Regression, Least Squares and Perceptron

#	Characteristics	Linear Regression & Polynomial Regression	Least Squares	Perceptron
1.	Type of learning	Supervised	Supervised	Supervised
2.	Type of problem solved	Regression	Classification	Classification
3.	Approach to learning	NA	Discriminative	Discriminative
4.	Nature of the problem solved	Single Label and Multi-Label	Single and Multi-Label n-ary Linearly and non-linearly separable	Single Label – Binary Linearly separable
5.	Training Data – Features n – samples m- features k – classes	(n,m)	(n,m)	(n,m)
6.	Training Data – Labels n – samples m- features k – classes	Single Label – (n,) Multi Label – (n,k)	Single Label – (n,) Multi Label – (n,k)	Single Label – (n,)
7.	Feature Transformation	Transformation is possible	Transformation is possible	Transformation is possible(??)
8.	Model	$y_{hat} = w^T x$	$y_{hat} = argmax(w^T x)$	$y_{hat} = sign(w^T x)$
9.	Loss	$J(w) = \frac{1}{2}(y_{hat} - y)^T(y_{hat} - y)$	$J(w) = \frac{1}{2}tr.(y_{hat} - y)^T(y_{hat} - y)$	$J(w) = \max(-1 * y_{hat} * y, 0)$
10.	Regularisation Penalty	Ridge - + $\lambda\|\|w\|\|^2$ Lasso - + $\lambda\|\|w\|\|$	Ridge - + $\lambda\|\|w\|\|^2$ Lasso - + $\lambda\|\|w\|\|$	NA
11.	Gradient without regularisation	$\frac{\partial J(w)}{\partial w} = X^T(y_{hat} - y)$	$\frac{\partial J(w)}{\partial w} = X^T(y_{hat} - y)$	$\frac{\partial J(w)}{\partial w} = -(y_{hat} - y) * x$
12.	Regularisation Penalty	Ridge + λw	Ridge + λw	NA
13.	Normal Equation without regularisation	$w = (X^T X)\text{^}(-1)\ (X^T y)$	$w = (X^T X)\text{^}(-1)\ (X^T y)$	NA

#	Characteristics	Linear Regression & Polynomial Regression	Least Squares	Perceptron
14.	Normal Equation with ridge regularisation	$w = (X^T X + \lambda I)\text{^} - 1\ (X^T y)$	$w = (X^T X + \lambda I)\text{^}(-1)\ (X^T y)$	NA
15.	Weight Update Rule	$w_{new} = w_{old} - \alpha * Gradient$	$w_{new} = w_{old} - \alpha * Gradient$	$w_{new} = w_{old} - \alpha * Gradient$
16.	Evaluation	$RMSE = \sqrt{\left\{\frac{2}{n} * Loss\right\}}$	Accuracy, F1 score, Precision, Recall	Accuracy, F1 score, Precision, Recall
17.	Notes	Features are transformed into Polynomials if required Mini-batch Gradient – 2/batch size * Gradient Stochastic Gradient – 2* Gradient	Features are transformed into Polynomials if required Mini-batch Gradient – 2/batch size * Gradient Stochastic Gradient – 2* Gradient	It is run like a stochastic gradient descent, for each sample individually, over a number of epochs
18.	Overfit vs Underfit	Low degree – Underfitting High degree – Overfitting High λ regularization - underfitting		

Logistic and Softmax Regression & KNN

#	Characteristics	Logistic Regression	Softmax Regression	K-nearest neighbours (KNN)
1.	Type of learning	Supervised	Supervised	Supervised
2.	Type of problem solved	Classification	Classification	Regression and Classification*
3.	Approach to learning	Discriminative	Discriminative	Instance Based Discriminative
4.	Nature of the problem solved	Binary Classification	Binary Classification Multi-class and Multi label classification	Binary & Multi-class
5.	Training Data – Features n – samples m- features k – classes	Binary – (n,m)	Binary – (n,m) Multi-class & Multi-label – (n,m)	(n,m)
6.	Training Data – Labels n – samples m- features k – classes	Binary – (n,)	Binary – (n,) Multi-class & Multi-label – (n,k)	
7.	Feature Transformation	Transformation possible	Transformation possible	
8.	Model	$P(y=1\|x) = g(w^T x) = \frac{1}{1+e^{-w^T x}}$ # parameters = m+1, where m is # features $y = 1\ if\ g(z) \geq threshold$ $y = 0\ if\ g(z) < threshold$	$p(y=i\|x) = \frac{e^{w^T x_{i+b}}}{\sum e^{w^T x+b}}$	No model per se, an example is labelled by the company it keeps **Euclidean distance** $\delta(x^1, x^2) = \sqrt{(x_1^1 - x_1^2)^2 + (x_2^1 - x_2^2)^2 + \ (x_m^1 - x_m^2)^2}$ Vectorized form $\left((x^1 - x^2)^T (x^1 - x^2)\right)^{\frac{1}{2}}$ **Manhattan distance** $\delta(x^1, x^2) = \|(x_1^1 - x_1^2)\| \ + \|(x_2^1 - x_2^2)\| \ + \ \|(x_m^1 - x_m^2)\|$

#	Characteristics	Logistic Regression	Softmax Regression	K-nearest neighbours (KNN)
				Vectorized form $1_{1xm}^{T}\|x^{1}-x^{2}\|_{mx1}$ **Classification** K neighbours take part in voting. The class that receives the highest number of votes is the predicted class **Regression** Prediction is calculated as the average of the ouputs/labels of k neighbours $\hat{y}=\frac{1}{k}\sum y_i$
9.	Loss	**Binary Entropy Loss** $P(y\|x;w)=(h_w(x))^{y}\left(1-h_w(x)\right)^{1-y}$ For in independent generated training examples , likelihood is $L(w)=p(y\|X;w)$ $=\prod(h_w(x_i)^{y_i}(1-h_w(x_i)^{1-y_i}$ Written as –ve log likelihood $J(w)=-\sum y_i\log(h_w(x_i)+(1-y_i)\log(1-h_w(x_i))$ **a.k.a Binary Cross Entropy Loss**	**Categorical Entropy Loss** $\boldsymbol{J(W,b)=-\sum y_k\, log\, \widehat{y_k}}$	
10.	Regularisation Penalty	L2 reg-$\frac{\lambda}{2}\|\|w\|\|^2$ L1 reg -$\frac{\lambda}{2}\ \|\|w\|\|$	L2 reg-$\frac{\lambda}{2}\|\|w\|\|^2$ L1 reg -$\frac{\lambda}{2}\ \|\|w\|\|$	

#	Characteristics	Logistic Regression	Softmax Regression	K-nearest neighbours (KNN)
11.	Gradient without regularisation	$\frac{\partial J(w)}{\partial w} = \sum(h_w(x_i) - y_i)x_i$ In vectorised form $X^T(h_w(x) - y)$ $h_w is\ the\ sigmoid\ function$	$\frac{\partial J(W,b)}{\partial w_{ji}} = x[j]\left(\widehat{y[i]} - y[i]\right)$ $= X^T(\hat{Y} - Y)$ $\frac{\partial J(W,b)}{\partial b} = \hat{Y} - Y$	
12.	Weight Update Rule	$w_{new} = w_{old} - lr * grad$	$w_{new} = w_{old} - lr * grad$ W $b_{new} = b_{old} - lr * grad\ b$	
13.	Evaluation	Accuracy, Precision, Recall, F1 score	Accuracy, Precision, Recall, F1 score	
14.	Notes	Gradient Descent, Mini batch and Stochastic Gradient Descent methods are available	Gradient Descent, Mini batch and Stochastic Gradient Descent methods are available	
15.	Overfit vs Underfit			When K is too small, model will overfit and sensitive to noise When K is too large, the model will underfit and biased

Naïve Bayes

#	Characteristics	Naïve Bayes – Bernoulli	Naïve Bayes – Multinomial	Naïve Bayes - Gaussian
1.	Type of learning	Supervised	Supervised	Supervised
2.	Type of problem solved	Classification	Classification	Classification
3.	Approach to learning	Generative	Generative	Generative
4.	Nature of the problem solved	Single Label and Multi-Label	Single and Multi-Label n-ary Linearly and non-linearly separable	Single Label – Binary Linearly separable
5.	Training Data – Features n – samples m- features k – classes	(n,m)	(n,m)	(n,m)
6.	Training Data – Labels n – samples m- features k – classes	Single Label – (n,) Multi Label – (n,k)	Single Label – (n,) Multi Label – (n,k)	Single Label – (n,)
7.	Feature Transformation	Not possible	Not possible	Not possible
8.	Model	Naïve Bayes assumes conditional independence of features $P(y = y_c\|x) = \frac{P(y_c)\prod P(x_j\|y_c)}{\sum P(y_r)\ \prod P(x_j\|y_r)}\sum P(y_r)$ **Inference** $y = argmax_{yc}(\sum\log(P(x_j\|y_c) + \log P(y_c))$	Same as Bernoulli	Same as Bernoulli
9.	Parameter	$P(x_j\|y_c) = \mu_{jc}^{x_j}(1-\mu_{jc})^{1-x_j}$	$P(x_j\|y_c) = \frac{n!}{x_1!\,x_2!..x_m!}\prod\mu_{jc}^{x_j}$	$P(x_j\|y_c) = \frac{1}{\sqrt{2}\pi^m\|\sum\|}e^{-\frac{1}{2}\left(x_j-w_{\underline{jc}}\right)^T\sum-1\,(x_j-w_{jc})}$
10.	Loss	$J(w) = -\sum\log(P(y^{(i)}) + \sum\sum\log P(x_j^{(i)}\|y^{(i)})$ Negative Log likelihood	$J(w) = -\sum\log(P(y^{(i)}) + \sum\sum\log P(x_j^{(i)}\|y^{(i)})$	A bit complex!

#	Characteristics	Naïve Bayes – Bernoulli	Naïve Bayes – Multinomial	Naïve Bayes - Gaussian
		$J(w) = \log \sum \mathrm{w}_{\mathrm{y}}^{(i)} + \sum x_j^{(i)} \log \sum w_{jy}^{(i)} + \left(1 - x_j^{(i)}\right) \log\left(1 - w_{jy}^{(i)}\right))$	Negative Log likelihood $J(w) = \sum \log w_y^{(i)} + \sum \sum log\left(\prod w_{jyr}^{x_j}\right)$	
11.	Weights	$w_{jyr} = \frac{\sum 1(y^{(i)} = y_r) x_j^{(i)}}{\sum 1(y^{(i)} = y_r)}$	$w_{jyr} = \frac{\sum 1(y^{(i)} = y_r) x_j^{(i)}}{\sum 1(y^{(i)} = y_r) \sum x_j^{(i)}}$	$\mu_{jr} = \frac{1}{n_r} \sum 1(y^{(i)} = y_r) x_j^{(i)}$ $\sigma_{jr}^2 = \frac{1}{n_r} \sum 1(y^{(i)} = y_r) \left(x_j^{(i)} - \mu_{jr}\right)^2$
12.	Weights with Laplace smoothing	$w_{jyr} = \frac{\sum 1(y^{(i)} = y_r) x_j^{(i)} + c}{\sum 1(y^{(i)} = y_r) + 2c}$	$w_{jyr} = \frac{\sum 1(y^{(i)} = y_r) x_j^{(i)} + c}{\sum 1(y^{(i)} = y_r) \sum x_j^{(i)} + cm}$	NA
13.	Prior	$P(y = y_c) = \sum(\frac{1(y = y_c)}{n})$	Same as Bernoulli	Same as Bernoulli
14.	Evaluation	Accuracy, Precision, Recall, F1 score	Accuracy, Precision, Recall, F1 score	Accuracy, Precision, Recall, F1 score
15.	Notes			

Generalized Linear Models

#	Topic	Details
1.	Generalized Linear Model	$$p(y;\eta) = b(y)e^{(\eta^T T(y) - a(\eta))}$$ Where η is the natural / canonical parameter of distribution T(y) is the sufficient statistic a is the log partition function $e^{-a(\eta)}$ plays the role of a normalisation constant that makes sure the distribution $p(y;\eta)$ integrates over y to 1 A fixed choice of T, a, b defines a family of distributions that is parameterize by η, we then get different distributions within the family The multinomial , Poisson, Gamma and Exponential, Beta and Dirichlet distribution
2.	Bernoulli Distribution	$p(y;\phi) = \phi^y(1-\phi)^{1-y}$ $= \mathrm{e}^{(\log(\phi^y(1-\phi)^{1-y})}$ $= \mathrm{e}^{(y \log\frac{\phi}{1-\phi} + \log(1-\phi))}$ b(y) = 1 $\eta = \log\frac{\phi}{1-\phi}$ T(y) = y $a(\eta) = -\log(1-\phi)$
3.	Gaussian Distribution	To simplify derivation let $\sigma^2 = 1$ $p(y;\mu) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(y-\mu)^2}{2}}$ $= \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} e^{-\frac{-2y\mu+\mu^2}{2}}$ $= \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} e^{y\mu - \frac{\mu^2}{2}}$ $b(y) = \frac{1}{\sqrt{2\pi}} e^{(-\frac{y^2}{2})}$

#	Topic	Details
		$\eta = \mu$ $T(y) = y$ $a(\eta) = \frac{\mu^2}{2} = \frac{\eta^2}{2}$
4.	Derivation of GLM for different Models	3 assumptions a. y|x;w ~ Exponential family – b. Our goal is to predict the expected value of T(y) given x. Since in most examples T(y) = y, our prediction needs to satisfy the following equality h(x) = E[y|x] c. The natural parameter η and the inputs x are linearly related i.e $\eta = w^T x$ The function g giving the distribution mean as a function of the natural parameter $g(\eta) = E[T(y); \eta]$ is called the canonical response function Its inverse g^{-1} is called the canonical link function Canonical response function for the Gaussian family is just the identity function and the canonical response function for the Bernoulli is the logistic function
5.	Ordinary Least Squares	Assume y|x is Gaussian $y\|x; w \sim N(\mu, \sigma^2)$ $h_w(x) = E[y\|x; w] - Assumption\ 2$ $= \mu - Since\ y\|x\ is\ Gaussian$ $= \eta - Assumption\ 1$ $= w^T x - Assumption\ 3$ This shows that Ordinary least squares is a special case of the GLM family of models
6.	Logistic Regression	$h_w(x) = E[y\|x; w]$ $= \phi$ $= \frac{1}{1 + e^{-\eta}}$ $= \frac{1}{1 + e^{-w^T x}}$

Support Vector Machines

#	Characteristics	Hard Margin SVM	Soft Margin SVM	Kernel SVM
1.	Type of learning	Supervised	Supervised	Supervised
2.	Type of problem solved	Regression & Classification(*)	Regression & Classification(*)	Regression & Classification(*)
3.	Approach to learning	Discriminative	Discriminative	Discriminative
4.	Nature of the problem solved	Binary Class(*) and Multi-class Single Label and Multi Label	Binary Class(*) and Multi-class Single Label and Multi Label	Binary Class(*) and Multi-class Single Label and Multi Label
5.	Training Data – Features n – samples m- features k – classes	Feature – (n,m)	Feature – (n,m)	Feature – (n,m)
6.	Training Data – Labels n – samples m- features k – classes	Binary (*) – Single Label – (n,) Multi Label, Multi class – (n,k)	Binary (*) – Single Label – (n,) Multi Label, Multi class – (n,k)	Binary (*) – Single Label – (n,) Multi Label, Multi class – (n,k)
7.	Feature Transformation	Transformation is possible	Transformation is possible	Kernel is a shorter method to transformation
8.	Model	$y_{hat} = sign(w^T\phi(x) + b)$	$y_{hat} = sign(w^T\phi(x) + b)$	$y_{hat} = sign(\alpha_i y_i K(x_i, x) + b)$
9.	Bounding Planes	$y(w^T\phi(x) + b) = 1$	$y(w^T\phi(x) + b) = 1$	$y(w^T\phi(x) + b) = 1$
10.	Margin	$\frac{2}{\|\|w\|\|}$	$\frac{2}{\|\|w\|\|}$	$\frac{2}{\|\|w\|\|}$
11.	Primal Problem	$\min_{\{w,b\}} \frac{1}{2}\|\|w\|\|^2$ Such that $y(w^T x + b) \geq 1$	$\min_{\{w,b\}} \frac{1}{2}\|\|w\|\|^2 + C\sum \xi_i$ Such that $y(w^T x + b) \geq 1 - \xi$ $\xi_i = \max(0, 1 - y_i(w^T x_i + b))$	
12.	Hinge Loss		$C * \sum max(0, 1 - y_i(w^T x_i + b))$	
13.	Dual Lagrangian	$J(w, b, \alpha) = \min_{\{w,b\}} \frac{1}{2}\|\|w\|\|^2 - \sum \alpha_{i(y_i(w^T x_i + b) - 1)}$	Converted to an unconstrained optimisation $\min_{\{w,b\}} \frac{1}{2}\|\|w\|\|^2 + C\sum \max(0, 1 - y_i(w^T x_i + b))$	

#	Characteristics	Hard Margin SVM	Soft Margin SVM	Kernel SVM
14.	Gradients	$\frac{\partial J}{\partial w} = w - \sum \alpha_i y_i x_i = 0$ $w = \sum \alpha_i y_i x_i$ $\frac{\partial J}{\partial b} = \sum \alpha_i y_i = 0$	$\frac{\partial J}{\partial w} = w - C \sum 1(1 - y_i(w^T x_i + b) > 0) y_i x_i$ $\frac{\partial J}{\partial b} = -C \sum 1(1 - y_i(w^T x_i + b) > 0) y_i$	
15.	Final Lagrangian	$J_\alpha = \sum \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j$		
16.	KKT Condition	**Complementary slackness** $\alpha_i(y_i(w^T x_i + b) - 1) = 0$ Thus $\alpha_i > 0 \; only \; for \; SV \; as \; w^T x + b = 1$		
17.	Weight Update Rule		$w_{new} = w_{old} - lr * grad(w)$ $b_{new} = b_{old} - lr * grad(b)$	
18.	Evaluation	Accuracy, F1, Precision, Recall	Accuracy, F1, Precision, Recall	Accuracy, F1, Precision, Recall
19.	Common Kernels			**Linear** – $K(x_i, x) = x_i^T x_j$ **Polynomial** – $K(x_i, x) = \left(1 + x_i^T x_j\right)^d$ **Radial basis functions –** $K(x_i, x) = \mathrm{e}^{(-\frac{(x_i - x_j)}{\sigma^2}}$
20.	Notes	Large C – SVM becomes strict and tries to get all points on the correct side of the hyperplane Small C – SVM is lenient		

*** - Only this covered in the course**

K – Means Clustering

#	Characteristics	K-Means Clustering
1.	Type of learning	Unsupervised
2.	Type of problem solved	
3.	Approach to learning	Clustering
4.	Nature of the problem solved	Multi-class - Hard Clustering* - Soft Clustering
5.	Training Data – Features n – samples m- features k – classes	Features - (n,m)
6.	Training Data – Labels n – samples m- features k – classes	NA
7.	Feature Transformation	NA
8.	Model	Centroid $$\mu_r = \frac{1}{\lvert x^{(i)} \in c_r \rvert} \sum 1\left(x^{(i)} \in c_r\right) x^{(i)}$$ Euclidean Distance $$d = \sqrt{\sum \left(x_j^{(i)} - \mu_j\right)^2}$$
9.	Loss	$$J(c) = \sum \sum 1\left(x^{(i)} \in c_r\right) \left(\left\lvert\left\lvert x^{(i)} - \mu^r \right\rvert\right\rvert\right)^2$$
10.	Evaluation	Sum squared error
11.	Notes	Iteration run either until the centroid does change or for a specific number of iterations
12.	Overfit vs Underfit	