WEEK- 7: Hyper parameter tuning for DecisionTreeClassifier.

Session No.7

Hyper parameter tuning for DecisionTreeClassifier

- The models can have many hyperparameters.
- Parameters like in decision criterion, max_depth, min_sample_split, etc. are called hyperparameters.
- We can find the best combination of the parameter using grid search methods.
- Grid search is a technique for tuning hyperparameter that may facilitate to build a model andevaluate a model for every combination of algorithms parameters per grid.
- In the Grid Search, all the mixtures of hyperparameters combinations will pass through one byone into the model and check the score on each model.
- It gives us the set of hyperparameters which gives the best score.
- Grid Search takes the model or objects you'd prefer to train and different values of the hyperparameters.
- It then calculates the error for various hyperparameter values, permitting you to choose the bestvalues.

Implementation of Hyper parameter tuning using GridSearchCV

```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
from sklearn.datasets import load_breast_cancer
cancer_data = load_breast_cancer()
```

```
cancer_df = pd.DataFrame(cancer_data.data, columns=cancer_data.feature_names)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(cancer_df,
                                                    cancer_data.target,
                                                    test_size = 0.33,
                                                    random_state = 42)
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
```

```
# Create Decision Tree classifer object
dtc = DecisionTreeClassifier()

# Train Decision Tree Classifer
dtc.fit(X_train,y_train)
```

```
DecisionTreeClassifier()
```

```
param_dict = {
"criterion":['gini','entropy'],
    "max_depth":range(1,10),
    "min_samples_split":range(1,10),
    "min_samples_leaf":range(1,5)
}
```

```
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(dtc,
                    param_grid=param_dict,
                    cv=10,
                    verbose=1,
                    n_jobs=-1)
grid.fit(X_train,y_train)
```

```
Fitting 10 folds for each of 648 candidates, totalling 6480 fits

GridSearchCV(cv=10, estimator=DecisionTreeClassifier(), n_jobs=-1,
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': range(1, 10),
                         'min_samples_leaf': range(1, 5),
                         'min_samples_split': range(1, 10)},
             verbose=1)
```

```
grid.best_params_
```

```
{'criterion': 'entropy',
 'max_depth': 3,
 'min_samples_leaf': 4,
 'min_samples_split': 4}
```

```
grid.best_estimator_
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=3, min_samples_leaf=4,
                       min_samples_split=4)
```

```
grid.best_score_
```

```
0.9474358974358974
```