Machine Learning Math

|  |  |  |
| --- | --- | --- |
| **Name** | **Formula** | **Notes** |
| L2 - Euclidian Distance (Norm Distance) |  | Sqrt of difference squared |
| L1 - Max Distance |  | Sum of the absolute values |
| – Infinite Distance |  | Maximum of the absolute values (singular biggest of the set of diffs) |
|  |  | Sum of absolute values |
|  |  | Sqrt of summed Squares |
|  |  |  |
| Angle Between Vectors |  | 1. Top: Dot product of two vectors 2. Bottom: Calculate the length of each vector, or L2 norm (magnitude) & multiply the lengths 3. Calculate the arc-cosine of the resulting quotient to get the degree 4. Note: two vectors are **orthonormal** when their dot product = 0 |
| Unit Vector Definition |  | * A unit vector has a length of 1 * To get a unit vector in the same direction as 𝑥, simply divide by ‖𝑥‖ (L2 norm) |
| Convert a Vector into a Unit Vector |  | * Calculate the L2 norm of the vector * Divide each dimension of the vector by the L2 norm for the new length * Test if it’s = 1 by calculating the new L2 norm |
| Projection of a Vector |  | * If *u* is a unit vector, then the projection is simply the dot product of the two vectors. |
| Bayes Rule |  |  |
| Dot Product of a Vector |  | Multiply Matching Members and then Sum |
| Dot Product of a Matrix |  | 1. We match the 1st members (1 and 7), multiply them, likewise for the 2nd members (2 and 9) and the 3rd members (3 and 11), and finally sum them up. 2. We can do the same thing for the 2nd row and 1st column: |
| Dot Product a Matrix & a Vector |  |  |
| Determinant of a Matrix |  | 1. You can only have a determinant of a square matrix |
| Singular Matrix |  | 1. A matrix is singular IFF the determinant is 0 2. There is no inverse when it is 0 |
| Mean Squared Error (MSE) |  | Note: This is the same formula for variance of two numbers. Actual – Expected Squared (s = expected) |
| Absolute Error |  | Similar to MSE but without the square. Use absolute values to avoid negatives. |
| Parameterized Line for Single Variable |  | a = slope, b = intercept |
| Line Fitting Loss Function |  | x = predictor, y = response (the two axes you’re comparing on the scatterplot) |
| Optimal setting for ***b*** (*derivative of b*) |  | mean Y value (response) – a \* mean X Value (predictor) |
| Optimal setting for ***a*** (*derivative of a*) |  | Covariance / Variance |
| Parameterized Line with Multiple Variables | *--or--* | Note: Used to calculate the distance to a decision boundary, positive or negative = classification |
| Least Squares Regression |  |  |
| Gradient Descent with Multivariate Differentiation |  | Find the derivative with respect to each variable, then create a vector from these. Use that vector with as follows = Current estimate of (1,2,3) with step size of 0.5 for the above function. |
| Calculate the Margin of a Classifier |  | Calculated as 1 / (L2) Norm of a vector. |
| Basis Expansion using Quadratics |  | 1. All the features 2. All the squares 3. All the pair wise terms |
| Number of Dimensions of Basis Expansion |  | Us this on a *d* dimensional data set to calculate the number of dims. |
| Dot products in 2d |  | Same worked for d dimensions. Take the dot products of all the vectors, add 1 and square it. |
| Cost of k-means Clustering |  | * The sum of the squared distances between every data point and its assigned cluster center * μ = Cluster Center |
| PCA – Direction with Maximum Variance |  | Σ = the covariance matrix  must be a unit vector – meaning if it’s not, convert it  \*This is the **first eigenvector** of the covariance matrix |
| PCA – *k*-Dimensional Projection |  | The dot product of the **first eigenvector** with each data point |
| PCA – Reconstruction from a Projection |  |  |