Decision Tree:

**Class Imbalance:**

* We should use up sampling or down sampling If there is a class imbalance, impacts the entropy and mse calculations

**Metrics:**

**Hyper Parameters:**

**Max Depth:** The first parameter to tune is *max\_depth*. This indicates how deep the tree can be. The deeper the tree, the more splits it has and it captures more information about the data. We fit a decision tree with depths ranging from 1 to 32 and plot the training and test auc scores.

**min\_samples\_split:** min\_samples\_split represents the minimum number of samples required to split an internal node. This can vary between considering at least one sample at each node to considering all of the samples at each node. When we increase this parameter, the tree becomes more constrained as it has to consider more samples at each node. Here we will vary the parameter from 10% to 100% of the samples

We can clearly see that when we consider 100% of the samples at each node, the model cannot learn enough about the data. This is an underfitting case.

**min\_samples\_leaf:** is The minimum number of samples required to be at a leaf node. This parameter is similar to *min\_samples\_splits*, however, this describe the minimum number of samples of samples at the leafs, the base of the tree.

**max\_features:** max\_features represents the number of features to consider when looking for the best split.

1. If int, then consider max\_features features at each split.
2. If float, then max\_features is a fraction and int(max\_features \* n\_features)features are considered at each split.
3. If “auto”, then max\_features=sqrt(n\_features).
4. If “sqrt”, then max\_features=sqrt(n\_features).
5. If “log2”, then max\_features=log2(n\_features)
6. If None, then max\_features=n\_features.

**Categorical Features with many values:**

* We should handle by applying some techniques

**Interpretability:**

**Train and Run time complexities:**

* Train Time complexities are ~O(nlogn\*d)
* n is the number of points
* d is the total number of features

**After Training Runtime and space complexities:**

* Runtime space: It is just a nested if else conditions
* Runtime time complexity: ~O(depth of tree)

**Low latency:**

* Yes. Because of Runtime complexities

**Large Data:**

* Good for large data

**Less Data:**

**Large Dimensions:**

* If number of dimension increases, time taken to train DT will increase
* Dimensionality should be less

**One hot encoding:**

* Should avoid one hot encoding if we have a very large levels (Zip Code)

**Column Standardization/Normalization:**

* Not required (As this is a distance based problem)

**Null Values:**

* It treat as a new level. Hence we have to handle the null values

**Co linearity:**

* Decision trees are by nature immune to multi-collinearity. For example, if you have 2 features which are 99% correlated, when deciding upon a split the tree will choose only one of them. Other models such as Logistic regression would use both the features.

**Multi Class classification:**

* It will be used for multi class classification

**Outliers:**

* Outliers will not impact, tree will become unstable

**Advantages:**

* Easy to interpret
* This does not consider linearity in the data. Thus we can use them in scenarios, where we know parameters are non-linearly related
* Very little effort in the data preparation
* Extremly Fast

**Disadvantages:**

* Decision-tree learners can create over-complex trees that do not generalise the data well. This is called overfitting. Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
* Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
* Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

**Assumptions:**

* Typically we don’t train decision tree with depth more than 5 or 10 (In case of massive datasets). Interpretability will be lost

**Classification and Regression:**

**Classification:**

* Entropy
* Gini
* Information Gain

**Regression:**

* MSE
* MAD





