CMPUT 267 Notes:

Chapter 1: Introduction

* ML uses range of techniques to learn from data
* Prediction is important
* F: Rd🡪 R inputs an attribute or features about an item and produces a prediction about said item
  + Might need previous examples to be able to make a formula
* Natural goal is to find a function which will accurately recreate data by formalizing it into an optimization problem
  + Imagine we have a space of possible functions F which we need to select our function (f), we try to find a function from the class od linear functions which minimize the squared differences.
  + E.g. imagine we have a linear function f(x)=w0 + x1w1+x2w2+x3w­3 for any w= [w0 w1 w2 w3] ∈ Rd. where w0 is linear intercept of linear function, our optimization problem becomes minf∈F ( sum from 1 to 10( f(xi­) -yi)2)
* When we got the function, we can hope it is close enough to the intended data we aim for.
* Always wonder if we interpolated well
* Use probability to deal with uncertainty
* ML can be an applied field of maths
* A set of items has to brackets used. {} indicates it contains only those items whereas [] indicates an interval between the two numbers in the brackets . Sets usually uses chi (χ), Ω or ϒ.
* Sets with multiple dimensions, have to specify the set of each dimension
  + Write it like [1,1]x[2,6] where the two sides of the x indicate the set of current dimension. The number of paired brackets match the number of dimensions
  + Not necessarily the same type can mix types of sets
  + Can be from two different sets. χ x ϒ
* Logs and exponentials are to be used due to probabilistic formulations
* Ln and e are inverses of each other (exp = e)
* Recall Log and exp rules:
  + Ln(ab)=ln(a)+ln(b)
  + Ln(a/b)=ln(a)-ln(b)
  + Ln(ab)=b\*ln(a)
  + Exp(a+b) = exp(a)exp(b)
  + Exp(aB)=exp(ab)
* Continuous: No abrupt changes in value/discontinuities
* Recall integrations is used to find the sum under a function between two given points
* Optimization will use derivatives and can be used to check If a function increases or decrease at a given point. Flatter the function, closer the derivative is to zero

Chapter 2

* Each experiment has an outcome in which an element is drawn from a setoff predefined options
* Rules of probability are based on the axioms of probability
  + Important things to note prior to definition
  + Let the sample space Ω be a non-empty set of outcomes and the event space ε be non empty subset of Ω with a possible event A. The event space must have the following properties to be called a measurable space
    - A ∈ ε 🡪 A’ ∈ ε where A’ is A compliment
    - A1,A2,… ∈ ε (The union of all possible events must be in the event space)
    - ε is non empty
  + A function P : ε🡪[0,1] satisfies probability axioms if
    - P(Ω) = 1
    - A1,A2,… ∈ ε, Ai ∩ Aj = ∅ ∀i, j. This means the probability of the union of disjoint sets equals sum of their probabilities ( P(A U B) = P(A) + P(B) as there is no overlap between the two events
* Probability space is the tuple (Ω, ε, P)
* The probability of the union of all Ai for i🡪N is equal to the sum of all possible events Ai.
  + This is because all AN+j for j>=1 can be set to the empty set ∅ where Ai ∩ ∅.
* Countable and uncountable indicate whether a set can b enumerated or not.
  + Discrete are usually countable but continuous aren’t
* Once we get a random variable, we can get a valid probability space
* We can define the distribution using a function defined directly on instances x ∈ χ rather than every possible event
* A probability mass function (pmf) is a function p: χ 🡪 [0,1] which when summed for all x in χ, equals one. Probability of the event A ∈ ε is defined as P(A) sum of all x ∈ A P(x).
* To specify P, we need to determine how to specify the pmf ( often specified as a table of probability values) .
* The Bernoulli distribution is derived from the Bernoulli trial which either has an outcome of success or failure. Probability of success is α ∈ [0,1] and thus failure occurs with a probability of 1 – α.
  + Graphical user interface, application

    Description automatically generated
  + Often written as Bernoulli(α)
  + Used for binary outcomes
* Uniform distribution has no parameters and is defined as by the size of sample space
  + Uniform(n)
  + P(x)=1/n for all x in χ from 1 to n
* Poisson Distribution reflects the probability of how many incidences occur.
  + Represented as Poisson(λ) which indicates the number of expected outcomes
  + χ = {0,1,….} and for all x in χ,
  + Graphical user interface, text, application

    Description automatically generated
  + This is hill shaped in which the top is centered around λ
  + Defined over countable infinite sample space
* Probability density functions (pdf) replace probability mass functions and integrals replace sums in continuous probability spaces
  + Cant uses tables and restricted to functional forms
  + Must find probabilities of intervals rather than a specific value due to chances being astronomically small and having zero mass
* We assume that the set of events ε consists of all possible intervals in continuous spaces (known as the Borel field B(χ) )
  + If χ is in R, the field contains all open, closed and semi intervals
  + Smaller than power set
  + Can be defined for higher dimensional sets
* Text, application

  Description automatically generated with medium confidence
  + Not restricted to [0,1] but [0,inf)
* Probability of a single event {X} in pmf is value of pmf at point X p(X)
* Probability of an event must be less than or equal to 1
* Density does indicate there is a high likelihood around said point which means there is a huge density around x and other points zero
* Cant easily define a pdf’s p to provide specific probabilities.
* We never use P(X=x) as it will be 0 for pdfs, instead we write P(X∈A) or <= / >= A
* 4 main pdfs
  + Uniform distribution: Defined by an equal value of a pdf over finite interval in R. χ=[a,b], pdf is defined as p(x)=1/(b-a). Also defined as Uniform(a,b) by taking χ =R .
  + Exponential distribution defined over a set on non negative numbers. Parameter λ>0, pdf p(x)= λe-λx. Sharply decreasing probability for values x as they increase.
  + Gaussian Distribution/Normal Distribution. Defined over χ=R with 2 parameters of μ ∈χ and σ >0. Pdf p(x)= 1/(√2πσ2) \* e-(1/2 σ^2 ) \*(x-μ)^2 
    - For a Gaussian Distributed random variable, parameter σ2 is variance and μ is mean. Referred as either Gaussian(μ, σ2) or N(μ, σ2).
  + Laplace Distribution is like Gaussian but peaks around mean. Two parameters of μ ∈χ and b>0 , Defined over χ=R and its pdf p(x)= 1/2b \* e-1/b \* abs(x-μ)
  + Gamma distribution used to model wait times. Akin to Poisson distribution but is used for continuous variables. Defined over χ=(0,inf) with a shape parameter α>0 and rate parameter β>0 with its pdf = βα/Γ(α) \* xα-1 \*e-βx
* Integrate your value of p(X) for given distribution between the interval of whose probability you wish to find.
* Multivariate variables are a vector of random variables.
* Similar procedures to above as the definition of outcome spaces and probabilities are general.
* Marginal distribution is probability distribution of specific probabilities within our multivariate variable.
* PMF can be table of join probabilities
* We can consider d-dimensional random variable X=(X1🡪Xd) with vector-valued outcomes x = (x1🡪xd) such that each xi is chosen for some χi.
* Graphical user interface, application

  Description automatically generated
* Conditional probabilities: A random variable’s probability given info about the value of another random variable
  + P(y|x)=p(x,y)/p(x). p(x,y) is the joint probability, p(x) is the marginal probability and p(y|x) is the conditional probability. Read as Probability of Y given X
* Graphical user interface, text, application

  Description automatically generated
* Graphical user interface, text, application

  Description automatically generated
* Order of variables do not matter for above formula
* Two variables are independent if joint probability factors into the product of the marginals
  + P(x,y)=P(x)P(y)
* If P(x|y)=p(x) this means that the Y has no effect on X, hence independent, hence giving us above formula.
* Can be generalized to 2+ variables
* Conditional independence is also used:
  + P(x,y|z)=p(x|z)p(y|z)
  + X and Y can be independent but not conditionally independent whereas X and Y can be conditional independent with Z but not independent.
* Distribution reflects beliefs and knowledge rather than the objective truth.
* Expected value of a random Variable X is the average of repeatedly sampled x in limit of sampling.
* E[X] = sum of x\*p(x) for all x in χ for discrete variables and integral of xp(x) dx for all χ
* For a function χ🡪R, its E[X] is defined as
* Graphical user interface, text

  Description automatically generated
* Moments: Expected value of functions of random variable X
  + Variances is important c=E[x]. This indicates the amount the random variable varies around mean. Large means Gaussian is wide but if variance is small, dense around the mean
  + Conditional Expectances:
    - Graphical user interface, application

      Description automatically generated with medium confidence
* For 2 random variables:
  + Graphical user interface, application

    Description automatically generated
* Covariance of X and Y
  + Cov[X,Y]=E[XY]-E[X]E[Y]
  + These indicate how the two variables vary together
  + V[X]=Cov[X,X] = Variance of X
* Correlation of X,y = Cov[X,Y]/( √V[x] \* √V[y])
  + Between -1 and 1
* Important Properties:
  + Graphical user interface, text, application

    Description automatically generated

Chapter 3

* Assume we have n random variables X1🡪Xn­ ­ where E[Xi] = μ for some unknown mean μ. Sample average estimator is Xbar= 1/n \* sum\_{i=1:n} of Xi
* As n increases, Xbar gets closer to the actual mean μ.
* Bias(XBar)=E[XBar] – μ
* If Bias =0, unbiased estimator
* Variance shrinks in proportion to number of samples but as this doesn’t tell us how close Xbar is to μ, we use concentration inequalities
* Aim is to get a confidence interval around our estimate.
* For any ε>0, there exists δ>=0 s.t
* Graphical user interface, text, application

  Description automatically generated
* Aim is to get a small property δ that XBar deviates by ε from the mean E[XBar]
  + Interval and δ must be small to be more confident
  + Graphical user interface, text, application

    Description automatically generated
* If δ = 0.05, we get a 95% confidence interval (95% confidant that the true value lies within our interval)
* Hoeffdings’s Inequality states for an ε>0,
* Graphical user interface, text, application

  Description automatically generated
* Hoeffdings assumes bounded random variables
* Chebyshevs inequality states that for random variables with variance σ2
* Graphical user interface, text, application

  Description automatically generated
* Not bounded between a and b unlike Hoeffdings and we know variance here but our interval is proportional to sqrt(1/n)
* Less strict conditions but gives a less tighter bound
* Consistency implies that XBar approaches μ and sqrt(σ2/δn) approaches 0 as n approaches infinity
* Sample complexity n is number of samples needed to obtain an accurate ε estimate,
* Make complexity small to be data efficient
* Determined by estimator and data properties
* We can improve complexity by using smarter estimators but have higher sample complexity for specific data sets
* Chebyshevs inequality has a convergence rate of O(1/sqrt(n)) and doesn’t make a distributional assumption on Xi
* E[Y-μ2]=Var[Y]+Bias(Y)2
* Steps:
* Graphical user interface, text

  Description automatically generated
* MSE is made of Variance of Y and Bias of Y and if estimator is unbiased, then solely the variance
* High Bias=High MSE

Practice Quiz:

Graphical user interface, text, application

Description automatically generated

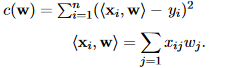
Graphical user interface, text, application

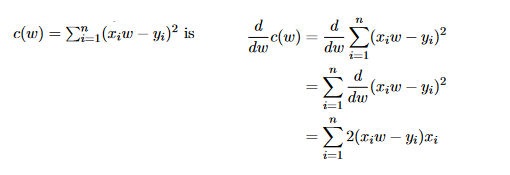
Description automatically generated

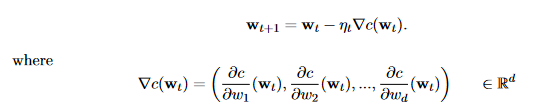
Graphical user interface, text, application

Description automatically generated

Chapter 4:

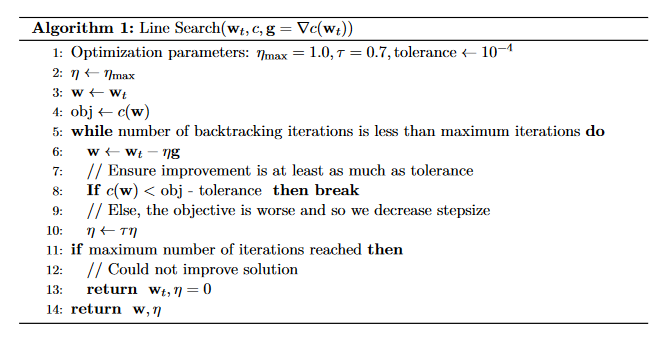
* Aim of optimization is to get a set of parameters w to minimize a function c where W is the set of all possible parameters
  + Linear regression which minimizes square distance, we use a dot product formula 
  + Continuous optimization problem as w is a real valued variable in a continuous objective
* Use objective over error when naming
* Combining error with other terms give us objective we aim to minimize
* In discrete optimization, the set of elements W is a finite(discrete) set.
  + Test each values and find which returns lowest value. Sometimes takes too long hence we may want to change it into an optimization problem
* Maximizing negative of object is the same as minimizing the objective
  + Note min and max values are not the same
* To find w (parameters), one could do a random search.
  + Idea is randomly generate w and then check to see if it is better than our previous solution and if so, save the w and repeat
  + Better to use gradient descent
* Aim is to search for stationary points where derivate is 0 (Gives us local, min/mx or saddle points)
  + E.g.



* We can infer type of stationary point from objective properties
  + Convex Functions are functions such that for any w1 w2 and t=[0,1] 
* We can use second derivate test to infer the type of point
  + Local min has a second derivate value above zero while a local min has a value less than 0 . If = 0, test is inconclusive
* Idea behind gradient descent is to approximate using a Taylor series
  + 
  + Cn is the nth derivate at point w0
* Locally w1, will be better than w and still be a stationary point of c but moving far enough from w0 makes the Taylor series inaccurate. Hence we iteratively use  until we get to a point that the derivate is nearly or equal to zero. (Second order gradient descent)
* In first order gradient descent, we get a worse approximation as we don’t know true second derivate nut instead we use a stepsize() s.t. .
  + Update is wt+1 = wt - after solving wt+1 argminw c(wt) + (w-wt) + (w-wt)2
  + We can set given access to second derivate
* Gradient descent for multivariate variables is :
  + 
  + Each partial derivate indicates how function c changes if all variable in wt are held constant for j’th element
* Selecting step-size is very important as if we take too small of one, we take too many iterations to get to a stationary point but if take too large of one, we hover around minimum. Thus we need an adaptive step size. One such example is line search



* + Solving is expensive but we can get approximate solutions quicker
* One method we can use is to use back track line search which selects largest reasonable step size and then decrease from there
  + Idea is to search along line for possible step sizes with the idea that a large step size is good. It is overshoots, reduce step size with reduction usually following r\*step size with r being between [0.5,0.9]
* Line search Alg

* + 
* After getting a stationary point, we need to check if it’s a local min max or saddle point while checking if we can say it’s the global minimum if we found out it is a local min
  + We have a global min if we have one stationary point and second derivative test says it’s a local min
  + If we have two bounds a,b and we have a stationary point for a convex function s.t. it is between [a,b] then boundaries aren’t part of solution but if the point is outside the feasible set, then one o the boundaries could be a solution
* Adding or multiplying by a constant doesn’t change solution

Chapter 5:

* Usually objective is to find a model that approximates true underlying model that generated the data.
  + Pick distribution and parameters
  + Write down optimization that formalizes which parameters are the best given observed data (Maximum a posteriori estimation (MAP) , maximum likelihood Estimation (MLE) and Bayesian Estimators )
* MLE
  + Data is drawn from some true distribution which is unknown to you. Only known fact is its in a set of possible distribution called hypothesis space
  + E.g.

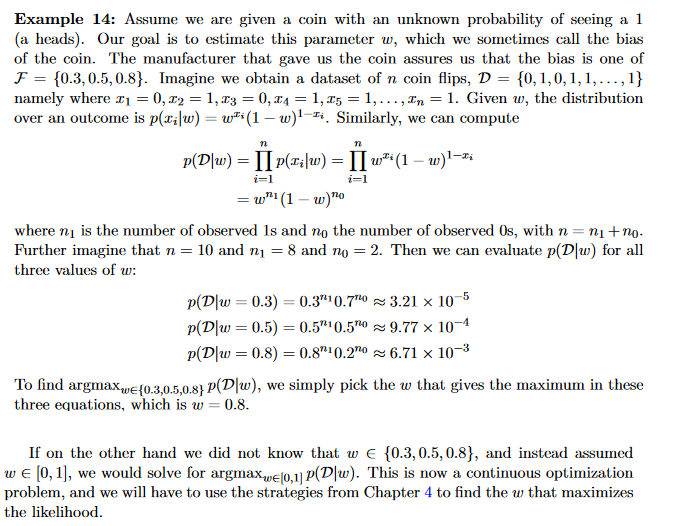
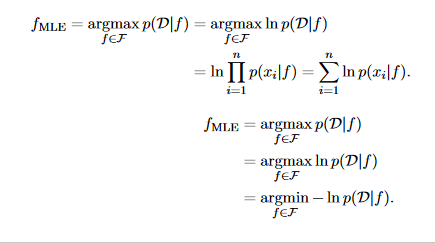
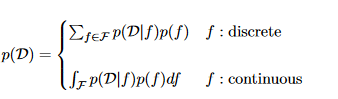
* + - 

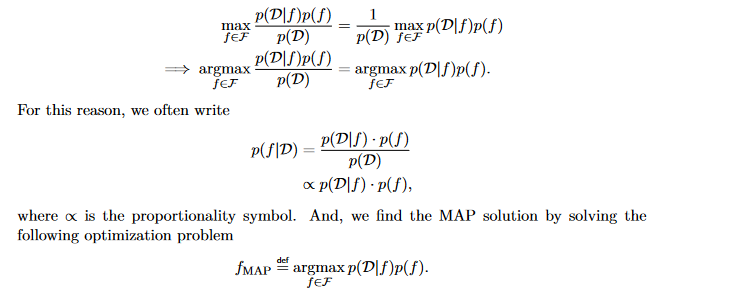
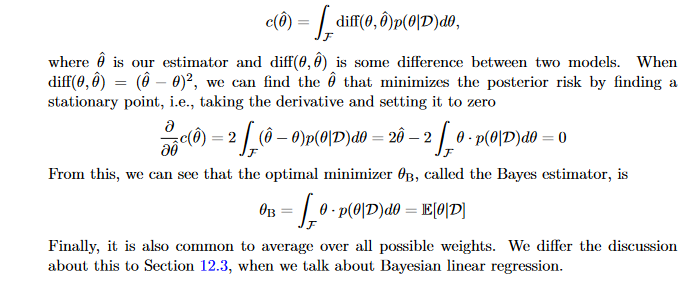


* + - True distribution has parameters μ\* and σ\* and aim is to use data to find μ and σ close to these as possible
  + One objective is to pick the function that makes data most likely known as the maximum likelihood :



* + - P(D|f) is called the likelihood of the data given model
    - If data has low likelihood for distribution given by f, then it is unlikely that the specific f corresponds to true parameters generated by data.
    - Conversely the f for which data is most likely is the more likely to correspond to f\* especially if we have a lot of data.
  + Model=Function and parameters are the functions co-efficients

* + 
  + As gradient can get annoying to complete, we can use the log likelihood or a negative log likelihood for minimization  
    
    - Note that the min and max values ARE NOT the same
    - Aim is to get the parameters not min/max value
* MAP estimation
  + Aim is to find most probable model given data
  + Given data set D, we formalize MAP solution as 
  + In discrete model spaces P(f|D) is pmf and the MAP estimate is exactly the most probable model while In continuous spaces, MAP estimate us model with largest value of posterior density function.
  + To get posterior probability, we use Bayes rule.
    - 
    - P(D|f) is likelihood function, p(f) is prior distribution and p(D) is marginal distribution of data
    - 
  + Finding FMAP  can be simplified because p(D) in denominator doesn’t affect the solution as it’s the same regardless of f in maximization so we can scale it due to not changing relative ordering

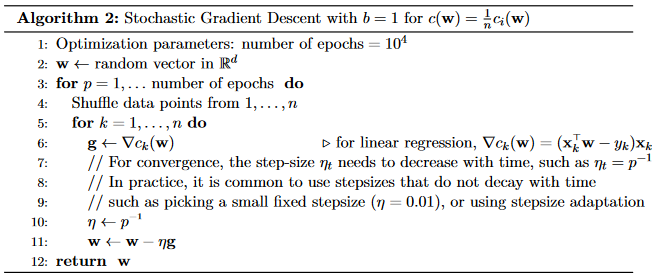
* + - 
    - We can take the log without changing the relative order and re-write as a min problem by taking -ve
    - 
  + Sometimes we don’t prefer a single model over others
  + Thus we can think p(f) as a constant over Model Space F s.t. if p(f)=c for some constant c. MAP reduces to the maximization of log likelihood 
    - This is because the constant c comes out of maximization and wont affect ordering
    - Solution above is an MLE(Think of it as a special ase of MLE)
    - P(f) being constant is problematic as a uniform distribution can always be defined
    - Recall you have to take partial derivates of parameters to get gradient
* MLE and MAP report solution according to likelihood function and mode of posterior solution respectively but both don’t consider skewed or multimodal distributions which Bayesian estimation does
  + Uses estimation of full posterior P(θ,D)
  + Can use posterior to assess range of plausible parameters given data
  + Credible Interval; When the density at just 1 s.d remaining high
  + Can use posterior to also define an alt objective for selecting point estimate rather than mode
  + We aim to minimize posterior risk in Bayesian stats
  + 
* To get P(D) explicitly we can do the following:
  + 
  + To get posterior or posterior mean, we need complex integrals
* Selection of prior distribution has important implications on calculation of posterior mean
* Conjugate prior: When likelihood is multiplied by the prior the resulting remained in the same class of function as prior
  + Simplify computations in 2 ways:
    - Non-trivial to actually derive p(λ|D)
    - Knowing form of resultant distribution allows us to get statistics of said distribution
* MAP and MLE for conditional probabilities are possible
  + Recall p(y|x) is a conditional probability on 2 random variables X and Y
  + We ask: given auxiliary info what’s distribution over data. When auxiliary info changes🡪distribution also changes
  + Can be formed from any distribution families and can formulate estimation problems
  + Parameters tied to X variable
* To find stationary points, we could use gradient descent instead
  + 

Chapter 6

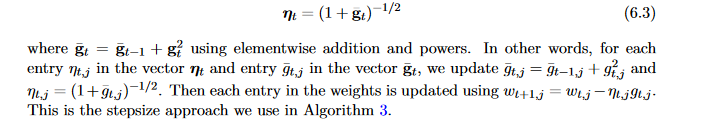
* One way to handle big data sets is to use stochastic approximation where the gradient is approximated using stochastic sample
  + We use a random subsample to get a good approximation
* We use a mini batch of b samples for stochastic gradient descent(SGD)
  + Using b=1,
  + Gradient for a single sample is an unbiased estimate of true gradient
  + True full gradient can be seen as an expectation with p(i)=1/n where K is a random index into (1🡪n) s.t.



* + We randomly sample K and as such, our update is an biased estimate of the average sum of gradients from 1🡪 n.
  + If one step points in wrong direction, on average it will still point to write direction and move towards the stationary point slowly
  + Instead of randomly sampling, we can shuffle the order of points and then iterate over dataset. Each iteration is called an epoch and we have multiple epochs to loop through dataset multiple times.
  + Algorithm:

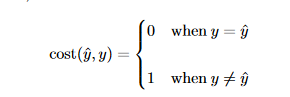
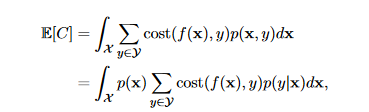
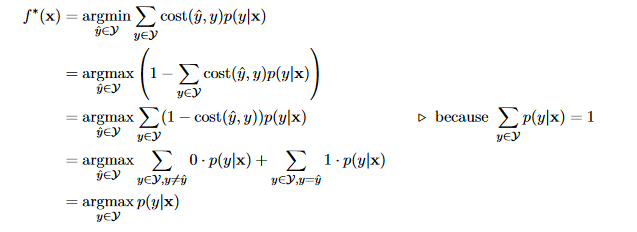
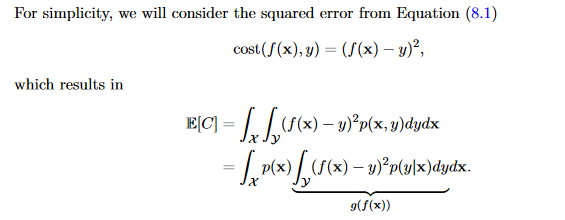
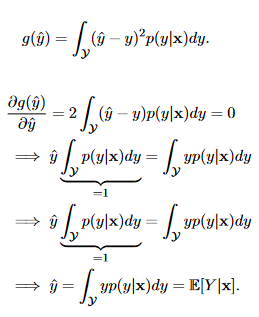


* + B=1 is usually too high of a variance and thus we pick b>1 and larger b, lower average. Setting b=n corresponds to full batch gradient descent. Thus extremes are not preferred
* SGD needs a new way to pick step sizes and is required to decrease over time.
* AdaGrad is a way in which wee] normalize step size by sum of accumulating gradients
  + 
  + After several epochs, SGDs will converge and oscillate around true vector weight
  + 
* Another way is to use different step sizes per dimension (how AdaGrad is designed)



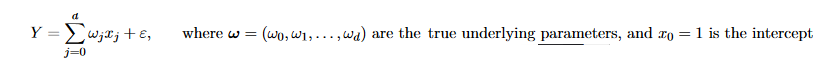
* We cant guarantee improvement on each update for b=1 only aggregate improvement but for full gradient we can
  + This is due to the number of updates to k
* In General SGC id better than gradient descent as GD can be seen as an extreme of SGD where b=n.
  + Set b as an intermediary value as b=1 is too noisy

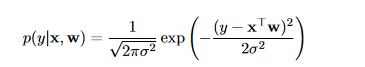
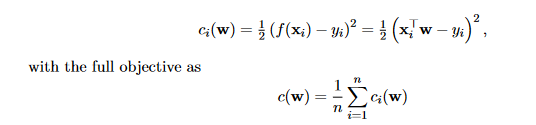
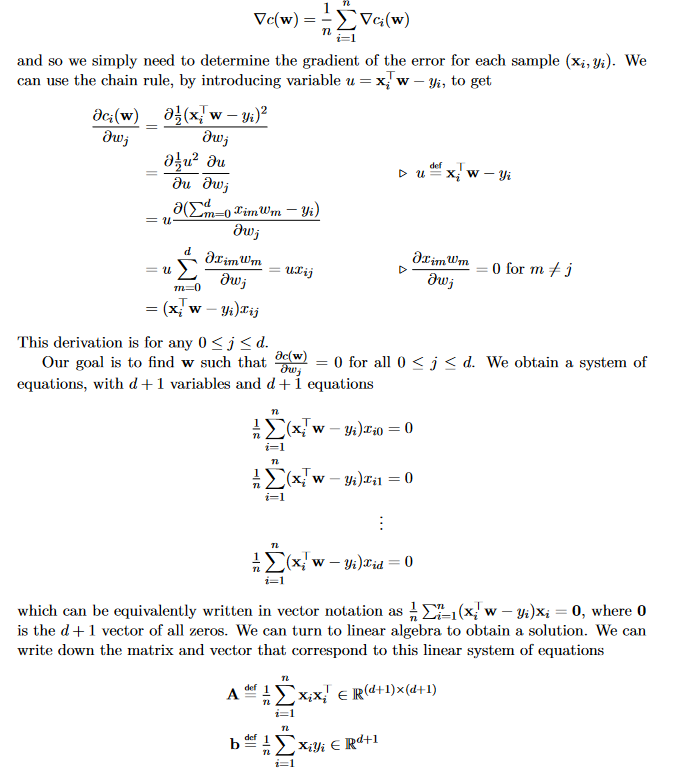
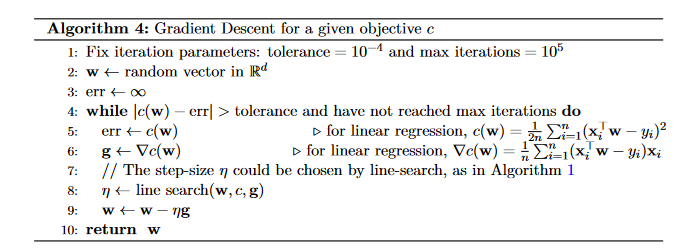
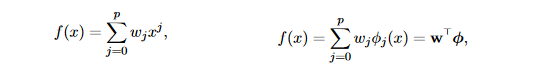
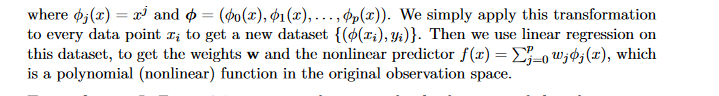
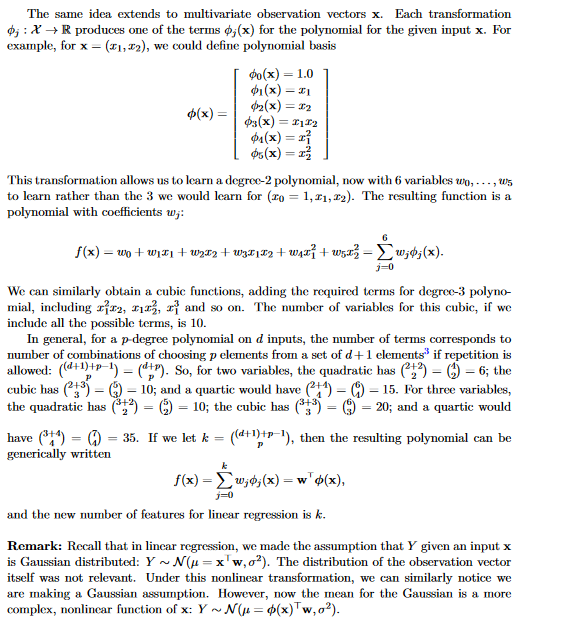
Chapter 7

* Dimensions to categorize machine learning problem:
  + Passive v active
  + i.i.d. (independent and identically distributed random variable) vs non i.i.d.
  + complete vs incomplete
  + Not necessarily will fit into these categories
* Data set D = [ (x1,y1) , … ,(xn,yn)] where xi∈X is i- th input and yi∈Y the corresponding target. We assume X=Rdin which case xi= (xi1 🡪xid) is a d-dimensional vector called instance/sample.
* Each dimension of xj  is called a feature/attribute.
* Organize dataset into a matrix X∈Rn x d where each row corresponds to sample xi and column to a feature
* Distinction between x and y is due to fact we assume features are easy to get for each object while target variable is difficult to observe or expensive to get, Thus we aim to build a model that predicts targets from set of input values.
* Differences in alg for prediction problems with i.i.d complete data typically arises from properties of inputs and properties of the targets.
* Common way to handle differences is to map different observations into Euclidean space where observation is re-represented as a real-valued vector due to many prediction algorithms being designed for real valued observations
* Properties of target are important and thus we get two distinctions for prediction problems which are classification(discrete Y) and regression(continuous Y) and can be further divided into multi-class(single label for an input) and multi label(related to more than one label) problems for classification.
  + Output for multi-class and multi-label is an indication vector
  + Regression considers order in target variables whereas classifications assigns label
* Key distinction is whether variables are ordered or unordered (regression vs classification respectively)
* Biggest question of supervised learning is what type of nonlinear functions we learn on x to best model distribution
  + Ordering also relies on this
* We can threshold the predictions returned by regression model by rounding them to closest integer
* Optimal predictor based off cost function where cost: Y x Y 🡪[0,inf) where cost (y\_hat,y) reflects cost/penalty of predicting y\_hat when true target is y.
  + As X and Y are random, cost(F(X),y) Is also random
* Types of cost:
  + 0-1 Cost :
    - 
    - Applies to binary or multi class classifications
* Squared Error:
  + 
  + Used in regression
* Absolute Error:
  + 
  + Used in regression
* To get expected cost using the predictor value f, Discrete set Y, and y\_hat=f(x)
  + 
  + Target will be random
  + We can obtain f\*=arg minE[C] by
    - 
    - Known as Bayes risk classifier
* Bayes risk classifier of 0-1 cost:
  + 
* Expected cost of regression:
  + 
* 
* If we assume f(x) is flexible to be separately optimized got each unit volume dx, we have an issue of finding y\_hat for each x to min. Thus we different g(y\_hat) w.r.t y\_hat till derivate =0
  + 
* We can know write expected cost in the cases of both optimal and suboptimal models f(x)
  + F(x)=E[Y|x]
  + F(x)!=E[Y|x]
* When f(x)=E[Y|X], E[X] can be expressed as
* 
* For second scenario
  + E[C]= E[ (f(x) - E[Y|X] )2 ] + E([Y|X]-Y)2))
  + E[ (f(x) - E[Y|X] )2 ] = Reducible error while E([Y|X]-Y)2))= irreducible error
  + 1st part reflects how far trained model is from optimal model while second term reflects inherent variability

Chapter 8

* Given a dataset , we aim to get the relationships between features and target
* Start by hypothesising functional form of relationship
* Aim to estimate linear functions modeled as a linear combination of features and parameters
  + 
* This is where we extend x to x0=1,x1 ,x2….,xd) for a dimension d. Allows us to use f(x) as a dot product and avoid dealing with intercept
* Linear Regression: Finding the best parameters w∈Rd+1
* Assume datapoints xi are according to some unknown distribution p(x). We then assume the target variable Y must have an underlying linear relationship with our input X (X1🡪Xd) plus an added noise term ε which follows a zero-mean gaussian distribution.
  + Target y is a realization of a random variable Y given input x s.t.

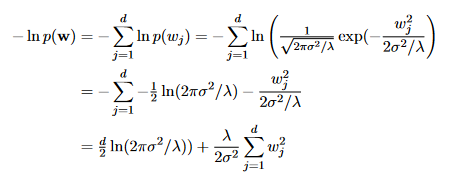
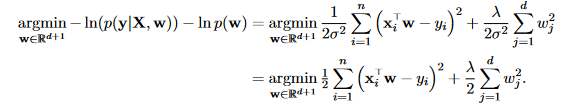
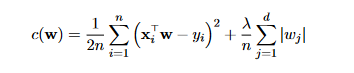
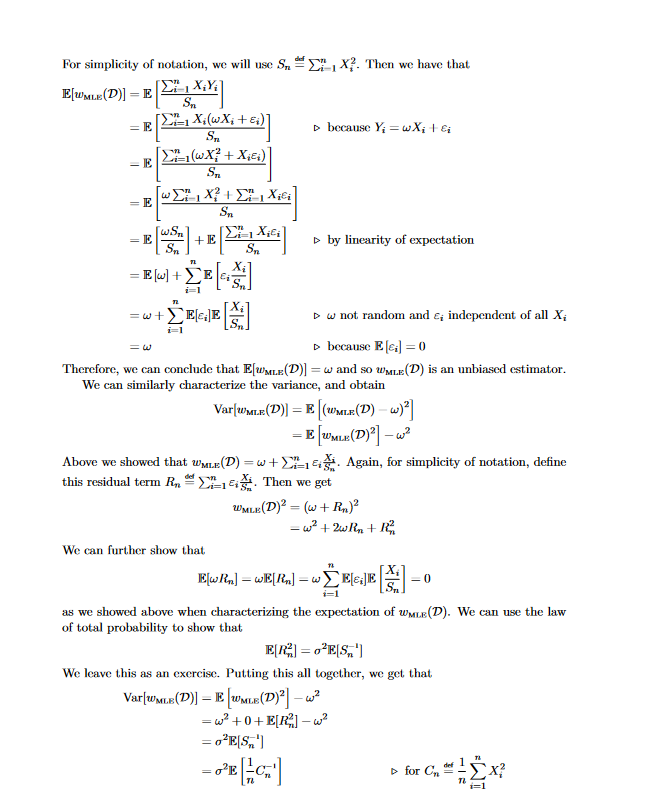
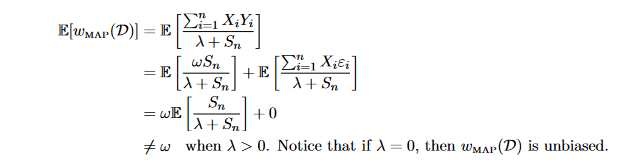
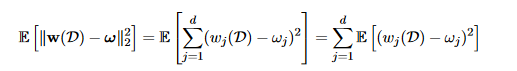
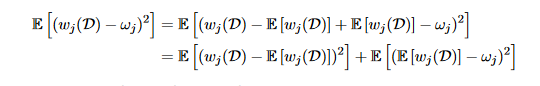
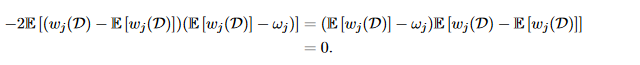


* Independence between ε and X doesn’t always hold but the assumption of normality for error is reasonable
* Seek to approximate target as f(x) = wT x where w is to be determined. We first write conditional likelihood for a single pair (x,y) with a conditional density of  as Y given x also is Gaussian
  + 
* MLE is where we assume space of possible values for the weight is 
  + 
* Our prediction
* MLE formation states we should find weights which min square difference between prediction and given value
* We use MLE to define a reasonable objective for our parameters and when we use other distributions, guessing a good objective isn’t entirely obvious. There is also an underlying linear relationship between features and target and an absence of noise in feature collection
* To min sum of squared errors:
  + 
* Add normalization by num of samples so we have an average squared error rather than cumulative error
* Average error doesn’t grow with more data whereas cumulative error can get really big as n increases
* Sum of gradients for each sample is the gradient of objective
* 
* Goal from this is to find w such that Aw=b and if A is invertible then w=A-1 ­b albeit using gradient descent is preferred due to being more efficient as each gradient update has an update cost O(nd). However, we tend to use stochastic gradient descent
* 
* Can apply linear regression to broader things such as getting non linear functions by applying a non linear transformation to observation vector x prior to fitting step
  + Linear function in this space provides a non linear function in original space
* To achieve a polynomial fit of degree p where p is degree of polynomial:
  + 
  + 
* 

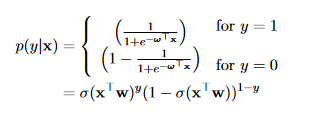
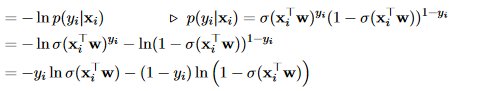
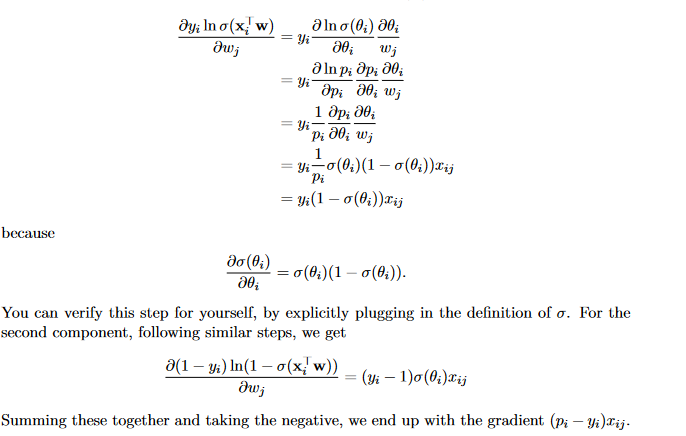
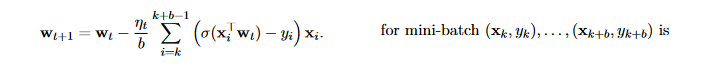
Chapter 9

* Generalization error: Expected cost of a model across all data points
  + Expected loss for a given loss is
  + 
* Cant directly min generalization error nut minimized empirical error
  + In regression we minimized sample average squared error 
  + Sample average reasonable estimate of expectation
* Sometimes minimizing the empirical cost can produce functions which generalize poorly
* Zero empirical error=training error
* Overfitting: Function is overly specialized to training set at expense of other data points
* Not obvious to see when it occurs but take a look at magnitudes of coefficients and see if there is an increase
* Can fit noise in targets by adding and subtracting large numbers to get precise y values
* Underfitting can also occur which means it is insufficient to represent true model
* Get samples of generalization error to determine overfitting
  + Use a test set by splitting data into training and test( n and m)
  + 
  + Use confidence intervals with sample avg error on test set to see the level of certainty in estimate of generalization error( if wide maybe get more data)
  + Can’t use all data for training which is a -ve
    - However it can be biased if using all data
* Significant mismatch between training and test error overfitting occurred
  + Expect training<test error
  + Weights chosen to minimize error
* Use different models to gauge
  + E.g. trying all polynomials to degree p and see what degree the training error stops decreasing then increasing
  + Another way to see this is by noting the that it is strictly more flexible to pick polynomials of higher degrees as we can always set coefficients of higher degrees to zero. Can also be used for underfitting
* We can compute sample average error for a learned function f using m smples of error from test set Dtest= {(Xi,Yi)}ni=1 by doing :
  + 
  + Ci(f)if error of ith sample
* Students t-distribution is used when we don’t know true variance
  + Confidence interval: where S2m =
  + depends on sample number
  + Not very powerful
* Use tests to compare two means
  + Can use a ranking system by comparing the two algs for each sample and awarding a point for the better alg
    - Use a binomial dist to express probability that f1 collected k more wins under null hypothesis by the following
    - 
    - Quality of f1=quality of f2 is null hypothesis and alternate is dependant on what you aim to find
    - P in this case is the p-value which represents the likelihood of observing these outcomes if null hypothesis is true.
    - Reject or fail to reject based on alpha value which is arbitrary
* Paired t test used when we have pairs of real values
  + is sample average where di is the difference between two values
  + T=
* Both tests make parametric assumptions
  + Binomial requires values to be either 0 or 1 while paired t test assumes equal variance
* Type II error fail to reject null when it should be
* Type I error is reject when it shouldn’t have been rejected

Chapter 10

* We discussed linear regression in terms of max likelihood but can also use a MAP objective
* Select a prior to regularize overfitting to observed data
* Two major prior which are the Gaussian prior (l2) norm and Laplace prior (l1 norm)
* Gaussian prior
  + Assume each element wj has a Gaussian prior of with a covariance of 0 between weights for some lambda>0 and under assumption p(y|x) = .
  + By taking a gaussian on each wj we get the prior p(w)= p(w1)p(w2)….p(wd). Upon taking log of this zero-mean gaussian prior we get 
  + We can drop 1st term as doesn’t affect selection of w as it is a constant
  + Can combine negative log likelihood and negative prior to get the following
  + 
  + Recall the following:
    - 
  + Regularization will not include w0 as this is intercept which will shift function only
* Laplace sist more concentrated around zero, produces more sparse solutions and penalizes large values in w
* Forcing entries to be 0 in w is similar to feature selection as zeroing entries Is the same as removing corresponding features
* If we drop entries in x and w where wj0, the Lasso can no longer have a closed form solution as we cant solve w in closed form that provides us a stationary point.
  + Use gradient descent instead but note that l1 is non-differentiable at 0
  + Lasso: 
* Selection of regularization parameter leads to a bias-variance trade off.
* Univariate setting🡪 input x∈ R and weights w ∈ R with f(x)=xw.
  + Presume assumptions behind linear regression are true
* There exists true parameter ω s.t Yi = ωXi + εi
  + εi are random variables according to N(0, σ2)
  + Can characterize MLE weights as a random variable where randomness across possible observed datasets
  + Another way to think about this is considering dataset D to be a random variable and solution wMLE(D) being from that dataset as a function of this random variable
  + Stochasticity comes from fact we could have drawn different datasets S and reason about resulting distribution over possible wMLE(D)
* We can show wMLE(D) is an unbiased estimator for ω.
  + wMLE(D) =
* 
* Final format given in terms of sample average estimate Cn which is the same as the sample average estimate of variance for Xi and reflects variability in X
  + Small Sample num Cn could vary widely and could be small
  + Inverse for above can be very big🡪 variance of w MLE(D) can be large for small amount of data
  + This implies wMLE(D) can be very different for different datasets. Undesirable
  + As n increases, variance decreases proportionally to n-1due to 1/n infront of Cn
* Regularized solution is less likely to have high variance but will be biased.
  + Let wMLE(D) be Map estimate for l2 regularized problem with some lambda>0. We can write MAP solution in this closed form solution 
* 
* When λ = 0 for above, corresponds to MLE solution for λ = 0 thus unbiased
* Bias is determined by how far  is from 1. And as λ🡪 0, solution is less biased and as λ🡪inf solution becomes maximally biased.
* Can characterize variance of wMAP(D) 🡪 
  + Even if Cn is small it will not cause variance to become very big as we aren’t inverting it but use inverse of Cn, λ and as thus for large λ, variance would not be bid and be smaller than wMAP(D) as we increase denominator by λ
* WMLE is unbiased but depending on number of samples can be high or low variance
  + As n increases variance decreases
* WMAP  is biased but depends heavily on λ. If λ is small, bias is minimal a it will be similar to WMLE . As λ increases however, bias increases but variance decreases
  + For large λ and small samples 🡪 high-bias, low variance but as sample n increases low bias and low variance
* Expected mean squared error to true weights can be decomposed to bias and variance
  + Optimal λ lowers bias-variance tradeoff
* Decomposition for weights for regression: 
* Can simplify further to
* 
* Second step follows from 
*  is variance of jth weight whereas  is bias of jth weight
  + =  as nothing is random in term so the outer expectation is dropped. This gives us the following 
* The above tells us that the expected mean-squared error to true weight vector ω decomposes into squared bias and variance.
* Bias variance trade off reflects fact we could reduce mean-squared error by incurring some bis as long as variance dereased more than squared bias
* Realizable setting where function class contains true function
* Bias can be introduced from
  + Regularization
  + Selecting simpler function class
* If true function is not linear, cant compare learned weightsfor linear function directly to true function
* Generally talk about bias-variance trade-off by thinking about reducible (aim is to reduce this as much as possible)
  + Recall reducible error is  where f(X) is optimal function
* Given an input x, expected mean squared error is
  + 
* If we select a simple function class the class is likely not large nor powerful enough to represent true function introducing some boas but lower variance
  + Underfitting can occur if too simple
* Too powerful of class leads to low bias but high variance as one can find a function that overfits a given dataset .
* Following bias-variance trade off scenarios:
  + Low Bias and Variance: Large F(high model complexity) to nearly represent f\* with big n or Small F but f\* is simple and as such don’t need many samples
  + Low Bias but High Variance: large F but n is not enough
  + High Bias Low Variance: Small F but f\* is complex
  + High Bias and variance: Avoid being here. Very little data and interim model complexity
* To select models
  + Use first principles and model evaluation
* Can measure bias so cant optimize it
* Small sample size🡪simple models
* Split dataset into training and test(validation) set

Chapter 11

* We need to parameterize p(y|x) which must be a Bernoulli distribution as Y is binary (i.e. 1 and 0). It has a parameter α(x) = p(y=1|x) which is the success probability.
* Use logistic regression to get said parameter
* Need to approximate p(y=1|x) which has a value between 0 and 1.
  + Use a simple linear function of x and then squash values between 0 and 1 using a sigmoid function
  + Sigmoid function takes t as a parameter and does the following: 1/(1+exp(-t))
  + Approximate p(y=1|x) with 
* Bernoulli distribution over Y with α a function of x is
  + 
* In realizable setting, assume true underlying parameters ω=( ω0 , ω1,…, ωd) that satisfy p(y=1|x) = σ(ωTx)
* Goal is identify said parameters
  + Goal is to predict the probability that the class is 1 and give this probability we can infer p(y=0 | x) = 1-p(y=1| x) .
* Function learned by logistic regression returns a probability and then we take said estimate and convert it to a suitable prediction of the class
* Given unseen data point x and set of learned coefficients w, er calculate conditional probability as
  + P(y=1| x ,w) = 1/(1+ exp(-xTw))
* If P(y=1| x ,w) >=0.5 we can say that data point x should be labeled as +ve () otherwise if P(y=1| x ,w\*)<0.5 we label the data point as negative (). The predictor maps a (d+1)-dimensional vector x = (x0=1, x1🡪xd) into a zero or one.
* Even if logistic regression can perfectly model p(y=1|x) it doesn’t mean we can obtain perfect classification accuracy. Sometimes given observations are insufficient to perfectly characterize the outcome. If we get a richer observation vector with more info, target can be more certain and distribution over y is more concentrated at 1 value but never the case
* Probability values themselves can be useful in that if they are accurate the they provide a measure of confidence in classification.
* Differences in estimates allows one to determine between classifiers
* We need objective to learn parameters for logistic regressions and use maximum likelihood for this
  + Assume Dataset D={(xi,yi)}i=1n is a sample from a fixed but unknown dist p(x,y) = p(y|x)p(x)
* Objective is negative log likelihood for condition distribution scaled by number of samples which can be written as 
* Ci(w) =
* 
* Objective is usually known as cross entropy
* Then take the derivate of each component in sum for above equation using chain rule for sigmoid
  + 
* Unlike linear regression, we cant get a closed form solution to thus we use methods like gradient descent
  + Initialize w0 to a random vector and stochastic gradient descent update is 
  + Mini batch stochastic gradient descent update is 
* Logistic regression classifier is still a linear classifier despite sigmoid being non linear as P(y=1| x ,w)>= 0.5 only when xTw >=0
  + xTw =0 is a hyperplane separating -ve and +ve values
* linear classifier is a linear function splitting Rd into two half spaces
* the 2 half spaces acts as decision redions for -ve and +ve values ( if xTw >0 positive and if xTw <0 its negative)
* Goal is to find surface which best splits the points with minimal classification errors
* Similar to regression we add a component x0=1 to each input (x1🡪xd) to model intercept term as without it , it would go through the origin skewing solution
* Line must satisfy equation xTw =w0 where w0 is intercept( if you want line to go through lets say 0.t, w0 =-0.5)
* Gradient descent on cross-entropy with gradually decreasing step sizes will converge to global min
  + For non convex squared error between sigmoid and true label, it is nearly impossible to converge to global min