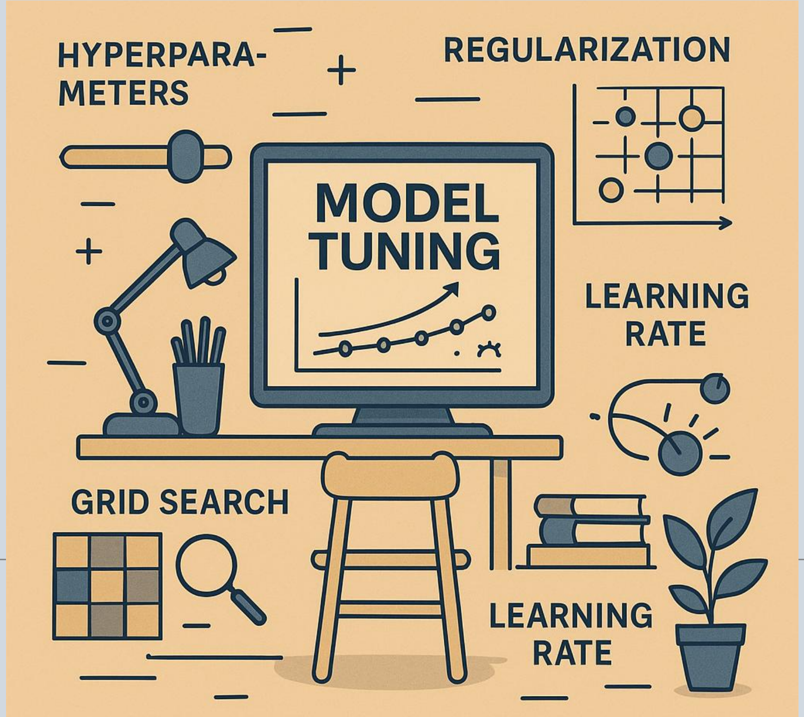
# Model Tuning




DR. ZVI BEN AMI

# Agenda

- Model Tuning
- Hyperparameters vs. parameters
- Common Hyperparameters
- Grid Search

# Why Model Tuning?

Optimizing models for better performance and accuracy is crucial in the field of data science for several reasons:

- Improved Predictive Power
- Enhanced Generalization
- Efficient Resource Utilization
- Cost Reduction
- Competitive Advantage
- Interpretability and Explainability

Model tuning involves finding the best set of hyperparameters for a given algorithm to achieve optimal results.

# Hyperparameters vs. Parameters

- Parameters are internal variables that are learned by the model during the training process.
  - Updated iteratively by an optimization algorithm, such as gradient descent, to minimize the difference between predicted and actual outputs.
  - Specific to the trained model and are adjusted based on the available data.
- Hyperparameters, on the other hand, are external variables that are set before the training process begins.
  - Determine the behavior and characteristics of the model and influence how the parameters are learned.
  - Not learned from the data but are predefined by the data scientist or model developer.

# Common Hyperparameters

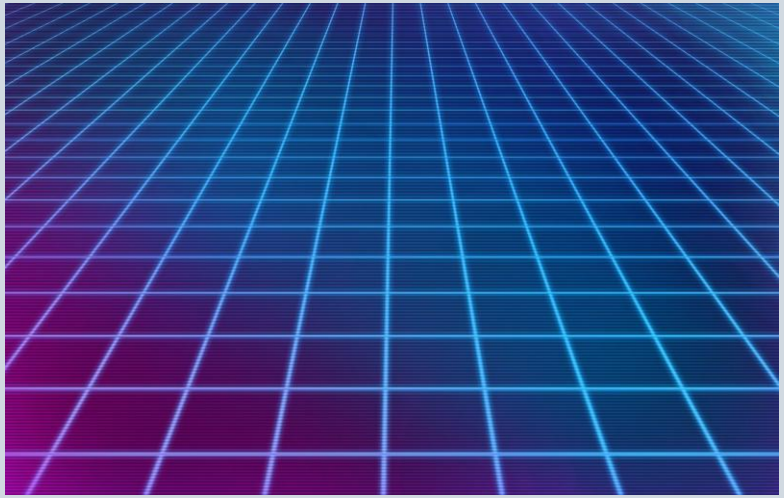
- Regularization Parameter:
  - Used to control the amount of regularization applied to prevent overfitting.
  - Common regularization techniques include L1 regularization (Lasso) and L2 regularization (Ridge).
  - The regularization parameter balances the trade-off between fitting the training data well and avoiding complex models that may overfit.
- Learning Rate:
  - Determines the step size at each iteration during the optimization process.
  - Affects how quickly the model converges to the optimal solution.
  - Too high of a learning rate may lead to overshooting the optimal solution, while too low of a learning rate may result in slow convergence.

# Common Hyperparameters

- Batch Size
  - Determines the number of training samples used in each iteration of the optimization algorithm.
- Number of Estimators
  - For ensemble models
- Kernel Parameters
  - for SVM
- Dropout Rate
  - For Neural Networks
- Different models may have additional specific hyperparameters that can be tuned

# Grid Search

- Grid search is a popular technique for hyperparameter tuning in machine learning.
- It involves systematically searching through a predefined set of hyperparameter values to identify the combination that produces the best model performance.
- Grid search is called so because it forms a grid-like structure with all possible combinations of hyperparameter values.



# Grid Search Process

1. **Define the Hyperparameter Grid:**
   - Identify the hyperparameters that need to be tuned for the given model.
   - Define a set of possible values or ranges for each hyperparameter.
   - These values can be discrete, such as a list of options, or continuous, such as a range of values.
2. **Create a Scoring Metric:**
   - Specify a scoring metric to evaluate the performance of different hyperparameter combinations.
   - The choice of metric depends on the problem at hand, such as accuracy, precision, recall, F1 score, or mean squared error.
   - The scoring metric provides an objective measure of how well the model performs with each hyperparameter combination.
3. **Perform Grid Search:**
   - Exhaustively iterate over all possible combinations of hyperparameters in the defined grid.
   - Train and evaluate a model for each combination using the chosen scoring metric.
   - This process typically involves cross-validation, where the data is split into training and validation sets, and the model is evaluated on multiple folds.
4. **Select the Best Hyperparameters:**
   - Identify the hyperparameter combination that yields the highest performance based on the scoring metric.
   - This combination represents the optimal set of hyperparameters for the given model and dataset.

# Best Practices For Model Tuning

- Start with a broad search space and refine it gradually.
- Prioritize tuning the most influential hyperparameters.
- Use appropriate evaluation metrics for the problem domain.
- Consider the computational cost of grid search and explore efficient alternatives.

# Model Tuning Example

model_tuning.ipynb

```
param_grid = {'n_estimators': [100, 200, 300],
              'max_depth': [None, 5, 10],
              'min_samples_split': [2, 5, 10],
              'min_samples_leaf': [1, 2, 4],
              'max_features': ['sqrt', 'log2']}
```

```
# Create a Random Forest classifier
rf = RandomForestClassifier()

# Perform grid search with cross-validation
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5)
grid_search.fit(X_train, y_train)

# Print the best hyperparameters and score
print("Best Hyperparameters: ", grid_search.best_params_)
print("Best Score: ", grid_search.best_score_)
```

```
Best Hyperparameters: {'max_depth': None, 'max_features': 'sqrt',
'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
```

# THANK YOU FOR LISTENING

ZVI.BENAMI@MAIL.HUJI.AC.IL