**Week 3: Model selection, training, hyper-parameter optimization and model performance evaluation**

1. Import the necessary libraries.
2. Run the Jupyter notebook from week 1, week 2 as a module and use its code in the new Jupyter notebook.

**Part A: Model Selection**:

**1) Train-Valid-Test Split:**

* Training data is the set of the data on which the actual training takes place.
* Validation split helps to improve the model performance by fine-tuning the model after each epoch.
* The test set informs us about the final accuracy of the model after completing the training phase.
* The train test validation split is a technique for partitioning data into training, validation, and test sets.
* Here we have train and test dataset. So, we decide to split the train dataset into train and validation in the ratio of 84:16.
* Then we compute the shape and size of the train, test, and validation dataset.

**2) Cross Validation (CV):**

* The cross-validation method is used to test trained machine-learning models and to evaluate their performance independently.

**i) K-Fold CV:**

* The training data used in the model is split into k number of smaller sets for validating the model.
* The model is then trained on k-1 folds of training set. The remaining fold is then used as a validation set to evaluate the model.
* In this K-fold method, we use estimator as decision tree classifier and we split K-fold into n=10.
* Then, we compute the cross validation score using cross\_val\_score from sklearn.model\_selection package along with average cross validation score and the number of cross validations used.

**3) Bootstrapping:**

* Bootstrapping is a resampling technique that helps in estimating the uncertainty of a statistical model.
* It includes sampling the original dataset with replacement and generating multiple new datasets of the same size as the original.

**Part B: Model Training**:

* Model training is the process wherein we feed an ML algorithm with the data to identify and learn proper values of attributes.
* Here we use supervised learning classifier since out dataset is labeled. The training data used for the problem statement contains both the input and output values.

**1) Decision Tree Classifier:**

* It is supervised learning technique which is used for both classification and regression problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome.
* A decision tree asks a question, and based on the answer either Yes or No, it is split into subtrees.
* Decision Tree Classifier is used because it is easy to understand as it mimics the human thinking ability at the time of making decisions.

**2) Random Forest Classifier:**

* Random forest is a supervised learning technique used for both the classification and regression.
* It combines many classifiers to solve a problem and improve the model performance. (Ensemble Learning)
* It contains number of decision trees on various subsets of the given dataset and take the average to improve the predictive accuracy.
* We use random forest since it takes less training time compared to other algorithms, predicts higher accuracy for smaller and larger datasets.

**3) XGBoost Classifier:**

* It is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library which provides parallel tree boosting.
* It is designed for speed, ease of use, and performance on large datasets.
* It makes it easy to scale up on multicore machines or clusters.
* It also uses cache awareness, which helps reduce memory usage when training models with large datasets.

**4) Gradient Boosting Classifier:**

* Gradient Boosting is a powerful boosting algorithm that combines several weak learners into strong learners, in which each new model is trained to minimize the loss function such as mean squared error or cross-entropy of the previous model using gradient descent.
* Gradient boosting trees can be more accurate than random forests. Because we train them to correct each other’s errors, they’re capable of capturing complex patterns in the data.

**5) Logistic Regression:**

* Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class.
* It can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

**6) Comparing all the classifier models using Voting Classifier:**

* A Voting Classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output.
* Voting classifier is used because instead of creating separate dedicated models and finding the accuracy for each of them, we create a single model which trains and predicts output based on their combined majority of voting for each output class.
* Various performance metrics that are used for every model are as follows:

1] Confusion Matrix: It is a tabular representation of prediction outcomes of any binary classifier, which is used to describe the performance of the classification model on a set of test data when true values are known.

* The table is divided into 4 categories:

1. True Positive (TP): In this case, the prediction outcome is true, and it is also true.
2. True Negative (TN): in this case, the prediction outcome is false, and it is also false.
3. False Positive (FP): In this case, prediction outcomes are true, but they are false.
4. False Negative (FN): In this case, predictions are false, and they are true.

2] Accuracy: It can be determined as the number of correct predictions to the total number of predictions. It is good to use the Accuracy metric when the target variable classes in data are approximately balanced.

3] Precision: It determines the proportion of positive prediction that was correct.

4] Recall or Sensitivity: It aims to calculate the proportion of actual positive that was identified incorrectly.

5] F1 Score: It is a metric to evaluate a binary classification model based on predictions that are made for the positive class.

* If we want to minimize the false negative, then, Recall should be as near to 100%, and if we want to minimize the false positive, then precision should be close to 100% as possible. If we maximize precision, it will minimize the FP errors, and if we maximize recall, it will minimize the FN error.

The results obtained for the training accuracy for decision tree classifier and xgboost classifier are the highest and for the logistic regression is the lowest.

Other parameters like testing accuracy, precision, recall and f1 score are higher for decision tree, xgboost and gradient boosting classifiers and for the logistic regression, they are the lowest.

**Part C: Hyper-parameter Optimization or Tuning**:

* Hyperparameter optimization is the process of finding the right combination of hyperparameter values to achieve maximum performance on the data in a reasonable amount of time.

**1) GridSearchCV:**

* GridSearchCV is the process of performing hyperparameter tuning to determine the optimal values for a given model.
* It tries all the combinations of the values passed in the dictionary and evaluates the model for each combination using the Cross-Validation method.

After performing GridSearch CV on the models used above, the algorithms used here like Decision Tree, Random Forest, XGBoost, Gradient Boosting are having the highest accuracy, precision, recall, f1 scores. Logistic Regression showed the lowest performance.

**2) RandomizedSearchCV:**

* An alternative way to perform hyperparameter optimization is to perform random sampling on the grid and perform k-fold cross-validation on some randomly selected cells.
* Random Search sets up a grid of hyperparameter values and selects random combinations to train the model and score.

Similar results are obtained after performing Randomized search CV on different classifiers like Decision Tree, Random Forest, XGBoost, Gradient Boosting. Logistic Regression showed the lowest performance.

**Part D: Model Evaluation**:

* After performing Hyperparameter Tuning, we perform evaluation of different models.

# **Export some of the trained ML models**

* To serve predictions from AI Platform Prediction, you must export your trained machine learning model.
* Joblib is a Python library for running computationally intensive tasks in parallel.
* It is useful because it allows you to save the state of your model and resume your work later or on a different machine.
* We use joblib because it automatically corrects errors when reading or writing files, making it more dependable than manual pickling.
* In this case we have saved 2 classifier models: random forest and decision tree and then used it to get the prediction of the target variable “class\_tranform”.
* Models are saved as model\_dt.ml and model\_rf.ml.
* We then load the saved models for predictions and use model.predict([[“List of Input variables to be passed”]]) to give the prediction of the target variables.
* Also, a main.py file is created to run the prediction on the localhost.
* In the main.py file, flask application is created and basically POST query is used to perform predictions.
* We create a predict() function and inside it the saved machine learning model is loaded. Then model.predict([[“List of Input variables to be passed”]]) is called and saved in class\_prediction variable. We then return str(class\_prediction).
* Once the python code is run in the command prompt, using the URL [127.0.0.1:5000/predict](http://127.0.0.1:5000/predict) in the browser will give us the predicted value.