# Machine Learning Techniques

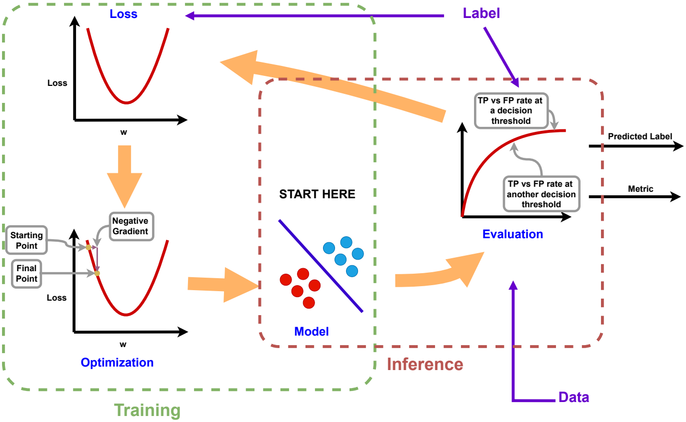
## ML Component Framework

Dr. Ashish Tendulkar

IIT Madras

1. ML Component Framework
2. Training Data
3. Model
4. Loss Function
5. Optimization
6. Evaluation
7. Summary

# Machine Learning Process



# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# ML Component Framework

- Training data
- Model
- Loss function
- Optimization procedure
- Evaluation criteria

# Training Data

- No data, no ML.
- In supervised learning, training data consists of input and output pairs.
- Each input is represented by a bunch of numbers called features or attributes.
- Apply certain transformations to convert input into a bunch of numbers.

# Training Data

- No data, no ML.
- In supervised learning, training data consists of input and output pairs.
- Each input is represented by a bunch of numbers called features or attributes.
- Apply certain transformations to convert input into a bunch of numbers.

# Training Data

- No data, no ML.
- In supervised learning, training data consists of input and output pairs.
- Each input is represented by a bunch of numbers called features or attributes.
- Apply certain transformations to convert input into a bunch of numbers.

# Training Data

- No data, no ML.
- In supervised learning, training data consists of input and output pairs.
- Each input is represented by a bunch of numbers called features or attributes.
- Apply certain transformations to convert input into a bunch of numbers.

# Training Data

- No data, no ML.
- In supervised learning, training data consists of input and output pairs.
- Each input is represented by a bunch of numbers called features or attributes.
- Apply certain transformations to convert input into a bunch of numbers.

# Training Data

For example,

- Input may be an image or a piece of text.
- Certain pre-processing steps or transformations have to be applied to obtain feature representation for such input data.

# Training Data

For example,

- Input may be an image or a piece of text.
- Certain pre-processing steps or transformations have to be applied to obtain feature representation for such input data.

# Training Data

For example,

- Input may be an image or a piece of text.
- Certain pre-processing steps or transformations have to be applied to obtain feature representation for such input data.

# Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
  - The expert would tell us which features are most important in determining the price.
  - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

# Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
    - The expert would tell us which features are most important in determining the price.
    - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

# Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
  - The expert would tell us which features are most important in determining the price.
  - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

## Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
  - The expert would tell us which features are most important in determining the price.
  - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

## Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
  - The expert would tell us which features are most important in determining the price.
  - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

# Training Data

- **Features**: are provided by the domain experts.
- For example, while predicting price of a house:
  - The expert would tell us which features are most important in determining the price.
  - The expert would also provide us with a dataset of houses with their features and prices (which is what we are interested in predicting.)
- **Output**: a real number or a discrete value from a predefined set.

# Training Data

Example of a real valued label:

- The housing price prediction problem.
- Output label is price, a real number.

# Training Data

Example of a real valued label:

- The housing price prediction problem.
- Output label is price, a real number.

# Training Data

Example of a real valued label:

- The housing price prediction problem.
- Output label is price, a real number.

# Training Data

Examples of a discrete valued label:

- The handwritten digit prediction problem. 10 different values as class labels - digit 0 to 9.
- Loan sanctioning. Binary labels - yes or no corresponding to application being sanctioned or rejected.

## Training Data

Examples of a discrete valued label:

- The handwritten digit prediction problem. 10 different values as class labels - digit 0 to 9.
- Loan sanctioning. Binary labels - yes or no corresponding to application being sanctioned or rejected.

# Training Data

Examples of a discrete valued label:

- The handwritten digit prediction problem. 10 different values as class labels - digit 0 to 9.
- Loan sanctioning. Binary labels - yes or no corresponding to application being sanctioned or rejected.

# Training Data

- Once we have obtained training data, we get an idea about **input** and **output**, which helps us in defining suitable ML problems and choosing appropriate components like **model**, **loss** and **optimization procedure** for training.

1. ML Component Framework
2. Training Data
3. **Model**
4. Loss Function
5. Optimization
6. Evaluation
7. Summary

# Model

- Model provides a mathematical form of mapping between input and output.
- The input is represented with a bunch of features and output can be a real number or a discrete value from some finite set.

# Model

- Model provides a mathematical form of mapping between input and output.
- The input is represented with a bunch of features and output can be a real number or a discrete value from some finite set.

# Model

- Model provides a mathematical form of mapping between input and output.
- The input is represented with a bunch of features and output can be a real number or a discrete value from some finite set.

# Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

## Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

## Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

# Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

# Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

## Model

Example of a linear model :

- $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- The key problem here is to estimate values of weights.
- All weights together form an entity called weight vector.
- We estimate the weight vector by training the model on the training data.
- The ideal weights are the ones that when used in the model, produce output that is close to the actual output for all training data points.

# Model

- Depending on the nature of the output, we choose our model.
- When the **output is a real number**, we choose models of regression - which are capable of producing a real valued output.
- When the **output is a discrete value**, we choose models of classification - which produce a discrete quantity as an output.

# Model

- Depending on the nature of the output, we choose our model.
- When the **output is a real number**, we choose models of regression - which are capable of producing a real valued output.
- When the **output is a discrete value**, we choose models of classification - which produce a discrete quantity as an output.

# Model

- Depending on the nature of the output, we choose our model.
- When the **output is a real number**, we choose models of regression - which are capable of producing a real valued output.
- When the **output is a discrete value**, we choose models of classification - which produce a discrete quantity as an output.

# Loss Function

- The key objective of training a model is to estimate the weight vector and we need a principled way of doing that.
- We need a suitable method for measuring the difference between predicted and actual output.
- Loss function provides that measure.
- Loss function is a function of weight vector - as we change the weight vector, we obtain a new model, which will have different a loss.

# Loss Function

- The key objective of training a model is to estimate the weight vector and we need a principled way of doing that.
- We need a suitable method for measuring the difference between predicted and actual output.
- Loss function provides that measure.
- Loss function is a function of weight vector - as we change the weight vector, we obtain a new model, which will have different a loss.

# Loss Function

- The key objective of training a model is to estimate the weight vector and we need a principled way of doing that.
- We need a suitable method for measuring the difference between predicted and actual output.
- Loss function provides that measure.
- Loss function is a function of weight vector - as we change the weight vector, we obtain a new model, which will have different a loss.

# Loss Function

- The key objective of training a model is to estimate the weight vector and we need a principled way of doing that.
- We need a suitable method for measuring the difference between predicted and actual output.
- Loss function provides that measure.
- Loss function is a function of weight vector - as we change the weight vector, we obtain a new model, which will have different a loss.

# Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W) =$ Difference between actual and predicted output for all training samples.
- In the following example model: $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

# Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W)$ = Difference between actual and predicted output for all training samples.
- In the following example model:
$Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

# Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W) =$ Difference between actual and predicted output for all training samples.
- In the following example model:
  $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

# Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W) =$ Difference between actual and predicted output for all training samples.
- In the following example model:
  $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

# Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W) =$ Difference between actual and predicted output for all training samples.
- In the following example model:
  $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

## Loss Function

- Denote loss with letter $J$.
- $J : W \longrightarrow R$
- $J(W) =$ Difference between actual and predicted output for all training samples.
- In the following example model:
  $Output = weight_0 + weight_1 \cdot feature_1 + weight_2 \cdot feature_2 + \cdots + weight_m \cdot feature_m$
- Everything except weight vector is fixed on the right side of the equation- features are specified as part of the training data, which is fixed and weights are variables.
- The weights can be changed to obtain a different result or output.

# Loss Function

- An example of loss function is squared loss function: It is calculated as a sum of square of differences between the actual and predicted values.

$$J(W) = \sum_{i=1}^{n} [predicted^{(i)} - actual^{(i)}]^2$$

- Above equation can be simplified to:
$J(W) = \sum_{i=1}^{n} [\{weight_0 + weight_1 \cdot {feature_1}^{(i)} + weight_2 \cdot {feature_2}^{(i)} + \cdots + weight_m \cdot {feature_m}^{(i)}\} - actual^{(i)}]^2$

# Loss Function

- An example of loss function is squared loss function: It is calculated as a sum of square of differences between the actual and predicted values.

$$J(W) = \sum_{i=1}^{n}[predicted^{(i)} - actual^{(i)}]^2$$

- Above equation can be simplified to: $J(W) = \sum_{i=1}^{n}[\{weight_0 + weight_1 \cdot feature_1{}^{(i)} + weight_2 \cdot feature_2{}^{(i)} + \cdots + weight_m \cdot feature_m{}^{(i)}\} - actual^{(i)}]^2$

## Loss Function

- An example of loss function is squared loss function: It is calculated as a sum of square of differences between the actual and predicted values.

$$J(W) = \sum_{i=1}^{n} [predicted^{(i)} - actual^{(i)}]^2$$

- Above equation can be simplified to:
$J(W) = \sum_{i=1}^{n} [\{weight_0 + weight_1 \cdot feature_1{}^{(i)} + weight_2 \cdot feature_2{}^{(i)} + \cdots + weight_m \cdot feature_m{}^{(i)}\} - actual^{(i)}]^2$

# Loss Function

- Our job is to find the weight vector that results in the lowest loss as per the defined loss function.
- How to estimate such a weight vector?
- Brute force search for the optimal weight vector, but that is not at all efficient.
- What is the best way to estimate it?
- Well that's what is the purpose of our next component: optimization procedure.

## Loss Function

- Our job is to find the weight vector that results in the lowest loss as per the defined loss function.
- How to estimate such a weight vector?
- Brute force search for the optimal weight vector, but that is not at all efficient.
- What is the best way to estimate it?
- Well that's what is the purpose of our next component: optimization procedure.

# Loss Function

- Our job is to find the weight vector that results in the lowest loss as per the defined loss function.
- How to estimate such a weight vector?
- Brute force search for the optimal weight vector, but that is not at all efficient.
- What is the best way to estimate it?
- Well that's what is the purpose of our next component: optimization procedure.

# Loss Function

- Our job is to find the weight vector that results in the lowest loss as per the defined loss function.
- How to estimate such a weight vector?
- Brute force search for the optimal weight vector, but that is not at all efficient.
- What is the best way to estimate it?
- Well that's what is the purpose of our next component: optimization procedure.

# Loss Function

- Our job is to find the weight vector that results in the lowest loss as per the defined loss function.
- How to estimate such a weight vector?
- Brute force search for the optimal weight vector, but that is not at all efficient.
- What is the best way to estimate it?
- Well that's what is the purpose of our next component: optimization procedure.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

## Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Finding Weight Vector

- Obtain a weight vector that minimizes the loss function.
- Formally,

$$W = argmin_W J(W)$$

- Applications of derivatives are typically in $12^{th}$ calculus in the context of such problems.
- Take a derivative of loss function w.r.t. the weight vector and set it to 0.

$$\frac{d}{dW} J(W) = 0$$

- Solve this equation directly or with analytical methods to obtain the optimal weight vector.
- The optimization procedure is a cornerstone of model training.

# Evaluation

After optimization:

- How to evaluate the model on unseen data?
- What metric to use?
- The evaluation metric changes as per the problem - different metric for regression and classification problems.

# Evaluation

After optimization:

- How to evaluate the model on unseen data?
- What metric to use?
- The evaluation metric changes as per the problem - different metric for regression and classification problems.

## Evaluation

After optimization:

- How to evaluate the model on unseen data?
- What metric to use?
- The evaluation metric changes as per the problem - different metric for regression and classification problems.

# Evaluation

After optimization:

- How to evaluate the model on unseen data?
- What metric to use?
- The evaluation metric changes as per the problem - different metric for regression and classification problems.

# Evaluation

- What if the model does not perform as per the expectation as found via evaluation metric?
- Reiterate the ML pipeline.
- Try to add superior
  - features
  - models
  - loss function
  - optimization procedure

# Evaluation

- What if the model does not perform as per the expectation as found via evaluation metric?
- Reiterate the ML pipeline.
- Try to add superior
  - features
  - models
  - loss function
  - optimization procedure

# Evaluation

- What if the model does not perform as per the expectation as found via evaluation metric?
- Reiterate the ML pipeline.
- Try to add superior
  - features
  - models
  - loss function
  - optimization procedure

# Summary

## Machine Learning Process