**Linear Regression**

1. **Input expects:**

**Linear Regression can be considered a Machine Learning algorithm that allows us to map numeric input**

1. **Output:**

* **Bestfitting a line into the data points.**
* Giving the model the ability to predict outputs for inputs it has never seen before.

1. **Technical terms and equations:**

Gradient decent methods:

Hypothesis function,

Loss function, error=

Gradient:

dm=

dc=

Update Rule:

m=m-l\*dm here, l=learning rate

c=c-l\*dc

1. **strength/weakness/where to use:**

* Fast and easy to model and is particularly useful when the relationship to be modeled is not extremely complex and if you don’t have a lot of data.
* Very intuitive to understand and interpret.
* Linear Regression is very sensitive to outliers.

**Logistic Regression**

1. **Input expects:**

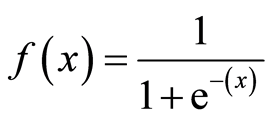
Any kind of real input

1. **output:**

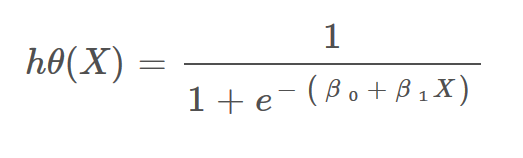
Ture or False

1. **Technical terms and equations:**

* Sigmoid function:



* Hypothesis function:



* Loss function:

Loss=

* Simplify version of loss function:

Loss = ()

* Gradients:

d

d

* Updating and values:

= - l\*d

= -l\* d

1. **Strength/weakness/where to use:**

* The output of a logistic regression is more informative than other classification algorithms. Like any regression approach
* Cannot solve non-linear problem with logistic regression because its decision making is linear
* Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables

**KNN**

1. **input expects:**

Label data set supervised

1. **output:**

Classify the required class

1. **Technical terms and Equations:**

Euledian distance : Ed=sqrt((x1-x1’)^2+(x2-x2’)^2)

Loss function : error=

Gradiant:

dm=

dc=

Update Rule:

m=m-l\*dm here, l=learning rate

c=c-l\*dc

1. **strength/weakness/where to use:**

* K-NN algorithm is very simple to understand and equally easy to implement. To classify the new data point K-NN algorithm reads through whole dataset to find out K nearest neighbors.
* K-NN does not explicitly build any model, it simply tags the new data entry based learning from historical data. New data entry would be tagged with majority class in the nearest neighbor.
* **K-NN slow algorithm**
* One of the biggest issues with K-NN is to choose the optimal number of neighbors to be consider while classifying the new data entry.
* K-NN algorithm is very sensitive to outliers as it simply chose the neighbors based on distance criteria.
* using this algorithm for face recognition

**Decision tree**

**1. Input expects:**

Realistic dataset

**2. Output:**

Creating a decision tree, which make decision on test data and say which class testing data.

**3. Technical terms and equations:**

* Gini = 1-sum(p(c)^2)
* Entropy = - sum(p(c)p(c))
* Information gain = current impurity - weighted sum of impurity

**4**. **strength/weakness/where to use:**

* A significant advantage of a decisiontree is that it forces the consideration of all possible outcomes of a decision and traces each path to a conclusion
* They can be extremely sensitive to small perturbations in the data: a slight change can result in a drastically different tree.
* They can easily overfit. This can be negated by validation methods and pruning, but this is a grey area.
* using this algorithm solve regression and classification problem