20101133

Touseef Ahmed

Assignment # 3

What do you know about hyper-parameters used in Classifiers in the Sklearn library in python?

In the scikit-learn library, hyperparameters are parameters that are not learned from the data. They must be set by the user in order to configure the model. Some examples of hyperparameters in sklearn classifiers include:

- The regularization parameter in logistic regression (C)
- The kernel and gamma parameters in support vector machines (SVMs)
- The maximum depth of a decision tree

Tuning these hyperparameters can have a big impact on the performance of the model, and finding the optimal values for your data is often a key step in the machine learning process. One way to do this is through grid search, which involves training a model with a range of different hyperparameter values and evaluating their performance using cross-validation.

1. C: This is the regularization parameter for linear models such as Logistic Regression and Support Vector Machines. It controls the strength of the regularization, with higher values resulting in stronger regularization.
2. kernel: This parameter is used in Support Vector Machines and specifies the kernel function to use. Options include linear, polynomial, and radial basis function (RBF).
3. gamma: This parameter is also used in Support Vector Machines and specifies the width of the RBF kernel. Higher values result in a wider kernel, which can lead to overfitting.
4. max_depth: This parameter is used in decision tree classifiers such as DecisionTreeClassifier and specifies the maximum depth of the tree. Higher values can result in overfitting.
5. n_estimators: This parameter is used in ensemble classifiers such as RandomForestClassifier and specifies the number of trees in the ensemble. Higher values can improve model performance but also increase computational time.
6. learning_rate: This parameter is used in gradient boosting classifiers such as XGBoost and specifies the learning rate of the model. Lower values can improve model performance but also increase training time.

Why do you think CNNs are successful for image classification?

There are several reasons why Convolutional Neural Networks (CNNs) are successful for image classification:

1. Spatial invariance: CNNs use convolutional layers which apply a fixed-size filter to small parts of an image. This allows the model to recognize features regardless of their position within the image, which is important for image classification.

2. Feature extraction: CNNs automatically extract features from the input image through the convolutional and pooling layers. These features are more relevant for image classification than raw pixel values.
3. Regularization: CNNs often use techniques such as dropout and weight decay to regularize the model, which helps to prevent overfitting.
4. Scalability: CNNs can be trained on large datasets and can handle images of different sizes and resolutions.
5. Transfer learning: Pre-trained CNN models can be used as a starting point for image classification tasks, which can save time and resources.