1.What are the three stages to build the hypotheses or model in machine learning?

1. Model building
2. Model testing
3. Applying the model

2. What is the standard approach to supervised learning?

Supervised algorithms can further be divided into following:

a. Classification: When the data is being used to predict a category, supervised learning is also called classification. This is the case when assigning an image as a picture of either a 'cat' or a 'dog'. When there are only two choices, it's called two-class or binomial classification. When there are more categories, like predicting the winner of the NCAA. March Madness tournament, this problem is known as multi-class classification.

b. Regression: When a value is being predicted, as with stock prices, supervised learning is

called regression.

c. Anomaly detection: Sometimes the goal is to identify data points that are simply unusual.

In fraud detection, for example, any highly unusual credit card spending patterns is

considered to be a suspect. The possible variations are so numerous and the training

examples so few, that it's not feasible to learn what fraudulent activity looks like. The

approach that anomaly detection takes is to simply learn what normal activity looks like

(using a history of non-fraudulent transactions) and identify anything that is significantly

different.

3.What is Training set and Test set?

What is a Training Set?

In Machine Learning, a training set is a dataset used to train a model. In training the model, specific

features are picked out from the training set. These features are then incorporated into the model.

Thereby, if the training set is labeled correctly, the model should be able to learn something from

these features.

What is a Test Set?

The test set is a dataset used to measure how well the model performs at making predictions on

that test set. If the prediction scores for the test set are unreasonable, we’ll have to make some

adjustments to our model and try again.

4. What is the general principle of an ensemble method and what is bagging and

boosting in ensemble method?

The general principle of an ensemble method is to combine the predictions of several models built with a given learning algorithm in order to improve robustness over a single model.

Bagging: Several estimators are built independently on subsets of the data and their predictions are averaged. Typically, the combined estimator is usually better than any of the single base estimator. Bagging is a method in ensemble for improving unstable estimation or classification schemes. Bagging can reduce variance with little to no effect on bias. Eg: Random Forest

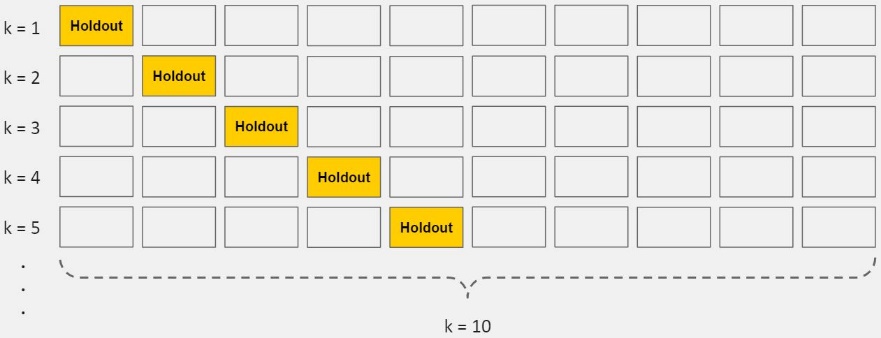
Boosting: Base estimators are built sequentially. Each subsequent estimator focuses on the weaknesses of the previous estimators. In essence several weak models "team up" to produce a powerful

ensemble model. Boosting can reduce bias without incurring higher variance.

ex: Gradient Boosted Trees, AdaBoost

5.How can you avoid overfitting:

**Cross-validation:** Cross-validation is a powerful preventative measure against overfitting. Use your initial training data to generate multiple mini train-test splits. Use these splits to tune your model.In standard k-fold cross-validation, we partition the data into k subsets, called folds. Then, we iteratively train the algorithm on k-1 folds while using the remaining fold as the test set (called the “holdout fold”).



*K-Fold Cross-Validation*

Cross-validation allows you to tune hyperparameters with only your original training set. This allows you to keep your test set as a truly unseen dataset for selecting your final model.

**Train with more data**: It won’t work every time, but training with more data can help algorithms detect the signal better. In the earlier example of modeling height vs. age in children, it’s clear how sampling more schools will help your model.

Of course, that’s not always the case. If we just add more noisy data, this technique won’t help.

**Remove features:** Some algorithms have built-in feature selection. For those that don’t, you can manually improve their generalizability by removing irrelevant input features.

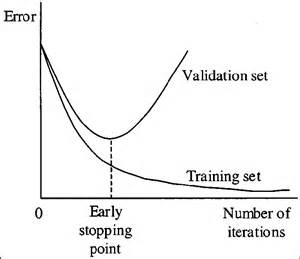
An interesting way to do so is to tell a story about how each feature fits into the model. This is like the data scientist's spin on software engineer’s [rubber duck debugging](https://en.wikipedia.org/wiki/Rubber_duck_debugging) technique, where they debug their code by explaining it, line-by-line, to a rubber duck.If anything doesn't make sense, or if it’s hard to justify certain features, this is a good way to identify them.  
In addition, there are several [feature selection heuristics](https://elitedatascience.com/dimensionality-reduction-algorithms#feature-selection) you can use for a good starting point.

**Early stopping**

When you’re [training a learning algorithm iteratively](https://elitedatascience.com/machine-learning-iteration#model), you can measure how well each iteration of the model performs.

Up until a certain number of iterations, new iterations improve the model. After that point, however, the model’s ability to generalize can weaken as it begins to overfit the training data.

Early stopping refers stopping the training process before the learner passes that point.



Today, this technique is mostly used in deep learning while other techniques (e.g. regularization) are preferred for classical machine learning.

**Regularization:** Regularization refers to a broad range of techniques for artificially forcing your model to be simpler. The method will depend on the type of learner you’re using. For example, you could prune a decision tree, use dropout on a neural network, or add a penalty parameter to the cost function in regression. Oftentimes, the regularization method is a hyperparameter as well, which means it can be tuned through cross-validation.