1jlao46)

[Generalization 37](#_43ky6rz)

[generalization curve 37](#_2iq8gzs)

[generative model 37](#_xvir7l)

[Gradient 38](#_3hv69ve)

[gradient clipping 38](#_1x0gk37)

[gradient descent 38](#_4h042r0)

[Graph 38](#_2w5ecyt)

[graph execution 38](#_1baon6m)

[ground truth 38](#_3vac5uf)

[group attribution bias 39](#_2afmg28)

[Hashing 39](#_pkwqa1)

[Heuristic 39](#_39kk8xu)

[hidden layer 39](#_1opuj5n)

[holdout data 39](#_48pi1tg)

[Hyperparameter 40](#_2nusc19)

[Hyperplane 40](#_1302m92)

[image recognition 40](#_3mzq4wv)

[imbalanced dataset 40](#_2250f4o)

[implicit bias 40](#_haapch)

[Inference 40](#_319y80a)

[in-group bias 41](#_1gf8i83)

[Instance 41](#_40ew0vw)

[Interpretability 41](#_2fk6b3p)

[Items 41](#_upglbi)

[Iteration 41](#_3ep43zb)

[Keras 41](#_1tuee74)

[Keypoints 42](#_4du1wux)

[L1 regularization 42](#_2szc72q)

[L2 loss 42](#_3s49zyc)

[L2 regularization 42](#_279ka65)

[Lambda 42](#_meukdy)

[Landmarks 42](#_36ei31r)

[Layer 42](#_1ljsd9k)

[learning rate 43](#_45jfvxd)

[Logits 43](#_2koq656)

[log-odds 43](#_zu0gcz)

[machine learning 43](#_3jtnz0s)

[Mean Squared Error (MSE) 44](#_1yyy98l)

[Metric 44](#_4iylrwe)

[ML 44](#_2y3w247)

[MNIST 44](#_1d96cc0)

[Model 44](#_3x8tuzt)

[model function 44](#_2ce457m)

[model training 45](#_rjefff)

[Momentum 45](#_3bj1y38)

[NaN trap 45](#_1qoc8b1)

[natural language understanding 45](#_4anzqyu)

[neural network 45](#_2pta16n)

[Neuron 45](#_14ykbeg)

[Noise 46](#_3oy7u29)

[Normalization 46](#_243i4a2)

[numerical data 46](#_j8sehv)

[Objective 46](#_338fx5o)

[Objective function 47](#_1idq7dh)

[one-shot learning 47](#_42ddq1a)

[Operation (op) 47](#_2hio093)

[Outliers 47](#_wnyagw)

[Overfitting 47](#_3gnlt4p)

Pandas 47

[Parameter 48](#_1vsw3ci)

[parameter update 48](#_4fsjm0b)

[partial derivative 48](#_2uxtw84)

[participation bias 48](#_1a346fx)

[partitioning strategy 48](#_3u2rp3q)

[Performance 48](#_2981zbj)

[Perplexity 49](#_odc9jc)

Pipeline 49

[Quantile 49](#_38czs75)

[quantile bucketing 49](#_1nia2ey)

[Quantization 49](#_47hxl2r)

[Queue 49](#_2mn7vak)

[rank (ordinality) 50](#_11si5id)

[Recall 50](#_3ls5o66)

[reporting bias 50](#_20xfydz)

[Representation 50](#_4kx3h1s)

[ridge regularization 50](#_302dr9l)

[sampling bias 50](#_1f7o1he)

[Scalar 51](#_3z7bk57)

[Scaling 51](#_2eclud0)

[Scoring 51](#_thw4kt)

[semi-supervised learning 51](#_3dhjn8m)

[sensitive attribute 51](#_1smtxgf)

[Serving 51](#_4cmhg48)

[sigmoid function 52](#_2rrrqc1)

[similarity measure 52](#_16x20ju)

[size invariance 52](#_3qwpj7n)

[Sketching 52](#_261ztfg)

[Softmax 53](#_l7a3n9)

[sparse feature 53](#_356xmb2)

[sparse representation 53](#_1kc7wiv)

[Stationarity 54](#_44bvf6o)

[Step 54](#_2jh5peh)

[step size 54](#_ymfzma)

[stochastic gradient descent (SGD) 55](#_3im3ia3)

[structural risk minimization (SRM) 55](#_1xrdshw)

[Subsampling 55](#_4hr1b5p)

[Summary 55](#_2wwbldi)

[supervised machine learning 55](#_1c1lvlb)

[Target 55](#_3w19e94)

[termination condition 56](#_2b6jogx)

[test set 56](#_qbtyoq)

[time series analysis 56](#_3abhhcj)

[Timestep 56](#_1pgrrkc)

[Training 56](#_49gfa85)

[training set 57](#_2olpkfy)

[Trajectory 57](#_13qzunr)

[transfer learning 57](#_3nqndbk)

[translational invariance 57](#_22vxnjd)

[Trigram 57](#_i17xr6)

[true negative (TN) 58](#_320vgez)

[true positive (TP) 58](#_1h65qms)

[unawareness (to a sensitive attribute) 58](#_415t9al)

[Underfitting 58](#_2gb3jie)

[unlabeled example 58](#_vgdtq7)

[unsupervised machine learning 58](#_3fg1ce0)

[Upweighting 59](#_1ulbmlt)

[user matrix 59](#_4ekz59m)

[Validation 59](#_2tq9fhf)

[validation set 60](#_18vjpp8)

[vanishing gradient problem 60](#_3sv78d1)

[Weight 60](#_280hiku)

[Weighted Alternating Least Squares (WALS) 60](#_n5rssn)

[Regression 61](#_375fbgg)

[attribute 61](#_46ad4c2)

[Binning 61](#_2lfnejv)

[Bucketing 61](#_10kxoro)

[continuous feature 61](#_3kkl7fh)

[Data set or dataset 62](#_2yutaiw)

[Dimensions 62](#_3xzr3ei)

[Ensemble 63](#_sabnu4)

[feature 63](#_3c9z6hx)

[feature engineering 63](#_1rf9gpq)

[feature extraction 63](#_4bewzdj)

[feature](#_2qk79lc)  set 64

[feature spec 64](#_15phjt5)

[feature vector 64](#_3pp52gy)

[Generalization 64](#_24ufcor)

[generalized linear model 65](#_33zd5kd)

[Hyperparameter 65](#_1j4nfs6)

[learning rate 66](#_2i9l8ns)

[linear model 66](#_xevivl)

[linear regression 67](#_3hej1je)

[Model 67](#_2vor4mt)

[model function 68](#_1au1eum)

[model training 68](#_3utoxif)

[Normalization 68](#_29yz7q8)

[numerical data 68](#_p49hy1)

[Optimizer 69](#_1o97atn)

[Parameter 69](#_2ne53p9)

[Prediction 70](#_gtnh0h)

[prediction bias 70](#_30tazoa)

[random forest 70](#_1fyl9w3)

[regression model 71](#_2f3j2rp)

[Regularization 71](#_u8tczi)

[regularization rate 71](#_3e8gvnb)

[ridge regularization 71](#_1tdr5v4)

[Scaling 72](#_4ddeoix)

[Subsampling](https://docs.google.com/document/d/1dssEOKjlUwT8nKLtnVvQEKEfB1khTzTk/edit#heading=h.2sioyqq) 72

[Target 72](#_3rnmrmc)

[Temporal data 72](#_26sx1u5)

[test set 72](#_ly7c1y)

[time series analysis 73](#_35xuupr)

[Training 73](#_1l354xk)

[training set 73](#_452snld)

[Validation 74](#_1yib0wl)

[validation set 74](#_4ihyjke)

[Classification 74](#_2xn8ts7)

[accuracy 74](#_1csj400)

[area under the PR curve 74](#_3ws6mnt)

[area under the ROC curve 75](#_2bxgwvm)

[AUC (Area under the ROC Curve) 75](#_r2r73f)

[Batch 75](#_1q7ozz1)

[batch normalization 75](#_4a7cimu)

[batch size 75](#_2pcmsun)

[Bayesian neural network 76](#_14hx32g)

[binary classification 76](#_3ohklq9)

[Boosting 76](#_23muvy2)

[categorical data 76](#_is565v)

[Class 77](#_32rsoto)

[classification model 77](#_1hx2z1h)

[classification threshold 77](#_41wqhpa)

[class-imbalanced dataset 77](#_2h20rx3)

[Clipping 77](#_w7b24w)

[confusion matrix 78](#_3g6yksp)

[cross-validation 78](#_1vc8v0i)

[Data set or dataset 79](#_4fbwdob)

[decision boundary 79](#_2uh6nw4)

[decision threshold 79](#_19mgy3x)

[decision tree 79](#_3tm4grq)

[Ensemble 80](#_nwp17c)

[false negative (FN) 80](#_37wcjv5)

[false positive (FP) 81](#_1n1mu2y)

[false positive rate (FPR) 81](#_471acqr)

[Feature 81](#_2m6kmyk)

[feature engineering 81](#_11bux6d)

[feature extraction 81](#_3lbifu6)

[feature set 81](#_20gsq1z)

[few-shot learning 82](#_4kgg8ps)

[generalized linear model 82](#_2zlqixl)

[Hashing 82](#_1er0t5e)

[Hyperparameter 83](#_3yqobt7)

[Hyperplane 83](#_2dvym10)

[image recognition 83](#_t18w8t)

[imbalanced dataset 83](#_3d0wewm)

[Kernel Support Vector Machines (KSVMs) 83](#_4c5u7s8)

[Label 85](#_16ges7u)

[labeled example 85](#_3qg2avn)

[logistic regression 85](#_25lcl3g)

[Logits 86](#_kqmvb9)

[majority class 86](#_34qadz2)

[MNIST 86](#_1jvko6v)

[model function 86](#_43v86uo)

[model training 87](#_2j0ih2h)

[multi-class classification 87](#_y5sraa)

[multi-class logistic regression 87](#_3i5g9y3)

[multinomial classification 87](#_1xaqk5w)

[negative class 87](#_4hae2tp)

[Normalization 87](#_2wfod1i)

[numerical data 88](#_1bkyn9b)

[one-hot encoding 88](#_3vkm5x4)

[one-shot learning 88](#_2apwg4x)

[one-vs.-all 88](#_pv6qcq)

[Parameter 89](#_2o52c3y)

[positive class 90](#_3na04zk)

[post-processing 90](#_22faf7d)

[PR AUC (area under the PR curve) 90](#_hkkpf6)

[Precision 90](#_31k882z)

[precision-recall curve 90](#_1gpiias)

[Prediction 91](#_40p60yl)

[prediction bias 91](#_2fugb6e)

[preprocessing 91](#_uzqle7)

[proxy (sensitive attributes) 91](#_3eze420)

[proxy labels 91](#_1u4oe9t)

[rank (ordinality) 92](#_4e4bwxm)

[Recall 92](#_2t9m75f)

[Regularization 92](#_18ewhd8)

[sampling bias 92](#_3sek011)

[sentiment analysis 92](#_mp4kgn)

[sigmoid function 93](#_36os34g)

[size invariance 93](#_1lu2dc9)

[Softmax 94](#_2kz067v)

[Subsampling 94](#_104agfo)

[synthetic feature 94](#_1z989ba)

[Target 94](#_4j8vrz3)

[test set 94](#_2ye626w)

[Training 95](#_1djgcep)

[training set 95](#_3xj3v2i)

[true negative (TN) 95](#_2coe5ab)

[true positive (TP) 95](#_rtofi4)

[true positive rate (TPR) 95](#_3btby5x)

[translational invariance 95](#_1qym8dq)

[unawareness (to a sensitive attribute) 96](#_4ay9r1j)

[Validation 96](#_158ubh5)

[validation set 96](#_3p8hu4y)

[Reinforcement learning 96](#_24ds4cr)

[activation function 97](#_33ipx8d)

[Agent 97](#_1io07g6)

[artificial intelligence 97](#_42nnq3z)

[Batch 97](#_2hsy0bs)

[batch normalization 97](#_wy8ajl)

[batch size 98](#_3gxvt7e)

[Bellman equation 98](#_1w363f7)

[deep model 98](#_2v83wat)

[deep neural network 99](#_1ade6im)

[Deep Q-Network (DQN) 99](#_3ud1p6f)

[DQN 99](#_29ibze8)

[dynamic model 99](#_onm9m1)

[Environment 99](#_38n9s9u)

[Episode 99](#_1nsk2hn)

[Epoch 99](#_47s7l5g)

[epsilon greedy policy 100](#_2mxhvd9)

[equality of opportunity 100](#_122s5l2)

[experience replay 101](#_3m2fo8v)

[GAN 102](#_217pygo)

[generative adversarial network (GAN) 102](#_4l7dh4h)

[generative model 102](#_30cnrca)

[greedy policy 102](#_1fhy1k3)

[Markov decision process (MDP) 103](#_2emvufp)

[Markov property 103](#_ts64ni)

[mini-batch 103](#_3drtnbb)

[mini-batch stochastic gradient descent 103](#_1sx3xj4)

[Model 103](#_4cwrg6x)

[model function 103](#_2s21qeq)

[model training 104](#_177c0mj)

[offline inference 104](#_3r6zjac)

[Policy 104](#_26c9ti5)

[Q-function 104](#_lhk3py)

[Q-learning 104](#_35h7mdr)

[random policy 105](#_1kmhwlk)

[reinforcement learning (RL) 105](#_44m5f9d)

[Return 105](#_2jrfph6)

[Reward 105](#_ywpzoz)

[State 105](#_3iwdics)

[state-action value function 105](#_1y1nskl)

[static model 106](#_4i1bb8e)

[Step 106](#_2x6llg7)

[synthetic feature 106](#_1cbvvo0)

[tabular Q-learning 106](#_3wbjebt)

[Target 106](#_2bgtojm)

Trajectory 106

[Model evaluation 107](#_3alrhf8)

[A/B testing 107](#_1pr1rn1)

[Accuracy 107](#_49qpaau)

[area under the PR curve 107](#_2ovzkin)

[area under the ROC curve 107](#_1419uqg)

[AUC (Area under the ROC Curve) 107](#_3o0xde9)

[confirmation bias 107](#_2367nm2)

[confusion matrix 108](#_ibhxtv)

[Cost 108](#_32b5gho)

[coverage bias 108](#_1hgfqph)

[cross-entropy 109](#_41g39da)

[cross-validation 109](#_2gldjl3)

[decision threshold 109](#_vqntsw)

[dropout regularization 109](#_3fqbcgp)

[early stopping 109](#_1uvlmoi)

[equality of opportunity 109](#_4ev95cb)

[equalized odds 111](#_2u0jfk4)

[experimenter's bias 112](#_195tprx)

[fairness constraint 112](#_3t5h8fq)

[fairness metric 113](#_28arinj)

[false negative (FN) 113](#_ng1svc)

[false positive (FP) 113](#_37fpbj5)

[false positive rate (FPR) 113](#_1mkzlqy)

[Generalization 113](#_46kn4er)

[generalization curve 114](#_2lpxemk)

[gradient descent 114](#_10v7oud)

[ground truth 114](#_3kuv7i6)

[group attribution bias 114](#_2005hpz)

[hinge loss 115](#_4jzt0ds)

[Holdout data 115](#_2z53all)

[implicit bias 115](#_1eadkte)

[Inference 116](#_3ya13h7)

[in-group bias 116](#_2dfbdp0)

[L2 loss 116](#_sklnwt)

[Log Loss 116](#_3ck96km)

[Loss 117](#_1rpjgsf)

[loss curve 117](#_4bp6zg8)

[loss surface 117](#_2quh9o1)

[Mean Squared Error (MSE) 118](#_15zrjvu)

[Metric 118](#_3pzf2jn)

[Metrics API (tf.metrics) 118](#_254pcrg)

[mini-batch stochastic gradient descent 118](#_k9zmz9)

[minimax loss 118](#_349n5n2)

[offline inference 118](#_1jexfuv)

[Overfitting 119](#_43ekyio)

[participation bias 119](#_2ijv8qh)

[Performance 119](#_xp5iya)

[Perplexity 119](#_3hot1m3)

[PR AUC (area under the PR curve) 119](#_1wu3btw)

[Precision 120](#_4gtquhp)

[precision-recall curve 120](#_2vz14pi)

[Recall 120](#_1b4bexb)

[reporting bias 120](#_3v3yxl4)

[Reward 120](#_2a997sx)

[ROC (receiver operating characteristic) Curve 121](#_peji0q)

[Root Mean Squared Error (RMSE) 121](#_39e70oj)

[sampling bias 121](#_1ojhawc)

[Scoring 121](#_48j4tk5)

[selection bias 121](#_2nof3ry)

[similarity measure 122](#_12tpdzr)

[size invariance 122](#_3mtcwnk)

[Sketching 122](#_21yn6vd)

[sparse feature 122](#_h3xh36)

[squared hinge loss 123](#_313kzqz)

[squared loss 123](#_1g8v9ys)

[stochastic gradient descent (SGD) 123](#_408isml)

[structural risk minimization (SRM) 123](#_2fdt2ue)

[tabular Q-learning 123](#_uj3d27)

[test set 124](#_3eiqvq0)

[true negative (TN) 124](#_1to15xt)

[true positive (TP) 124](#_4dnoolm)

[true positive rate (TPR) 124](#_2ssyytf)

[Underfitting 124](#_17y9918)

[user matrix 125](#_3rxwrp1)

[Validation 125](#_27371wu)

[validation set 125](#_m8hc4n)

[Clustering 125](#_3684usg)

[agglomerative clustering 125](#_1ldf509)

[Centroid 126](#_45d2no2)

[centroid-based clustering 126](#_2kicxvv)

[Class 126](#_znn83o)

[Clustering 126](#_3jnaqrh)

[hierarchical clustering 127](#_1ysl0za)

[k-means 127](#_4is8jn3)

[k-median 128](#_2xxituw)

[out-group homogeneity bias 129](#_1d2t42p)

[Outliers 129](#_3x2gmqi)

[selection bias 130](#_2c7qwyb)

[similarity measure 130](#_rd1764)

[Subsampling 131](#_3bcoptx)

[time series analysis 131](#_1qhz01q)

[unsupervised machine learning 131](#_4ahmipj)

[Neural Networks 131](#_2pmwsxc)

[accuracy 131](#_14s7355)

[activation function 132](#_3orulsy)

[artificial intelligence 132](#_23x4w0r)

[Backpropagation 132](#_j2f68k)

[Batch 132](#_3322owd)

[batch normalization 132](#_1i7cz46)

[batch size 133](#_4270hrz)

[Bayesian neural network 133](#_2hcarzs)

[bounding box 133](#_whl27l)

[Class 134](#_3gh8kve)

[Convergence 134](#_1vmiv37)

[convex function 134](#_4fm6dr0)

[convex optimization 135](#_2urgnyt)

[convex set 136](#_19wqy6m)

[Convolution 136](#_3tweguf)

[convolutional filter 136](#_291or28)

[convolutional layer 137](#_o6z1a1)

[convolutional neural network 137](#_386mjxu)

[convolutional operation 138](#_1nbwu5n)

[Cost 138](#_47bkctg)

[coverage bias 139](#_2mgun19)

[cross-entropy 139](#_11m4x92)

[deep model 139](#_3llsfwv)

[deep neural network 139](#_20r2q4o)

[Deep Q-Network (DQN) 139](#_4kqq8sh)

[dense feature 139](#_2zw0j0a)

[dense layer 139](#_1f1at83)

[Depth 139](#_3z0ybvw)

[DQN 140](#_2e68m3p)

[dynamic model 140](#_tbiwbi)

[Epoch 140](#_3db6ezb)

[exploding gradient problem 140](#_1sggp74)

[feedback loop 141](#_4cg47ux)

[feedforward neural network (FFN) 141](#_2rlei2q)

[fine tuning 141](#_16qosaj)

[forget gate 141](#_3qqcayc)

[full softmax 141](#_25vml65)

[fully connected layer 141](#_l0wvdy)

[GAN 141](#_350ke1r)

[Generalization 142](#_1k5uo9k)

[generalization curve 142](#_445i6xd)

[generative adversarial network (GAN) 142](#_2jash56)

[generative model 142](#_yg2rcz)

[Gradient 143](#_3ifqa0s)

[gradient clipping 143](#_1xl0k8l)

[gradient descent 143](#_4hko2we)

[ground truth 143](#_2wpyd47)

[group attribution bias 143](#_1bv8nc0)

[hidden layer 144](#_3vuw5zt)

[holdout data 144](#_2b06g7m)

[Hyperparameter 144](#_q5gqff)

[Hyperplane 144](#_3a54938)

[in-group bias 145](#_49a21yu)

[input function 145](#_2ofcc6n)

[input layer 145](#_13kmmeg)

[Iteration 145](#_3nka529)

[Keras 145](#_22pkfa2)

[Keypoints 145](#_huuphv)

[Layer 146](#_31ui85o)

[learning rate 146](#_1gzsidh)

[linear model 146](#_40zg11a)

[Log Loss 147](#_2g4qb93)

[Long Short-Term Memory (LSTM) 147](#_va0lgw)

[Loss 147](#_3f9o44p)

[loss curve 148](#_1ueyeci)

[loss surface 148](#_4eelx0b)

[LSTM 148](#_2tjw784)

[mini-batch 148](#_18p6hfx)

[mini-batch stochastic gradient descent 148](#_3sou03q)

[minimax loss 149](#_27u4abj)

[Momentum 149](#_mzekjc)

[neural network 149](#_36z2375)

[Neuron 149](#_1m4cdey)

[node (neural network) 149](#_463zw2r)

[Noise 149](#_2l9a6ak)

[Operation (op) 150](#_3ke7z66)

[Optimizer 150](#_1zji9dz)

[output layer 150](#_4jj5s1s)

[Perceptron 150](#_2yog29l)

[pooling 151](#_1dtqche)

[post-processing 152](#_3xtdv57)

[Regularization 152](#_2cyo5d0)

[regularization rate 153](#_s3yfkt)

[RNN 153](#_3c3ly8m)

[sequence model 153](#_1r8w8gf)

[Serving 153](#_4b8jr48)

[sigmoid function 153](#_2qdu1c1)

[similarity measure 154](#_15j4bju)

[Softmax 154](#_3piru7n)

[sparse feature 154](#_24o24fg)

[sparse representation 154](#_jtcen9)

[Sparsity 155](#_33szxb2)

[spatial pooling 155](#_1iya7iv)

[Step 155](#_42xxq6o)

[step size 156](#_2i380eh)

[Stride 156](#_x8iama)

[Subsampling 156](#_3h85ta3)

[target network 156](#_1wdg3hw)

[Tensor 156](#_4gd3m5p)

[TensorBoard 157](#_2vidwdi)

[TensorFlow 157](#_1ano6lb)

[TensorFlow Playground 157](#_3unbp94)

[TensorFlow Serving 157](#_29slzgx)

[Tensor Processing Unit (TPU) 157](#_oxw9oq)

[Tensor rank 157](#_38xjscj)

[Tensor shape 157](#_1o2u2kc)

[Tensor size 158](#_482hl85)

[time series analysis 158](#_2n7rvfy)

[Training 158](#_12d25nr)

[Trajectory 158](#_3mcpobk)

[transfer learning 158](#_21hzyjd)

[translational invariance 158](#_gna8r6)

[Upweighting 159](#_30mxrez)

[user matrix 159](#_1fs81ms)

[Validation 159](#_3zrvkal)

[validation set 159](#_2ex5uie)

[vanishing gradient problem 159](#_u2g4q7)

[Weight 160](#_3e23ne0)

[Weighted Alternating Least Squares (WALS) 160](#_1t7dxlt)

[Width 160](#_4d71g9m)

[Natural Language Processing 160](#_2scbqhf)

[Bag of words 160](#_17hm0p8)

[Embeddings 161](#_3rh9jd1)

[Embedding space 161](#_26mjtku)

[Ground truth 161](#_lru3sn)

[Natural language understanding 162](#_35rhmgg)

[Neuron 162](#_1kwrwo9)

[N-gram 162](#_44wffc2)

[NLU 162](#_2k1ppjv)

[recommendation system 162](#_z6zzro)

[sentiment analysis 163](#_1ybxsna)

[transfer learning 163](#_4iblbb3)

[Trigram 163](#_2xgvliw)

# Background and References

This glossary is created as a mechanism to teach AI (machine learning and deep learning) because I could not find a concise machine learning glossary for getting started in AI.

It is mainly based on the excellent [Google Machine Learning Glossary](https://developers.google.com/machine-learning/glossary/) (this document is a subset of the Google glossary based on the following categories).

The categories are

* Foundations of Machine Learning
* Regression
* Classification
* Reinforcement learning
* Model evaluation
* Clustering
* Neural Network
* Natural Language Processing

Maintained by Ajit Jaokar. If you want to get updates to this Glossary and other content about our teaching, please [sign up for my newsletter on LinkedIn](https://www.linkedin.com/newsletters/artificial-intelligence-6793973274368856064/)

Over time, we plan to enhance it with other references such as

<https://github.com/afshinea/stanford-cs-229-machine-learning/blob/master/en/super-cheatsheet-machine-learning.pdf>

<https://docs.microsoft.com/en-us/azure/machine-learning/algorithm-cheat-sheet>

Any suggestions, enhancements etc please contact ajit.jaokar at feynlabs.ai

# Foundations of Machine Learning

## Accuracy

* + The fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:



In binary classification, accuracy has the following definition:



## artificial general intelligence

* + A non-human mechanism that demonstrates a *broad range* of problem solving, creativity, and adaptability. For example, a program demonstrating artificial general intelligence could translate text, compose symphonies, *and* excel at games that have not yet been invented.

## artificial intelligence

* + A non-human program or model that can solve sophisticated tasks. For example, a program or model that translates text or a program or model that identifies diseases from radiologic images both exhibit artificial intelligence.

Formally, [machine learning](https://developers.google.com/machine-learning/glossary/#machine_learning) is a sub-field of artificial intelligence. However, in recent years, some organizations have begun using the terms *artificial intelligence* and *machine learning* interchangeably.

## Attribute

* + Synonym for [feature](https://developers.google.com/machine-learning/glossary/#feature). In fairness, attributes often refer to characteristics pertaining to individuals.

## Backpropagation

* + The primary algorithm for performing gradient descent on neural networks. First, the output values of each node are calculated (and cached) in a forward pass. Then, the partial derivative of the error with respect to each parameter is calculated in a backward pass through the graph.

## Baseline

* + A model used as a reference point for comparing how well another model (typically, a more complex one) is performing. For example, a logistic regression model might serve as a good baseline for a deep model.

For a particular problem, the baseline helps model developers quantify the minimal expected performance that a new model must achieve for the new model to be useful.

## Batch

* + The set of examples used in one iteration (that is, one gradient update) of model training.

See also batch size.

## batch size

* + The number of examples in a batch. For example, the batch size of SGD is 1, while the batch size of a mini-batch is usually between 10 and 1000. Batch size is usually fixed during training and inference; however, TensorFlow does permit dynamic batch sizes.

## bias (math)

* + An intercept or offset from an origin. Bias (also known as the bias term) is referred to as b or w0 in machine learning models. For example, bias is the b in the following formula:



Not to be confused with bias in ethics and fairness or prediction bias.

## Binning

* + See bucketing.

## Boosting

* + A machine learning technique that iteratively combines a set of simple and not very accurate classifiers (referred to as "weak" classifiers) into a classifier with high accuracy (a "strong" classifier) by upweighting the examples that the model is currently misclassifying.

## Broadcasting

* + Expanding the shape of an operand in a matrix math operation to dimensions compatible for that operation. For instance, linear algebra requires that the two operands in a matrix addition operation must have the same dimensions. Consequently, you can't add a matrix of shape (m, n) to a vector of length n. Broadcasting enables this operation by virtually expanding the vector of length n to a matrix of shape (m,n) by replicating the same values down each column.

For example, given the following definitions, linear algebra prohibits A+B because A and B have different dimensions:

A = [[7, 10, 4],

[13, 5, 9]]

B = [2]

However, broadcasting enables the operation A+B by virtually expanding B to:

[[2, 2, 2],

[2, 2, 2]]

Thus, A+B is now a valid operation:

[[7, 10, 4], + [[2, 2, 2], = [[ 9, 12, 6],

[13, 5, 9]] [2, 2, 2]] [15, 7, 11]]

See the following description of broadcasting in NumPy for more details.

## Bucketing

* + Converting a (usually continuous) feature into multiple binary features called buckets or bins, typically based on value range. For example, instead of representing temperature as a single continuous floating-point feature, you could chop ranges of temperatures into discrete bins. Given temperature data sensitive to a tenth of a degree, all temperatures between 0.0 and 15.0 degrees could be put into one bin, 15.1 to 30.0 degrees could be a second bin, and 30.1 to 50.0 degrees could be a third bin.

## categorical data

* + Features having a discrete set of possible values. For example, consider a categorical feature named house style, which has a discrete set of three possible values: Tudor, ranch, colonial. By representing house style as categorical data, the model can learn the separate impacts of Tudor, ranch, and colonial on house price.

Sometimes, values in the discrete set are mutually exclusive, and only one value can be applied to a given example. For example, a car maker categorical feature would probably permit only a single value (Toyota) per example. Other times, more than one value may be applicable. A single car could be painted more than one different color, so a car color categorical feature would likely permit a single example to have multiple values (for example, red and white).

Categorical features are sometimes called discrete features.

Contrast with numerical data.

## Clipping

* + A technique for handling outliers. Specifically, reducing feature values that are greater than a set maximum value down to that maximum value. Also, increasing feature values that are less than a specific minimum value up to that minimum value.

For example, suppose that only a few feature values fall outside the range 40–60. In this case, you could do the following:

Clip all values over 60 to be exactly 60.

Clip all values under 40 to be exactly 40.

In addition to bringing input values within a designated range, clipping can also used to force gradient values within a designated range during training.

## confirmation bias

* + The tendency to search for, interpret, favor, and recall information in a way that confirms one's preexisting beliefs or hypotheses. Machine learning developers may inadvertently collect or label data in ways that influence an outcome supporting their existing beliefs. Confirmation bias is a form of implicit bias.

Experimenter's bias is a form of confirmation bias in which an experimenter continues training models until a preexisting hypothesis is confirmed.

## confusion matrix

* + An NxN table that summarizes how successful a classification model's predictions were; that is, the correlation between the label and the model's classification. One axis of a confusion matrix is the label that the model predicted, and the other axis is the actual label. N represents the number of classes. In a binary classification problem, N=2. For example, here is a sample confusion matrix for a binary classification problem:

|  |  |  |
| --- | --- | --- |
|  | Tumor (predicted) | Non-Tumor (predicted) |
| Tumor (actual) | 18 | 1 |
| Non-Tumor (actual) | 6 | 452 |

The preceding confusion matrix shows that of the 19 samples that actually had tumors, the model correctly classified 18 as having tumors (18 true positives), and incorrectly classified 1 as not having a tumor (1 false negative). Similarly, of 458 samples that actually did not have tumors, 452 were correctly classified (452 true negatives) and 6 were incorrectly classified (6 false positives).

The confusion matrix for a multi-class classification problem can help you determine mistake patterns. For example, a confusion matrix could reveal that a model trained to recognize handwritten digits tends to mistakenly predict 9 instead of 4, or 1 instead of 7.

Confusion matrices contain sufficient information to calculate a variety of performance metrics, including precision and recall.

## continuous feature

* + A floating-point feature with an infinite range of possible values. Contrast with discrete feature.

## convenience sampling

* + Using a dataset not gathered scientifically in order to run quick experiments. Later on, it's essential to switch to a scientifically gathered dataset.

## Convergence

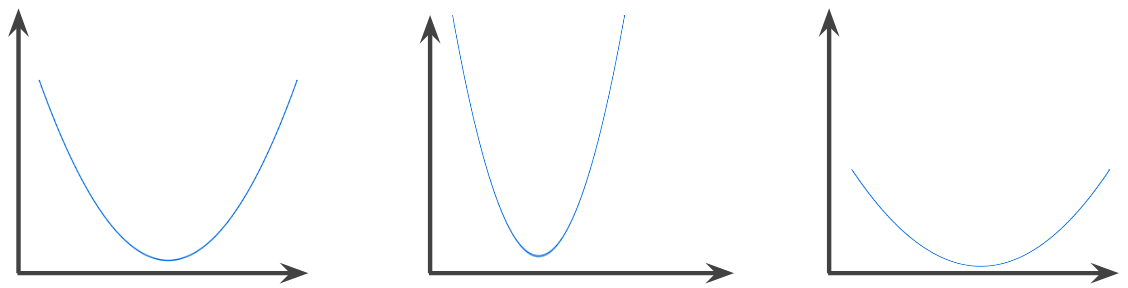
* + Informally, often refers to a state reached during training in which training loss and validation loss change very little or not at all with each iteration after a certain number of iterations. In other words, a model reaches convergence when additional training on the current data will not improve the model. In deep learning, loss values sometimes stay constant or nearly so for many iterations before finally descending, temporarily producing a false sense of convergence.

See also early stopping.

See also Boyd and Vandenberghe, Convex Optimization.

## convex function

* A function in which the region above the graph of the function is a [convex set](https://developers.google.com/machine-learning/glossary/#convex_set). The prototypical convex function is shaped something like the letter U. For example, the following are all convex functions:



By contrast, the following function is not convex. Notice how the region above the graph is not a convex set:



A strictly convex function has exactly one local minimum point, which is also the global minimum point. The classic U-shaped functions are strictly convex functions. However, some convex functions (for example, straight lines) are not U-shaped.

A lot of the common [loss functions](https://developers.google.com/machine-learning/glossary/#loss), including the following, are convex functions:

* L2 loss
* Log Loss
* L1 regularization
* L2 regularization

Many variations of gradient descent are guaranteed to find a point close to the minimum of a strictly convex function. Similarly, many variations of stochastic gradient descent have a high probability (though, not a guarantee) of finding a point close to the minimum of a strictly convex function.

The sum of two convex functions (for example, L2 loss + L1 regularization) is a convex function.

Deep models are never convex functions. Remarkably, algorithms designed for convex optimization tend to find reasonably good solutions on deep networks anyway, even though those solutions are not guaranteed to be a global minimum.

## convex optimization

* + The process of using mathematical techniques such as gradient descent to find the minimum of a convex function. A great deal of research in machine learning has focused on formulating various problems as convex optimization problems and in solving those problems more efficiently.

For complete details, see Boyd and Vandenberghe, Convex Optimization.

convex set

A subset of Euclidean space such that a line drawn between any two points in the subset remains completely within the subset. For instance, the following two shapes are convex sets:

A rectangle
and a semi-ellipse are both convex sets.

By contrast, the following two shapes are not convex sets:

A pie-chart
with a missing slice and a firework are both nonconvex sets.

## data augmentation

* + Artificially boosting the range and number of training examples by transforming existing examples to create additional examples. For example, suppose images are one of your features, but your dataset doesn't contain enough image examples for the model to learn useful associations. Ideally, you'd add enough labeled images to your dataset to enable your model to train properly. If that's not possible, data augmentation can rotate, stretch, and reflect each image to produce many variants of the original picture, possibly yielding enough labeled data to enable excellent training.

## Data set or dataset

* + A collection of examples.

## decision boundary

* The separator between classes learned by a model in a [binary class](https://developers.google.com/machine-learning/glossary/#binary_classification) or [multi-class classification problems](https://developers.google.com/machine-learning/glossary/#multi-class). For example, in the following image representing a binary classification problem, the decision boundary is the frontier between the orange class and the blue class:

A
well-defined boundary between one class and another.

## deep model

* + A type of neural network containing multiple hidden layers.

Contrast with wide model.

## deep neural network

* + Synonym for deep model.

## Deep Q-Network (DQN)

* + In Q-learning, a deep neural network that predicts Q-functions.

Critic is a synonym for Deep Q-Network.

## dense feature

* + A feature in which most values are non-zero, typically a Tensor of floating-point values. Contrast with sparse feature.

## dense layer

* + Synonym for fully connected layer.

## Depth

* + The number of layers (including any embedding layers) in a neural network that learn weights. For example, a neural network with 5 hidden layers and 1 output layer has a depth of 6.

## dimension reduction

* + Decreasing the number of dimensions used to represent a particular feature in a feature vector, typically by converting to an embedding.

## Dimensions

* Overloaded term having any of the following definitions:
* The number of levels of coordinates in a [Tensor](https://developers.google.com/machine-learning/glossary/#tensor). For example:
  + A scalar has zero dimensions; for example, ["Hello"].
  + A vector has one dimension; for example, [3, 5, 7, 11].
  + A matrix has two dimensions; for example, [[2, 4, 18], [5, 7, 14]].

You can uniquely specify a particular cell in a one-dimensional vector with one coordinate; you need two coordinates to uniquely specify a particular cell in a two-dimensional matrix.

* The number of entries in a [feature vector](https://developers.google.com/machine-learning/glossary/#feature_vector).
* The number of elements in an [embedding](https://developers.google.com/machine-learning/glossary/#embeddings) layer.

## discrete feature

* + A feature with a finite set of possible values. For example, a feature whose values may only be animal, vegetable, or mineral is a discrete (or categorical) feature. Contrast with continuous feature.

## discriminative model

* + A model that predicts labels from a set of one or more features. More formally, discriminative models define the conditional probability of an output given the features and weights; that is:
    - p(output | features, weights)

## discriminator

* + A system that determines whether examples are real or fake.

The subsystem within a generative adversarial network that determines whether the examples created by the generator are real or fake.

## Downsampling

* + Overloaded term that can mean either of the following:
    - Reducing the amount of information in a feature in order to train a model more efficiently. For example, before training an image recognition model, downsampling high-resolution images to a lower-resolution format.
    - Training on a disproportionately low percentage of over-represented class examples in order to improve model training on under-represented classes. For example, in a class-imbalanced dataset, models tend to learn a lot about the majority class and not enough about the minority class. Downsampling helps balance the amount of training on the majority and minority classes.

## dropout regularization

* + A form of regularization useful in training neural networks. Dropout regularization works by removing a random selection of a fixed number of the units in a network layer for a single gradient step. The more units dropped out, the stronger the regularization. This is analogous to training the network to emulate an exponentially large ensemble of smaller networks. For full details, see Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

## dynamic model

* + A model that is trained online in a continuously updating fashion. That is, data is continuously entering the model.

## eager execution

* + A TensorFlow programming environment in which operations run immediately. By contrast, operations called in graph execution don't run until they are explicitly evaluated. Eager execution is an imperative interface, much like the code in most programming languages. Eager execution programs are generally far easier to debug than graph execution programs.

## early stopping

* + A method for regularization that involves ending model training before training loss finishes decreasing. In early stopping, you end model training when the loss on a validation dataset starts to increase, that is, when generalization performance worsens.

## Embeddings

* + A categorical feature represented as a continuous-valued feature. Typically, an embedding is a translation of a high-dimensional vector into a low-dimensional space. For example, you can represent the words in an English sentence in either of the following two ways:
    - As a million-element (high-dimensional) sparse vector in which all elements are integers. Each cell in the vector represents a separate English word; the value in a cell represents the number of times that word appears in a sentence. Since a single English sentence is unlikely to contain more than 50 words, nearly every cell in the vector will contain a 0. The few cells that aren't 0 will contain a low integer (usually 1) representing the number of times that word appeared in the sentence.
    - As a several-hundred-element (low-dimensional) dense vector in which each element holds a floating-point value between 0 and 1. This is an embedding.

In TensorFlow, embeddings are trained by backpropagating loss just like any other parameter in a neural network.

## embedding space

* + The d-dimensional vector space that features from a higher-dimensional vector space are mapped to. Ideally, the embedding space contains a structure that yields meaningful mathematical results; for example, in an ideal embedding space, addition and subtraction of embeddings can solve word analogy tasks.

The dot product of two embeddings is a measure of their similarity.

## Ensemble

* + A merger of the predictions of multiple models. You can create an ensemble via one or more of the following:
    - different initializations
    - different hyperparameters
    - different overall structure

Deep and wide models are a kind of ensemble.

## Environment

* + In reinforcement learning, the world that contains the agent and allows the agent to observe that world's state. For example, the represented world can be a game like chess, or a physical world like a maze. When the agent applies an action to the environment, then the environment transitions between states.

## Episode

* + In reinforcement learning, each of the repeated attempts by the agent to learn an environment.

## Epoch

* + A full training pass over the entire dataset such that each example has been seen once. Thus, an epoch represents N / batch size training iterations, where N is the total number of examples.

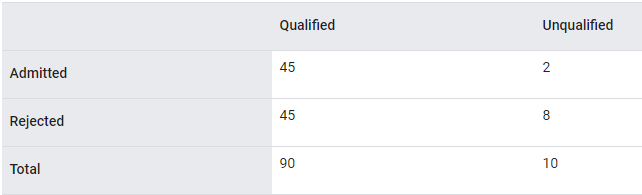
## equalized odds

* + A fairness metric that checks if, for any particular label and attribute, a classifier predicts that label equally well for all values of that attribute.

For example, suppose Glubbdubdrib University admits both Lilliputians and Brobdingnagians to a rigorous mathematics program. Lilliputians' secondary schools offer a robust curriculum of math classes, and the vast majority of students are qualified for the university program. Brobdingnagians' secondary schools don’t offer math classes at all, and as a result, far fewer of their students are qualified. Equalized odds is satisfied provided that no matter whether an applicant is a Lilliputian or a Brobdingnagian, if they are qualified, they are equally as likely to get admitted to the program, and if they are not qualified, they are equally as likely to get rejected.

Let’s say 100 Lilliputians and 100 Brobdingnagians apply to Glubbdubdrib University, and admissions decisions are made as follows:

Table 3. Lilliputian applicants (90% are qualified)



Percentage of qualified students admitted: 45/90 = 50%

Percentage of unqualified students rejected: 8/10 = 80%

Total percentage of Lilliputian students admitted: (45+2)/100 = 47%

Table 4. Brobdingnagian applicants (10% are qualified):



Percentage of qualified students admitted: 5/10 = 50%

Percentage of unqualified students rejected: 72/90 = 80%

Total percentage of Brobdingnagian students admitted: (5+18)/100 = 23%

Equalized odds is satisfied because qualified Lilliputian and Brobdingnagian students both have a 50% chance of being admitted, and unqualified Lilliputian and Brobdingnagian have an 80% chance of being rejected.

Note: While equalized odds is satisfied here, demographic parity is not satisfied. Lilliputian and Brobdingnagian students are admitted to Glubbdubdrib University at different rates; 47% of Lilliputian students are admitted, and 23% of Brobdingnagian students are admitted.

Equalized odds is formally defined in "Equality of Opportunity in Supervised Learning" as follows: "predictor Ŷ satisfies equalized odds with respect to protected attribute A and outcome Y if Ŷ and A are independent, conditional on Y."

Note: Contrast equalized odds with the more relaxed equality of opportunity metric.

## Example

* + One row of a dataset. An example contains one or more features and possibly a label. See also labeled example and unlabeled example.

## experience replay

* + In reinforcement learning, a DQN technique used to reduce temporal correlations in training data. The agent stores state transitions in a replay buffer, and then samples transitions from the replay buffer to create training data.

## experimenter's bias

* + See confirmation bias.

## fairness constraint

* + Applying a constraint to an algorithm to ensure one or more definitions of fairness are satisfied. Examples of fairness constraints include:
    - Post-processing your model's output.
    - Altering the loss function to incorporate a penalty for violating a fairness metric.
    - Directly adding a mathematical constraint to an optimization problem.

## false negative (FN)

* + An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam.

## false positive (FP)

* + An example in which the model mistakenly predicted the positive class. For example, the model inferred that a particular email message was spam (the positive class), but that email message was actually not spam.

## false positive rate (FPR)

* + The x-axis in an ROC curve. The false positive rate is defined as follows:



## Feature

* + An input variable used in making predictions.

## feature engineering

* + The process of determining which features might be useful in training a model, and then converting raw data from log files and other sources into said features. In TensorFlow, feature engineering often means converting raw log file entries to tf.Example protocol buffers. See also tf.Transform.
  + Feature engineering is sometimes called feature extraction.

## feature extraction

* + Overloaded term having either of the following definitions:
    - Retrieving intermediate feature representations calculated by an unsupervised or pretrained model (for example, hidden layer values in a neural network) for use in another model as input.
    - Synonym for feature engineering.

## feature set

* + The group of features your machine learning model trains on. For example, postal code, property size, and property condition might comprise a simple feature set for a model that predicts housing prices.

## feature spec

* + Describes the information required to extract features data from the tf.Example protocol buffer. Because the tf.Example protocol buffer is just a container for data, you must specify the following:
    - the data to extract (that is, the keys for the features)
    - the data type (for example, float or int)
    - The length (fixed or variable)

The Estimator API provides facilities for producing a feature spec from a list of FeatureColumns.

## feature vector

* + The list of feature values representing an example passed into a model.

## federated learning

* + A distributed machine learning approach that trains machine learning models using decentralized examples residing on devices such as smartphones. In federated learning, a subset of devices downloads the current model from a central coordinating server. The devices use the examples stored on the devices to make improvements to the model. The devices then upload the model improvements (but not the training examples) to the coordinating server, where they are aggregated with other updates to yield an improved global model. After the aggregation, the model updates computed by devices are no longer needed, and can be discarded.

Since the training examples are never uploaded, federated learning follows the privacy principles of focused data collection and data minimization.

## feedback loop

* + In machine learning, a situation in which a model's predictions influence the training data for the same model or another model. For example, a model that recommends movies will influence the movies that people see, which will then influence subsequent movie recommendation models.

## few-shot learning

* + A machine learning approach, often used for object classification, designed to learn effective classifiers from only a small number of training examples.

See also one-shot learning.

## fine tuning

* + Perform a secondary optimization to adjust the parameters of an already trained model to fit a new problem. Fine tuning often refers to refitting the weights of a trained unsupervised model to a supervised model.

## forget gate

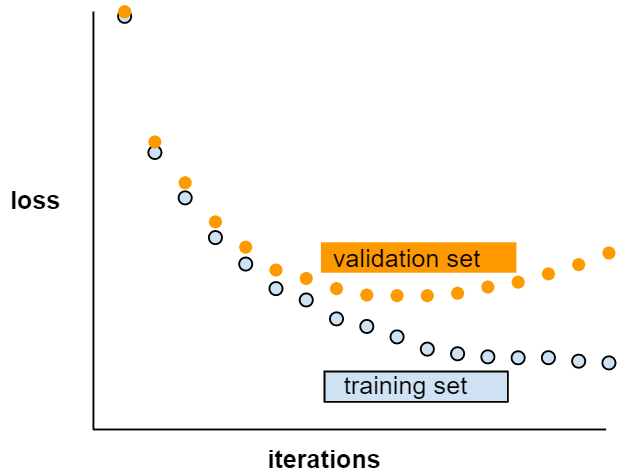
* + The portion of a Long Short-Term Memory cell that regulates the flow of information through the cell. Forget gates maintain context by deciding which information to discard from the cell state.

## Generalization

* + Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.

## generalization curve

* + A loss curve showing both the training set and the validation set. A generalization curve can help you detect possible overfitting. For example, the following generalization curve suggests overfitting because loss for the validation set ultimately becomes significantly higher than for the training set.



## generative model

* + Practically speaking, a model that does either of the following:
    - Creates (generates) new examples from the training dataset. For example, a generative model could create poetry after training on a dataset of poems. The generator part of a generative adversarial network falls into this category.
    - Determines the probability that a new example comes from the training set, or was created from the same mechanism that created the training set. For example, after training on a dataset consisting of English sentences, a generative model could determine the probability that new input is a valid English sentence.

A generative model can theoretically discern the distribution of examples or particular features in a dataset. That is:

p(examples)

Unsupervised learning models are generative.

Contrast with discriminative models.

## Gradient

* + The vector of partial derivatives with respect to all of the independent variables. In machine learning, the gradient is the vector of partial derivatives of the model function. The gradient points in the direction of steepest ascent.

## gradient clipping

* + A commonly used mechanism to mitigate the exploding gradient problem by artificially limiting (clipping) the maximum value of gradients when using gradient descent to train a model.

## gradient descent

* + A technique to minimize loss by computing the gradients of loss with respect to the model's parameters, conditioned on training data. Informally, gradient descent iteratively adjusts parameters, gradually finding the best combination of weights and bias to minimize loss.

## Graph

* + In TensorFlow, a computation specification. Nodes in the graph represent operations. Edges are directed and represent passing the result of an operation (a Tensor) as an operand to another operation. Use TensorBoard to visualize a graph.

## graph execution

* + A TensorFlow programming environment in which the program first constructs a graph and then executes all or part of that graph. Graph execution is the default execution mode in TensorFlow 1.x.
  + Contrast with eager execution.

## ground truth

* + The correct answer. Reality. Since reality is often subjective, expert raters typically are the proxy for ground truth.

## group attribution bias

* + Assuming that what is true for an individual is also true for everyone in that group. The effects of group attribution bias can be exacerbated if a convenience sampling is used for data collection. In a non-representative sample, attributions may be made that do not reflect reality.
  + See also out-group homogeneity bias and in-group bias.

## Hashing

* + In machine learning, a mechanism for bucketing categorical data, particularly when the number of categories is large, but the number of categories actually appearing in the dataset is comparatively small.

For example, Earth is home to about 60,000 tree species. You could represent each of the 60,000 tree species in 60,000 separate categorical buckets. Alternatively, if only 200 of those tree species actually appear in a dataset, you could use hashing to divide tree species into perhaps 500 buckets.

A single bucket could contain multiple tree species. For example, hashing could place baobab and red maple—two genetically dissimilar species—into the same bucket. Regardless, hashing is still a good way to map large categorical sets into the desired number of buckets. Hashing turns a categorical feature having a large number of possible values into a much smaller number of values by grouping values in a deterministic way.

## Heuristic

* + A quick solution to a problem, which may or may not be the best solution. For example, "With a heuristic, we achieved 86% accuracy. When we switched to a deep neural network, accuracy went up to 98%."

## hidden layer

* + A synthetic layer in a neural network between the input layer (that is, the features) and the output layer (the prediction). Hidden layers typically contain an activation function (such as ReLU) for training. A deep neural network contains more than one hidden layer.

## holdout data

* + Examples intentionally not used ("held out") during training. The validation dataset and test dataset are examples of holdout data. Holdout data helps evaluate your model's ability to generalize to data other than the data it was trained on. The loss on the holdout set provides a better estimate of the loss on an unseen dataset than does the loss on the training set.

## Hyperparameter

* + The "knobs" that you tweak during successive runs of training a model. For example, learning rate is a hyperparameter.

Contrast with parameter.

## Hyperplane

* + A boundary that separates a space into two subspaces. For example, a line is a hyperplane in two dimensions and a plane is a hyperplane in three dimensions. More typically in machine learning, a hyperplane is the boundary separating a high-dimensional space. Kernel Support Vector Machines use hyperplanes to separate positive classes from negative classes, often in a very high-dimensional space.

## image recognition

* + A process that classifies object(s), pattern(s), or concept(s) in an image. Image recognition is also known as image classification.

## imbalanced dataset

* + Synonym for class-imbalanced dataset.

## implicit bias

* + Automatically making an association or assumption based on one’s mental models and memories. Implicit bias can affect the following:
    - How data is collected and classified.
    - How machine learning systems are designed and developed.

For example, when building a classifier to identify wedding photos, an engineer may use the presence of a white dress in a photo as a feature. However, white dresses have been customary only during certain eras and in certain cultures

## Inference

* + In machine learning, often refers to the process of making predictions by applying the trained model to unlabeled examples. In statistics, inference refers to the process of fitting the parameters of a distribution conditioned on some observed data.

## in-group bias

* + In machine learning, often refers to the process of making predictions by applying the trained model to unlabeled examples. In statistics, inference refers to the process of fitting the parameters of a distribution conditioned on some observed data.

In-group bias is a form of group attribution bias.

## Instance

* + Synonym for example.

## Interpretability

* + The degree to which a model's predictions can be readily explained. Deep models are often non-interpretable; that is, a deep model's different layers can be hard to decipher. By contrast, linear regression models and wide models are typically far more interpretable.

## Items

* + In a recommendation system, the entities that a system recommends. For example, videos are the items that a video store recommends, while books are the items that a bookstore recommends.

## Iteration

* + A single update of a model's weights during training. An iteration consists of computing the gradients of the parameters with respect to the loss on a single batch of data.

## Keras

* + A popular Python machine learning API. Keras runs on several deep learning frameworks, including TensorFlow, where it is made available as tf.keras.

## Keypoints

* + The coordinates of particular features in an image. For example, for an image recognition model that distinguishes flower species, keypoints might be the center of each petal, the stem, the stamen, and so on.

## L1 regularization

* + A type of regularization that penalizes weights in proportion to the sum of the absolute values of the weights. In models relying on sparse features,  L1 regularization helps drive the weights of irrelevant or barely relevant features to exactly 0, which removes those features from the model. Contrast with[L2](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization)regularization.

## L2 loss

* + See squared loss.

## L2 regularization

* + A type of regularization that penalizes weights in proportion to the sum of the squares of the weights. [L2](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization)regularization helps drive outlier weights (those with high positive or low negative values) closer to 0 but not quite to 0. (Contrast with  L1 regularization.) [L2](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization)regularization always improves generalization in linear models.

## Lambda

* + Synonym for regularization rate.

## Landmarks

* + Synonym for keypoints.

## Layer

* + A set of neurons in a neural network that process a set of input features, or the output of those neurons.
  + Also, an abstraction in TensorFlow. Layers are Python functions that take Tensors and configuration options as input and produce other tensors as output. Once the necessary Tensors have been composed, the user can convert the result into an Estimator via a model function.

## learning rate

* + A scalar used to train a model via gradient descent. During each iteration, the gradient descent algorithm multiplies the learning rate by the gradient. The resulting product is called the gradient step.

Learning rate is a key hyperparameter.

## Logits

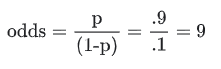
* + The vector of raw (non-normalized) predictions that a classification model generates, which is ordinarily then passed to a normalization function. If the model is solving a multi-class classification problem, logits typically become an input to the softmax function. The softmax function then generates a vector of (normalized) probabilities with one value for each possible class.

In addition, logits sometimes refer to the element-wise inverse of the sigmoid function.

## log-odds

* + The logarithm of the odds of some event.

If the event refers to a binary probability, then odds refers to the ratio of the probability of success (p) to the probability of failure (1-p). For example, suppose that a given event has a 90% probability of success and a 10% probability of failure. In this case, odds is calculated as follows:



The log-odds is simply the logarithm of the odds. By convention, "logarithm" refers to natural logarithm, but logarithm could actually be any base greater than 1. Sticking to convention, the log-odds of our example is therefore:



The log-odds are the inverse of the sigmoid function.

## machine learning

* + A program or system that builds (trains) a predictive model from input data. The system uses the learned model to make useful predictions from new (never-before-seen) data drawn from the same distribution as the one used to train the model. Machine learning also refers to the field of study concerned with these programs or systems.

## Mean Squared Error (MSE)

* + The average squared loss per example. MSE is calculated by dividing the squared loss by the number of examples. The values that TensorFlow Playground displays for "Training loss" and "Test loss" are MSE.

## Metric

* + A number that you care about. May or may not be directly optimized in a machine-learning system. A metric that your system tries to optimize is called an objective.

## ML

* + Abbreviation for machine learning.

## MNIST

* + A public-domain dataset compiled by LeCun, Cortes, and Burges containing 60,000 images, each image showing how a human manually wrote a particular digit from 0–9. Each image is stored as a 28x28 array of integers, where each integer is a grayscale value between 0 and 255, inclusive.

MNIST is a canonical dataset for machine learning, often used to test new machine learning approaches.

## Model

* + The representation of what a machine learning system has learned from the training data. Within TensorFlow, model is an overloaded term, which can have either of the following two related meanings:
    - The TensorFlow graph that expresses the structure of how a prediction will be computed.
    - The particular weights and biases of that TensorFlow graph, which are determined by training.

## model function

* + The function within an Estimator that implements machine learning training, evaluation, and inference. For example, the training portion of a model function might handle tasks such as defining the topology of a deep neural network and identifying its optimizer function. When using premade Estimators, someone has already written the model function for you. When using custom Estimators, you must write the model function yourself.

## model training

* + The process of determining the best model.

## Momentum

* + A sophisticated gradient descent algorithm in which a learning step depends not only on the derivative in the current step, but also on the derivatives of the step(s) that immediately preceded it. Momentum involves computing an exponentially weighted moving average of the gradients over time, analogous to momentum in physics. Momentum sometimes prevents learning from getting stuck in local minima.

## NaN trap

* + When one number in your model becomes a NaN during training, which causes many or all other numbers in your model to eventually become a NaN.

NaN is an abbreviation for "Not a Number."

## natural language understanding

* + Determining a user's intentions based on what the user typed or said. For example, a search engine uses natural language understanding to determine what the user is searching for based on what the user typed or said.

## neural network

* + A model that, taking inspiration from the brain, is composed of layers (at least one of which is hidden) consisting of simple connected units or neurons followed by nonlinearities.

## Neuron

* + A node in a neural network, typically taking in multiple input values and generating one output value. The neuron calculates the output value by applying an activation function (nonlinear transformation) to a weighted sum of input values.

## Noise

* + Broadly speaking, anything that obscures the signal in a dataset. Noise can be introduced into data in a variety of ways. For example:
    - Human raters make mistakes in labeling.
    - Humans and instruments mis-record or omit feature values.

## Normalization

* + The process of converting an actual range of values into a standard range of values, typically -1 to +1 or 0 to 1. For example, suppose the natural range of a certain feature is 800 to 6,000. Through subtraction and division, you can normalize those values into the range -1 to +1.

See also scaling.

## numerical data

* + Features represented as integers or real-valued numbers. For example, in a real estate model, you would probably represent the size of a house (in square feet or square meters) as numerical data. Representing a feature as numerical data indicates that the feature's values have a mathematical relationship to each other and possibly to the label. For example, representing the size of a house as numerical data indicates that a 200 square-meter house is twice as large as a 100 square-meter house. Furthermore, the number of square meters in a house probably has some mathematical relationship to the price of the house.

Not all integer data should be represented as numerical data. For example, postal codes in some parts of the world are integers; however, integer postal codes should not be represented as numerical data in models. That's because a postal code of 20000 is not twice (or half) as potent as a postal code of 10000. Furthermore, although different postal codes do correlate to different real estate values, we can't assume that real estate values at postal code 20000 are twice as valuable as real estate values at postal code 10000. Postal codes should be represented as categorical data instead.

Numerical features are sometimes called continuous features.

## Objective

* + A metric that your algorithm is trying to optimize.

## Objective function

* + The mathematical formula or metric that a model aims to optimize. For example, the objective function for linear regression is usually squared loss. Therefore, when training a linear regression model, the goal is to minimize squared loss.

In some cases, the goal is to maximize the objective function. For example, if the objective function is accuracy, the goal is to maximize accuracy.

See also loss.

## one-shot learning

* + A machine learning approach, often used for object classification, designed to learn effective classifiers from a single training example.

See also few-shot learning.

## Operation (op)

* + A node in the TensorFlow graph. In TensorFlow, any procedure that creates, manipulates, or destroys a Tensor is an operation. For example, a matrix multiply is an operation that takes two Tensors as input and generates one Tensor as output.

## Outliers

* + Values distant from most other values. In machine learning, any of the following are outliers:
    - Weights with high absolute values.
    - Predicted values relatively far away from the actual values.
    - Input data whose values are more than roughly 3 standard deviations from the mean.

Outliers often cause problems in model training. Clipping is one way of managing outliers.

## Overfitting

* + Creating a model that matches the training data so closely that the model fails to make correct predictions on new data.

## pandas

* + A column-oriented data analysis API. Many machine learning frameworks, including TensorFlow, support pandas data structures as input. See the pandas documentation for details.

## Parameter

* + A variable of a model that the machine learning system trains on its own. For example, weights are parameters whose values the machine learning system gradually learns through successive training iterations. Contrast with hyperparameter.

## parameter update

* + The operation of adjusting a model's parameters during training, typically within a single iteration of gradient descent.

## partial derivative

* + A derivative in which all but one of the variables is considered a constant. For example, the partial derivative of f(x, y) with respect to x is the derivative of f considered as a function of x alone (that is, keeping y constant). The partial derivative of f with respect to x focuses only on how x is changing and ignores all other variables in the equation.

## participation bias

* + Synonym for non-response bias. See selection bias.

## partitioning strategy

* + The algorithm by which variables are divided across parameter servers.

## Performance

* + Overloaded term with the following meanings:
    - The traditional meaning within software engineering. Namely: How fast (or efficiently) does this piece of software run?
    - The meaning within machine learning. Here, performance answers the following question: How correct is this model? That is, how good are the model's predictions?

## Perplexity

* + One measure of how well a model is accomplishing its task. For example, suppose your task is to read the first few letters of a word a user is typing on a smartphone keyboard, and to offer a list of possible completion words. Perplexity, P, for this task is approximately the number of guesses you need to offer in order for your list to contain the actual word the user is trying to type.
  + Perplexity is related to cross-entropy as follows:



## Pipeline

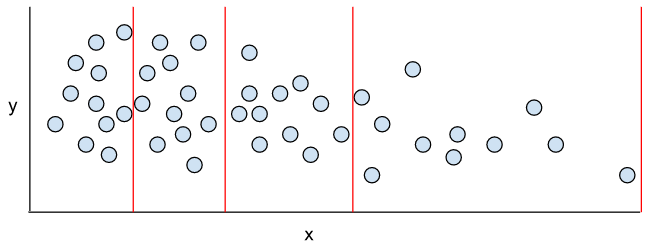
* + The infrastructure surrounding a machine learning algorithm. A pipeline includes gathering the data, putting the data into training data files, training one or more models, and exporting the models to production.

## Quantile

* + Each bucket in quantile bucketing.

## quantile bucketing

* + Distributing a feature's values into buckets so that each bucket contains the same (or almost the same) number of examples. For example, the following figure divides 44 points into 4 buckets, each of which contains 11 points. In order for each bucket in the figure to contain the same number of points, some buckets span a different width of x-values.



## Quantization

* + An algorithm that implements quantile bucketing on a particular feature in a dataset.

## Queue

* + A TensorFlow Operation that implements a queue data structure. Typically used in I/O.

## rank (ordinality)

* + The ordinal position of a class in a machine learning problem that categorizes classes from highest to lowest. For example, a behavior ranking system could rank a dog's rewards from highest (a steak) to lowest (wilted kale).

## Recall

* + A metric for classification models that answers the following question: Out of all the possible positive labels, how many did the model correctly identify? That is:



## reporting bias

* + The fact that the frequency with which people write about actions, outcomes, or properties is not a reflection of their real-world frequencies or the degree to which a property is characteristic of a class of individuals. Reporting bias can influence the composition of data that machine learning systems learn from.

For example, in books, the word laughed is more prevalent than breathed. A machine learning model that estimates the relative frequency of laughing and breathing from a book corpus would probably determine that laughing is more common than breathing.

## Representation

* + The process of mapping data to useful features.

## ridge regularization

* + Synonym for [L2](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization)regularization. The term ridge regularization is more frequently used in pure statistics contexts, whereas [L2](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization)regularization is used more often in machine learning.

## sampling bias

* + See selection bias.

## Scalar

* + A single number or a single string that can be represented as a tensor of rank 0. For example, the following lines of code each create one scalar in TensorFlow:

breed = tf.Variable("poodle", tf.string)  
temperature = tf.Variable(27, tf.int16)  
precision = tf.Variable(0.982375101275, tf.float64)

## Scaling

* + A commonly used practice in feature engineering to tame a feature's range of values to match the range of other features in the dataset. For example, suppose that you want all floating-point features in the dataset to have a range of 0 to 1. Given a particular feature's range of 0 to 500, you could scale that feature by dividing each value by 500.

See also normalization.

## Scoring

* + The part of a recommendation system that provides a value or ranking for each item produced by the candidate generation phase.

## semi-supervised learning

* + Training a model on data where some of the training examples have labels but others don’t. One technique for semi-supervised learning is to infer labels for the unlabeled examples, and then to train on the inferred labels to create a new model. Semi-supervised learning can be useful if labels are expensive to obtain but unlabeled examples are plentiful.

## sensitive attribute

* + A human attribute that may be given special consideration for legal, ethical, social, or personal reasons.

## Serving

* + A synonym for inferring.

## sigmoid function

* + A function that maps logistic or multinomial regression output (log odds) to probabilities, returning a value between 0 and 1. The sigmoid function has the following formula:



where *σ* in logistic regression problems is simply:



In other words, the sigmoid function converts *σ* into a probability between 0 and 1.

In some neural networks, the sigmoid function acts as the activation function.

## similarity measure

* + In clustering algorithms, the metric used to determine how alike (how similar) any two examples are.

## size invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the size of the image changes. For example, the algorithm can still identify a cat whether it consumes 2M pixels or 200K pixels. Note that even the best image classification algorithms still have practical limits on size invariance. For example, an algorithm (or human) is unlikely to correctly classify a cat image consuming only 20 pixels.

See also translational invariance and rotational invariance.

## Sketching

* + In unsupervised machine learning, a category of algorithms that perform a preliminary similarity analysis on examples. Sketching algorithms use a locality-sensitive hash function to identify points that are likely to be similar, and then group them into buckets.

Sketching decreases the computation required for similarity calculations on large datasets. Instead of calculating similarity for every single pair of examples in the dataset, we calculate similarity only for each pair of points within each bucket.

## Softmax

* + A function that provides probabilities for each possible class in a multi-class classification model. The probabilities add up to exactly 1.0. For example, softmax might determine that the probability of a particular image being a dog at 0.9, a cat at 0.08, and a horse at 0.02. (Also called full softmax.)

Contrast with candidate sampling.

## sparse feature

* + Feature vector whose values are predominately zero or empty. For example, a vector containing a single 1 value and a million 0 values is sparse. As another example, words in a search query could also be a sparse feature—there are many possible words in a given language, but only a few of them occur in a given query.

Contrast with dense feature.

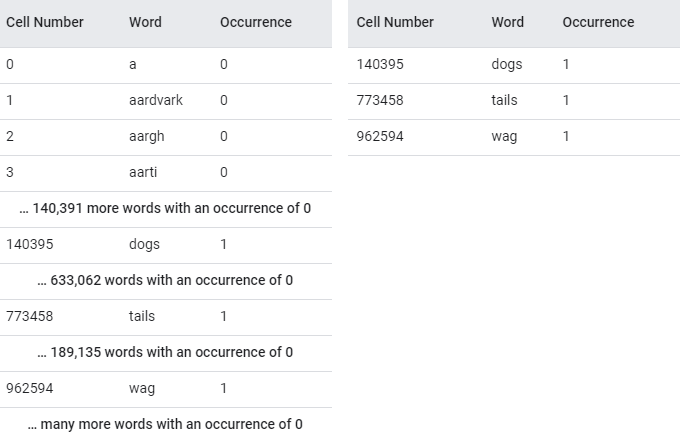
## sparse representation

* + A representation of a tensor that only stores nonzero elements.

For example, the English language consists of about a million words. Consider two ways to represent a count of the words used in one English sentence:

* + - * A dense representation of this sentence must set an integer for all one million cells, placing a 0 in most of them, and a low integer into a few of them.
      * A sparse representation of this sentence stores only those cells symbolizing a word actually in the sentence. So, if the sentence contained only 20 unique words, then the sparse representation for the sentence would store an integer in only 20 cells.

For example, consider two ways to represent the sentence, "Dogs wag tails." As the following tables show, the dense representation consumes about a million cells; the sparse representation consumes only 3 cells:



## Stationarity

* + A property of data in a dataset, in which the data distribution stays constant across one or more dimensions. Most commonly, that dimension is time, meaning that data exhibiting stationarity doesn't change over time. For example, data that exhibits stationarity doesn't change from September to December.

## Step

* + A forward and backward evaluation of one batch.

## step size

* + Synonym for learning rate.

## stochastic gradient descent (SGD)

* + A gradient descent algorithm in which the batch size is one. In other words, SGD relies on a single example chosen uniformly at random from a dataset to calculate an estimate of the gradient at each step.

## structural risk minimization (SRM)

* + An algorithm that balances two goals:
    - The desire to build the most predictive model (for example, lowest loss).
    - The desire to keep the model as simple as possible (for example, strong regularization).

For example, a function that minimizes loss+regularization on the training set is a structural risk minimization algorithm.

For more information, see http://www.svms.org/srm/.

Contrast with empirical risk minimization.

## Subsampling

* + See pooling.

## Summary

* + In TensorFlow, a value or set of values calculated at a particular step, usually used for tracking model metrics during training.

## supervised machine learning

* + Training a model from input data and its corresponding labels. Supervised machine learning is analogous to a student learning a subject by studying a set of questions and their corresponding answers. After mastering the mapping between questions and answers, the student can then provide answers to new (never-before-seen) questions on the same topic. Compare with unsupervised machine learning.

## Target

* + Synonym for label.

## termination condition

* + In reinforcement learning, the conditions that determine when an episode ends, such as when the agent reaches a certain state or exceeds a threshold number of state transitions. For example, in tic-tac-toe (also known as noughts and crosses), an episode terminates either when a player marks three consecutive spaces or when all spaces are marked.

## test set

* + The subset of the dataset that you use to test your model after the model has gone through initial vetting by the validation set.

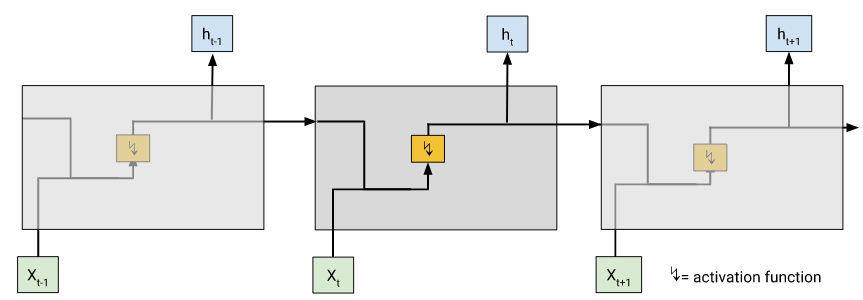
Contrast with training set and validation set.

## time series analysis

* + A subfield of machine learning and statistics that analyzes temporal data. Many types of machine learning problems require time series analysis, including classification, clustering, forecasting, and anomaly detection. For example, you could use time series analysis to forecast the future sales of winter coats by month based on historical sales data.

## Timestep

* + One "unrolled" cell within a recurrent neural network. For example, the following figure shows three timesteps (labeled with the subscripts t-1, t, and t+1):



## Training

* + The process of determining the ideal parameters comprising a model.

## training set

* + The subset of the dataset used to train a model.

Contrast with validation set and test set.

## Trajectory

* + In reinforcement learning, a sequence of tuples that represent a sequence of state transitions of the agent, where each tuple corresponds to the state, action, reward, and next state for a given state transition.

## transfer learning

* + Transferring information from one machine learning task to another. For example, in multi-task learning, a single model solves multiple tasks, such as a deep model that has different output nodes for different tasks. Transfer learning might involve transferring knowledge from the solution of a simpler task to a more complex one, or involve transferring knowledge from a task where there is more data to one where there is less data.
  + Most machine learning systems solve a single task. Transfer learning is ababy step towards artificial intelligence in which a single program can solve multiple tasks.

## translational invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the position of objects within the image changes. For example, the algorithm can still identify a dog, whether it is in the center of the frame or at the left end of the frame.

See also size invariance and rotational invariance.

## Trigram

* + An N-gram in which N=3.

## true negative (TN)

* + An example in which the model correctly predicted the negative class. For example, the model inferred that a particular email message was not spam, and that email message really was not spam.

## true positive (TP)

* + An example in which the model correctly predicted the positive class. For example, the model inferred that a particular email message was spam, and that email message really was spam.

## unawareness (to a sensitive attribute)

* + A situation in which sensitive attributes are present, but not included in the training data. Because sensitive attributes are often correlated with other attributes of one’s data, a model trained with unawareness about a sensitive attribute could still have disparate impact with respect to that attribute, or violate other fairness constraints.

## Underfitting

* + Producing a model with poor predictive ability because the model hasn't captured the complexity of the training data. Many problems can cause underfitting, including:
    - Training on the wrong set of features.
    - Training for too few epochs or at too low a learning rate.
    - Training with too high a regularization rate.
    - Providing too few hidden layers in a deep neural network.

## unlabeled example

* + An example that contains features but no label. Unlabeled examples are the input to inference. In semi-supervised and unsupervised learning, unlabeled examples are used during training.

## unsupervised machine learning

* + Training a model to find patterns in a dataset, typically an unlabeled dataset.

The most common use of unsupervised machine learning is to cluster data into groups of similar examples. For example, an unsupervised machine learning algorithm can cluster songs together based on various properties of the music. The resulting clusters can become an input to other machine learning algorithms (for example, to a music recommendation service). Clustering can be helpful in domains where true labels are hard to obtain. For example, in domains such as anti-abuse and fraud, clusters can help humans better understand the data.

Another example of unsupervised machine learning is principal component analysis (PCA). For example, applying PCA on a dataset containing the contents of millions of shopping carts might reveal that shopping carts containing lemons frequently also contain antacids.

Compare with supervised machine learning.

## Upweighting

* + Applying a weight to the downsampled class equal to the factor by which you downsampled.

## user matrix

* + In recommendation systems, an embedding generated by matrix factorization that holds latent signals about user preferences. Each row of the user matrix holds information about the relative strength of various latent signals for a single user. For example, consider a movie recommendation system. In this system, the latent signals in the user matrix might represent each user's interest in particular genres, or might be harder-to-interpret signals that involve complex interactions across multiple factors.

The user matrix has a column for each latent feature and a row for each user. That is, the user matrix has the same number of rows as the target matrix that is being factorized. For example, given a movie recommendation system for 1,000,000 users, the user matrix will have 1,000,000 rows.

## Validation

* + A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model’s performance generalizes beyond the training set.

Contrast with test set.

## validation set

* + A subset of the dataset—disjoint from the training set—used in validation.

Contrast with training set and test set.

## vanishing gradient problem

* + The tendency for the gradients of early hidden layers of some deep neural networks to become surprisingly flat (low). Increasingly lower gradients result in increasingly smaller changes to the weights on nodes in a deep neural network, leading to little or no learning. Models suffering from the vanishing gradient problem become difficult or impossible to train. Long Short-Term Memory cells address this issue.

Compare to exploding gradient problem.

## Weight

* + A coefficient for a feature in a linear model, or an edge in a deep network. The goal of training a linear model is to determine the ideal weight for each feature. If a weight is 0, then its corresponding feature does not contribute to the model.

## Weighted Alternating Least Squares (WALS)

* + Width An algorithm for minimizing the objective function during matrix factorization in recommendation systems, which allows a downweighting of the missing examples. WALS minimizes the weighted squared error between the original matrix and the reconstruction by alternating between fixing the row factorization and column factorization. Each of these optimizations can be solved by least squares convex optimization. For details, see the Recommendation Systems course

# Regression

## attribute

* + Synonym for feature. In fairness, attributes often refer to characteristics pertaining to individuals.

## Binning

* + See bucketing.

## Bucketing

* + Converting a (usually continuous) feature into multiple binary features called buckets or bins, typically based on value range. For example, instead of representing temperature as a single continuous floating-point feature, you could chop ranges of temperatures into discrete bins. Given temperature data sensitive to a tenth of a degree, all temperatures between 0.0 and 15.0 degrees could be put into one bin, 15.1 to 30.0 degrees could be a second bin, and 30.1 to 50.0 degrees could be a third bin.

## continuous feature

* + A floating-point feature with an infinite range of possible values. Contrast with discrete feature.

## Data set or dataset

* + A collection of examples.

## Dimensions

* Overloaded term having any of the following definitions:
* The number of levels of coordinates in a [Tensor](https://developers.google.com/machine-learning/glossary/#tensor). For example:
  + A scalar has zero dimensions; for example, ["Hello"].
  + A vector has one dimension; for example, [3, 5, 7, 11].
  + A matrix has two dimensions; for example, [[2, 4, 18], [5, 7, 14]].

You can uniquely specify a particular cell in a one-dimensional vector with one coordinate; you need two coordinates to uniquely specify a particular cell in a two-dimensional matrix.

* The number of entries in a [feature vector](https://developers.google.com/machine-learning/glossary/#feature_vector).
* The number of elements in an [embedding](https://developers.google.com/machine-learning/glossary/#embeddings) layer.

## Ensemble

* + A merger of the predictions of multiple models. You can create an ensemble via one or more of the following:
    - different initializations
    - different hyperparameters
    - different overall structure

Deep and wide models are a kind of ensemble.

## feature

* + An input variable used in making predictions.

## feature engineering

* + The process of determining which features might be useful in training a model, and then converting raw data from log files and other sources into said features. In TensorFlow, feature engineering often means converting raw log file entries to tf.Example protocol buffers. See also tf.Transform.

Feature engineering is sometimes called feature extraction.

## feature extraction

* + Overloaded term having either of the following definitions:
    - Retrieving intermediate feature representations calculated by an unsupervised or pretrained model (for example, hidden layer values in a neural network) for use in another model as input.
    - Synonym for feature engineering.

## feature set

* + The group of features your machine learning model trains on. For example, postal code, property size, and property condition might comprise a simple feature set for a model that predicts housing prices.

## feature spec

* + Describes the information required to extract features data from the tf.Example protocol buffer. Because the tf.Example protocol buffer is just a container for data, you must specify the following:
    - the data to extract (that is, the keys for the features)
    - the data type (for example, float or int)
    - The length (fixed or variable)

The Estimator API provides facilities for producing a feature spec from a list of FeatureColumns.

## feature vector

* + The list of feature values representing an example passed into a model.

## Generalization

* + Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.

## generalized linear model

* + A generalization of least squares regression models, which are based on Gaussian noise, to other types of models based on other types of noise, such as Poisson noise or categorical noise. Examples of generalized linear models include:
    - logistic regression
    - multi-class regression
    - least squares regression

The parameters of a generalized linear model can be found through convex optimization.

Generalized linear models exhibit the following properties:

* The average prediction of the optimal least squares regression model is equal to the average label on the training data.
* The average probability predicted by the optimal logistic regression model is equal to the average label on the training data.

The power of a generalized linear model is limited by its features. Unlike a deep model, a generalized linear model cannot "learn new features."

## Hyperparameter

* + The "knobs" that you tweak during successive runs of training a model. For example, learning rate is a hyperparameter.

Contrast with parameter.

## learning rate

* + A scalar used to train a model via gradient descent. During each iteration, the gradient descent algorithm multiplies the learning rate by the gradient. The resulting product is called the gradient step.

Learning rate is a key hyperparameter.

## linear model

* + A model that assigns one weight per feature to make predictions. (Linear models also incorporate a bias.) By contrast, the relationship of weights to features in deep models is not one-to-one.

A linear model uses the following formula:



where:

* is the raw prediction. (In certain kinds of linear models, this raw prediction will be further modified. For example, see logistic regression.)
* is the bias.
* is a weight, so is the weight of the first feature, is the weight of the second feature, and so on.
* is a feature, so is the value of the first feature, is the value of the second feature, and so on.

For example, suppose a linear model for three features learns the following bias and weights:

* + - = 7
    - = -2.5
    - = -1.2
    - = 1.4

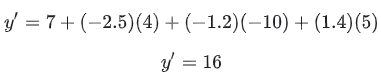
Therefore, given three features , the linear model uses the following equation to generate each prediction:



Suppose a particular example contains the following values:

* = 4
* = -10
* = 5

Plugging those values into the formula yields a prediction for this example:



Linear models tend to be easier to analyze and train than deep models. However, deep models can model complex relationships between features.

Linear regression and logistic regression are two types of linear models. Linear models include not only models that use the linear equation but also a broader set of models that use the linear equation as part of the formula. For example, logistic regression post-processes the raw prediction to calculate the prediction.

## linear regression

* + Using the raw output ( of a linear model as the actual prediction in a regression model. The goal of a regression problem is to make a real-valued prediction. For example, if the raw output ( a linear model is 8.37, then the prediction is 8.37.

Contrast linear regression with logistic regression. Also, contrast regression with classification.

## Model

* + The representation of what a machine learning system has learned from the training data. Within TensorFlow, model is an overloaded term, which can have either of the following two related meanings:

The TensorFlow graph that expresses the structure of how a prediction will be computed.

The particular weights and biases of that TensorFlow graph, which are determined by training.

## model function

* + The function within an Estimator that implements machine learning training, evaluation, and inference. For example, the training portion of a model function might handle tasks such as defining the topology of a deep neural network and identifying its optimizer function. When using premade Estimators, someone has already written the model function for you. When using custom Estimators, you must write the model function yourself.

For details about writing a model function, see the Creating Custom Estimators chapter in the TensorFlow Programmers Guide.

## model training

* + The process of determining the best model.

## Normalization

* + The process of converting an actual range of values into a standard range of values, typically -1 to +1 or 0 to 1. For example, suppose the natural range of a certain feature is 800 to 6,000. Through subtraction and division, you can normalize those values into the range -1 to +1.

See also scaling.

## numerical data

* + Features represented as integers or real-valued numbers. For example, in a real estate model, you would probably represent the size of a house (in square feet or square meters) as numerical data. Representing a feature as numerical data indicates that the feature's values have a mathematical relationship to each other and possibly to the label. For example, representing the size of a house as numerical data indicates that a 200 square-meter house is twice as large as a 100 square-meter house. Furthermore, the number of square meters in a house probably has some mathematical relationship to the price of the house.

Not all integer data should be represented as numerical data. For example, postal codes in some parts of the world are integers; however, integer postal codes should not be represented as numerical data in models. That's because a postal code of 20000 is not twice (or half) as potent as a postal code of 10000. Furthermore, although different postal codes do correlate to different real estate values, we can't assume that real estate values at postal code 20000 are twice as valuable as real estate values at postal code 10000. Postal codes should be represented as categorical data instead.

Numerical features are sometimes called continuous features.

## Optimizer

* + A specific implementation of the gradient descent algorithm. TensorFlow's base class for optimizers is tf.train.Optimizer. Popular optimizers include:
    - AdaGrad, which stands for ADAptive GRADient descent.
    - Adam, which stands for ADAptive with Momentum.

Different optimizers may leverage one or more of the following concepts to enhance the effectiveness of gradient descent on a given training set:

* + - momentum (Momentum)
    - update frequency
    - sparsity/regularization (Ftrl)
    - more complex math (Proximal, and others)

You might even imagine an NN-driven optimizer.

## Parameter

* + A variable of a model that the machine learning system trains on its own. For example, weights are parameters whose values the machine learning system gradually learns through successive training iterations. Contrast with hyperparameter.

## Prediction

* + A model's output when provided with an input example.

## prediction bias

* + A value indicating how far apart the average of predictions is from the average of labels in the dataset.

Not to be confused with the bias term in machine learning models or with bias in ethics and fairness.

## random forest

* + An ensemble approach to finding the decision tree that best fits the training data by creating many decision trees and then determining the "average" one. The "random" part of the term refers to building each of the decision trees from a random selection of features; the "forest" refers to the set of decision trees.

## regression model

* + A type of model that outputs continuous (typically, floating-point) values. Compare with classification models, which output discrete values, such as "day lily" or "tiger lily."

## Regularization

* + The penalty on a model's complexity. Regularization helps prevent overfitting. Different kinds of regularization include:
    - [**L1**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L1_regularization) regularization
    - [**L2**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization) regularization
    - dropout regularization
    - early stopping (this is not a formal regularization method, but can effectively limit overfitting)

## regularization rate

* + A scalar value, represented as lambda, specifying the relative importance of the regularization function. The following simplified loss equation shows the regularization rate's influence:



Raising the regularization rate reduces overfitting but may make the model less accurate.

## ridge regularization

* + Synonym for [**L2**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization) regularization. The term ridge regularization is more frequently used in pure statistics contexts, whereas [**L2**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization) regularization is used more often in machine learning.

## Scaling

* + A commonly used practice in feature engineering to tame a feature's range of values to match the range of other features in the dataset. For example, suppose that you want all floating-point features in the dataset to have a range of 0 to 1. Given a particular feature's range of 0 to 500, you could scale that feature by dividing each value by 500.

See also normalization.

## Subsampling

* + See pooling.

## Target

* + Synonym for label.

## Temporal data

* + Data recorded at different points in time. For example, winter coat sales recorded for each day of the year would be temporal data.

## test set

* + The subset of the dataset that you use to test your model after the model has gone through initial vetting by the validation set.

Contrast with training set and validation set.

## time series analysis

* + A subfield of machine learning and statistics that analyzes temporal data. Many types of machine learning problems require time series analysis, including classification, clustering, forecasting, and anomaly detection. For example, you could use time series analysis to forecast the future sales of winter coats by month based on historical sales data.

## Training

* + The process of determining the ideal parameters comprising a model.

## training set

* + The subset of the dataset used to train a model.

Contrast with validation set and test set.

## Validation

* + A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model’s performance generalizes beyond the training set.

Contrast with test set.

## validation set

A subset of the dataset—disjoint from the training set—used in validation. Contrast with training set and test set.

# Classification

## accuracy

* + The fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:



In binary classification, accuracy has the following definition:



## area under the PR curve

* + See PR AUC (Area under the PR Curve).

## area under the ROC curve

* + See AUC (Area under the ROC curve).

## AUC (Area under the ROC Curve)

* + An evaluation metric that considers all possible classification thresholds.

The Area Under the ROC curve is the probability that a classifier will be more confident that a randomly chosen positive example is actually positive than that a randomly chosen negative example is positive.

## Batch

* + The set of examples used in one iteration (that is, one gradient update) of model training.

See also batch size.

## batch normalization

* + Normalizing the input or output of the activation functions in a hidden layer. Batch normalization can provide the following benefits:
    - Make neural networks more stable by protecting against outlier weights.
    - Enable higher learning rates.
    - Reduce overfitting.

## batch size

* + The number of examples in a batch. For example, the batch size of SGD is 1, while the batch size of a mini-batch is usually between 10 and 1000. Batch size is usually fixed during training and inference; however, TensorFlow does permit dynamic batch sizes.

## Bayesian neural network

* + A probabilistic neural network that accounts for uncertainty in weights and outputs. A standard neural network regression model typically predicts a scalar value; for example, a model predicts a house price of 853,000. By contrast, a Bayesian neural network predicts a distribution of values; for example, a model predicts a house price of 853,000 with a standard deviation of 67,200. A Bayesian neural network relies on Bayes' Theorem to calculate uncertainties in weights and predictions. A Bayesian neural network can be useful when it is important to quantify uncertainty, such as in models related to pharmaceuticals. Bayesian neural networks can also help prevent overfitting.

## binary classification

* + A type of classification task that outputs one of two mutually exclusive classes. For example, a machine learning model that evaluates email messages and outputs either "spam" or "not spam" is a binary classifier.

## Boosting

* + A machine learning technique that iteratively combines a set of simple and not very accurate classifiers (referred to as "weak" classifiers) into a classifier with high accuracy (a "strong" classifier) by upweighting the examples that the model is currently misclassfying.

## categorical data

* + Features having a discrete set of possible values. For example, consider a categorical feature named house style, which has a discrete set of three possible values: Tudor, ranch, colonial. By representing house style as categorical data, the model can learn the separate impacts of Tudor, ranch, and colonial on house price.

Sometimes, values in the discrete set are mutually exclusive, and only one value can be applied to a given example. For example, a car maker categorical feature would probably permit only a single value (Toyota) per example. Other times, more than one value may be applicable. A single car could be painted more than one different color, so a car color categorical feature would likely permit a single example to have multiple values (for example, red and white).

Categorical features are sometimes called discrete features.

Contrast with numerical data.

## Class

* + One of a set of enumerated target values for a label. For example, in a binary classification model that detects spam, the two classes are spam and not spam. In a multi-class classification model that identifies dog breeds, the classes would be poodle, beagle, pug, and so on.

## classification model

* + A type of machine learning model for distinguishing among two or more discrete classes. For example, a natural language processing classification model could determine whether an input sentence was in French, Spanish, or Italian. Compare with regression model.

## classification threshold

* + A scalar-value criterion that is applied to a model's predicted score in order to separate the positive class from the negative class. Used when mapping logistic regression results to binary classification. For example, consider a logistic regression model that determines the probability of a given email message being spam. If the classification threshold is 0.9, then logistic regression values above 0.9 are classified as spam and those below 0.9 are classified as not spam.

## class-imbalanced dataset

* + A binary classification problem in which the labels for the two classes have significantly different frequencies. For example, a disease dataset in which 0.0001 of examples have positive labels and 0.9999 have negative labels is a class-imbalanced problem, but a football game predictor in which 0.51 of examples label one team winning and 0.49 label the other team winning is not a class-imbalanced problem.

## Clipping

* + A technique for handling outliers. Specifically, reducing feature values that are greater than a set maximum value down to that maximum value. Also, increasing feature values that are less than a specific minimum value up to that minimum value.

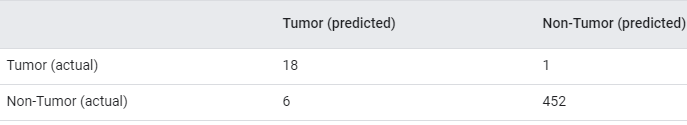
For example, suppose that only a few feature values fall outside the range 40–60. In this case, you could do the following:

* Clip all values over 60 to be exactly 60.
* Clip all values under 40 to be exactly 40.

In addition to bringing input values within a designated range, clipping can also used to force gradient values within a designated range during training.

## confusion matrix

* + An NxN table that summarizes how successful a classification model's predictions were; that is, the correlation between the label and the model's classification. One axis of a confusion matrix is the label that the model predicted, and the other axis is the actual label. N represents the number of classes. In a binary classification problem, N=2. For example, here is a sample confusion matrix for a binary classification problem:



The preceding confusion matrix shows that of the 19 samples that actually had tumors, the model correctly classified 18 as having tumors (18 true positives), and incorrectly classified 1 as not having a tumor (1 false negative). Similarly, of 458 samples that actually did not have tumors, 452 were correctly classified (452 true negatives) and 6 were incorrectly classified (6 false positives).

The confusion matrix for a multi-class classification problem can help you determine mistake patterns. For example, a confusion matrix could reveal that a model trained to recognize handwritten digits tends to mistakenly predict 9 instead of 4, or 1 instead of 7.

Confusion matrices contain sufficient information to calculate a variety of performance metrics, including precision and recall.

## cross-validation

* + A mechanism for estimating how well a model will generalize to new data by testing the model against one or more non-overlapping data subsets withheld from the training set.

## Data set or dataset

* + A collection of examples.

## decision boundary

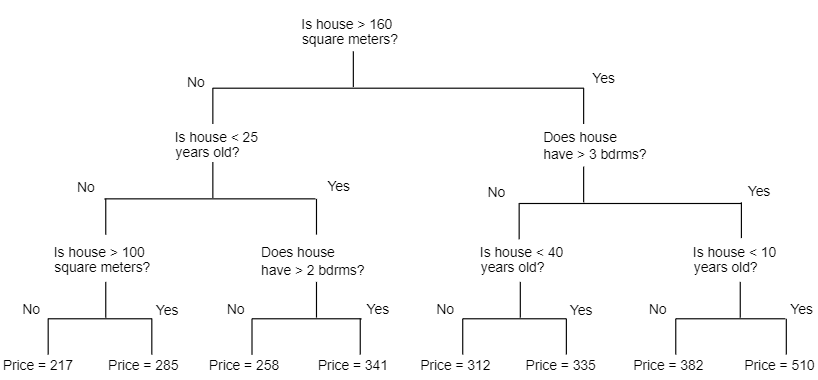
* + The separator between classes learned by a model in a binary class or multi-class classification problems. For example, in the following image representing a binary classification problem, the decision boundary is the frontier between the orange class and the blue class:A
    well-defined boundary between one class and another.

## decision threshold

* + Synonym for classification threshold.

## decision tree

* + A model represented as a sequence of branching statements. For example, the following over-simplified decision tree branches a few times to predict the price of a house (in thousands of USD). According to this decision tree, a house larger than 160 square meters, having more than three bedrooms, and built less than 10 years ago would have a predicted price of 510 thousand USD.



## 

## Ensemble

* + A merger of the predictions of multiple models. You can create an ensemble via one or more of the following:
    - different initializations
    - different hyperparameters
    - different overall structure

Deep and wide models are a kind of ensemble.

## false negative (FN)

* + An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam.

## false positive (FP)

* + An example in which the model mistakenly predicted the positive class. For example, the model inferred that a particular email message was spam (the positive class), but that email message was actually not spam.

## false positive rate (FPR)

* + The x-axis in an ROC curve. The false positive rate is defined as follows:



## Feature

* + An input variable used in making predictions.

## feature engineering

* + The process of determining which features might be useful in training a model, and then converting raw data from log files and other sources into said features. In TensorFlow, feature engineering often means converting raw log file entries to tf.Example protocol buffers. See also tf.Transform.

Feature engineering is sometimes called feature extraction.

## feature extraction

* + Overloaded term having either of the following definitions:
    - Retrieving intermediate feature representations calculated by an unsupervised or pretrained model (for example, hidden layer values in a neural network) for use in another model as input.
    - Synonym for feature engineering.

## feature set

* + The group of features your machine learning model trains on. For example, postal code, property size, and property condition might comprise a simple feature set for a model that predicts housing prices.

## few-shot learning

* + A machine learning approach, often used for object classification, designed to learn effective classifiers from only a small number of training examples.

See also one-shot learning.

## generalized linear model

* + A generalization of least squares regression models, which are based on Gaussian noise, to other types of models based on other types of noise, such as Poisson noise or categorical noise. Examples of generalized linear models include:
    - logistic regression
    - multi-class regression
    - least squares regression

The parameters of a generalized linear model can be found through convex optimization.

Generalized linear models exhibit the following properties:

* The average prediction of the optimal least squares regression model is equal to the average label on the training data.
* The average probability predicted by the optimal logistic regression model is equal to the average label on the training data.

The power of a generalized linear model is limited by its features. Unlike a deep model, a generalized linear model cannot "learn new features."

## Hashing

* + In machine learning, a mechanism for bucketing categorical data, particularly when the number of categories is large, but the number of categories actually appearing in the dataset is comparatively small.

For example, Earth is home to about 60,000 tree species. You could represent each of the 60,000 tree species in 60,000 separate categorical buckets. Alternatively, if only 200 of those tree species actually appear in a dataset, you could use hashing to divide tree species into perhaps 500 buckets.

A single bucket could contain multiple tree species. For example, hashing could place baobab and red maple—two genetically dissimilar species—into the same bucket. Regardless, hashing is still a good way to map large categorical sets into the desired number of buckets. Hashing turns a categorical feature having a large number of possible values into a much smaller number of values by grouping values in a deterministic way.

## Hyperparameter

* + The "knobs" that you tweak during successive runs of training a model. For example, learning rate is a hyperparameter.

Contrast with parameter.

## Hyperplane

* + A boundary that separates a space into two subspaces. For example, a line is a hyperplane in two dimensions and a plane is a hyperplane in three dimensions. More typically in machine learning, a hyperplane is the boundary separating a high-dimensional space. Kernel Support Vector Machines use hyperplanes to separate positive classes from negative classes, often in a very high-dimensional space.

## image recognition

* + A process that classifies object(s), pattern(s), or concept(s) in an image. Image recognition is also known as image classification.

For more information, see ML Practicum: Image Classification.

## imbalanced dataset

* + Synonym for class-imbalanced dataset.

## Kernel Support Vector Machines (KSVMs)

* + A classification algorithm that seeks to maximize the margin between positive and negative classes by mapping input data vectors to a higher dimensional space. For example, consider a classification problem in which the input dataset has a hundred features. To maximize the margin between positive and negative classes, a KSVM could internally map those features into a million-dimension space. KSVMs uses a loss function called hinge loss.

## Label

* + In supervised learning, the "answer" or "result" portion of an example. Each example in a labeled dataset consists of one or more features and a label. For instance, in a housing dataset, the features might include the number of bedrooms, the number of bathrooms, and the age of the house, while the label might be the house's price. In a spam detection dataset, the features might include the subject line, the sender, and the email message itself, while the label would probably be either "spam" or "not spam."

## labeled example

* + An example that contains features and a label. In supervised training, models learn from labeled examples.

## logistic regression

* + A classification model that uses a sigmoid function to convert a linear model's raw prediction into a value between 0 and 1. You can interpret the value between 0 and 1 in either of the following two ways:
    - As a probability that the example belongs to the positive class in a binary classification problem.
    - As a value to be compared against a classification threshold. If the value is equal to or above the classification threshold, the system classifies the example as the positive class. Conversely, if the value is below the given threshold, the system classifies the example as the negative class. For example, suppose the classification threshold is 0.82:
      * Imagine an example that produces a raw prediction of 2.6. The sigmoid of 2.6 is 0.93. Since 0.93 is greater than 0.82, the system classifies this example as the positive class.
      * Imagine a different example that produces a raw prediction of 1.3. The sigmoid of 1.3 is 0.79. Since 0.79 is less than 0.82, the system classifies that example as the negative class.

Although logistic regression is often used in binary classification problems, logistic regression can also be used in multi-class classification problems (where it becomes called multi-class logistic regression or multinomial regression).

## Logits

* + The vector of raw (non-normalized) predictions that a classification model generates, which is ordinarily then passed to a normalization function. If the model is solving a multi-class classification problem, logits typically become an input to the softmax function. The softmax function then generates a vector of (normalized) probabilities with one value for each possible class.

In addition, logits sometimes refer to the element-wise inverse of the sigmoid function. For more information, see tf.nn.sigmoid\_cross\_entropy\_with\_logits.

## majority class

* + The more common label in a class-imbalanced dataset. For example, given a dataset containing 99% non-spam labels and 1% spam labels, the non-spam labels are the majority class.

## MNIST

* + A public-domain dataset compiled by LeCun, Cortes, and Burges containing 60,000 images, each image showing how a human manually wrote a particular digit from 0–9. Each image is stored as a 28x28 array of integers, where each integer is a grayscale value between 0 and 255, inclusive.

MNIST is a canonical dataset for machine learning, often used to test new machine learning approaches. For details, see The MNIST Database of Handwritten Digits.

## model function

* + The function within an Estimator that implements machine learning training, evaluation, and inference. For example, the training portion of a model function might handle tasks such as defining the topology of a deep neural network and identifying its optimizer function. When using premade Estimators, someone has already written the model function for you. When using custom Estimators, you must write the model function yourself.

For details about writing a model function, see the Creating Custom Estimators chapter in the TensorFlow Programmers Guide.

## model training

* + The process of determining the best model.

## multi-class classification

* + Classification problems that distinguish among more than two classes. For example, there are approximately 128 species of maple trees, so a model that categorized maple tree species would be multi-class. Conversely, a model that divided emails into only two categories (spam and not spam) would be a binary classification model.

## multi-class logistic regression

* + Using logistic regression in multi-class classification problems.

## multinomial classification

* + Synonym for multi-class classification.

## negative class

* + In binary classification, one class is termed positive and the other is termed negative. The positive class is the thing we're looking for and the negative class is the other possibility. For example, the negative class in a medical test might be "not tumor." The negative class in an email classifier might be "not spam." See also positive class.

## Normalization

* + The process of converting an actual range of values into a standard range of values, typically -1 to +1 or 0 to 1. For example, suppose the natural range of a certain feature is 800 to 6,000. Through subtraction and division, you can normalize those values into the range -1 to +1.

See also scaling.

## numerical data

* + Features represented as integers or real-valued numbers. For example, in a real estate model, you would probably represent the size of a house (in square feet or square meters) as numerical data. Representing a feature as numerical data indicates that the feature's values have a mathematical relationship to each other and possibly to the label. For example, representing the size of a house as numerical data indicates that a 200 square-meter house is twice as large as a 100 square-meter house. Furthermore, the number of square meters in a house probably has some mathematical relationship to the price of the house.

Not all integer data should be represented as numerical data. For example, postal codes in some parts of the world are integers; however, integer postal codes should not be represented as numerical data in models. That's because a postal code of 20000 is not twice (or half) as potent as a postal code of 10000. Furthermore, although different postal codes do correlate to different real estate values, we can't assume that real estate values at postal code 20000 are twice as valuable as real estate values at postal code 10000. Postal codes should be represented as categorical data instead.

Numerical features are sometimes called continuous features.

## one-hot encoding

* + A sparse vector in which:
    - One element is set to 1.
    - All other elements are set to 0.

One-hot encoding is commonly used to represent strings or identifiers that have a finite set of possible values. For example, suppose a given botany dataset chronicles 15,000 different species, each denoted with a unique string identifier. As part of feature engineering, you'll probably encode those string identifiers as one-hot vectors in which the vector has a size of 15,000.

## one-shot learning

* + A machine learning approach, often used for object classification, designed to learn effective classifiers from a single training example.

See also few-shot learning.

## one-vs.-all

* + Given a classification problem with N possible solutions, a one-vs.-all solution consists of N separate binary classifiers—one binary classifier for each possible outcome. For example, given a model that classifies examples as animal, vegetable, or mineral, a one-vs.-all solution would provide the following three separate binary classifiers:
    - animal vs. not animal
    - vegetable vs. not vegetable
    - mineral vs. not mineral

## Parameter

* + A variable of a model that the machine learning system trains on its own. For example, weights are parameters whose values the machine learning system gradually learns through successive training iterations. Contrast with hyperparameter.

## positive class

* + In binary classification, the two possible classes are labeled as positive and negative. The positive outcome is the thing we're testing for. (Admittedly, we're simultaneously testing for both outcomes, but play along.) For example, the positive class in a medical test might be "tumor." The positive class in an email classifier might be "spam."

Contrast with negative class.

## post-processing

* + Processing the output of a model after the model has been run. Post-processing can be used to enforce fairness constraints without modifying models themselves.

For example, one might apply post-processing to a binary classifier by setting a classification threshold such that equality of opportunity is maintained for some attribute by checking that the true positive rate is the same for all values of that attribute.

## PR AUC (area under the PR curve)

* + See PR AUC (Area under the PR Curve).

## Precision

* + A metric for classification models. Precision identifies the frequency with which a model was correct when predicting the positive class. That is:



## precision-recall curve

* + A curve of precision vs. recall at different classification thresholds.

## Prediction

* + A model's output when provided with an input example.

## prediction bias

* + A value indicating how far apart the average of predictions is from the average of labels in the dataset.

Not to be confused with the bias term in machine learning models or with bias in ethics and fairness.

## preprocessing

* + Processing data before it's used to train a model. Preprocessing could be as simple as removing words from an English text corpus that don't occur in the English dictionary, or could be as complex as re-expressing data points in a way that eliminates as many attributes that are correlated with sensitive attributes as possible. Preprocessing can help satisfy fairness constraints.

## proxy (sensitive attributes)

* + An attribute used as a stand-in for a sensitive attribute. For example, an individual's postal code might be used as a proxy for their income, race, or ethnicity.

## proxy labels

* + Data used to approximate labels not directly available in a dataset.

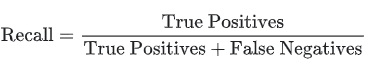
For example, suppose you want is it raining? to be a Boolean label for your dataset, but the dataset doesn't contain rain data. If photographs are available, you might establish pictures of people carrying umbrellas as a proxy label for is it raining? However, proxy labels may distort results. For example, in some places, it may be more common to carry umbrellas to protect against sun than the rain.

## rank (ordinality)

* + The ordinal position of a class in a machine learning problem that categorizes classes from highest to lowest. For example, a behavior ranking system could rank a dog's rewards from highest (a steak) to lowest (wilted kale).

## Recall

* + A metric for classification models that answers the following question: Out of all the possible positive labels, how many did the model correctly identify? That is:



## Regularization

* + The penalty on a model's complexity. Regularization helps prevent overfitting. Different kinds of regularization include:
    - [**L1**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L1_regularization) regularization
    - [**L2**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization) regularization
    - dropout regularization
    - early stopping (this is not a formal regularization method, but can effectively limit overfitting)

## sampling bias

* + See selection bias.

## sentiment analysis

* + Using statistical or machine learning algorithms to determine a group's overall attitude—positive or negative—toward a service, product, organization, or topic. For example, using natural language understanding, an algorithm could perform sentiment analysis on the textual feedback from a university course to determine the degree to which students generally liked or disliked the course.

## sigmoid function

* + A function that maps logistic or multinomial regression output (log odds) to probabilities, returning a value between 0 and 1. The sigmoid function has the following formula:



where *σ* in logistic regression problems is simply:



In other words, the sigmoid function converts *σ* into a probability between 0 and 1.

In some neural networks, the sigmoid function acts as the activation function.

## size invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the size of the image changes. For example, the algorithm can still identify a cat whether it consumes 2M pixels or 200K pixels. Note that even the best image classification algorithms still have practical limits on size invariance. For example, an algorithm (or human) is unlikely to correctly classify a cat image consuming only 20 pixels.

See also translational invariance and rotational invariance.

## Softmax

* + A function that provides probabilities for each possible class in a multi-class classification model. The probabilities add up to exactly 1.0. For example, softmax might determine that the probability of a particular image being a dog at 0.9, a cat at 0.08, and a horse at 0.02. (Also called full softmax.)

Contrast with candidate sampling.

## Subsampling

* + See pooling.

## synthetic feature

* + A feature not present among the input features, but created from one or more of them. Kinds of synthetic features include:
    - Bucketing a continuous feature into range bins.
    - Multiplying (or dividing) one feature value by other feature value(s) or by itself.
    - Creating a feature cross.

Features created by normalizing or scaling alone are not considered synthetic features.

## Target

* + Synonym for label.

## test set

* + The subset of the dataset that you use to test your model after the model has gone through initial vetting by the validation set.

Contrast with training set and validation set.

## Training

* + The process of determining the ideal parameters comprising a model.

## training set

* + The subset of the dataset used to train a model.

Contrast with validation set and test set.

## true negative (TN)

* + An example in which the model correctly predicted the negative class. For example, the model inferred that a particular email message was not spam, and that email message really was not spam.

## true positive (TP)

* + An example in which the model correctly predicted the positive class. For example, the model inferred that a particular email message was spam, and that email message really was spam.

## true positive rate (TPR)

* + Synonym for recall. That is:



True positive rate is the y-axis in an ROC curve.

## translational invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the position of objects within the image changes. For example, the algorithm can still identify a dog, whether it is in the center of the frame or at the left end of the frame.

See also size invariance and rotational invariance.

## unawareness (to a sensitive attribute)

* + A situation in which sensitive attributes are present, but not included in the training data. Because sensitive attributes are often correlated with other attributes of one’s data, a model trained with unawareness about a sensitive attribute could still have disparate impact with respect to that attribute, or violate other fairness constraints.

## Validation

* + A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model’s performance generalizes beyond the training set.

Contrast with test set.

## validation set

* + A subset of the dataset—disjoint from the training set—used in validation.

Contrast with training set and test set.

# Reinforcement learning

## activation function

* + A function (for example, ReLU or sigmoid) that takes in the weighted sum of all of the inputs from the previous layer and then generates and passes an output value (typically nonlinear) to the next layer.

## Agent

* + In reinforcement learning, the entity that uses a policy to maximize expected return gained from transitioning between states of the environment.

## artificial intelligence

* + A non-human program or model that can solve sophisticated tasks. For example, a program or model that translates text or a program or model that identifies diseases from radiologic images both exhibit artificial intelligence.

Formally, machine learning is a sub-field of artificial intelligence. However, in recent years, some organizations have begun using the terms artificial intelligence and machine learning interchangeably.

## Batch

* + The set of examples used in one iteration (that is, one gradient update) of model training.

## batch normalization

* + Normalizing the input or output of the activation functions in a hidden layer. Batch normalization can provide the following benefits:
    - Make neural networks more stable by protecting against outlier weights.
    - Enable higher learning rates.
    - Reduce overfitting.

## batch size

* + The number of examples in a batch. For example, the batch size of SGD is 1, while the batch size of a mini-batch is usually between 10 and 1000. Batch size is usually fixed during training and inference; however, TensorFlow does permit dynamic batch sizes.

## Bellman equation

* + In reinforcement learning, the following identity satisfied by the optimal Q-function:



Reinforcement learning algorithms apply this identity to create Q-learning via the following update rule:



Beyond reinforcement learning, the Bellman equation has applications to dynamic programming. See the Wikipedia entry for Bellman Equation.

## deep model

* + A type of neural network containing multiple hidden layers.

Contrast with wide model.

## deep neural network

* + Synonym for deep model.

## Deep Q-Network (DQN)

* + In Q-learning, a deep neural network that predicts Q-functions.

Critic is a synonym for Deep Q-Network.

## DQN

* + Abbreviation for Deep Q-Network.

## dynamic model

* + A model that is trained online in a continuously updating fashion. That is, data is continuously entering the model.

## Environment

* + In reinforcement learning, the world that contains the agent and allows the agent to observe that world's state. For example, the represented world can be a game like chess, or a physical world like a maze. When the agent applies an action to the environment, then the environment transitions between states.

## Episode

* + In reinforcement learning, each of the repeated attempts by the agent to learn an environment.

## Epoch

* + A full training pass over the entire dataset such that each example has been seen once. Thus, an epoch represents N / batch size training iterations, where N is the total number of examples.

## epsilon greedy policy

* + In reinforcement learning, a policy that either follows a random policy with epsilon probability or a greedy policy otherwise. For example, if epsilon is 0.9, then the policy follows a random policy 90% of the time and a greedy policy 10% of the time.

Over successive episodes, the algorithm reduces epsilon’s value in order to shift from following a random policy to following a greedy policy. By shifting the policy, the agent first randomly explores the environment and then greedily exploits the results of random exploration.

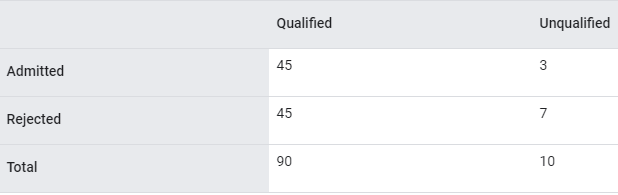
## equality of opportunity

* + A fairness metric that checks whether, for a preferred label (one that confers an advantage or benefit to a person) and a given attribute, a classifier predicts that preferred label equally well for all values of that attribute. In other words, equality of opportunity measures whether the people who should qualify for an opportunity are equally likely to do so regardless of their group membership.

For example, suppose Glubbdubdrib University admits both Lilliputians and Brobdingnagians to a rigorous mathematics program. Lilliputians’ secondary schools offer a robust curriculum of math classes, and the vast majority of students are qualified for the university program. Brobdingnagians’ secondary schools don’t offer math classes at all, and as a result, far fewer of their students are qualified. Equality of opportunity is satisfied for the preferred label of "admitted" with respect to nationality (Lilliputian or Brobdingnagian) if qualified students are equally likely to be admitted irrespective of whether they're a Lilliputian or a Brobdingnagian.

For example, let's say 100 Lilliputians and 100 Brobdingnagians apply to Glubbdubdrib University, and admissions decisions are made as follows:

Table 1. Lilliputian applicants (90% are qualified)

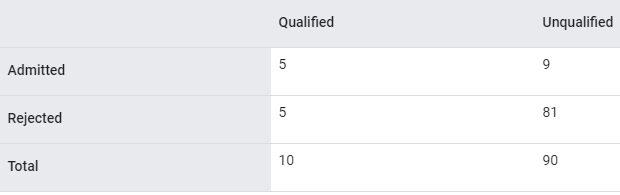


Percentage of qualified students admitted: 45/90 = 50%

Percentage of unqualified students rejected: 7/10 = 70%

Total percentage of Lilliputian students admitted: (45+3)/100 = 48%

Table 2. Brobdingnagian applicants (10% are qualified):



Percentage of qualified students admitted: 5/10 = 50%

Percentage of unqualified students rejected: 81/90 = 90%

Total percentage of Brobdingnagian students admitted: (5+9)/100 = 14%

The preceding examples satisfy equality of opportunity for acceptance of qualified students because qualified Lilliputians and Brobdingnagians both have a 50% chance of being admitted.

Note: While equality of opportunity is satisfied, the following two fairness metrics are not satisfied:

demographic parity: Lilliputians and Brobdingnagians are admitted to the university at different rates; 48% of Lilliputians students are admitted, but only 14% of Brobdingnagian students are admitted.

equalized odds: While qualified Lilliputian and Brobdingnagian students both have the same chance of being admitted, the additional constraint that unqualified Lilliputians and Brobdingnagians both have the same chance of being rejected is not satisfied. Unqualified Lilliputians have a 70% rejection rate, whereas unqualified Brobdingnagians have a 90% rejection rate.

See "Equality of Opportunity in Supervised Learning" for a more detailed discussion of equality of opportunity. Also see "Attacking discrimination with smarter machine learning" for a visualization exploring the tradeoffs when optimizing for equality of opportunity.

## experience replay

* + In reinforcement learning, a DQN technique used to reduce temporal correlations in training data. The agent stores state transitions in a replay buffer, and then samples transitions from the replay buffer to create training data.

## GAN

* + Abbreviation for generative adversarial network.

## generative adversarial network (GAN)

* + A system to create new data in which a generator creates data and a discriminator determines whether that created data is valid or invalid.

## generative model

* + Practically speaking, a model that does either of the following:
    - Creates (generates) new examples from the training dataset. For example, a generative model could create poetry after training on a dataset of poems. The generator part of a generative adversarial network falls into this category.
    - Determines the probability that a new example comes from the training set, or was created from the same mechanism that created the training set. For example, after training on a dataset consisting of English sentences, a generative model could determine the probability that new input is a valid English sentence.

A generative model can theoretically discern the distribution of examples or particular features in a dataset. That is:

p(examples)

Unsupervised learning models are generative.

Contrast with discriminative models.

## greedy policy

* + In reinforcement learning, a policy that always chooses the action with the highest expected return.

## Markov decision process (MDP)

* + A graph representing the decision-making model where decisions (or actions) are taken to navigate a sequence of states under the assumption that the Markov property holds. In reinforcement learning, these transitions between states return a numerical reward.

## Markov property

* + A property of certain environments, where state transitions are entirely determined by information implicit in the current state and the agent’s action.

## mini-batch

* + A small, randomly selected subset of the entire batch of examples run together in a single iteration of training or inference. The batch size of a mini-batch is usually between 10 and 1,000. It is much more efficient to calculate the loss on a mini-batch than on the full training data.

## mini-batch stochastic gradient descent

* + A gradient descent algorithm that uses mini-batches. In other words, mini-batch stochastic gradient descent estimates the gradient based on a small subset of the training data. Regular stochastic gradient descent uses a mini-batch of size 1.

## Model

* + The representation of what a machine learning system has learned from the training data. Within TensorFlow, model is an overloaded term, which can have either of the following two related meanings:
    - The TensorFlow graph that expresses the structure of how a prediction will be computed.
    - The particular weights and biases of that TensorFlow graph, which are determined by training.

## model function

* + The function within an Estimator that implements machine learning training, evaluation, and inference. For example, the training portion of a model function might handle tasks such as defining the topology of a deep neural network and identifying its optimizer function. When using premade Estimators, someone has already written the model function for you. When using custom Estimators, you must write the model function yourself.

For details about writing a model function, see the Creating Custom Estimators chapter in the TensorFlow Programmers Guide.

## model training

* + The process of determining the best model.

## offline inference

* + Generating a group of predictions, storing those predictions, and then retrieving those predictions on demand. Contrast with online inference.

## Policy

* + In reinforcement learning, an agent's probabilistic mapping from states to actions.

## Q-function

* + In reinforcement learning, the function that predicts the expected return from taking an action in a state and then following a given policy.

Q-function is also known as state-action value function.

## Q-learning

* + In reinforcement learning, an algorithm that allows an agent to learn the optimal Q-function of a Markov decision process by applying the Bellman equation. The Markov decision process models an environment.

## random policy

* + In reinforcement learning, a policy that chooses an action at random.

## reinforcement learning (RL)

* + A family of algorithms that learn an optimal policy, whose goal is to maximize return when interacting with an environment. For example, the ultimate reward of most games is victory. Reinforcement learning systems can become expert at playing complex games by evaluating sequences of previous game moves that ultimately led to wins and sequences that ultimately led to losses.

## Return

* + In reinforcement learning, given a certain policy and a certain state, the return is the sum of all rewards that the agent expects to receive when following the policy from the state to the end of the episode. The agent accounts for the delayed nature of expected rewards by discounting rewards according to the state transitions required to obtain the reward.

Therefore, if the discount factor is and ,. . . , denote the rewards until the end of the episode, then the return calculation is as follows:



## Reward

* + In reinforcement learning, the numerical result of taking an action in a state, as defined by the environment.

## State

* + In reinforcement learning, the parameter values that describe the current configuration of the environment, which the agent uses to choose an action.

## state-action value function

* + Synonym for Q-function.

## static model

* + A model that is trained offline.

## Step

* + A forward and backward evaluation of one batch.

## synthetic feature

* + A feature not present among the input features, but created from one or more of them. Kinds of synthetic features include:
    - Bucketing a continuous feature into range bins.
    - Multiplying (or dividing) one feature value by other feature value(s) or by itself.
    - Creating a feature cross.

Features created by normalizing or scaling alone are not considered synthetic features.

## tabular Q-learning

* + In reinforcement learning, implementing Q-learning by using a table to store the Q-functions for every combination of state and action.

## Target

* + Synonym for label.

## Trajectory

* + In reinforcement learning, a sequence of tuples that represent a sequence of state transitions of the agent, where each tuple corresponds to the state, action, reward, and next state for a given state transition.

# Model evaluation

## A/B testing

## Accuracy

* + The fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:



In binary classification, accuracy has the following definition:



## area under the PR curve

* + See PR AUC (Area under the PR Curve).

## area under the ROC curve

* + See AUC (Area under the ROC curve).

## AUC (Area under the ROC Curve)

* + An evaluation metric that considers all possible classification thresholds.

The Area Under the ROC curve is the probability that a classifier will be more confident that a randomly chosen positive example is actually positive than that a randomly chosen negative example is positive.

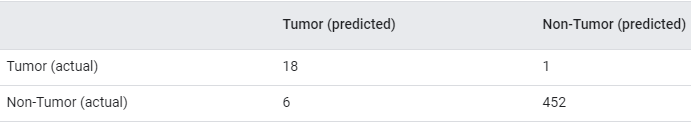
## confirmation bias

* + The tendency to search for, interpret, favor, and recall information in a way that confirms one's preexisting beliefs or hypotheses. Machine learning developers may inadvertently collect or label data in ways that influence an outcome supporting their existing beliefs. Confirmation bias is a form of implicit bias.

Experimenter's bias is a form of confirmation bias in which an experimenter continues training models until a preexisting hypothesis is confirmed.

## confusion matrix

* + An NxN table that summarizes how successful a classification model's predictions were; that is, the correlation between the label and the model's classification. One axis of a confusion matrix is the label that the model predicted, and the other axis is the actual label. N represents the number of classes. In a binary classification problem, N=2. For example, here is a sample confusion matrix for a binary classification problem:



The preceding confusion matrix shows that of the 19 samples that actually had tumors, the model correctly classified 18 as having tumors (18 true positives), and incorrectly classified 1 as not having a tumor (1 false negative). Similarly, of 458 samples that actually did not have tumors, 452 were correctly classified (452 true negatives) and 6 were incorrectly classified (6 false positives).

The confusion matrix for a multi-class classification problem can help you determine mistake patterns. For example, a confusion matrix could reveal that a model trained to recognize handwritten digits tends to mistakenly predict 9 instead of 4, or 1 instead of 7.

Confusion matrices contain sufficient information to calculate a variety of performance metrics, including precision and recall.

## Cost

* + Synonym for loss.

## coverage bias

* + See selection bias.

## cross-entropy

* + A generalization of Log Loss to multi-class classification problems. Cross-entropy quantifies the difference between two probability distributions. See also perplexity.

## cross-validation

* + A mechanism for estimating how well a model will generalize to new data by testing the model against one or more non-overlapping data subsets withheld from the training set.

## decision threshold

* + Synonym for classification threshold.

## dropout regularization

* + A form of regularization useful in training neural networks. Dropout regularization works by removing a random selection of a fixed number of the units in a network layer for a single gradient step. The more units dropped out, the stronger the regularization. This is analogous to training the network to emulate an exponentially large ensemble of smaller networks. For full details, see Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

## early stopping

* + A method for regularization that involves ending model training before training loss finishes decreasing. In early stopping, you end model training when the loss on a validation dataset starts to increase, that is, when generalization performance worsens.

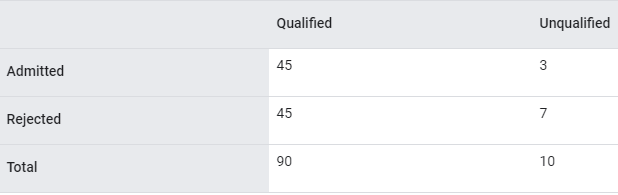
## equality of opportunity

* + A fairness metric that checks whether, for a preferred label (one that confers an advantage or benefit to a person) and a given attribute, a classifier predicts that preferred label equally well for all values of that attribute. In other words, equality of opportunity measures whether the people who should qualify for an opportunity are equally likely to do so regardless of their group membership.

For example, suppose Glubbdubdrib University admits both Lilliputians and Brobdingnagians to a rigorous mathematics program. Lilliputians’ secondary schools offer a robust curriculum of math classes, and the vast majority of students are qualified for the university program. Brobdingnagians’ secondary schools don’t offer math classes at all, and as a result, far fewer of their students are qualified. Equality of opportunity is satisfied for the preferred label of "admitted" with respect to nationality (Lilliputian or Brobdingnagian) if qualified students are equally likely to be admitted irrespective of whether they're a Lilliputian or a Brobdingnagian.

For example, let's say 100 Lilliputians and 100 Brobdingnagians apply to Glubbdubdrib University, and admissions decisions are made as follows:

Table 1. Lilliputian applicants (90% are qualified)

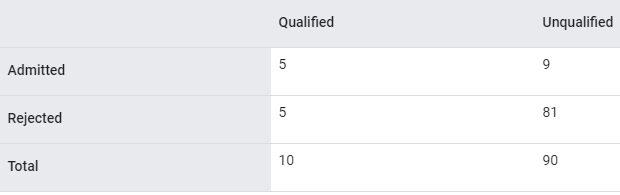


Percentage of qualified students admitted: 45/90 = 50%

Percentage of unqualified students rejected: 7/10 = 70%

Total percentage of Lilliputian students admitted: (45+3)/100 = 48%

Table 2. Brobdingnagian applicants (10% are qualified):



Percentage of qualified students admitted: 5/10 = 50%

Percentage of unqualified students rejected: 81/90 = 90%

Total percentage of Brobdingnagian students admitted: (5+9)/100 = 14%

The preceding examples satisfy equality of opportunity for acceptance of qualified students because qualified Lilliputians and Brobdingnagians both have a 50% chance of being admitted.

Note: While equality of opportunity is satisfied, the following two fairness metrics are not satisfied:

demographic parity: Lilliputians and Brobdingnagians are admitted to the university at different rates; 48% of Lilliputians students are admitted, but only 14% of Brobdingnagian students are admitted.

equalized odds: While qualified Lilliputian and Brobdingnagian students both have the same chance of being admitted, the additional constraint that unqualified Lilliputians and Brobdingnagians both have the same chance of being rejected is not satisfied. Unqualified Lilliputians have a 70% rejection rate, whereas unqualified Brobdingnagians have a 90% rejection rate.

See "Equality of Opportunity in Supervised Learning" for a more detailed discussion of equality of opportunity. Also see "Attacking discrimination with smarter machine learning" for a visualization exploring the tradeoffs when optimizing for equality of opportunity.

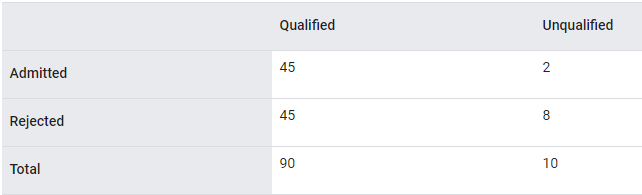
## equalized odds

* + A fairness metric that checks if, for any particular label and attribute, a classifier predicts that label equally well for all values of that attribute.

For example, suppose Glubbdubdrib University admits both Lilliputians and Brobdingnagians to a rigorous mathematics program. Lilliputians' secondary schools offer a robust curriculum of math classes, and the vast majority of students are qualified for the university program. Brobdingnagians' secondary schools don’t offer math classes at all, and as a result, far fewer of their students are qualified. Equalized odds is satisfied provided that no matter whether an applicant is a Lilliputian or a Brobdingnagian, if they are qualified, they are equally as likely to get admitted to the program, and if they are not qualified, they are equally as likely to get rejected.

Let’s say 100 Lilliputians and 100 Brobdingnagians apply to Glubbdubdrib University, and admissions decisions are made as follows:

Table 3. Lilliputian applicants (90% are qualified)



Percentage of qualified students admitted: 45/90 = 50%

Percentage of unqualified students rejected: 8/10 = 80%

Total percentage of Lilliputian students admitted: (45+2)/100 = 47%

Table 4. Brobdingnagian applicants (10% are qualified):



Percentage of qualified students admitted: 5/10 = 50%

Percentage of unqualified students rejected: 72/90 = 80%

Total percentage of Brobdingnagian students admitted: (5+18)/100 = 23%

Equalized odds is satisfied because qualified Lilliputian and Brobdingnagian students both have a 50% chance of being admitted, and unqualified Lilliputian and Brobdingnagian have an 80% chance of being rejected.

Note: While equalized odds is satisfied here, demographic parity is not satisfied. Lilliputian and Brobdingnagian students are admitted to Glubbdubdrib University at different rates; 47% of Lilliputian students are admitted, and 23% of Brobdingnagian students are admitted.

Equalized odds is formally defined in "Equality of Opportunity in Supervised Learning" as follows: "predictor Ŷ satisfies equalized odds with respect to protected attribute A and outcome Y if Ŷ and A are independent, conditional on Y."

Note: Contrast equalized odds with the more relaxed equality of opportunity metric.

## experimenter's bias

* + See confirmation bias.

## fairness constraint

* + Applying a constraint to an algorithm to ensure one or more definitions of fairness are satisfied. Examples of fairness constraints include:
    - Post-processing your model's output.
    - Altering the loss function to incorporate a penalty for violating a fairness metric.
    - Directly adding a mathematical constraint to an optimization problem.

## fairness metric

* + A mathematical definition of “fairness” that is measurable. Some commonly used fairness metrics include:
    - equalized odds
    - predictive parity
    - counterfactual fairness
    - demographic parity

Many fairness metrics are mutually exclusive; see incompatibility of fairness metrics.

## false negative (FN)

* + An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam.

## false positive (FP)

* + An example in which the model mistakenly predicted the positive class. For example, the model inferred that a particular email message was spam (the positive class), but that email message was actually not spam.

## false positive rate (FPR)

* + The x-axis in an ROC curve. The false positive rate is defined as follows:

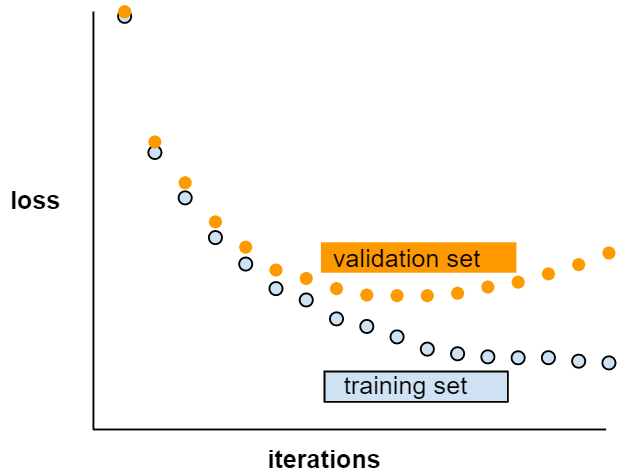


## Generalization

* + Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.

## generalization curve

* + A loss curve showing both the training set and the validation set. A generalization curve can help you detect possible overfitting. For example, the following generalization curve suggests overfitting because loss for the validation set ultimately becomes significantly higher than for the training set.



## gradient descent

* + A technique to minimize loss by computing the gradients of loss with respect to the model's parameters, conditioned on training data. Informally, gradient descent iteratively adjusts parameters, gradually finding the best combination of weights and bias to minimize loss.

## ground truth

* + The correct answer. Reality. Since reality is often subjective, expert raters typically are the proxy for ground truth.

## group attribution bias

* + Assuming that what is true for an individual is also true for everyone in that group. The effects of group attribution bias can be exacerbated if a convenience sampling is used for data collection. In a non-representative sample, attributions may be made that do not reflect reality.

See also out-group homogeneity bias and in-group bias.

## hinge loss

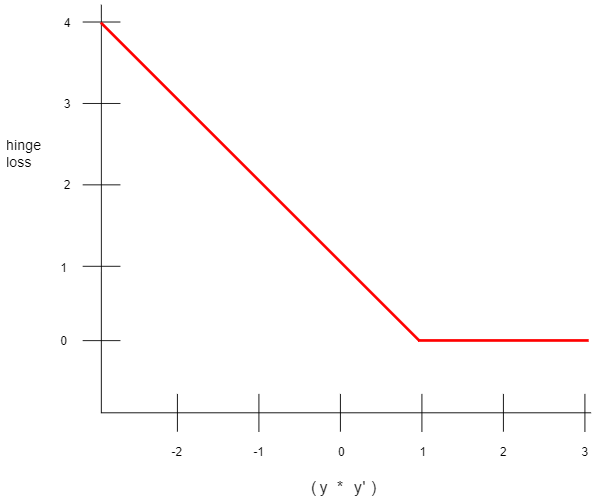
* + A family of loss functions for classification designed to find the decision boundary as distant as possible from each training example, thus maximizing the margin between examples and the boundary. KSVMs use hinge loss (or a related function, such as squared hinge loss). For binary classification, the hinge loss function is defined as follows:



where y is the true label, either -1 or +1, and y' is the raw output of the classifier model:



Consequently, a plot of hinge loss vs. (y \* y') looks as follows:



## Holdout data

* + Examples intentionally not used ("held out") during training. The validation dataset and test dataset are examples of holdout data. Holdout data helps evaluate your model's ability to generalize to data other than the data it was trained on. The loss on the holdout set provides a better estimate of the loss on an unseen dataset than does the loss on the training set.

## implicit bias

* + Automatically making an association or assumption based on one’s mental models and memories. Implicit bias can affect the following:
    - How data is collected and classified.
    - How machine learning systems are designed and developed.

For example, when building a classifier to identify wedding photos, an engineer may use the presence of a white dress in a photo as a feature. However, white dresses have been customary only during certain eras and in certain cultures.

## Inference

* + In machine learning, often refers to the process of making predictions by applying the trained model to unlabeled examples. In statistics, inference refers to the process of fitting the parameters of a distribution conditioned on some observed data. (See the Wikipedia article on statistical inference.)

## in-group bias

* + Showing partiality to one's own group or own characteristics. If testers or raters consist of the machine learning developer's friends, family, or colleagues, then in-group bias may invalidate product testing or the dataset.

In-group bias is a form of group attribution bias. See also out-group homogeneity bias.

## L2 loss

* + See squared loss.

## Log Loss

* + The logarithm of the odds of some event.
  + If the event refers to a binary probability, then odds refers to the ratio of the probability of success (p) to the probability of failure (1-p). For example, suppose that a given event has a 90% probability of success and a 10% probability of failure. In this case, odds is calculated as follows:



* + The log-odds is simply the logarithm of the odds. By convention, "logarithm" refers to natural logarithm, but logarithm could actually be any base greater than 1. Sticking to convention, the log-odds of our example is therefore:



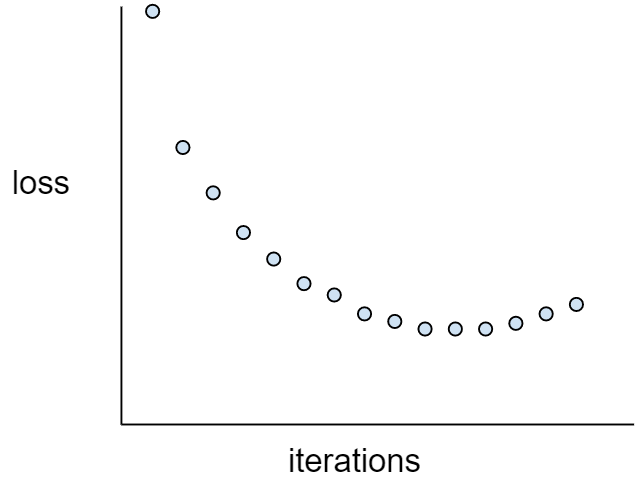
The log-odds are the inverse of the sigmoid function.

## Loss

* + A measure of how far a model's predictions are from its label. Or, to phrase it more pessimistically, a measure of how bad the model is. To determine this value, a model must define a loss function. For example, linear regression models typically use mean squared error for a loss function, while logistic regression models use Log Loss.

## loss curve

* + A graph of loss as a function of training iterations. For example:



The loss curve can help you determine when your model is converging, overfitting, or underfitting.

## loss surface

* + A graph of weight(s) vs. loss. Gradient descent aims to find the weight(s) for which the loss surface is at a local minimum.

## Mean Squared Error (MSE)

* + The average squared loss per example. MSE is calculated by dividing the squared loss by the number of examples. The values that TensorFlow Playground displays for "Training loss" and "Test loss" are MSE.

## Metric

* + A number that you care about. May or may not be directly optimized in a machine-learning system. A metric that your system tries to optimize is called an objective.

## Metrics API (tf.metrics)

* + A TensorFlow API for evaluating models. For example, tf.metrics.accuracy determines how often a model's predictions match labels. When writing a custom Estimator, you invoke Metrics API functions to specify how your model should be evaluated.

## mini-batch stochastic gradient descent

* + A gradient descent algorithm that uses mini-batches. In other words, mini-batch stochastic gradient descent estimates the gradient based on a small subset of the training data. Regular stochastic gradient descent uses a mini-batch of size 1.

## minimax loss

* + A loss function for generative adversarial networks, based on the cross-entropy between the distribution of generated data and real data.

Minimax loss is used in the first paper to describe generative adversarial networks.

## offline inference

* + Generating a group of predictions, storing those predictions, and then retrieving those predictions on demand. Contrast with online inference.

## Overfitting

* + Creating a model that matches the training data so closely that the model fails to make correct predictions on new data.

## participation bias

* + Synonym for non-response bias. See selection bias.

## Performance

* + Overloaded term with the following meanings:
    - The traditional meaning within software engineering. Namely: How fast (or efficiently) does this piece of software run?
    - The meaning within machine learning. Here, performance answers the following question: How correct is this model? That is, how good are the model's predictions?

## Perplexity

One measure of how well a model is accomplishing its task. For example, suppose your task is to read the first few letters of a word a user is typing on a smartphone keyboard, and to offer a list of possible completion words. Perplexity, P, for this task is approximately the number of guesses you need to offer in order for your list to contain the actual word the user is trying to type.

Perplexity is related to cross-entropy as follows:



## PR AUC (area under the PR curve)

* + Area under the interpolated precision-recall curve, obtained by plotting (recall, precision) points for different values of the classification threshold. Depending on how it's calculated, PR AUC may be equivalent to the average precision of the model.

## Precision

* + A metric for classification models. Precision identifies the frequency with which a model was correct when predicting the positive class. That is:



## precision-recall curve

* + A curve of precision vs. recall at different classification thresholds

## Recall

* + A metric for classification models that answers the following question: Out of all the possible positive labels, how many did the model correctly identify? That is:



## reporting bias

* + The fact that the frequency with which people write about actions, outcomes, or properties is not a reflection of their real-world frequencies or the degree to which a property is characteristic of a class of individuals. Reporting bias can influence the composition of data that machine learning systems learn from.

For example, in books, the word laughed is more prevalent than breathed. A machine learning model that estimates the relative frequency of laughing and breathing from a book corpus would probably determine that laughing is more common than breathing.

## Reward

* + In reinforcement learning, the numerical result of taking an action in a state, as defined by the environment.

## ROC (receiver operating characteristic) Curve

* + A curve of true positive rate vs. false positive rate at different classification thresholds. See also AUC.

## Root Mean Squared Error (RMSE)

* + The square root of the Mean Squared Error.

## sampling bias

* + See selection bias.

## Scoring

* + The part of a recommendation system that provides a value or ranking for each item produced by the candidate generation phase.

## selection bias

* + Errors in conclusions drawn from sampled data due to a selection process that generates systematic differences between samples observed in the data and those not observed. The following forms of selection bias exist:
    - coverage bias: The population represented in the dataset does not match the population that the machine learning model is making predictions about.
    - sampling bias: Data is not collected randomly from the target group.
    - non-response bias (also called participation bias): Users from certain groups opt-out of surveys at different rates than users from other groups.

For example, suppose you are creating a machine learning model that predicts people's enjoyment of a movie. To collect training data, you hand out a survey to everyone in the front row of a theater showing the movie. Offhand, this may sound like a reasonable way to gather a dataset; however, this form of data collection may introduce the following forms of selection bias:

* + - coverage bias: By sampling from a population who chose to see the movie, your model's predictions may not generalize to people who did not already express that level of interest in the movie.
    - sampling bias: Rather than randomly sampling from the intended population (all the people at the movie), you sampled only the people in the front row. It is possible that the people sitting in the front row were more interested in the movie than those in other rows.
    - non-response bias: In general, people with strong opinions tend to respond to optional surveys more frequently than people with mild opinions. Since the movie survey is optional, the responses are more likely to form a bimodal distribution than a normal (bell-shaped) distribution.

## similarity measure

* + In clustering algorithms, the metric used to determine how alike (how similar) any two examples are.

## size invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the size of the image changes. For example, the algorithm can still identify a cat whether it consumes 2M pixels or 200K pixels. Note that even the best image classification algorithms still have practical limits on size invariance. For example, an algorithm (or human) is unlikely to correctly classify a cat image consuming only 20 pixels.

See also translational invariance and rotational invariance.

## Sketching

* + In unsupervised machine learning, a category of algorithms that perform a preliminary similarity analysis on examples. Sketching algorithms use a locality-sensitive hash function to identify points that are likely to be similar, and then group them into buckets.

Sketching decreases the computation required for similarity calculations on large datasets. Instead of calculating similarity for every single pair of examples in the dataset, we calculate similarity only for each pair of points within each bucket.

## sparse feature

* + Feature vector whose values are predominately zero or empty. For example, a vector containing a single 1 value and a million 0 values is sparse. As another example, words in a search query could also be a sparse feature—there are many possible words in a given language, but only a few of them occur in a given query.

Contrast with dense feature.

## squared hinge loss

* + The square of the hinge loss. Squared hinge loss penalizes outliers more harshly than regular hinge loss.

## squared loss

* + The loss function used in linear regression. (Also known as L2 Loss.) This function calculates the squares of the difference between a model's predicted value for a labeled example and the actual value of the label. Due to squaring, this loss function amplifies the influence of bad predictions. That is, squared loss reacts more strongly to outliers than L1 loss.

## stochastic gradient descent (SGD)

* + A gradient descent algorithm in which the batch size is one. In other words, SGD relies on a single example chosen uniformly at random from a dataset to calculate an estimate of the gradient at each step.

## structural risk minimization (SRM)

* + An algorithm that balances two goals:
    - The desire to build the most predictive model (for example, lowest loss).
    - The desire to keep the model as simple as possible (for example, strong regularization).

For example, a function that minimizes loss+regularization on the training set is a structural risk minimization algorithm.

For more information, see http://www.svms.org/srm/.

Contrast with empirical risk minimization.

## tabular Q-learning

* + In reinforcement learning, implementing Q-learning by using a table to store the Q-functions for every combination of state and action.

## test set

* + The subset of the dataset that you use to test your model after the model has gone through initial vetting by the validation set.

Contrast with training set and validation set.

## true negative (TN)

* + An example in which the model correctly predicted the negative class. For example, the model inferred that a particular email message was not spam, and that email message really was not spam.

## true positive (TP)

* + An example in which the model correctly predicted the positive class. For example, the model inferred that a particular email message was spam, and that email message really was spam.

## true positive rate (TPR)

* + Synonym for recall. That is:



True positive rate is the y-axis in an ROC curve.

## Underfitting

* + Producing a model with poor predictive ability because the model hasn't captured the complexity of the training data. Many problems can cause underfitting, including:
    - Training on the wrong set of features.
    - Training for too few epochs or at too low a learning rate.
    - Training with too high a regularization rate.
    - Providing too few hidden layers in a deep neural network.

## user matrix

* + In recommendation systems, an embedding generated by matrix factorization that holds latent signals about user preferences. Each row of the user matrix holds information about the relative strength of various latent signals for a single user. For example, consider a movie recommendation system. In this system, the latent signals in the user matrix might represent each user's interest in particular genres, or might be harder-to-interpret signals that involve complex interactions across multiple factors.

The user matrix has a column for each latent feature and a row for each user. That is, the user matrix has the same number of rows as the target matrix that is being factorized. For example, given a movie recommendation system for 1,000,000 users, the user matrix will have 1,000,000 rows.

## Validation

* + A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model’s performance generalizes beyond the training set.

Contrast with test set.

## validation set

* + A subset of the dataset—disjoint from the training set—used in validation.

Contrast with training set and test set.

# Clustering

## agglomerative clustering

* + See hierarchical clustering.

## Centroid

* + The center of a cluster as determined by a k-means or k-median algorithm. For instance, if k is 3, then the k-means or k-median algorithm finds 3 centroids.

## centroid-based clustering

* + A category of clustering algorithms that organizes data into nonhierarchical clusters. k-means is the most widely used centroid-based clustering algorithm.

Contrast with hierarchical clustering algorithms.

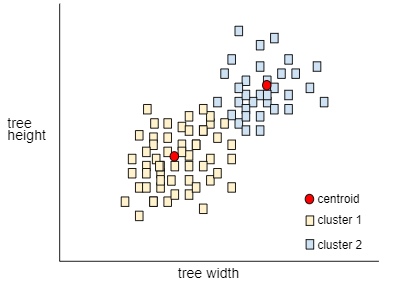
## Class

* + One of a set of enumerated target values for a label. For example, in a binary classification model that detects spam, the two classes are spam and not spam. In a multi-class classification model that identifies dog breeds, the classes would be poodle, beagle, pug, and so on.

## Clustering

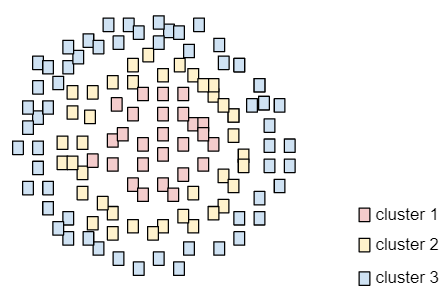
* + Grouping related examples, particularly during unsupervised learning. Once all the examples are grouped, a human can optionally supply meaning to each cluster.

Many clustering algorithms exist. For example, the k-means algorithm clusters examples based on their proximity to a centroid, as in the following diagram:



A human researcher could then review the clusters and, for example, label cluster 1 as "dwarf trees" and cluster 2 as "full-size trees."

As another example, consider a clustering algorithm based on an example's distance from a center point, illustrated as follows:



## hierarchical clustering

* + A category of clustering algorithms that create a tree of clusters. Hierarchical clustering is well-suited to hierarchical data, such as botanical taxonomies. There are two types of hierarchical clustering algorithms:
* Agglomerative clustering first assigns every example to its own cluster, and iteratively merges the closest clusters to create a hierarchical tree.
* Divisive clustering first groups all examples into one cluster and then iteratively divides the cluster into a hierarchical tree.

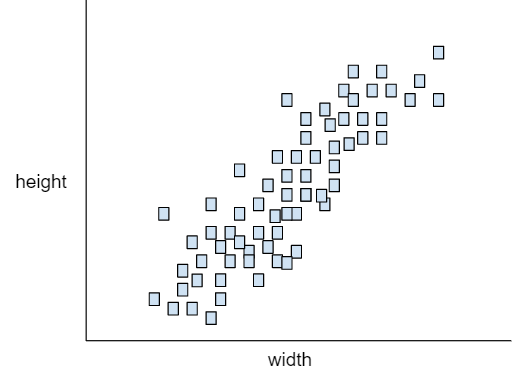
Contrast with centroid-based clustering.

## k-means

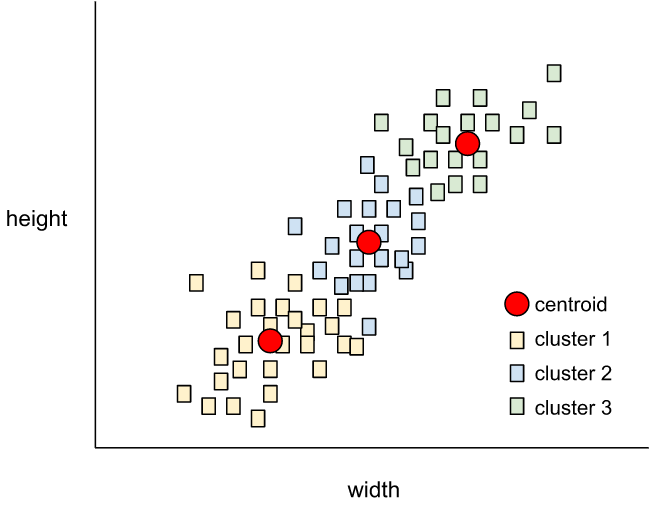
* + A popular clustering algorithm that groups examples in unsupervised learning. The k-means algorithm basically does the following:
    - Iteratively determines the best k center points (known as centroids).
    - Assigns each example to the closest centroid. Those examples nearest the same centroid belong to the same group.

The k-means algorithm picks centroid locations to minimize the cumulative square of the distances from each example to its closest centroid.

For example, consider the following plot of dog height to dog width:



If k=3, the k-means algorithm will determine three centroids. Each example is assigned to its closest centroid, yielding three groups:



Imagine that a manufacturer wants to determine the ideal sizes for small, medium, and large sweaters for dogs. The three centroids identify the mean height and mean width of each dog in that cluster. So, the manufacturer should probably base sweater sizes on those three centroids. Note that the centroid of a cluster is typically not an example in the cluster.

The preceding illustrations shows k-means for examples with only two features (height and width). Note that k-means can group examples across many features.

## k-median

* + A clustering algorithm closely related to k-means. The practical difference between the two is as follows:
    - In k-means, centroids are determined by minimizing the sum of the squares of the distance between a centroid candidate and each of its examples.
    - In k-median, centroids are determined by minimizing the sum of the distance between a centroid candidate and each of its examples.

Note that the definitions of distance are also different:

* + - k-means relies on the Euclidean distance from the centroid to an example. (In two dimensions, the Euclidean distance means using the Pythagorean theorem to calculate the hypotenuse.) For example, the k-means distance between (2,2) and (5,-2) would be:



* + - k-median relies on the Manhattan distance from the centroid to an example. This distance is the sum of the absolute deltas in each dimension. For example, the k-median distance between (2,2) and (5,-2) would be:



## out-group homogeneity bias

* + The tendency to see out-group members as more alike than in-group members when comparing attitudes, values, personality traits, and other characteristics. In-group refers to people you interact with regularly; out-group refers to people you do not interact with regularly. If you create a dataset by asking people to provide attributes about out-groups, those attributes may be less nuanced and more stereotyped than attributes that participants list for people in their in-group.

For example, Lilliputians might describe the houses of other Lilliputians in great detail, citing small differences in architectural styles, windows, doors, and sizes. However, the same Lilliputians might simply declare that Brobdingnagians all live in identical houses.

Out-group homogeneity bias is a form of group attribution bias.

See also in-group bias.

## Outliers

* + Values distant from most other values. In machine learning, any of the following are outliers:
    - Weights with high absolute values.
    - Predicted values relatively far away from the actual values.
    - Input data whose values are more than roughly 3 standard deviations from the mean.

Outliers often cause problems in model training. Clipping is one way of managing outliers.

## selection bias

* + Errors in conclusions drawn from sampled data due to a selection process that generates systematic differences between samples observed in the data and those not observed. The following forms of selection bias exist:
    - coverage bias: The population represented in the dataset does not match the population that the machine learning model is making predictions about.
    - sampling bias: Data is not collected randomly from the target group.
    - non-response bias (also called participation bias): Users from certain groups opt-out of surveys at different rates than users from other groups.

For example, suppose you are creating a machine learning model that predicts people's enjoyment of a movie. To collect training data, you hand out a survey to everyone in the front row of a theater showing the movie. Offhand, this may sound like a reasonable way to gather a dataset; however, this form of data collection may introduce the following forms of selection bias:

* + - coverage bias: By sampling from a population who chose to see the movie, your model's predictions may not generalize to people who did not already express that level of interest in the movie.
    - sampling bias: Rather than randomly sampling from the intended population (all the people at the movie), you sampled only the people in the front row. It is possible that the people sitting in the front row were more interested in the movie than those in other rows.
    - non-response bias: In general, people with strong opinions tend to respond to optional surveys more frequently than people with mild opinions. Since the movie survey is optional, the responses are more likely to form a bimodal distribution than a normal (bell-shaped) distribution.

## similarity measure

* + In clustering algorithms, the metric used to determine how alike (how similar) any two examples are.

## Subsampling

* + See pooling

## time series analysis

* + A subfield of machine learning and statistics that analyzes temporal data. Many types of machine learning problems require time series analysis, including classification, clustering, forecasting, and anomaly detection. For example, you could use time series analysis to forecast the future sales of winter coats by month based on historical sales data.

## unsupervised machine learning

* + Training a model to find patterns in a dataset, typically an unlabeled dataset.

The most common use of unsupervised machine learning is to cluster data into groups of similar examples. For example, an unsupervised machine learning algorithm can cluster songs together based on various properties of the music. The resulting clusters can become an input to other machine learning algorithms (for example, to a music recommendation service). Clustering can be helpful in domains where true labels are hard to obtain. For example, in domains such as anti-abuse and fraud, clusters can help humans better understand the data.

Another example of unsupervised machine learning is principal component analysis (PCA). For example, applying PCA on a dataset containing the contents of millions of shopping carts might reveal that shopping carts containing lemons frequently also contain antacids.

* + Compare with supervised machine learning.

# Neural Networks

## accuracy

* + The fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:



In binary classification, accuracy has the following definition:



## activation function

* + A function (for example, ReLU or sigmoid) that takes in the weighted sum of all of the inputs from the previous layer and then generates and passes an output value (typically nonlinear) to the next layer.

## artificial intelligence

* + A non-human program or model that can solve sophisticated tasks. For example, a program or model that translates text or a program or model that identifies diseases from radiologic images both exhibit artificial intelligence.

Formally, machine learning is a sub-field of artificial intelligence. However, in recent years, some organizations have begun using the terms artificial intelligence and machine learning interchangeably.

## Backpropagation

* + The primary algorithm for performing gradient descent on neural networks. First, the output values of each node are calculated (and cached) in a forward pass. Then, the partial derivative of the error with respect to each parameter is calculated in a backward pass through the graph.

## Batch

* + The set of examples used in one iteration (that is, one gradient update) of model training.

See also batch size.

## batch normalization

* + Normalizing the input or output of the activation functions in a hidden layer. Batch normalization can provide the following benefits:
    - Make neural networks more stable by protecting against outlier weights.
    - Enable higher learning rates.
    - Reduce overfitting.

## batch size

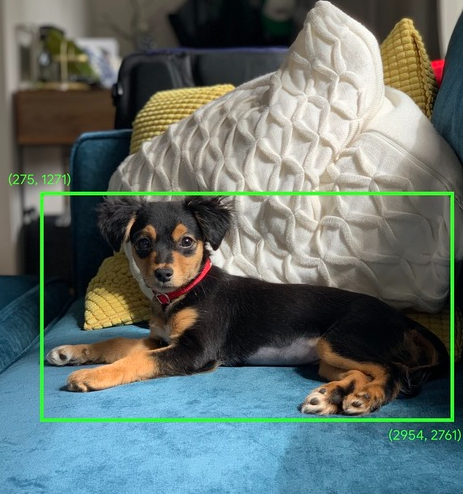
* + The number of examples in a batch. For example, the batch size of SGD is 1, while the batch size of a mini-batch is usually between 10 and 1000. Batch size is usually fixed during training and inference; however, TensorFlow does permit dynamic batch sizes.

## Bayesian neural network

* + A probabilistic neural network that accounts for uncertainty in weights and outputs. A standard neural network regression model typically predicts a scalar value; for example, a model predicts a house price of 853,000. By contrast, a Bayesian neural network predicts a distribution of values; for example, a model predicts a house price of 853,000 with a standard deviation of 67,200. A Bayesian neural network relies on Bayes' Theorem to calculate uncertainties in weights and predictions. A Bayesian neural network can be useful when it is important to quantify uncertainty, such as in models related to pharmaceuticals. Bayesian neural networks can also help prevent overfitting.

## bounding box

* + In an image, the (x, y) coordinates of a rectangle around an area of interest, such as the dog in the image below.



## Class

* + One of a set of enumerated target values for a label. For example, in a binary classification model that detects spam, the two classes are spam and not spam. In a multi-class classification model that identifies dog breeds, the classes would be poodle, beagle, pug, and so on.

## Convergence

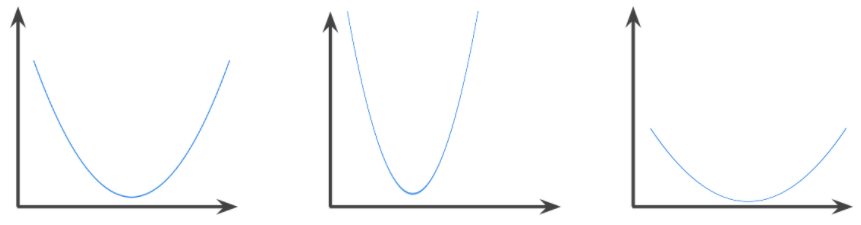
* + Informally, often refers to a state reached during training in which training loss and validation loss change very little or not at all with each iteration after a certain number of iterations. In other words, a model reaches convergence when additional training on the current data will not improve the model. In deep learning, loss values sometimes stay constant or nearly so for many iterations before finally descending, temporarily producing a false sense of convergence.

See also early stopping.

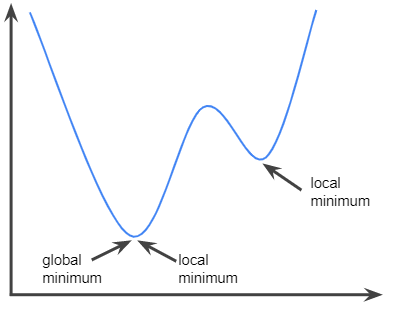
See also Boyd and Vandenberghe, Convex Optimization.

## convex function

* + A function in which the region above the graph of the function is a convex set. The prototypical convex function is shaped something like the letter U. For example, the following are all convex functions:



By contrast, the following function is not convex. Notice how the region above the graph is not a convex set:



A strictly convex function has exactly one local minimum point, which is also the global minimum point. The classic U-shaped functions are strictly convex functions. However, some convex functions (for example, straight lines) are not U-shaped.

A lot of the common loss functions, including the following, are convex functions:

* + - L2 loss
    - Log Loss
    - L1 regularization
    - L2 regularization

Many variations of gradient descent are guaranteed to find a point close to the minimum of a strictly convex function. Similarly, many variations of stochastic gradient descent have a high probability (though, not a guarantee) of finding a point close to the minimum of a strictly convex function.

The sum of two convex functions (for example, L2 loss + L1 regularization) is a convex function.

Deep models are never convex functions. Remarkably, algorithms designed for convex optimization tend to find reasonably good solutions on deep networks anyway, even though those solutions are not guaranteed to be a global minimum.

## convex optimization

* + The process of using mathematical techniques such as gradient descent to find the minimum of a convex function. A great deal of research in machine learning has focused on formulating various problems as convex optimization problems and in solving those problems more efficiently.

For complete details, see Boyd and Vandenberghe, Convex Optimization.

## convex set

* + A subset of Euclidean space such that a line drawn between any two points in the subset remains completely within the subset. For instance, the following two shapes are convex sets:

A rectangle
and a semi-ellipse are both convex sets.

By contrast, the following two shapes are not convex sets:

A pie-chart
with a missing slice and a firework are both nonconvex sets.

## Convolution

* + In mathematics, casually speaking, a mixture of two functions. In machine learning, a convolution mixes the convolutional filter and the input matrix in order to train weights.

The term "convolution" in machine learning is often a shorthand way of referring to either convolutional operation or convolutional layer.

Without convolutions, a machine learning algorithm would have to learn a separate weight for every cell in a large tensor. For example, a machine learning algorithm training on 2K x 2K images would be forced to find 4M separate weights. Thanks to convolutions, a machine learning algorithm only has to find weights for every cell in the convolutional filter, dramatically reducing the memory needed to train the model. When the convolutional filter is applied, it is simply replicated across cells such that each is multiplied by the filter.

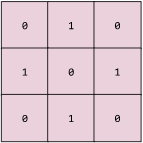
## convolutional filter

* + One of the two actors in a convolutional operation. (The other actor is a slice of an input matrix.) A convolutional filter is a matrix having the same rank as the input matrix, but a smaller shape. For example, given a 28x28 input matrix, the filter could be any 2D matrix smaller than 28x28.

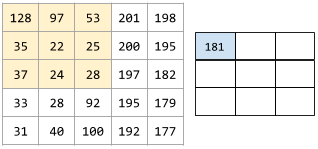
In photographic manipulation, all the cells in a convolutional filter are typically set to a constant pattern of ones and zeroes. In machine learning, convolutional filters are typically seeded with random numbers and then the network trains the ideal values.

## convolutional layer

* + A layer of a deep neural network in which a convolutional filter passes along an input matrix. For example, consider the following 3x3 convolutional filter:



The following animation shows a convolutional layer consisting of 9 convolutional operations involving the 5x5 input matrix. Notice that each convolutional operation works on a different 3x3 slice of the input matrix. The resulting 3x3 matrix (on the right) consists of the results of the 9 convolutional operations:



## convolutional neural network

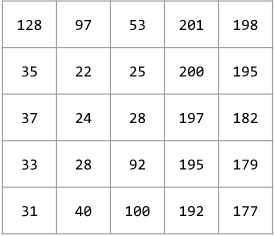
* + A neural network in which at least one layer is a convolutional layer. A typical convolutional neural network consists of some combination of the following layers:
    - convolutional layers
    - pooling layers
    - dense layers

Convolutional neural networks have had great success in certain kinds of problems, such as image recognition.

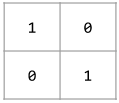
## convolutional operation

* + The following two-step mathematical operation:
    - Element-wise multiplication of the convolutional filter and a slice of an input matrix. (The slice of the input matrix has the same rank and size as the convolutional filter.)
    - Summation of all the values in the resulting product matrix.

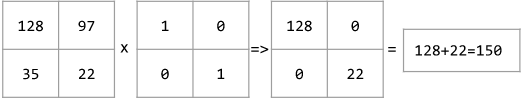
For example, consider the following 5x5 input matrix:



Now imagine the following 2x2 convolutional filter:



Each convolutional operation involves a single 2x2 slice of the input matrix. For instance, suppose we use the 2x2 slice at the top-left of the input matrix. So, the convolution operation on this slice looks as follows:



A convolutional layer consists of a series of convolutional operations, each acting on a different slice of the input matrix.

## Cost

* + Synonym for loss.

## coverage bias

* + See selection bias.

## cross-entropy

* + A generalization of Log Loss to multi-class classification problems. Cross-entropy quantifies the difference between two probability distributions. See also perplexity.

## deep model

* + A type of neural network containing multiple hidden layers.

Contrast with wide model.

## deep neural network

* + Synonym for deep model.

## Deep Q-Network (DQN)

* + In Q-learning, a deep neural network that predicts Q-functions.

Critic is a synonym for Deep Q-Network.

## dense feature

* + A feature in which most values are non-zero, typically a Tensor of floating-point values. Contrast with sparse feature.

## dense layer

* + Synonym for fully connected layer.

## Depth

* + The number of layers (including any embedding layers) in a neural network that learn weights. For example, a neural network with 5 hidden layers and 1 output layer has a depth of 6.

## DQN

* + Abbreviation for Deep Q-Network.

dropout regularization

* + A form of regularization useful in training neural networks. Dropout regularization works by removing a random selection of a fixed number of the units in a network layer for a single gradient step. The more units dropped out, the stronger the regularization. This is analogous to training the network to emulate an exponentially large ensemble of smaller networks. For full details, see Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

## dynamic model

* + A model that is trained online in a continuously updating fashion. That is, data is continuously entering the model.

## Epoch

* + A full training pass over the entire dataset such that each example has been seen once. Thus, an epoch represents N / batch size training iterations, where N is the total number of examples.

## exploding gradient problem

* + The tendency for gradients in a deep neural networks (especially recurrent neural networks) to become surprisingly steep (high). Steep gradients result in very large updates to the weights of each node in a deep neural network.

Models suffering from the exploding gradient problem become difficult or impossible to train. Gradient clipping can mitigate this problem.

Compare to vanishing gradient problem.

## feedback loop

* + In machine learning, a situation in which a model's predictions influence the training data for the same model or another model. For example, a model that recommends movies will influence the movies that people see, which will then influence subsequent movie recommendation models.

## feedforward neural network (FFN)

* + A neural network without cyclic or recursive connections. For example, traditional deep neural networks are feedforward neural networks. Contrast with recurrent neural networks, which are cyclic.

## fine tuning

* + Perform a secondary optimization to adjust the parameters of an already trained model to fit a new problem. Fine tuning often refers to refitting the weights of a trained unsupervised model to a supervised model.

## forget gate

* + The portion of a Long Short-Term Memory cell that regulates the flow of information through the cell. Forget gates maintain context by deciding which information to discard from the cell state.

## full softmax

* + See softmax. Contrast with candidate sampling.

## fully connected layer

* + A hidden layer in which each node is connected to every node in the subsequent hidden layer.

A fully connected layer is also known as a dense layer.

## GAN

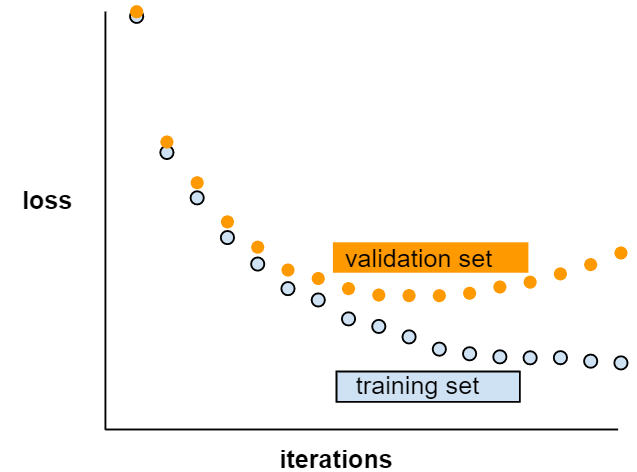
* + Abbreviation for generative adversarial network.

## Generalization

* + Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.

## generalization curve

* + A loss curve showing both the training set and the validation set. A generalization curve can help you detect possible overfitting. For example, the following generalization curve suggests overfitting because loss for the validation set ultimately becomes significantly higher than for the training set.



## generative adversarial network (GAN)

* + A system to create new data in which a generator creates data and a discriminator determines whether that created data is valid or invalid.

## generative model

* + Practically speaking, a model that does either of the following:
    - Creates (generates) new examples from the training dataset. For example, a generative model could create poetry after training on a dataset of poems. The generator part of a generative adversarial network falls into this category.
    - Determines the probability that a new example comes from the training set, or was created from the same mechanism that created the training set. For example, after training on a dataset consisting of English sentences, a generative model could determine the probability that new input is a valid English sentence.

A generative model can theoretically discern the distribution of examples or particular features in a dataset. That is:

p(examples)

Unsupervised learning models are generative.

Contrast with discriminative models.

## Gradient

* + The vector of partial derivatives with respect to all of the independent variables. In machine learning, the gradient is the vector of partial derivatives of the model function. The gradient points in the direction of steepest ascent.

## gradient clipping

* + A commonly used mechanism to mitigate the exploding gradient problem by artificially limiting (clipping) the maximum value of gradients when using gradient descent to train a model.

## gradient descent

* + A technique to minimize loss by computing the gradients of loss with respect to the model's parameters, conditioned on training data. Informally, gradient descent iteratively adjusts parameters, gradually finding the best combination of weights and bias to minimize loss.

## ground truth

* + The correct answer. Reality. Since reality is often subjective, expert raters typically are the proxy for ground truth.

## group attribution bias

* + Assuming that what is true for an individual is also true for everyone in that group. The effects of group attribution bias can be exacerbated if a convenience sampling is used for data collection. In a non-representative sample, attributions may be made that do not reflect reality.

See also out-group homogeneity bias and in-group bias.

## hidden layer

* + A synthetic layer in a neural network between the input layer (that is, the features) and the output layer (the prediction). Hidden layers typically contain an activation function (such as ReLU) for training. A deep neural network contains more than one hidden layer.

## holdout data

* + Examples intentionally not used ("held out") during training. The validation dataset and test dataset are examples of holdout data. Holdout data helps evaluate your model's ability to generalize to data other than the data it was trained on. The loss on the holdout set provides a better estimate of the loss on an unseen dataset than does the loss on the training set.

## Hyperparameter

* + The "knobs" that you tweak during successive runs of training a model. For example, learning rate is a hyperparameter.

Contrast with parameter.

## Hyperplane

* + A boundary that separates a space into two subspaces. For example, a line is a hyperplane in two dimensions and a plane is a hyperplane in three dimensions. More typically in machine learning, a hyperplane is the boundary separating a high-dimensional space. Kernel Support Vector Machines use hyperplanes to separate positive classes from negative classes, often in a very high-dimensional space.

## in-group bias

* + Showing partiality to one's own group or own characteristics. If testers or raters consist of the machine learning developer's friends, family, or colleagues, then in-group bias may invalidate product testing or the dataset.

In-group bias is a form of group attribution bias. See also out-group homogeneity bias.

## input function

* + In TensorFlow, a function that returns input data to the training, evaluation, or prediction method of an Estimator. For example, the training input function returns a batch of features and labels from the training set.

## input layer

* + The first layer (the one that receives the input data) in a neural network.

## Iteration

* + A single update of a model's weights during training. An iteration consists of computing the gradients of the parameters with respect to the loss on a single batch of data.

## Keras

* + A popular Python machine learning API. Keras runs on several deep learning frameworks, including TensorFlow, where it is made available as tf.keras.

## Keypoints

* + The coordinates of particular features in an image. For example, for an image recognition model that distinguishes flower species, keypoints might be the center of each petal, the stem, the stamen, and so on.

## Layer

* + A set of neurons in a neural network that process a set of input features, or the output of those neurons.

Also, an abstraction in TensorFlow. Layers are Python functions that take Tensors and configuration options as input and produce other tensors as output. Once the necessary Tensors have been composed, the user can convert the result into an Estimator via a model function.

## learning rate

* + A scalar used to train a model via gradient descent. During each iteration, the gradient descent algorithm multiplies the learning rate by the gradient. The resulting product is called the gradient step.

Learning rate is a key hyperparameter.

## linear model

* + A model that assigns one weight per feature to make predictions. (Linear models also incorporate a bias.) By contrast, the relationship of weights to features in deep models is not one-to-one.

A linear model uses the following formula:



where:

* is the raw prediction. (In certain kinds of linear models, this raw prediction will be further modified. For example, see logistic regression.)
* is the bias.
* is a weight, so is the weight of the first feature, is the weight of the second feature, and so on.
* is a feature, so is the value of the first feature, is the value of the second feature, and so on.

For example, suppose a linear model for three features learns the following bias and weights:

* + - = 7
    - = -2.5
    - = -1.2
    - = 1.4

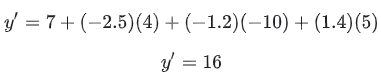
Therefore, given three features , the linear model uses the following equation to generate each prediction:



Suppose a particular example contains the following values:

* = 4
* = -10
* = 5

Plugging those values into the formula yields a prediction for this example:



Linear models tend to be easier to analyze and train than deep models. However, deep models can model complex relationships between features.

Linear regression and logistic regression are two types of linear models. Linear models include not only models that use the linear equation but also a broader set of models that use the linear equation as part of the formula. For example, logistic regression post-processes the raw prediction to calculate the prediction.

## Log Loss

* + The loss function used in binary logistic regression.

## Long Short-Term Memory (LSTM)

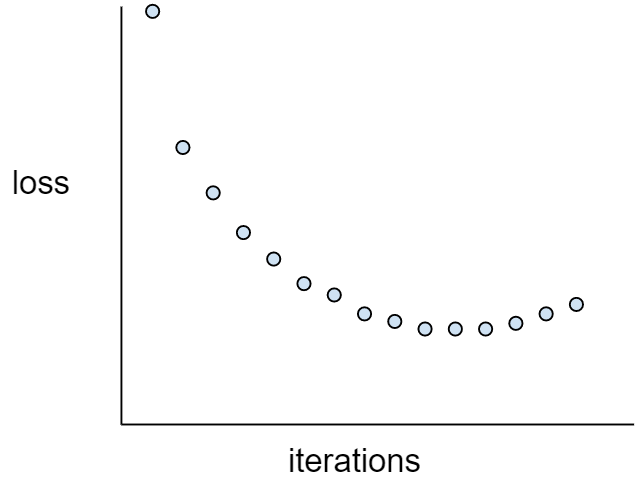
* + A type of cell in a recurrent neural network used to process sequences of data in applications such as handwriting recognition, machine translation, and image captioning. LSTMs address the vanishing gradient problem that occurs when training RNNs due to long data sequences by maintaining history in an internal memory state based on new input and context from previous cells in the RNN.

## Loss

* + A measure of how far a model's predictions are from its label. Or, to phrase it more pessimistically, a measure of how bad the model is. To determine this value, a model must define a loss function. For example, linear regression models typically use mean squared error for a loss function, while logistic regression models use Log Loss.

## loss curve

* + A graph of loss as a function of training iterations. For example:



The loss curve can help you determine when your model is converging, overfitting, or underfitting.

## loss surface

* + A graph of weight(s) vs. loss. Gradient descent aims to find the weight(s) for which the loss surface is at a local minimum.

## LSTM

* + Abbreviation for Long Short-Term Memory.

## mini-batch

* + A small, randomly selected subset of the entire batch of examples run together in a single iteration of training or inference. The batch size of a mini-batch is usually between 10 and 1,000. It is much more efficient to calculate the loss on a mini-batch than on the full training data.

## mini-batch stochastic gradient descent

* + A gradient descent algorithm that uses mini-batches. In other words, mini-batch stochastic gradient descent estimates the gradient based on a small subset of the training data. Regular stochastic gradient descent uses a mini-batch of size 1.

## minimax loss

* + A loss function for generative adversarial networks, based on the cross-entropy between the distribution of generated data and real data.

Minimax loss is used in the first paper to describe generative adversarial networks.

## Momentum

* + A sophisticated gradient descent algorithm in which a learning step depends not only on the derivative in the current step, but also on the derivatives of the step(s) that immediately preceded it. Momentum involves computing an exponentially weighted moving average of the gradients over time, analogous to momentum in physics. Momentum sometimes prevents learning from getting stuck in local minima.

## neural network

* + A model that, taking inspiration from the brain, is composed of layers (at least one of which is hidden) consisting of simple connected units or neurons followed by nonlinearities.

## Neuron

* + A node in a neural network, typically taking in multiple input values and generating one output value. The neuron calculates the output value by applying an activation function (nonlinear transformation) to a weighted sum of input values.

## node (neural network)

* + A neuron in a hidden layer.

## Noise

* + Broadly speaking, anything that obscures the signal in a dataset. Noise can be introduced into data in a variety of ways. For example:
    - Human raters make mistakes in labeling.
    - Humans and instruments mis-record or omit feature values.

## Operation (op)

* + A node in the TensorFlow graph. In TensorFlow, any procedure that creates, manipulates, or destroys a Tensor is an operation. For example, a matrix multiply is an operation that takes two Tensors as input and generates one Tensor as output.

## Optimizer

* + A specific implementation of the gradient descent algorithm. TensorFlow's base class for optimizers is tf.train.Optimizer. Popular optimizers include:
    - AdaGrad, which stands for ADAptive GRADient descent.
    - Adam, which stands for ADAptive with Momentum.

Different optimizers may leverage one or more of the following concepts to enhance the effectiveness of gradient descent on a given training set:

* momentum (Momentum)
* update frequency
* sparsity/regularization (Ftrl)
* more complex math (Proximal, and others)

You might even imagine an NN-driven optimizer.

## output layer

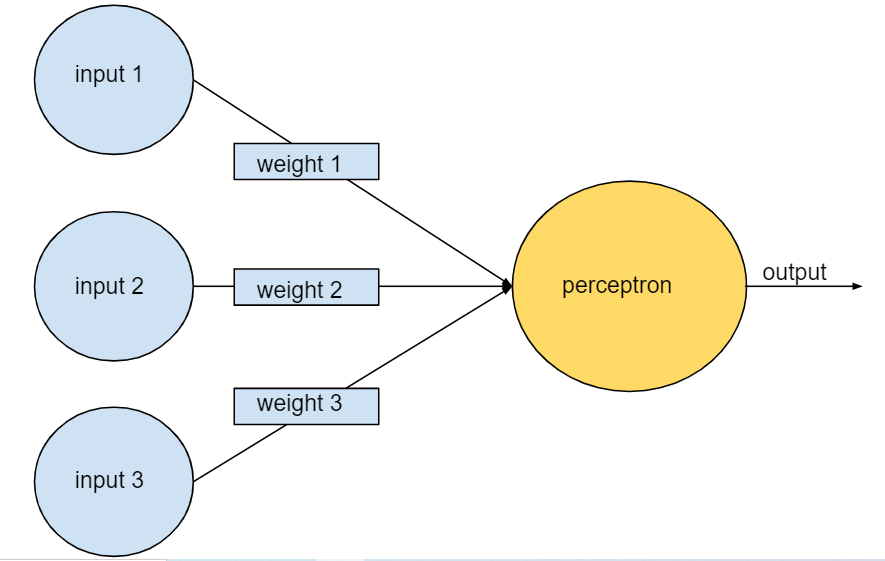
* + The "final" layer of a neural network. The layer containing the answer(s).

## Perceptron

* + A system (either hardware or software) that takes in one or more input values, runs a function on the weighted sum of the inputs, and computes a single output value. In machine learning, the function is typically nonlinear, such as ReLU, sigmoid, or tanh. For example, the following perceptron relies on the sigmoid function to process three input values:



In the following illustration, the perceptron takes three inputs, each of which is itself modified by a weight before entering the perceptron:



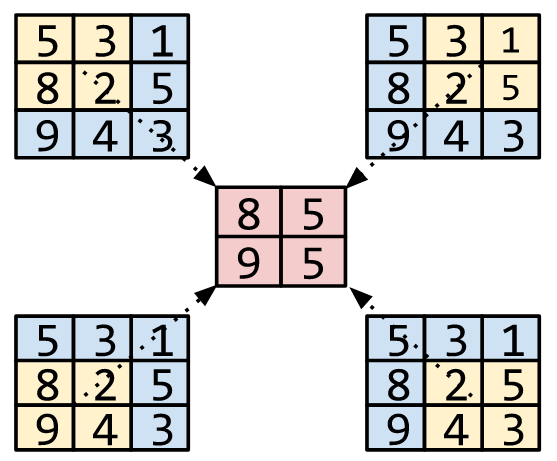
Perceptrons are the (nodes) in deep neural networks. That is, a deep neural network consists of multiple connected perceptrons, plus a backpropagation algorithm to introduce feedback.

## pooling

* + Reducing a matrix (or matrices) created by an earlier convolutional layer to a smaller matrix. Pooling usually involves taking either the maximum or average value across the pooled area. For example, suppose we have the following 3x3 matrix:



A pooling operation, just like a convolutional operation, divides that matrix into slices and then slides that convolutional operation by strides. For example, suppose the pooling operation divides the convolutional matrix into 2x2 slices with a 1x1 stride. As the following diagram illustrates, four pooling operations take place. Imagine that each pooling operation picks the maximum value of the four in that slice:



Pooling helps enforce translational invariance in the input matrix.

Pooling for vision applications is known more formally as spatial pooling. Time-series applications usually refer to pooling as temporal pooling. Less formally, pooling is often called subsampling or downsampling.

## post-processing

* + Processing the output of a model after the model has been run. Post-processing can be used to enforce fairness constraints without modifying models themselves.

For example, one might apply post-processing to a binary classifier by setting a classification threshold such that equality of opportunity is maintained for some attribute by checking that the true positive rate is the same for all values of that attribute.

## Regularization

* + The penalty on a model's complexity. Regularization helps prevent overfitting. Different kinds of regularization include:
    - [**L1**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L1_regularization) regularization
    - [**L2**](https://developers.google.com/machine-learning/glossary/?fbclid=IwAR3aySRnZN0QDfVoNPHWjRgSt3biIkoo0Ref3WTzqeShQCOP0COkGRHSMiE#L2_regularization) regularization
    - dropout regularization
    - early stopping (this is not a formal regularization method, but can effectively limit overfitting)

## regularization rate

* + A scalar value, represented as lambda, specifying the relative importance of the regularization function. The following simplified loss equation shows the regularization rate's influence:



Raising the regularization rate reduces overfitting but may make the model less accurate.

## RNN

* + Abbreviation for recurrent neural networks.

## sequence model

* + A model whose inputs have a sequential dependence. For example, predicting the next video watched from a sequence of previously watched videos.

## Serving

* + A synonym for inferring.

## sigmoid function

* + A function that maps logistic or multinomial regression output (log odds) to probabilities, returning a value between 0 and 1. The sigmoid function has the following formula:



where *σ* in logistic regression problems is simply:



In other words, the sigmoid function converts into a probability between 0 and 1.

In some neural networks, the sigmoid function acts as the activation function.

## similarity measure

* + In clustering algorithms, the metric used to determine how alike (how similar) any two examples are.

## Softmax

* + A function that provides probabilities for each possible class in a multi-class classification model. The probabilities add up to exactly 1.0. For example, softmax might determine that the probability of a particular image being a dog at 0.9, a cat at 0.08, and a horse at 0.02. (Also called full softmax.)

Contrast with candidate sampling.

## sparse feature

* + Feature vector whose values are predominately zero or empty. For example, a vector containing a single 1 value and a million 0 values is sparse. As another example, words in a search query could also be a sparse feature—there are many possible words in a given language, but only a few of them occur in a given query.

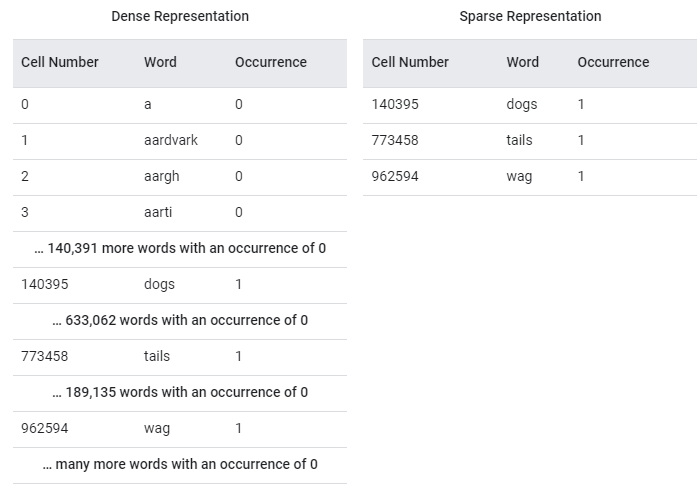
Contrast with dense feature.

## sparse representation

* + A representation of a tensor that only stores nonzero elements.

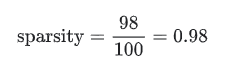
For example, the English language consists of about a million words. Consider two ways to represent a count of the words used in one English sentence:

* + - A dense representation of this sentence must set an integer for all one million cells, placing a 0 in most of them, and a low integer into a few of them.
    - A sparse representation of this sentence stores only those cells symbolizing a word actually in the sentence. So, if the sentence contained only 20 unique words, then the sparse representation for the sentence would store an integer in only 20 cells.
  + For example, consider two ways to represent the sentence, "Dogs wag tails." As the following tables show, the dense representation consumes about a million cells; the sparse representation consumes only 3 cells:



## Sparsity

* + The number of elements set to zero (or null) in a vector or matrix divided by the total number of entries in that vector or matrix. For example, consider a 10x10 matrix in which 98 cells contain zero. The calculation of sparsity is as follows:



Feature sparsity refers to the sparsity of a feature vector; model sparsity refers to the sparsity of the model weights.

## spatial pooling

* + See pooling

## Step

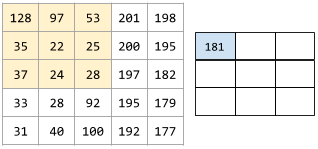
* + A forward and backward evaluation of one batch.

## step size

* + Synonym for learning rate.

## Stride

* + In a convolutional operation or pooling, the delta in each dimension of the next series of input slices. For example, the following animation demonstrates a (1,1) stride during a convolutional operation. Therefore, the next input slice starts one position to the right of the previous input slice. When the operation reaches the right edge, the next slice is all the way over to the left but one position down.



## Subsampling

* + See pooling

## target network

* + In Deep Q-learning, a neural network that is a stable approximation of the main neural network, where the main neural network implements either a Q-function or a policy. Then, you can train the main network on the Q-values predicted by the target network. Therefore, you prevent the feedback loop that occurs when the main network trains on Q-values predicted by itself. By avoiding this feedback, training stability increases.

## Tensor

* + The primary data structure in TensorFlow programs. Tensors are N-dimensional (where N could be very large) data structures, most commonly scalars, vectors, or matrices. The elements of a Tensor can hold integer, floating-point, or string values.

## TensorBoard

* + The dashboard that displays the summaries saved during the execution of one or more TensorFlow programs.

## TensorFlow

* + A large-scale, distributed, machine learning platform. The term also refers to the base API layer in the TensorFlow stack, which supports general computation on dataflow graphs.

Although TensorFlow is primarily used for machine learning, you may also use TensorFlow for non-ML tasks that require numerical computation using dataflow graphs.

## TensorFlow Playground

* + A program that visualizes how different hyperparameters influence model (primarily neural network) training. Go to http://playground.tensorflow.org to experiment with TensorFlow Playground.

## TensorFlow Serving

* + A platform to deploy trained models in production.

## Tensor Processing Unit (TPU)

* + An application-specific integrated circuit (ASIC) that optimizes the performance of machine learning workloads. These ASICs are deployed as multiple TPU chips on a TPU device.

## Tensor rank

* + See rank (Tensor).

## Tensor shape

* + The number of elements a Tensor contains in various dimensions. For example, a [5, 10] Tensor has a shape of 5 in one dimension and 10 in another.

## Tensor size

* + The total number of scalars a Tensor contains. For example, a [5, 10] Tensor has a size of 50.

## time series analysis

* + A subfield of machine learning and statistics that analyzes temporal data. Many types of machine learning problems require time series analysis, including classification, clustering, forecasting, and anomaly detection. For example, you could use time series analysis to forecast the future sales of winter coats by month based on historical sales data.

## Training

* + The process of determining the ideal parameters comprising a model.

## Trajectory

* + In reinforcement learning, a sequence of tuples that represent a sequence of state transitions of the agent, where each tuple corresponds to the state, action, reward, and next state for a given state transition.

## transfer learning

* + Transferring information from one machine learning task to another. For example, in multi-task learning, a single model solves multiple tasks, such as a deep model that has different output nodes for different tasks. Transfer learning might involve transferring knowledge from the solution of a simpler task to a more complex one, or involve transferring knowledge from a task where there is more data to one where there is less data.

Most machine learning systems solve a single task. Transfer learning is a baby step towards artificial intelligence in which a single program can solve multiple tasks.

## translational invariance

* + In an image classification problem, an algorithm's ability to successfully classify images even when the position of objects within the image changes. For example, the algorithm can still identify a dog, whether it is in the center of the frame or at the left end of the frame.

See also size invariance and rotational invariance.

## Upweighting

* + Applying a weight to the downsampled class equal to the factor by which you downsampled.

## user matrix

* + In recommendation systems, an embedding generated by matrix factorization that holds latent signals about user preferences. Each row of the user matrix holds information about the relative strength of various latent signals for a single user. For example, consider a movie recommendation system. In this system, the latent signals in the user matrix might represent each user's interest in particular genres, or might be harder-to-interpret signals that involve complex interactions across multiple factors.

The user matrix has a column for each latent feature and a row for each user. That is, the user matrix has the same number of rows as the target matrix that is being factorized. For example, given a movie recommendation system for 1,000,000 users, the user matrix will have 1,000,000 rows.

## Validation

* + A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model’s performance generalizes beyond the training set.

Contrast with test set.

## validation set

* + A subset of the dataset—disjoint from the training set—used in validation.

Contrast with training set and test set.

## vanishing gradient problem

* + The tendency for the gradients of early hidden layers of some deep neural networks to become surprisingly flat (low). Increasingly lower gradients result in increasingly smaller changes to the weights on nodes in a deep neural network, leading to little or no learning. Models suffering from the vanishing gradient problem become difficult or impossible to train. Long Short-Term Memory cells address this issue.

Compare to exploding gradient problem.

## Weight

* + A coefficient for a feature in a linear model, or an edge in a deep network. The goal of training a linear model is to determine the ideal weight for each feature. If a weight is 0, then its corresponding feature does not contribute to the model.

## Weighted Alternating Least Squares (WALS)

* + An algorithm for minimizing the objective function during matrix factorization in recommendation systems, which allows a downweighting of the missing examples. WALS minimizes the weighted squared error between the original matrix and the reconstruction by alternating between fixing the row factorization and column factorization. Each of these optimizations can be solved by least squares convex optimization. For details, see the Recommendation Systems course.

## Width

* + The number of neurons in a particular layer of a neural network.

# Natural Language Processing

## Bag of words

* + A representation of the words in a phrase or passage, irrespective of order. For example, bag of words represents the following three phrases identically:
    - the dog jumps
    - jumps the dog
    - dog jumps the

Each word is mapped to an index in a sparse vector, where the vector has an index for every word in the vocabulary. For example, the phrase the dog jumps is mapped into a feature vector with non-zero values at the three indices corresponding to the words the, dog, and jumps. The non-zero value can be any of the following:

* + - A 1 to indicate the presence of a word.
    - A count of the number of times a word appears in the bag. For example, if the phrase were the maroon dog is a dog with maroon fur, then both maroon and dog would be represented as 2, while the other words would be represented as 1.
    - Some other value, such as the logarithm of the count of the number of times a word appears in the bag.

## Embeddings

* + A categorical feature represented as a continuous-valued feature. Typically, an embedding is a translation of a high-dimensional vector into a low-dimensional space. For example, you can represent the words in an English sentence in either of the following two ways:
    - As a million-element (high-dimensional) sparse vector in which all elements are integers. Each cell in the vector represents a separate English word; the value in a cell represents the number of times that word appears in a sentence. Since a single English sentence is unlikely to contain more than 50 words, nearly every cell in the vector will contain a 0. The few cells that aren't 0 will contain a low integer (usually 1) representing the number of times that word appeared in the sentence.
    - As a several-hundred-element (low-dimensional) dense vector in which each element holds a floating-point value between 0 and 1. This is an embedding.

In TensorFlow, embeddings are trained by backpropagating loss just like any other parameter in a neural network.

## Embedding space

* + The d-dimensional vector space that features from a higher-dimensional vector space are mapped to. Ideally, the embedding space contains a structure that yields meaningful mathematical results; for example, in an ideal embedding space, addition and subtraction of embeddings can solve word analogy tasks.

The dot product of two embeddings is a measure of their similarity.

## Ground truth

* + The correct answer. Reality. Since reality is often subjective, expert raters typically are the proxy for ground truth.

## Natural language understanding

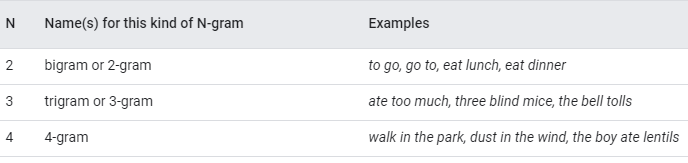
* + Determining a user's intentions based on what the user typed or said. For example, a search engine uses natural language understanding to determine what the user is searching for based on what the user typed or said.

## Neuron

* + A model that, taking inspiration from the brain, is composed of layers (at least one of which is hidden) consisting of simple connected units or neurons followed by nonlinearities.

## N-gram

* + An ordered sequence of N words. For example, truly madly is a 2-gram. Because order is relevant, madly truly is a different 2-gram than truly madly.



Many natural language understanding models rely on N-grams to predict the next word that the user will type or say. For example, suppose a user typed three blind. An NLU model based on trigrams would likely predict that the user will next type mice.

Contrast N-grams with bag of words, which are unordered sets of words.

## NLU

* + Abbreviation for natural language understanding.

## recommendation system

* + A system that selects for each user a relatively small set of desirable items from a large corpus. For example, a video recommendation system might recommend two videos from a corpus of 100,000 videos, selecting Casablanca and The Philadelphia Story for one user, and Wonder Woman and Black Panther for another. A video recommendation system might base its recommendations on factors such as:
    - Movies that similar users have rated or watched.
    - Genre, directors, actors, target demographic...

## sentiment analysis

* + Using statistical or machine learning algorithms to determine a group's overall attitude—positive or negative—toward a service, product, organization, or topic. For example, using natural language understanding, an algorithm could perform sentiment analysis on the textual feedback from a university course to determine the degree to which students generally liked or disliked the course.

## transfer learning

* + Transferring information from one machine learning task to another. For example, in multi-task learning, a single model solves multiple tasks, such as a deep model that has different output nodes for different tasks. Transfer learning might involve transferring knowledge from the solution of a simpler task to a more complex one, or involve transferring knowledge from a task where there is more data to one where there is less data.

Most machine learning systems solve a single task. Transfer learning is a baby step towards artificial intelligence in which a single program can solve multiple tasks.

## Trigram

* + An N-gram in which N=3.