**Week 6 : Reading Assignment**

**HW 4**

**Hyper parameter optimization:**

Parameters which define the model architecture are referred to as hyperparameters and the process of searching for the ideal model architecture is referred to as hyperparameter tuning.

This starts with us specifying a range of possible values for all the hyperparameters.

The next step after you define the range of values is to use a hyperparameter tuning method, there’s a bunch, the most common and expensive being Grid Search where others like Random Search and Bayesian Optimization will provide a “smarter”, less expensive tuning.

### **Hyperparameter tuning methods**: Hyper parameter tuning is searching of hyperparameter space for a set of values that will optimize our model architecture.

###### **Grid search**

Grid search is arguably the most basic hyperparameter tuning method. With this technique, we simply build a model for each possible combination of all of the hyperparameter values provided, evaluating each model, and selecting the architecture which produces the best results.

For example, we would define a list of values to try for both n\_estimators and max\_depth and a grid search would build a model for each possible combination.

###### **Random search**

Random search differs from grid search in that we longer provide a discrete set of values to explore for each hyperparameter; rather, we provide a statistical distribution for each hyperparameter from which values may be randomly sampled.

A Gaussian process analysis of the function from hyper-parameters to validation set performance reveals that for most data sets only a few of the hyper-parameters really matter, but that different hyper-parameters are important on different data sets. This phenomenon makes grid search a poor choice for configuring algorithms for new data sets

**Bayesian optimization**

Bayesian optimization belongs to a class of sequential model-based optimization (SMBO) algorithms that allow for one to use the results of our previous iteration to improve our sampling method of the next experiment.

**Decision tree:**

It is popular because of its robustness to noise and tolerance against missing information, handling of irrelevant, redundant predictive attribute values, low computational cost, interpretability, fast run time and robust predictors

**DecisionTreeClassifier**

criterion: string, optional (default=”gini”):

Decision tree nodes are split by using impurity. Impurity is a measure of the homogeneity of the labels on a node. Two ways to implement impurity measure is implemented by scikit learn using Information gain and Gini Impurity or Gini Index. Information gain uses the entropy measure as the impurity measure and splits a node such that it gives the most amount of information gain. Whereas Gini Impurity measures the divergences between the probability distributions of the target attribute’s values and splits a node such that it gives the least amount of impurity.

Gini: Gini(E) = 1-

Entropy : H(E) = -

**“No Free Lunch”** theorem  suggests that the choice of splitting criteria will not make much difference in the tree performance. Each criterion is superior in some cases and inferior in others.

**splitter: string, optional (default=”best”)**

Supported strategies are “best” to choose the best split and “random” to choose the best random split.

The **“best” splitter** it evaluate all splits using the criterion before splitting. When we have a hundreds of features, then “best” splitter would be ideal because it will calculate the best features to split based on the impurity measure and use that to split the nodes, whereas if you choose “random” you have a high chance of ending up with features that don’t really give you that much information, which would lead to a more deeper less precise tree.

**The “random” splitter** uses a random uniform function with min\_feature\_value, max\_feature\_value and random\_state as inputs. It selects a set of features randomly and splits, it doesn’t have the computational overhead of computing the optimal split. Next, it is also less prone to overfitting because you are not essentially calculating the best split before each split and the additional randomness will help you here, so if your model is overfitting, then you can change the splitter to “random” and retrain.

For a tree with few features without any overfitting, the “best” splitter to be safe so that you get the best possible model architecture.max\_depth: int or None, optional (default=None).

A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

The weighted impurity decrease equation is the following:

N\_t/N\*(impurity – N\_t\_R / N\_t \* right\_impurity – N\_t\_L / N\_T \* left\_impurity

where N is the total number of samples, N\_t is the number of samples at the current node, N\_t\_L is the number of samples in the left child, and N\_t\_R is the number of samples in the right child. N, N\_t, N\_t\_R and N\_t\_L all refer to the weighted sum, if sample\_weight is passed.

Interpreting a decision tree should be fairly easy if you have the domain knowledge on the dataset you are working with because a leaf node will have 0 gini index because it is pure, meaning all the samples belong to one class. Then you can look into the splits that lead to 0 gini index and see if it makes sense to classify your classes as such or whether you can reduce the depth thereby leading to a more generalizable tree, if so, you can increase the min\_impurity\_decrease to prevent further division because now, the node will not be further split if the impurity doesn’t decrease by the amount you specified.

Decision tree complexity crucially affects accuracy and is explicitly controlled by stopping stopping criteria used and pruning method employed. Usually, the tree complexity is measured by one of the following metrics: the total number of nodes, total number of leaves, tree depth and number of attributes used [8]. max\_depth, min\_samples\_split, and min\_samples\_leaf are all stopping criteria whereas min\_weight\_fraction\_leaf and min\_impurity\_decrease are pruning methods.

Regression analysis mathematically describes the relationship between a set of independent variables and a dependent variable.

Linear Regression

Linear regression, also known as ordinary least squares (OLS) and linear least squares, is the real workhorse of the regression world. Use linear regression to understand the mean change in a dependent variable given a one-unit change in each independent variable.

Types of linear regression

1. **Ridge regression** allows you to analyze data even when severe multicollinearity is present and helps prevent overfitting. This type of model reduces the large, problematic variance that multicollinearity causes by introducing a slight bias in the estimates.
2. **Lasso regression** (least absolute shrinkage and selection operator) performs variable selection that aims to increase prediction accuracy by identifying a simpler model. It is similar to Ridge regression but with variable selection.
3. **Partial least squares (PLS) regression**is useful when you have very few observations compared to the number of independent variables or when your independent variables are highly correlated. PLS decreases the independent variables down to a smaller number of uncorrelated components, similar to Principal Components Analysis. Then, the procedure performs linear regression on these components rather the original data. PLS emphasizes developing predictive models and is not used for screening variables. Unlike OLS, you can include multiple continuous dependent variables. PLS uses the correlation structure to identify smaller effects and model multivariate patterns in the dependent variables.

### **Nonlinear regression**

Nonlinear regression also requires a continuous dependent variable, but it provides a greater flexibility to fit curves than linear regression.

### **Regression with categorical dependent variables**

### **Binary Logistic Regression:** Use binary logistic regression to understand how changes in the independent variables are associated with changes in the probability of an event occurring. This type of model requires a binary dependent variable. A binary variable has only two possible values, such as pass and fail.

### **Ordinal Logistic Regression:** Ordinal logistic regression models the relationship between a set of predictors and an ordinal response variable. An ordinal response has at least three groups which have a natural order, such as hot, medium, and cold.

### **Nominal Logistic Regression:** Nominal logistic regression models the relationship between a set of independent variables and a nominal dependent variable. A nominal variable has at least three groups which do not have a natural order, such as scratch, dent, and tear.

## **Regression Analysis with Count Dependent Variables**

### **Poisson regression:** Count data frequently follow the Poisson distribution, which makes Poisson Regression a good possibility. Poisson variables are a count of something over a constant amount of time, area, or another consistent length of observation. With a Poisson variable, you can calculate and assess a rate of occurrence.