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

There are three different stages to build the model in machine learning which are listed as below:

a)      Model building

b)      Model testing

c)       Applying the model

**Model building:**

Once the data is in good shape, we can start building the actual model. It involves a lot of experimentation and discovery -selecting the most relevant features, testing multiple algorithms etc. It’s not always a straightforward execution task, and therefore the timeline of getting to a production-ready model can be very unpredictable. There are cases where the first algorithm tested gives great results, and cases where nothing you try works well.

**Model testing:**

At this stage, we have to ensure final model is as good as it can be. They’ll assess model performance based on the predefined quality metrics, compare the performance of various algorithms they tried, tune any parameters that affect model performance and eventually test the performance of the final model. In the case of supervised learning they’ll need to determine whether the predictions of the model when compared to the ground truth data are good enough for your purposes.

In the case of unsupervised learning, there are various techniques to assess performance, depending on the problem. In the case of clustering for example, you may be able to easily plot the objects you cluster across multiple dimensions, or even consume objects that are a form of media to see if the clustering seems intuitively reasonable.

Iterate: At this point we need to decide whether further iterations are necessary or not. How does the model perform vs. expectations? Does it perform well enough to constitute a significant improvement over the current state of your business? Are there areas where it is particularly weak? Is a greater number of data points required? Can you think of additional features that will improve performance? Are there alternative data sources that would improve the quality of inputs to the model?

**Applying Model:**

At this stage, the finalized model can be launched in production. Note that you need to figure out which dimensions you want to scale your model on first if you’re not ready to commit to full productization. Say your product is a movie recommendation tool: You may want to only open access to a handful of users but provide a complete experience for each user, in which case your model needs to rank every movie in your database by relevance to each of the users. That’s a different set of scaling requirements than say providing recommendations only for action movies, but opening up access to all users.

1. What is the standard approach to supervised learning?

Supervised learning enables the model to predict future outcomes after they are trained based on past data. To train the model, first, a set of inputs and outputs are fed to it. Say for example, you want your model to be able to recognize a car from given data. Training is done by providing a set of inputs and outputs that help the system understand what the essential features are, that define a car. These features may include a bonnet, a headlight, a steering wheel and so on. Supervised machine learning is also used to detect which emails can be marked as spam and not spam.

Supervised learning problems can be grouped into regression and classification problems.

**Classification**: A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.

**Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.Some popular examples of supervised machine learning algorithms are:

1. Linear regression for regression problems.
2. Random forest for classification and regression problems.
3. Support vector machines for classification problems.

In order to solve a given problem of supervised learning, one has to perform the following steps:

1. Determine the type of training examples. Before doing anything else, the user should decide what kind of data is to be used as a training set. In case of handwriting analysis, for example, this might be a single handwritten character, an entire handwritten word, or an entire line of handwriting.
2. Gather a training set. The training set needs to be representative of the real-world use of the function. Thus, a set of input objects is gathered and corresponding outputs are also gathered, either from human experts or from measurements.
3. Determine the input feature representation of the learned function. The accuracy of the learned function depends strongly on how the input object is represented. Typically, the input object is transformed into a [feature vector](https://en.wikipedia.org/wiki/Feature_vector), which contains a number of features that are descriptive of the object. The number of features should not be too large, because of the [curse of dimensionality](https://en.wikipedia.org/wiki/Curse_of_dimensionality); but should contain enough information to accurately predict the output.
4. Determine the structure of the learned function and corresponding learning algorithm. For example, the engineer may choose to use [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machine) or [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning).
5. Complete the design. Run the learning algorithm on the gathered training set. Some supervised learning algorithms require the user to determine certain control parameters. These parameters may be adjusted by optimizing performance on a subset (called a validation set) of the training set, or via [cross-validation](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).
6. Evaluate the accuracy of the learned function. After parameter adjustment and learning, the performance of the resulting function should be measured on a test set that is separate from the training set.

3.What is Training set and Test set?

**Training set:** It 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.  In sentiment analysis, using n-grams as features, i.e. single words or sequences of 2 or 3 consecutive words in tweets as features.  Thereby, if the training set is labelled correctly, the model should be able to learn something from these features, i.e. the degree to which each feature affects the sentiment of a sentence.

**Test set:** The test set is a dataset used to measure how well the model performs at making predictions on that test set.  In the case of sentiment analysis, a test set is a dataset of tweets that are distinct from the tweets in the training set. If the prediction scores (sentiment scores) for the test set are unreasonable, we’ll need to make some adjustments to our model and try again.

The test would yield misleading results if we test our model with the training data.  The model itself was created by learning from the training set, so it will likely do quite well at making predictions on the training set itself- it knows this data too well.  We need to test the model with a test set, i.e. a dataset the model hasn’t seen before.

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. Ensemble learning is used when you build component classifiers that are more accurate and independent from each other.

(a) Sequential ensemble methods

(b) Parallel ensemble methods

**Bagging:** This method in ensemble for improving unstable estimation or classification schemes. Bagging both can reduce errors by reducing the variance term.

**Boosting:** This method used sequentially to reduce the bias of the combined model. Boosting can reduce errors by reducing the variance term.

5.How can you avoid overfitting ?

Following are the commonly used methodologies:

1. **Cross-Validation**: Cross Validation in its simplest form is a one round validation, where we leave one sample as in-time validation and rest for training the model. But for keeping lower variance a higher fold cross validation is preferred.
2. **Early Stopping**: Early stopping rules provide guidance as to how many iterations can be run before the learner begins to over-fit.
3. **Pruning**: Pruning is used extensively while building CART models. It simply removes the nodes which add little predictive power for the problem in hand.
4. **Regularization**: This is the technique we are going to discuss in more details. Simply put, it introduces a cost term for bringing in more features with the objective function. Hence, it tries to push the coefficients for many variables to zero and hence reduce cost term.